GE VERNOVA

PROFICY®SOFTWARE & SERVICES

PROFICY iFIX HMI/SCADA

iFIX Automation Reference

Proprietary Notice

Trademark Notices

We want to hear from you. If you have any comments, questions, or suggestions about our documentation, send them to the following email address:
doc@ge.com

Table of Contents

iFIX Automation Reference 1

Object Summary 2

A 2

B 2

C 2

D 2

E 2

F 3

G-K 3

L-N 3

O 3

P-Q 3

R 4

S 4

T-U 4

V 4

W-Z 4

A-C 5

Alarm Summary Object 5

Application Object 5

Arc Object 6

Bitmap Object 6

Chart Object 6

Chord Object 6

ColorButton Object 6

ControlContainer Object 6

D-F 7

DataItem Object 7

DataItems Object 7

DataLink Object 7
DataServer Object 7
DataServers Object 7
Document Object 8
Documents Object 8
Dynamo Object 8
DynamoSet Object 8
ESignature Object 8
Event Object 9
ExpressionEditor Object 9
FindReplace Object 10
FixDataSystem Object 10
FixFloatPoint Object 10
FixGeometryHelper Object 10
FixKeyMacroCollection Object 10
FixKeyMacro Object 11
Format Object 11
G-O 11
GeneralDataSet Object 11
Group Object 11
Group (DataSystem) Object 11
Groups Object 11
HistDatalink Object 12
HistogramChart Object 12
Legend Object 12
Line Object 12
Linear Object 12
LineChart Object 12
LineConnector Object 13
Lines Object 13
Lookup Object 13

Oval Object 13

P-S 13

Pen Object 13

Picture Object 14

Pie Object 14

Pipe Object 14

PipeConnector Object 14

Polygon Object 14

Polyline Object 14

Procedures Object 14

RealTimeSPCDataSet Object 15

Rectangle Object 15

RightAngleLineConnector Object 15

RoundRectangle Object 15

Scheduler Object 15

ScriptLine Object 15

ScriptProcedure Object 16

ScriptSource Object 16

SecuritySynchronizer Object 16

Sources Object 16

SPCBarChart Object 16

System Object 16

T-Z 17

Tag Group Object 17

Text Object 17

TimeAxis Object 17

Timer Object 17

ToolbarManager Object 17

UserGlobals Object 17

UserPreferences Object 18

ValueAxis Object 18

Variable Object 18

Window Object 18

XYChart Object 19

Property Summary 20

A 20

B 21

C 22

D 23

E 24

F 25

G 26

H 27

I-K 28

L 28

M 30

N 31

O 31

P 32

Q 33

R 33

S 34

T 36

U 38

V 38

W-Y 39

Z 39

A 40

Active Property 40

Syntax 40

Properties 40

Settings 40

Remarks 40

ActiveDocument Property 40

Syntax 40

Properties 40

Remarks 40

ActiveWindow Property 41

Syntax 41

Properties 41

Remarks 41

AdvancedGraphics Property 41

Syntax 41

Properties 41

Settings 41

Remarks 42

AlarmHornEnabled Property 42

Syntax 42

Properties 42

Return Value 42

AlarmRefreshInterval Property 42

Syntax 42

Properties 42

AlarmUserdefField1 Property 43

Syntax 43

Properties 43

AlarmUserdefField2 Property 43

Syntax 43

Properties 43

Alignment Property 43

Syntax 43

Properties 44

Settings 44

Remarks 44
AllowsDrillDown Property 44
Syntax 44
Properties 44
Settings 44
Remarks 45
AllowTimeAxisReset Property 45
Syntax 45
Properties 45
Settings 45
AllowValueAxisReset Property 45
Syntax 45
Properties 45
Settings 46
AlwaysOnTop Property 46
Syntax 46
Properties 46
Settings 46
Remarks 46
AnalogError Property 46
Syntax 46
Properties 47
Remarks 47
AnalogErrorTag Property 47
Syntax 47
Properties 47
Remarks 47
AngleUnits Property 47
Syntax 47
Properties 47
Settings 48

Remarks 48

Application Property 48

Syntax 48

Properties 48

Remarks 48

ApplyProperties Property 48

Syntax 48

Properties 48

Settings 49

AssignedID Property 49

Syntax 49

Properties 49

Remarks 49

Author Property 49

Syntax 49

Properties 49

Remarks 49

Autofetch Property 49

Syntax 50

Properties 50

Settings 50

Remarks 50

AutoMinMaxPaddingX Property 50

Syntax 50

Properties 50

AutoMinMaxPaddingY Property 50

Syntax 51

Properties 51

AutoSize Property 51

Syntax 51

Properties 51

Settings 51

Remarks 51

AutoUpdateRate Property 52

Syntax 52

Properties 52

Remarks 52

AverageDataValue Property 52

Syntax 52

Properties 53

Remarks 53

AxisColor Property 53

Syntax 53

Properties 53

AxisLength Property 53

Syntax 53

Properties 53

Remarks 53

AxisTitle Property 53

Syntax 54

Properties 54

B 54

BackDropBackgroundColor Property 54

Syntax 54

Properties 54

Remarks 54

BackDropBackgroundStyle Property 54

Syntax 55

Properties 55

Settings 55

BackDropBlend Property 55

Syntax 55

Properties 55

Remarks 55

BackDropBorderColor Property 55

Syntax 55

Properties 56

BackDropBorderStyle Property 56

Syntax 56

Properties 56

Settings 56

BackDropColor Property 56

Syntax 56

Properties 57

Remarks 57

BackdropFadeColor Property 57

Syntax 57

Properties 57

BackdropFadeType Property 57

Syntax 57

Properties 57

Settings 58

Remarks 58

BackDropGradAngle Property 58

Syntax 58

Properties 58

Remarks 58

BackDropStyle Property 58

Syntax 58

Properties 58

Settings 59

BackDropVisible Property 59

Syntax 59

Properties 59

Settings 59

BackgroundColor Property 59

Syntax 60

Properties 60

BackgroundEdgeColor Property 60

Syntax 60

Properties 60

BackgroundEdgeStyle Property 60

Syntax 60

Properties 60

Settings 60

BackgroundEdgeWidth Property 61

Syntax 61

Properties 61

BackgroundStyle Property 61

Syntax 61

Properties 61

Settings 61

BackupSecPath Property 62

Syntax 62

Properties 62

Remarks 62

BarVal Property 62

Syntax 62

Properties 62

BasePath Property 62

Syntax 62

Properties 63

Remarks 63

BestFitWithCenter Property 63

Syntax 63

Properties 63

Settings 63

Remarks 63

BitmapGradientMode Property 63

Syntax 64

Properties 64

Settings 64

Blend Property 64

Syntax 64

Properties 64

Remarks 64

BlinkEnabled Property 64

Syntax 64

Properties 65

Settings 65

BlinkRate Property 65

Syntax 65

Properties 65

Remarks 65

BorderTypes Property 65

Syntax 65

Properties 65

Bottom Property 66

Syntax 66

Properties 66

Remarks 66

BottomCenter Property 66

Syntax 66

Properties 66

Remarks 66

BottomLeft Property 66

Syntax 67

Properties 67

Remarks 67

BottomRight Property 67

Syntax 67

Properties 67

Remarks 67

BottomVisibleRow Property 67

Syntax 67

Properties 67

Remarks 68

BoundRect Property 68

Syntax 68

Properties 68

Remarks 68

ButtonState Property 68

Syntax 68

Properties 68

Settings 68

Remarks 68

ButtonStyle Property 69

Syntax 69

Properties 69

Settings 69

Remarks 69

C 69

CacheEnabled Property 69

Syntax 69

Properties 69

Settings 70

- Remarks 70
- Cancel Property 70
  - Syntax 70
  - Properties 70
  - Settings 70
  - Remarks 71
- Caption Property 71
  - Syntax 71
  - Properties 71
  - Remarks 71
- Category Property 71
  - Syntax 71
  - Properties 71
  - Remarks 71
- Center Property 72
  - Syntax 72
  - Properties 72
  - Remarks 72
- CenterOfRotation Property 72
  - Syntax 72
  - Properties 72
  - Remarks 72
- CenterPoint Property 72
  - Syntax 73
  - Properties 73
  - Remarks 73
- CenterX Property 73
  - Syntax 73
  - Properties 73
- CenterY Property 73
  - Syntax 73

Properties 73

CharactersPerLine Property 74

Syntax 74

Properties 74

ChartFontSize Property 74

Syntax 74

Properties 74

CheckForAlarmListChanged Property 74

Syntax 74

Properties 75

Settings 75

Remarks 75

CheckForNewAlarms Property 75

Syntax 75

Properties 75

Settings 75

Remarks 75

CheckForSeverityIncrease Property 75

Syntax 76

Properties 76

Settings 76

Remarks 76

ClassName Property 76

Syntax 76

Properties 76

Remarks 76

Color Property 77

Syntax 77

Properties 77

ColorTable Property 77

Syntax 77

Properties 77
Settings 77
Remarks 77
CombinationKey Property 77
Syntax 78
Properties 78
Comments Property 78
Syntax 78
Properties 78
CompletionStatus Property 78
Syntax 78
Properties 78
CompletionStatusTag Property 79
Syntax 79
Properties 79
Remarks 79
ConfirmDataEntry Property 79
Syntax 79
Properties 79
Settings 79
ConnectionFailed Property 80
Syntax 80
Properties 80
Remarks 80
Return Values 80
ConstantLine Property 80
Syntax 80
Properties 80
Settings 81
Remarks 81
ContainedObjects Property 81

Syntax 81

Properties 81

Remarks 81

ContainedSelections Property 81

Syntax 81

Properties 81

Remarks 81

ContextID Property 82

Syntax 82

Properties 82

ControlOrderIndex Property 82

Syntax 82

Properties 82

Count Property 82

Syntax 82

Properties 82

Remarks 83

CurrentDataSet Property 83

Syntax 83

Properties 83

Remarks 83

CurrentDataSource Property 83

Syntax 83

Properties 83

Remarks 83

CurrentDate Property 83

Syntax 84

Properties 84

CurrentDateDay Property 84

Syntax 84

Properties 84

CurrentDateMonth Property 84

Syntax 84

Properties 84

CurrentDateYear Property 84

Syntax 85

Properties 85

CurrentImage Property 85

Syntax 85

Properties 85

CurrentPen Property 85

Syntax 85

Properties 85

CurrentPicture Property 85

Syntax 86

Properties 86

CurrentTime Property 86

Syntax 86

Properties 86

CurrentTimeHour Property 86

Syntax 86

Properties 86

CurrentTimeMinute Property 87

Syntax 87

Properties 87

CurrentTimeSecond Property 87

Syntax 87

Properties 87

CurrentValue Property 87

Syntax 87

Properties 87

D 88

DataEntry Property 88

Syntax 88

Properties 88

Settings 88

DataItems Property 88

Syntax 88

Properties 88

Remarks 88

DataRefreshInterval Property 88

Syntax 89

Properties 89

DataServers Property 89

Syntax 89

Properties 89

Remarks 89

DataSetColor Property 89

Syntax 89

Properties 89

DataShadows Property 90

Syntax 90

Properties 90

DaylightSavingsTime Property 90

Syntax 90

Properties 90

Settings 90

Remarks 91

DaysBeforeNow Property 91

Syntax 91

Properties 91

Remarks 91

DaysOfMonth Property 91

Syntax 91

Properties 91

Remarks 92

DaysOfWeek Property 92

Syntax 92

Properties 92

Remarks 92

Deadband Property 92

Syntax 92

Properties 92

DecimalDigits Property 92

Syntax 92

Properties 93

Default Property 93

Syntax 93

Properties 93

Settings 93

Remarks 93

DefaultDataSystem Property 93

Syntax 94

Properties 94

Remarks 94

DefaultExternalDatasourceUpdateRate Property 94

Syntax 94

Properties 94

Remarks 94

DefaultOutputValue Property 94

Syntax 94

Properties 94

DefaultServer Property 95

Syntax 95

- Properties 95
- Return Values 95
- Remarks 95

Description Property 95

- Syntax 95
- Properties 95

DeskColor Property 96

- Syntax 96
- Properties 96

DigitalError Property 96

- Syntax 96
- Properties 96
- Remarks 96

DigitalErrorTag Property 96

- Syntax 96
- Properties 97
- Remarks 97

DigitsOfPrecision Property 97

- Syntax 97
- Properties 97

DisableAutoScale Property 97

- Syntax 97
- Properties 97
- Settings 98

DisplayLayer Property 98

- Syntax 98
- Properties 98
- Remarks 98

DisplayMilliseconds Property 98

- Syntax 98
- Properties 98

Settings 99

DisplayShelvedAlarms Property 99

Syntax 99

Properties 99

Settings 99

This property used to filter the shelved alarms in the Alarm Summary object. Syntax object.DisplayShelvedAlarms [=Boolean] PropertiesThe DisplayShelvedAlarms property syntax has these parts: PartDescriptionobjectAn object expression that evaluates to the Alarm Summary object in the Applies To list.BooleanWhether the shelved alarms are displayed in the Alarm Summary object.Settings The settings for DisplayShelvedAlarms are: ConstantDescriptionTrue When set to True, only shelved alarms are displayed in the Alarm Summary object. (Default) False When set to False, alarms that are not shelved are displayed in the Alarm Summary object. Example 99

DisplayStatusBar Property 100

Syntax 100

Properties 100

Settings 100

DisplayString Property 100

Syntax 101

Properties 101

DisplaySystemTree Property 101

Syntax 101

Properties 101

Settings 101

DocumentHeight Property 101

Syntax 101

Properties 101

Remarks 102

DocumentHeightEx Property 102

Syntax 102

Properties 102

Remarks 102

DocumentPath Property 102

Syntax 102

Properties 102

Remarks 102

Documents Property 102

Syntax 103

Properties 103

Remarks 103

DocumentWidth Property 103

Syntax 103

Properties 103

Remarks 103

DocumentWidthEx Property 103

Syntax 103

Properties 103

Remarks 104

Domain Property 104

Syntax 104

Properties 104

Remarks 104

DownImageDisplayed Property 104

Syntax 104

Properties 104

Settings 105

DSDescription Property 105

Syntax 105

Properties 105

DSLegendAvgerageOverRangeColWidth Property 105

Syntax 105

Properties 105

DSLegendCurrentValColWidth Property 105

Syntax 105

Properties 106

DSLegendDescriptionColWidth Property 106
Syntax 106
Properties 106
DSLegendEngUnitsColWidth Property 106
Syntax 106
Properties 106
DSLegendHighLimitColWidth Property 106
Syntax 107
Properties 107
DSLegendHighOverRangeColWidth Property 107
Syntax 107
Properties 107
DSLegendLowLimitColWidth Property 107
Syntax 107
Properties 107
DSLegendLowOverRangeColWidth Property 107
Syntax 108
Properties 108
DSLegendMask Property 108
Syntax 108
Properties 108
Remarks 108
DSLegendQualityColWidth Property 109
Syntax 109
Properties 109
DSLegendSourceColWidth Property 109
Syntax 109
Properties 109
DSPosition Property 109
Syntax 109
Properties 109

Remarks 110

Duration Property 110

Syntax 110

Properties 110

Dynamo_Description Property 110

Syntax 110

Properties 110

Return Value 110

Dynamo_ID Property 110

Syntax 111

Properties 111

Return Value 111

E 111

EdgeColor Property 111

Syntax 111

Properties 111

EdgeStyle Property 111

Syntax 111

Properties 111

Settings 112

Bitmap Object Syntax 112

Properties 112

Object Settings 112

Remarks 112

EdgeWidth Property 112

Syntax 113

Properties 113

EditText Property 113

Syntax 113

Properties 113

ElbowStyle Property 113

Syntax 113
Properties 113
Settings 113
EnableAcknowledgeAll Property 114
Syntax 114
Properties 114
Settings 114
Remarks 114
EnableAlarmAcknowledge Property 114
Syntax 114
Properties 114
Settings 114
EnableAlarmDeletion Property 115
Syntax 115
Properties 115
Settings 115
Remarks 115
EnableAsVbaControl Property 115
Syntax 115
Properties 115
Remarks 116
EnableColumnQuickSort Property 116
Syntax 116
Properties 116
Settings 116
Remarks 116
Enabled Property 116
Syntax 116
Properties 116
Return Values 117
Remarks 117

EnableGlobalEndTime Property 117

Syntax 117

Properties 117

Settings 117

Remarks 118

EnableGlobalScrollPercentage Property 118

Syntax 118

Properties 118

Settings 118

Remarks 118

EnableEndTime Property 118

Syntax 118

Properties 119

Settings 119

EnableRightMouseClick Property 119

Syntax 119

Properties 119

Settings 119

EnableRunTimeConfiguration Property 119

Syntax 120

Properties 120

Settings 120

EnableTooltips Property 120

Syntax 120

Properties 120

Settings 120

Remarks 120

EndAngle Property 120

Syntax 121

Properties 121

Remarks 121

EndCap Property 121

Syntax 121

Properties 121

Settings 121

EndPoint Property 122

Syntax 122

Properties 122

Remarks 122

EndTime Property 122

Chart and Pen Syntax 122

Properties 122

Remarks 123

Timer Syntax 123

Properties 123

Remarks 123

EndX Property 123

Syntax 123

Properties 123

EndY Property 123

Syntax 124

Properties 124

EngUnits Property 124

Syntax 124

Properties 124

EnhancedCoordinates Property 124

Syntax 124

Properties 124

Settings 125

Remarks 125

ErrorMode Property 125

Syntax 125

Properties 125

Format and Lookup Object Settings 125

Linear Object Settings 125

Remarks 126

EventParameter Property 126

EventType Property 126

Syntax 126

Properties 126

Settings 126

ExactMatch Property 127

Syntax 127

Properties 127

Settings 127

Expandable Property 127

Syntax 127

Properties 127

Settings 128

ExtendMaxSpace Property 128

Syntax 128

Properties 128

ExtendType Property 128

Syntax 128

Properties 128

Settings 129

F 129

FadeColor Property 129

Syntax 129

Properties 129

FadeType Property 129

Syntax 129

Properties 129

Settings 129
Remarks 130
FailedSource Property 130
Syntax 130
Properties 130
Remarks 130
FetchDataSetLimits Property 130
Syntax 130
Properties 130
Settings 131
FetchPenLimits Property 131
Syntax 131
Properties 131
Settings 131
Remarks 131
FileName Property 131
Syntax 131
Properties 131
Remarks 132
FillStyle Property 132
Syntax 132
Properties 132
Settings 132
Remarks 132
FilterString Property 132
Syntax 133
Properties 133
FixedDate Property 133
Syntax 133
Example 133
Properties 133

- Remarks 133
- FixedTime Property 133
  - Syntax 133
  - Example 134
  - Properties 134
  - Remarks 134
- FixPath Property 134
  - Syntax 134
  - Properties 134
  - Settings 134
  - Remarks 135
- Font Property 135
  - Syntax 135
  - Properties 135
- FontName Property 135
  - Syntax 135
  - Properties 135
- FontSize Property 136
  - Syntax 136
  - Properties 136
- FontStyle Property 136
  - Syntax 136
  - Properties 136
  - Settings 136
- ForceVerticalPoints Property 136
  - Syntax 137
  - Properties 137
- ForegroundColor Property 137
  - Syntax 137
  - Properties 137
- ForegroundEdgeColor Property 137

Syntax 137

Properties 137

ForegroundEdgeStyle Property 138

Syntax 138

Properties 138

Settings 138

ForegroundEdgeWidth Property 138

Syntax 138

Properties 138

Format Property 139

Syntax 139

Properties 139

FormatDataType Property 139

Syntax 139

Properties 139

Settings 139

FullName Property 139

Syntax 140

Properties 140

Remarks 140

FullScreen Property 140

Syntax 140

Properties 140

Settings 140

FullyQualifiedName Property 140

Syntax 140

Properties 141

Remarks 141

G-J 141

GlobalDuration Property 141

Syntax 141

Properties 141
GlobalEndTime Property 141
Syntax 141
Properties 142
Remarks 142
GlobalFastScrollOption Property 142
Syntax 142
Properties 142
Remarks 142
GlobalHistoricalUpdateRate Property 142
Syntax 143
Properties 143
GlobalMovingEndTime Property 143
Syntax 143
Properties 143
Returns 143
Remarks 143
GlobalMovingStartTime Property 143
Syntax 143
Properties 143
Returns 144
Remarks 144
GlobalOutputToggle Property 144
Syntax 144
Properties 144
Settings 144
Remarks 144
GlobalPlayBack Property 144
Syntax 145
Properties 145
Settings 145

GlobalPlayBackFrameSize Property 145

Syntax 145

Properties 145

GlobalPlayBackNumberOfFrames Property 145

Syntax 146

Properties 146

GlobalPlayBackSpeed Property 146

Syntax 146

Properties 146

GlobalSlowScrollOption Property 146

Syntax 147

Properties 147

Remarks 147

GlobalSlowScrollRate Property 147

Syntax 147

Properties 147

Remarks 147

GlobalStartTime Property 147

Syntax 148

Properties 148

GlobalTimerPause Property 148

Syntax 148

Properties 148

Settings 148

GlobalTimeSync Property 149

Syntax 149

Properties 149

Settings 149

GlobalToggle Property 149

Syntax 149

Properties 149

Remarks 150

Gradient Property 150

Syntax 150

Properties 150

GlobalFastScrollRate Property 150

Syntax 150

Properties 150

Remarks 150

GradientAngle Property 151

Syntax 151

Properties 151

Remarks 151

GraphBackColor Property 151

Syntax 151

Properties 151

GraphForeColor Property 151

Syntax 151

Properties 152

GraphPlusTable Property 152

Syntax 152

Properties 152

GraphPlusTableMenu Property 152

Syntax 152

Properties 152

GridEnabled Property 153

Syntax 153

Properties 153

Settings 153

GridInFront Property 153

Syntax 153

Properties 153

Settings 153

GridInterval Property 154

Syntax 154

Properties 154

GridLinesToShow Property 154

Syntax 154

Properties 154

GridStyle Property 154

Syntax 155

Properties 155

GridWidth Property 155

Syntax 155

Properties 155

Groups Property 155

Syntax 155

Properties 156

Remarks 156

Height Property 156

Syntax 156

Properties 156

Remarks 156

HelpFile Property 156

Syntax 156

Properties 156

HelpPath Property 156

Syntax 157

Properties 157

Remarks 157

HideMathFunctionsButton Property 157

Syntax 157

Properties 157

Settings 157
HiDisplay Property 157
TimeAxis Syntax 157
Properties 157
ValueAxis Syntax 158
Properties 158
HighestDataValue Property 158
Syntax 158
Properties 158
HighlightEnabled Property 158
Syntax 158
Properties 158
Settings 159
Remarks 159
HighlightedDatasource Property 159
Syntax 159
Properties 159
HiInValue Property 159
Syntax 159
Properties 160
HiLimit Property 160
Syntax 160
Properties 160
HiOutValue Property 160
Syntax 160
Properties 160
HistMode Property 160
Syntax 161
Properties 161
HistoricalSampleType Property 161
Syntax 161

- Properties 161
- Settings 161

HistUpdateRate Property 162

- Syntax 162
- Properties 162
- Remarks 162

HorizontalFillDirection Property 162

- Syntax 162
- Properties 162
- Settings 162

HorizontalFillPercentage Property 162

- Syntax 163
- Properties 163

HorizontalGridColor Property 163

- Syntax 163
- Properties 163

HorizontalGridStyle Property 163

- Syntax 163
- Properties 163
- Settings 163

HorizontalPosition Property 164

- Syntax 164
- Properties 164
- Remarks 164

HorizontalScaleDirection Property 164

- Syntax 164
- Properties 164
- Settings 165

HorizontalScalePercentage Property 165

- Syntax 165
- Properties 165

Remarks 165

ImageCount Property 165

Syntax 166

Properties 166

Remarks 166

IncludeDataLabels Property 166

Syntax 166

Properties 166

Settings 166

Index Property 166

Syntax 166

Properties 166

Remarks 167

InitialValue Property 167

Syntax 167

Properties 167

InputValue Property 167

Syntax 167

Properties 167

Remarks 167

Interval Property 168

Chart and Pen Syntax 168

Properties 168

Timer and Event Syntax 168

Properties 168

Remarks 168

IntervalMilliseconds Property 168

Syntax 168

Properties 168

Remarks 169

IsDirty Property 169

Syntax 169

Properties 169

Return Values 169

Remarks 169

IsInterpolated Property 169

Syntax 169

Properties 169

Settings 170

IsModifiable Property 170

Syntax 170

Properties 170

Settings 170

IsSelectable Property 170

Syntax 171

Properties 171

Settings 171

IsSelected Property 171

Syntax 171

Properties 171

Return Values 171

Remarks 171

Item Property 171

Syntax 172

Properties 172

Remarks 172

Justification Property 172

Syntax 172

Properties 172

Settings 172

K-L 172

KeyCode Property 172

- Syntax 173
- Properties 173

LabelBold Property 173

- Syntax 173
- Properties 173
- Settings 173

LabelColor Property 173

- Syntax 173
- Properties 174

LabelFont Property 174

- Syntax 174
- Properties 174

LabelItalic Property 174

- Syntax 174
- Properties 174
- Settings 174

LabelUnderline Property 175

- Syntax 175
- Properties 175
- Settings 175

Layer Property 175

- Syntax 175
- Properties 175
- Remarks 175

LCL Property 176

- Syntax 176
- Properties 176

Left Property 176

- Syntax 176
- Properties 176

LeftCenter Property 176

Syntax 176

Properties 177

Remarks 177

Legend Property 177

Syntax 177

Properties 177

Remarks 177

LegendAvgOver Property 177

Syntax 177

Properties 177

LegendDesc Property 177

Syntax 178

Properties 178

LegendHeadingLine Property 178

Syntax 178

Properties 178

Remarks 178

LegendHigh Property 178

Syntax 178

Properties 178

LegendHighOver Property 179

Syntax 179

Properties 179

LegendInterval Property 179

Syntax 179

Properties 179

LegendItemColor Property 179

Syntax 179

Properties 179

LegendLow Property 180

Syntax 180

- Properties 180

LegendLowOver Property 180

- Syntax 180
- Properties 180

LegendMode Property 180

- Syntax 180
- Properties 180

LegendTag Property 181

- Syntax 181
- Properties 181

LegendUnits Property 181

- Syntax 181
- Properties 181
- Remarks 181

LegendUser1 Property 181

- Syntax 181
- Properties 181
- Remarks 182

LegendUser10 Property 182

- Syntax 182
- Properties 182

LegendUser2 Property 182

- Syntax 182
- Properties 182

LegendUser3 Property 182

- Syntax 183
- Properties 183

LegendUser4 Property 183

- Syntax 183
- Properties 183

LegendUser5 Property 183

Syntax 183

Properties 183

LegendUser6 Property 183

Syntax 184

Properties 184

LegendUser7 Property 184

Syntax 184

Properties 184

LegendUser8 Property 184

Syntax 184

Properties 184

LegendUser9 Property 184

Syntax 185

Properties 185

LegendValue Property 185

Syntax 185

Properties 185

Linear Property 185

Syntax 185

Properties 185

Remarks 185

Lines Property 186

Syntax 186

Properties 186

Remarks 186

LinesofCode Property 186

Syntax 186

Properties 186

LineType Property 186

Syntax 186

Properties 186

LockStartTime Property 187

- Syntax 187
- Properties 187
- Settings 188

LoDisplay Property 188

- TimeAxis Syntax 188
- Properties 188
- ValueAxis Syntax 188
- Properties 188

LoginGroup Property 188

- Syntax 188
- Properties 189

LoginTimeout Property 189

- Syntax 189
- Properties 189
- Remarks 189

LoginUserFullName Property 189

- Syntax 189
- Properties 189

LoginUserName Property 190

- Syntax 190
- Properties 190

LoInValue Property 190

- Syntax 190
- Properties 190

LoLimit Property 190

- Syntax 190
- Properties 190

LoOutValue Property 191

- Syntax 191
- Properties 191

LowestDataValue Property 191
Syntax 191
Properties 191
LWL Property 191
Syntax 191
Properties 191
M-N 192
MainTitle Property 192
Syntax 192
Properties 192
MainTitleBold Property 192
Syntax 192
Properties 192
Settings 192
MainTitleFont Property 192
Syntax 193
Properties 193
MainTitleItalic Property 193
Syntax 193
Properties 193
Settings 193
MainTitleUnderline Property 193
Syntax 193
Properties 193
Settings 194
ManualMaxX Property 194
Syntax 194
Properties 194
Remarks 194
ManualMaxY Property 194
Syntax 194

- Properties 194
- ManualMinX Property 195
  - Syntax 195
  - Properties 195
  - Remarks 195
- ManualMinY Property 195
  - Syntax 195
  - Properties 195
- ManualScaleControlX Property 195
  - Syntax 196
  - Properties 196
- ManualScaleControlY Property 196
  - Syntax 196
  - Properties 196
- MapMode Property 196
  - Syntax 196
  - Properties 197
  - Remarks 197
- MarkDataPoints Property 197
  - Syntax 197
  - Properties 197
- MarkerChar Property 197
  - Syntax 197
  - Properties 197
- MarkerStyle Property 197
  - Syntax 198
  - Properties 198
  - Settings 198
- Master Property 198
  - Syntax 198
  - Properties 198

Return Value 198

Max_Dynamo_Desc_Length Property 198

Syntax 199

Properties 199

MaxCharactersPerLine Property 199

Syntax 199

Properties 199

Remarks 199

MaxLines Property 199

Syntax 199

Properties 199

Remarks 199

MaxPts Property 200

Syntax 200

Properties 200

MaxXAxisLabels Property 200

Syntax 200

Properties 200

MonoDeskColor Property 201

Syntax 201

Properties 201

MonoGraphBackColor Property 201

Syntax 201

Properties 201

MonoGraphForeColor Property 201

Syntax 202

Properties 202

MonoShadowColor Property 202

Syntax 202

Properties 202

MonoTableBackColor Property 202

Syntax 202

Properties 202

MonoTableForeColor Property 202

Syntax 203

Properties 203

MonoTextColor Property 203

Syntax 203

Properties 203

MultipleEGU Property 203

Syntax 203

Properties 203

Settings 203

Remarks 204

MultipleTimes Property 204

Syntax 204

Properties 204

Settings 204

MyNodeName Property 204

Syntax 204

Properties 204

Name Property 205

Syntax 205

Properties 205

Next Property 205

Syntax 205

Properties 205

Remarks 205

NlsPath Property 205

Syntax 205

Properties 205

Remarks 206

NoSaveOnClose Property 206

Syntax 206

Properties 206

Remarks 206

NumberOfCharacters Property 206

Syntax 206

Properties 206

Remarks 207

NumberOfHorizontalGridLines Property 207

Syntax 207

Properties 207

NumberOfItems Property 207

Syntax 207

Properties 207

Remarks 207

NumberOfLines Property 207

Syntax 208

Properties 208

Remarks 208

NumberOfPoints Property 208

Syntax 208

Properties 208

Remarks 208

NumberOfTargets Property 208

Syntax 208

Properties 208

Remarks 209

NumberOfVerticalGridLines Property 209

Syntax 209

Properties 209

NumOfPoints Property 209

Syntax 209

Properties 209

NumPointsToGraph Property 209

Syntax 210

Properties 210

Remarks 210

NumHGridLines Property 210

Syntax 210

Properties 210

NumLabels Property 210

Syntax 210

Properties 210

NumPts Property 211

Syntax 211

Properties 211

Remarks 211

NumRandomSubsets Property 211

Syntax 211

Properties 211

Remarks 211

NumScrollingSubsets Property 211

Syntax 212

Properties 212

NumTicks Property 212

Syntax 212

Properties 212

NumVGridLines Property 212

Syntax 212

Properties 212

O-P 212

Object Property 212

OpcAccessPath Property 213

Syntax 213

Properties 213

Remarks 213

OpcDataSource Property 213

Syntax 213

Properties 213

Remarks 213

OpcProgID Property 213

Syntax 214

Properties 214

Remarks 214

OpcServerMachineName Property 214

Syntax 214

Properties 214

Remarks 214

OriginalScreenHeight Property 214

Syntax 214

Properties 215

Remarks 215

OriginalScreenWidth Property 215

Syntax 215

Properties 215

Remarks 215

OriginX Property 215

Syntax 215

Properties 215

Remarks 215

OriginY Property 216

Syntax 216

Properties 216

- Remarks 216
- OutputValue Property 216
  - Syntax 216
  - Properties 216
- Owner Property 216
  - Syntax 216
  - Properties 216
  - Remarks 217
- Page Property 217
  - Syntax 217
  - Properties 217
  - Remarks 217
- Parent Property 217
  - Syntax 217
  - Properties 217
  - Remarks 218
- Path Property 218
  - Syntax 218
  - Properties 218
  - Remarks 218
- PauseIndicatorBlink Property 218
  - Syntax 218
  - Properties 218
  - Settings 218
- PauseIndicatorColor Property 219
  - Syntax 219
  - Properties 219
- PauseWithNewAlarmIndicatorBlink Property 219
  - Syntax 219
  - Properties 219
  - Settings 219

PauseWithNewAlarmIndicatorColor Property 220

Syntax 220

Properties 220

PenDescription Property 220

Syntax 220

Properties 220

PenLineColor Property 220

Syntax 220

Properties 220

PenLineStyle Property 221

Syntax 221

Properties 221

Settings 221

Remarks 221

PenLineWidth Property 221

Syntax 221

Properties 221

PenNum Property 222

Syntax 222

Properties 222

Remarks 222

Pens Property 222

Syntax 222

Properties 222

Remarks 222

PenType Property 222

Syntax 222

Properties 223

Return Values 223

Remarks 223

PictureDefaultAlwaysOnTop Property 223

Syntax 223

Example 223

Properties 223

PictureDefaultBackColor Property 223

Syntax 223

Example 224

Properties 224

PictureDefaultHeight Property 224

Syntax 224

Example 224

Properties 224

PictureDefaultResizable Property 224

Syntax 224

Example 224

Properties 224

PictureDefaultRuntimeVisible Property 225

Syntax 225

Example 225

Properties 225

PictureDefaultSystemMenu Property 225

Syntax 225

Example 225

Properties 225

PictureDefaultTitlebar Property 226

Syntax 226

Example 226

Properties 226

PictureDefaultWidth Property 226

Syntax 226

Example 226

Properties 226

PictureHeight Property 227

Syntax 227

Properties 227

PictureName Property 227

Syntax 227

Properties 227

PicturePath Property 227

Syntax 227

Properties 227

Remarks 228

PictureWidth Property 228

Syntax 228

Properties 228

PieType Property 228

Syntax 228

Properties 228

Settings 228

PlotOnChartRefresh Property 229

Syntax 229

Properties 229

PlottingMethod Property 229

Syntax 229

Properties 229

PointType Property 229

Syntax 230

Properties 230

Previous Property 230

Syntax 230

Properties 230

Remarks 230

PrimarySecPath Property 230

- Syntax 231
- Properties 231
- Remarks 231

ProcedureDeclaration Property 231

- Syntax 231
- Properties 231
- Remarks 231

ProcedureName Property 231

- Syntax 231
- Properties 231

Procedures Property 232

- Syntax 232
- Properties 232
- Remarks 232

ProcedureStatement Property 232

- Syntax 232
- Properties 232

ProgId Property 232

- Syntax 232
- Properties 233
- Remarks 233

ProjectPath Property 233

- Syntax 233
- Properties 233

Property1 Property 233

- Syntax 233
- Properties 233
- Remarks 233

Property10 Property 233

- Syntax 234
- Properties 234

Remarks 234

Property2 Property 234

Syntax 234

Properties 234

Remarks 234

Property3 Property 234

Syntax 234

Properties 234

Remarks 235

Property4 Property 235

Syntax 235

Properties 235

Remarks 235

Property5 Property 235

Syntax 235

Properties 235

Remarks 235

Property6 Property 235

Syntax 236

Properties 236

Remarks 236

Property7 Property 236

Syntax 236

Properties 236

Remarks 236

Property8 Property 236

Syntax 236

Properties 236

Remarks 237

Property9 Property 237

Syntax 237

Properties 237

Remarks 237

Q-R 237

Quality Property 237

Syntax 237

Properties 237

Remarks 237

QueueEvents Property 238

Syntax 238

Properties 238

Settings 238

Remarks 238

QuickConfigure Property 238

Syntax 238

Properties 239

Settings 239

QuickStyle Property 239

Syntax 239

Properties 239

Radius Property 240

Syntax 240

Properties 240

RandomSubsetsToGraph Property 240

Syntax 240

Properties 240

RawFormat Property 240

Syntax 240

Properties 241

Settings 241

RecalculateViewport Property 241

Syntax 241

Properties 241

Settings 241

Remarks 242

RefreshRate Property 242

Syntax 242

Properties 242

RemoveNonWindowsUsers Property 242

Syntax 242

Properties 242

Remarks 242

ResetPercentage Property 242

Syntax 243

Properties 243

Resizable Property 243

Syntax 243

Properties 243

Settings 243

ResolveSourceName Property 243

Syntax 243

Properties 243

Remarks 244

Revision Property 244

Syntax 244

Properties 244

Return Value 244

RevisionNumber Property 244

Syntax 244

Properties 244

Remarks 244

Right Property 244

Syntax 245

Properties 245

Remarks 245

RightCenter Property 245

Syntax 245

Properties 245

Remarks 245

RotationAngle Property 245

Syntax 245

Properties 245

Remarks 246

RoundnessX Property 246

Syntax 246

Properties 246

RoundnessY Property 246

Syntax 246

Properties 246

RunIndicatorBlink Property 246

Syntax 247

Properties 247

Settings 247

RunIndicatorColor Property 247

Syntax 247

Properties 247

RuntimeVisible Property 247

Syntax 247

Properties 247

Settings 248

Remarks 248

S 248

Saved Property 248

Syntax 248

Properties 248
Return Values 248
SaveThumbnail Property 248
Syntax 249
Properties 249
Settings 249
ScalesWidth Property 249
Syntax 249
Properties 249
Settings 249
Remarks 249
SchedulePath Property 250
Syntax 250
Properties 250
Remarks 250
ScreenHeight Property 250
Syntax 250
Properties 250
Remarks 250
ScreenWidth Property 250
Syntax 251
Properties 251
Remarks 251
ScrollDirection Property 251
Syntax 251
Properties 251
Settings 251
ScrollGrid Property 251
Syntax 251
Properties 252
Settings 252

ScrollItems Property 252

- Syntax 252
- Properties 252
- Settings 252

ScrollPercentage Property 252

- Syntax 252
- Properties 253
- Remarks 253

SecondaryImageDisplayed Property 253

- Syntax 253
- Properties 253
- Settings 253
- Remarks 253

SecurityArea Property 253

- Syntax 253
- Properties 254

SelectedDatasource Property 254

- Syntax 254
- Properties 254

SelectedFieldName Property 254

- Syntax 254
- Properties 254

SelectedNodeName Property 254

- Syntax 255
- Properties 255

SelectedShapes Property 255

- Syntax 255
- Properties 255
- Remarks 255

SelectedTagName Property 255

- Syntax 255

- Properties 255
- SelectionTimeout Property 255
  - Syntax 256
  - Properties 256
- SendAlarmMessages Property 256
  - Syntax 256
  - Properties 256
  - Remarks 256
- ShadowColor Property 256
  - Syntax 256
  - Properties 256
- SharedTableName Property 257
  - Syntax 257
  - Properties 257
  - Remarks 257
- ShowAxis Property 257
  - Syntax 257
  - Properties 257
  - Settings 258
- ShowDatabaseTab Property 258
  - Syntax 258
  - Properties 258
  - Settings 258
- ShowDataServersTab Property 258
  - Syntax 258
  - Properties 258
  - Settings 259
- ShowDate Property 259
  - Syntax 259
  - Properties 259
  - Settings 259

ShowDSLegend Property 259
Syntax 259
Properties 259
Settings 260
ShowGaps Property 260
Syntax 260
Properties 260
Settings 260
Remarks 260
ShowGlobalsTab Property 260
Syntax 261
Properties 261
Settings 261
ShowGridLines Property 261
Syntax 261
Properties 261
Settings 261
ShowHeaders Property 261
Syntax 262
Properties 262
Settings 262
ShowHistoricalTab Property 262
Syntax 262
Properties 262
Settings 262
ShowHorizontalGrid Property 262
Syntax 263
Properties 263
Settings 263
ShowLegend Property 263
Syntax 263

Properties 263

Settings 263

ShowLine Property 263

Syntax 264

Properties 264

Settings 264

ShowPicturesTab Property 264

Syntax 264

Properties 264

Settings 264

ShowRowNumbers Property 264

Syntax 265

Properties 265

Settings 265

ShowScrollBars Property 265

Syntax 265

Properties 265

Settings 265

ShowStatusBar Property 265

Syntax 266

Properties 266

Settings 266

ShowTimeAxis Property 266

Syntax 266

Properties 266

Settings 266

ShowTimeAxisTitle Property 266

Syntax 267

Properties 267

Settings 267

ShowTimeCursor Property 267

Syntax 267

Properties 267

Settings 267

ShowTimeCursorToolTips Property 267

Syntax 268

Properties 268

Settings 268

ShowTimeStamp Property 268

Syntax 268

Properties 268

Settings 268

ShowTitle Property 269

Syntax 269

Properties 269

Settings 269

ShowValueAxis Property 269

Syntax 269

Properties 269

Settings 269

ShowValueAxisTitle Property 269

Syntax 270

Properties 270

Settings 270

ShowVerticalGrid Property 270

Syntax 270

Properties 270

Settings 270

ShowXAxis Property 270

Syntax 271

Properties 271

ShowYAxis Property 271

Syntax 271

Properties 271

SmoothingMode Property 271

Syntax 272

Properties 272

SmoothShapeOption Property 272

Syntax 272

Properties 272

SmoothShapes Property 273

Syntax 273

Properties 273

Settings 273

SnapToGrid Property 273

Syntax 273

Properties 273

Settings 274

SortColumnName Property 274

Syntax 274

Properties 274

SortOrderAscending Property 274

Syntax 274

Properties 275

Settings 275

Source Property 275

Syntax 275

Properties 275

Remarks 275

Sources Property 275

Syntax 276

Properties 276

Remarks 276

SourceValidated Property 276

Syntax 276

Properties 276

Settings 276

SPCChartType Property 276

Syntax 276

Properties 276

SPCInterval Property 277

Syntax 277

Properties 277

SPCType Property 277

Syntax 277

Properties 277

Remarks 278

StartAngle Property 278

Syntax 278

Properties 278

Remarks 278

StartCap Property 278

Syntax 278

Properties 278

Settings 279

StartDateMode Property 279

Syntax 279

Properties 279

Settings 279

Remarks 279

StartDateType Property 279

Syntax 280

Properties 280

Settings 280

Remarks 280

StartPoint Property 280

Syntax 280

Properties 280

Remarks 280

StartTime Property 281

Chart Syntax 281

Properties 281

Timer Syntax 281

Remarks 281

StartTimeMode Property 281

Syntax 281

Properties 281

Settings 282

Remarks 282

StartTimeType Property 282

Syntax 282

Properties 282

Settings 282

Remarks 282

StartX Property 282

Syntax 282

Properties 283

StartY Property 283

Syntax 283

Properties 283

Status Property 283

Syntax 283

Properties 283

Return Values 283

StatusBar Property 284

- Syntax 284
- Properties 284

StatusFontSize Property 284
- Syntax 284
- Properties 284

SteppedTrend Property 284
- Syntax 284
- Properties 284
- Settings 285

StorageMode Property 285
- Syntax 285
- Properties 285
- Remarks 285

StretchMode Property 286
- Syntax 286
- Properties 286

StrikeThrough Property 286
- Syntax 286
- Properties 286
- Settings 287

SubTitle Property 287
- Syntax 287
- Properties 287

SubTitleBold Property 287
- Syntax 287
- Properties 287
- Settings 287

SubTitleFont Property 288
- Syntax 288
- Properties 288

SubTitleItalic Property 288

Syntax 288

Properties 288

Settings 288

SubTitleUnderline Property 288

Syntax 289

Properties 289

Settings 289

System Property 289

Syntax 289

Properties 289

Remarks 289

SystemMenu Property 289

Syntax 290

Properties 290

Settings 290

T 290

TableBackColor Property 290

Syntax 290

Properties 290

TableFont Property 290

Syntax 290

Properties 290

TableForeColor Property 291

Syntax 291

Properties 291

TextColor Property 291

Syntax 291

Properties 291

Thickness Property 291

Syntax 291

Properties 292

ThicknessType Property 292
- Syntax 292
- Properties 292

Thumbnail Property 292
- Syntax 292
- Properties 292
- Settings 293

TimeAxis Property 293
- Syntax 293
- Properties 293
- Remarks 293

TimeAxisNumLabels Property 293
- Syntax 294
- Properties 294

TimeAxisNumTicks Property 294
- Syntax 294
- Properties 294
- Remarks 294

TimeAxisTitle Property 294
- Syntax 294
- Properties 294

TimeBeforeNow Property 295
- Syntax 295
- Properties 295
- Remarks 295

TimeCursorColor Property 295
- Syntax 295
- Properties 295

TimeCursorPos Property 295
- Syntax 295
- Properties 296

TimeCursorStyle Property 296
- Syntax 296
- Properties 296

TimeCursorTooltipColor Property 296
- Syntax 296
- Properties 296

Timeout Property 297
- Syntax 297
- Properties 297
- Remarks 297

TimerEnabled Property 297
- Syntax 297
- Properties 297
- Settings 297

Timestamp Property 298
- Syntax 298
- Properties 298
- Remarks 298

TimeZoneBiasExplicit Property 298
- Syntax 298
- Properties 298
- Remarks 299

TimeZoneBiasRelative Property 299
- Syntax 299
- Properties 299
- Settings 299
- Remarks 299

Titlebar Property 299
- Syntax 299
- Properties 299
- Settings 300

ToggleRate Property 300

Syntax 300

Properties 300

ToggleSource Property 300

Syntax 300

Properties 300

Remarks 300

Tolerance Property 300

Syntax 301

Properties 301

Remarks 301

ToolbarManager Property 301

Syntax 301

Properties 301

Remarks 301

ToolbarPath Property 301

Syntax 301

Properties 302

Remarks 302

TooltipOption Property 302

Syntax 302

Properties 302

Top Property 302

Syntax 302

Properties 303

TopCenter Property 303

Syntax 303

Properties 303

Remarks 303

TopLeft Property 303

Syntax 303

- Properties 303
- Remarks 303

TopRight Property 303

- Syntax 304
- Properties 304
- Remarks 304

TopVisibleRow Property 304

- Syntax 304
- Properties 304
- Remarks 304

TotalFilteredAlarms Property 304

- Syntax 304
- Properties 304

TranslateOnOpen Property 305

- Syntax 305
- Properties 305
- Settings 305

Transparency Property 305

- Syntax 305
- Properties 305
- Settings 305
- Remarks 306

Transparent Property 306

- Syntax 306
- Properties 306
- Settings 306
- Remarks 306

TransparentColor Property 306

- Syntax 306
- Properties 306

TreatSinglePointsAsLines Property 307

Syntax 307

Properties 307

Settings 307

TriggerType Property 307

Syntax 307

Properties 307

Settings 307

Remarks 308

TrimMaxLength Property 308

Syntax 308

Properties 308

TrimType Property 308

Syntax 308

Properties 308

Settings 308

TruncateTitles Property 309

Syntax 309

Properties 309

Settings 309

Type Property 309

Syntax 309

Properties 309

Remarks 310

U-V 310

UCL Property 310

Syntax 310

Properties 310

UnacknowledgedAlarmColor Property 310

Syntax 310

Properties 310

Underline Property 310

- Syntax 311
- Properties 311
- Settings 311

UniformScale Property 311

- Syntax 311
- Properties 311
- Settings 311
- Remarks 311

Units Property 312

- Syntax 312
- Properties 312

UpdateOnPropChange Property 312

- Syntax 312
- Properties 312
- Settings 312

UpdateRate Property 312

- Syntax 313
- Properties 313

UseDefaultYAxisSettings Property 313

- Syntax 313
- Properties 313
- Settings 313

UseDelta Property 314

- Syntax 314
- Properties 314
- Settings 314
- Remarks 314

UseDomainSecurity Property 314

- Syntax 315
- Properties 315
- Remarks 315

UseDSLimits Property 315

Syntax 315

Properties 315

Settings 315

UseLocalSecurity Property 316

Syntax 316

Properties 316

Remarks 316

UseMarker Property 316

Syntax 316

Properties 316

Settings 316

Remarks 317

UserDef1ColumnName Property 317

Syntax 317

Properties 317

Remarks 317

UserDef2ColumnName Property 317

Syntax 317

Properties 317

Remarks 317

UserPreferences Property 318

Syntax 318

Properties 318

Remarks 318

UseUnacknowledgedAlarmColor Property 318

Syntax 318

Properties 318

Settings 318

UWL Property 318

Syntax 319

Properties 319

Value Property 319

Syntax 319

Properties 319

Remarks 319

ValueAxis Property 319

Syntax 319

Properties 319

Remarks 320

ValueAxisNumLabels Property 320

Syntax 320

Properties 320

ValueAxisNumTicks Property 320

Syntax 320

Properties 320

ValueAxisTitle Property 320

Syntax 320

Properties 320

VariableType Property 321

Syntax 321

Properties 321

Settings 321

Version Property 321

Syntax 321

Properties 321

Remarks 322

VerticalFillDirection Property 322

Syntax 322

Properties 322

Settings 322

VerticalFillPercentage Property 322

Syntax 322

Properties 322

Remarks 323

VerticalGridColor Property 323

Syntax 323

Properties 323

VerticalGridStyle Property 323

Syntax 323

Properties 323

Settings 323

VerticalPosition Property 324

Syntax 324

Properties 324

Remarks 324

VerticalScaleDirection Property 324

Syntax 324

Properties 324

Settings 324

Remarks 325

VerticalScalePercentage Property 325

Syntax 325

Properties 325

Remarks 325

ViewingStyle Property 325

Syntax 325

Properties 326

ViewportHeight Property 326

Syntax 326

Properties 326

Remarks 326

ViewportLeft Property 326

Syntax 326

Properties 327

Remarks 327

ViewportTop Property 327

Syntax 327

Properties 327

Remarks 327

ViewportWidth Property 327

Syntax 328

Properties 328

Remarks 328

Visible Property 328

Syntax 328

Properties 328

Settings 328

Remarks 328

VisibleUnacknowledgedAlarms Property 328

Syntax 329

Properties 329

W-Z 329

WholeDigits Property 329

Syntax 329

Properties 329

Width Property 329

Syntax 329

Properties 329

Remarks 330

WindowHeightPercentage Property 330

Syntax 330

Properties 330

Remarks 330

WindowLeftPercentage Property 330

Syntax 330

Properties 330

Remarks 330

WindowName Property 331

Syntax 331

Properties 331

Remarks 331

WindowState Property 331

Syntax 331

Properties 331

Settings 331

WindowTopPercentage Property 332

Syntax 332

Properties 332

Remarks 332

WindowWidthPercentage Property 332

Syntax 332

Properties 332

Remarks 332

WizardName Property 332

Syntax 332

Properties 333

Remarks 333

WorkSpaceStartupMode Property 333

Syntax 333

Properties 333

X Property 333

Syntax 333

Properties 333

XAxisDatasetPosition Property 334

Syntax 334

Properties 334

Remarks 334

XAxisLabel Property 334

Syntax 334

Properties 334

XAxisScaleControl Property 334

Syntax 335

Properties 335

XAxisType Property 335

Syntax 335

Properties 335

Remarks 335

Y Property 335

Syntax 335

Properties 336

YAxesStyle Property 336

Syntax 336

Properties 336

YAxisAlwaysVisible Property 336

Syntax 336

Properties 336

Settings 337

YAxisLabel Property 337

Syntax 337

Properties 337

YAxisLongTicks Property 337

Syntax 337

Properties 337

Settings 337

YAxisScaleControl Property 337

Syntax 338

Properties 338

YAxisTitle Property 338

Syntax 338

Properties 338

Zoom Property 338

Syntax 338

Properties 338

ZoomDirection Property 339

Syntax 339

Properties 339

Settings 339

ZoomType Property 339

Syntax 339

Properties 339

Method Summary 341

A 341

B 341

C 341

D 342

E 343

F 343

G-H 344

I-K 346

L 346

M-N 347

O 347

P 347

Q 347

R 348

S 348

T 350
U 351
V-W 351
X-Y 351
Z 351
A-B 352
AboutBox Method 352
Syntax 352
Properties 352
AckAlarm Method 352
Syntax 352
Properties 352
Return Value 352
AckAlarmPage Method 352
Syntax 352
Properties 353
Return Value 353
AckAlarmPageEx Method 353
Syntax 353
Properties 353
Return Value 353
AckAllAlarms Method 353
Syntax 353
Properties 353
Return Value 354
ActivateWorkspaceUI Method 354
Syntax 354
Properties 354
Remarks 354
Add Method 354
Documents Collection Syntax 354

© 2024 General Electric Company. All rights reserved.

Properties 354

Return Value 355

Remarks 355

Procedures Collection Syntax 355

Properties 355

Lines Collection Syntax 355

Properties 355

DataItems and Groups Collection Syntax 356

Properties 356

Return Value 356

AddDataSet Method 356

Syntax 356

Properties 356

Return Value 356

AddEventHandler Method 356

Syntax 357

Properties 357

AddImage Method 357

Syntax 357

Properties 357

AddKeyMacro Method 357

Syntax 357

Properties 357

AddLegendItem Method 358

Syntax 358

Properties 358

AddLevel Method 358

Syntax 359

Properties 359

AddObject Method 359

Syntax 359

Properties 359
Remarks 359
AddPen Method 359
Syntax 359
Properties 360
Return Value 360
AddPictureToStartupList Method 360
Syntax 360
Properties 360
AddPoint Method 360
Syntax 361
Properties 361
Remarks 361
AddProcedure Method 361
Syntax 361
Properties 361
Align Method 361
Syntax 361
Properties 362
Remarks 362
ApplyProperty Method 362
AutoScaleDisplayLimits Method 362
Syntax 362
Properties 362
Remarks 362
BringToFront Method 362
Syntax 363
Properties 363
Remarks 363
BuildObject Method 363
Syntax 363

Properties 363

Return Value 363

Remarks 364

C 364

CanConstruct Method 364

Syntax 364

Properties 364

Remarks 364

CheckAccountExpiration Method 364

Syntax 364

Properties 364

Return Value 365

CheckforDuplicateKeyMacros Method 365

Syntax 365

Properties 365

CheckSecurityEnabled Method 365

Syntax 365

Properties 365

Return Value 366

CheckSyntax Method 366

Syntax 366

Properties 366

Return Value 366

CheckUserApplicationAccess Method 366

Syntax 366

Properties 366

CheckUserAreaAccess Method 367

Syntax 367

Properties 367

Return Value 367

Clear Method 367

- Syntax 367
- Properties 367

ClearUndo Method 368

- Syntax 368
- Properties 368
- Remarks 368

Close Method 368

- Documents Collection Syntax 368
- Properties 368
- Remarks 368
- Document Object Syntax 369
- Remarks 369

Commit Method 369

- Syntax 369
- Properties 369

Connect Method 369

- Syntax 370
- Properties 370

ConnectDataSet Method 370

- Syntax 370
- Properties 370

ConnectedPropertyCount Method 370

- Syntax 371
- Properties 371

Construct Method 371

- Syntax 371
- Properties 371

Convert_A_Group_To_A_Dynamo_By_Name Method 371

- Syntax 372
- Properties 372

Convert_A_Group_To_A_Dynamo_By_Ref Method 373

Syntax 373

Properties 374

ConvertPipe Method 375

Syntax 375

Properties 375

Remarks 375

ConvertSecurityAreaNameToNumber Method 376

Syntax 376

Properties 376

Return Value 376

ConvertSecurityAreaNumberToName Method 376

Syntax 376

Properties 376

Return Value 376

ConvertToEnhancedCoordinates Method 376

Syntax 377

Properties 377

Return Value 377

Remarks 377

ConvertToOriginalCoordinates Method 377

Syntax 377

Properties 377

Return Value 377

Remarks 377

Copy Method 378

Syntax 378

Properties 378

Remarks 378

CopyAsBitmap Method 378

Syntax 378

Properties 378

Remarks 378

CopytoClipboard Method 378

Syntax 378

Properties 378

Coupled_Activate_Workspace_UI Method 379

Syntax 379

Properties 379

Remarks 379

Coupled_DeActivate_Workspace_UI Method 379

Syntax 379

Properties 379

Remarks 380

CreateDynamoByGrouping Method 380

Syntax 380

Properties 380

CreateFromDialog Method 380

Syntax 380

Properties 380

Remarks 380

CreateFromProgID Method 380

Syntax 381

Properties 381

Remarks 381

CreateWithMouse Method 381

Syntax 381

Properties 381

Cut Method 381

Syntax 381

Properties 381

Remarks 382

D-E 382

DeActivateWorkspaceUI Method 382

Syntax 382

Properties 382

Remarks 382

DefaultView Method 382

Syntax 382

Properties 383

DelAlarm Method 383

Syntax 383

Properties 383

Return Value 383

DeleteAllAlarms Method 383

Syntax 383

Properties 383

Return Value 383

DeleteAllDataSets Method 384

Syntax 384

Properties 384

DeleteDataSet Method 384

Syntax 384

Properties 384

DeleteImage Method 384

Syntax 384

Properties 384

DeletePen Method 385

Syntax 385

Properties 385

DeletePoint Method 385

Syntax 385

Properties 385

DeleteSelectedObjects Method 385

Syntax 385

Properties 385

Remarks 386

DemandFire Method 386

Scheduler Object Syntax 386

Properties 386

Timer and Event Object Syntax 386

Properties 386

Remarks 386

DeselectObject Method 386

Syntax 387

Properties 387

Remarks 387

DestroyObject Method 387

Syntax 387

Properties 387

DisableNonSelectionEvents Method 387

Disconnect Method 387

Syntax 387

Properties 387

Remarks 388

DisplaysControlPoints Method 388

Syntax 388

Properties 388

DoesPropertyHaveTargets Method 388

Syntax 388

Properties 388

DoExtendLines Method 389

Syntax 389

Properties 389

DoLinesToPolyline Method 389

Syntax 389

Properties 389

DoMenuCommand Method 389

Syntax 389

Properties 390

DoTrimLines Method 390

Syntax 390

Properties 390

DumpProperties Method 390

Syntax 390

Properties 390

Duplicate Method 391

Syntax 391

Properties 391

Remarks 391

EditPicture Method 391

Syntax 391

Properties 391

Remarks 392

Enable Method 392

Syntax 392

Properties 392

Enumerate_All_Dynamos Method 392

Syntax 392

Properties 392

Enumerate_All_Groups Method 392

Syntax 392

Properties 393

Enumerate_Top_Level_Dynamos Method 393

Syntax 393

Properties 393

Enumerate_Top_Level_Groups Method 393
Syntax 393
Properties 393
Remarks 394
ExchangePenPositions Method 394
Syntax 394
Properties 394
Remarks 394
Execute Method 394
Syntax 394
Properties 394
Return Value 394
ExecuteKeyMacro Method 394
Syntax 395
Properties 395
ExportData Method 395
Syntax 395
Properties 395
ExportImage Method 396
Syntax 396
Properties 396
Remarks 399
ExportLanguageFile Method 399
Syntax 399
Properties 399
F 400
FindAndReplaceDialog Method 400
Syntax 400
Properties 400
Remarks 400
FindInString Method 400

Syntax 400

Properties 401

Remarks 401

FindObject Method 401

Syntax 401

Properties 401

Return Value 402

Remarks 402

FindReplaceInObject Method 402

Syntax 402

Properties 402

FindReplaceInString Method 402

Syntax 403

Properties 403

FitDocumentToWindow Method 403

Syntax 403

Properties 403

FitWindowToDocument Method 404

Syntax 404

Properties 404

FixCheckApplicationAccess Method 404

Syntax 404

Properties 404

Return Value 404

FixCheckApplicationAccessQuiet Method 404

Syntax 405

Properties 405

Return Value 405

FixCheckAreaAccess Method 405

Syntax 405

Properties 405

Return Value 405

Remarks 405

FixCheckAreaAccessQuiet Method 405

Syntax 406

Properties 406

Return Value 406

Remarks 406

FixCheckSecurityEnabled Method 406

Syntax 406

Properties 406

Return Value 406

FixGetManualAlmDeleteEnabled Method 406

Syntax 406

Properties 406

FixGetUserInfo Method 407

Syntax 407

Properties 407

FixLogin Method 407

Syntax 407

Properties 407

FixLogout Method 407

Syntax 407

Properties 408

FontProperties Method 408

Syntax 408

Properties 408

FullView Method 408

Syntax 408

Properties 408

G-H 408

Get_Last_Prompt_Value Method 408

Syntax 409

Properties 409

Return Value 409

Get_Last_Result_String Method 409

Syntax 409

Properties 409

Return Value 409

GetAlarmBackgroundColor Method 409

Syntax 410

Properties 410

Return Value 410

GetAlarmForegroundColor Method 410

Syntax 410

Properties 410

Return Value 411

GetBoundRect Method 411

Syntax 411

Properties 411

Remarks 412

GetChartEndTime Method 412

Syntax 412

Properties 412

GetChartStartTime Method 412

Syntax 412

Properties 412

GetColHeadings Method 413

Syntax 413

Properties 413

Remarks 413

GetColumnInfo Method 413

Syntax 413

Properties 413

GetConnectionInformation Method 413

Syntax 413

Properties 414

GetConnectionParameters Method 414

Syntax 414

Properties 414

GetContinuousUser Method 414

Syntax 414

Properties 415

Return Value 415

GetCurrentDataSet Method 415

Syntax 415

Properties 415

Return Value 415

GetCurrentValue Method 415

Syntax 415

Properties 415

GetCurrentValueWithQuality Method 416

Syntax 416

Properties 416

GetDataSetByPosition Method 416

Syntax 416

Properties 416

Return Value 417

GetDeviceRect Method 417

Syntax 417

Properties 417

GetDuration Method 417

Syntax 417

Properties 417

GetGlobalDuration Method 417

Syntax 418

Properties 418

GetErrorString Method 418

Syntax 418

Properties 418

Return Value 418

GetEventHandlerIndex Method 418

Syntax 419

Properties 419

GetFullname Method 419

Syntax 419

Properties 419

Return Value 419

GetGlobalHistoricalUpdateRate Method 419

Syntax 419

Properties 420

GetIndirectionInfo Method 420

GetInterval Method 420

Syntax 420

Properties 420

GetKeyMacro Method 420

Syntax 420

Properties 421

GetKeyMacroIndex Method 421

Syntax 421

Properties 421

GetLevel Method 421

Syntax 421

Properties 421

GetNumberOfDataSets Method 422

Syntax 422

Properties 422

Return Value 422

GetObjectInfo Method 422

Syntax 422

Properties 422

Return Value 422

Remarks 423

GetPenDataArray Method 423

Syntax 423

Properties 423

Remarks 423

GetPenDataArrayEx Method 423

Syntax 423

Properties 423

Remarks 424

GetPointAt Method 424

Syntax 424

Properties 424

Return Value 424

Remarks 424

GetPriorityColor Method 424

Syntax 424

Properties 424

Return Value 425

GetProcedureIndex Method 425

Syntax 425

Properties 425

GetProperty Method 425

Syntax 425

Properties 426

Remarks 426

GetPropertyAttributes Method 426

Syntax 426

Properties 426

GetPropertyTargets Method 427

Syntax 427

Properties 427

Remarks 427

GetRibbonView Method 428

Syntax 428

Properties 428

Return Value 428

GetSelectedAlmExt Method 428

Syntax 428

Properties 428

Return Value 428

Remarks 428

GetSelectedNodeTag Method 429

Syntax 429

Properties 429

Return Value 429

Remarks 429

GetSelectedRow Method 429

Syntax 429

Properties 429

Return Value 430

Remarks 430

GetSelectedRowAlarmInfo Method 430

Syntax 430

Properties 430

Return Value 431

Remarks 431

GetSelectedRowsAlarmInfo Method 431

Syntax 431

Properties 431

Return Value 432

Remarks 432

GetSelectedUserDefFields Method 432

Syntax 432

Properties 433

Return Value 433

Remarks 433

GetSignature Method 433

Syntax 433

Properties 433

Return Value 434

GetSignatureAndWriteValue Method 434

Syntax 435

Properties 435

Return Value 436

GetStatusColor Method 436

Syntax 436

Properties 436

Return Value 437

GetStatusFont Method 437

Syntax 437

Properties 437

Return Value 438

GetTimeBeforeNow Method 438

Syntax 438

Properties 438

GetTimeCursorInfo Method 439

Syntax 439

Properties 439

GetUserID Method 439

Syntax 439

Properties 439

Return Value 439

GetWindowLocation Method 439

Syntax 440

Properties 440

GlobalScrollBackFast Method 440

Syntax 440

Properties 440

GlobalScrollBackSlow Method 440

Syntax 441

Properties 441

GlobalScrollForwardFast Method 441

Syntax 441

Properties 441

GlobalScrollForwardSlow Method 441

Syntax 441

Properties 441

GlobalTimerApply Method 442

Syntax 442

Properties 442

Group Method 442

Syntax 442

Properties 442

Remarks 442

HiLoDisplay Method 442

Syntax 443

Properties 443

I-L 443
ImportToolbar Method 443
Syntax 443
Properties 443
Initialize Method 443
Syntax 443
Properties 444
Return Value 444
InitializeList Method 444
Syntax 444
Properties 444
Return Value 444
InsertPoint Method 444
Syntax 444
Properties 445
InteractiveExport Method 445
Syntax 445
Properties 445
IsColorSelectionVisible Method 445
Syntax 445
Properties 445
Return Value 445
Remarks 445
IsConnected Method 446
Syntax 446
Properties 446
IsEmpty Method 446
Syntax 446
Properties 446
IsKeyMacroDefined Method 446
Syntax 447

- Properties 447
- IsNodeSignEnabled Method 447
  - Syntax 447
  - Properties 447
  - Return Value 447
- IsSignatureRequired Method 447
  - Syntax 447
  - Properties 447
  - Return Value 448
- IsSignatureRequiredForList Method 448
  - Syntax 449
  - Properties 449
  - Return Value 449
- Item Method 449
  - Syntax 450
  - Properties 450
  - Return Value 450
  - Remarks 450
- ListEvents Method 450
  - Syntax 450
  - Properties 450
- ListMethods Method 450
  - Syntax 450
  - Properties 451
- ListProperties Method 451
  - Syntax 451
  - Properties 451
  - Remarks 451
- ListWindowsGroupNames Method 451
  - Syntax 451
  - Properties 451

Remarks 452

Load_TS_List Method 452

Syntax 452

Properties 452

LoadImage Method 452

Syntax 452

Properties 452

LoadTagGroupFile Method 453

Syntax 453

Properties 453

Remarks 453

LogicalToPercentage Method 453

Syntax 453

Properties 453

LogicalToUserFormPoint Method 454

Syntax 454

Properties 454

Remarks 454

M-P 454

MakeLinesHorizontal Method 454

Syntax 454

Properties 454

MakeLinesVertical Method 455

Syntax 455

Properties 455

MakeSameSize Method 455

Syntax 455

Properties 455

Remarks 455

Modify Method 455

Syntax 456

Properties 456

ModifyColumnLength Method 456

Syntax 456

Properties 456

Move Method 456

Syntax 456

Properties 456

Open Method 457

Syntax 457

Properties 457

Return Value 457

Open_QT_Pic Method 457

Syntax 457

Properties 458

Open_QT_Pic_Ex Method 458

Syntax 458

Properties 458

Open_TCP_Pic Method 458

Syntax 458

Properties 458

Open_TCP_Pic_Ex Method 458

Syntax 459

Properties 459

Open_TS_Pic Method 459

Syntax 459

Properties 459

Open_TS_Pic_Ex Method 459

Syntax 459

Properties 459

Open_TS_Pic_Type Method 460

Syntax 460

Properties 460

Open_TS_Pic_Type_Ex Method 460

Syntax 460

Properties 460

ParseConnectionSource Method 461

Syntax 461

Properties 461

Remarks 461

Paste Method 461

Syntax 462

Properties 462

Remarks 462

PasteFromClipboard Method 462

Syntax 462

Properties 462

PasteSpecial Method 462

Syntax 462

Properties 462

Remarks 463

Pause Method 463

Syntax 463

Properties 463

Remarks 463

PauseAlarmRead Method 463

Syntax 463

Properties 463

Remarks 463

PercentageToLogical Method 463

Syntax 464

Properties 464

PercentageToPixel Method 464

Syntax 464

Properties 464

PixelToPercentage Method 465

Syntax 465

Properties 465

PrintChart Method 465

Syntax 465

Properties 465

Remarks 466

PrintOut Method 466

Syntax 466

Properties 466

Return Value 466

PromptToChangePassword Method 467

Syntax 467

Properties 467

Return Value 467

Q-R 467

Quit Method 467

Syntax 467

Properties 467

Read Method 467

Syntax 468

Properties 468

Refresh Method 468

Syntax 468

Properties 468

RefreshChartData Method 468

Syntax 468

Properties 468

Remove Method 468

Syntax 469

Properties 469

DataItems and Groups Collection Syntax 469

Properties 469

RemoveAll Method 469

Syntax 469

Properties 469

RemoveAllLevels Method 469

Syntax 469

Properties 470

RemoveItem Method 470

Syntax 470

Properties 470

RemoveKeyMacro Method 470

Syntax 470

Properties 470

RemoveLegendItem Method 470

Syntax 470

Properties 471

RemoveLevel Method 471

Syntax 471

Properties 471

RemoveObject Method 471

Syntax 472

Properties 472

Remarks 472

RemovePictureFromStartupList Method 472

Syntax 472

Properties 472

ReplaceDocument Method 472

Syntax 472

- Properties 473
- Return Value 473
- Remarks 473

ReplaceInString Method 473

- Syntax 473
- Properties 473
- Remarks 474

Replace_QT_Pic Method 474

- Syntax 474
- Properties 474

Replace_TCP_Pic Method 474

- Syntax 474
- Properties 474

Replace_TS_Pic Method 474

- Syntax 475
- Properties 475

Replace_TS_Pic_Type Method 475

- Syntax 475
- Properties 475

ResetChartData Method 475

- Syntax 475
- Properties 475

ResetObjectStats Method 475

- Syntax 476
- Properties 476

ResetStats Method 476

- Syntax 476
- Properties 476

ResetZoom Method 476

- Syntax 476
- Properties 476

Remarks 477

ResolveTagGroupFile Method 477

Syntax 477

Properties 477

Remarks 477

Resume Method 477

Syntax 477

Properties 477

Remarks 477

ResumeAlarmRead Method 478

Syntax 478

Properties 478

Remarks 478

RetrieveDefinition Method 478

Syntax 478

Properties 478

RetrieveTagGroupVariables Method 478

Syntax 478

Properties 479

Remarks 479

Rotate Method 479

Syntax 479

Properties 479

RunObject Method 479

Syntax 479

Properties 479

Remarks 480

S 480

Save Method 480

DocumentsCollection Object Syntax 480

Properties 480

Remarks 480

Document Object Syntax 480

Properties 480

Remarks 481

Save_TS_List Method 481

Syntax 481

Properties 481

SaveAsSVG Method 481

Syntax 482

Properties 482

SaveToHistoryList Method 482

Syntax 482

Properties 482

Return Value 482

ScrollBack Method 482

Syntax 482

Properties 482

ScrollForward Method 483

Syntax 483

Properties 483

ScrollTimeBack Method 483

Syntax 483

Properties 483

ScrollTimeForward Method 483

Syntax 483

Properties 483

ScrollToPosition Method 483

Syntax 484

Properties 484

Remarks 484

Select Method 484

Syntax 484

Properties 484

SetAlarmBackgroundColor Method 484

Syntax 484

Properties 484

SetAlarmForegroundColor Method 485

Syntax 485

Properties 485

SelectAlarmRow Method 486

Syntax 486

Properties 486

Return Value 486

Remarks 486

SelectAll Method 487

Syntax 487

Properties 487

SelectObject Method 487

Syntax 487

Properties 487

Remarks 487

SendOperatorMessage Method 487

Syntax 487

Properties 487

SendSignedOperatorMessage Method 488

Syntax 488

Properties 488

Return Value 488

SendToBack Method 488

Syntax 488

Properties 488

Remarks 489

SetContinuousUser Method 489

Syntax 489

Properties 489

Return Value 489

SetCurrrentValue Method 489

Syntax 489

Properties 489

SetDispatch Method 490

SetDispid Method 490

SetDuration Method 490

Syntax 490

Properties 490

SetGlobalDuration Method 490

Syntax 490

Properties 490

SetGlobalHistoricalUpdateRate Method 491

Syntax 491

Properties 491

SetGlobalMovingEndTimeToCurrent Method 491

Syntax 491

Properties 491

SetFocusToComboBox Method 492

Syntax 492

Properties 492

SetIndirectionInfo Method 492

SetInterval Method 492

Syntax 492

Properties 492

SetKeyCombination Method 492

Syntax 493

Properties 493

SetLegendMask Method 493

Syntax 493

Properties 493

Remarks 494

SetNumericFormat Method 494

Syntax 494

Properties 494

SetPenDataArray Method 494

Syntax 495

Properties 495

SetPointAt Method 495

Syntax 495

Properties 495

Remarks 495

SetPriorityColor Method 495

Syntax 495

Properties 496

SetProperty Method 496

Syntax 496

Properties 496

Remarks 496

SetScriptWindow Method 496

Syntax 496

Properties 497

Remarks 497

SetSource Method 497

Syntax 497

Properties 497

Remarks 497

SetStatusColor Method 497

Syntax 498

Properties 498
SetStatusFont Method 498
Syntax 499
Properties 499
Remarks 500
SetStringFormat Method 500
Syntax 500
Properties 500
SetTabSelection Method 500
Syntax 500
Properties 500
Return Value 500
SetTimeBeforeNow Method 500
Syntax 501
Properties 501
Remarks 501
SetTimeCursorTime Method 501
Syntax 501
Properties 501
Remarks 501
SetWindowLocation Method 501
Syntax 502
Properties 502
Remarks 502
ShelveAlarm Method 502
Syntax 502
Properties 502
Return Values 503
ShowAnimations Method 503
Syntax 503
Properties 503

ShowBrowseDialog Method 503
- Syntax 503
- Properties 503

ShowColorBox Method 504
- Syntax 504
- Properties 504

ShowColorSelection Method 504
- Syntax 504
- Properties 504
- Remarks 504

ShowCustomPages Method 504
- Syntax 505
- Properties 505

ShowPipePreviewDialog Method 505
- Syntax 505
- Properties 505

ShowTaskWizard Method 505
- Syntax 505
- Properties 505

ShowVBAProcedure Method 505
- Syntax 506
- Properties 506
- Remarks 506

ShowVisualBasicEditor Method 506

Opens the WorkSpace's Visual Basic Editor. 506
- Syntax 506
- Properties 506

SilenceAlarmHorn Method 506
- Syntax 506
- Properties 507
- Remarks 507

SnapObjectsToGrid Method 507
Syntax 507
Properties 507
Remarks 507
SpaceEvenly Method 507
Syntax 507
Properties 507
Remarks 508
StartEvent Method 508
Syntax 508
Properties 508
StartTimer Method 508
Syntax 508
Properties 508
Remarks 508
StickToCursor Method 508
Syntax 509
Properties 509
Remarks 509
StopGlobalPlayBack Method 509
Syntax 509
Properties 509
StopEvent Method 509
Syntax 509
Properties 509
StopTimer Method 509
Syntax 510
Properties 510
Stretch Method 510
Syntax 510
Properties 510

SwitchLanguage Method 510

Syntax 510

Properties 510

SwitchMode Method 511

Syntax 511

Properties 511

Remarks 512

SynchronizeSecurity Method 512

Syntax 512

Properties 512

Remarks 512

T 512

TagGroupSubstitution Method 512

Syntax 513

Properties 513

Remarks 513

TagGroupValue Method 513

Syntax 513

Properties 513

Remarks 513

U-Z 513

UIActivate Method 513

Syntax 513

Properties 514

UIDeActivate Method 514

Syntax 514

Properties 514

Undo Method 514

Syntax 514

Properties 514

Remarks 514

UndoTransaction Method 514

Syntax 514

Properties 515

Settings 515

UndoZoom Method 515

Syntax 515

Properties 515

Remarks 515

UnGroup Method 515

Syntax 515

Properties 516

Remarks 516

UnloadTagGroupFile Method 516

Syntax 516

Properties 516

Remarks 516

UnShelveAlarm Method 516

Syntax 516

Properties 516

Return Values 517

Update_A_Dynamo_By_Name Method 517

Syntax 517

Properties 517

Update_A_Dynamo_By_Name2 Method 518

Syntax 518

Properties 518

Update_A_Dynamo_By_Ref Method 520

Syntax 520

Properties 520

Update_A_Dynamo_By_Ref2 Method 521

Syntax 521

Properties 521
UpdateBackgroundObject Method 522
Syntax 523
Properties 523
UpdateConnectionParameters Method 523
Syntax 523
Properties 523
Remarks 523
UpdateDefinition Method 524
Syntax 524
Properties 524
Remarks 524
UserFormPointToLogical Method 524
Syntax 524
Properties 524
Remarks 525
ValidateSignature Method 525
Syntax 525
Properties 525
Return Value 525
ValidateSignatureAndWriteValue Method 525
Syntax 526
Properties 526
Return Value 527
ValidateSource Method 527
Syntax 527
Properties 527
ValueTimeFromXY Method 527
Syntax 527
Properties 527
Write Method 528

DataItem Object Syntax 528
Properties 528
Group (DataSystem) Object Syntax 528
Properties 528
XYFromValueTime Method 528
Syntax 528
Properties 528
XYHitTest Method 529
Syntax 529
Properties 529
Remarks 529
Zoom Method 529
Syntax 529
Properties 530
ZoomToFit Method 530
Syntax 530
Properties 530
Remarks 530
Event Summary 531
A-B 531
C 531
D 531
E-H 531
I-J 531
K 531
L 531
M-N 532
O 532
P-R 532
S-V 532
W-Z 532

A-D 532

Activated Event 532

Syntax 533

Properties 533

AfterKillFocus Event 533

Syntax 533

Properties 533

AlarmAck Event 533

Syntax 533

Properties 533

AlarmAcknowledged Event 533

Syntax 534

Properties 534

Remarks 534

AlarmListChanged Event 534

Syntax 534

Properties 534

Remarks 534

Click Event 535

Syntax 535

Properties 535

Remarks 535

Close Event 535

Syntax 535

Properties 535

ColorChanged Event 536

Syntax 536

Properties 536

DataChange Event 536

Syntax 536

Properties 536

Event Firing Definition 536
Using the DataChange Event in a Datalink 537
To create your own object: 537
DblClick Event 537
Syntax 537
Properties 537
Remarks 537
Alarm Summary Syntax 537
Properties 538
DeActivated Event 538
Syntax 538
Properties 538
Remarks 538
E-N 538
Edit Event 538
Syntax 538
Properties 538
Remarks 539
EditChange Event 539
Syntax 539
Properties 539
Initialize Event 539
Syntax 539
Properties 539
InitializeConfigure Event 539
Syntax 540
Properties 540
KeyDown Event 540
Syntax 540
Properties 540
Settings 540

Remarks 540

KeyUp Event 541

Syntax 541

Properties 541

Settings 541

Remarks 541

LMouseClick Event 541

Syntax 542

Properties 542

LoadedTagGroup Event 542

Syntax 542

Properties 542

MouseDown Event 542

Syntax 542

Properties 542

Settings 543

Remarks 543

MouseMove Event 543

Syntax 544

Properties 544

Settings 544

Remarks 544

MouseUp Event 544

Syntax 545

Properties 545

Settings 545

Remarks 545

MouseUpOffObject Event 546

Syntax 546

Properties 546

Remarks 546

NewAlarm Event 546
Syntax 547
Properties 547
Remarks 547
O-Z 547
OnChange Event 547
Syntax 547
Properties 547
OnChartFull Event 547
Syntax 548
Properties 548
Remarks 548
OnChartRefresh Event 548
Syntax 548
Properties 548
OnFalse Event 548
Syntax 548
Properties 548
Remarks 549
OnPenSelect Event 549
Syntax 549
Properties 549
OnTimeOut Event 549
Syntax 549
Properties 549
OnTrue Event 550
Syntax 550
Properties 550
Remarks 550
RMouseClick Event 550
Syntax 550

Properties 550

SelectionChanged Event 550

SeverityIncreased Event 551

Syntax 551

Properties 551

UIDeactivate Event 551

WhileFalse Event 551

Syntax 551

Properties 551

Remarks 551

WhileTrue Event 552

Syntax 552

Properties 552

Remarks 552

Subroutine Summary 553

A-B 553

C 553

D 553

E 553

F-K 553

L-N 554

O 554

P-Q 554

R 554

S 554

T 554

U-Z 555

A-F 555

AcknowledgeAllAlarms Subroutine 555

Syntax 555

Properties 555

Remarks 555

AcknowledgeAnAlarm Subroutine 556

Syntax 556

Properties 556

AlarmHornEnabled Subroutine 556

Syntax 556

Properties 556

Return Value 557

AlarmHornEnabledToggle Subroutine 557

Syntax 557

Properties 557

Return Value 557

AlarmHornSilence Subroutine 557

Syntax 557

Properties 558

Remarks 558

CloseDigitalPoint Subroutine 558

Syntax 558

Properties 558

Remarks 559

ClosePicture Subroutine 559

Syntax 559

Properties 559

DisableAlarm Subroutine 559

Syntax 559

Properties 559

EnableAlarm Subroutine 560

Syntax 560

Properties 560

FetchLimits Subroutine 560

Syntax 560

Properties 561
FindDataSource Subroutine 561
Syntax 561
Properties 561
Return Value 561
Remarks 562
FindLocalObject Subroutine 562
Syntax 562
Properties 562
Return Value 562
Remarks 562
G-I 562
GeneratePicture Subroutine 562
Syntax 563
Properties 563
GetAllConnections Subroutine 563
Syntax 563
Properties 563
Return Value 564
GetDecimalSeparator Subroutine 564
Syntax 564
Properties 564
GetFormDynamoColor Subroutine 564
Syntax 564
GetFormNumeric Subroutine 564
Syntax 564
GetFormPushbutton Subroutine 564
Syntax 565
GetFormRamp Subroutine 565
Syntax 565
GetFormSlider Subroutine 565

Syntax 565

GetLocaleInfoA Subroutine 565

Syntax 565

Properties 565

Return Value 565

Remarks 566

GetUserDefaultLCID Subroutine 566

Syntax 566

Return Value 566

Remarks 566

HandleError Subroutine 566

Syntax 566

Properties 566

IsUserFxg Subroutine 566

Syntax 567

Return Value 567

Remarks 567

L-R 567

LocateObject Subroutine 567

Syntax 567

Properties 567

LogIn Subroutine 567

Syntax 568

Properties 568

OffScan Subroutine 568

Syntax 568

Properties 568

OnScan Subroutine 569

Syntax 569

Properties 569

OpenDigitalPoint Subroutine 569

- Syntax 570
- Properties 570
- Remarks 570

OpenPicture Subroutine 570

- Syntax 570
- Properties 570
- Remarks 571

OpenTGDPicture Subroutine 571

- Syntax 572
- Properties 572
- Remarks 572

PictureAlias Subroutine 572

- Syntax 573
- Properties 573

PrintReport Subroutine 573

- Syntax 573
- Properties 573

QuickAdd Subroutine 574

- Syntax 574
- Properties 574
- Return Value 574

RampValue Subroutine 574

- Syntax 575
- Properties 575

ReadValue Subroutine 575

- Syntax 575
- Properties 575
- Return Value 576

RegCloseKey Subroutine 576

- Syntax 576
- Properties 576

Return Value 576

Remarks 576

RegOpenKeyEx Subroutine 576

Syntax 576

Properties 576

Return Value 577

Remarks 577

ReplacePicture Subroutine 577

Syntax 577

Properties 577

Remarks 578

ReplaceTGDPicture Subroutine 578

Syntax 579

Properties 579

Remarks 579

S-Z 579

SetAuto Subroutine 579

Syntax 579

Properties 579

SetManual Subroutine 580

Syntax 580

Properties 580

SetSymbolValues Subroutine 580

Syntax 581

Properties 581

Example 581

ShellExecute Subroutine 581

Syntax 581

Properties 581

Return Value 581

Remarks 581

ToggleDigitalPoint Subroutine 581

Syntax 582

Properties 582

ToggleManual Subroutine 582

Syntax 582

Properties 582

ToggleScan Subroutine 583

Syntax 583

Properties 583

WriteValue Subroutine 583

Syntax 583

Properties 583

Database Functions Summary 585

eda_add_block Function 585

Syntax 585

Properties 585

Return Value 585

Remarks 585

eda_delete_block Function 585

Syntax 586

Properties 586

Return Value 586

Remarks 586

eda_get_pdb_name Function 586

Syntax 586

Properties 586

Return Value 586

Remarks 586

eda_reload_database Function 587

Syntax 587

Properties 587

Return Value 587

Remarks 587

eda_save_database Function 587

Syntax 587

Properties 587

Return Value 588

Remarks 588

eda_type_to_index Function 588

Syntax 588

Properties 588

Return Value 588

Remarks 588

FixGetMyname Function 589

Syntax 589

Properties 589

Return Value 589

Remarks 589

NlsGetText Function 590

Syntax 590

Properties 590

Return Value 590

Remarks 590

Examples 590

A 591

B 591

C 591

D 592

E 593

F 594

G 594

H 596

I-K 596

L 597

M-N 597

O 597

P-Q 598

R 598

S 599

T 601

U 601

V-Z 602

A 602

AboutBox Method Example 602

AckAlarm Method Example 602

AckAlarmPage Method Example 603

AckAlarmPageEx Method Example 603

AckAllAlarms Method Example 603

AcknowledgeAllAlarms Subroutine Example 603

AcknowledgeAnAlarm Subroutine Example 603

ActivateWorkspaceUI Method Example 603

Add Method Example 604

AddDataSet Method Example 604

AddEventHandler Method Example 604

AddImage Method Example 604

AddLegendItem Method Example 605

AddLevel Method Example 605

AddObject Method Example 605

AddPen Method Example 605

AddPictureToStartupList Example 605

AddPoint Method Example 606

AddProcedure Method Example 606

AlarmHornEnabled Example 606

Example 1 606

Example 2 606

AlarmHornEnabledToggle Example 606

AlarmHornSilence Example 607

Align Method Example 607

ApplyProperty Method Example 607

AutoScaleDisplayLimits Method Example 607

B 607

BringToFront Method Example 607

BuildObject Method Example 608

C 608

CanConstruct Method Example 608

CheckAccountExpiration Method Example 608

CheckSecurityEnabled Method Example 609

CheckSyntax Method Example 609

CheckUserApplicationAccess Method Example 609

CheckUserAreaAccess Method Example 610

Clear Method Example 610

ClearUndo Method Example 611

Close Method Example 611

CloseDigitalPoint Subroutine Example 611

ClosePicture Subroutine Example 611

Commit Method Example 611

Connect Method Example 612

ConnectDataSet Method Example 612

ConnectedPropertyCount Method Example 612

Construct Method Example 612

Convert_A_Group_To_A_Dynamo_By_Name Method Example 613

Convert_A_Group_To_A_Dynamo_By_Ref Method Example 613

ConvertPipe Method Example 614

ConvertSecurityAreaNameToNumber Method Example 614

ConvertSecurityAreaNumberToName Method Example 614
ConvertToEnhancedCoordinates Method Example 614
ConvertToOriginalCoordinates Method Example 615
Copy Method Example 615
Coupled_Activate_Workspace_UI Method Example 615
Coupled_DeActivate_Workspace_UI Method Example 615
CopyAsBitmap Method Example 616
CreateDynamoByGrouping Method Example 616
CreateFromDialog Method Example 616
CreateFromProgID Method Example 616
CreateWithMouse Method Example 617
Cut Method Example 617
D 617
DeActivateWorkspaceUI Method Example 617
DefaultView Method Example 617
DelAlarm Method Example 618
DeleteAllAlarms Method Example 618
DeleteAllDataSets Method Example 618
DeleteDataSet Method Example 618
DeleteImage Method Example 618
DeletePen Method Example 618
DeletePoint Method Example 620
DeleteSelectedObjects Method Example 620
DemandFire Method Example 620
DeselectObject Method Example 620
DestroyObject Method Example 620
DisableAlarm Subroutine Example 620
DisableNonSelectionEvents Method Example 621
Disconnect Method Example 621
DisplaysControlPoints Method Example 621
DoesPropertyHaveTargets Method Example 621

DoExtendLines Method Example 621

DoLinestoPolyline Method Example 622

DoMenuCommand Method Example 622

DoTrimLines Method Example 622

DumpProperties Method Example 622

Duplicate Method Example 622

E 623

EditPicture Method Example 623

Enable Method Example 623

EnableAlarm Subroutine Example 623

Enumerate_All_Dynamos Method Example 623

Enumerate_All_Groups Method Example 624

Enumerate_Top_Level_Dynamos Method Example 624

Enumerate_Top_Level_Groups Method Example 625

ESignature Object Example 625

ExchangePenPositions Method Example 626

ExportData Method Example 626

ExportImage Method Example 626

ExportLanguageFile Method Example 627

F 627

FetchLimits Subroutine Example 627

FindAndReplaceDialog Method Example 627

FindDataSource Subroutine Example 627

FindInString Method Example 627

FindLocalObject Subroutine Example 628

FindObject Method Example 628

FindReplaceInObject Method Example 628

FindReplaceInString Method Example 628

FitDocumentToWindow Method Example 629

FitWindowToDocument Method Example 629

FixCheckApplicationAccess Method Example 629

FixCheckApplicationAccessQuiet Method Example 629

FixCheckAreaAccess Method Example 630

FixCheckAreaAccessQuiet Method Example 630

FixCheckSecurityEnabled Method Example 630

FixGetManualAlmDeleteEnabled Method Example 630

FixGetUserInfo Method Example 630

FixLogin Method Example 631

FixLogout Method Example 631

FontProperties Method Example 631

FullView Method Example 631

G 631

GeneratePicture Subroutine Example 631

Get_Last_Prompt_Value Method Example 632

To view this code in context: 633

Get_Last_Result_String Method Example 633

To view this code in context: 634

GetAlarmBackgroundColor Method Example 634

GetAlarmForegroundColor Method Example 634

GetAllConnections Subroutine Example 635

GetBoundRect Method Example 635

GetChartEndTime Method Example 635

GetChartStartTime Method Example 635

GetColHeadings Method Example 636

GetColumnInfo Method Example 636

GetConnectionInformation Method Example 636

GetConnectionParameters Method Example 636

GetContinuousUser Method Example 636

GetCurrentDataSet Method Example 637

GetCurrentValueWithQuality Method Example 637

GetCurrentValue Method Example 638

GetDataSetByPosition Method Example 638

GetDecimalSeparator Subroutine Example 638
GetDeviceRect Method Example 638
GetDuration Method Example 638
GetErrorString Method Example 639
GetEventHandlerIndex Method Example 639
GetFormDynamoColor Subroutine Example 639
GetFormNumeric Subroutine Example 639
GetFormPushbutton Subroutine Example 639
GetFormRamp Subroutine Example 639
GetFormSlider Subroutine Example 640
GetFullname Method Example 640
GetIndirectionInfo Method Example 640
GetInterval Method Example 640
GetNumberOfDataSets Method Example 641
GetLevel Method Example 641
GetLocaleInfoA Subroutine Example 641
GetObjectInfo Method Example 642
GetPenDataArray Method Example 642
GetPenDataArrayEx Method Example 643
GetPointAt Method Example 643
GetPriorityColor Method Example 643
GetProcedureIndex Example 643
GetProperty Method Example 644
GetPropertyAttributes Method Example 644
GetPropertyTargets Method Example 644
GetSelectedAlmExt Method Example 644
GetSelectedNodeTag Method Example 645
GetSelectedRow Method Example 645
GetSelectedRowAlarmInfo Method Example 645
GetSelectedRowsAlarmInfo Method Example 646
GetSelectedUserDefFields Method Example 647

GetSignature Method Example 647
GetSignatureAndWriteValue Method Example 648
GetStatusColor Method Example 649
GetStatusFont Method Example 649
GetTimeBeforeNow Method Example 649
GetTimeCursorInfo Method Example 649
GetUserDefaultLCID Subroutine Example 650
GetUserID Method Example 650
GetWindowLocation Method Example 651
GlobalScrollBackFast Method Example 651
GlobalScrollBackSlow Method Example 651
GlobalScrollForwardFast Method Example 651
GlobalScrollForwardSlow Method Example 652
GlobalTimerApply Method Example 652
Group Method Example 652
H 652
HandleError Subroutine Example 652
HiLoDisplay Method Example 653
I-K 653
ImportToolbar Method Example 653
Initialize Method Example 653
InitializeList Method Example 654
InsertPoint Method Example 654
InteractiveExport Method Example 655
IsColorSelectionVisible Method Example 655
IsConnected Method Example 655
IsEmpty Method Example 655
IsNodeSignEnabled Method Example 655
IsSignatureRequired Method Example 656
IsSignatureRequiredForList Method Example 657
IsUserFxg Subroutine Example 657

Item Method Example 657

L 658

ListEvents Method Example 658

ListMethods Method Example 658

ListProperties Method Example 658

ListWindowsGroupNames Method Example 659

Load_TS_List Method Example 659

LoadImage Method Example 659

LoadTagGroupFile Example 659

LocateObject Subroutine Example 660

LogicalToPercentage Method Example 660

LogicalToUserFormPoint Method Example 660

LogIn Subroutine Example 660

M-N 661

MakeLinesHorizontal Method Example 661

MakeLinesVertical Method Example 661

MakeSameSize Method Example 661

Modify Method Example 661

ModifyColumnLength Method Example 661

Move Method Example 661

NewAlarm Event Example 662

O 662

OffScan Subroutine Example 662

OnScan Subroutine Example 662

Open Method Example 662

Open_QT_Pic Method Example 663

Open_QT_Pic_Ex Method Example 663

Open_TCP_Pic Method Example 663

Open_TCP_Pic_Ex Method Example 663

Open_TS_Pic Method Example 663

Open_TS_Pic_Ex Method Example 664

Open_TS_Pic_Type Method Example 664

Open_TS_Pic_Type_Ex Method Example 665

OpenDigitalPoint Subroutine Example 665

OpenPicture Subroutine Example 666

OpenTGDPicture Subroutine Example 667

P-Q 667

ParseConnectionSource Method Example 667

Paste Method Example 667

PasteSpecial Method Example 667

Pause Method Example 668

PauseAlarmRead Method Example 668

PercentageToLogical Method Example 668

PercentageToPixel Method Example 668

PictureAlias Subroutine Example 669

PixelToPercentage Method Example 669

PrintChart Method Example 669

PrintOut Method Example 669

PrintReport Subroutine Example 670

PromptToChangePassword Method Example 670

QuickAdd Subroutine Example 670

Quit Method Example 671

R 671

RampValue Subroutine Example 671

Read Method Example 671

ReadValue Subroutine Example 672

Refresh Method Example 672

RefreshChartData Method Example 672

RegCloseKey Subroutine Example 672

RegOpenKeyEx Subroutine Example 673

Remove Method Example 674

RemoveAll Method Example 674

RemoveAllLevels Method Example 674
RemoveItem Method Example 675
RemoveLegendItem Method Example 675
RemoveLevel Method Example 675
RemoveObject Method Example 675
ReplacePicture Subroutine Example 675
RemovePictureFromStartupList Example 676
Replace_QT_Pic Method Example 676
Replace_TCP_Pic Method Example 676
Replace_TS_Pic_Type Method Example 676
Replace_TS_Pic Method Example 677
ReplaceDocument Method Example 677
ReplaceInString Method Example 677
ReplaceTGDPicture Subroutine Example 677
ResetChartData Method Example 678
ResetObjectStats Method Example 678
ResetStats Method Example 678
ResetZoom Method Example 678
ResolveTagGroupFile Example 678
Resume Method Example 679
ResumeAlarmRead Method Example 679
RetrieveDefinition Method Example 679
RetrieveTagGroupVariables Method Example 679
Rotate Method Example 680
RunObject Method Example 680
S 680
Save Method Example 680
Save_TS_List Method Example 680
SaveAsSVG Method Example 681
SaveToHistoryList Method Example 681
ScrollBack Method Example 681

ScrollForward Method Example 681
ScrollTimeBack Method Example 681
ScrollTimeForward Method Example 682
ScrollToPosition Method Example 682
Select Method Example 683
SelectAlarmRow Method Example 683
SelectAll Method Example 684
SelectObject Method Example 684
SendOperatorMessage Method Example 684
SendSignedOperatorMessage Method Example 684
SendToBack Method Example 685
SetAlarmBackgroundColor Method Example 685
SetAlarmForegroundColor Method Example 685
SetAuto Subroutine Example 686
SetContinuousUser Method Example 686
SetCurrentValue Method Example 686
SetDispatch Method Example 686
SetDispid Method Example 687
SetDuration Method Example 687
SetFocusToComboBox Method Example 687
SetGlobalMovingEndTimeToCurrent Method Example 687
SetIndirectionInfo Method Example 687
SetInterval Method Example 687
SetManual Subroutine Example 687
SetLegendMask Method Example 688
SetNumericFormat Method Example 688
SetPenDataArray Method Example 688
SetPointAt Method Example 689
SetPriorityColor Method Example 689
SetProperty Method Example 689
SetScriptWindow Method Example 689

SetSource Method Example 690

SetStatusColor Method Example 690

SetStatusFont Method Example 690

SetStringFormat Method Example 690

SetSymbolValues Subroutine Example 690

SetTabSelection Method Example 691

SetTimeBeforeNow Method Example 691

SetTimeCursorTime Method Example 691

SetWindowLocation Method Example 691

ShellExecute Subroutine Example 691

ShelveAlarm Method Example 692

ShowAnimations Method Example 692

ShowBrowseDialog Method Example 692

ShowColorBox Method Example 693

ShowColorSelection Method Example 693

ShowCustomPages Method Example 693

ShowPipePreviewDialog Method Example 693

ShowTaskWizard Method Example 693

ShowVBAProcedure Method Example 693

ShowVisualBasicEditor Method Example 694

AlarmHornSilence Example 694

SnapObjectsToGrid Method Example 694

SpaceEvenly Method Example 694

StartEvent Method Example 694

StartTimer Method Example 695

StickToCursor Method Example 695

StopEvent Method Example 695

StopGlobalPlayBack Method Example 695

StopTimer Method Example 695

Stretch Method Example 696

SwitchLanguage Method Examples 696

SwitchMode Method Example 696

SynchronizeSecurity Method Example 696

To add a VBA reference to SecuritySynchronizerDLL.DLL: 697

T 697

TagGroupSubstitution Method Example 697

TagGroupValue Method Example 697

ToggleDigitalPoint Subroutine Example 698

ToggleManual Subroutine Example 698

ToggleScan Subroutine Example 698

U 698

UIActivate Method Example 698

UIDeActivate Method Example 698

Undo Method Example 698

UndoTransaction Method Example 699

UndoZoom Method Example 699

UnGroup Method Example 699

UnloadTagGroupFile Method Example 699

UnShelveAlarm Method Example 699

Update_A_Dynamo_By_Name Method Example 700

Update_A_Dynamo_By_Name2 Method Example 701

Update_A_Dynamo_By_Ref Method Example 701

To view this code in context: 702

Update_A_Dynamo_By_Ref2 Method Example 702

To view this code in context: 704

UpdateBackgroundObject Method Example 704

UpdateConnectionParameters Method Example 704

UpdateDefinition Method Example 704

UserFormPointToLogical Method Example 705

V-Z 705

ValidateSignature Method Example 705

ValidateSignatureAndWriteValue Method Example 706

ValidateSource Method Example 706

ValueTimeFromXY Method Example 707

WriteValue Subroutine Example 707

Write Method Example 707

XYFromValueTime Method Example 707

XYHitTest Method Example 708

Zoom Method Example 708

ZoomToFit Method Example 708

Index 709

iFIX Automation Reference

The iFIX Automation Reference is intended for integrators and programmers who want to develop applications that access and manipulate information within the iFIX environment through a set of automation interfaces. This help file assumes the reader is proficient in the Microsoft® Visual Basic® programming language.

The following sections provide more details on how to use the objects, properties, methods, events, subroutines, and database functions associated with iFIX:

- Quick Reference
- Object Summary
- Property Summary
- Method Summary
- Event Summary
- Examples
- Subroutine Summary
- Database Functions Summary

Object Summary

The following list contains the iFIX objects that are available to the Automation Interface. For information on non iFIX objects, refer to the appropriate help system.

A

Alarm Summary

Application

Arc

B

Bitmap

C

Chart

Chord

Color Button

ControlContainer

D

DataItem

DataItems

DataLink

DataServer

DataServers

Document

Documents

Dynamo

DynamoSet

E

ESignature

Event

ExpressionEditor

F

FindReplace

FixDataSystem

FixFloatPoint

FixGeometryHelper

FixKeyMacroCollection

FixKeyMacro

Format

G-K

GeneralDataSet

Group

Group (DataSystem)

Groups

HistDatalink

HistogramChart

L-N

Legend

Line

Linear

LineChart

LineConnector

Lines

Lookup

O

Oval

P-Q

Pen

Picture

Pie

Pipe

PipeConnector

Polygon

Polyline

Procedures

R

RealTimeSPCDataSet

Rectangle

RightAngleLineConnector

RoundRectangle

S

Scheduler

ScriptLine

ScriptProcedure

ScriptSource

SecuritySynchronizer

Sources

SPCBarChart

System

T-U

Tag Group

Text

TimeAxis

Timer

ToolbarManager

UserGlobals

UserPreferences

V

ValueAxis

Variable

W-Z

Window

XYChart Object

A-C

Alarm Summary Object

The **Alarm Summary** object provides operators a real-time list of active alarms, and lets operators see and respond to the alarms that the computer receives. Operators can respond by acknowledging, sorting, and filtering alarms as needed.

You can also color-code alarms by alarm status and priority with the **Alarm Summary** object to provide visual cues to your operators.

The **AlarmSummary** object is contained by the **ControlContainer** object and therefore will inherit the Properties and Methods of the **ControlContainer** object.

For more information on the **Alarm Summary** object, see the "Understanding the Alarm Summary Object" chapter in the *Implementing Alarms and Messages* manual.

Application Object

The **Application** object represents the iFIX WorkSpace application. It includes the properties and methods that allow you to access and return top-level objects. For example, the **ActiveDocument** property returns a **Document** object.

If you run schedules in the background, you need to be aware that there are actually two different **Application** objects - one for the WorkSpace and one for **FixBackgroundServer**. The **FixBackgroundServer** application loads and runs **Scheduler** documents. It will fire VBA scripts as the WorkSpace does. However, the **FixBackgroundServer** application does not compete with the single VBA thread of the WorkSpace. For more information on the **FixBackgroundServer** application, see the "Scheduler" chapter in the *Mastering iFIX* manual.

The main difference between the WorkSpace **Application** object and the **FixBackgroundServerApplication** object is that the **FixBackgroundServerApplication** object does not provide access to any windowing or display properties since **FixBackgroundServer** only runs in the background. The properties that are available in **FixBackgroundServer** are:

- **Documents**
- **FixPath**
- **FullName**
- **Name**
- **Object**
- **Owner**
- **Parent**
- **Path**
- **System**
- **Version**

The methods that are available in **FixBackgroundServer** are:

- **BuildObject**
- **ShowVisualBasicEditor**

Arc Object

The **Arc** object is an iFIX shape that can be added to a **Picture**, **DynamoSet** or **UserGlobals** object.

Bitmap Object

The **Bitmap** object is an iFIX shape used to store and display bitmap images that can be added to a **Picture**, **DynamoSet** or **UserGlobals** object.

Chart Object

The **Chart** object holds information describing the real-time or historical data that is being collected and how the data should be displayed in a Standard Chart.

Chord Object

The **Chord** object is an iFIX shape that can be added to a **Picture**, **DynamoSet** or **UserGlobals** object.

ColorButton Object

The **ColorButton** object is an owner drawn push button ocx. It is associated with the color selection dialog, which pops up when the user clicks on the button. The dialog allows the user to select a color from a list of colors and display it on the button face.

The color of the **ColorButton** can be associated with a color property of an object. By passing on the object's dispatch pointer and the dispid of the property to the **ColorButton**, the user can let the **ColorButton** update the property whenever the color is changed.

The **ColorButton** object is contained by the **ControlContainer** object and therefore will inherit the Properties and Methods of the **ControlContainer** object.

ControlContainer Object

The **ControlContainer** object is a graphical shape used to support the embedding of third party ActiveX controls and insertable OLE objects within a picture.

NOTE: If an ActiveX control has the same property or method name as a ControlContainer object property or method, you will only be able to access the ControlContainer's property or method in VB. This occurs because COM does not allow duplicates.

D-F

DataItem Object

The **DataItem** object is a member of the **DataItems** collection. The name of the **DataItem** has to be a data source that exists somewhere in the iFIX data system. Operations available on the **DataItem** are **Read** and **Write**. A **Read** operation will read the current value, timestamp, and quality from the data system and store them in the **Value**, **Timestamp**, and **Quality** variables of the **DataItem**. A **Write** operation will write the passed in value to the data system.

DataItems Object

The **DataItems** object is a user defined collection of **DataItem** objects. **DataItem** objects can be added and removed from this collection. When adding a **DataItem**, the **DataItem** must exist within the iFIX data system or it will not be added to the collection. **DataItem** names must be unique.

DataLink Object

The **Datalink** object is an iFIX shape used to display data that can be added to a **Picture**, **DynamoSet** or **UserGlobals** object.

DataServer Object

The **DataServer** object is an individual data server in the **DataServers** collection. This object contains information about the data server such as the server name, OPC access path, OPC data source, OPC prog ID, OPC server machine name, and default server status.

DataServers Object

The **DataServers** object is a read-only collection of installed data servers in the iFIX data system. This is the same list of servers that can be viewed in the DataServerInstaller application.

Document Object

The **Document** object is the storage for the document, represented by the file name. It is a pointer to the actual document. The type of document varies between Pictures, Schedules, User Globals, Word Documents, Excel Spreadsheet, etc. Use the **Page** object to get to the object model of the underlying type of document. Also be sure to check the **Type** property of the **Document** before using the **Page** object.

Documents Object

The collection of the open documents in the WorkSpace, including Pictures, Schedules, Word Documents, and so forth. The collection also includes the User Global Page but does not include FactoryGlobals.

Dynamo Object

The **Dynamo** object is a type of object that stores re-useable work for the user. The Dynamo Object behaves the same way as a group object, with added functionality. You can make changes to a Master Dynamo, and update the Dynamo instances, while maintaining animation information within the Instances.

DynamoSet Object

The **DynamoSet** object is a type of document (page) that stores re-useable work for the user.

ESignature Object

The **ESignature** object is a COM object that implements the IESignature interface. The **ESignature** object supports electronic signatures for FIX32 data sources, non-FIX32 data sources, and other actions, such as recipe download. You can create a script or another application that prompts the operator to enter an electronic signature using the **ESignature** object. This allows you to:

- Integrate with badge readers and other signing mechanisms.
- Sign for writes to OPC sources.
- Sign for other actions when writing to multiple data points, such as recipe download.
- The object can be instantiated by both VB/VBA and C/C++ code. You can call methods in the IESignature interface to:

- Determine if a tag requires a signature.
- Display the Electronic Signature dialog box.
- Validate a signature without displaying the Electronic Signature dialog box.
- Send a signed operator message to the audit trail.

NOTE: Do not use scripts that use signing from the Scheduler. Signing does not work well from the background task. This is also an important consideration when implementing scripts that call global subroutines.

ESignature methods support the following actions for all data sources:

- Initialize and display the Electronic Signature dialog box, validate signatures, and perform security checks based on information and settings passed in.
- Validate signatures and perform security checks for specific users based on user names, passwords, and settings passed in, without using the Electronic Signature dialog box.
- Send a signed operator message to the audit trail, logging the signature and action.

ESignature methods support the following actions for FIX32 data sources:

- Read electronic signature settings associated with the tag directly from the process database.
- Display the Electronic Signature dialog box based on those settings, validate the signature and perform security checks, write the value to the database or acknowledge an alarm or page of alarms, and send the signed operator message to the audit trail.
- Validate signatures and perform security checks for specific users based on user names, passwords, and settings passed in, without using the Electronic Signature dialog box, and then write the value to the database or acknowledge an alarm or page of alarms and send the signed operator message to the audit trail.

  NOTE: In order to use the enumerations listed for the ESignature Object methods, you must reference the Electronic Signature type library in VBA.

Event Object

The **Event** object contains information about event-based events that are monitored by the **Scheduler**. The **Event** object will fire a VBA event based on the evaluation of the data source. It can be configured to fire when the value is *OnTrue*, *OnFalse*, *WhileTrue*, *WhileFalse*, or *OnChange*.

For more information on the **Event** object, see the "Scheduler" chapter in the *Mastering iFIX* manual.

ExpressionEditor Object

The **ExpressionEditor** object is an OCX that is a helpful tool in building an expression or data source. The user can browse database tag, Picture and Global objects and properties, Historical tags, and OPC data sources. The **ExpressionEditor** also has mathematical buttons to aid in building complex expressions.

The **ExpressionEditor** object is contained by the **ControlContainer** object and therefore will inherit the Properties and Methods of the **ControlContainer** object.

FindReplace Object

The **FindReplace** object allows you to find and replace string properties that are exposed in the object's automation interface. **FindReplace** is capable of operating on a user-specified string, or analyzing a stipulated object and manipulating all matching items found within that object.

NOTE: The **FindReplace** object is not accessible from clients that reside in a process outside the iFIX WorkSpace. Any executables you create using Visual Basic will not support the **FindReplace** feature.

For more information on the FindReplace object and its capabilities, see the "Managing iFIX Nodes" chapter of the *Understanding iFIX* electronic book.

FixDataSystem Object

The **FixDataSystem** object allows users to access data systems in the iFIX architecture through Visual Basic. Currently, the only object available in this OCX is the **FixDataSystem** object.

NOTE: The FindReplace object is not accessible from clients that reside in a process outside the iFIX WorkSpace. Any executables you create using Visual Basic will not support the FindReplace feature.

FixFloatPoint Object

The **FixFloatPoint** object encapsulates the x and y drawing coordinates used by shapes.

The following is an example for declaring and initializing a **FixFloatPoint** object:

```
Dim iPoint as FixFloatPoint
Set iPoint = New FixFloatPoint
iPoint.x = 50.5
iPoint.y = 60.1
Polygon1.AddPoint iPoint
```

NOTE: The user must add a reference to Fix2DGeometry in the References dialog from the Tools menu within VBE.

FixGeometryHelper Object

The **FixGeometryHelper** object a helper object that contains helper methods mostly associated with geometric operations and operations on graphical objects.

FixKeyMacroCollection Object

The **FixKeyMacroCollection** object is a collection of key macros.

FixKeyMacro Object

The **FixKeyMacro** object is a key macro.

Format Object

The **Format** object is an animation object that converts the source data into a string. It contains information about the connection between the data source and the input property of the format object, and the output property of the format object and the animated property of the object being animated. For example, you can use the **Format** object to animate the caption property of a text object.

G-O

GeneralDataSet Object

NOTE: In iFIX 5.5, the GeneralDataSet object replaces both the HistoricalDataSet (used by Historical Datalinks and animations, and the Enhanced Line and XY Charts) and the RealTimeDataSet (used by Enhanced Line and XY Charts) objects.

The **GeneralDataSet** object is a real-time or historical data set type.

For real-time data, this object holds information describing the real-time data set type and how the data should display in a Line Chart or XYChart.

For historical data, it describes how the historical data archived in Proficy Historian should display in a Line Chart or XYChart.

Group Object

The **Group** object holds information describing a number of objects used as a unit. For example, you may have two rectangles and two circles that are used to draw a pump. You can group them together so that they function as a unit allowing you to manipulate all objects in the group as a unit instead of individually.

Group (DataSystem) Object

The **Group** object is a member of the **Groups** collection. This object contains the **DataItems** collection nested within it. Operations that can be performed on a group include **Read** and **Write**. **Group** operations allow the user to do operations on the whole **DataItems** collection within the group.

Groups Object

The **Groups** object is a user defined collection of Group objects. **Group** objects can be added and removed from this collection. **Group** names must be unique.

HistDatalink Object

The **HistDatalink** object is an iFIX shape used to display historical data (from Proficy Historian) that can be added to a Picture, DynamoSet or UserGlobals object. If you insert the HistDatalink into a picture, the Expression Builder only browses historical data that you configured.

IMPORTANT: The refresh rate for the Historical Datalink object is hard-coded at 10 seconds.

HistogramChart Object

The **HistogramChart** object holds information describing data that is being collected from the Histogram (HS) database block and how the data should display.

The Histogram Chart displays a frequency distribution.

Legend Object

The Legend object displays certain information or statistics for a Pen in a Chart.

Line Object

The **Line** object is an iFIX shape that can be added to a Picture, DynamoSet or UserGlobals object.

Linear Object

The **Linear** object is an animation object that converts data from one form into another, effectively performing standard signal conditioning. It contains information about the connection between the data source and the input property of the linear object, and the output property of the linear object and the property of the object being animated. For example, you can use the **Linear** object to animate the tank level of a picture of a tank by mapping it's high and low EGU to reflect the scale of the picture.

LineChart Object

The **LineChart** object holds information describing the real-time or historical data that is being collected and how the data should be displayed.

The Line/MultiLine Chart displays the trend of a variable(s) over time. In this chart, the X-Axis always represents the time. Both real time and historical data are allowed to co-exist within the same Enhanced Chart.

LineConnector Object

The **LineConnector** object is an iFIX connector that is used to join two shapes together.

Lines Object

A collection of the lines of code in one of an object's **Procedures**. Each line of code is a member of the **Lines** collection. The following example sets the string sLine1 to the first line of code in the first Event member of the **Procedures** collection for the object Rect1:

```
Dim sLine1 As String
sLine1 = Rect1.Procedures.Item(1).Lines.Item(1).ProcedureStatement
```

Lookup Object

The **Lookup** object is an animation object that uses the input value to perform either a range comparison or exact match to a table and provides the output value based on the values in the table. It contains information about the connection between the data source and the input property of the lookup object, and the output property of the lookup object and the animated property of the object being animated. For example, you can use the **Lookup** object to blink on a new alarm.

Oval Object

The **Oval** object is an iFIX shape that can be added to a **Picture**, **DynamoSet** or **UserGlobals** object.

P-S

Pen Object

The **Pen** object provides the data source connection and plotting functionality for use in a **Chart** object.

Picture Object

The **Picture** object is a type of document (page) that stores graphical displays.

Pie Object

The **Pie** object is an iFIX shape that can be added to a Picture, DynamoSet or UserGlobals object.

Pipe Object

The **Pipe** object is an iFIX shape that can be added to a Picture, DynamoSet or UserGlobals object.

PipeConnector Object

The **PipeConnector** object is an iFIX connector, shaped like a pipe, that is used to join two other shapes together.

Polygon Object

The **Polygon** object is an iFIX shape that can be added to a Picture, DynamoSet or UserGlobals object.

Polyline Object

The **Polyline** object is an iFIX shape that can be added to a Picture, DynamoSet or UserGlobals object.

Procedures Object

The Procedures object is a collection of VBA procedures that belong to an object, such as a Picture or a Rectangle. Individual procedures can be accessed in the collection by using a one-based index. Standard collection **Add** and **Remove** methods are supported as well as special methods to handle event procedures.

Example:

```
Dim iProc As Object
Set iProc = object.Procedures.Item(1)
```

RealTimeSPCDataSet Object

The **RealTimeSPCDataSet** object holds information describing the real-time SPC data set type and how the data should display in a SPC Bar Chart or Histogram Chart.

Rectangle Object

The **Rectangle** object is an iFIX shape that can be added to a **Picture**, **DynamoSet** or **UserGlobals** object.

RightAngleLineConnector Object

The **RightAngleLineConnector** object is an iFIX connector, formed in a right angle shape, that is used to join two other shapes together.

RoundRectangle Object

The **RoundRectangle** object is an iFIX shape that can be added to a Picture, DynamoSet or UserGlobals object.

Scheduler Object

The **Scheduler** object contains information about iFIX schedules. The Scheduler is the application that displays spreadsheets and contains UI to configure **Timer** and **Event** objects.

For more information on the **Scheduler** object, see the "Scheduler" chapter in the *Mastering iFIX* manual.

ScriptLine Object

The **ScriptLine** object contains a single procedure statement from a script. A **ScriptLine** object is a member of the Lines collection contained within a ScriptProcedure object.

ScriptProcedure Object

The **ScriptProcedure** object contains a collection of ScriptLine objects. A **ScriptProcedure** object is a member of the Procedures collection contained within an object which has event scripts.

ScriptSource Object

The **ScriptSource** object contains a single data source string from a script. A **ScriptSource** object is a member of the Sources collection contained within a ScriptProcedure object. An example of a data source string in a procedure is: Fix32.MyNode.MyTag.F_CV.

SecuritySynchronizer Object

The **SecuritySynchronizer** object is a way to synchronize iFIX security with your Windows security configuration.

You can destroy the **SecurtiySynchronizer** object by setting it equal to Nothing after the SynchronizeSecurity method is called.

Sources Object

The **Sources** object is a collection of ScriptSource objects. The **Sources** object is contained within a ScriptProcedure object. An example of a data source string in a procedure is: Fix32.MyNode.MyTag.F_CV.

SPCBarChart Object

The **SPCBarChart** object holds information describing statistical data that is being collected from a Statistical Data (SD) database block, and how the data should be displayed.

System Object

The **System** object contains information about the iFIX system configuration such as file paths and login data.

T-Z

Tag Group Object

The **Tag Group** object is used to store an array of symbols, substitutions, and optional descriptions. The symbols are placeholders for data sources or text in a picture. Substitutions are alphanumeric strings that replace symbols in a picture. The description is text about the substitution.

Text Object

The **Text** object is an iFIX shape that can be added to a **Picture**, **DynamoSet** or **UserGlobals** object.

TimeAxis Object

The **TimeAxis** object contains time and date display information for use in the **Chart** Object. It also provides the necessary information so that the **Pen** objects within the **Chart** may plot within a given area

Timer Object

The **Timer** object contains information about time-based events that are monitored by the **Scheduler**. The **Timer** object will fire the **OnTimeOut** event based on either a *OneShot*, *Continuous*, *Daily* or *Monthly* time interval.

For more information on the **Timer** object, see the "Scheduler" chapter in the *Mastering iFIX* manual.

ToolbarManager Object

The **ToolbarManager** object holds information describing the iFIX toolbars.

UserGlobals Object

UserGlobals is a special picture that contains user-defined variables, color threshold tables and procedures that are globally available to all pictures. **UserGlobals** is not visible in the WorkSpace display area, but is listed as an entry in the system tree. **UserGlobals** also appears in the Visual Basic Environment (VBE). System tree right-mouse button menu support is provided to create **Variable** objects and color threshold tables, to show the **UserGlobals** property page and to display VBE. **UserGlobals** procedures can be added in VBE.

UserGlobals variables and color threshold tables can be used in connections with other pictures.

As each new picture is created, a reference is automatically added from it to the **UserGlobals** object, allowing its procedures to be accessed. References can be added or deleted manually from within VBE using the Tools|References… menu command.

UserGlobals procedures and declarations should be placed in standard code modules so that they are accessible to procedures in other pictures without using the User.<procedure> scoping syntax. Standard modules can be created from within VBE using the Insert|Module menu command.

If the **UserGlobals** file is not present in the base picture directory, a new empty one is created when the WorkSpace is launched.

UserPreferences Object

The **UserPreferences** object holds information describing the iFIX user preferences.

ValueAxis Object

The **ValueAxis** object contains high and low display information for use in the **Chart** Object. It also provides the necessary information so that the **Pen** objects within the **Chart** may plot within a given area

Variable Object

The **Variable** object is used for storing information to be used elsewhere by your application. The storage must be defined as a particular data type (float, integer, string, etc.) and can be assigned an initial value.

Window Object

The **Window** object contains information about the graphical representation of a document such as its size and location on the screen. The **Window** object has properties that allow the user to set the Top and Left position of a page object.

XYChart Object

The **XYChart** object holds information describing the data that is being collected and how the data should display.

The XY Chart displays the relationship between two data sets. In the XY Chart, the data is refreshed and plotted based on the settings for the X axis.

Property Summary

The following list contains the iFIX properties that are available to the Automation Interface. For information on non iFIX properties, refer to the appropriate help system.

A

Active

ActiveDocument

ActiveWindow

AdvancedGraphics

AlarmHornEnabled

AlarmRefreshInterval

AlarmUserdefField1

AlarmUserdefField2

Alignment

AllowsDrillDown

AllowTimeAxisReset

AllowValueAxisReset

AlwaysOnTop

AnalogError

AnalogErrorTag

AngleUnits

Application

ApplyProperties

AssignedID

Author

Autofetch

AutoMinMaxPaddingX

AutoMinMaxPaddingY

AutoSize

AutoUpdateRate

AverageDataValue

AxisColor

AxisLength

AxisTitle

B

BackDropBackgroundColor

BackDropBackgroundStyle

BackDropBlend

BackDropBorderColor

BackDropBorderStyle

BackDropColor

BackDropFadeColor

BackDropFadeType

BackDropGradAngle

BackDropStyle

BackDropVisible

BackgroundColor

BackgroundEdgeColor

BackgroundEdgeStyle

BackgroundEdgeWidth

BackgroundStyle

BackupSecPath

BarVal

BasePath

BestFitWithCenter Property

BitmapGradientMode

Blend

BlinkEnabled

BlinkRate

BorderTypes

Bottom

BottomCenter

BottomLeft

BottomRight

BottomVisibleRow

BoundRect

ButtonState

ButtonStyle

C

CacheEnabled

Cancel

Caption

Category

Center

CenterOfRotation

CenterPoint

CenterX

CenterY

CharactersPerLine

ChartFontSize

CheckForAlarmListChanged

CheckForNewAlarms

CheckForSeverityIncrease

ClassName

Color

ColorTable

CombinationKey

Comments

CompletionStatus

CompletionStatusTag

ConfirmDataEntry

ConnectionFailed

ConstantLine

ContainedObjects

ContainedSelections

ContextID

ControlOrderIndex

Count

CurrentDataSet

CurrentDataSource

CurrentDate

CurrentDateDay

CurrentDateMonth

CurrentDateYear

CurrentImage

CurrentPen

CurrentPicture

CurrentTime

CurrentTimeHour

CurrentTimeMinute

CurrentTimeSecond

CurrentValue

D

DataEntry

DataItems

DataRefreshInterval

DataServers

DataSetColor

DataShadows

DaylightSavingTime

DaysBeforeNow

DaysOfMonth

DaysOfWeek

Deadband

DecimalDigits

Default

DefaultDataSystem

DefaultExternalDatasourceUpdateRate

DefaultOutputValue

DefaultServer

Description

DeskColor

DigitalError

DigitalErrorTag

DigitsOfPrecision

DisableAutoScale

DisplayLayer

DisplayMilliseconds

DisplayShelvedAlarms

DisplayStatusBar

DisplayString

DisplaySystemTree

DocumentHeight

DocumentHeightEx

DocumentPath

Documents

DocumentWidth

DocumentWidthEx

Domain

DownImageDisplayed

DSDescription

DSLegendAvgerageOverRangeColWidth

DSLegendCurrentValColWidth

DSLegendDescriptionColWidth

DSLegendEngUnitsColWidth

DSLegendHighLimitColWidth

DSLegendHighOverRangeColWidth

DSLegendLowLimitColWidth

DSLegendLowOverRangeColWidth

DSLegendMask

DSLegendQualityColWidth

DSLegendSourceColWidth

DSPosition

Duration

Dynamo_Description

Dynamo_ID

E

EdgeColor

EdgeStyle

EdgeWidth

EditText

ElbowStyle

EnableAcknowledgeAll

EnableAlarmAcknowledge

EnableAlarmDeletion

EnableAsVbaControl

EnableColumnQuickSort

Enabled

EnableEndTime

EnableGlobalEndTime

EnableGlobalScrollPercentage

EnableRightMouseClick

EnableRunTimeConfiguration

EnableTooltips

EndAngle

EndCap

EndPoint

EndTime

EndX

EndY

EngUnits

EnhancedCoordinates

ErrorMode

EventParameter

EventType

ExactMatch

Expandable

ExtendMaxSpace

ExtendType

F

FadeColor

FadeType

FailedSource

FetchDataSetLimits

FetchPenLimits

FileName

FillStyle

FilterString

FixedDate

FixedTime

FixPath

Font

FontName

FontSize

FontStyle

ForceVerticalPoints

ForegroundColor

ForegroundEdgeColor

ForegroundEdgeStyle

ForegroundEdgeWidth

Format

FormatDataType

FullName

FullScreen

FullyQualifedName

G

GlobalDuration

GlobalEndTime

GlobalFastScrollOption

GlobalFastScrollRate

GlobalHistoricalUpdateRate

GlobalMovingEndTime

GlobalMovingStartTime

GlobalOutputToggle

GlobalOutputToggle

GlobalPlayBack

GlobalPlayBackFrameSize

GlobalPlayBackNumberOfFrames

GlobalPlayBackSpeed

GlobalSlowScrollOption

GlobalSlowScrollRate

GlobalStartTime

GlobalTimeSync

GlobalTimerPause

GlobalToggle

Gradient

GradientAngle

GraphBackColor

GraphForeColor

GraphPlusTable

GraphPlusTableMenu

GridEnabled

GridInFront

GridInterval

GridLinesToShow

GridStyle

GridWidth

Groups

H

Height

HelpFile

HelpPath

HideMathFunctionsButton

HiDisplay

HighestDataValue

HighlightEnabled

HighlightedDatasource

HiInValue

HiLimit

HiOutValue

HistMode

HistoricalSampleType

HistUpdateRate

HorizontalFillDirection

HorizontalFillPercentage

HorizontalGridColor

HorizontalGridStyle

HorizontalPosition

HorizontalScaleDirection

HorizontalScalePercentage

I-K

ImageCount

IncludeDataLabels

Index

InitialValue

InputValue

Interval

IntervalMilliseconds

IsDirty

IsInterpolated

IsModifiable

IsSelectable

IsSelected

Item

Justification

Keycode

L

LabelBold

LabelColor

LabelFont

LabelItalic

LabelUnderline

LanguageDesired

Layer

LCL

Left

LeftCenter

Legend

LegendAvgOver

LegendDesc

LegendHeadingLine

LegendHigh

LegendHighOver

LegendInterval

LegendItemColor

LegendLow

LegendLowOver

LegendMode

LegendTag

LegendUnits

LegendUser1

LegendUser10

LegendUser2

LegendUser3

LegendUser4

LegendUser5

LegendUser6

LegendUser7

LegendUser8

LegendUser9

LegendValue

Linear

Lines

LinesofCode

LineType

LockStartTime

LoDisplay

LoginGroup

LoginTimeout

LoginUserFullName

LoginUserName

LoInValue

LoLimit

LoOutValue

LowestDataValue

LWL

M

MainTitle

MainTitleBold

MainTitleFont

MainTitleItalic

MainTitleUnderline

ManualMaxX

ManualMaxY

ManualMinX

ManualMinY

ManualScaleControlX

ManualScaleControlY

MapMode

MarkDataPoints

MarkerChar

MarkerStyle

Master

Max_Dynamo_Desc_Length

MaxCharactersPerLine

MaxLines

MaxPts

MaxXAxisLabels

MonoDeskColor

MonoGraphBackColor

MonoGraphForeColor

MonoShadowColor

MonoTableBackColor

MonoTableForeColor

MonoTextColor

MultipleEGU

MultipleTimes

MyNodeName

N

Name

Next

NlsPath

NoSaveOnClose

NumberOfCharacters

NumberOfHorizontalGridLines

NumberOfItems

NumberOfLines

NumberOfPoints

NumberOfTargets

NumberOfVerticalGridLines

NumOfPoints

NumPointsToGraph

NumHGridLines

NumLabels

NumPts

NumRandomSubsets

NumScrollingSubsets

NumTicks

NumVGridLines

O

Object

OpcAccessPath

OpcDataSource

OpcProgID

OpcServerMachineName

OriginalScreenHeight

OriginalScreenWidth

OriginX

OriginY

OutputValue

Owner

P

Page

Parent

Path

PauseIndicatorBlink

PauseIndicatorColor

PauseWithNewAlarmIndicatorBlink

PauseWithNewAlarmIndicatorColor

PenDescription

PenLineColor

PenLineStyle

PenLineWidth

PenNum

Pens

PenType

PictureDefaultAlwaysOnTop

PictureDefaultBackColor

PictureDefaultHeight

PictureDefaultResizable

PictureDefaultRuntimeVisible

PictureDefaultSystemMenu

PictureDefaultTitlebar

PictureDefaultWidth

PictureHeight

PictureName

PicturePath

PictureWidth

PieType

PlotOnChartRefresh

PlottingMethod

PointType

Previous

PrimarySecPath

ProcedureDeclaration

ProcedureName

Procedures

ProcedureStatement

ProgId

ProjectPath

Property1

Property10

Property2

Property3

Property4

Property5

Property6

Property7

Property8

Property9

Q

Quality

QueueEvents

QuickConfigure

QuickStyle

R

Radius

RawFormat

RecalculateViewport

RefreshRate

RemoveNonWindowsUsers

ResetPercentage

Resizable

ResolveSourceName

Revision

RevisionNumber

Right

RightCenter

RotationAngle

RoundnessX

RoundnessY

RunIndicatorBlink

RunIndicatorColor

RuntimeVisible

S

Saved

SaveThumbnail

ScalesWidth

SchedulePath

ScreenHeight

ScreenWidth

ScrollDirection

ScrollGrid

ScrollItems

ScrollPercentage

SecondaryImageDisplayed

SecurityArea

SelectedDatasource

SelectedFieldName

SelectedNodeName

SelectedShapes

SelectedTagName

SelectionTimeout

SendAlarmMessages

ShadowColor

SharedTableName

ShowAxis

ShowDatabaseTab

ShowDataServersTab

ShowDate

ShowDSLegend

ShowGaps

ShowGlobalsTab

ShowGridLines

ShowHeaders

ShowHistoricalTab

ShowHorizontalGrid

ShowLegend

ShowLine

ShowPicturesTab

ShowRowNumbers

ShowScrollBars

ShowStatusBar

ShowTimeAxis

ShowTimeAxisTitle

ShowTimeCursor

ShowTimeCursorToolTips

ShowTimeStamp

ShowTitle

ShowValueAxis

ShowValueAxisTitle

ShowVerticalGrid

ShowXAxis

ShowYAxis

SmoothingMode

SmoothShapeOption

SmoothShapes

SnapToGrid

SortColumnName

SortOrderAscending

Source

Sources

SourceValidated

SPCChartType

SPCInterval

SPCType

StartAngle

StartCap

StartDateMode

StartDateType

StartPoint

StartTime

StartTimeMode

StartTimeType

StartX

StartY

Status

StatusBar

StatusFontSize

SteppedTrend

StorageMode

StretchMode

StrikeThrough

SubTitle

SubTitleBold

SubTitleFont

SubTitleItalic

SubTitleUnderline

System

SystemMenu

T

TableBackColor

TableFont

TableForeColor

TextColor

Thickness

ThicknessType

Thumbnail

TimeAxis

TimeAxisNumLabels

TimeAxisNumTicks

TimeAxisTitle

TimeBeforeNow

TimeCursorColor

TimeCursorPos

TimeCursorStyle

TimeCursorTooltipColor

Timeout

TimerEnabled

Timestamp

TimeZoneBiasExplicit

TimeZoneBiasRelative

Titlebar

ToggleRate

ToggleSource

Tolerance

ToolbarManager

ToolbarPath

TooltipOption

Top

TopCenter

TopLeft

TopRight

TopVisibleRow

TotalFilteredAlarms

TranslateOnOpen

Transparency

Transparent

TransparentColor

TreatSinglePointsAsLines

TriggerType

TrimMaxLength

TrimType

TruncateTitles

Type

U

UCL

UnacknowledgedAlarmColor

Underline

UniformScale

Units

UpdateOnPropChange

UpdateRate

UseDefaultYAxisSettings

UseDelta

UseDomainSecurity

UseDSLimits

UseLocalSecurity

UseMarker

UserDef1ColumnName

UserDef2ColumnName

UserPreferences

UseUnacknowledgedAlarmColor

UWL

V

Value

ValueAxis

ValueAxisNumLabels

ValueAxisNumTicks

ValueAxisTitle

VariableType

Version

VerticalFillDirection

VerticalFillPercentage

VerticalGridColor

VerticalGridStyle

VerticalPosition

VerticalScaleDirection

VerticalScalePercentage

ViewingStyle

ViewportHeight

ViewportLeft

ViewportTop

ViewportWidth

Visible

VisibleUnacknowledgedAlarms

W-Y

WholeDigits

Width

WindowHeightPercentage

WindowLeftPercentage

WindowName

WindowState

WindowTopPercentage

WindowWidthPercentage

WizardName

X

XAxisDatasetPosition

XAxisLabel

XAxisScaleControl

XAxisType

Y

YAxesStyle

YAxisAlwaysVisible

YAxisLabel

YAxisLongTicks

YAxisScaleControl

YAxisTitle

Z

Zoom

ZoomDirection

ZoomType

A

Active Property

Specifies whether the specified object has focus.

Syntax

*object.***Active** [= *Boolean*]

Properties

The **Active** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the document has focus.

Settings

The settings for *Boolean* are:

Value	Description
True	The document is active.
False	The document is not active.

Remarks

Call this property to force the selection of a document through scripting. Also, if you open a document as hidden, setting the document's active state to **True** makes the document visible.

ActiveDocument Property

Returns the currently active document in the WorkSpace.

Syntax

*object.***ActiveDocument**

Properties

The **ActiveDocument** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ActiveDocument is a read-only property of type *Object*.

ActiveDocument and document objects accessed using **Application.Documents** hold objects for ActiveX Documents. These objects are called **FixFileLink** objects. To access the actual user document (picture, schedule, dynamo set), the **FixFileLink** object contains a **Page** property that is the OLE object for the actual user document.

ActiveWindow Property

Returns the currently active window in the WorkSpace or the **Document** object.

Syntax

*object.***ActiveWindow**

Properties

The **ActiveWindow** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ActiveWindow is a read-only property of type *Object*.

When more than one window is open in the WorkSpace, the active window property setting is the window with the focus. If no documents are open, **ActiveWindow** returns nothing.

The active window is the window that appears in the foreground with a highlighted title bar.

The **ActiveWindow** property is useful for accessing the currently active window object.

AdvancedGraphics Property

Enables graphical enhancements such as gradients, alpha blending, and anti-aliasing for text and graphics in an Enhanced Chart. Disable this option to increase performance.

Syntax

*object.***AdvancedGraphics** [= *Boolean*]

Properties

The **AdvancedGraphics** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether graphical enhancements are enabled.

Settings

The settings for *Boolean* are:

Value	Description
True	Graphical enhancements enabled.
False	Graphical enhancements disabled. (Default)

Remarks

AdvancedGraphics is a read-only property.

AlarmHornEnabled Property

Gets or sets the alarm horn enabled status.

Syntax

AlarmHornEnabled *([blnNewValue], [intErrorMode])*

Properties

The **AlarmHornEnabled** property syntax has these parts:

Part	Description

blnNewValue	Boolean. (Optional). The value to which you want to set the alarm horn enable property.
	TRUE = enabled
	FALSE = disabled

IntErrMode	Integer. (Optional). The error mode. 0 (default) – Errors are displayed in the form of a message box. 1 – Errors are not handled so that they can be handled in the calling routine. 2 – Errors are dispatched to the alarm destinations using SendOperatorMessage.

Return Value

Boolean. The status of the AlarmHornEnable after the call is completed.

True = The horn sounds on any new alarm.

AlarmRefreshInterval Property

Specifies the rate at which the **Alarm Summary** object checks for a change in the list of alarms.

Syntax

*object.***AlarmRefreshInterval** [= *Single*]

Properties

The **AlarmRefreshInterval** property syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
Single	The rate, in seconds, that the **Alarm Summary** will update the list of alarms. The valid values are 0.1 to 300. The default is 0.5 seconds.

AlarmUserdefField1 Property

Specifies the value for the user defined field 1 column.

Syntax

*object.***AlarmUserdefField1** [= *String*]

Properties

The **AlarmUserdefField1** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The value for the user defined field 1 column.

AlarmUserdefField2 Property

Specifies the value for the user defined field 2 column.

Syntax

*object.***AlarmUserdefField2** [= *String*]

Properties

The **AlarmUserdefField2** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The value for the user defined field 2 column.

Alignment Property

The alignment property defines the alignment of the text string within the text's bounding rectangle.

Syntax

*object.***Alignment** [= *enumTextAlign*]

Properties

The **Alignment** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
EnumTextAlign	Text position.

Settings

The settings for *enumTextAlign* are:

Constant	Value	Description
Left	0	Align text to left of its bounding rectangle.
Center	1	Center the text.
Right	2	Align text to right of its bounding rectangle.

Remarks

Use this property to define how a **Text** object aligns within it's bounding rectangle. This is useful for aligning a column of numbers, or bar graph labels. For example, if you want to align the decimal points in a column of data links, you can set the alignment property to “Right alignment”. Numbers with similar precision numbers will align properly regardless of the size of the value to the left of the decimal.

This property can also be used to control the behavior of text during a scale operation. Setting the alignment to center for text in a dynamo or group keeps an equal relationship between itself and other objects.

AllowsDrillDown Property

Specifies whether or not the user can drill into the **Group** object.

Syntax

object.**AllowsDrillDown** [= *Boolean*]

Properties

The **AllowsDrillDown** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the **Group** object allows manipulation of it's contained objects.

Settings

The settings for *Boolean* are:

Value	Description
True	The **Group** allows drill down. (Default)
False	The **Group** does not allow drill down.

Remarks

This property could be used to prevent users from inadvertently change the visual representation of an object or from modifying a group's internal animation properties.

AllowTimeAxisReset Property

Specifies how the time axis limits of a **Pen** are reset when you right-click a chart.

Syntax

*object.***AllowTimeAxisReset** [= *Boolean*]

Properties

The **AllowTimeAxisReset** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether or not to reset the time axis limits in a chart to Start Time and End Time pen properties.

Settings

The settings for *Boolean* are:

Value	Description
True	Resets the time axis limits of the chart to the StartTime and EndTime properties of a chart's pen.
False	Resets the time axis limits of the chart to the FixedTime and FixedData properties or the DaysBeforeNow and TimeBeforeNow properties.

AllowValueAxisReset Property

Specifies how the value axis limits of a **Pen** are reset when you right-click a chart.

Syntax

*object.***AllowValueAxisReset** [= *Boolean*]

Properties

The **AllowValueAxisReset** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether or not to reset the value axis limits in a chart to HiLimit and LoLimit pen properties.

Settings

The settings for *Boolean* are:

Value	Description
True	Resets the value axis limits of the chart to the HiLimit and LoLimit properties of a chart's pen.
False	Resets the value axis limits of the chart to the EGU limits of the tag associated with the pen.

AlwaysOnTop Property

Specifies whether the specified document is to be on top of any other window owned by the WorkSpace application. A document that has this property is always on top.

Syntax

*object.***AlwaysOnTop** [= *Boolean*]

Properties

The **AlwaysOnTop** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether or not the document is always on top.

Settings

The settings for *Boolean* are:

Value	Description
True	The document is always on top.
False	The document is not always on top. (Default)

Remarks

This property is vital for creating pop up subpictures that will stay on top of the main display, even if the user selects and activates the main window. By convention, the display that opens a subpicture is responsible for closing the display if that display is closed.

AnalogError Property

Retrieves the last value written to the analog error tag by the SecuritySynchronizer object. The value is represented as a string.

Syntax

*object.***AnalogError**

Properties

The **AnalogError** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

AnalogError is a read-only property. It is only updated by the SecuritySynchronizer object at the end of the security synchronization process.

AnalogErrorTag Property

Sets or retrieves the analog iFIX database tag and floating point field to which the last, most severe, error code is written when the security synchronization process completes.

Syntax

*object.***AnalogErrorTag** [= *String*]

Properties

The **AnalogErrorTag** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The Node.Tag.Field to be set or retrieved.

Remarks

AnalogErrorTag corresponds to the /E command line parameter of the Security Synchronizer application.

AngleUnits Property

Specifies whether any property that requires an angle is measured in degrees or radians.

Syntax

*object.***AngleUnits** [= *enumAngleUnits*]

Properties

The **AngleUnits** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
EnumAngleUnits	Specifies the angle measurement units.

Settings

The settings for *enumAngleUnits* are:

Constant	Value	Description
Degrees	0	Angle is measured in degrees. (Default)
Radians	1	Angle is measured in radians.

Remarks

Not all objects contain the **RotationAngle** property. For example, the **Oval** object does not have a **RotationAngle** property.
The **StartAngle** and **EndAngle** properties are affected by the **AngleUnits** property.

Application Property

Returns a pointer to the **Application** object.

Syntax

*object.***Application**

Properties

The **Application** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Application is a read-only property of type *Object*.

ApplyProperties Property

Determines how to apply properties to the images in the list.

Syntax

*object.***ApplyProperties** [= *enumApplyProperties*]

Properties

The **ApplyProperties** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumApplyProperties	Specifies how properties are applied to the list of images.

Settings

The settings for *enumApplyProperties* are:

Part	Value	Description
ApplyPropertiesToCurrent	0	Apply properties to current image. (Default)
ApplyPropertiesToAll	1	Apply properties to all images.

AssignedID Property

Retrieves the ID assigned to the dataset.

Syntax

*object.***AssignedID**

Properties

The **AssignedID** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Read-only property

Author Property

Specifies the author of the specified document.

Syntax

*object.***Author** [= *String*]

Properties

The **Author** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The author of the document.

Remarks

The default **Author** is the name of the currently logged-in user, in Windows.

Autofetch Property

Specifies whether to automatically fetch the low and high EGU limits on run-time initialization.

Syntax

*object.***Autofetch** [= *Boolean*]

Properties

The **Autofetch** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether or not to automatically fetch EGU limits.

Settings

The settings for *Boolean* are:

Value	Description
True	Fetch the limits.
False	Do not fetch the limits. (Default)

Remarks

Enabling the **Autofetch** property fetches the EGU limits of the data source at run-time. If the EGU limits of a data source change, then this field allows the animation or chart to detect this change and update it's internal input range limits. This is useful if a data source needs to be modified at run-time.

AutoMinMaxPaddingX Property

Allows iFIX to automatically scale the X-axis so that all data points are within the visible area of the XY Chart. This property only applies to XY Enhanced Charts.

Syntax

*object.***AutoMinMaxPaddingX** [= *Long*]

Properties

The **AutoMinMaxPaddingX** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Sets the percentage added above and below the automatically determined range for the X-axis.

AutoMinMaxPaddingY Property

Allows iFIX to automatically scale the Y-axis so that all data points are within the visible area of the Enhanced Chart.

Syntax

*object.***AutoMinMaxPaddingY** [= *Long*]

Properties

The **AutoMinMaxPaddingY** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Sets the percentage added above and below the automatically determined range for the Y-axis.

AutoSize Property

AutoSize controls whether the **Text** object automatically picks a new font that will fit into the bounding rectangle as the **Caption** changes. If **AutoSize** is set to **True**, the size of the bounding rectangle is recalculated. If set to **False**, the **FontSize** of the **Text** object is recalculated.

Syntax

*object.***AutoSize** [= *Boolean*]

Properties

The **AutoSize** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether or not to automatically update the object's size to fit its contents.

Settings

The settings for *Boolean* are:

Value	Description
True	Automatically resize the object to display its entire contents. (Default)
False	Maintain the size of the object constant, adjust the **FontSize** to fit inside the bounding rectangle.

Remarks

For controls with captions, the **AutoSize** property specifies whether the control automatically adjusts to display the entire caption. For controls without captions, this property specifies whether the control automatically adjusts to display the information stored in the control. In a ComboBox, for example, setting **AutoSize** to **True** automatically sets the width of the display area to match the length of the current text. For a single-line text box, setting **AutoSize** to **True** automatically sets the width of the display area to the length of the text in the text box.

For a multi-line text box that does not contain text, setting **AutoSize** to **True** automatically displays the text as a column. The width of the text column is set to accommodate the widest letter of that font size. The height of the text column is set to display the entire text of the TextBox. For a multi-line text box that contains text, setting **AutoSize** to **True** automatically enlarges the TextBox vertically to display the entire text. The width of the TextBox does not change.

The behavior of the object by changing the **AutoSize** property is dependent upon the **ScalesWidth** property. The following table illustrates the dependencies:

Value	ScalesWidth Value	Resulting Behavior
True	True	Bounding rectangle is recalculated to fit the text.
False	True	The **FontSize** is recalculated to fit the bounding rectangle.
True	False	Bounding rectangle is recalculated to fit the text.
False	False	The text is clipped.

NOTE: If you manually change the size of a control while **AutoSize** is **True**, the manual change overrides the size previously set by **AutoSize**.

AutoUpdateRate Property

Specifies the automatic historical update rate of the chart.

Syntax

*object.***AutoUpdateRate** [=*Single*]

Properties

The **AutoUpdateRate** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Single	The autoupdate rate of the chart.

Remarks

AutoUpdateRate is 0 by default. When the value is set to 0, autoupdate is disabled.

AverageDataValue Property

Returns the average of the currently displayed values in the **Chart** for the specified **Pen**.

Syntax

*object.***AverageDataValue** [= *Double*]

Properties

The **AverageDataValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The average of the currently displayed values of the pen.

Remarks

This property is valid for both real-time and historical pen configurations. (See **HighestDataValue** and **LowestDataValue**)

AxisColor Property

Specifies the axis color.

Syntax

*object.***AxisColor** [= *Long*]

Properties

The **AxisColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the axis color.

AxisLength Property

Returns the axis length in postscript points or logical units.

Syntax

*object.***AxisLength**

Properties

The **AxisLength** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

AxisLength is a read-only property of type *Double*.

AxisTitle Property

Specifies the axis title caption.

Syntax

*object.***AxisTitle** [= *String*]

Properties

The **AxisTitle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The caption of the axis' title.

B

BackDropBackgroundColor Property

Defines the background color of a Group object when the background color pattern requires two colors (hatched, diagnal, etc.).

Syntax

*object.***BackDropBackgroundColor** [= *Long*]

Properties

The **BackDropBackgroundColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the group's background color.

Remarks

The group's background and foreground color properties apply to the children contained in the group. The group itself is a shape and can have it's own bounding rectangle color and style. By default this is transparent. Enabling the BackDropVisible property activates the group's BackDrop color properties. The group's bounding rectangle fill color is defined by the BackDropColor and **BackDropBackgroundColor**. The **BackDropColor** can be considered the foreground color of the BackDrop fill area, therefore, is used for a solid style, and is the line color for the hatched patterns. For non-solid styles, the **BackDropBackgroundColor** is the background fill area.

BackDropBackgroundStyle Property

Defines the background style for the group's bounding rectangle.

Syntax

*object.***BackDropBackgroundStyle** [= *enumBackgroundStyle*]

Properties

The **BackDropBackgroundStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumBackgroundStyle	Specifies the style to display for the group's background.

Settings

The settings for *enumBackgroundStyle* are:

Constant	Value	Description
Transparent	1	The background style is transparent.
Opaque	2	The background style is opaque. (Default)

BackDropBlend Property

Specifies the percentage of **BackDropFadeColor** to blend with the group's **BackdropColor**.

Syntax

*object.***BackDropBlend** [=*Double*]

Properties

The **BackDropBlend** propertysyntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The percentage of blend to use in the gradient fill.

Remarks

The **BackDropBlend** property is useful only when the group's FillStyle is set to *FillStyleGradient*.

BackDropBorderColor Property

Defines the back drop border color for the **Group** object's bounding rectangle.

Syntax

*object.***BackDropBorderColor** [= *Long*]

Properties

The **BackDropBorderColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the back drop border color.

BackDropBorderStyle Property

Sets the border style of the Group object's bounding rectangle.

Syntax

*object.***BackDropBorderStyle** [= *enumEdgeStyle*]

Properties

The **BackDropBorderStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumEdgeStyle	The style to display for the edge of the backdrop.

Settings

The settings for *enumEdgeStyle* are:

Constant	Value	Description
EdgeStyleSolid	0	Solid.
EdgeStyleDash	1	Dash.
EdgeStyleDot	2	Dot.
EdgeStyleDashDot	3	Dash-Dot.
EdgeStyleDashDotDot	4	Dash-Dot-Dot.
EdgeStyleNone	5	No border.
EdgeStyleInsideFrame	6	Inside Frame.

BackDropColor Property

Defines the back drop color of a group of objects.

Syntax

*object.***BackDropColor** [= *Long*]

Properties

The **BackDropColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the group's back drop color.

Remarks

The group's background and foreground color properties apply to the children contained in the group. The group itself is a shape and can have it's own bounding rectangle color and style. By default this is transparent. Enabling the **BackDropVisible** property activates the group's BackDrop color properties. The group's bounding rectangle fill color is defined by the **BackDropColor** and **BackDropBackgroundColor**. The **BackDropColor** can be considered the foreground color of the BackDrop fill area, therefore, is used for a solid style, and is the line color for the hatched patterns. For non-solid styles, the **BackDropBackgroundColor** is the background fill area.

BackdropFadeColor Property

Specifies the fade color of a group when the **FillStyle** pattern is *Gradient*.

Syntax

*object.***BackdropFadeColor** [=*Long*]

Properties

The **BackdropFadeColor** propertysyntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the group's fade color.

BackdropFadeType Property

Specifies the kind of fade used for the group's gradient fill.

Syntax

*object.***BackdropFadeType** [=*enumFadeType*]

Properties

The **BackdropFadeType** propertysyntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumFadeType	The pattern to display.

Settings

The settings for *enumFadeType* are:

Constant	Value	Description
Linear	0	Linear
Reflected	1	Reflected
Radial	2	Radial
Concentric	3	Concentric

Remarks

The **GradientAngle** property is useful when the **FadeType** is either *Linear* or *Reflected*.

BackDropGradAngle Property

Specifies the angle (in radians or degrees) of the group's gradient fill.

Syntax

*object.***BackDropGradAngle** [=*Double*]

Properties

The **BackDropGradAngle** propertysyntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The angle of the group's gradient fill.

Remarks

In Configuration mode, **BackDropGradAngle** changes as you rotate the group.

The units to be used when creating the angle is specified as either degrees or radians, depending on the value of the **AngleUnits** property.

BackDropStyle Property

Defines the **Group** object's back drop fill style.

Syntax

*object.***BackDropStyle** [= *enumFillStyle*]

Properties

The **BackDropStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumFillStyle	Specifies the pattern style to display for the back drop.

Settings

The settings for *enumFillStyle* are:

Constant	Value	Description
FillStyleSolid	0	Solid.
FillStyleHollow	1	Hollow.
FillStyleHorizontal	2	Horizontal.
FillStyleVertical	3	Vertical.
FillStyleDownDiagonal	4	Downward diagonal.
FillStyleUpDiagonal	5	Upward diagonal.
FillStyleCrossHatch	6	Crosshatch.
FillStyleDiagonalCrossHatch	7	Diagonal crosshatch.

BackDropVisible Property

Enables the Group object's back drop properties.

Syntax

*object.***BackDropVisible** [= *Boolean*]

Properties

The **BackDropVisible** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether or not the back drop properties are enabled.

Settings

The settings for *Boolean* are:

Value	Description
True	Back drop is visible.
False	Back drop is not visible.

BackgroundColor Property

Specifies the background color of an object.

Syntax

*object.***BackgroundColor** [= *Long*]

Properties

The **BackgroundColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the object's background color.

BackgroundEdgeColor Property

Specifies the Chart object's background edge color.

Syntax

*object.***BackgroundEdgeColor** [= *Long*]

Properties

The **BackgroundEdgeColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the background edge color.

BackgroundEdgeStyle Property

Specifies the Chart object's background edge style.

Syntax

*object.***BackgroundEdgeStyle** [= *enumEdgeStyle*]

Properties

The **BackgroundEdgeStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumEdgeStyle	The style to display for the background edge.

Settings

The settings for *enumEdgeStyle* are:

Constant	Value	Description

EdgeStyleSolid	0	Solid.
EdgeStyleDash	1	Dash.
EdgeStyleDot	2	Dot.
EdgeStyleDashDot	3	Dash-Dot.
EdgeStyleDashDotDot	4	Dash-Dot-Dot.
EdgeStyleNone	5	No border.
EdgeStyleInsideFrame	6	Inside Frame.

BackgroundEdgeWidth Property

Specifies the Chart object's background edge width.

Syntax

*object.***BackgroundEdgeWidth** [= *Long*]

Properties

The **BackgroundEdgeWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The value to specify the width of the background edge.

BackgroundStyle Property

Specifies whether the specified object's background style is opaque or transparent.

Syntax

*object.***BackgroundStyle** [= *enumBackgroundStyle*]

Properties

The **BackgroundStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumBackgroundStyle	The style of the object's background.

Settings

The settings for *enumBackgroundStyle* are:

Constant	Value	Description
Transparent	1	The background style is transparent.

Opaque	2	The background style is opaque. (Default)

BackupSecPath Property

Returns the backup security file path for the specified document.

Syntax

*object.***BackupSecPath** [= *String*]

Properties

The **BackupSecPath** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The backup security file path.

Remarks

BackupSecPath is a read-only property of type *String*.

BarVal Property

Sets the bar value for the real-time SPC data set in the Enhanced Chart.

Syntax

*object.***BarVal** [= *Double*]

Properties

The **BarVal** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The bar value for the real-time SPC data set in the RealTimeSPCDataSet Object.

BasePath Property

Returns the iFIX base path as defined in the System Configuration Utility (SCU). This is typically the main directory where the product is installed.

Syntax

*object.***BasePath**

Properties

The **BasePath** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

BasePath is a read-only property of type *String*.

BestFitWithCenter Property

When the global setting named "ZoomToFitFromCenter" is disabled in the FixUserPreferences.ini file, use **BestFitWithCenter** to override the global setting on a picture.

Syntax

object.**BestFitWithCenter** [= *Boolean*]

Properties

The **BestFitWithCenter** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to disable the ZoomToFitFromCenter setting in the FixUserPreferences.ini file at the picture-level.

Settings

The settings for *Boolean* are:

Value	Description
True	Enables the setting. (Default)
False	Does not enable the setting.

Remarks

The global setting named "ZoomToFitFromCenter" is available for use in the FixUserPreferences.ini file. Use this global setting to disable centering when ZoomToFit is applied. (By default, ZoomToFit centers a picture.) Add ZoomToFitFromCenter=0 to the [AppDesignPreferences] in the FixUserPreferences.ini file (located at the "LOCAL" folder of the iFIX install) to disable centering when ZoomToFit is enabled.

BitmapGradientMode Property

Allows you to apply a bitmap or gradient style to the background of an Enhanced Chart.

Syntax

*object.*__BitmapGradientMode__ [= *Boolean*]

Properties

The **BitmapGradientMode** property syntax has thse parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether or not a bitmap or gradient style is applied to the background of the HistogramChart, LineChart, or SPCBarChart Object.

Settings

The settings for *Boolean* are:

Value	Description
True	Bitmap or gradient style enabled. (Default)
False	Bitmap or gradient style disabled.

Blend Property

Specifies the percentage of the **FadeColor** to blend with the object's **ForegroundColor**. In the case of a chart object, specifies the percentage of the FadeColor to blend with the object's **BackgroundColor**.

Syntax

*object.*__Blend__ [=*Double*]

Properties

The **Blend property** syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The percentage of blend to use in the gradient fill.

Remarks

The **Blend** property is useful only when the object's **FillStyle** is set to *FillStyleGradient*.

BlinkEnabled Property

Specifies whether unacknowledged alarms in the **Alarm Summary** object blink. If blinking is enabled, unacknowledged alarms blink at the rate set in the **BlinkRate** property.

Syntax

*object.*__BlinkEnabled__ [= *Boolean*]

Properties

The **BlinkEnabled** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether or not unacknowledged alarms blink.

Settings

The settings for *Boolean* are:

Value	Description
True	Unacknowledged alarms blink
False	Unacknowledged alarms do not blink. (Default)

BlinkRate Property

Specifies the rate at which the **Alarm Summary** spreadsheet row blinks.

Syntax

object.**BlinkRate** [= *Integer*]

Properties

The **BlinkRate** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The rate, in seconds, that the speadsheet row blinks. The valid values are 1 - 10. The default is 1 second.

Remarks

The **BlinkRate** property only takes effect if the **BlinkEnabled** property is set to **True**.

BorderTypes Property

Sets the border type for the Enhanced Chart.

Syntax

object.**BorderTypes***enumBorderTypes*

Properties

The **BorderTypes** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumBorderTypes	Integer. Specifies the border type: **Valid entries:** 0 – DropShadow 1 – SingleLine 2 – NoBorder 3 – InSet

Bottom Property

Returns the value of the bottom edge of the shape's bounding rectangle.

Syntax

*object.***Bottom**

Properties

The **Bottom** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Bottom is a read-only property of type *Double*.

BottomCenter Property

Returns the value of the bottom center point of the shape's bounding rectangle as a **FixFloatPoint**.

Syntax

*object.***BottomCenter**

Properties

The **BottomCenter** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

BottomCenter is a read-only property of type *Object*.

BottomLeft Property

Returns the value of the bottom left point of the shape's bounding rectangle as a **FixFloatPoint**.

Syntax

*object.***BottomLeft**

Properties

The **BottomLeft** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

BottomLeft is a read-only property of type *Object*.

BottomRight Property

Returns the value of the bottom right point of the shape's bounding rectangle as a **FixFloatPoint**.

Syntax

*object.***BottomRight**

Properties

The **BottomRight** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

BottomRight is a read-only property of type *Object*.

BottomVisibleRow Property

Specifies the last visible row in the **Alarm Summary** object's spreadsheet.

Syntax

*object.***BottomVisibleRow**

Properties

The **BottomVisibleRow** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

You can use the value of **BottomVisibleRow** to compute the visible page size.

BoundRect Property

Returns the top left and bottom right values of the shape's bounding rectangle.

Syntax

*object.***BoundRect**

Properties

The **BoundRect** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

BoundRect is a read-only property of type *Object*.

ButtonState Property

Indicates whether or not the **Bitmap** is pushed in the Run-time environment.

Syntax

*object.***ButtonState** [= *enumButtonState*]

Properties

The **ButtonState** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumButtonState	Whether or not the button is pushed.

Settings

The settings for *enumButtonState* are:

Constant	Value	Description
ButtonStateUp	0	Button is not pushed.
ButtonStateDown	1	Button is pushed.

Remarks

The **ButtonState** property only applies to multi-state bitmaps.

ButtonStyle Property

Specifies the behavior style of the Bitmap when used as a push button.

Syntax

*object.***ButtonStyle** [= *enumButtonStyle*]

Properties

The **ButtonStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumButtonStyle	The behavior style.

Settings

The settings for *enumButtonStyle* are:

Constant	Value	Description
BitmapButtonNone	0	None.
BitmapButtonPush	1	Push button.
BitmapButtonMultiState	2	Multi-state button.

Remarks

This property can be used to make a three dimensional button.

C

CacheEnabled Property

Specifies whether caching is enabled for the picture object.

Syntax

*object.***CacheEnabled** [= *Boolean*]

Properties

The **CacheEnabled** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether caching is enabled for the particular picture.

Settings

The settings for *Boolean* are:

Value	Description
True	Once you have closed a picture in run mode, the next time you open the picture in run mode it will be read from memory cache, instead of from disk, if picture caching is enabled globally. This is the default.
False	Once you have closed a picture in run mode, the next time you open the picture in run mode it will be read from disk, instead of from memory cache, whether or not picture caching is enabled globally.

Remarks

This object property directly affects the Allow this picture to be cached option in the Create Picture wizard. However, picture caching will only truly occur if global picture caching is also enabled from the User Preferences dialog box. This option is available in three places:

- On the Attributes page of the Modify Configuration dialog box
- On the Define Custom Picture Attributes page of the Create Picture Wizard dialog box.
- In the Edit Picture dialog box, as Disable caching for this picture.

Cancel Property

Specifies whether the specified control is the Cancel button in a picture. This control can be the push-button control or any control that behaves like a button.

Syntax

object.**Cancel** [= *Boolean*]

Properties

The **Cancel** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the control is the cancel button.

Settings

The settings for *Boolean* are:

Value	Description
True	This control is the Cancel button of the picture in the Run-time environment.
False	The control is not a cancel button for the picture. (Default)

Remarks

Only one control in a picture can be the Cancel button. When the **Cancel** property is set to **True** for one control, it is automatically set to **False** for all other controls in the picture. When a control's **Cancel** property setting is **True** and the picture is the active picture, the user can choose the control by clicking it, pressing the ESC key, or pressing ENTER when the button has the focus.

The **Cancel** property of a control can be set to **True** only if the control is a pushbutton control or any control that behaves like a button i.e., marked with OLEMISC_ACTSLIKEBUTTON flag.

Caption Property

Specifies the caption text of the Text and Datalink objects and/or the caption of the Window or Application object.

Syntax

*object.***Caption** [= *String*]

Properties

The **Caption** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The string displayed as the caption of the object.

Remarks

To change the caption of a Picture or **Window** object, the user would do the following:
Application.ActiveWindow.Caption = "NewCaption"

Category Property

Returns a base component string which defines the general classification of a component such as “Animation” and “Picture”.

Syntax

*object.***Category**

Properties

The **Category** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Category is a read-only property of type *String*.

Center Property

Returns the value of the center point of the shape's bounding rectangle as a **FixFloatPoint**.

Syntax

*object.***Center**

Properties

The **Center** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Center is a read-only property of type *Object*.

CenterOfRotation Property

Specifies the point about which an object is rotated.

Syntax

*object.***CenterOfRotation** [= *Object*]

Properties

The **CenterOfRotation** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Object	Specifies the point about which to rotate.

Remarks

The *Object* is a point of type **FixFloatPoint**.

An example of how to set the **CenterOfRotation** property for a **Rectangle** to (10, 20) would be:

```
Dim Point As Object
Set Point = New FixFloatPoint

Point.X = 10
Point.Y = 20
Rect1.CenterOfRotation = Point
```

CenterPoint Property

Specifies the coordinates of the object's center point.

Syntax

*object.***CenterPoint** [= *Object*]

Properties

The **CenterPoint** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Object	Specifies the value of the center point.

Remarks

The *Object* is a point of type **FixFloatPoint**.

CenterX Property

Specifies the value of the x-coordinate of the center point of the specified object.

Syntax

*object.***CenterX** [= *Double*]

Properties

The **CenterX** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The x-coordinate of the center point.

CenterY Property

Specifies the value of the y-coordinate of the center point of the specified object.

Syntax

*object.***CenterY** [= *Double*]

Properties

The **CenterY** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The y-coordinate of the center point.

CharactersPerLine Property

Specifies the number of characters allowed per line.

Syntax

*object.***CharactersPerLine** [=*Integer*]

Properties

The **CharactersPerLine** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The number of characters allowed per line.

ChartFontSize Property

Specifies the font size for the Enhanced Chart.

Syntax

*object.***ChartFontSize** [=*enumChartFontSize*]

Properties

The **ChartFontSize** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumChartFontSize	Specifies the font size: **Valid entries:** 0 – Large 1 – Medium (Default) 2 – Small

CheckForAlarmListChanged Property

Specifies whether the Alarm Summary object tracks changes to the list of alarms and fires the AlarmListChanged event.

Syntax

*object.***CheckForAlarmListChanged** [= *Boolean*]

Properties

The **CheckForAlarmListChanged** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the **Alarm Summary** object tracks changes to the list of alarms and fires the **AlarmListChanged** event.

Settings

The settings for *Boolean* are:

Value	Description
True	Tracks changes to the list of alarms so that the **AlarmListChanged** event fires.
False	Does not track changes to the list of alarms. Consequently the **AlarmListChanged** event never fires. (Default)

Remarks

The best place to set this property is in the Initialize event handler of your picture.

CheckForNewAlarms Property

Specifies whether the **Alarm Summary** object tracks new alarms and fires a **NewAlarm** event.

Syntax

object.**CheckForNewAlarms** [= *Boolean*]

Properties

The **CheckForNewAlarms** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the **Alarm Summary** object tracks new alarms and fires a **NewAlarm** event.

Settings

The settings for *Boolean* are:

Value	Description
True	Tracks new alarms and so that the **NewAlarm** event fires.
False	Does not track new alarms. Consequently, the **NewAlarm** event never fires. (Default)

Remarks

The best place to set the **CheckForNewAlarms** property is in the Initialize event handler of your picture.

CheckForSeverityIncrease Property

Specifies whether the **Alarm Summary** object tracks when an alarm's status increases in severity and fires the **SeverityIncreased** event.

The **CheckForSeverityIncrease** property must be set in run mode. The value you enter here is not persisted. In other words, when you switch from run mode to configure mode, the value changes back to FALSE (0), which is the default. If you enter TRUE (1) in configure mode, it switches back to FALSE (0) when you enter run mode. You must set this value in run mode.

Syntax

*object.***CheckForSeverityIncrease** [= *Boolean*]

Properties

The **CheckForSeverityIncrease** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Specifies whether the **Alarm Summary** object tracks when an alarm's status increases in severity and fires the **SeverityIncreased** event.

Settings

The settings for *Boolean* are:

Value	Description
True	Tracks changes to the list of alarms so that the **SeverityIncreased** event fires.
False	Does not track changes to the list of alarms. Consequently, the **SeverityIncreased** event never fires. (Default)

Remarks

The best place to set this property is in the Initialize event handler of your picture.

ClassName Property

Returns the class of the specified object.

Syntax

*object.***ClassName**

Properties

The **ClassName** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ClassName is a read-only property of type *String*.

Color Property

Specifies the color of the ColorButton object.

Syntax

object.**Color** [= *Long*]

Properties

The **Color** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the color of the **ColorButton**.

ColorTable Property

Specifies whether the user is setting up a color table for the Lookup object.

Syntax

object.**ColorTable** [= *Boolean*]

Properties

The **ColorTable** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the Lookup table is a color table or not.

Settings

The settings for *Boolean* are:

Value	Description
True	The table is a color table.
False	The table is not a color table. (Default)

Remarks

This property must be specified when using color tables so that iFIX knows to interpret the range values as colors.

CombinationKey Property

Indicates how the Control and Shift keys are used in defining the key combination for a key macro object.

Syntax

*object.***CombinationKey** [= *Value*]

Properties

The **CombinationKey** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Value	0 - ComboKeyNone Neither the Control or Shift key is used in combination with the key code. 1- ComboKeyCtrl Only the Control key is used in combination with the key code. 2 - ComboKeyShift Only the Shift key is used in combination with the key code. 3 - ComboKeyCtrlShift Both the Control and the Shift key are used in combination with the key code.

Comments Property

Specifies the comments associated with the current document.

Syntax

*object.***Comments** [= *String*]

Properties

The **Comments** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The user-defined comments associated with the current document.

CompletionStatus Property

Retrieves the last value written to the completion status tag by the SecuritySynchronizer object.

Syntax

*object.***CompletionStatus** [= *Boolean*]

Properties

The **CompletionStatus** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	True (1) is written to this property only when the security synchronization process completes. You must manually set the value to **False** (0) before calling the SynchronizeSecurity method, to see this property value change to **True** (1) when the synchronization process completes.

CompletionStatusTag Property

Sets or retrieves the iFIX database tag and floating point field that indicates the status of the completion flag.

Syntax

object.**CompletionStatusTag** [= *String*]

Properties

The **CompletionStatusTag** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The Node.Tag.Field to be set or retrieved.

Remarks

CompletionStatusTag corresponds to the /C command line parameter of the Security Synchronizer application.

ConfirmDataEntry Property

Specifies whether to confirm data entry.

Syntax

object.**ConfirmDataEntry** [= *Boolean*]

Properties

The **ConfirmDataEntry** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether or not to confirm data entry.

Settings

The settings for *Boolean* are:

Value	Description
True	Data entry is confirmed.
False	Data entry is not confirmed. (Default)

ConnectionFailed Property

Returns whether the connection attempt was successful or not.

Syntax

*object.***ConnectionFailed**

Properties

The **ConnectionFailed** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ConnectionFailed **is a read-only property of type *Object*.**

Return Values

The **ConnectionFailed** property return values are:

Value	Description
True	The connection failed.
False	The connection was successful.

ConstantLine Property

Specifies whether to display a constant line for the specified **Pen**.

Syntax

*object.***ConstantLine** [= *Boolean*]

Properties

The **ConstantLine** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display the **Pen** as a constant line.

Settings

The settings for *Boolean* are:

Value	Description
True	The **Pen** is displayed as a constant line.
False	The **Pen** is not displayed as a constant line.

Remarks

The value displayed in the line is the current value for the specified **Pen**.

ContainedObjects Property

Returns a collection of objects contained within the specified object.

Syntax

*object.***ContainedObjects**

Properties

The **ContainedObjects** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ContainedObjects is a read-only property of type *Object*.

ContainedSelections Property

Returns a collection of objects contained within the current object which are currently selected.

Syntax

*object.***ContainedSelections**

Properties

The **ContainedSelections** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ContainedSelections is a read-only property of type *Object*.

ContextID Property

Specifies the context ID for the user's context sensitive help file.

Syntax

object.**ContextID** [= *Long*]

Properties

The **ContextID** property syntax has these parts:

Part	Description
objecContainedSelections Propertyt	An object expression that evaluates to an object in the Applies To list.
Long	Specifies the ID for help file.

ControlOrderIndex Property

Specifies the order in which the object will be selected via the "Up" and "Down" arrow keys.

Syntax

object.**ControlOrderIndex** [= *Long*]

Properties

The **ControlOrderIndex** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Specifies the order of selection.

Count Property

Returns the number of items in the specified collection or the number of levels in a **Lookup** object.

Syntax

object.**Count**

Properties

The **Count** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Count is a read-only property of type *Long*.

CurrentDataSet Property

Returns or sets the current data set of a chart by position.

Syntax

*object.***CurrentDataSet**

Properties

The **CurrentDataSet** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

The **CurrentDataSet** property is a zero-based long integer indicating the current data set of the chart.

CurrentDataSource Property

Returns the current data source.

Syntax

*object.***CurrentDataSource**

Properties

The **CurrentDataSource** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

CurrentDataSource is a read-only property of type *String*.

CurrentDate Property

Returns the current system date. The date string is formatted according to the "short date" format in the Regional and Language Options in the Control Panel.

Syntax

*object.***CurrentDate** [= *String*]

Properties

The **CurrentDate** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The current date.

CurrentDateDay Property

Returns the day component of the current system date.

Syntax

*object.***CurrentDateDay** [= *String*]

Properties

The **CurrentDateDay** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The day component of the current date.

CurrentDateMonth Property

Returns the month component of the current system date.

Syntax

*object.***CurrentDateMonth** [= *String*]

Properties

The **CurrentDateMonth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrEventName	String. The name of the event.

CurrentDateYear Property

Returns the year component of the current system date.

Syntax

*object.***CurrentDateYear** [= *String*]

Properties

The **CurrentDateYear** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The year component of the current date.

CurrentImage Property

Specifies the index of the image that is currently being displayed.

Syntax

*object.***CurrentImage** [= *Integer*]

Properties

The **CurrentImage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The index of the currently displayed image.

CurrentPen Property

Specifies the current **Pen** by it's index in the **Pens** collection.

Syntax

*object.***CurrentPen** [= *Long*]

Properties

The **CurrentPen** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The index of the current **Pen**.

CurrentPicture Property

Returns the currently active picture displayed in the iFIX WorkSpace.

Syntax

*object.***CurrentPicture**[= *String*]

Properties

The **CurrentPicture** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The current active picture alias or name.

CurrentTime Property

Specifies the current system time. The time string is formatted according to the "time" format in the Regional and Language Options in the Control Panel.

Syntax

*object.***CurrentTime** [= *String*]

Properties

The **CurrentTime** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The current time.

CurrentTimeHour Property

Returns the hour component of the current system time.

Syntax

*object.***CurrentTimeHour** [= *String*]

Properties

The CurrentTimeHourproperty syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The hour component of the current time.

CurrentTimeMinute Property

Returns the minute component of the current system time.

Syntax

object.**CurrentTimeMinute** [= *String*]

Properties

The **CurrentTimeMinute** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The minute component of the current time.

CurrentTimeSecond Property

Returns the second component of the current system time.

Syntax

object.**CurrentTimeSecond** [= *String*]

Properties

The **CurrentTimeSecond** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The second component of the current time.

CurrentValue Property

Specifies the current value for the specified **Pen**or **Variable**.

Syntax

object.**CurrentValue** [= *Double*]

Properties

The **CurrentValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The current value.

D

DataEntry Property

Specifies the data entry type.

Syntax

*object.***DataEntry** [= *Integer*]

Properties

The **DataEntry** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The data entry type.

Settings

The settings for *Integer* are:

Value	Description
0	None. (Default)
1	In line.

DataItems Property

Returns a user defined collection of **DataItem** objects. **DataItem** objects can be added and removed from this collection using the **Add** and **Remove** methods. When adding a **DataItem**, the **DataItem** must exist within the iFIX data system or it will not be added to the collection. **DataItem** names must be unique.

Syntax

*object.***DataItems**

Properties

The **DataItems** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

DataItems is a read-only property of type *Object*.

DataRefreshInterval Property

Specifies the rate at which real-time values are retrieved for tags in the current alarm list in the **Alarm Summary** object.

Syntax

*object.***DataRefreshInterval** [= *Single*]

Properties

The **DataRefreshInterval** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Single	The rate, in seconds, at which real time values are retrieved for tags in the current alarm list. The valid values are 0.1 - 300.0 The default is 0.5 seconds.

DataServers Property

Returns a collection of installed data servers in the **FixDataSystem**.

Syntax

*object.***DataServers**

Properties

The **DataServers** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

DataServers is a read-only property of type *Object*.

The **DataServers** information is registered during installation of the OPC server using the DataServer-Installer program.

DataSetColor Property

Sets the color used for the data set in the Enhanced Chart.

Syntax

*object.***DataSetColor** [= *Long*]

Properties

The **DataSetColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The color for the GeneralDataSet Object or RealTimeSPCDataSet Object in the Enhanced Chart.

DataShadows Property

Sets whether shadows or 3D effects will be used in the plotting method for an Enhanced Chart.

Syntax

*object.***DataShadows** [=*enumDataShadows*]

Properties

The **DataShadows** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumDataShadows	Specifies the shadow effects for the plotting method in the HistogramChart, LineChart, SPCBarChart, or XYChart: **Valid entries:** 0 – DataShadowsNone 1 – DataShadows 2 – Data3D

DaylightSavingsTime Property

Takes daylight saving time changes into account.

Syntax

*object.***DaylightSavingsTime** [= *Boolean*]

Properties

The **DaylightSavingsTime** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Specifies whether daylight saving time is enabled.

Settings

The settings for *Boolean* are:

Value	Description
True	Consider the daylight saving time changes.
False	Do not consider the daylight saving time changes.

Remarks

The **DaylightSavingsTime** property defaults to what is set in the Date/Time control panel under “automatically adjust clock for daylight saving changes.”

DaysBeforeNow Property

Specifies the initial start date for the **Chart**,**Pen** **GeneralDataSet Object** or,**Formatted Object**,**Line**,**Lookup Object**,relative to the date the parent **Picture** is opened.

Syntax

*object.***DaysBeforeNow** [= *Integer*]

Properties

The **DaysBeforeNow** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The relative initial start date.

Remarks

DaysBeforeNow is a one-shot property.

This property is not impacted by any Global Time Control property settings.

DaysOfMonth Property

Specifies which days in the month to run the current **Timer** object.

Syntax

*object.***DaysOfMonth** [= *Long*]

Properties

The **DaysOfMonth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The value corresponding to the bit mask for each day, where day 1 is the low order bit and the end of the month is the 32nd bit.

Remarks

DaysOfMonth only applies if the **TriggerType** is set to *Monthly*.

DaysOfWeek Property

Specifies which days in the week to run the current **Timer** object.

Syntax

*object.***DaysOfWeek** [= *Long*]

Properties

The **DaysOfWeek** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The value corresponding to the bit mask for each day, where day 1 is the low order bit.

Remarks

DaysOfWeek only applies if the **TriggerType** is set to *Daily*.

Deadband Property

Specifies the amount a value must change by + or - before a data change is recognized.

Syntax

*object.***Deadband** [= *Single*]

Properties

The **Deadband** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Single	The amount the value must change.

DecimalDigits Property

Specifies the number of digits to be displayed after the decimal point.

Syntax

*object.***DecimalDigits** [=*Integer*]

Properties

The **DecimalDigits** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The number of digits.

Default Property

Specifies whether the specified control is the default button in a picture.

Syntax

*object.***Default** [= *Boolean*]

Properties

The **Default** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Settings

The settings for *Boolean* are:

Value	Description
True	The control is the Default button of the picture in the Run-time environment.
False	The control is not the Default button of the picture. (Default)

Remarks

Only one control in a picture can be the default button. When **Default** is set to **True** for one control, it is automatically set to **False** for all other controls in the picture. When the control's **Default** property setting is **True** and its parent picture is active, the user can choose the command button (invoking its Click event) by pressing ENTER. Any other control with the focus doesn't receive a keyboard event (KeyDown, KeyPress, or KeyUp) for the ENTER key unless the user has moved the focus to another button in the same picture. In this case, pressing ENTER chooses the button that has the focus instead of the default button.

The **Default** property of a control can be set to **True** only if the control is a pushbutton control or any control that behaves like a button i.e., marked with OLEMISC_ACTSLIKEBUTTON flag.

DefaultDataSystem Property

Returns which is the default datasystem when a user types in a tag in an animation. For example, if the default datasystem is "Fix32" and a users enters "AI1" into an animations dialog, the Fix32 datasystem will resolve the tag (i.e. Fix32.AI1).

Syntax

*object.***DefaultDataSystem**

Properties

The **DefaultDataSystem** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

DefaultDataSystem is a read-only property of type *String*.

The **DefaultDataSytem** information is registered during installation of the OPC server using the DataServerInstaller program.

DefaultExternalDatasourceUpdateRate Property

Returns the refresh rate (in seconds) the Basic Animations dialog box uses as a default for a connection.

Syntax

*object.***DefaultExternalDatasourceUpdateRate**

Properties

The **DefaultExternalDatasourceUpdateRate** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

DefaultExternalDatasourceUpdateRate is a read-only property of type *Single*.

DefaultOutputValue Property

Specifies the default output value.

Syntax

*object.***DefaultOutputValue** [= *Variant*]

Properties

The **DefaultOutputValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	Variant. The default output value.

DefaultServer Property

Returns a flag signifying whether this OPC DataServer was installed as the default.

Syntax

*object.***DefaultServer**

Properties

The **DefaultServer** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Values

The **DefaultServer** property return values are:

Value	Description
0	The specified data server is not the default server.
1	The specified data server is the default server.

Remarks

DefaultServer is a read-only property of type *String*.

The **DefaultServer** information is registered during installation of the OPC server using the DataServer-Installer program.

Description Property

A user defined description of the specified object's function to be displayed in the object's tooltip.

Syntax

*object.***Description** [= *String*]

Properties

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The description as defined by the user.

DeskColor Property

Allows you to specify the surrounding color of the Enhanced Chart, behind the title, sub-titles, and legends.

Syntax

*object.***DeskColor** [= *Long*]

Properties

The **DeskColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The surrounding color represented as an Integer value.

DigitalError Property

Retrieves the last value written to the digital error tag by the SecuritySynchronizer object. The value is represented as a boolean.

Syntax

*object.***DigitalError**

Properties

The **DigitalError** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

DigitalError is a read-only property. It is only updated by SecuritySynchronizer at the end of the security synchronization process. A value of 1 is written if an error is detected.

DigitalErrorTag Property

Sets or retrieves the digital iFIX database tag and floating point field to which a digital failure code is written when the security synchronization process completes.

Syntax

*object.***DigitalErrorTag** [= *String*]

Properties

The **DigitalErrorTag** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The Node.Tag.Field to be set or retrieved.

Remarks

DigitalErrorTag corresponds to the /F command line parameter of the Security Synchronizer application.

DigitsOfPrecision Property

Sets the number of decimal positions that are used in outputting data to the object Cursor Prompt, Tables, Data Labels, and the Clipboard. Even though you set the numeric precision, the number of decimal points specified does not appear on the chart unless it is necessary. For example, if you specified the use of two decimal places, but all of the data values in your chart are whole numbers, decimal points are not be used because they are not necessary. This setting does not apply to the axes graduations and ticks.

Syntax

*object.***DigitsOfPrecision** [= *Long*]

Properties

The **DigitsOfPrecision** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the number of decimal positions.

DisableAutoScale Property

Describes whether auto scaling is disabled. When you enable this setting, you override the picture's logical units to a pixel ratio, when changing the resolution of your screen. This may be helpful for multiple monitor configurations.

Syntax

*object.***DisableAutoScale** [= *Boolean*]

Properties

The **DisableAutoScale** property syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether auto scaling is enabled.

Settings

The settings for *Boolean* are:

Value	Description
True	Automatic picture scaling disabled.
False	Automatic picture scaling enabled (Default).

DisplayLayer Property

Specifies the display layer for the current picture or dynamo set.

Syntax

*object.***DisplayLayer** [= *Long*]

Properties

The **DisplayLayer** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The display layer.

Remarks

The **DisplayLayer** property is not saved to disk, it is a transient property which is reset each time the document is opened. To specify a particular display layer for a **Picture** when the picture is opened in the Run-time environment, open the picture as "hidden" using the **Open** method and set the **DisplayLayer** property in the pictures **Activated** event.

DisplayMilliseconds Property

If set, displays the millisecond component of time on the time axis and tooltip of the chart.

Syntax

*object.***DisplayMilliseconds** [= *Boolean*]

Properties

The **DisplayMilliseconds** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display milliseconds or not.

Settings

The settings for *Boolean* are:

Value	Description
True	Display the millisecond component of the time.
False	Do not display the millisecond component of the time (Default).

DisplayShelvedAlarms Property

This property used to filter the shelved alarms in the **Alarm Summary** object.

Syntax

object.**DisplayShelvedAlarms** [=Boolean]

Properties

The **DisplayShelvedAlarms** property syntax has these parts:

Part	Description
object	An object expression that evaluates to the Alarm Summary object in the Applies To list.
Boolean	Whether the shelved alarms are displayed in the Alarm Summary object.

Settings

The settings for *DisplayShelvedAlarms* are:

Constant	Description
True	When set to True, only shelved alarms are displayed in the Alarm Summary object. (Default)
False	When set to False, alarms that are not shelved are displayed in the Alarm Summary object.

Example

This example applies a shelved filter on the Alarm Summary object.

```
' Applies shelved filter on alarm summary object
Public Sub ShelvedAlarmsFilter(FilterFlag As Boolean)
Dim AppObj As Object
Dim PictureObj As Object
Dim CurrentObj As Object
If TypeName(Application) = "CFixApp" Then
' running in the workspace
Set AppObj = Application

Else
Set AppObj = App

If AppObj Is Nothing Then
Exit Sub
End If

End If
' Search for Alarm summary object and apply the shelved alarms filter
Set PictureObj = AppObj.ActiveDocument
For Each CurrentObj In PictureObj.Page.ContainedObjects
If TypeName(CurrentObj) = "AlarmSummaryOCX" Then
If CurrentObj.Name = "AlarmSummaryOCX1" Then
CurrentObj.DisplayShelvedAlarms = FilterFlag
Exit Sub
End If
End If
Next
End Sub
```

DisplayStatusBar Property

Specifies whether the WorkSpace's Status Bar is visible.

Syntax

object.**DisplayStatusBar** [= *Boolean*]

Properties

The **DisplayStatusBar** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the Status Bar is visible or not.

Settings

The settings for *Boolean* are:

Value	Description
True	The Status Bar is visible.
False	The Status Bar is not visible.

DisplayString Property

Retrieves and sets the name of the procedure to execute when the key seqeunce of the key macro is matched.

Syntax

*object.***DisplayString** [= *DisplayString*]

Properties

The **DisplayString** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
DisplayString	String. The new string representing the key combination.

DisplaySystemTree Property

Specifies whether the WorkSpace's System Tree is visible.

Syntax

*object.***DisplaySystemTree** [= *Boolean*]

Properties

The **DisplaySystemTree** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the System Tree is visible or not.

Settings

The settings for *Boolean* are:

Value	Description
True	The document is active.
False	The document is not active.

DocumentHeight Property

Specifies the height of the document in logical units, for Logical Coordinates.

Syntax

*object.***DocumentHeight** [= *Double*]

Properties

The **DocumentHeight** property syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
Double	A user-defined number in logical units for the vertical height.

Remarks

The default **DocumentHeight** is 75.

DocumentHeightEx Property

Specifies the height of the document in postscript points, for Enhanced Coordinates.

Syntax

*object.***DocumentHeightEx** [= *Double*]

Properties

The **DocumentHeightEx** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	A user-defined number in postscript points for the vertical height.

Remarks

The default **DocumentHeightEx** is different for each screen resolution. The default value is calculated based on the current screen resolution.

DocumentPath Property

Returns the path used to store non-FIX related documents such as Word and Excel documents.

Syntax

*object.***DocumentPath**

Properties

The **DocumentPath** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

DocumentPath is a read-only property of type *String*.

Documents Property

Returns a collection of the documents that are open in the WorkSpace.

Syntax

*object.***Documents**

Properties

The **Documents** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Documents is a read-only property of type *Object*.

Each member of the collection is represented by a **Document** object.

DocumentWidth Property

Specifies the width of the document in logical units, for Logical Coordinates.

Syntax

*object.***DocumentWidth** [= *Double*]

Properties

The **DocumentWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	A user-defined number in logical units for the horizontal width.

Remarks

The default **DocumentWidth** is 100.

DocumentWidthEx Property

Specifies the width of the document in postscript points, for Enhanced Coordinates. .

Syntax

*object.***DocumentWidthEx** [= *Double*]

Properties

The **DocumentWidthEx** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	A user-defined number in postscript points for the horizontal width.

Remarks

The default **DocumentWidthEx** is different for each screen resolution. The default value is calculated based on the current screen resolution.

Domain Property

Sets or retrieves the name of the Windows domain that acts as the source of security information for the security synchronization process.

Syntax

*object.***Domain** [= *String*]

Properties

The **Domain** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A valid Windows domain name.

Remarks

You must set **Domain** to a valid domain name when the **UseDomainSecurity** property is equal to **True**.

This property corresponds to the domain name following the /D command line parameter of the Security Synchronizer application.

DownImageDisplayed Property

Specifies whether the secondary image of the **Bitmap** is to be displayed when the mouse is down.

Syntax

*object.***DownImageDisplayed** [= *Boolean*]

Properties

The **DownImageDisplayed** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the secondary image is displayed on when the mouse is down.

Settings

The settings for *Boolean* are:

Value	Description
True	The secondary image is displayed when the mouse is down.
False	The secondary image is not displayed when the mouse is down.

DSDescription Property

Sets the data set description in the GeneralDataSet Object or RealTimeSPCDataSet object.

Syntax

*object.***DSDescription** [= *String*]

Properties

The **DSDescription** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	String. The string representing the data set description.

DSLegendAvgerageOverRangeColWidth Property

Sets the width of the Average Over Range Legend column in an Enhanced Chart.

Syntax

*object.***DSLegendAvgerageOverRangeColWidth** [= *Long*]

Properties

The **DSLegendAvgerageOverRangeColWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the width of the Average Over Range Legend column.

DSLegendCurrentValColWidth Property

Specifies the column width of the data value in the legend.

Syntax

*object.***DSLegendCurrentValColWidth** [= *Long*]

Properties

The **DSLegendCurrentValColWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the column width of the data value in the legend.

DSLegendDescriptionColWidth Property

Returns the top, left, and bottom-right values of the shape's bounding rectangle.

Syntax

object.**DSLegendDescriptionColWidth** [= *Long*]

Properties

The **DSLegendDescriptionColWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the top, left, and bottom-right values of the shape's bounding rectangle.

DSLegendEngUnitsColWidth Property

Specifies the column width of the engineering units column in the legend for an Enhanced Chart.

Syntax

object.**DSLegendEngUnitsColWidth** [= *Long*]

Properties

The **DSLegendEngUnitsColWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number, from 0 – 80, representing the column width of the engineering units in the legend in the HistogramChart, LineChart, SPCBarChart, or XYChart Object. For pictures created before iFIX 5.5, the default value is 0. For all other pictures, the default value is 10.

DSLegendHighLimitColWidth Property

Specifies the column width of the high limit in the legend in an Enhanced Chart.

Syntax

*object.***DSLegendHighLimitColWidth** [= *Long*]

Properties

The **DSLegendHighLimitColWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the column width of the high limit in the legend in the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

DSLegendHighOverRangeColWidth Property

Specifies the column width of the high over range limit in the legend in an Enhanced Chart.

Syntax

*object.***DSLegendHighOverRangeColWidth** [= *Long*]

Properties

The **DSLegendHighOverRangeColWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the column width of the high over range limit in the legend in the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

DSLegendLowLimitColWidth Property

Specifies the column width of the low limit in the legend in an Enhanced Chart.

Syntax

*object.***DSLegendLowLimitColWidth** [= *Long*]

Properties

The **DSLegendLowLimitColWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the column width of the low limit in the legend in the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

DSLegendLowOverRangeColWidth Property

Specifies the column width of the low over range limit in the legend in an Enhanced Chart.

Syntax

*object.***DSLegendLowOverRangeColWidth** [= *Long*]

Properties

The **DSLegendLowOverRangeColWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the column width of the low over range limit in the legend in the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

DSLegendMask Property

Indicates which legend items to show in the GeneralDataSet Object or RealTimeSPCDataSet object.

Syntax

*object.***DSLegendMask** [= *Long*]

Properties

The **DSLegendMask** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	A value that represents the Legend items to show in the data set: DS_LEGEND_BITMASK_SOURCE_NAME (UINT32) – 0x00000001 DS_LEGEND_BITMASK_DESCRIPTION (UINT32) – 0x00000002 DS_LEGEND_BITMASK_CURRENT_VALUE (UINT32) – 0x00000004 DS_LEGEND_BITMASK_LOW_LIMIT (UINT32) – 0x00000008 DS_LEGEND_BITMASK_HIGH_LIMIT (UINT32) – 0x00000010 DS_LEGEND_BITMASK_AVG_OVER_RANGE (UINT32) – 0x00000020 DS_LEGEND_BITMASK_LOW_OVER_RANGE (UINT32) – 0x00000040 DS_LEGEND_BITMASK_HIGH_OVER_RANGE (UINT32) – 0x00000080 DS_LEGEND_BITMASK_QUALITY (UINT32) – 0x00000100

Remarks

DSLegendMask is a read-only property.

DSLegendQualityColWidth Property

Specifies the column width of the data quality in the legend in the Enhanced Chart.

Syntax

*object.***DSLegendQualityColWidth** [= *Long*]

Properties

The **DSLegendQualityColWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the column width of the data quality in the legend in the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

DSLegendSourceColWidth Property

Specifies the column width of the data source name in the legend in an Enhanced Chart.

Syntax

*object.***DSLegendSourceColWidth** [= *Long*]

Properties

The **DSLegendSourceColWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the column width of the data source name in the legend in the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

DSPosition Property

DSPosition is a read-only data set property that allows you to retrieve the position of a data set in the chart. For example, if you have one real-time data set, the DSPosition is 0. If you have two data sets in a chart, the second data set's DSPosition is 1.

Syntax

*object.***DSPosition** [= *Long*]

Properties

The **DSPosition** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the position of the data set.

Remarks

DSPosition is a read-only property.

Duration Property

Specifies the time duration, in seconds, to display data in the **Chart** ,**HistogramChart** ,**LineChart** ,or **SPCBarChart**.

Syntax

*object.***Duration** [= *Long*]

Properties

The **Duration** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The length of time for which the **Chart** displays data.

Dynamo_Description Property

Returns the text description of a Dynamo object, if one exists. This property is read-only.

Syntax

*DynamoObject.***Dynamo_Description**

Properties

The **Dynamo_Description** property syntax has this part:

Part	Description
DynamoObject	A Dynamo object. **TIP:** The maximum number of characters that you enter into the Dynamo_Description property is available in the Max_Dynamo_Desc_Length Property.

Return Value

String. A text description of the Dynamo object.

Dynamo_ID Property

Returns the unique identifier (GUID) for the Dynamo Object. This property is read-only.

Syntax

*DynamoObject.***Dynamo_ID**

Properties

The **Dynamo_ID** property syntax has this part:

Part	Description
DynamoObject.	A Dynamo object.

Return Value

String. This string represents a unique 128-bit number used as the Globally Unique Identifier (GUID) for the Dynamo object.

E

EdgeColor Property

Specifies a shape's edge color.

Syntax

*object.***EdgeColor** [= *Long*]

Properties

The **EdgeColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the shape's edge color.

EdgeStyle Property

Specifies the value representing a shape's edge style.

Syntax

*object.***EdgeStyle** [= *enumEdgeStyle*]

Properties

The **EdgeStyle** property syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
EnumEdgeStyle	The edge style to be displayed for the specified shape.

Settings

The settings for *enumEdgeStyle* are:

Constant	Value	Description
EdgeStyleSolid	0	Solid
EdgeStyleDash	1	Dash
EdgeStyleDot	2	Dot
EdgeStyleDashDot	3	Dash-Dot
EdgeStyleDashDotDot	4	Dash-Dot-Dot
EdgeStyleNone	5	No border.
EdgeStyleInsideFrame	6	Inside Frame.

Bitmap Object Syntax

*object.***EdgeStyle** [= *enumBitmapEdgeStyle*]

Properties

The **EdgeStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
EnumBitmapEdgeStyle	The edge style to be displayed for the specified bitmap.

Object Settings

The settings for *enumBitmapEdgeStyle* are:

Constant	Value	Description
BitmapEdgeNone	0	No edge.
BitmapEdgeSunken	1	Sunken edge.
BitmapEdgeRaised	2	Raised edge.
BitmapEdgeEtched	3	Etched edge.
BitmapEdgeBump	4	Bump edge.

Remarks

Changes to the **EdgeStyle** property are only visible when the **EdgeWidth** property of the object is set to 1.

EdgeWidth Property

Specifies a shape's border width.

Syntax

*object.***EdgeWidth** [= *Long*]

Properties

The **EdgeWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	A number from 0 to 200.

EditText Property

Specifies the text to be displayed in the combo box of the ExpressionEditor.

Syntax

*object.***EditText** [= *String*]

Properties

The **EditText** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The text displayed in the **ExpressionEditor's** combo box.

ElbowStyle Property

Specifies the elbow style to be applied to the current pipe object.

Syntax

*object.***ElbowStyle** [= *enumElbowStyle*]

Properties

The **ElbowStyle** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Settings

The settings for *enumElbowStyle* are:

Value	Description
0	ElbowStyleRound
1	ElbowStyleSquare

EnableAcknowledgeAll Property

Specifies whether Acknowledge All Alarms can be performed from the Alarm Summary object.

Syntax

object.**EnableAcknowledgeAll** [=Boolean]

Properties

The **EnableAcknowledgeAll** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether Acknowledge All Alarms can be performed.

Settings

The settings for Boolean are:

Value	Description
True	Acknowledge All Alarms can be performed from the Alarm Summary object. (Default)
False	Acknowledge All Alarms cannot be performed from the Alarm Summary object.

Remarks

Setting EnableAcknowledgeAll to True allows the user to acknowledge all alarms from the Alarm Summary object. Acknowledge All Alarms is not supported by electronic signature. Therefore, it is recommend that EnableAcknowledgeAll be set to False on systems enabled for electronic signatures.

EnableAlarmAcknowledge Property

Specifies whether alarms can be acknowledged using the **Alarm Summary** window.

Syntax

object.**EnableAlarmAcknowledge** [= *Boolean*]

Properties

The **EnableAlarmAcknowledge** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether alarms can be acknowledged.

Settings

The settings for *Boolean* are:

Value	Description
True	Alarms can be acknowledged using the **Alarm Summary** window. (Default)
False	Alarms cannot be acknowledged using the **Alarm Summary** window.

EnableAlarmDeletion Property

Specifies whether alarms can be deleted from the Alarm Summary object.

Syntax

object.**EnableAlarmDeletion** [= *Boolean*]

Properties

The **EnableAlarmDeletion** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether alarms can be deleted.

Settings

The settings for *Boolean* are:

Value	Description
True	Alarms can be deleted from the **Alarm Summary** object. (Default)
False	Alarms cannot be deleted from the **Alarm Summary** object.

Remarks

Setting **EnableAlarmDeletion** to **True** allows the user to delete alarms whether they have been acknowledged or not.

EnableAsVbaControl Property

Specifies whether a variable object is registered in VBA and if you can use it in VBA scripts. For example, you may want to disable the registration of some variable objects in VBA if you do not need these objects in scripts or need to create event procedures for these objects (i.e. OnChange, OnFalse, or OnTrue). By minimizing the number of VBA objects, you optimize performance.

Syntax

object.**EnableAsVbaControl** [= *Boolean*]

Properties

The **EnableAsVbaControl** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the object is registered in VBA and if it can be used in VBA scripts.

Remarks

An example of how to set the **EnableAsVbaControl** property for a **picture** would be:

```
picture_name.object_name.EnableAsVbaControl False
```

EnableColumnQuickSort Property

Specifies whether columns can be sorted in the **Alarm Summary** object.

Syntax

*object.***EnableColumnQuickSort** [= *Boolean*]

Properties

The **EnableColumnQuickSort** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether columns can be sorted.

Settings

The settings for *Boolean* are:

Value	Description
True	A left mouse click on a column header toggles the sort order of the list of alarms between ascending and descending order.
False	The list cannot be sorted.

Remarks

Only the Node, Priority, Tagname, and Time In columns support sorting.

Enabled Property

Returns whether the **Color Button** or **FixKeyMacro Object** is enabled.

Syntax

*object.***Enabled**

Properties

The **Enabled** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Values

The **Enabled** property return values are:

Value	Description
True	The specified object is enabled.
False	The specified object is not enabled.

Remarks

Enabled is a read-only property of type *Boolean*.

EnableGlobalEndTime Property

Specifies whether the end time is utilized for the object display. When this property is false, the combination of the GlobalStartTime and the GlobalDuration define the GlobalEndTime of the Global Time Control.

Syntax

*object.***EnableGlobalEndTime** [= *Boolean*]

Properties

The **EnableGlobalEndTime** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the end time is enabled

Settings

The settings for *Boolean* are:

Value	Description
True	End time is enabled.
False	End time is not enabled (Default).

Remarks

The value for this property must be set to True before you can set the value for the GlobalEndTime property.

EnableGlobalScrollPercentage Property

Specifies whether the scroll percentage is enabled for the Global Time Control.

Syntax

object.**EnableGlobalScrollPercentage** [= *Boolean*]

Properties

The **EnableGlobalScrollPercentage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the scroll percentage is enabled.

Settings

The settings for *Boolean* are:

Value	Description
True	Scroll percentage is enabled (Default).
False	Scroll percentage is not enabled.

Remarks

This property must be disabled before you can use the GlobalFastScrollOption or GlobalSlowScrollOption properties.

EnableEndTime Property

Specifies whether the end time is utilized or not for a Timer object.

Syntax

object.**EnableEndTime** [= *Boolean*]

Properties

The **EnableEndTime** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the end time is enabled.

Settings

The settings for *Boolean* are:

Value	Description
True	End time is enabled.
False	End time is not enabled.

EnableRightMouseClick Property

Specifies whether the right mouse menu is displayed when the user clicks in the Alarm Summary object.

Syntax

object.**EnableRightMouseClick** [= *Boolean*]

Properties

The **EnableRightMouseClick** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the right mouse menu is displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The right mouse menu is displayed when the user clicks the right mouse in the spreadsheet. (Default)
False	The right mouse menu is not displayed.

EnableRunTimeConfiguration Property

Specifies whether the user is allowed to change the Alarm Summary filter and sort in the Run-time environment.

Syntax

*object.***EnableRunTimeConfiguration** [= *Boolean*]

Properties

The **EnableRunTimeConfiguration** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the filter and sort can be changed in the run environment.

Settings

The settings for *Boolean* are:

Value	Description
True	The user can change the filter and sort in the **Alarm Summary** object in the Run-time environment. (Default)
False	The user is not allowed to change the filter and sort.

EnableTooltips Property

Specifies whether the tooltips are shown for the specified object.

Syntax

*object.***EnableTooltips** [= *Boolean*]

Properties

The **EnableTooltips** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the shape's tooltips are displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The tooltips for the specified shape are displayed.
False	The tooltips for the specified shape are not displayed. (Default)

Remarks

When **EnableTooltips** is set to **True**, the text displayed is the text set in the Description property for the specified object.

EndAngle Property

Specifies which portion of the object is visible.

Syntax

*object.***EndAngle** [= *Double*]

Properties

The **EndAngle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The number of units to display the radial segment specifying the end of the angle.

Remarks

The **Pie** is merely a visible piece of an oval. The **StartAngle** and **EndAngle** properties specify which portions of that oval will be visible. These properties define radial segments from the center of the oval between which **Pie** is formed.

An **EndAngle** of 0 units will define a horizontal radial from the center of the oval to the right.
An **EndAngle** of 90 units will define a vertical radial from the center of the oval to the top of the screen.
An **EndAngle** of 180 units will define a horizontal radial from the center of the oval to the left side of the screen.
An **EndAngle** of 270 units will define a vertical radial from the center of the oval to the bottom of the screen.

The units to be used when creating the angle is specified as either degrees or radians depending on the value of the **AngleUnits** property.

Changing the **EndAngle** property will change the **EndPoint** property.

EndCap Property

Specifies the end cap to apply to the selected pipe object.

Syntax

*object.***EndCap** [= *enumEndCap*]

Properties

The **EndCap** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Settings

The settings for *enumEndCap* are:

Value	Description

0	EndCapRound
1	EndCapSquare
2	EndCapHorizontalDiagonal
3	EndCapVerticalDiagonal

EndPoint Property

Specifies the ending point of the object.

Syntax

*object.***EndPoint** [= *Object*]

Properties

The **EndPoint** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Object	A point of type **FixFloatPoint**.

Remarks

The **EndPoint** property specifies a point object which contains an x and y double value which are logical coordinate values equivalent to the **StartX** and **StartY** properties.

The **EndPoint** for the **Pie** also defines the end points of the line segments which define the **StartAngle** and **EndAngle** of the object.

The **EndPoint** for the **Arc**, **Chord**, and **Line** objects is the point located at index 1.

The **EndPoint** for the **Pie** object is the point located at index 2.

EndTime Property

Specifies the last time displayed in the **Chart** for all pens, for a specific **Pen** and/or the end time for that day to stop running the specified **Timer** object, in the **Lookup Object**,**Line**, **Formatted Object**, or **GeneralDataSet Object** in a object.

Chart and Pen Syntax

*object.***EndTime** [= *Date*]

Properties

The **EndTime** property syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
Date	The ending time for the chart.

Remarks

The **EndTime** property is a convenience property enabling the user to set the end times for all pens displayed in the **Chart**. The end times that are displayed in the **Chart** are those for the specific pens. Therefore, the user may not see what he/she expects because the chart's end time is overruled by each of the pen's ending times.

This property is not impacted by any Global Time Control property settings.

Timer Syntax

*object.***EndTime** [= *Variant*]

Properties

The **EndTime** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	The end time of that day that you want the timer to stop running.

Remarks

EndTime takes a *DATE* for the **Timer** object. EndTime is a read-only property.

EndX Property

Specifies the horizontal location of the last point in the specified shape.

Syntax

*object.***EndX** [= *Double*]

Properties

The **EndX** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The value of the last point's X coordinate.

EndY Property

Specifies the vertical location of the last point in the specified shape.

Syntax

*object.***EndY** [= *Double*]

Properties

The **EndY** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The value of the last point's Y coordinate.

EngUnits Property

EngUnits specifies the Engineering Units for a given data source in run mode. This property only applies to data sources in Enhanced Charts.

NOTE: EngUnits is not an exported property because it is only valid during run mode.

Syntax

*object.***EngUnits** [= *String*]

Properties

The **EngUnits** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A string value representing the value of the engineering units.

EnhancedCoordinates Property

Read-only. Specifies whether the current picture uses screen independent coordinates (Enhanced Coordinates) or the legacy logical coordinates. For more information on Enhanced Coordinates, refer the Picture Coordinate Systems topic in the Creating Pictures e-book.

Syntax

*object.***EnhancedCoordinates** [= *Boolean*]

Properties

The **EnhancedCoordinates** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Boolean	Whether or not Enhanced Coordinates are used.

Settings

The settings for *Boolean* are:

Value	Description
True	Enhanced Coordinates are used.
False	Enhanced Coordinates are not used.

Remarks

The default for **EnhancedCoordinates** is True on all new pictures created in iFIX 5.8 and greater. On pictures created in previous versions of iFIX, the default for EnhancedCoordinates is False, unless you choose to upgrade to the Enhanced Coordinate system by running the Picture Upgrade expert and then the property gets set to True.

ErrorMode Property

Specifies which mode to use if the quality of the data received by the animation object is not reliable.

Syntax

object.**ErrorMode** [= *Long*]

Properties

The **ErrorMode** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The error mode.

Format and Lookup Object Settings

The settings for *Long* are:

Value	Description
0	Use old.
1	Use error.

Linear Object Settings

The settings for *Long* are:

Value	Description
0	Use old.
1	Use error.
50	Use min.
51	Use max.

Remarks

The ErrorMode property is related to the error defines specified in User Preferences for the animation objects.

When a value is sent from the data system, it has an associated quality. If the quality is bad and the user has specified *0 - Use old*, the user will not see any change of data on the screen. The last good value that the animation object sent to its target will be sent again. If the user has specified *1 - Use error*, the corresponding value specified in the User Preference will be obtained and displayed on the screen (this is most notable when the user sees "????" or "@@@@" for datalinks). If the object is a **Linear** object, and *50 - Use min* or *51 - Use max* are specified, the data displayed will be the minimum or maximum ouput values, respectively.

EventParameter Property

Reserved for internal purposes.

EventType Property

Specifies the type of event for the **Event** object.

Syntax

object.**EventType** [= *Long*]

Properties

The **EventType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The type of event.

Settings

The settings for *Long* are:

Value	Description
0	OnChange
1	OnTrue
2	OnFalse
3	WhileTrue
4	WhileFalse

NOTE: If you change the EventType property from the Properties window, you do not change the script. If you change the EventType from the Modify Event Entry dialog box then you are prompted to apply the existing script to the new event.

ExactMatch Property

Specifies if the Lookup object is a range or an exact match table.

Syntax

object.**ExactMatch** [= *Boolean*]

Properties

The **ExactMatch** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the table is a range or exact match.

Settings

The settings for *Boolean* are:

Value	Description
True	The table is an exact match.
False	The table is a range. (Default)

Expandable Property

When Expandable is set to true in an Enhanced Chart, an Expand or Contract button displays in the upper right-hand corner of the chart when the cursor hovers over that area. Pressing the Expand button causes the chart to display in full screen, while pressing the Contract button causes the chart to reset to its original size and position.

If the Thumbnail property is True, when the Expand button is pressed the Thumbnail property changes to False. When set to False, the chart does not display with the optimized thumbnail settings, but instead displays with the user configured legend, axes, and other settings. When the Contract button is pressed, the Thumbnail property is set back to True, and the chart displays using the optimized thumbnail settings.

The hot keys 'E' and 'C' (upper and lower case) can also be used to Expand or Contract the chart when the chart is selected.

Syntax

object.**Expandable** [= *Boolean*]

Properties

The **Expandable** property syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
Boolean	Describes whether the Enhanced Chart will display an Expand or Contract button in the upper right-hand corner of the chart when the mouse is moved to that area.

Settings

The settings for *Boolean* are:

Value	Description
True	The Expand or Contract button displays in the upper right-hand corner of your Enhanced Chart.
False	The Expand or Contract button does not display in the upper right-hand corner of your Enhanced Chart. (Default)

ExtendMaxSpace Property

Specifies the maximum pixel length of the space between the line to be extended and the intersection point. If the space is greater than this number, the line will not be extended to the intersection point.

Syntax

*object.***ExtendMaxSpace** [=*Integer*]

Properties

The **ExtendMaxSpace** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The maximum number of pixels allowed between the line to be extended and the intersection point.

ExtendType Property

Specifies the line extension option to apply to all line objects.

Syntax

*object.***ExtendType** [=*enumExtendType*]

Properties

The **ExtendType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumExtendType	When to extend a line to the intersection.

Settings

The settings for *enumExtendType* are:

Constant	Value	Description
Always	0	Always extend lines.
ShorterthanHalf	1	Extend only when the extension is shorter than half the line.
ShorterThanSpecified	2	Extend only when the extension is shorter than the specified pixels.

F

FadeColor Property

Specifies the fade color of an object when the **FillStyle** pattern is *Gradient*.

Syntax

object.**FadeColor** [= *Long*]

Properties

The **FadeColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the object's fade color.

FadeType Property

Specifies the type of fade effect used for the object's gradient fill.

Syntax

object.**FadeType** [=*enumFadeType*]

Properties

The **FadeType** propertysyntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumFadeType	The pattern to display.

Settings

The settings for *enumFadeType* are:

Constant	Value	Description
Linear	0	Linear
Reflected	1	Reflected
Radial	2	Radial
Concentric	3	Concentric

Remarks

The **GradientAngle** property is useful when the **FadeType** is either *Linear* or *Reflected*.

FailedSource Property

Returns the source of a failed connection attempt.

Syntax

*object.***FailedSource**

Properties

The **FailedSource** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

FailedSource is a read-only property of type *String*.

When a user calls **SetSource** with "AI1+AI2" where AI1 exists and AI2 does not, **FailedSource** would contain AI2 (provided that the *bUseAnyway* parameter was not set to **True**).

FetchDataSetLimits Property

Allows the low and high limits of the selected data source to be retrieved at run-time for a GeneralDataSet Object or RealTimeSPCDataSet object. Disable this property to use the High and Low Limit properties instead.

Syntax

*object.***FetchDataSetLimits** [= *Boolean*]

Properties

The **FetchDataSetLimits** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to fetch the high and low limits of the data set.

Settings

The settings for *Boolean* are:

Value	Description
True	Fetch the limits. (Default)
False	Do not fetch the limits.

FetchPenLimits Property

Specifies whether to fetch the limits for the specified **Pen**.

Syntax

*object.***FetchPenLimits** [= *Boolean*]

Properties

The **FetchPenLimits** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to fetch the limits.

Settings

The settings for *Boolean* are:

Value	Description
True	Fetch the limits. (Default)
False	Do not fetch the limits.

Remarks

FetchPenLimits is a one-shot property in the Run-time environment.

FileName Property

Returns the file name of the specified **Document**.

Syntax

*object.***FileName**

Properties

The **FileName** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

FileName is a read-only property of type *String*.

FillStyle Property

Specifies the pattern that will be used to fill the interior of the shape.

Syntax

*object.***FillStyle** [= *enumFillStyle*]

Properties

The **FillStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumFillStyle	The pattern to display.

Settings

The settings for *enumFillStyle* are:

Constant	Value	Description
FillStyleSolid	0	Solid.
FillStyleHollow	1	Hollow.
FillStyleHorizontal	2	Horizontal.
FillStyleVertical	3	Vertical.
FillStyleDownDiagonal	4	Downward diagonal.
FillStyleUpDiagonal	5	Upward diagonal.
FillStyleCrossHatch	6	Crosshatch.
FillStyleDiagonalCrossHatch	7	Diagonal crosshatch.
FillStyleGradient	8	Gradient.

Remarks

The **FillStyle** pattern is generated by alternating the **BackgroundColor** and **ForegroundColor** of the shape, except when the **FillStyle** pattern is Gradient. When the **FillStyle** pattern is Gradient, the ForegroundColor and **FadeColor** of the shape alternate.

FilterString Property

Specifies the expression on which the **Alarm Summary** object is filtering.

Syntax

*object.***FilterString** [= *String*]

Properties

The **FilterString** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The expression used to filter.

FixedDate Property

Specifies a fixed date for the Chart,PenGeneralDataSet Objector ,Formatted Object,Line,Lookup Object,.

Syntax

*object.***FixedDate** [= *DateTime*]

Example

#1/1/2000 12:00:00 AM#

Properties

The **FixedDate** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
DateTime	The fixed date.

Remarks

FixedDateis a one-shot property.

The default value for FixedDate is the date at which the Pen or Chart was created. Although this property is passed as a complete Date and Time datatype, the Time portion is ignored.

This property is not impacted by any Global Time Control property settings.

FixedTime Property

Specifies a fixed time in the Chart,PenGeneralDataSet Object, or Formatted Object ,Line,Lookup Object,.

Syntax

*object.***FixedTime** [= *DateTime*]

Example

```
#1/1/2000 12:00:00 AM#
```

Properties

The **FixedTime** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
DateTime	The fixed time.

Remarks

FixedTime is a one-shot property.

The default value for the **FixedTime property is the time at which the**Pen or Chart was created. Although this property is passed as a complete Date and Time datatype, the Date portion is ignored.

This property is not impacted by any Global Time Control property settings.

FixPath Property

Returns the requested iFIX system path for the specified path index.

Syntax

object.**FixPath**(*ePathID as PathID*)

Properties

The **FixPath** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
index	The path index.

Settings

The settings for *PathID* are:

Constant	Value	Description
Base_Path	0	Base path.
Pic_Path	1	Picture path.
ToolBar_Path	2	Toolbar path.
Documents_Path	3	Documents path.
Schedule_Path	4	Schedule path.
Local_Path	5	Local path.
Pdb_Path	6	PDB path.

Nls_Path	7	NLS path.
App_Path	9	Application path.
Htc_Path	10	HTC path.
Htd_Path	11	HTD path.
Alm_Path	12	Alarm path.
Rcm_Path	13	RCM path.
Rcc_Path	14	RCC path.
Project_Path	15	iFIX project path.

NOTE: To access these constants, add a reference to the iFIX Global System Information Type Library in the Visual Basic Editor.

Remarks

FixPath is a read-only property of type *String*.

Font Property

Specifies the Font to be displayed in the **ExpressionEditor**.

Syntax

*object.***Font** [= *StdFont*]

Properties

The **Font** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
StdFont	The font to display in the **ExpressionEditor**.

FontName Property

Specifies the font family which will be used to display text.

Syntax

*object.***FontName** [= *String*]

Properties

The **FontName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The name of the font.

FontSize Property

Specifies the point size for text display.

Syntax

*object.***FontSize** [= *Long*]

Properties

The **FontSize** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The point size of the font.

FontStyle Property

Determines if the text will display a Bold, Italic, or combination of styles.

Syntax

*object.***FontStyle** [= *enumFontStyle*]

Properties

The **FontStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumFontStyle	The style to display.

Settings

The settings for *enumFontStyle* are:

Constant	Value	Description
Regular	0	The text is displayed as regular.
Bold	1	The text is displayed as bold.
Italic	2	The text is displayed as italic.
BoldItalic	3	The text is displayed as both bold and italic.

ForceVerticalPoints Property

Sets whether point labels are forced into vertical, horizontal, or slanted orientation, or whether the orientation is automatically determined by the Enhanced Chart object (HistogramChart, LineChart, SPCBarChart, or XYChart Object).

Syntax

*object.***ForceVerticalPoints** [= *enumForceVerticalPoints*]

Properties

The **ForceVerticalPoints** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumForceVerticalPoints	An enumeration that represents the point label display settings in the Enhanced Chart: **Valid entries:** 0 – PointLabelAuto 1 – PointLabelVertical 2 – PointLabelHorizontal 3 – PointLabelSlanted

ForegroundColor Property

Specifies the color to be used to fill the interior of a shape.

Syntax

*object.***ForegroundColor** [= *Long*]

Properties

The **ForegroundColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the shape's foreground color.

ForegroundEdgeColor Property

Specifies the foreground edge color of the **Chart**.

Syntax

*object.***ForegroundEdgeColor** [= *Long*]

Properties

The **ForegroundEdgeColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the chart's foreground edge color.

ForegroundEdgeStyle Property

Specifies the foreground edge style of the **Chart**.

Syntax

object.**ForegroundEdgeStyle** [= *enumEdgeStyle*]

Properties

The **ForegroundEdgeStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumEdgeStyle	The edge style to display.

Settings

The settings for *enumEdgeStyle* are:

Constant	Value	Description
EdgeStyleSolid	0	Solid.
EdgeStyleDash	1	Dash.
EdgeStyleDot	2	Dot.
EdgeStyleDashDot	3	Dash-Dot.
EdgeStyleDashDotDot	4	Dash-Dot-Dot.
EdgeStyleNone	5	No border.
EdgeStyleInsideFrame	6	Inside Frame.

ForegroundEdgeWidth Property

Specifies the foreground edge width of the **Chart**.

Syntax

object.**ForegroundEdgeWidth** [= *Long*]

Properties

The **ForegroundEdgeWidth** property syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
Long	The width of the edge.

Format Property

Specifies the C sprintf format string into which the input is formatted for the Format object.

Syntax

*object.***Format** [= *String*]

Properties

The **Format** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The standard C sprintf format string.

FormatDataType Property

Specifies whether the format object is alphanumeric or numeric.

Syntax

*object.***FormatDataType**

Properties

The **FormatDataType** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Settings

FormatDataType is a read-only property of type *enumFormatDataType*. The settings for *enumFormatDataType* are:

Constant	Value
Alphanumeric	0
Numeric	1

FullName Property

Returns the full path name for the WorkSpace executable file or the specified **Document** object.

Syntax

*object.***FullName**

Properties

The **FullName** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

FullName is a read-only property of type *String*.

FullScreen Property

Specifies whether the client area of the open document covers the entire screen.

Syntax

*object.***FullScreen** [= *Boolean*]

Properties

The **FullScreen** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the client area of the document covers the entire screen.

Settings

The settings for *Boolean* are:

Value	Description
True	The current document covers the entire screen.
False	The current document does not cover the entire screen. (Default)

FullyQualifiedName Property

Returns the containment hierarchy for the specified object.

Syntax

*object.***FullyQualifiedName**

Properties

The **FullyQualifiedName** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

FullyQualifiedName is a read-only property of type *String*.

G-J

GlobalDuration Property

Specifies the time duration, in seconds, to display historical data in run mode.

NOTE: When using any of the Global Time Control or Playback properties, you need to call the System.GlobalTimerApply method after setting the property.

Syntax

object.**GlobalDuration** [= *Long*]

Properties

The **GlobalDuration** property syntax has these parts:

3	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The length of time, in seconds, for which the object displays data.

GlobalEndTime Property

Specifies the end time displayed in the Global Time Control for all historical data sources in run mode. This property is not applied until the GlobalTimerApply method is called. The date string is formatted according to the short date format in the Regional and Language Options in the Control Panel.

NOTE: When using any of the Global Time Control or Playback properties, you need to call the System.GlobalTimerApply method after setting the property.

Syntax

object.**GlobalEndTime** [= *String*]

Properties

The **GlobalEndTime** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	Date. The end time property of the Global Time Control.

Remarks

The value for EnableGlobalEndTime must be set to True before you can set the value for this property.

GlobalFastScrollOption Property

Allows you to specify whether the fast scroll rate for historical data in run mode is in days, hours, minutes, or seconds.

NOTE: When using any of the Global Time Control or Playback properties, you need to call the System.GlobalTimerApply method after setting the property.

Syntax

*object.***GlobalFastScrollOption** [= *enumGlobalFastScroll*]

Properties

The **GlobalFastScrollOption** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumGlobalFastScroll	The units in which the Global Time Control can be scrolled when the System object's EnableGlobalScrollPercentage property is False. The enumeration values are as follows: 0 = Days 1 = Hours 2 = Mins 3 = Secs

Remarks

The EnableGlobalScrollPercentage property must be disabled before you can use this property.

GlobalHistoricalUpdateRate Property

Allows you to specify how quickly historical data sources update in run mode.

NOTE: When using any of the Global Time Control or Playback properties, you need to call the System.GlobalTimerApply method after setting the property.

Syntax

*object.***GlobalHistoricalUpdateRate** [= *Long*]

Properties

The **GlobalHistoricalUpdateRate** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Specifies, in seconds, how quickly an object updates in run mode.

GlobalMovingEndTime Property

Returns the GlobalMovingEndTime. This is a read-only property.

Syntax

*object.***GlobalMovingEndTime**

Properties

The **GlobalMovingEndTime** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies to list.

Returns

Returns the date in string format.

Remarks

GlobalMovingEndTime is a read-only property of type Date.

GlobalMovingStartTime Property

Returns the GlobalMovingStartTime. This is a read-only property.

Syntax

*object.***GlobalMovingStartTime**

Properties

The **GlobalMovingStartTime** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies to list.

Returns

Returns the date in string format.

Remarks

GlobalMovingStartTime is a read-only property of the type Date.

GlobalOutputToggle Property

Specifies whether the table has a global toggle source.

Syntax

object.**GlobalOutputToggle** [= *Boolean*]

Properties

The **GlobalOutputToggle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the table has a global toggle source.

Settings

The settings for *Boolean* are:

Value	Description
True	The table has a global toggle.
False	The table does not have a global toggle. (Default)

Remarks

If **GlobalOutputToggle** is set to **True**, the output will be toggled based on a different data source (blink on a new alarm, for example).

GlobalPlayBack Property

Specifies whether the playback of Historian data is allowed in run mode of the iFIX WorkSpace. GlobalHistoricalUpdateRate, GlobalSlowScrollRate, and GlobalFastScrollRate properties are not applicable when the GlobalPlayBack property is enabled.

NOTE: When using any of the Global Time Control or Playback properties, you need to call the System.GlobalTimerApply method after setting the property.

Syntax

object.**GlobalPlayBack** [= *Boolean*]

Properties

The **GlobalPlayBack** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the picture is enabled for playback on the Historical tab in run mode for the iFIX WorkSpace.

Settings

The settings for *Boolean* are:

Value	Description
True	Playback is enabled in a run mode.
False	Playback is not enabled (Default) in run mode.

GlobalPlayBackFrameSize Property

Allows you to specify the size of data to fetch from Historian when playback is enabled. . The maximum time for playback is 24 hours.

NOTE: When using any of the Global Time Control or Playback properties, you need to call the System.GlobalTimerApply method after setting the property.

Syntax

object.**GlobalPlayBackFrameSize** [= *Long*]

Properties

The **GlobalPlayBackFrameSize** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Specifies playback data by a specified frame size in seconds.

GlobalPlayBackNumberOfFrames Property

Specifies the number of times the historical data is being fetched if playback is enabled. This property is read-only.

Syntax

*object.***GlobalPlayBackNumberOfFrames** [= *Long*]

Properties

The **GlobalPlayBackNumberOfFrames** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Specifies the number of frames per second.

GlobalPlayBackSpeed Property

Allows you to specify a speed factor for a system generated frame size for historical data when playback is enabled.

NOTE: When using any of the Global Time Control or Playback properties, you need to call the System.GlobalTimerApply method after setting the property.

Syntax

*object.***GlobalPlayBackSpeed** [= *Long*]

Properties

The **GlobalPlayBackSpeed** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Specifies the playback speed factor: 1, 10, 20, 40, 50, 100, or 200 (with the largest frame size being 200x). The higher the speed, the larger the frame size.

GlobalSlowScrollOption Property

Allows you to specify whether the slow scroll rate for historical data in run mode is in days, hours, minutes, or seconds.

NOTE: When using any of the Global Time Control or Playback properties, you need to call the System.GlobalTimerApply method after setting the property.

Syntax

object.**GlobalSlowScrollOption** [= *enumGlobalSlowScroll*]

Properties

The **GlobalSlowScrollOption** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumGlobalSlowScroll	The units in which the Global Time Control can be scrolled when the System object's EnableGlobalScrollPercentage property is False. The enumeration values are as follows: 0 = Days 1 = Hours 2 = Mins 3 = Secs

Remarks

The EnableGlobalScrollPercentage property must be disabled before you can use this property.

GlobalSlowScrollRate Property

Specifies how slowly historical data can be scrolled in run mode.

NOTE: When using any of the Global Time Control or Playback properties, you need to call the System.GlobalTimerApply method after setting the property.

Syntax

object.**GlobalSlowScrollRate** [= *Double*]

Properties

The **GlobalSlowScrollRate** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The rate at which the object can be scrolled. The unit of measure depends on if the EnableGlobalScrollPercentage Property is enabled. If it is, then the unit of measure is a percentage. If it is not, then it is the unit of measure specified in the GlobalSlowScrollOption property.

Remarks

This property cannot be set to a value greater than the GlobalFastScrollRate.

GlobalStartTime Property

Specifies the start time of the Global Time Control for historical data sources in all open pictures in run mode. This property is not applied until the GlobalTimerApply method is called. The date string is formatted according to the short date format in the Regional and Language Options in the Control Panel.

NOTE: When using any of the Global Time Control or Playback properties, you need to call the System.GlobalTimerApply method after setting the property.

Syntax

*object.***GlobalStartTime** [= *String*]

Properties

The **GlobalStartTime** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies to list.
String	Date. The starting time for the Global Time Control.

GlobalTimerPause Property

Pauses the configured global time control settings to historical data sources in all open pictures in run mode.

Syntax

*object.***GlobalTimerPause** [= Boolean]

Properties

The **GlobalTimerPause** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to play or pause the Global Time Control.

Settings

The settings for *Boolean* are:

Value	Description
True	The Global Time Control is in a paused state.
False	The Global Time Control is in a play state.

GlobalTimeSync Property

Specifies whether the picture will be sensitive to changes in the Global Time Control. This property is read-only.

Syntax

object.**GlobalTimeSync** [= *Boolean*]

Properties

The **GlobalTimeSync** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the picture is enabled for time synchronization with the Global Time Control.

Settings

The settings for *Boolean* are:

Value	Description
True	The Global Time Control is enabled (Default).
False	The Global Time Control is not enabled.

GlobalToggle Property

Specifies the value that will be displayed when the source evaluates to **True** if the user has set up a global toggle source.

Syntax

object.**GlobalToggle** [= *Variant*]

Properties

The **GlobalToggle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	The value to display when the source evaluates to **True**.

Remarks

The value will blink between the current value and this **GlobalToggle** value. This overrides any blink set up specifically in the table.

Gradient Property

Enables the gradient effect in the picture background. The gradient blends the ForegroundColor and BackgroundColor of the picture.

Syntax

*object.***Gradient**

Properties

The **Gradient** propertysyntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

GlobalFastScrollRate Property

Specifies how quickly the historical data can be scrolled in run mode.

NOTE: When using any of the Global Time Control or Playback properties, you need to call the System.GlobalTimerApply method after setting the property.

Syntax

*object.***GlobalFastScrollRate** [= *Double*]

Properties

The **GlobalFastScrollRate** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The rate at which the object can be scrolled. The unit of measure depends on if the EnableGlobalScrollPercentage Property is enabled. If it is, then the unit of measure is a percentage. If it is not, then it is the unit of measure specified in the GlobalFastScrollOption property.

Remarks

This property cannot be set to a value less than the GlobalSlowScrollRate.

GradientAngle Property

Specifies the angle (in radians or degrees) of the object's gradient fill.

Syntax

object.**GradientAngle** [=*Double*]

Properties

The **GradientAngle** propertysyntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The angle of the object's gradient fill.

Remarks

In Configuration mode, **GradientAngle** changes as you rotate the object.

The units to be used when creating the angle is specified as either degrees or radians, depending on the value of the **AngleUnits** property.

GraphBackColor Property

Allows you to specify the background color of the graph in an Enhanced Chart.

Syntax

object.**GraphBackColor** [=*Long*]

Properties

The **GraphBackColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	An Integer representing the color value of the background of a HistogramChart, LineChart, SPCBarChart, or XYChart Object.

GraphForeColor Property

Allows you to specify the grid color of the graph (foreground color) in an Enhanced Chart. The grid lines and tick marks are drawn in this color.

Syntax

object.**GraphForeColor** [=*Long*]

Properties

The **GraphForeColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	An Integer representing the color value of the grid foreground color of a HistogramChart, LineChart, SPCBarChart, or XYChart Object.

GraphPlusTable Property

Sets whether the Enhanced Chart displays a graph, table, or both a graph and table.

Syntax

*object.***GraphPlusTable** [=*enumGraphPlusTable*]

Properties

The **GraphPlusTable** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumGraphPlusTable	An enumeration representing the graph and table display preferences in a HistogramChart, LineChart, SPCBarChart, or XYChart Object: **Valid entries:** 0 – Graph 1 – Table 2 – BothGraphPlusTable

GraphPlusTableMenu Property

Sets the visibility of the menu in an Enhanced Chart with both a graph and table.

Syntax

*object.***GraphPlusTableMenu** [=*enumGraphPlusTableMenu*]

Properties

The **GraphPlusTableMenu** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumGraphPlusTableMenu	An enumeration representing the menu display preference in a HistogramChart, LineChart, SPCBarChart, or XYChart Object: **Valid entries:**

	0 – Hide 1 – Show 2 – Greyed

GridEnabled Property

Specifies whether the grid is being used for the specified **Picture** or **DynamoSet**.

Syntax

*object.***GridEnabled** [= *Boolean*]

Properties

The **GridEnabled** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the grid is displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The grid is displayed.
False	The grid is not displayed. (Default)

GridInFront Property

Specifies that the grid appears in front of the data in an Enhanced Chart.

Syntax

*object.***GridInFront** [= *Boolean*]

Properties

The **GridInFront** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the grid displays in front of the data in a HistogramChart, LineChart, SPCBarChart, or XYChart Object.

Settings

The settings for *Boolean* are:

Value	Description
True	The grid is displayed in front of the data.
False	The grid is not displayed in front of the data. (Default)

GridInterval Property

Specifies the amount of pixels between grid points.

Syntax

*object.***GridInterval** [= *Long*]

Properties

The **GridInterval** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The amount of pixels between grid points.

GridLinesToShow Property

Specifies which grid lines to show in an Enhanced Chart.

Syntax

*object.***GridLinesToShow** [=*enumGridLinesToShow*]

Properties

The **GridLinesToShow** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumGridLinesToShow	An enumeration representing the grid line display preference in a HistogramChart, LineChart, SPCBarChart, or XYChart Object: **Valid entries:** 0 – GridBothXY 1 – GridYAxis 2 – GridXAxis 3 – GridNone

GridStyle Property

Specifies the style of the gird lines in an Enhanced Chart.

Syntax

*object.***GridStyle** [=*enumGridStyle*]

Properties

The **GridStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumGridStyle	An enumeration representing the grid style display preference in a HistogramChart, LineChart, SPCBarChart, or XYChart Object: **Valid entries:** 0 – GridThin 1 – GridThick 2 – GridDot 3 – GridDash 4 – GridOnePixel

GridWidth Property

Specifies the width of each grid line in a **Chart**.

Syntax

*object.***GridWidth** [= *Long*]

Properties

The **GridWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The width of each grid line in the chart.

Groups Property

Returns a user defined collection of **Group (DataSystem)** objects. **Group (DataSystem)** objects can be added and removed from this collection using the **Add** and **Remove** methods. Group names must be unique.

Syntax

*object.***Groups**

Properties

The **Groups** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Groups is a read-only property of type *Object*.

Height Property

Specifies the height, in postscript points or logical units, of the specified object.

Syntax

*object.***Height** [= *Double*]

Properties

The **Height** property syntax has these parts:

Remarks

For shapes, the units are in postscript points or logical units defined by the **Picture** document size. The coordinate systems allow pictures to be developed and saved independently of screen resolution. It also supports panning and zooming.

HelpFile Property

Specifies the user defined context sensitive help file which should be associated with the specified document.

Syntax

*object.***HelpFile** [= *String*]

Properties

The **HelpFile** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The help file name.

HelpPath Property

Returns the path of the iFIX Help files.

Syntax

*object.***HelpPath**

Properties

The **HelpPath** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

HelpPath is a read-only property of type *String*.

HideMathFunctionsButton Property

Specifies whether to hide the Mathematical Functions button in the **ExpressionEditor** dialog box.

Syntax

*object.***HideMathFunctionsButton** [= *Boolean*]

Properties

The **HideMathFunctionsButton** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Settings

The settings for *Boolean* are:

Value	Description
True	The Mathematical Functions button is hidden.
False	The Mathematical Functions button is not hidden. (Default)

HiDisplay Property

Specifies the high display limit of the **TimeAxis** or **ValueAxis**.

TimeAxis Syntax

*object.***HiDisplay** [= *Date*]

Properties

The **HiDisplay** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Date	The time and date to display for the **Time Axis**.

ValueAxis Syntax

object.**HiDisplay** [= *Double*]

Properties

The **HiDisplay** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The value to display for the **Value Axis**.

HighestDataValue Property

Specifies the highest value for the specified **Pen**.

Syntax

object.**HighestDataValue**[= *Double*]

Properties

The **HighestDataValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The highest value for the **Pen**.

HighlightEnabled Property

Specifies whether the specified shape should have a "highlight" rectangle drawn around it when the mouse passes over it in the Run-time environment.

Syntax

object.**HighlightEnabled**[= *Boolean*]

Properties

The **HighlightEnabled** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the shape should appear highlighted.

Settings

The settings for *Boolean* are:

Value	Description
True	The object should appear highlighted in the Run-time environment when the mouse passes over it.
False	The object should not appear highlighted in the Run-time environment when the mouse passes over it. (Default)

Remarks

The default setting for **HighlightEnabled** is **False** for all objects except Ole Controls.

HighlightEnabled is set to **True** in the following cases:

- **IsSelectable** is set to **True**
- The user configures in-line Data Entry
- When the user writes a script for **MouseUp**, **MouseDown**, **Click**, **DblClick** or **MouseMove** events

The user then has the option of setting **HighlightEnabled** back to **False**.

HighlightedDatasource Property

Specifies the datasource of the currently highlighted object.

Syntax

*object.***HighlightedDatasource**

Properties

The **HighlightdDatasource** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

HiInValue Property

Specifies the upper limit on the input value.

Syntax

*object.***HiInValue** [= *Variant*]

Properties

The **HiInValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	The high input value.

HiLimit Property

Specifies the upper limit for the specified **Pen** or **GeneralDataSet** object.

Syntax

object.**HiLimit** [= *Double*]

Properties

The **HiLimit** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The high limit.

HiOutValue Property

Specifies the upper limit on the output value.

Syntax

object.**HiOutValue** [= *Variant*]

Properties

The **HiOutValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	The high output value.

HistMode Property

Allows you to determine how iFIX selects data from a historical data source and displays it in the chart, and determines what each displayed value represents.

Syntax

*object.***HistMode** [= *enumHistMode*]

Properties

The **HistMode** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumHistMode	An enumeration representing how iFIX selects data from a historical data source and displays it in a GeneralDataSet Object: **Valid entries:** 0 – HDS_Sample 1 – HDS_Avg 2 – HDS_High 3 – HDS_Low 4 – HDS_Interpolated 5 - HDS_Trend 7 - HDS_StandardDeviation 8 - HDS_Total 13 – HDS_CurrentValue

HistoricalSampleType Property

Specifies the data retrieval mode for the specified **Pen**.

Syntax

*object.***HistoricalSampleType** [= *enumHTRMode*]

Properties

The **HistoricalSampleType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumHTRMode	The data retrieval mode.

Settings

The settings for *enumHTRMode* are:

Constant	Value	Description
Sample	0	Sample.
High	1	High.
Low	2	Low.
Average	3	Average.
Interpolated	4	Interpolated.

HistUpdateRate Property

Sets the historical update rate, in seconds, of the Enhanced Chart.

Syntax

*object.***HistUpdateRate** [= *Long*]

Properties

The **HistUpdateRate** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The historical update rate, in seconds, for the HistogramChart, LineChart, SPCBarChart, or the XYChart Object.

Remarks

This property is not impacted by any Global Time Control property settings.

HorizontalFillDirection Property

Specifies a value representing the direction of a shape's horizontal fill.

Syntax

*object.***HorizontalFillDirection** [= *enumHorizontalDirection*]

Properties

The **HorizontalFillDirection** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumHorizontalDirection	The horizontal direction from which to fill.

Settings

The settings for *enumHTRMode* are:

Constant	Value	Description
HorizontalFromLeft	0	Fill from the left.
HorizontalFromRight	1	Fill from the right.
HorizontalFromCenter	2	Fill outward from the center.

HorizontalFillPercentage Property

Specifies the percentage to horizontally fill a shape.

Syntax

*object.***HorizontalFillPercentage** [= *Double*]

Properties

The **HorizontalFillPercentage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The percentage to fill the shape.

HorizontalGridColor Property

Specifies the color of the horizontal grid lines.

Syntax

*object.***HorizontalGridColor** [= *Long*]

Properties

The **HorizontalGridColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the horizontal grid lines.

HorizontalGridStyle Property

Specifies the style of the horizontal grid lines.

Syntax

*object.***HorizontalGridStyle** [= *enumEdgeStyle*]

Properties

The **HorizontalGridStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumEdgeStyle	The style to display for the grid lines.

Settings

The settings for *enumEdgeStyle* are:

Constant	Value	Description
EdgeStyleSolid	0	Solid.
EdgeStyleDash	1	Dash.
EdgeStyleDot	2	Dot.
EdgeStyleDashDot	3	Dash-Dot.
EdgeStyleDashDotDot	4	Dash-Dot-Dot.
EdgeStyleNone	5	No border.
EdgeStyleInsideFrame	6	Inside Frame.

HorizontalPosition Property

Specifies a shape's distance, in postscript points or logical units, from the left of the **Picture**or **DynamoSet**.

Syntax

*object.***HorizontalPosition** [= *Double*]

Properties

The **HorizontalPosition** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The shape's horizontal position.

Remarks

For shapes, the units are in postscript points (for the Enhanced Coordinate System) or logical units (for the Logical Coordinate System) as defined by the **Picture** document size.

HorizontalScaleDirection Property

Specifies if the direction in which the specified shape will expand or contract when the **HorizontalScalePercentage** property is changed.

Syntax

*object.***HorizontalScaleDirection** [= *enumHorizontalDirection*]

Properties

The **HorizontalScaleDirection** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumHorizontalDirection	The direction to scale.

Settings

The settings for *enumHorizontalDirection* are:

Constant	Value	Description
HorizontalFromLeft	0	Scale from the left.
HorizontalFromRight	1	Scale from the right.
HorizontalFromCenter	2	Scale from the center outward.

HorizontalScalePercentage Property

Specifies the scale percentage to apply to a shape's width.

Syntax

*object.***HorizontalScalePercentage** [= *Double*]

Properties

The **HorizontalScalePercentage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The percentage to scale the shape horizontally.

Remarks

In the Configuration environment, **HorizontalScalePercentage** will not be set back to 100 until the object is de-selected. Therefore, while selected, the object will contain it's current percentage value relative to the size of the object when it was last selected. Once de-selected, the object's **HorizontalScalePercentage** property will be reset back to 100.

In the Run-time environment, animating the **HorizontalScalePercentage** property modifies the object's width based on the size of the object when it initially came off disk.

An object's scale percentage can be negative. This causes the object to flip over its left axis. This effect is useful for creating differential bar graphs by using an expression in the data source that takes the value and subtracts a setpoint. The resulting difference from the setpoint can be used to by the **HorizontalScalePercentage** property. For example, you could animate a color table to change color based on the sign of a result.

ImageCount Property

Returns the number of images loaded in the **Bitmap**.

Syntax

object.**ImageCount**

Properties

The **ImageCount** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ImageCount is a read-only property of type *Integer*.

IncludeDataLabels Property

Specifies whether the data points include labels. This property only applies to XY Enhanced Charts.

Syntax

object.**IncludeDataLabels** [= *Boolean*]

Properties

The **IncludeDataLabels** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the data points include a label.

Settings

The settings for *Boolean* are:

Value	Description
True	The data points are labeled.
False (default)	The data points are not labeled.

Index Property

Returns the one-based index in the collection of the specified object.

Syntax

object.**Index**

Properties

The **Index** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Index is a read-only property of type *Long*.

InitialValue Property

Specifies the inital value for the specified variable.

Syntax

object.**InitialValue** [= *Variant*]

Properties

The **InitialValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	The initial value.

InputValue Property

Specifies the data which is to be transformed by the animation object.

Syntax

object.**InputValue** [= *Variant*]

Properties

The **InputValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	The input value.

Remarks

If you are manipulating a color threshold table within a script, the variable must be of the same type as the threshold table.

NOTE: If you are reading an Input Value after a source change, you must allow time for the Input Value to be updated. If iFIX has not had sufficient time to establish the new connection, the first attempt to obtain the Event object's Input Value will result in an Automation Error. The amount of time that iFIX requires to establish the connection depends upon the scan time of the Event object source tag. If you need the data immediately, read from an already connected object.

Additionally, if you set a steady state tag (a tag with a static value) as the source twice in a row, there will be no change in the Input Value and you will get an Automation Error.

Interval Property

Specifies the length of time between data points for the GeneralDataSet, Chart, or Pen, or how often the Timer or Event object is fired.

Chart and Pen Syntax

object.**Interval** [= *Long*]

Properties

The **Interval** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The time between points in seconds.

Timer and Event Syntax

object.**Interval** [= *Variant*]

Properties

The **Interval** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	How often the object's configured event is fired.

Remarks

Interval accepts a *DATE* for the Timer object and a *Long* (specifying the number of milliseconds) for the Event object. For the Event object, **Interval** only applies if the EventType property is *WhileTrue* or *WhileFalse*

IntervalMilliseconds Property

Contains the millisecond component of the time interval between data points.

Syntax

object.**IntervalMilliseconds** [= *Long*]

Properties

The **IntervalMilliseconds** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The time between points in milliseconds.

Remarks

The **IntervalMillisecond** default value is 0. This is used when precision under 1 second is desired for the returned data.

IsDirty Property

Returns whether the contents of the object have changed since the last time the document was saved in the Configuration envrionment.

Syntax

*object.***IsDirty**

Properties

The **IsDirty** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the object has been modified.

Return Values

The **IsDirty** property return values are:

Value	Description
True	The object has been modified since the last time the document was saved.
False	The object has not been modified since the last time the document was saved.

Remarks

IsDirty is a read-only property of type *Boolean*.

IsInterpolated Property

Controls whether interpolation should be used for the specified data set.

Syntax

*object.***IsInterpolated** =[Boolean]

Properties

The **IsInterpolated** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether interpolation is used for the dataset given in the object.

Settings

The settings for *Boolean* are:

Value	Description
True	The data set uses interpolation.
False	The data set does not use interpolation.

IsModifiable Property

Specifies whether an object can be modified.

Syntax

object.**IsModifiable**

Properties

The **IsModifiable** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the object can be modified.

Settings

The settings for *Boolean* are:

Value	Description
True	The object can be modified.
False	The object cannot be modified.

IsSelectable Property

Specifies whether the specified object can be selected in the Run-time environment.

Syntax

*object.***IsSelectable**[= *Boolean*]

Properties

The **IsSelectable** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the object can be selected.

Settings

The settings for *Boolean* are:

Value	Description
True	The object can be selected in the Run-time environment.
False	The object can not be selected in the Run-time environment. (Default)

IsSelected Property

Returns whether the specified object is selected.

Syntax

*object.***IsSelected**

Properties

The **IsSelected** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Values

The **IsSelected** property return values are:

Value	Description
True	The object is selected.
False	The object is not selected.

Remarks

IsSelected is a read-only property of type *Boolean*.

Item Property

Returns a member of the specified Collection object.

Syntax

*object.***Item** (*vtIndex*)

Properties

The **Item** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
tIndex	An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's **Count** property.

Remarks

Item is a read-only property of type *Object*.

Justification Property

Specifies whether the text format is left, center, or right-justified.

Syntax

*object.***Justification** [=*enumJustification*]

Properties

The **Justification** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumJustification	The text justification.

Settings

The settings for *enumJustification* are:

Constant	Value
LeftJustify	0
CenterJustify	1
RightJustify	2

K-L

KeyCode Property

The ASCII value of the primary key used in defining the key combination.

Syntax

*object.***KeyCode** [= *KeyCode*]

Properties

The **KeyCode** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
KeyCode	Integer. The new ASCII value.

LabelBold Property

Specifies whether the labels in an Enhanced Chart are bold.

Syntax

*object.***LabelBold** [= *Boolean*]

Properties

The **LabelBold** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the label in a HistogramChart, LineChart, SPCBarChart, or XYChart Object displays in bold.

Settings

The settings for *Boolean* are:

Value	Description
True	The label displays in bold.
False	The label does not display in bold. (Default)

LabelColor Property

Specifies the color of the labels for the axes in a **Chart**.

Syntax

*object.***LabelColor** [= *Long*]

Properties

The **LabelColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the label color.

LabelFont Property

Specifies the font face of labels in an Enhanced Chart.

Syntax

object.**LabelFont** [= *String*]

Properties

The **LabelFont** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The name of the font. By default, the font face is "Arial."

LabelItalic Property

Specifies whether the labels in an Enhanced Chart appear in italics.

Syntax

object.**LabelItalic** [= *Boolean*]

Properties

The **LabelItalic** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the label in a HistogramChart, LineChart, SPCBarChart, or XYChart Object displays in italics.

Settings

The settings for *Boolean* are:

Value	Description
True	The label displays in italics.
False	The label does not display in italics. (Default)

LabelUnderline Property

Specifies whether the labels in an Enhanced Chart appear underlined.

Syntax

*object.***LabelUnderline** [= *Boolean*]

Properties

The **LabelUnderline** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the label in a HistogramChart, LineChart, SPCBarChart, or XYChart Object appears underlined.

Settings

The settings for *Boolean* are:

Value	Description
True	The label appears underlined.
False	The label does not appear underlined. (Default)

Layer Property

Specifies the hex mask of the layers the object is currently a part of.

Syntax

*object.***Layer** [= *Long*]

Properties

The **Layer** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The layer of the shape.

Remarks

The **Layer** property is used in conjunction with the **DisplayLayer** property of the **Picture** or **DynamoSet** to determine which objects are visible and accessible. For example, an object who's **Layer** property is set to 3 will not be visible when the **DisplayLayer** property of the document is set to 8. However, the object would be visible if the **DisplayLayer** property of the document is set to 1, 2 or any number who's first and second bits are set. The **DisplayLayer** property is also a mask of bits.

A **Layer** value of 3 does not signify that the object is a part of layer three, but rather a part of both layers one and two.

LCL Property

Specifies the lower control limit (LCL) for the real-time SPC data set.

Syntax

object.**LCL** [= *Double*]

Properties

The **LCL** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The lower control limit (LCL) for the RealTimeSPCDataSet Object.

Left Property

Specifies the distance, in screen percentage, between the left edge of the physical screen and the WorkSpace in which it is contained, or specifies the position of the left edge of the **Window** object.

Syntax

object.**Left** [= *Double*]

Properties

The **Left** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The distance in screen percentage.

LeftCenter Property

Returns the left center point of the shape's bounding rectangle as a **FixFloatPoint**.

Syntax

object.**LeftCenter**

Properties

The **LeftCenter** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

LeftCenter is a read-only property of type *Object*.

Legend Property

Returns the pointer to the **Legend** object for the specified **Pen**.

Syntax

*object.***Legend**

Properties

The **Legend** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Legend is a read-only property of type *Object*.

LegendAvgOver Property

Specifies the Average Over Range field for the specified **Legend**.

Syntax

*object.***LegendAvgOver** [= *String*]

Properties

The **LegendAvgOver** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The Average Over Range for the **Legend**.

LegendDesc Property

Specifies the Description field for the specified Legend.

Syntax

*object.***LegendDesc** [= *String*]

Properties

The **LegendDesc** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrEventName	String. The name of the event.

LegendHeadingLine Property

Returns the heading for the specified Legend.

Syntax

*object.***LegendHeadingLine**

Properties

The **LegendHeadingLine** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

LegendHeadingLine is a read-only property of type *String*.

LegendHigh Property

Specifies the High Limit field for the specified Legend.

Syntax

*object.***LegendHigh** [= *String*]

Properties

The **LegendHigh** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The High Limit for the **Legend**.

LegendHighOver Property

Specifies the High Over Range field for the specified Legend.

Syntax

*object.***LegendHighOver** [= *String*]

Properties

The **LegendHighOver** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The High Over Range for the **Legend**.

LegendInterval Property

Specifies the Interval field for the specified Legend.

Syntax

*object.***LegendInterval** [= *String*]

Properties

The **LegendInterval** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The Interval for the **Legend**.

LegendItemColor Property

Specifies the color of the specified legend item.

Syntax

*object.***LegendItemColor** [= *Long*]

Properties

The **LegendItemColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the items in the **Legend**.

LegendLow Property

Specifies the Low Limit field for the specified Legend.

Syntax

object.**LegendLow** [= *String*]

Properties

The **LegendLow** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The Low Limit for the **Legend**.

LegendLowOver Property

Specifies the Low Over Range field for the specified Legend.

Syntax

object.**LegendLowOver** [= *String*]

Properties

The **LegendLowOver** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The Low Over Range for the **Legend**.

LegendMode Property

Specifies the Mode field for the specified Legend.

Syntax

object.**LegendMode** [= *String*]

Properties

The **LegendMode** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The Mode for the **Legend**.

LegendTag Property

Specifies the data source connected to the Legend item for the specified Pen.

Syntax

object.**LegendTag** [= *String*]

Properties

The **LegendTag** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The data source.

LegendUnits Property

Specifies the engineering units for the data source connected to the Legend item for the specified Pen.

Syntax

object.**LegendUnits** [= *String*]

Properties

The **LegendUnits** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The engineering units.

Remarks

The **LegendUnits** property allows users to custom define engineering units that directly apply to the data that they are displaying.

LegendUser1 Property

A general purpose property which can contain any user value.

Syntax

object.**LegendUser1** [= *String*]

Properties

The **LegendUser1** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A user defined value.

Remarks

An example of how to set the **LegendUser1** property would be:

```
Set mypen = Chart1.Pens.Item(1)
Chart1.AddLegendItem "USER1", 2, 8
mypen.Legend.legenduser1 = "Square"
Set mypen = Nothing
```

LegendUser10 Property

A general purpose property which can contain any user value.

Syntax

*object.***LegendUser10** [= *String*]

Properties

The **LegendUser10** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A user defined value.

LegendUser2 Property

A general purpose property which can contain any user value.

Syntax

*object.***LegendUser2** [= *String*]

Properties

The **LegendUser2** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A user defined value.

LegendUser3 Property

A general purpose property which can contain any user value.

Syntax

*object.***LegendUser3** [= *String*]

Properties

The **LegendUser3** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A user defined value.

LegendUser4 Property

A general purpose property which can contain any user value.

Syntax

*object.***LegendUser4** [= *String*]

Properties

The **LegendUser4** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A user defined value.

LegendUser5 Property

A general purpose property which can contain any user value.

Syntax

*object.***LegendUser5** [= *String*]

Properties

The **LegendUser5** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A user defined value.

LegendUser6 Property

A general purpose property which can contain any user value.

Syntax

*object.***LegendUser6** [= *String*]

Properties

The **LegendUser6** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A user defined value.

LegendUser7 Property

A general purpose property which can contain any user value.

Syntax

*object.***LegendUser7** [= *String*]

Properties

The **LegendUser7** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A user defined value.

LegendUser8 Property

A general purpose property which can contain any user value.

Syntax

*object.***LegendUser8** [= *String*]

Properties

The **LegendUser8** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A user defined value.

LegendUser9 Property

A general purpose property which can contain any user value.

Syntax

*object.***LegendUser9** [= *String*]

Properties

The **LegendUser9** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A user defined value.

LegendValue Property

Specifies the current data value for the **Legend** for the specified **Pen**. For a historical pen, **LegendValue** specifies the pen's value when it crosses the Time Cursor.

Syntax

*object.***LegendValue** [= *String*]

Properties

The **LegendValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The current value.

Linear Property

Returns the **Linear** object for the specified **Pen**.

Syntax

*object.***Linear**

Properties

The **Linear** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Linear is a read-only property of type *Object*.

Lines Property

Returns the **Lines** collection for the specified **ScriptProcedure** object.

Syntax

*object.***Lines**

Properties

The **Lines** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Lines is a read-only property of type *Object*.

LinesofCode Property

Pass through property used to get and set the lines of code for the procedure.

Syntax

*object.***LinesofCode** [= *LinesofCode*]

Properties

The **LinesofCode** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
LinesofCode	String. The new lines of code.

LineType Property

Specifies the line type in a GeneralDataSet Object or RealTimeSPCDataSet object.

Syntax

*object.***LineType** [= *enumLineType*]

Properties

The **LineType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumLineType	An enumeration that represents the type of line to use for the GeneralDataSet Object or RealTimeSPCDataSet object. **Valid entries:** 1 – Line_Dash 3 – Line_DashDot 4 – Line_DashDotDot 2 – Line_Dot 32 – Line_ExtraThickDash 34 – Line_ExtraThickDashDot 35 – Line_ExtraThickDashDotDot 33 – Line_ExtraThickDot 11 – Line_ExtraThickSolid 20 – Line_MediumDash 22 – Line_MediumDashDot 23 – Line_MediumDashDotDot 21 – Line_MediumDot 5 – Line_MediumSolid 24 – Line_MediumThickDash 26 – Line_MediumThickDashDot 27 – Line_MediumThickDashDotDot 25 – Line_MediumThickDot 10 – Line_MediumThickSolid 16 – Line_MediumThinDash 18 – Line_MediumThinDashDot 19 – Line_MediumThinDashDotDot 17 – Line_MediumThinDot 9 – Line_MediumThinSolid 28 – Line_ThickDash 30 – Line_ThickDashDot 31 – Line_ThickDashDotDot 29 – Line_ThickDot 6 – Line_ThickSolid 0 – Line_ThinSolid

LockStartTime Property

Prevents the start time from changing when the computer's time zone is changed.

Syntax

*Object.***LockStartTime** [=*Boolean*]

Properties

The **LockStartTime** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to lock the start time.

Settings

The settings for Boolean are:

Value	Description
True	Lock the start time.
False	Do not lock the start time.

LoDisplay Property

Specifies the low display limit of the Time Axis or Value Axis.

TimeAxis Syntax

*object.***LoDisplay** [= *Date*]

Properties

The **LoDisplay** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Date	The time and date to display for the **Time Axis**.

ValueAxis Syntax

*object.***LoDisplay** [= *Double*]

Properties

The **LoDisplay** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The value to display for the **Value Axis**.

LoginGroup Property

Returns the first group name that the currently logged in user belongs to. If security is disabled, this string is empty.

Syntax

*object.***LoginGroup** *GroupName*

Properties

The **LoginGroup** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
GroupName	String. Sets the name of the group.

LoginTimeout Property

Sets or retrieves the number of seconds to be used as the iFIX security Login Timeout value for user accounts that are created as a result of the security synchronization process.

Syntax

*object.***LoginTimeout** [= *LongInteger*]

Properties

The **LoginTimeout** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
LongInteger	Values from 0 to 86399 seconds. The default value is 0, which indicates no login timeout.

Remarks

This property corresponds to the /T command line parameter of the Security Synchronizer application.

LoginUserFullName Property

Returns the full name of the currently logged in iFIX user. If security is disabled, this string is empty.

Syntax

UserFullName = *object.***LoginUserFullName**

Properties

The **LoginUserFullName** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
UserFullName	String. The full login name of the user.

LoginUserName Property

Returns the user ID of the currently logged in iFIX user. If security is disabled, this string is empty.

Syntax

UserName = *object*.**LoginUserName**

Properties

The **LoginUserName** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
UserName	String. The login name of the user.

LoInValue Property

Specifies the lower limit on the input value.

Syntax

object.**LoInValue** [= *Variant*]

Properties

The **LoInValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	The low limit on the input value.

LoLimit Property

Specifies the lower limit for the specified **Pen** **GeneralDataSet**, , or **RealTimeSPCDataSet**.

Syntax

object.**LoLimit** [= *Double*]

Properties

The **LoLimit** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The low limit.

LoOutValue Property

Specifies the lower limit on the output value.

Syntax

*object.***LoOutValue** [= *Variant*]

Properties

The **LoOutValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	The low limit on the output value.

LowestDataValue Property

Specifies the lowest value for the specified **Pen**.

Syntax

*object.***LowestDataValue** [= *Double*]

Properties

The **LowestDataValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The lowest value for the **Pen**.

LWL Property

Sets the lower warning limits (LWL) for the real-time SPC data set.

Syntax

*object.***LWL** [= *Double*]

Properties

The **LWL** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The lower warning limits (LWL) for the RealTimeSPCDataSet Object.

M-N

MainTitle Property

Specifies the main title for your Enhanced Chart.

Syntax

*object.***MainTitle** [= *String*]

Properties

The **MainTitle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	Text that appears as the main title in the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

MainTitleBold Property

Specifies whether the main title for your Enhanced Chart appears in a bold typeface.

Syntax

*object.***MainTitleBold** [= *Boolean*]

Properties

The **MainTitleBold** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the main title in a HistogramChart, LineChart, SPCBarChart, or XYChart Object displays in bold.

Settings

The settings for *Boolean* are:

Value	Description
True	The main title displays in bold. (Default)
False	The main title does not display in bold.

MainTitleFont Property

Specifies the font face of the main title in your Enhanced Chart.

Syntax

*object.***MainTitleFont** [= *String*]

Properties

The **MainTitleFont** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The name of the font. By default, the font face is "Times New Roman."

MainTitleItalic Property

Specifies whether the main title for your Enhanced Chart appears in italics.

Syntax

*object.***MainTitleItalic** [= *Boolean*]

Properties

The **MainTitleItalic** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the main title in a HistogramChart, LineChart, SPCBarChart, or XYChart Object displays in italics.

Settings

The settings for *Boolean* are:

Value	Description
True	The main title displays in italics.
False	The main title does not display in italics. (Default)

MainTitleUnderline Property

Specifies whether the main title for your Enhanced Chart appears underlined.

Syntax

*object.***MainTitleUnderline** [= *Boolean*]

Properties

The **MainTitleUnderline** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the main title in a HistogramChart, LineChart, SPCBarChart, or XYChart Object appears underlined.

Settings

The settings for *Boolean* are:

Value	Description
True	The label appears underlined.
False	The label does not appear underlined. (Default)

ManualMaxX Property

Sets the maximum floating point value set for the X axis in an Enhanced Chart.

Syntax

*object.***ManualMaxX** [= *Double*]

Properties

The **ManualMaxX** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The maximum floating point value set for the X axis in a HistogramChart, LineChart, SPCBarChart, or XYChart.

Remarks

ManualMaxX is a read-only property.

ManualMaxY Property

Sets the maximum floating point value set for the Y axis in an Enhanced Chart.

Syntax

*object.***ManualMaxY** [= *Double*]

Properties

The **ManualMaxY** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The maximum floating point value set for the Y axis in a HistogramChart, LineChart, SPCBarChart, or XYChart.

ManualMinX Property

Sets the minimum floating point value set for the X axis in an Enhanced Chart.

Syntax

object.**ManualMinX** [= *Double*]

Properties

The **ManualMinX** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The minimum floating point value set for the X axis in a HistogramChart, LineChart, SPCBarChart, or XYChart.

Remarks

ManualMinX is a read-only property.

ManualMinY Property

Sets the minimum floating point value set for the Y axis in an Enhanced Chart.

Syntax

object.**ManualMinY** [= *Double*]

Properties

The **ManualMinY** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The minimum floating point value set for the Y axis in a HistogramChart, LineChart, SPCBarChart, or XYChart.

ManualScaleControlX Property

Sets the grid scale used for the X axis. This property only applies to XY Enhanced Charts.

Syntax

*object.***ManualScaleControlX** [= *enumManualScaleControl*]

Properties

The **ManualScaleControlX** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumManualScaleControl	An enumeration that represents the grid scale for the X axis in an XYChart. **Valid entries:** 0 – XYManScaleNone 3 – XYManScaleMinMax

ManualScaleControlY Property

Sets the grid scale used for the Y axis.

Syntax

*object.***ManualScaleControlY** [= *enumManualScaleControl*]

Properties

The **ManualScaleControlY** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumManualScaleControl	An enumeration that represents the grid scale for the Y axis in a HistogramChart, LineChart, SPCBarChart, or XYChart. **Valid entries:** 0 – ManScaleNone 1 – ManScaleMin 2 – ManScaleMax 3 – ManScaleMinMax

MapMode Property

(For future use only.) Sets or retrieves the security mapping mode to be used in synchronizing security.

Syntax

*object.***MapMode** [= *ShortInteger*]

Properties

The **MapMode** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ShortInteger	0 is the only valid value.

Remarks

This property can be ignored.

MarkDataPoints Property

Specifies whether each individual data point is plotted on the Enhanced Chart.

Syntax

*object.***MarkDataPoints** [= *Boolean*]

Properties

The **MarkDataPoints** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	**True** – Indicates that each individual data point is plotted. **False** – Indicates that each individual data point is not plotted. (Default)

MarkerChar Property

Specifies the character to use as markers on the trend lines when the **UseMarker** property for the pen to **True** and the **MarkerStyle** property is set to *CharacterMarker*.

Syntax

*object.***MarkerChar**[= *String*]

Properties

The **MarkerChar** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The character to use for markers.

MarkerStyle Property

Specifies the style to use as markers on the trend lines when the UseMarker property is set to **True**.

Syntax

*object.***MarkerStyle** [= *enumMarkerStyle*]

Properties

The **MarkerStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumMarkerStyle	The marker style.

Settings

The settings for *enumMarkerStyle* are:

Constant	Value	Description
NoMarker	0	Display no markers.
RectangleMarker	1	Display rectangles as markers.
OvalMarker	2	Display ovals as markers.
DiamondMarker	3	Display diamonds as markers.
CharacterMarker	4	Display the character specified in the MarkerChar property as markers.

Master Property

Indicates whether the specified Dynamo is a master Dynamo. This property is read-only.

Syntax

*DynamoObject.***Master**

Properties

The **Master** property syntax has this part:

Part	Description
DynamoObject	A Dynamo object.

Return Value

Boolean. **True** indicates that the Dynamo is a master Dynamo, while **False** indicates that it is not.

Max_Dynamo_Desc_Length Property

The maximum amount of characters that you can enter for the Dynamo object's description. This property is read-only.

Syntax

*DynamoObject.***Dynamo_Max_Dynamo_Desc_Length** *(plMaxDynamoDescLength)*

Properties

The **Dynamo_Max_Dynamo_Desc_Length** property syntax has these parts:

Part	Description
DynamoObject	A Dynamo object.
plMaxDynamoDescLength	Long. The maximum string length of the Dynamo object description. This length does not include a space for the NULL character. So, some languages will require the array size to be plMaxDynamoDescLength + 1.

MaxCharactersPerLine Property

Returns the maximum number of characters allowed per line.

Syntax

*object.***MaxCharactersPerLine**

Properties

The **MaxCharactersPerLine** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

MaxCharactersPerLine is a read-only property of type *Long*.

MaxLines Property

Specifies the maximum number of text lines allowed.

Syntax

*object.***MaxLines**

Properties

The **MaxLines** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

The **MaxLines** property is of type *Long*.

MaxPts Property

Specifies the maximum number of points to be displayed for the trend line for the specified **Pen**.

Syntax

*object.***MaxPts** [= *Long*]

Properties

The **MaxPts** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The maximum number of points. (Default = 500)

MaxXAxisLabels Property

The MaxXAxisLabels property applies only to the Line/Multiline Enhanced Chart. This property allows you to specify the maximum number of labels (for the time and date) to be displayed on the X axis for this type of chart. If the width of the chart is not sufficient to accommodate this number without creating additional rows, fewer will be displayed.

When maximum is set to anything other than 0, the placement of labels starts at the most current time (greatest time) and then is spaced evenly towards the left. In most cases, there will not be a label lined up with the extreme left-side of the plot; the only time this is guaranteed to happen is when the maximum is set to 2. The labels will always display in a single line except for a value of 0, in which case the values may appear in two or more rows.

Syntax

*object.***MaxXAxisLabels** [= *Long*]

Properties

The **MaxXAxisLabels** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number, from 0 – 20, representing the maximum number of labels for the time and date to be displayed on the X axis. The chart logic calculates the number of labels to display on the X axis based on settings such as chart size, duration, and number of points. However, when a maximum number is specified, the user can have partial control over this logic not to display more than the specified number of labels regardless of the chart configuration. If 0 is selected, the MaxXAxisLabels is considered to be “Unspecified” and the labels are dis-

	played as they were before iFIX 5.5, and may appear in two or more rows. A value of “Unspecified” means the label spacing is automatically controlled by the chart. For pictures created before iFIX 5.5, the default value is 0. For all other pictures, the default value is 10.

MonoDeskColor Property

When Monochrome is enabled in an Enhanced Chart, this property specifies the shade of black, gray, or white surrounding the graph and/or table displays.

Syntax

object.**MonoDeskColor** [= *Long*]

Properties

The **MonoDeskColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	An Integer value representing the color.

MonoGraphBackColor Property

When Monochrome is enabled in an Enhanced Chart, this property specifies the shade of black, gray, or white for the graph background color.

Syntax

object.**MonoGraphBackColor** [= *Long*]

Properties

The **MonoGraphBackColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	An Integer value representing the color.

MonoGraphForeColor Property

When Monochrome is enabled in an Enhanced Chart, this property specifies the shade of black, gray, or white for the graph foreground color.

Syntax

*object.***MonoGraphForeColor** [= *Long*]

Properties

The **MonoGraphForeColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	An Integer value representing the color.

MonoShadowColor Property

When Monochrome is enabled in an Enhanced Chart, this property specifies the shade of black, gray, or white for the drop shadow color.

Syntax

*object.***MonoShadowColor** [= *Long*]

Properties

The **MonoShadowColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	An Integer value representing the color.

MonoTableBackColor Property

When Monochrome is enabled in an Enhanced Chart, this property specifies the shade of black, gray, or white for the table background color.

Syntax

*object.***MonoTableBackColor** [= *Long*]

Properties

The **MonoTableBackColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	An Integer value representing the color.

MonoTableForeColor Property

When Monochrome is enabled in an Enhanced Chart, this property specifies the shade of black, gray, or white for the table foreground color.

Syntax

*object.***MonoTableForeColor** [= *Long*]

Properties

The **MonoTableForeColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	An Integer value representing the color.

MonoTextColor Property

When Monochrome is enabled in an Enhanced Chart, this property specifies the shade of black, gray, or white for the text color.

Syntax

*object.***MonoTextColor** [= *Long*]

Properties

The **MonoTextColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	An Integer value representing the color.

MultipleEGU Property

Specifies whether to display the **ValueAxis** for each **Pen** in the **Chart** that contains multiple EGUs.

Syntax

*object.***MultipleEGU** [= *Boolean*]

Properties

The **MultipleEGU** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Settings

The settings for *Boolean* are:

Value	Description
True	The **ValueAxis** is displayed for each **Pen.**
False	The **ValueAxis** is not displayed for each **Pen.**

Remarks

The **MultipleEGU** property corresponds to the Show Multiple Values check box in the Chart Configuration dialog box.

MultipleTimes Property

Specifies whether to display the **Time Axis** for each **Pen** in the **Chart**.

Syntax

*object.***MultipleTimes** [= *Boolean*]

Properties

The **MultipleTimes** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Settings

The settings for *Boolean* are:

Value	Description
True	Display all time axes.
False	Do not display all time axes. (Default)

MyNodeName Property

Returns the iFIX physical node name.

Syntax

*object.***MyNodeName** *MyNodeName*

Properties

The **MyNodeName** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
MyNodeName	String. Sets the node name of the current node.

Name Property

Specifies the name of the current object.

Syntax

*object.***Name** [= *String*]

Properties

The **Name** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The name of the object.

NOTE: The **Name** property for the **Application** object returns "Application" for internal purposes. When attempting to identify the iFIX application, the **FullName** property should be used.

Next Property

Returns the next **Window** object.

Syntax

*object.***Next**

Properties

The **Next** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Next is a read-only property of type *Object*.

NlsPath Property

Returns the path used to store the language files used to create dialog boxes. If you to choose to implement a native language other than English, the new language files replace the files found in this directory.

Syntax

*object.***NlsPath**

Properties

The **NlsPath** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

NlsPath is a read-only property of type *String*.

NoSaveOnClose Property

Specifies whether a user will be prompted to save a picture when closing it. This property is especially useful when you are generating objects in a picture from the VBA scripts on-the-fly, and do not want to be prompted to save the picture when closing.

Using this property allows you to close a picture without saving it, and without a prompt to save it.

Syntax

object.**NoSaveOnClose** [= *Boolean*]

Properties

The **NoSaveOnClose** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	**True** - Indicates that the user *will not* be prompted to save the picture when it is closed. **False** - Indicates that the user *will* be prompted to save the picture when it is closed.

Remarks

The following example shows how to set the NoSaveOnClose property:

```
Private Sub CFixPicture_Initialize()
       Dim objFileLink As Object
       Set objFileLink = Me.Parent
       objFileLink.NoSaveOnClose = True
       Set objFileLink = Nothing
End sub
```

NumberOfCharacters Property

Returns the total number of characters for all items displayed in the Legend.

Syntax

object.**NumberOfCharacters**[= *Integer*]

Properties

The **NumberOfCharacters** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

NumberOfCharacters is a read-only property of type *Integer*.

The number of characters for specific columns in the **Legend** can be set using the **ModifyColumnLength** method.

NumberOfHorizontalGridLines Property

Specifies the number of horizontal grid lines in the specified **Chart**.

Syntax

*object.***NumberOfHorizontalGridLines** [= *Long*]

Properties

The **NumberOfHorizontalGridLines** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The number of horizontal grid lines in the **Chart**.

NumberOfItems Property

Returns the number of columns displayed in the **Legend**.

Syntax

*object.***NumberOfItems**

Properties

The **NumberOfItems** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

NumberOfItems is a read-only property of type *Integer*.

Columns can be added to and/or removed from the **Legend** by calling the **AddLegendItem**, **RemoveItem** and/or the **RemoveLegendItem** methods.

NumberOfLines Property

Returns the number of lines being used by the specified object.

Syntax

*object.***NumberOfLines**

Properties

The **NumberOfLines** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

NumberOfLines is a read-only property of type *Long*.

The value returned by the **NumberOfLines** property will always be less than or equal to the value of the **MaxLines** property.

NumberOfPoints Property

Returns the number of points in the specified object.

Syntax

*object.***NumberOfPoints**

Properties

The **NumberOfPoints** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

NumberOfPoints is a read-only property of type *Long*.

NumberOfTargets Property

Returns the total number of objects which are currently connected to any of the specified object's properties. This object is the source of data and the “NumberOfTargets” allows a user to determine how many targets are connected to receive information from this object.

Syntax

*object.***NumberOfTargets**

Properties

The **NumberOfTargets** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

NumberOfTargets is a read-only property of type *Long*.

NumberOfVerticalGridLines Property

Specifies the number of vertical grid lines displayed in the Enhanced Chart.

Syntax

*object.***NumberOfVerticalGridLines** [= *Long*]

Properties

The **NumberOfVerticalGridLines** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The number of vertical grid lines in the **Chart**.

NumOfPoints Property

The return value depends on the object specified. For the LineChart Object, HistogramChart Object, SPCBarChart Object, or the XYChart Object, this property returns the number of points in the specified object. For the GeneralDataSet Object or RealTimeSPCDataSet Object this property returns the maximum display points.

Syntax

*object.***NumOfPoints** [= *Long*]

Properties

The **NumOfPoints** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The number of points in the specified object or the maximum display points.

NumPointsToGraph Property

A read-only property that matches the NumOfPoints property of the LineChart Object, HistogramChart Object, SPCBarChart Object, or the XYChart Object.

Syntax

object.**NumPointsToGraph** [= *Long*]

Properties

The **NumPointsToGraph** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The number of points in the specified object.

Remarks

NumPointsToGraph is a read-only property.

NumHGridLines Property

Specifies the number of horizontal grid lines displayed for the specified **Pen**.

Syntax

object.**NumHGridLines** [= *Long*]

Properties

The **NumHGridLines** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The number of horizontal grid lines for the **Pen**.

NumLabels Property

Specifies the number of labels on the **Time Axis** or **Value Axis**.

Syntax

object.**NumLabels** [= *Integer*]

Properties

The **NumLabels** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The number of labels.

NumPts Property

Returns the number of data points for the specified **Pen**.

Syntax

*object.***NumPts**

Properties

The **NumPts** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

NumPts is a read-only property of type *Long*.

NumRandomSubsets Property

Sets the number of baseline subsets selected in the Enhanced Chart.

Syntax

*object.***NumRandomSubsets** [= *Long*]

Properties

The **NumRandomSubsets** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The number of baseline subsets selected for the HistogramChart, LineChart, or SPCBarChart.

Remarks

NumRandomSubsets is a read-only property.

NumScrollingSubsets Property

In an Enhanced Chart, specifies the number of data sources to view in one visualization of the chart, in addition to the baseline. If zero is selected, only baseline data sources are graphed in a single visualization and no scroll bar appears. For this property to apply, you must specify the baseline data sources, using the RandomSubsetsToGraph property.

Syntax

object.**NumScrollingSubsets** [= *Long*]

Properties

The **NumScrollingSubsets** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The number of subsets to view in one group for the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

NumTicks Property

Specifies the number of interval markers to place on each **Time Axis** or **Value Axis**.

Syntax

object.**NumTicks** [= *Integer*]

Properties

The **NumTicks** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The number of interval markers.

NumVGridLines Property

Specifies the number of vertical grid lines displayed for the specified **Pen**.

Syntax

object.**NumVGridLines** [= *Long*]

Properties

The **NumVGridLines** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The number of vertical grid lines for the **Pen**.

O-P

Object Property

Reserved for internal purposes.

OpcAccessPath Property

Returns the OPC access path of the installed OPC DataServer.

Syntax

*object.***OpcAccessPath**

Properties

The **OpcAccessPath** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

OpcAccessPath is a read-only property of type *String*.

The **OpcAccessPath** information is registered during installation of the OPC server using the DataServerInstaller program.

OpcDataSource Property

Returns the OPC data source name of the installed OPC DataServer.

Syntax

*object.***OpcDataSource**

Properties

The **OpcDataSource** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

OpcDataSource is a read-only property of type *String*.

The **OpcDataSource** information is registered during installation of the OPC server using the DataServerInstaller program.

OpcProgID Property

Returns the OPC ProgID of the installed OPC **DataServer**.

Syntax

*object.***OpcProgID**

Properties

The **OpcProgID** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

OpcProgID is a read-only property of type *String*.

The **OpcProgID** information is registered during installation of the OPC server using the DataServer-Installer program.

OpcServerMachineName Property

This is the name of the machine where the installed OPC **DataServer** resides.

Syntax

*object.***OpcServerMachineName**

Properties

The **OpcServerMachineName** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

OpcServerMachineName is a read-only property of type *String*.

The **OpcServerMachineName** information is registered during installation of the OPC server using the DataServerInstaller program.

OriginalScreenHeight Property

Returns the screen height resolution (in pixels) when the document was created.

Syntax

*object.***OriginalScreenHeight**

Properties

The **OriginalScreenHeight** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

OriginalScreenHeight is a read-only property of type *Long*.

OriginalScreenHeight will only be changed if the document is saved on a different resolution.

OriginalScreenWidth Property

Returns the screen width resolution (in pixels) when the document was created.

Syntax

*object.***OriginalScreenWidth**

Properties

The **OriginalScreenWidth** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

OriginalScreenWidth is a read-only property of type *Long*.

OriginalScreenWidth will only be changed if the document is saved on a different resolution.

OriginX Property

Returns the x coordinate for the starting position of the specified axis.

Syntax

*object.***OriginX**

Properties

The **OriginX** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

OriginX is a read-only property of type *Double*.

OriginY Property

Returns the y coordinate for the starting position of the specified axis.

Syntax

*object.***OriginX**

Properties

The **OriginX** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

OriginY is a read-only property of type *Double*.

OutputValue Property

Specifies the data which has resulted from the animation object's transformation of the "Input Value" data.

Syntax

*object.***OutputValue** [= *Variant*]

Properties

The **OutputValue** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	The output value.

Owner Property

Returns the object at the next highest level in the object hierarchy.

Syntax

*object.***Owner**

Properties

The **Owner** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Owner is a read-only property of type *Object*.

The **Owner** property is particularly useful when an object belongs to a **Group**. The **Owner** property of the object returns the **Group**.

See the **Parent** property.

Page Property

Returns the contents of the document.

Syntax

*object.***Page**

Properties

The **Page** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Page is a read-only property of type *Object*.

The Workspace is an OLE document container which can contain OLE documents of different types (pictures, schedules, Word Documents, Excel Spreadsheets). The **Document** object does not know about the internals of the different types of OLE documents. The **Page** property exists to allow you access to the object model of the particular OLE document you are working with. Use the**Type** property of the Document object to identify the type of OLE document before accessing the **Page** property for the **Document**.

Parent Property

Returns the **Picture** in which the specified object resides.

Syntax

*object.***Parent**

Properties

The **Parent** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Parent is a read-only property of type *Object*.

If an object is contained within a **Group**, the **Parent** property returns the **Picture**, not the **Group**.

The **Parent** of the **Picture** object is a **Document** object.

See the **Owner** property.

Path Property

Returns the path of the WorkSpace's executable file or the path of the **Document** object.

Syntax

*object.***Path**

Properties

The **Path** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Path is a read-only property of type *String*.

PauseIndicatorBlink Property

Specifies if the **Alarm Summary** object's pause indicator blinks.

Syntax

object.**PauseIndicatorBlink** [= *Boolean*]

Properties

The **PauseIndicatorBlink** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the pause indicator blinks.

Settings

The settings for *Boolean* are:

Value	Description
True	The pause indicator blinks.
False	The pause indicator does not blink.

PauseIndicatorColor Property

Specifies the color of the **Alarm Summary** object's pause indicator.

Syntax

object.**PauseIndicatorColor** [= *colorref*]

Properties

The **PauseIndicatorColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
colorref	The COLORREF used to set the pause indicator color.

PauseWithNewAlarmIndicatorBlink Property

Specifies whether the **Alarm Summary** object's pause indicator blinks when new alarms are received while the spreadsheet is paused.

Syntax

object.**PauseWithNewAlarmIndicatorBlink** [= *Boolean*]

Properties

The **PauseWithNewAlarmIndicatorBlink** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the pause indicator blinks when new alarms are received while the spreadsheet is paused.

Settings

The settings for *Boolean* are:

Value	Description
True	The pause indicator blinks.
False	The pause indicator does not blink.

PauseWithNewAlarmIndicatorColor Property

Specifies the color of the Alarm Summary object's pause indicator when new alarms are received while the spreadsheet is paused.

Syntax

object.**PauseWithNewAlarmIndicatorColor** [= *colorref*]

Properties

The **PauseWithNewAlarmIndicatorColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
colorref	The COLORREF used to set the pause indicator color.

PenDescription Property

Specifies the description displayed in the Legend for the specified Pen.

Syntax

*object.***PenDescription** [= *String*]

Properties

The **PenDescription** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The description for the current **Pen**.

PenLineColor Property

Specifies the color of the line for the specified Pen.

Syntax

*object.***PenLineColor** [= *Long*]

Properties

The **PenLineColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the line color.

PenLineStyle Property

Specifies the style of the line for the specified **Pen**.

Syntax

*object.***PenLineStyle** [= *enumEdgeStyle*]

Properties

The **PenLineStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumEdgeStyle	The style to display for the line.

Settings

The settings for *enumEdgeStyle* are:

Constant	Value	Description
EdgeStyleSolid	0	Solid.
EdgeStyleDash	1	Dash.
EdgeStyleDot	2	Dot.
EdgeStyleDashDot	3	Dash-Dot.
EdgeStyleDashDotDot	4	Dash-Dot-Dot.
EdgeStyleNone	5	No border.
EdgeStyleInsideFrame	6	Inside Frame.

Remarks

Changes to the **PenLineStyle** property are only visible when the **PenLineWidth** property for the pen is set to 1.

PenLineWidth Property

Specifies the width of the line for the specified **Pen**.

Syntax

*object.***PenLineWidth** [= *Long*]

Properties

The **PenLineWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Long	The width of the line.

PenNum Property

Returns the index of the specified pen in the current chart's **Pens** collection.

Syntax

object.**PenNum**

Properties

The **PenNum** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

PenNum is a read-only property of type *Integer*.

Pens Property

Returns the collection (list) of pens currently configured for the specified **Chart**.

Syntax

object.**Pens**

Properties

The **Pens** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Pens is a read-only property of type *Object*.

PenType Property

Returns whether the specified **Pen** is a real time or historical **Pen**.

Syntax

object.**PenType**

Properties

The **PenType** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Values

The **PenType** property return values are:

Constant	Value	Description
Realtime	0	Real time pen.
Historical	1	Dash.

Remarks

Pens is a read-only property of type *enumHTRModes*.

PictureDefaultAlwaysOnTop Property

Specifies the default Picture Always on Top preference.

Syntax

bValue = *object.***PictureDefaultAlwaysOnTop**

Example

```
Dim bValue as Boolean
bValue = Application.UserPreferences.PictureDefaultAlwaysOnTop
```

Properties

The **PictureDefaultAlwaysOnTop** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bValue	Boolean. Determines whether iFIX pictures appear as always on top. Valid Entries: 1 - Always on top 0 - Not always on top

PictureDefaultBackColor Property

Specifies the default background color of iFIX pictures.

Syntax

lPictureDefaultBackColor = *object.***PictureDefaultBackColor**

Example

```
Dim lPictureDefaultBackColoras Long
lPictureDefaultBackColor= Application.UserPreferences.PictureDefaultBackColor
```

Properties

The **PictureDefaultBackColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
lPictureDefaultBackColor	Long. Determines the default background color of iFIX pictures.

PictureDefaultHeight Property

Specifies the default picture height.

Syntax

dblPictureDefaultHeight = *object.***PictureDefaultHeight**

Example

```
Dim dblPictureDefaultHeightas Double
dblPictureDefaultHeight = Application.UserPreferences.PictureDefaultHeight
```

Properties

The **PictureDefaultHeight** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
dblPictureDefaultHeight	Double. A number between 0 and 100 that determines the default picture height.

PictureDefaultResizable Property

Specifies the default Picture resizable preference.

Syntax

bDefaultResizable = *object.***PictureDefaultResizable**

Example

```
Dim bDefaultResizable as Boolean
bDefaultResizable = Application.UserPreferences.PictureDefaultResizable
```

Properties

The **PictureDefaultResizable** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bDefaultResizable	Boolean. Determines whether iFIX pictures are resizable. Valid Entries: 1 - Resizable 0 - Not resizable

PictureDefaultRuntimeVisible Property

Specifies the default runtime visible preference.

Syntax

bPictureDefaultRuntimeVisible = *object.***PictureDefaultRuntimeVisible**

Example

```
Dim bPictureDefaultRuntimeVisibleas Boolean
bPictureDefaultRuntimeVisible= Application.UserPreferences.PictureDefaultRuntimeVisible
```

Properties

The **PictureDefaultRuntimeVisible** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bPictureDefaultRuntimeVisible	Boolean. Determines whether iFIX pictures appear in the Runtime environment. Valid Entries: 1 - Pictures appear in the Runtime environment 0 - Pictures do not appear in the Runtime environment

PictureDefaultSystemMenu Property

Specifies the default Picture system menu preference.

Syntax

bDefaultSystemMenu = *object.***PictureDefaultSystemMenu**

Example

```
Dim bDefaultSystemMenu as Boolean
bDefaultSystemMenu = Application.UserPreferences.PictureDefaultSystemMenu
```

Properties

The **PictureDefaultSystemMenu** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bDefaultSystemMenu	Boolean. Determines whether the system menu appears in iFIX pictures. Valid Entries: 1 - System menu 0 - No system menu

PictureDefaultTitlebar Property

Specifies the default Picture titlebar preference.

Syntax

bDefaultTitlebar = *object*.**PictureDefaultTitlebar**

Example

```
Dim bDefaultTitlebar as Boolean
bDefaultTitlebar = Application.UserPreferences.PictureDefaultTitlebar
```

Properties

The **PictureDefaultTitlebar** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bDefaultTitlebar	Boolean. Determines whether the Picture titlebar appears. Valid Entries: 1 - Titlebar 0 - No titlebar

PictureDefaultWidth Property

Specifies the default picture width.

Syntax

dblPictureDefaultWidth = *object*.**PictureDefaultWidth**

Example

```
Dim dblPictureDefaultWidth as Double
dblPictureDefaultWidth= Application.UserPreferences.PictureDefaultWidth
```

Properties

The **PictureDefaultWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

dblPictureDefaultWidth	Double. A number between 0 and 100 that determines the default picture width.

PictureHeight Property

Specifies the percentage of the vertical screen that the picture extends to.

Syntax

*object.***PictureHeight***[=dblPictureHeight]*

Properties

The **PictureHeight** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
dblPictureHeight	Double. A number between 0 and 100 that determines the picture height.

PictureName Property

Specifies the name or alias of the current picture.

Syntax

*object.***PictureName** [=*String*]

Properties

The **PictureName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The name of the picture.

PicturePath Property

Returns the path used to store iFIX pictures.

Syntax

*object.***PicturePath**

Properties

The **PicturePath** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

PicturePath is a read-only property of type *String*.

PictureWidth Property

Specifies the percentage of the horizontal screen that the picture extends to.

Syntax

*object.***PictureWidth***[=dblPictureWidth]*

Properties

The **PictureWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
dblPictureWidth	Double. A number between 0 and 100 that determines the picture width.

PieType Property

Specifies the constraints of the angle included between the **StartAngle** and **EndAngle** properties.

Syntax

*object.***PieType** [= *Integer*]

Properties

The **PieType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The type of pie to display.

Settings

The settings for *Integer* are:

Value	Description
1	The included angle is <= 180. (The pie is a wedge). (Default)
2	The included angle is > 180. (The pie is a cut out).
3	There is no restriction on the included angle.

PlotOnChartRefresh Property

This property is unavailable. It is reserved for future use.

Syntax

*object.***PlotOnChartRefresh**

Properties

The **PlotOnChartRefresh** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

PlottingMethod Property

In an Enhanced Chart, this property sets the plotting method used to render the object's data.

Syntax

*object.***PlottingMethod***[=enumGraphPlottingMethod]*

Properties

The **PlottingMethod** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumGraphPlottingMethod	An enumeration that represents the data plotting method in the Enhanced Chart: **Valid entries:** 3 – Plot_Area 1 – Plot_Bar 10 – Plot_Histogram 0 – Plot_Line 2 – Plot_Point 17 – Plot_PointPlusLine 13 – Plot_PointsPlusBestFitCurve 14 – Plot_PointsPlusBestFitCurveGraphed 8 – Plot_PointsPlusBestFitLine 9 – Plot_PointsPlusBestFitLineGraphed 16 – Plot_PointsPlusSpline 15 – Plot_Spline

PointType Property

In an Enhanced Chart, sets the predefined point types for the GeneralDataSet Object or RealTimeSPCDataSet object.

Syntax

object.**PointType** *[=enumPointsType]*

Properties

The **PointType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumPointsType	An enumeration that represents the predefined point types in the Enhanced Chart: **Valid entries:** 1 – Point_Cross 6 – Point_Diamond 7 – Point_DiamondSolid 2 – Point_Dot 3 – Point_DotSolid 10 – Point_DownTriangle 11 – Point_DownTriangleSolid 0 – Point_Plus 4 – Point_Square 5 – Point_SquareSolid 8 – Point_UpTriangle 9 – Point_UpTriangleSolid

Previous Property

Returns the previous **Window** object.

Syntax

object.**Previous**

Properties

The **Previous** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Previous is a read-only property of type *Object*.

PrimarySecPath Property

Returns the primary security file path for the specified document.

Syntax

*object.***PrimarySecPath** [= *String*]

Properties

The **PrimarySecPath** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The backup security file path.

Remarks

PrimarySecPath is a read-only property of type *String*.

ProcedureDeclaration Property

Specifies the header line of the ScriptProcedure.

Syntax

*object.***ProcedureDeclaration** [= *String*]

Properties

The **ProcedureDeclaration** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The header line for the procedure.

Remarks

An example of a header line is: Private Sub MyObject_Click().

ProcedureName Property

(Read-Only) The name of the procedure to be executed.

Syntax

*object.***ProcedureName** [= *ProcedureName*]

Properties

The **ProcedureName** property syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
ProcedureName	String. The new name of the procedure to execute.

Procedures Property

Returns the **Procedures** collection object.

Syntax

*object.***Procedures**

Properties

The **Procedures** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Procedures is a read-only property of type *Object*.

ProcedureStatement Property

Specifies the string to be used in the procedure statement for the specified **ScriptLine** object.

Syntax

*object.***ProcedureStatement** [= *String*]

Properties

The **ProcedureStatement** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The procedure statement for the specified **ScriptLine** object.

ProgId Property

Returns the ProgID for the specified control.

Syntax

*object.***ProgId** [= *String*]

Properties

The **ProgId** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ProgId is a read-only property of type *String*.

ProjectPath Property

Returns the path used for storing project files. If you intall iFIX to the default location, this path is C:\Program Files (x86)\Proficy\iFIX

Syntax

object.**ProjectPath**[=*String*]

Properties

The **ProjectPath** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The path of the project.

Property1 Property

A general purpose property which can contain any user value.

Syntax

object.**Property1** [= *Variant*]

Properties

The **Property1** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	A user defined value.

Remarks

Property1 is used internally to store information entered in the script authoring experts.

Property10 Property

A general purpose property which can contain any user value.

Syntax

*object.***Property10** [= *Variant*]

Properties

The **Property10** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	A user defined value.

Remarks

Property10 is used internally to store information entered in the script authoring experts.

Property2 Property

A general purpose property which can contain any user value.

Syntax

*object.***Property2** [= *Variant*]

Properties

The **Property2** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	A user defined value.

Remarks

Property2 is used internally to store information entered in the script authoring experts.

Property3 Property

A general purpose property which can contain any user value.

Syntax

*object.***Property3** [= *Variant*]

Properties

The **Property3** property syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
Variant	A user defined value.

Remarks

Property3 is used internally to store information entered in the script authoring experts.

Property4 Property

A general purpose property which can contain any user value.

Syntax

object.**Property4** [= *Variant*]

Properties

The **Property4** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	A user defined value.

Remarks

Property4 is used internally to store information entered in the script authoring experts.

Property5 Property

A general purpose property which can contain any user value.

Syntax

object.**Property5** [= *Variant*]

Properties

The **Property5** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	A user defined value.

Remarks

Property5 is used internally to store information entered in the script authoring experts.

Property6 Property

A general purpose property which can contain any user value.

Syntax

*object.***Property6** [= *Variant*]

Properties

The **Property6** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	A user defined value.

Remarks

Property6 is used internally to store information entered in the script authoring experts.

Property7 Property

A general purpose property which can contain any user value.

Syntax

*object.***Property7** [= *Variant*]

Properties

The **Property7** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	A user defined value.

Remarks

Property7 is used internally to store information entered in the script authoring experts.

Property8 Property

A general purpose property which can contain any user value.

Syntax

*object.***Property8** [= *Variant*]

Properties

The **Property8** property syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
Variant	A user defined value.

Remarks

Property8 is used internally to store information entered in the script authoring experts.

Property9 Property

A general purpose property which can contain any user value.

Syntax

object.**Property9** [= *Variant*]

Properties

The **Property9** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	A user defined value.

Remarks

Property9 is used internally to store information entered in the script authoring experts.

Q-R

Quality Property

Returns the quality of the OPC data source represented by this DataItem. The **Quality** property becomes populated either from Read method of the **DataItem** or Group (DataSystem) object.

Syntax

object.**Quality**

Properties

The **Quality** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Quality is a read-only property of type *Integer*.

The Value and Timestamp properties are associated with the **Quality**.

QueueEvents Property

Specifies whether to allow multiple entries of the same event in the event queue.

Syntax

*object.***QueueEvents** [= *Boolean*]

Properties

The **QueueEvents** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to allow multiple entries in the queue.

Settings

The settings for *Boolean* are:

Value	Description
True	Allow multiple entries in the queue.
False	Prevent multiple entries in the queue.

Remarks

If a **Timer** event is configured to occur every 5 seconds and it takes longer than 5 seconds to execute the event, the event will not be placed in the queue if **QueueEvents** is set to **False**. If **QueueEvents** is **True**, the event handler will occur for each event fired once the first event is acknowledged.

QuickConfigure Property

When QuickConfigure is set to true for a Line/MultiLine or XY Enhanced Chart, this property allows you to change chart properties while in run mode without having to display the configuration dialog box or the right-click menu. The Quick Configure chart properties that can be modified in run mode include:

- **Y Axes Style –** Hovering over the upper left hand corner of the chart displays Multiple, Stack, or Single buttons that can be used to display the Y Axes of the chart in the specified format.
- **Y Axes Always Visible** – Clicking the far right column in the Data Source Legend area allows you to toggle between Yes and No to set the Y Axis Always Visible property for each data source.
- **Plot Visible** – Clicking the next-to-last column in the Data Source Legend area allows you to toggle between Yes and No to set the Plot Visible property for each data source.

Syntax

*object.***QuickConfigure** [= *Boolean*]

Properties

The **QuickConfigure** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Describes whether you can change chart properties while in run mode without having to display the configuration dialog box or the right-click menu.

Settings

The settings for *Boolean* are:

Value	Description
True	Allows you to change chart properties while in run mode without having to display the configuration dialog box or the right-click menu.
False	Does not allow you to change chart properties while in run mode without having to display the configuration dialog box or the right-click menu. (Default)

QuickStyle Property

Allows you to apply a predefined chart style to an Enhanced Chart.

Syntax

*object.***QuickStyle***[=enumQuickStyle]*

Properties

The **QuickStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumQuickStyle	An enumeration that represents the predefined chart style in the Enhanced Chart: **Valid entries:** 9 – DarkInset 11 – DarkLine 12 – DarkNoBorder 10 – DarkShadow 1 – LightInset 3 – LightLine 4 – LightNoBorder 2 – LightShadow 5 – MediumInset 7 – MediumLine 8 – MediumShadow 6 – MediumNoBorder 0 – NoStyle

Radius Property

Specifies the current radius for the specified **Pie** object.

Syntax

*object.***Radius** [= *Double*]

Properties

The **Radius** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The value of the radius of the **Pie**.

RandomSubsetsToGraph Property

In an Enhanced Chart, this property specifies the subset of data sources to be graphed. Each specified data source is included in the subset. For example, if you have a chart with four data sources and you specify data sources one and two as part of the subset, then both data sources will appear in all visualizations of the chart. On the other hand, each of the other data sources of the chart will be only graphed in one of the chart's visualizations. For this reason, the data sources specified in this property are also referred to as "baseline" collectively. By specifying baseline and non-baseline data sources, you can compare data sources in different visualizations.

The data sources are specified in the form of dataset indices, separated by a comma. For example, a value of "0, 2" specifies that data sources one and three are to be graphed in all visualizations.

Syntax

*object.***RandomSubsetsToGraph** [= *String*]

Properties

The **RandomSubsetsToGraph** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The subsets to be graphed for the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

RawFormat Property

Specifies whether the text is a string formatted for the C programming language.

Syntax

*object.***RawFormat**

Properties

The **RawFormat** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Settings

RawFormat is a read-only property of type boolean.

Value
True
False

RecalculateViewport Property

IMPORTANT: The RecalculateViewport property does not apply to documents that use the Enhanced Coordinate System. It is only available for documents using legacy coordinates.

Specifies whether persisted ratios are used when calculating the size and viewport for a picture when opening it in the same screen resolution in which it was saved.

Syntax

object.**RecalculateViewport** [= *Boolean*]

Properties

The **RecalculateViewport** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the persisted ratios used to determine the picture's viewport are applied upon reopening the picture in the same resolution in which it was saved.

Settings

The settings for *Boolean* are:

Value	Description
True	The ratios used to determine the picture's viewport are recalculated upon opening, regardless of screen resolution.
False (Default)	The persisted ratios used to determine the picture's viewport are applied when reopening the picture in the same resolution in which it was saved.

Remarks

If you set this property to True, test the picture to make sure it does not cause any unexpected behavior before saving it. When using Fit Picture To Window, this property will automatically be set to False.

RefreshRate Property

Specifies the refresh rate of the Chart or ExpressionEditor.

Syntax

*object.***RefreshRate** [= *Single*]

Properties

The **RefreshRate** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Single	The refresh rate.

RemoveNonWindowsUsers Property

Sets or retrieves the boolean value that indicates whether or not iFIX security users who are not configured to use Windows security will be deleted.

Syntax

*object.***RemoveNonWindowsUsers** [= *Boolean*]

Properties

The **RemoveNonWindowsUsers** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	The default value is **False**, which indicates that users who are not using Windows security will not be deleted. **True** indicates that iFIX users not using Windows security will be deleted.

Remarks

This property corresponds to the /R command line parameter of the Security Synchronizer application.

ResetPercentage Property

Specifies the percentage by which the Chart object's display is shifted when the Pen reaches the right side of the display area when scrolling left to right.

Syntax

*object.***ResetPercentage** [= *Single*]

Properties

The **ResetPercentage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Single	The reset percentage.

Resizable Property

Whether or not the specified document is able to be resized.

Syntax

*object.***Resizable** [= *Boolean*]

Properties

The **Resizable** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the document is resizable.

Settings

The settings for *Boolean* are:

Value	Description
True	The document is resizable.
False	The document is not resizable.

ResolveSourceName Property

Returns the resolved tag name if the source is a tag group symbol.

Syntax

*object.***ResolvedSourceName**

Properties

The **ResolvedSourceName** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Read-only property

Revision Property

Returns the revision number for the specified Dynamo object. This property is read-only.

Syntax

*DynamoObject.***Revision**

Properties

The **Revision** property syntax has this part:

Part	Description
DynamoObject	A Dynamo object.

Return Value

Long. This number represents the revision number that iFIX internally assigns to the Dynamo object.

RevisionNumber Property

Returns the revision number for the specified document.

Syntax

*object.***RevisionNumber**

Properties

The **RevisionNumber** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

RevisionNumber is a read-only property of type *Long*.

RevisionNumber property is incremented with each save of the document.

Right Property

Returns the value of the right edge of the shape's bounding rectangle.

Syntax

*object.***Right**

Properties

The **Right** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Right is a read-only property of type *Double*.

RightCenter Property

Returns the right center point of the shape's bounding rectangle.

Syntax

*object.***RightCenter**

Properties

The **RightCenter** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

RightCenter is a read-only property of type *Object*.

RotationAngle Property

Specifies the angle (in radians or degrees) by which to rotate the specified object.

Syntax

*object.***RotationAngle** [= *Double*]

Properties

The **RotationAngle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The angle to rotate the shape.

Remarks

In the Configuration environment, **RotationAngle** will not be set back to 0 until the object is de-selected. Therefore, while selected, the object will contain it's current **RotationAngle** value of the object when it was last selected. Once de-selected, the object's **RotationAngle** property will be reset back to 0.

In the Run-time environment, the **RotationAngle** property displayes the object's angle based on the number of degrees the object had been rotated when it initially came off disk.

The units to be used when creating the angle is specified as either degrees or radians depending on the value of the **AngleUnits** property.

RoundnessX Property

Specifies the horizontal roundness of a **RoundRectangle**.

Syntax

*object.***RoundnessX** [= *Long*]

Properties

The **RoundnessX** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	A value from 0 to 100.

RoundnessY Property

Specifies the vertical roundness of a **Round Rectangle**.

Syntax

*object.***RoundnessY** [= *Long*]

Properties

The **RoundnessY** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	A value from 0 to 100.

RunIndicatorBlink Property

Specifies whether the **Alarm Summary** object's run indicator blinks.

Syntax

object.**RunIndicatorBlink** [= *Boolean*]

Properties

The **RunIndicatorBlink** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the run indicator blinks.

Settings

The settings for *Boolean* are:

Value	Description
True	The run indicator blinks.
False	The run indicator does not blink.

RunIndicatorColor Property

Specifies the color of the **Alarm Summary** object's run indicator.

Syntax

object.**RunIndicatorBlink** [= *Long*]

Properties

The **RunIndicatorBlink** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the run indicator color.

RuntimeVisible Property

Whether or not the document is visible in the Run-time environment.

Syntax

*object.***RuntimeVisible** [= *Boolean*]

Properties

The **RuntimeVisible** property syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the document is visible.

Settings

The settings for *Boolean* are:

Value	Description
True	The document is visible in the Run-time environment. (Default)
False	The document is not visible in the Run-time environment.

Remarks

To display a picture that was opened with the RuntimeVisible property set to False, you must set the active window's Active property to True.

The following is an example of code that will set the RuntimeVisible and Active properties:

```
Dim doc As Object
Set doc = Application.Documents.Open("C:\Program Files (x86)\Proficy\iFIX\PIC\SAMPLE.GRF")
doc.ActiveWindow.Active = True
```

S

Saved Property

Returns whether the specified document has been edited since the last time it was saved.

Syntax

object.**Saved**

Properties

The **Saved** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Values

The **Saved** property return values are:

Value	Description
True	The document has not been edited since it was last saved.
False	The document has been edited since it was last saved.

SaveThumbnail Property

Allows you to save a thumbnail image of an existing picture, so that you can view the image from the system tree in the iFIX WorkSpace.

Syntax

*object.***SaveThumbnail** [= *Boolean*]

Properties

The **SaveThumbnail** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether thumbnails are saved.

Settings

The settings for *Boolean* are:

Value	Description
True	Thumbnail image is saved with the picture.
False	Thumbnail image is not saved with the picture. (Default)

ScalesWidth Property

Specifies whether the user is allowed to change the width of the **Text** or **Datalink** object.

Syntax

*object.***ScalesWidth** [= *Boolean*]

Properties

The **ScalesWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the text width can be changed.

Settings

The settings for *Boolean* are:

Value	Description
True	The object's width can be changed.
False	The object's width cannot be changed. (There can be clipping of the text)

Remarks

The behavior of the object by changing the **ScalesWidth** property is dependent upon the **Autosize** property. The following table illustrates the dependencies:

Autosize	Value	ScalesWidth Value Resulting Behavior
True	True	Bounding rectangle is recalculated to fit the text.
False	True	The FontSize is recalculated to fit the bounding rectangle.
True	False	Bounding rectangle is recalculated to fit the text.
False	False	The text is clipped.

SchedulePath Property

Returns the path used to store iFIX schedule files.

Syntax

*object.***SchedulePath**

Properties

The **SchedulePath** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

SchedulePath is a read-only property of type *String*.

ScreenHeight Property

Returns the screen resolution height in pixel units.

Syntax

*object.***ScreenHeight**

Properties

The **ScreenHeight** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ScreenHeight is a read-only property of type *Long*.

ScreenWidth Property

Returns the screen resolution width in pixel units.

Syntax

*object.***ScreenWidth**

Properties

The **ScreenWidth** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ScreenWidth is a read-only property of type *Long*.

ScrollDirection Property

Specifies the direction that the **Chart** will scroll.

Syntax

*object.***ScrollDirection** [= *enumScrollDirection*]

Properties

The **ScrollDirection** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrEventName	String. The name of the event.

Settings

The settings for *enumScrollDirection* are:

Constant	Value	Description
RigthToLeft	0	Scroll from right to left.
LeftToRight	1	Scroll from left to right.

ScrollGrid Property

Specifies whether the **Chart** object's grid can be scrolled.

Syntax

*object.***ScrollGrid** [= *Boolean*]

Properties

The **ScrollGrid** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the grid can be scrolled.

Settings

The settings for *Boolean* are:

Value	Description
True	The grid can be scrolled.
False	The grid cannot be scrolled.

ScrollItems Property

Specifies whether to scroll the tick marks and their labels with the data displayed for the **Time Axis**.

Syntax

object.**ScrollItems** [= *Boolean*]

Properties

The **ScrollItems** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to scroll the tick marks and labels.

Settings

The settings for *Boolean* are:

Value	Description
True	Scroll the tick marks and labels with the data.
False	Do not scroll the tick marks and labels with the data.

ScrollPercentage Property

Specifies the percentage to scroll the **Chart**.

Syntax

object.**ScrollPercentage** [= *Single*]

Properties

The **ScrollPercentage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Single	The amount to scroll the **Chart**.

Remarks

This value is used to scroll the **Chart** by using the **ScrollBack** and **ScrollForward** methods.

SecondaryImageDisplayed Property

Specifies whether a **Bitmap** object's secondary image at the current index is being displayed as the primary (or mouse-up) image.

Syntax

*object.***SecondaryImageDisplayed** [= *Boolean*]

Properties

The **SecondaryImageDisplayed** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Settings

The settings for *Boolean* are:

Value	Description
True	Display the secondary image.
False	Do not display the secondary image.

Remarks

If **SecondaryImageDisplayed** is **True**, the primary image is displayed when the mouse is down. In all other cases the primary image is displayed. If a secondary image is not loaded, the primary image is displayed when the mouse is down.

SecurityArea Property

Specifies the Security Area for the specified document.

Syntax

*object.***SecurityArea** [= *String*]

Properties

The **SecurityArea** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The security area.

SelectedDatasource Property

Returns the data source of the currently selected object.

Syntax

object.**SelectedDatasource**[=*String*]

Properties

The **SelectedDatasource** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The data source.

SelectedFieldName Property

Returns the field name of the currently selected object.

Syntax

object.**SelectedFieldName**[=*String*]

Properties

The **SelectedFieldName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The field name.

SelectedNodeName Property

Returns the SCADA node from the currently selected object.

Syntax

object.**SelectedNodeName**[=*String*]

Properties

The **SelectedNodeName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The node name.

SelectedShapes Property

Returns a collection of shapes currently selected within the specified **Picture**, **DynamoSet** or **User-Globals** document.

Syntax

*object.***SelectedShapes**

Properties

The **SelectedShapes** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

SelectedShapes is a read-only property of type *Object*.

SelectedTagName Property

Returns the tag name or OPC element (item) of the currently selected object.

Syntax

object.**SelectedTagName**[=*String*]

Properties

The **SelectedTagName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The tag name or OPC item.

SelectionTimeout Property

Specifies the number of seconds an alarm stays selected once it is selected. The alarm becomes deselected after the timeout period.

Syntax

*object.***SelectionTimeout** [= *Integer*]

Properties

The **SelectionTimeout** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The number of seconds that an alarm stay selected once it is selected. The valid range is 3 - 120 seconds. The default is 10.

SendAlarmMessages Property

Sets or retrieves the boolean value that indicates whether or not the Audit Trail messages will be sent to iFIX alarm destinations.

Syntax

*object.***SendAlarmMessages** [= *Boolean*]

Properties

The **SendAlarmMessages** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	The default value is **False**, which indicates that alarm messages will not be sent to alarm destinations. **True** indicates that alarm messages will be sent to alarm destinations.

Remarks

This property corresponds to the /A command line parameter of the Security Synchronizer application.

ShadowColor Property

In an Enhanced Chart, specifies of the color of the drop shadow, if present.

Syntax

*object.***ShadowColor** [= *Long*]

Properties

The **ShadowColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whole number representing the color of the drop shadow in the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

SharedTableName Property

Specifies the name of the shared **Lookup** table.

Syntax

object.**SharedTableName** [= *String*]

Properties

The **SharedTableName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The name of the shared table.

Remarks

For ease of use the user is able to create shared threshold tables in the global pages for its **Lookup** objects to use. The **Lookup** object will then redirect all its lookups to this table.

The syntax would be something like:

"User.SharedThresholdTable1".

If the user does specify a shared table, all of its own levels are destroyed. The shared table allows for reuse of similar tables (color by value, for example) – making all objects consistent and allowing for any necessary changes to be made in one place.

ShowAxis Property

Specifies whether or not to display the specified **Time Axis** or **Value Axis**.

Syntax

object.**ShowAxis** [= *Boolean*]

Properties

The **ShowAxis** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display the axis.

Settings

The settings for *Boolean* are:

Value	Description
True	The axis is displayed.
False	The axis is not displayed.

ShowDatabaseTab Property

Specifies whether to display the database tab in the Expression Builder dialog.

Syntax

object.**ShowDatabaseTab** [= *Boolean*]

Properties

The **ShowDatabaseTab** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display the database tab.

Settings

The settings for *Boolean* are:

Value	Description
True	The database tab is displayed. (Default)
False	The database tab is not displayed.

ShowDataServersTab Property

Specifies whether to display the DataServers tab in the Expression Builder dialog.

Syntax

object.**ShowDataServersTab** [= *Boolean*]

Properties

The **ShowDataServersTab** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display the data servers tab.

Settings

The settings for *Boolean* are:

Value	Description
True	The data servers tab is displayed. (Default)
False	The data servers tab is not displayed.

ShowDate Property

Specifies whether or not to display the date on the **Time Axis** of the **Chart**.

Syntax

object.**ShowDate** [= *Boolean*]

Properties

The **ShowDate** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display the date.

Settings

The settings for *Boolean* are:

Value	Description
True	The date is displayed.
False	The date is not displayed.

ShowDSLegend Property

Sets whether the data source legends are visible or invisible in a HistogramChart, LineChart, SPCBarChart, or XYChart Object.

Syntax

object.**ShowDSLegend** [= *Boolean*]

Properties

The **ShowDSLegend** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the data source legends are visible or invisible in a HistogramChart, LineChart,

	SPCBarChart, or XYChart Object.

Settings

The settings for *Boolean* are:

Value	Description
True	The data source legends are visible. (Default)
False	The data source legends are invisible.

ShowGaps Property

Specifies whether the **Pen** ,**GeneralDataSet**, or **RealTimeSPCDataSet** should show gaps in the line indicating errors in the data sets.

Syntax

*object.***ShowGaps** [= *Boolean*]

Properties

The **ShowGaps** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display gaps in the line.

Settings

The settings for *Boolean* are:

Value	Description
True	Data error gaps are displayed.
False	Data error gaps are not displayed.

Remarks

Every data point has a value, timestamp, and a quality. If the quality code indicates a bad data sample, for example, device off line or communication access error, this code indicates that the sampling process could not get a valid reading at this point in time. With this field enabled, the **Pen's** data line will show a gap in the data line that indicates that there is no valid data.

ShowGlobalsTab Property

Specifies whether to display the Globals tab in the Expression Builder dialog.

Syntax

*object.***ShowGlobalsTab** [= *Boolean*]

Properties

The **ShowGlobalsTab** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display the globals tab.

Settings

The settings for *Boolean* are:

Value	Description
True	The globals tab is displayed. (Default)
False	The globals tab is not displayed.

ShowGridLines Property

Specifies whether gridlines are displayed for the **Alarm Summary** object.

Syntax

*object.***ShowGridLines** [= *Boolean*]

Properties

The **ShowGridLines** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether grid lines are displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	Grid lines are displayed. (Default)
False	Grid lines are not displayed.

ShowHeaders Property

Specifies whether column headers are displayed for the **Alarm Summary** object.

Syntax

*object.***ShowHeaders** [= *Boolean*]

Properties

The **ShowHeaders** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether column headers are displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	Column headers are displayed. (Default)
False	Column headers are not displayed.

ShowHistoricalTab Property

Specifies whether to display the Historical tab in the Expression Builder dialog.

Syntax

*object.***ShowHistoricalTab** [= *Boolean*]

Properties

The **ShowHistoricalTab** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display the historical tab.

Settings

The settings for *Boolean* are:

Value	Description
True	The historical tab is displayed. (Default)
False	The historical tab is not displayed.

ShowHorizontalGrid Property

Specifies whether to display the **Chart** object's horizontal grid lines.

Syntax

object.**ShowHorizontalGrid** [= *Boolean*]

Properties

The **ShowHorizontalGrid** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display the horizontal grid lines.

Settings

The settings for *Boolean* are:

Value	Description
True	The horizontal grid lines are displayed.
False	The horizontal grid lines are not displayed.

ShowLegend Property

Specifies whether the Chart,HistogramChart,LineChart, orSPCBarChart object's Legend is displayed.

Syntax

object.**ShowLegend** [= *Boolean*]

Properties

The **ShowLegend** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the legend is displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The legend is displayed.
False	The legend is not displayed.

ShowLine Property

Specifies whether the Pen should display the line representing its data points.

Syntax

*object.***ShowLine** [= *Boolean*]

Properties

The **ShowLine** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display the line.

Settings

The settings for *Boolean* are:

Value	Description
True	The line is displayed.
False	The line is not displayed.

ShowPicturesTab Property

Specifies whether to display the Pictures tab in the Expression Builder dialog.

Syntax

*object.***ShowPicturesTab** [= *Boolean*]

Properties

The **ShowPicturesTab** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display the pictures tab.

Settings

The settings for *Boolean* are:

Value	Description
True	The pictures tab is displayed. (Default)
False	The pictures tab is not displayed.

ShowRowNumbers Property

Specifies whether to display row numbers for the **Alarm Summary** object.

Syntax

*object.***ShowRowNumbers** [= *Boolean*]

Properties

The **ShowRowNumbers** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the row numbers are displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The row numbers are displayed.
False	The row numbers are not displayed. (Default)

ShowScrollBars Property

Specifies whether the **Alarm Summary** object displays both horizontal and vertical scroll bars.

Syntax

*object.***ShowScrollBars** [= *Boolean*]

Properties

The **ShowScrollBars** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether scroll bars are displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The horizontal and vertical scroll bars are displayed. (Default)
False	No scroll bars are displayed.

ShowStatusBar Property

Specifies whether the status bar is displayed for the **Alarm Summary** object.

Syntax

*object.***ShowStatusBar** [= *Boolean*]

Properties

The **ShowStatusBar** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the status bar is displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The status bar is displayed. (Default)
False	The status bar is not displayed.

ShowTimeAxis Property

Specifies whether the **Time Axis** is displayed for the specified **Chart**.

Syntax

*object.***ShowTimeAxis** [= *Boolean*]

Properties

The **ShowTimeAxis** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the **Time Axis** is displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The **Time Axis** is displayed. (Default)
False	The **Time Axis** is not displayed.

ShowTimeAxisTitle Property

Specifies whether the title of the **Time Axis** is displayed for the specified **Chart**.

Syntax

*object.***ShowTimeAxisTitle** [= *Boolean*]

Properties

ShowTimeAxisTitle

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the **Time Axis** title is displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The **Time Axis** title is displayed. (Default)
False	The **Time Axis** title is not displayed.

ShowTimeCursor Property

Specifies whether or not to show the time cursor for the **Chart** ,**HistogramChart**, **LineChart**, or **SPCBarChart**.

Syntax

*object.***ShowTimeCursor** [= *Boolean*]

Properties

The **ShowTimeCursor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the time cursor is displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The time cursor is displayed. (Default)
False	The time cursor is not displayed.

ShowTimeCursorToolTips Property

Displays or hides the time cursor's data tool tips.

Syntax

*object.***ShowTimeCursorToolTips** [= *Boolean*]

Properties

The **ShowTimeCursorToolTips** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the time cursor's tool tips are displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The time cursor's tooltips are displayed.
False	The time cursor's tooltips are not displayed.

ShowTimeStamp Property

Specifies whether the **HistDatalink** object displays a timestamp.

Syntax

*object.***ShowTimeStamp** [= *Boolean*]

Properties

The **ShowTimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the timestamp is displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The timestamp is displayed. (Default)
False	The timestamp is not displayed.

ShowTitle Property

Sets whether or not to show the title of the specified **Time Axis** or **Value Axis**.

Syntax

object.**ShowTitle** [= *Boolean*]

Properties

The **ShowTitle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the axis' title is displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The axis' title is displayed.
False	The axis' title is not displayed.

ShowValueAxis Property

Specifies whether the **Value Axis** is displayed for the specified **Chart**.

Syntax

object.**ShowValueAxis** [= *Boolean*]

Properties

The **ShowValueAxis** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the **Value Axis** is displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The **Value Axis** is displayed.
False	The **Value Axis** is not displayed.

ShowValueAxisTitle Property

Specifies whether the title of the **Value Axis** is displayed for the specified **Chart**.

Syntax

object.**ShowValueAxisTitle** [= *Boolean*]

Properties

The **ShowValueAxisTitle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the **Value Axis** title is displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The **Value Axis** title is displayed.
False	The **Value Axis** title is not displayed.

ShowVerticalGrid Property

Specifies whether to display the **Chart** object's vertical grid lines.

Syntax

object.**ShowVerticalGrid** [= *Boolean*]

Properties

The **ShowVerticalGrid** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the vertical grid lines are displayed.

Settings

The settings for *Boolean* are:

Value	Description
True	The vertical grid is displayed.
False	The vertical grid is not displayed.

ShowXAxis Property

Specifies the combination of Grid, Labels, and Title to display on the X axis. Applies to all Enhanced Charts.

Syntax

object.**ShowXAxis** [=*enumShowAxis*]

Properties

The **ShowXAxis** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumShowAxis	An enumeration that represents whether the labels and/or a title displays for the X axis in the Enhanced Chart. **Valid entries:** Grid_Labels_Title (0) – Displays labels, grid, and a title on the X axis. (Default) Grid_Title (1) – Displays the X axis title and grid. Grid_Labels (2) – Displays the grid and labels for the X axis. Grid_only (3) – Displays only the X axis grid. Title_only (4) – Displays only the X axis title.

ShowYAxis Property

Specifies the combination of Grid, Labels, and Title to display on the Y axis. Applies to all Enhanced Charts.

Syntax

object.**ShowYAxis** [= *enumShowAxis*]

Properties

The **ShowYAxis** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumShowAxis	An enumeration that represents whether the labels and/or a title displays for the Y axis in the Enhanced Chart. **Valid entries:** Grid_Labels_Title (0) – Displays labels, grid, and a title on the Y axis. (Default) Grid_Title (1) – Displays the Y axis title and grid. Grid_Labels (2) – Displays the grid and labels for the Y axis. Grid_only (3) – Displays only the Y axis grid. Title_only (4) – Displays only the Y axis title.

SmoothingMode Property

Determines how the pipe will be drawn, either using high quality or high speed. Pipes drawn with the high quality setting use antialiasing; their edges do not appear jagged. Pipes that use high speed do not use antialiasing and their edges are jagged. However, their demand on system resources is not as great as that of pipes that use high quality. If you have a large number of animated pipes, you may want to consider using pipes with the quality high speed, to limit their use of your system resources.

Syntax

*object.***SmoothingMode***[= enumSmoothingMode]*

Properties

The **SmoothingMode** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumSmoothingMode	An enumeration that represents whether a pipe is drawn using high speed or high quality. **Valid entries:** 0 – High Quality. 1 – High Speed.

SmoothShapeOption Property

Describes how **Line** object is rendered in a picture where the SmoothShapes property is set to True. The SmoothShapeOption property is not effective when the pictures's SmoothShape property is set to False.

Syntax

*object.***SmoothShapeOption** [= *enumSmoothShapeOptions*]

Properties

The **SmoothShapeOption** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumSmoothShapeOptions	An enumeration that describes how the object is rendered. Valid values include: • **SmoothLineWithFlatEnds (0)** – This option is typically used to draw smooth line with flat ends. Prior to iFIX 2022, lines were rendered with this option. This is the default (and compatible option) for pictures created prior to iFIX 2022. • **SmoothLineWithEndCaps (1)** – This option renders smooth line with end caps. This option can be used resolve issue related to end caps in a Line object that was created prior to iFIX 2022. This is the default for pictures created in iFIX 2022 and greater.

	• **SharpLineWithFlatEnds (2)** – This option renders a sharp line with flat ends. This option can be used resolve the issue of blur rendering caps in a Line object that created prior to iFIX 2022. • **SharpLineWithEndCaps (3)** – This option renders sharp lines with end caps. This option can be used to render a Line object if a picture's SmoothShapes property is set to False. (Similar to rendering prior to iFIX 5.9.)

SmoothShapes Property

Specifies whether to use smooth borders on shapes, applying anti-aliasing effect to shapes within a picture. This property is only processed in configure mode, not in run mode.

NOTE: Be aware that if you enable this property on pictures with lots of graphics, it may have an impact on the performance of the iFIX WorkSpace.

Syntax

*object.***SmoothShapes** [= *Boolean*]

Properties

The **SmoothShapes** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the object applies anti-aliasing.

Settings

The settings for *Boolean* are:

Value	Description
True	Applies the SmoothShapes property to shapes (the default on new pictures).
False	Does not apply the SmoothShapes property to shapes in pictures (the default on pictures prior to iFIX 5.9).

SnapToGrid Property

Specifies whether to snap objects to the grid points.

Syntax

*object.***SnapToGrid** [= *Boolean*]

Properties

The **SnapToGrid** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to snap objects to grid points.

Settings

The settings for *Boolean* are:

Value	Description
True	Snap the objects to the grid points.
False	Do not snap the objects to the grid points.

SortColumnName Property

Specifies the column heading of the column currently being used for sorting in the Alarm Summary object.

Syntax

object.**SortColumnName** [= *String*]

Properties

The **SortColumnName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The name of the column to be used for sorting. Valid entries are: Time In Block Type Tagname Priority Node Ack/Time Ack/Priority Shelvable

SortOrderAscending Property

Specifies whether the sort order for the Alarm Summary object is ascending or descending.

Syntax

object.**SortOrderAscending** [= *Boolean*]

Properties

The **SortOrderAscending** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the sort is ascending or descending.

Settings

The settings for *Boolean* are:

Value	Description
True	The sort order is ascending.
False	The sort order is descending. (Default)

Source Property

Specifies the source string for an animation. This field specifies where the data should be retrieved from to drive this animation, that is, the input data source. The syntax can be either a fully qualified string, or, when using intelligent defaults for the server, node, and field names, the valid syntax for the FIX data server could be as simple as “AI1”.

A fully qualified string is the server name and OPC string separated by a period. For example, FIX32.Scada1.AI1.F_CV.

Syntax

*object.***Source** [= *String*]

Properties

The **Source** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A valid data source string.

Remarks

This property internally builds a connection (**Connect**) between the input value property of the animation and the data source specified by this property.

Sources Property

Returns the **Sources** collection for the specified **ScriptProcedure** object.

Syntax

*object.***Sources**

Properties

The **Sources** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Sources is a read-only property of type *Object*.

SourceValidated Property

Specifies whether the source property has a valid data source connection.

Syntax

*object.***SourceValidated** [= *Boolean*]

Properties

The **SourceValidated** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the source is valid.

Settings

The settings for *Boolean* are:

Value	Description
True	The source property is valid.
False	The source property is not valid.

SPCChartType Property

Specifies the type of SPC bar chart: R-Bar, X-Bar, or S-Bar.

Syntax

*object.***SPCChartType** [= *enumSPCChartType*]

Properties

The **SPCChartType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumSPCChartType	An enumeration that represents the type of SPCBarChart: **Valid entries:** 1 – XBar 2 – RBar 3 – SBar

SPCInterval Property

Sets the SPC interval for the RealTimeSPCDataSet object.

Syntax

*object.***SPCInterval** [= *Double*]

Properties

The **SPCInterval** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The SPC interval.

SPCType Property

A read-only property that determines how control limits should be set, depending on the SPC chart type. For example, if you add a RealTimeSPCDataSet to an X-Bar Chart, the SPCType data set is also specified as X-bar.

Syntax

*object.***SPCType** [= *enumSPCType*]

Properties

The **SPCType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumSPCType	An enumeration that represents the upper control limits in the Enhanced Chart: **Valid entries:** 1 – StatBlock_XBar 2 – StatBlock_RBar 3 – StatBlock_SBar 4 – HS_Block

Remarks

SPCType is a read-only property.

StartAngle Property

Specifies which portion of the object is visible.

Syntax

*object.***StartAngle** [= *Double*]

Properties

The **StartAngle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The number of units to display the radial segment specifying the start of the angle.

Remarks

The **Pie** is merely a visible piece of an oval. The **StartAngle** and **EndAngle** properties specify which portions of that oval will be visible. These properties define radial segments from the center of the oval between which the **Pie** is formed.

- A **StartAngle** of 0 units will define a horizontal radial from the center of the oval to the right.
- A **StartAngle** of 90 units will define a vertical radial from the center of the oval to the top of the screen.
- A **StartAngle** of 180 units will define a horizontal radial from the center of the oval to the left side of the screen.
- A **StartAngle** of 270 units will define a vertical radial from the center of the oval to the bottom of the screen.

The units to be used when creating the angle is specified as either degrees or radians depending on the value of the **AngleUnits** property.

Changing the **StartAngle** property will change the **StartPoint** property.

StartCap Property

Specifies the start cap to apply to the selected pipe object.

Syntax

*object.***StartCap** [= *enumStartCap*]

Properties

The **StartCap** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Settings

The settings for **StartCap** are:

Value	Description
0	StartCapRound
1	StartCapSquare
2	StartCapHorizontalDiagonal
3	StartCapVerticalDiagonal

StartDateMode Property

Specifies the **GeneralDataSet**, **Lookup**,**Line**, **Formatted Object**, or **Chart** object's initial starting date operational mode as either fixed or relative.

Syntax

*object.***StartDateMode** [= *enumStartMode*]

Properties

The **StartDateMode** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
EnumStartMode	The starting date operational mode.

Settings

The settings for *enumStartMode* are:

Constant	Value	Description
Relative	0	Relative - Uses the **DaysBeforeNow** setting.
Fixed	1	Fixed - Uses the **FixedDate** setting.

Remarks

StartDateMode is a one-shot property.

This property is not impacted by any Global Time Control property settings.

StartDateType Property

Specifies the Pen object's starting date operational mode as either fixed or relative .

Syntax

*object.***StartDateType** [= *enumStartMode*]

Properties

The **StartDateType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumStartMode	The pen's starting date operational mode.

Settings

The settings for *enumStartMode* are:

Constant	Value	Description
Relative	0	Relative.
Fixed	1	Fixed.

Remarks

Use this field to configure either a fixed starting date or a relative date. Configuring a relative start date means that the pen will use the DaysBeforeNow property for determining the start date.

StartPoint Property

Specifies the starting point of the object.

Syntax

*object.***StartPoint** [= *Object*]

Properties

The **StartPoint** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Object	A point of type FixFloatPoint.

Remarks

The **StartPoint** property specifies a point object which contains an x and y double value which are logical coordinate values equivalent to the StartX and StartY properties.

The **StartPoint** for the Pie also defines the end points of the line segments which define the StartAngle and EndAngle of the object.

The **StartPoint** for the each object is the point located at index 0.

StartTime Property

Specifies the GeneralDataSet, Chart, Lookup Object,Line, Formatted Object, and/or Timer object's start time.

Chart Syntax

*object.*StartTime [= *Date*]

Properties

The **StartTime** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Date	The start time for the **Chart** or **GeneralDataSet**.

Timer Syntax

*object.*StartTime [= *Variant*]

The **StartTime** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	The start time for the **Timer**.

Remarks

This property is not impacted by any Global Time Control property settings.

StartTimeMode Property

Specifies the GeneralDataSet, Lookup Object,Line, Formatted Object, or Chart object's initial starting time operational mode to either fixed or relative time-based.

Syntax

*object.*StartTimeMode [= *enumStartMode*]

Properties

The **StartTimeMode** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumStartMode	The starting time operational mode for the **Chart or GeneralDataSet**.

Settings

The settings for *enumStartMode* are:

Constant	Value	Description
Relative	0	Relative - Uses **TimeBeforeNow** setting.
Fixed	1	Fixed - Uses the **FixedTime** setting.

Remarks

StartTimeMode is a one-shot property.

StartTimeType Property

Specifies the **Pen** object's starting time operational mode to either fixed or relative time-based.

Syntax

*object.***StartTimeType** [= *enumStartMode*]

Properties

The **StartTimeType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumStartMode	The starting time operational mode for the Pen.

Settings

The settings for *enumStartMode* are:

Constant	Value	Description
Relative	0	Relative.
Fixed	1	Fixed.

Remarks

Use this field to configure either a fixed start time or a relative start time. Configuring a relative start time mean that the pen will use the **TimeBeforeNow** property for determining the start time.

StartX Property

Specifies the X coordinate of the object's start point.

Syntax

*object.***StartX** [= *Double*]

Properties

The **StartX** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The coordinate value of the X coordinate of the start point.

StartY Property

Specifies the Y coordinate of the object's start point.

Syntax

*object.***StartY** [= *Double*]

Properties

The **StartY** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The coordinate value of the Y coordinate of the start point.

Status Property

Returns whether the **Timer** and/or **Event** is running or stopped.

Syntax

*object.***Status***[=Boolean]*

Properties

The **Status** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the **Timer** and/or **Event** is running or stopped.

Return Values

The **Status** property return values are:

Value	Description
True	The **Timer** or **Event** is running.
False	The **Timer** or **Event** is stopped.

StatusBar Property

Specifies the text displayed in the status bar of the WorkSpace.

Syntax

*object.***StatusBar** [= *String*]

Properties

The **StatusBar** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The text displayed in the status bar.

StatusFontSize Property

Specifies the size of the font displayed for the **Alarm Summary** object.

Syntax

*object.***StatusFontSize** [= *Integer*]

Properties

The **StatusFontSize** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The font size.

SteppedTrend Property

Specifies whether the data points are written with each **Chart** refresh.

Syntax

*object.***SteppedTrend** [= *Boolean*]

Properties

The **SteppedTrend** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Boolean	Whether data points are written with each refresh.

Settings

The settings for *Boolean* are:

Value	Description
True	Data points are written with each refresh of the **Chart**, displaying a flat line.
False	Only the actual datapoints are connected.

StorageMode Property

Describes how data is stored in a **Bitmap** object. This property is applied to the Bitmap object when picture it is saved (or switches between configuration and run mode).

IMPORTANT: JPEG storage mode uses the lossy image compression algorithm that results in a little degradation of the image data.
After JPEG storage mode is applied to a Bitmap control and picture is saved and closed, the original bitmap data cannot be reverted back.

Syntax

object.**StorageMode**[= *enumBitmapStorageMode*]

Properties

The **StorageMode** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumBitmapStorageMode	An enumeration that describes how the bitmap object is rendered. Valid values include: • **PNG (0)** – This mode is typically used to store a Bitmap object as raw data that gets compressed with the lossless algorithm (PNG). It can persist the quality of the image. When the Transparency option is about to be used for masking a specific color in a Bitmap object, this option must be selected as the StorageMode. The PNG (0) option is the default for pictures created prior to iFIX 2022. • **JPEG (2)** – This mode is used to save the Bitmap object with the lossy image compression algorithm (JPEG). A little perceptible loss in image quality is possible when the Bitmap ojbect is saved with JPEG compression. This storage mode is preferred to store large sized Bitmap objects that having real image, natural scene, vivid image, or photograph settings. This mode of storage is not recommended for a Bitmap object where color masking (Transparency) is required.

Remarks

StorageMode is a read-only property of type *Integer*.

StretchMode Property

Describes how color is rendered in a Bitmap object.

Syntax

*object.***StretchMode** [= *enumBitmapStretchMode*]

Properties

The **StretchMode** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumBitmapStretchMode	An enumeration that describes how the bitmap object is rendered. **Valid Values:** • **BitmapStretchBlackOnWhite (1)** – This mode is typically used to preserve foreground pixels in monochrome bitmaps. (This is the default for pictures created prior to iFIX 4.5.) • **BitmapStretchColorOnColor (2)** – This mode is typically used to preserve color in color bitmaps. (This is the default for pictures created in iFIX 4.5 and greater.) • **BitmapStretchHalfTone (3)** – This mode maps pixels from the source rectangle into blocks of pixels in the destination rectangle. The average color over the destination block of pixels approximates the color of the source pixels. This mode is slower and requires more processing of the source image than the other modes, but it produces higher quality images.

StrikeThrough Property

Specifies whether the text has a strike through it.

Syntax

*object.***StrikeThrough** [= *Boolean*]

Properties

The **StrikeThrough** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the text has a strike through it.

Settings

The settings for *Boolean* are:

Value	Description
True	The text is stricken.
False	There is no strikethrough.

SubTitle Property

Allows you to enter a subtitle for your Enhanced Chart.

Syntax

object.**SubTitle** [= *String*]

Properties

The **SubTitle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	Text that appears as the subtitle in the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

SubTitleBold Property

Displays the subtitle for the Enhanced Chart in bold.

Syntax

object.**SubTitleBold** [= *Boolean*]

Properties

The **SubTitleBold** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the subtitle in a HistogramChart, LineChart, SPCBarChart, or XYChart Object displays in bold.

Settings

The settings for *Boolean* are:

Value	Description
True	The subtitle displays in bold.
False	The subtitle does not display in bold. (Default)

SubTitleFont Property

Specifies the font face of the subtitle for the Enhanced Chart.

Syntax

*object.***SubTitleFont** [= *String*]

Properties

The **SubTitleFont** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The name of the font. By default, the font face is "Times New Roman."

SubTitleItalic Property

In an Enhanced Chart, specifies whether the subtitle displays in italics.

Syntax

*object.***SubTitleItalic** [= *Boolean*]

Properties

The **SubTitleItalic** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the subtitle in a HistogramChart, LineChart, SPCBarChart, orXYChart Object displays in italics.

Settings

The settings for *Boolean* are:

Value	Description
True	The subtitle displays in italics. (Default)
False	The subtitle does not display in italics.

SubTitleUnderline Property

In an Enhanced Chart, specifies whether the subtitle appears underlined.

Syntax

*object.***SubTitleUnderline** [= *Boolean*]

Properties

The **SubTitleUnderline** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the subtitle in a HistogramChart, LineChart, SPCBarChart, or XYChart Object appears underlined.

Settings

The settings for *Boolean* are:

Value	Description
True	The subtitle appears underlined.
False	The subtitle does not appear underlined. (Default)

System Property

Returns the dispatch pointer to the **System** object.

Syntax

*object.***System**

Properties

The **System** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

System is a read-only property of type *Object*.

SystemMenu Property

Specifies whether the System Menu is enabled.

Syntax

*object.***SystemMenu** [= *Boolean*]

Properties

The **SystemMenu** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the system menu is enabled.

Settings

The settings for *Boolean* are:

Value	Description
True	The system menu is enabled.
False	The system menu is not enabled.

T

TableBackColor Property

For an Enhanced Chart, specifies the background color of the table, if displayed.

Syntax

*object.***TableBackColor** [= *Long*]

Properties

The **TableBackColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	An whole number representing the color.

TableFont Property

For an Enhanced Chart, specifies the font of the table, if displayed.

Syntax

*object.***TableFont** [= *String*]

Properties

The **TableFont** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The name of the font. By default, the font face is "Arial."

TableForeColor Property

For an Enhanced Chart, specifies the foreground color of the table, if displayed.

Syntax

*object.***TableForeColor** [= *Long*]

Properties

The **TableForeColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	A whole number representing the color.

TextColor Property

Sets the color used as the text color for the object's titles and labels in an Enhanced Chart.

Syntax

*object.***TextColor** [= *Long*]

Properties

The **TextColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	A whole number representing the color.

Thickness Property

Specifies the thickness, in pixels, of the selected pipe object.

Syntax

*object.***Thickness** [=Integer]

Properties

The **Thickness** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The pipe thickness, in pixels.

ThicknessType Property

Specifies whether the thickness type for pipes can be stretched or is fixed.

Syntax

*object.***ThicknessType** [= *enumThicknessType*]

Properties

The **ThicknessType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumThicknessType	An enumeration that represents whether a pipe's thickness can be stretched or is fixed. **Valid entries:** 0 – The pipe's thickness is always what is specified in the Thickness property. 1 – The pipe's thickness can be stretched to accommodate picture scaling.

Thumbnail Property

When the Thumbnail property is set to True in an Enhanced Chart, the chart can be reduced in size and still show useful information. By reducing the size of the chart to a smaller size, multiple charts can be displayed in the same picture.

Syntax

*object.***Thumbnail** [= *Boolean*]

Properties

The **Thumbnail** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Sets Thumbnail property to either True or False.

Settings

The settings for *Boolean* are:

Value	Description
True	The ShowYAxis, ShowXAxis, ShowTimeCursor, ShowLegend, ForceVerticalPoints, ShowDSLegend, and ShowTimeCursorTooltips will be set to the values listed in the following table. These values override any previously configured values for these properties.
False	All properties in following table are rendered to whatever you configure them to. (Default)

The following properties will be set to the values outlined in this table when the Thumbnail property is set to TRUE.

Property	Value when the Thumbnail Property is TRUE
ShowYAxis	Grid_only (3).
ShowXAxis	Grid_only (3).
ShowTimeCursor	FALSE.
ShowLegend	FALSE.
ForceVerticalPoints	PointLabelVertical (1).
ShowDSLegend	TRUE, if the data source legend is being displayed. However, only the Value column will appear.
ShowTimeCursorTooltips	FALSE.

TimeAxis Property

Returns the **Pen** object's **Time Axis**.

Syntax

*object.***TimeAxis**

Properties

The **TimeAxis** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

TimeAxis is a read-only property of type *Object*.

Use this object to access **Time Axis** properties, which include **AxisColor**, **AxisTitle**, **AxisLength**, **NumLabels**, **NumTicks**, **ShowAxis**, and **ShowTitle**.

TimeAxisNumLabels Property

Specifies the number of labels on the **Time Axis**.

Syntax

*object.***TimeAxisNumLabels** [= *Integer*]

Properties

The **TimeAxisNumLabels** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The number of labels on the **Time Axis**.

TimeAxisNumTicks Property

Specifies the number of tick marks on the **Time Axis**.

Syntax

*object.***TimeAxisNumTicks** [= *Integer*]

Properties

The **TimeAxisNumTicks** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The number of ticks on the **Time Axis**.

Remarks

The number of tick marks includes the left and right edge markers. The first tick is the left edge, followed by the right edge, then the middle ticks.

TimeAxisTitle Property

Specifies the title of the specified **Chart** object's **Time Axis**.

Syntax

*object.***TimeAxisTitle** [= *String*]

Properties

The **TimeAxisTitle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The title of the **Time Axis**.

TimeBeforeNow Property

Specifies the initial start time for a **GeneralDataSet**, **Chart**,**Line**, **Lookup Object**, **Formatted Object**, or **Pen** relative to the time the parent **Picture** is opened.

Syntax

*object.***TimeBeforeNow** [= *Long*]

Properties

The **TimeBeforeNow** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The time in seconds.

Remarks

This property is not impacted by any Global Time Control property settings.

TimeCursorColor Property

Specifies the color of the **Chart** object's Time Cursor.

Syntax

*object.***TimeCursorColor** [= *Long*]

Properties

The **TimeCursorColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the time cursor's color.

TimeCursorPos Property

Specifies the horizontal position of the **Chart** object's time cursor in postscript points or logical units relative to the side of the document frame (document relative not chart relative).

Syntax

*object.***TimeCursorPos** [= *Double*]

Properties

The **TimeCursorPos** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The logical horizontal screen relative coordinate.

TimeCursorStyle Property

Specifies the type of time cursor to display in an Enhanced Chart in run mode, when the ShowTimeCursor property is set to true.

Syntax

*object.***TimeCursorStyle** [= *enumTimeC ursorStyle*]

Properties

The **TimeCursorStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumTimeCursorStyle	An enumeration value representing the type of tool tip to display: • **Style_Tooltips (1)** – Displays the time cursor as a tool tip when in run mode. Clicking the plot area displays the time cursor in the area where you clicked. Clicking again will free it up. (The default for charts created in iFIX 5.5 and greater.) • **Style_Table (0)** – Displays the time cursor as an annotation (cross hair cursor) in run mode. This style is the only style used for Enhanced Charts prior to iFIX 5.5. For iFIX 5.5, the cross hair cursor has been replaced with a single vertical line. (Style_Table (0) is the default for charts created prior to iFIX 5.5.)

TimeCursorTooltipColor Property

Specifies the color to display as the background for the specified **Pen** object's time cursor tooltips.

Syntax

*object.***TimeCursorTooltipColor** [= *Long*]

Properties

The **TimeCursorTooltipColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the time cursor's color.

Timeout Property

Specifies the maximum amount of time the **Chart** may be paused (see **Pause**) before it will automatically resume (see **Resume**).

Syntax

*object.***Timeout** [= *Long*]

Properties

The **Timeout** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The maximum amount of time.

Remarks

If the **Timeout** property is set to zero (0), the chart will pause and not update until the Resume method is called.

TimerEnabled Property

Specifies the run-time operation of the **Timer** object.

Syntax

*object.***TimerEnabled** [= *Boolean*]

Properties

The **TimerEnabled** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	The run-time operation of the **Timer** object.

Settings

The settings for *Boolean* are:

Value	Description
True	If set in the Configuration environment, the **Timer** is started when the WorkSpace is switched to the Run-time environment. If set from **False** in the Run-time environment, nothing will happen, the user must call **StartTimer**.
False	If set in the Configuration environment, the **Timer** will not be started when the WorkSpace is switched to the Run-time environment. If set in the Run-time environment, the timer will be stopped.

Timestamp Property

Returns the timestamp of the OPC data source represented by this **DataItem**. The **Timestamp** property becomes populated either from **Read** method of the **DataItem** or **Group (DataSystem)** object.

Syntax

*object.***Timestamp**

Properties

The **Timestamp** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Timestamp is a read-only property of type *Date*.

The **Quality** and **Value** properties are associated with the **Timestamp**.

TimeZoneBiasExplicit Property

Specifies the TimeZoneBiasExplicit.

Syntax

*object.***TimeZoneBiasExplicit** [=*enumTimeZoneBiasExplicit*]

Properties

The **TimeZoneBiasExplicit** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enum TimeZoneBiasExplicit	The explicit time zone for a chart or pen.

Remarks

The **TimeZoneBiasExplicit** property is only used when the **TimeZoneBiasRelative** property is set to explicit time zone.

TimeZoneBiasRelative Property

Specifies the TimeZoneBiasRelative.

Syntax

*object.***TimeZoneBiasRelative** [=*enumTimeZoneBiasRelative*]

Properties

The **TimeZoneBiasRelative** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enum TimeZoneBiasExplicit	The relative time zone for a historical data set, chart, or pen.

Settings

Constant	Value	Description
ClientTimeZone	0	Uses client time zone settings.
ServerTimeZone	1	Uses server time zone settings.
TagTimeZone	2	Uses the tag time zone settings.
ExplicitTImeZone	3	Uses the explicit time zone settings.

Remarks

If the value is explicit, it uses whatever the TimeZoneExplicitBias is.

Titlebar Property

Specifies whether the window's title bar is visible or not.

Syntax

*object.***Titlebar** [= *Boolean*]

Properties

The **Titlebar** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the title bar is visible.

Settings

The settings for *Boolean* are:

Value	Description
True	The title bar is visible.
False	The title bar is not visible.

ToggleRate Property

Specifies the rate at which the output of the Lookup object toggles between *output1* and *output2*. For example, in a color table, this property is the blink rate.

Syntax

object.**ToggleRate** [= *Single*]

Properties

The **ToggleRate** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Single	The toggle rate in seconds.

ToggleSource Property

Specifies the data source or expression used to determine if the output of the Lookup object should toggle between *output1* and *output2*.

Syntax

object.**ToggleSource** [= *String*]

Properties

The **ToggleSource** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The data source or expression.

Remarks

The **ToggleSource** property stores and exposes the fully qualified name.

Tolerance Property

Specifies the tolerance for exact match Lookup tables.

Syntax

object.**Tolerance** [= *Single*]

Properties

The **Tolerance** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Single	The tolerance value.

Remarks

The tolerance will work just like in an expression. If the value is within +/- tolerance, the value will pass the condition imposed – in this case equality.

For example, the user sets up an exact match table such that an object is green when the value is 0 and red when the value is 1. Most likely the value will end up being 0.0000009, and therefore the object may never be green. Tolerance will prevent this.

ToolbarManager Property

Holds information describing the iFIX picture toolbars.

Syntax

object.**ToolbarManager**

Properties

The **ToolbarManager** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ToolbarManager is a read-only property of type *Object*.

ToolbarPath Property

Returns the path used for storing toolbar files.

Syntax

object.**ToolbarPath**

Properties

The **ToolbarPath** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ToolbarPath is a read-only property of type *String*.

TooltipOption Property

When used with the EnableTooltip and HighlightEnabled properties, the TooltipOption property allows you to specify where the tooltip text comes from:

- The object's Description property.
- The Dynamo_Description property.
- From a run mode tag value supplied when you load tag status symbols for your picture.

A tooltip will be displayed for the specified object only when EnableTooltip and HighlightEnabled properties are set to TRUE.

Syntax

*object.***TooltipOption** [= *enumTooltipOptions*]

Properties

The **TooltipOption** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumTooltipOptions	An enumeration that represents the type of description used for the tooltip text: **Valid entries:** 0 – TooltipDescription (uses the configured object's Description) 1 – TooltipTagStatus (allows you to change the object's Description in run mode when you load a new set of tag groups - see LoadTagGroupFile Method) 2 – TooltipDynamoDescription (uses Dynamo object's Description for tooltip text)

Top Property

Specifies the distance, in screen percentage, between the top edge of the physical screen and the WorkSpace in which it is contained, or Specifies the position of the top edge of the **Window** object.

Syntax

*object.***Top** [= *Double*]

Properties

The **Top** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The distance in screen percentage.

TopCenter Property

Returns the value of the top center point of the shape's bounding rectangle as a **FixFloatPoint**.

Syntax

*object.***TopCenter**

Properties

The **TopCenter** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

TopCenter is a read-only property of type *Object*.

TopLeft Property

Returns the top left point of the shape's bounding rectangle as a **FixFloatPoint**.

Syntax

*object.***TopLeft**

Properties

The **TopLeft** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

TopLeft is a read-only property of type *Object*.

TopRight Property

Returns the top right point of the shape's bounding rectangle as a **FixFloatPoint**.

Syntax

*object.***TopRight**

Properties

The **TopRight** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

TopRight is a read-only property of type *Object*.

TopVisibleRow Property

Specifies the first visible row in the **Alarm Summary** object's spreadsheet.

Syntax

*object.***TopVisibleRow**

Properties

The **TopVisibleRow** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

You can use the value of **TopVisibleRow** to compute the visible page size.

TotalFilteredAlarms Property

Displays the total number of alarms that match the current filter. This is the same as the number displayed in the left corner of the status bar.

Syntax

*object.***TotalFilteredAlarms**

Properties

The **TotalFilteredAlarms** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

TranslateOnOpen Property

Specifies whether or not to switch the language of the selected picture when it is opened in run mode.

Syntax

*object.***TranslateOnOpen** [= *Boolean*]

Properties

The **TranslateOnOpen** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the picture's language is switched when opened in run mode.

Settings

The settings for *Boolean* are:

Value	Description
True	The language is switched when the picture is opened in run mode.
False	The language is not switched when the picture is opened in run mode.

Transparency Property

Specifies whether or not the **Bitmap** supports the transparent color.

Syntax

*object.***Transparency** [= *Boolean*]

Properties

The **Transparency** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the bitmap supports the transparent color.

Settings

The settings for *Boolean* are:

Value	Description
True	The bitmap supports transparency.
False	The bitmap does not support transparency.

Remarks

This property enables a **Bitmap** to have a transparent color (**TransparentColor**). When enabled, bitmaps can effectively have a bleed-through area that can be used for animation and visualization.

Transparent Property

Specifies whether or not the **Chart** is transparent.

Syntax

object.**Transparent** [= *Boolean*]

Properties

The **Transparent** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the **Chart** is transparent.

Settings

The settings for *Boolean* are:

Value	Description
True	The **Chart** is transparent.
False	The **Chart** is opaque.

Remarks

Setting the **Transparent** property to **True** allows the user to see what is displayed behind the **Chart**.

TransparentColor Property

Specifies the **Bitmap** object's transparent color. If transparency is enabled, this color is the see-through color. Any pixels with this color display the screen's background image.

Syntax

object.**TransparentColor** [= *Long*]

Properties

The **TransparentColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the transparent color.

TreatSinglePointsAsLines Property

In an Enhanced Chart, specifies whether to draw a straight horizontal line if only one point.

Syntax

*object.***TreatSinglePointsAsLines** [= *Boolean*]

Properties

The **TreatSinglePointsAsLines** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to draw a straight horizontal line if only one point exists in an Enhanced Chart.

Settings

The settings for *Boolean* are:

Value	Description
True	Draws a straight horizontal line if only one point exists in an Enhanced Chart.
False	Does not draw a straight horizontal line if only one point exists in an Enhanced Chart.(Default)

TriggerType Property

Specifies when the timer's **OnTimeOut** event is retriggered.

Syntax

*object.***TriggerType** [= *Long*]

Properties

The **TriggerType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	When the **OnTimeOut** event is triggered.

Settings

The settings for *Long* are:

Value	Description
0	One-shot.
1	Continuous.
2	Daily.
3	Monthly.

Remarks

One-Shot The **Timer** triggers at the start time, then disables itself.

Continuous The **Timer** triggers at the start time then re-schedules the timer based on the current time plus the configured interval.

Daily The **Timer** triggers at the start time for those days that have been enabled (see **DaysOfWeek**). If the user has enabled end time then the timer is retriggered using the interval until the time exceeds the configured end time.

Monthly The **Timer** triggers at the start time for every day the user has enabled (see **DaysOfMonth**). If the user has enabled end time then the timer is retriggered using the interval until the time exceeds the configured end time.

TrimMaxLength Property

Specifies the maximum pixel length of the line to be trimmed. If the pixel length is greater than this number, the line will not be trimmed.

Syntax

*object.***TrimMaxLength** [=*Integer*]

Properties

The **TrimMaxLength** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The maximum number of pixels allowed for the line to be trimmed.

TrimType Property

Specifies the trim option to apply to all line objects.

Syntax

*object.***TrimType** [=*enumTrimType*]

Properties

The **TrimType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumTrimType	When to trim a line at the intersection point.

Settings

The settings for *enumTrimType* are:

Constant	Value	Description
Always	0	Always trim lines.
ShorterThanSpecified	1	Trim only when the trimmed area is shorter than the specified pixels.

TruncateTitles Property

When TruncateTitles is set to true in an Enhanced Chart, the chart title, chart sub-title and axes titles are all truncated to fit the allowable space, before any scaling is applied. In many cases, this results in easier to read titles and labels. When set to false, these titles are all scaled to display the full title, despite how small the font size appears.

Syntax

*object.***TruncateTitles** [= *Boolean*]

Properties

The **TruncateTitles** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Describes whether the chart title, chart sub-title and axes titles are all truncated to fit the allowable space, before any scaling is applied.

Settings

The settings for *Boolean* are:

Value	Description
True	Titles are truncated to fit the allowable space, before any scaling is applies.
False	No titles are truncated, and instead all titles are scaled to fit the allowable space. (Default)

Type Property

Returns the type of document that is displayed by the specified **Document** object.

Syntax

*object.***Type**

Properties

The **Type** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Type is a read-only property of type *Object*.

This property identifies what kind of **Page** is opened for this document. The WorkSpace can open and manage any OLE Active Document object. The **Type** property contains the OLE document type ProgID field as found in the registry for this class. iFIX uses Fix.Picture and FixSchedule.FixSchedulerServer.1.

U-V

UCL Property

Sets the upper control limits (UCL) for the RealTimeSPCDataSet object.

Syntax

object.**UCL** [= *Double*]

Properties

The **UCL** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The upper control limits (UCL) for the RealTimeSPCDataSet Object.

UnacknowledgedAlarmColor Property

Specifies the color in which all unacknowledged alarms will be displayed in the **Alarm Summary** object if the **UseUnacknowledgedAlarmColor** property is **True**.

Syntax

object.**UnacknowledgedAlarmColor** [= *Long*]

Properties

The **UnacknowledgedAlarmColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the unacknowledged alarms.

Underline Property

Specifies whether the text in the **Text** or **Datalink** object is underlined.

Syntax

object.**Underline** [= *Boolean*]

Properties

The **Underline** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the text is underlined.

Settings

The settings for *Boolean* are:

Value	Description
True	The text is underlined.
False	The text is not underlined.

UniformScale Property

Specifies whether a shape scales horizontally and vertically by the same amount.

Syntax

object.**UniformScale** [= *Boolean*]

Properties

The **UniformScale** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the object scales uniformly.

Settings

The settings for *Boolean* are:

Value	Description
True	The shapes scale both horizontally and vertically by the same percentage.
False	The shapes will horizontally and vertically scale independently. (Default)

Remarks

Vertical and horizontal scaling can either be performed independently of one another, or uniformly using the same percentages. Uniform scaling is also used when the user holds the control key down while scaling the object with the mouse.

Setting this flag to **True** before rubber band creation allows you to create a square from the **Rectangle** object and a circle from the **Oval** object.

Units Property

Specifies the units descriptor field for a **Pen**.

Syntax

*object.***Units** [= *String*]

Properties

The **Units** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	Data value units descriptor.

UpdateOnPropChange Property

Specifies whether changes to properties in the **Alarm Summary** object take effect immediately. Set this property to **False** if you are changing multiple properties and you want the changes to take effect simultaneously.

Syntax

*object.***UpdateOnPropChange** [= *Boolean*]

Properties

The **UpdateOnPropChange** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether property changes take immediate effect.

Settings

The settings for *Boolean* are:

Value	Description
True	Changes made to properties take immediate effect. (Default)
False	Changes do not take immediate effect.

UpdateRate Property

Allows you to specify how quickly a chart updates the data plot in run mode. This value is not used for Histogram and SPC charts, as these charts update at the rate of their data sources.

Syntax

*object.***UpdateRate** [= *Long*]

Properties

The **UpdateRate** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Specifies, in Milliseconds, how quickly a chart updates the data plot in run mode.

UseDefaultYAxisSettings Property

When UseDefaultYAxisSettings is set to True for a data source in an XY or LineChart, the following data source properties are configured according to the corresponding values set on the Axis tab in the Enhanced Chart Customization dialog box in the iFIX WorkSpace:

- ManualScaleControlY
- AutoMinMaxPaddingY
- ManualMinY
- ManualMaxY
- YAxisScaleControl
- UseDSLimits

Syntax

*object.***UseDefaultYAxisSettings** [= *Boolean*]

Properties

The **UseDefaultYAxisSettings** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Describes whether the Y axis information is visible for the specified data source.

Settings

The settings for *Boolean* are:

Value	Description
True	The data source properties for ManualScaleControlY, AutoMinMaxPaddingY, ManualMinY, ManualMaxY, YAxisScaleControl and UseDSLimits are configured according to the corresponding values set on the Axis tab in the Enhanced Chart Customization dialog box. (Default)

False	The data source properties for ManualScaleControlY, AutoMinMaxPaddingY, ManualMinY, ManualMaxY, YAxisScaleControl and UseDSLimits are configured according to the corresponding values set on Y Axis sub tab on the Data Sources tab in the Enhanced Chart Customization dialog box.

UseDelta Property

Specifies whether to use the absolute or relative value to set the output value.

Syntax

object.**UseDelta** [= *Boolean*]

Properties

The **UseDelta** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to use the absolute or relative value to set the output value.

Settings

The settings for *Boolean* are:

Value	Description
True	The target current value is added to the calculated output value before the output value is written to its target. (Default)
False	The output value is written to its target with no current value added in.

Remarks

This property controls whether the output value of the animation is written as calculated (absolute) to the property or if this value is used as a delta or offset from the property's initial setting (relative). This property should be enabled if the animation is relative to the current location of the object. If the object has a fixed screen location for animation (even if the object is moved in the Configuration environment) then disable this property. For most position animations this property should be enabled.

If the user sets up a connection to the **HorizontalPosition** property and set **UseDelta** to **True**, the base position of the object will be added to to the output value when the linear object evaluates. For example, the input range for the object is 0 to 100, the output range is 0 to 200, and the initial position of the object is 15. If the value of 50 comes in from the data system and **UseDelta** is **True**, the value of 115 is written to the **HorizontalPosition** of the object. If **UseDelta** is **False**, the value written would be 100.

UseDomainSecurity Property

Sets or retrieves the Boolean value that indicates whether or not Windows domain security is to be used in the security synchronization process.

Syntax

object.**UseDomainSecurity** [= *Boolean*]

Properties

The **UseDomainSecurity** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	The default value is **False**, which indicates that Windows domain security should not be used. **True** indicates that Windows domain security should be used.

Remarks

When the **UseDomainSecurity** property is equal to **True**, you must supply a valid domain name in the **Domain** property.
This property corresponds to the /D command line parameter of the Security Synchronizer application.

UseDSLimits Property

If the UseDSLimits is set to True, then for a given data source in an Enhanced Chart, the high and low values for the Y axis are obtained from the High and Low Limit fields in the Data sub tab on the Data Sources tab in the Enhanced Chart Customization dialog box.

If the UseDSLimits is set to False, then the high and low values for the Y axis are configured using the settings configured in the Y Axis panel of the Axis tab in the Enhanced Chart Customization dialog box.

NOTE: If the Chart is an XY or Line Chart and the UseDefaultYAxisSettings Property is set to False for a given data source, then the Y axis high and low values are configured using the Y Axis sub tab on the Data Sources tab in the Enhanced Chart Customization dialog box.

Syntax

object.**UseDSLimits** [= *Boolean*]

Properties

The **UseDSLimits** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Describes where to get the High and Low values for the Y axis

Settings

The settings for *Boolean* are:

Value	Description
True	Sets the Y axis High and Low values from High and Low Limit fields for a given data source in

	the Data sub tab on the Data Sources tab in the Enhanced Chart Customization dialog box.
False	Sets the Y axis High and Low values from High and Low Limit fields using the settings configured in the Y Axis panel of the Axis tab in the Enhanced Chart Customization dialog box.

UseLocalSecurity Property

Sets or retrieves the boolean value that indicates whether or not Windows local security is to be used in the security synchronization process.

Syntax

*object.***UseLocalSecurity** [= *Boolean*]

Properties

The **UseLocalSecurity** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	The default value is **False**, which indicates that Windows local security should not be used. **True** indicates that Windows local security should be used.

Remarks

This property corresponds to the /L command line parameter of the Security Synchronizer application.

UseMarker Property

Specifies whether to display markers on trend lines for the specified **Pen**.

Syntax

*object.***UseMarker** [= *Boolean*]

Properties

The **UseMarker** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether to display markers.

Settings

The settings for *Boolean* are:

Value	Description
True	Display markers on trend lines.
False	Do not display markers on trend lines.

Remarks

Markers are useful for identifying lines on a black and white printer.

UserDef1ColumnName Property

Specifies the text displayed in the header of the **Alarm Summary** object's User Defined Field1 column.

Syntax

object.**UserDef1ColumnName** [= *String*]

Properties

The **UserDef1ColumnName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The text to display in the header of the User Defined Field1 column.

Remarks

The text you specify cannot match the text of any existing column header, including the text used in the User Defined Field2 column.

UserDef2ColumnName Property

Specifies the text displayed in the header of the **Alarm Summary** object's User Defined Field2 column.

Syntax

object.**UserDef2ColumnName** [= *String*]

Properties

The **UserDef2ColumnName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The text to display in the header of the User Defined Field2 column.

Remarks

The text you specify cannot match the text of any existing column header, including the text used in the User Defined Field1 column.

UserPreferences Property

Holds information describing the iFIX user preferences.

Syntax

*object.***UserPreferences**

Properties

The **UserPreferences** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

UserPreferences is a read-only property of type *Object*.

UseUnacknowledgedAlarmColor Property

Specifies whether the foreground color of unacknowledged alarms is set to the color identified by the UnacknowledgedAlarmColor property.

Syntax

*object.***UseUnacknowledgedAlarmColor** [= *Boolean*]

Properties

The **UseUnacknowledgedAlarmColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the foreground color of unacknowledged alarms is set to the color in the **UnacknowledgedAlarmColor** property.

Settings

The settings for *Boolean* are:

Value	Description
True	The foreground color of all unacknowledged alarms is set to the color set in the **UnacknowledgedAlarmColor** property.
False	The foreground color of all unacknowledged alarms is not set to the color set in the **UnacknowledgedAlarmColor** property. (Default)

UWL Property

Sets the upper warning limits (UWL) for the RealTimeSPCDataSet object.

Syntax

*object.***UWL** [= *Double*]

Properties

The **UWL** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The upper warning limits (UWL) for the RealTimeSPCDataSet Object.

Value Property

Returns the value of the OPC data source represented by this **DataItem**. The **Value** property becomes populated either from **Read** method of the **DataItem** or **Group (DataSystem)** object.

Syntax

*object.***Value**

Properties

The **Value** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Value is a read-only property of type *Object*.

The **Quality** and **Timestamp** properties are associated with the **Value**.

ValueAxis Property

Returns the **Pen** object's **ValueAxis**.

Syntax

*object.***ValueAxis**

Properties

The **ValueAxis** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ValueAxis is a read-only property of type *Object*.

ValueAxisNumLabels Property

Specifies the number of labels on the ValueAxis.

Syntax

object.**ValueAxisNumLabels** [= *Integer*]

Properties

The **ValueAxisNumLabels** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The number of labels on the **Value Axis**.

ValueAxisNumTicks Property

Specifies the number of tick marks displayed on the Value Axis. Tick marks are evenly spaced.

Syntax

object.**ValueAxisNumTicks** [= *Integer*]

Properties

The **ValueAxisNumTicks** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The number of ticks on the **Value Axis**.

ValueAxisTitle Property

Specifies the title of the Value Axis.

Syntax

object.**ValueAxisTitle** [= *String*]

Properties

The **ValueAxisTitle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The title of the **Value Axis**.

VariableType Property

Specifies the **Variable** object's data type for the data it stores. Data retrieved from a data source via connections to its **CurrentValue** property will attempt to be coerced to the defined **VariableType**.

Syntax

*object.***VariableType** [= *enumFixVariableType*]

Properties

The **VariableType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumFixVariableType	The data type.

Settings

The settings for *enumFixVariableType* are:

Constant	Value	Description
tShort	2	Short
tLong	3	Long
tFloat	4	Float
tDouble	5	Double
tString	8	String
tBoolean	11	Boolean

Version Property

Returns the version number of the object.

Syntax

*object.***Version**

Properties

The **Version** property syntax has this part:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.

Remarks

Version is a read-only property of type *Integer*.

Version is initialized to 1 when a “page” (picture, schedule, toolbar, dynamo set, global page) is created.

VerticalFillDirection Property

Specifies a value representing the direction of a shape's vertical fill.

Syntax

object.**VerticalFillDirection** [= *enumVerticalDirection*]

Properties

The **VerticalFillDirection** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumVerticalDirection	The direction of the shape's vertical fill.

Settings

The settings for *enumVerticalDirection* are:

Constant	Value	Description
VerticalFromTop	0	Fill from the top
VerticalFromBottom	1	Fill from the bottom
VerticalFromCenter	2	Fill outward from the center.

VerticalFillPercentage Property

Specifies the percentage of a shape's vertical fill.

Syntax

object.**VerticalFillPercentage** [= *Double*]

Properties

The **VerticalFillPercentage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The percentage to fill the shape.

Remarks

Animating this property performs a dynamic vertical fill.

VerticalGridColor Property

Specifies the color of the vertical grid lines displayed in the Chart.

Syntax

*object.***VerticalGridColor** [= *Long*]

Properties

The **VerticalGridColor** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	The COLORREF used to set the vertical grid color.

VerticalGridStyle Property

Specifies the style of the vertical grid lines displayed in the Chart.

Syntax

*object.***VerticalGridStyle** [= *enumEdgeStyle*]

Properties

The **VerticalGridStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumEdgeStyle	The style of the vertical grid lines.

Settings

The settings for *enumEdgeStyle* are:

Constant	Value	Description
EdgeStyleSolid	0	Solid.
EdgeStyleDash	1	Dash.
EdgeStyleDot	2	Dot.
EdgeStyleDashDot	3	Dash-Dot.
EdgeStyleDashDotDot	4	Dash-Dot-Dot.
EdgeStyleNone	5	No border.

EdgeStyleInsideFrame	6	Inside Frame.

VerticalPosition Property

Specifies a shape's distance, in postscript points or logical units, from the top of the **Picture**.

Syntax

object.**VerticalPosition** [= *Double*]

Properties

The **VerticalPosition** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The shape's vertical position.

Remarks

For shapes, the units are in postscript points (for the Enhanced Coordinate System) or logical units (for the Logical Coordinate System) as defined by the **Picture** document size.

VerticalScaleDirection Property

Specifies if the direction in which the specified shape will expand or contract when the **VerticalScalePercentage** property is changed.

Syntax

object.**VerticalScaleDirection** [= *enumVerticalDirection*]

Properties

The **VerticalScaleDirection** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumVerticalDirection	The direction to scale.

Settings

The settings for *enumVerticalDirection* are:

Constant	Value	Description
VerticalFromTop	0	Fill from the top
VerticalFromBottom	1	Fill from the bottom
VerticalFromCenter	2	Fill outward from the center

Remarks

Scaling from center can be used to create the illusion that an object is moving towards or away from the user.

VerticalScalePercentage Property

Specifies the scale percentage to apply to a shape's height.

Syntax

*object.***VerticalScalePercentage** [= *Double*]

Properties

The **VerticalScalePercentage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The percentage to scale the shape vertically.

Remarks

In the Configuration environment, **VerticalScalePercentage** will not be set back to 100 until the object is de-selected. Therefore, while selected, the object will contain it's current percentage value relative to the size of the object when it was last selected. Once de-selected, the object's **VerticalScalePercentage** property will be reset back to 100.

In the Run-time environment, animating the **VerticalScalePercentage** property modifies the object's height based on the size of the object when it initially came off disk.

An object's scale percentage can be negative. This causes the object to flip over its bottom axis. This effect is useful for creating differential bar graphs by using an expression in the data source that takes the value and subtracts a setpoint. The resulting difference from the setpoint can be used to by the **VerticalScalePercentage** property. For example, you could animate a color table to change color based on the sign of a result.

ViewingStyle Property

In an Enhanced Chart, specifies the viewing style: Color, Monochrome, or Monochrome and Symbols.

Syntax

*object.***ViewingStyle** [= *enumViewingStyle*]

Properties

The **ViewingStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumViewingStyle	An enumeration that represents the viewing style in the Enhanced Chart: **Valid entries:** 0 – ViewColor 1 – ViewMono 2 – ViewMonoWithSymbols

ViewportHeight Property

Specifies the available vertical viewing area of the document.

IMPORTANT: The ViewportHeight property does not apply to documents that use the Enhanced Coordinate system. It is only available for documents using Logical Coordinates.

Syntax

object.**ViewportHeight** [= *Double*]

Properties

The **ViewportHeight** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The available vertical viewing area.

Remarks

ViewportHeight is a logical unit constrained to the document height dimension in documents that use the Logical Coordinate System (legacy coordinates).

ViewportLeft Property

Specifies the leftmost origin of the available viewing area of the document that uses the Logical Coordinate system.

IMPORTANT: The ViewportLeft property does not apply to documents that use the Enhanced Coordinate system. It is only available for documents using Logical Coordinates.

Syntax

object.**ViewportLeft** [= *Double*]

Properties

The **ViewportLeft** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The leftmost origin of the available viewing area.

Remarks

ViewportLeft is a logical unit constrained to the left position of the document, in documents that use the Logical Coordinate System (legacy coordinates).

ViewportTop Property

Specifies the topmost origin of the available viewing area of the document.

IMPORTANT: The ViewportTop property does not apply to documents that use the Enhanced Coordinate system. It is only available for documents using Logical Coordinates.

Syntax

object.**ViewportTop** [= *Double*]

Properties

The **ViewportTop** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The topmost origin of the available viewing area.

Remarks

ViewportTop is a logical unit constrained to the top position of the document, in documents that use the Logical Coordinate System (legacy coordinates).

ViewportWidth Property

Specifies the available horizontal viewing area of the document that uses the Logical Coordinate system.

IMPORTANT: The ViewportWidth property does not apply to documents that use the Enhanced Coordinate System. It is only available for documents using Logical Coordinates.

Syntax

object.**ViewportWidth** [= *Double*]

Properties

The **ViewportWidth** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The available horizontal viewing area.

Remarks

ViewportWidth is a logical unit constrained to the document width dimension in documents that use the Logical Coordinate System (legacy coordinates).

Visible Property

Specifies whether the shape or the dataset appear on screen.

Syntax

object.**Visible** [= *Boolean*]

Properties

The **Visible** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the object or dataset is visible.

Settings

The settings for *Boolean* are:

Value	Description
True	The object or dataset is visible.
False	The object or dataset is not visible.

Remarks

For the RealTimeSPCDataSet Object this property is always set to TRUE, irrespective of the value to which you set it.

VisibleUnacknowledgedAlarms Property

Displays the number of unacknowledged alarms in the currently displayed portion of the alarm summary object. This number is not affected by the alarm summary being partially off screen; it is the number of alarms in the scrolled region of the object.

Syntax

*object.***VisibleUnacknowledgedAlarms**

Properties

The **VisibleUnacknowledgedAlarms** property syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

W-Z

WholeDigits Property

Specifies the number of digits to be displayed before the decimal point.

Syntax

*object.***WholeDigits** [=*Integer*]

Properties

The **WholeDigits** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Integer	The number of digits.

Width Property

Specifies the width, in postscript points or logical units, of the specified object.

Syntax

*object.***Width** [= Double]

Properties

The **Width** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The width of the object.

Remarks

For shapes, the units are in postscript points or logical units defined by the **Picture** document size. The coordinate systems allow pictures to be developed and saved independently of screen resolution. It also supports panning and zooming. Mapping is is based on the window location.

WindowHeightPercentage Property

Specifies the percentage of the vertical screen that the window extends to.

Syntax

*object.***WindowHeightPercentage** [= *Double*]

Properties

The **WindowHeightPercentage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The percentage of vertical screen.

Remarks

WindowHeightPercentage is applied to the client area of the WorkSpace for window position. The value specified in **WindowHeightPercentage** is saved to disk.

WindowLeftPercentage Property

Specifies the percentage of the horizontal screen that the window originates.

Syntax

*object.***WindowLeftPercentage** [= *Double*]

Properties

The **WindowLeftPercentage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The percentage of the horizontal screen.

Remarks

WindowLeftPercentage is applied to the client area of the WorkSpace for window position. The value specified in **WindowLeftPercentage** is saved to disk.

WindowName Property

Specifies the name of the **Window** object.

Syntax

object.**WindowName** [= *String*]

Properties

The **WindowName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	The name of the window.

Remarks

This property stores the string representing the window name which is different than the document name. Window names are useful for managing multiple screens. For example, if you have a main window area and a navigation bar area, and the scripts behind the navigation bar replace whatever picture is in the main window with a specific document, you can use the **WindowName** property to store the string “main”. This allows all scripts to find the window named main and perform the necessary replace **Picture**.

WindowState Property

Specifies the state of the application window of the WorkSpace.

Syntax

object.**WindowState** [= *WinState*]

Properties

The **WindowState** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
state	The state of the window.

Settings

The settings for *WinState* are:

Constant	Value	Description
Normal	1	The window is in the normal state.
Minimized	2	The window is minimized to an icon.
Maximized	3	The window is maximized.

WindowTopPercentage Property

Specifies the percentage of the vertical screen that the window originates.

Syntax

*object.***WindowTopPercentage** [= *Double*]

Properties

The **WindowTopPercentage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The percentage of the vertical screen.

Remarks

WindowTopPercentage is applied to the client area of the WorkSpace for window position.

The value specified in **WindowTopPercentage** is saved to disk.

WindowWidthPercentage Property

Specifies the percentage of the horizontal screen that the window extends to.

Syntax

*object.***WindowWidthPercentage** [= *Double*]

Properties

The **WindowWidthPercentage** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The percentage of horizontal screen.

Remarks

WindowWidthPercentage is applied to the client area of the WorkSpace for window position.

The value specified in **WindowWidthPercentage** is saved to disk.

WizardName Property

Specifies the name of the script configured for the specified event.

Syntax

*object.***WizardName** [= *Variant*]

Properties

The **WizardName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Variant	The name of the script.

Remarks

WizardName gets set when a script authoring expert is run, however, the user can enter any name that describes the function of the script for this object.

WorkSpaceStartupMode Property

Specifies the iFIX startup mode.

Syntax

*object.***WorkSpaceStartupMode***[=bWorkSpaceStartupMode]*

Properties

The **WorkSpaceStartupMode** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bWorkSpaceStartupMode	Boolean. Specifies whether to start the WorkSpace in the Configuration environment or the Runtime environment. Valid Entries: 0 - AppConfigurePicturePreferences 1 - AppRunPicturePreferences

X Property

The horizontal coordinate that corresponds to the given point.

Syntax

*object.***X** [= *Double*]

Properties

The **X** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The horizontal coordinate.

XAxisDatasetPosition Property

Allows you to retrieve or change the position of a X-Axis data set in an XY chart. When changed, it will use the data set in the specified position as the data source for the X axis. This property only applies to XY Enhanced Charts.

Syntax

*object.***XAxisDatasetPosition** *[= Long]*

Properties

The **XAxisDatasetPosition** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Long	Whole number representing the position of the data set.

Remarks

XAxisDatasetPosition is a read-only property.

XAxisLabel Property

In an Enhanced Chart, specifies the x-axis label.

Syntax

*object.***XAxisLabel** [= *String*]

Properties

The **XAxisLabel** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	Text that appears as the x-axis label in the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

XAxisScaleControl Property

In the XY Chart, sets the grid scale used for the X-axis: Normal or Log. This property only applies to XY Enhanced Charts.

Syntax

object.**XAxisScaleControl** [= *enumScaleControl*]

Properties

The **XAxisScaleControl** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumScaleControl	An enumeration that represents the grid scale used for the X-axis in the Enhanced Chart: **Valid entries:** 1 – ScaleNormal 2 – ScaleLog

XAxisType Property

In an Enhanced Chart, specifies the type of x-axis the chart will display: time, point numbers, or point values.

Syntax

object.**XAxisType** [= *enumViewingStyle*]

Properties

The **XAxisType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumViewingStyle	An enumeration that describes the type of x-axis the chart will display in the Enhanced Chart: **Valid entries:** 0 – XAxisType_Time 1 – XAxisType_PointNumbers 2 – XAxisType_PointValues

Remarks

XAxisType is a read-only property.

Y Property

The vertical coordinate that corresponds to the given point.

Syntax

object.**Y** [= *Double*]

Properties

The **Y** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The vertical coordinate.

YAxesStyle Property

The YAxesStyle property allows you to specify a style for the Y axis in a LineChart or XYChart Enhanced Chart.

Syntax

object.**YAxesStyle** [= *enumYAxesStyle*]

Properties

The **YAxesStyle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumYAxesStyle	An enumeration which represents the type of Y axis to use for this chart: • Multiple (1) (Default) • Stack (2) • SingleAxis (3)

YAxisAlwaysVisible Property

Specifies whether the Y axis for this data source is always visible in the Enhanced Chart (LineChart or XYChart).

NOTE: This property applies only when the YAxesStyle Property when is set to Multiple.

Syntax

object.**YAxisAlwaysVisible** [= *Boolean*]

Properties

The **YAxisAlwaysVisible** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Specifies whether the Y axis for this data source is always visible.

Settings

The settings for *Boolean* are:

Value	Description
True	Allows you to make the Y axis always visible. (Default)
False	Leaves the Y axis not visible.

YAxisLabel Property

In an Enhanced Chart, specifies the y-axis label.

Syntax

object.**YAxisLabel** [= *String*]

Properties

The **YAxisLabel** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	Text that appears as the y-axis label in the HistogramChart, LineChart, SPCBarChart, or XYChart Object.

YAxisLongTicks Property

In an Enhanced Chart, use this property to extend/reset the length of minor grid tick marks.

Syntax

object.**YAxisLongTicks** [= *Boolean*]

Properties

The **YAxisLongTicks** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Boolean	Whether the length of minor grid tick marks are extended or reset.

Settings

The settings for *Boolean* are:

Value	Description
True	The length of minor grid tick marks are extended or reset.
False	The length of minor grid tick marks are not extended or reset. (Default)

YAxisScaleControl Property

In an Enhanced Chart, sets the grid scale used for the y-axis: Normal or Log.

Syntax

object.**YAxisScaleControl** [= *enumScaleControl*]

Properties

The **YAxisScaleControl** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumScaleControl	An enumeration that represents the grid scale used for the y-axis in the Enhanced Chart: **Valid entries:** 1 – ScaleNormal 2 – ScaleLog

YAxisTitle Property

Allows you to specify a Y axis title for the specified data source in an Enhanced Chart (LineChart or XYChart).

Syntax

object.**YAxisTitle** [= *String*]

Properties

The **YAxisTitle** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
String	A string that represents the Y axis title for the specified data source in this Enhanced Chart.

Zoom Property

Specifies the current zoom percentage of the specified object. When iFIX scales objects by the zoom factor, coordinate values of the object in the Enhanced Coordinate System do not change. For more information on zooming, refer to the Controlling a Picture's Magnification section in the Creating Pictures e-book.

Syntax

object.**Zoom** [= *Double*]

Properties

The **Zoom** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Double	The percentage to zoom.

ZoomDirection Property

Specifies the direction to zoom in on.

Syntax

object.**ZoomDirection** [= *enumZoomDirection*]

Properties

The **ZoomDirection** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumZoomDirection	The direction to zoom.

Settings

The settings for *enumZoomDirection* are:

Constant	Value	Description
ZoomBoth	0	Zoom both horizontally and vertically (default).
ZoomVertical	1	Zoom vertically.
ZoomHorizontal	2	Zoom horizontally.

ZoomType Property

In an Enhanced Chart, specifies the zoom type capability: Horizontal, Vertical, or Both.

Syntax

object.**ZoomType** [= *enumAllowZooming*]

Properties

The **ZoomType** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enumAllowZooming	An enumeration that represents the zoom type capability in an Enhanced Chart: **Valid entries:** 1 – Zooming_Horizontal

	2 – Zooming_Vertical 3 – Zooming_Both

Method Summary

The following list contains the iFIX object methods that are available to the Automation Interface. For information on non-iFIX methods, refer to the appropriate help system.

A

AboutBox

AckAlarm

AckAlarmPage

AckAlarmPageEx

AckAllAlarms

ActivateWorkspaceUI

Add

AddDataSet

AddEventHandler

AddImage

AddKeyMacro

AddLegendItem

AddLevel

AddObject

AddPen

AddPictureToStartupList

AddPoint

AddProcedure

Align

ApplyProperty

AutoScaleDisplayLimits

B

BringToFront

BuildObject

C

CanConstruct

CheckAccountExpiration

CheckforDuplicateKeyMacros

CheckSecurityEnabled

CheckSyntax

CheckUserApplicationAccess

CheckUserAreaAccess

Clear

ClearUndo

Close

Commit

Connect

ConnectDataSet

ConnectedPropertyCount

Construct

Convert_A_Group_To_A_Dynamo_By_Name

Convert_A_Group_To_A_Dynamo_By_Ref

ConvertPipe

ConvertSecurityAreaNameToNumber

ConvertSecurityAreaNumberToName

ConvertToEnhancedCoordinates

ConvertToOriginalCoordinates

Copy

CopyAsBitmap

CopytoClipboard

Coupled_Activate_Workspace_UI

Coupled_DeActivate_Workspace_UI

CreateDynamoByGrouping

CreateFromDialog

CreateFromProgID

CreateWithMouse

Cut

D

DeActivateWorkspaceUI

DefaultView

DelAlarm

DeleteAllAlarms

DeleteAllDataSets

DeleteDataSet

DeleteImage

DeletePen

DeletePoint

DeleteSelectedObjects

DemandFire

DeselectObject

DestroyObject

DisableNonSelectionEvents

Disconnect

DisplaysControlPoints

DoesPropertyHaveTargets

DoExtendLines

DoLinesToPolyline

DoMenuCommand

DoTrimLines

DumpProperties

Duplicate

E

EditPicture

Enable

Enumerate_All_Dynamos

Enumerate_All_Groups

Enumerate_Top_Level_Dynamos

Enumerate_Top_Level_Groups

ExchangePenPositions

Execute

ExecuteKeyMacro

ExportData

ExportImage

ExportLanguageFile

F

FindAndReplaceDialog

FindInString

FindObject

FindReplaceInObject

FindReplaceInString

FitDocumentToWindow

FitWindowToDocument

FixCheckApplicationAccess

FixCheckApplicationAccessQuiet

FixCheckAreaAccess

FixCheckAreaAccessQuiet

FixCheckSecurityEnabled

FixGetManualAlmDeleteEnabled

FixGetUserInfo

FixLogin

FixLogout

FontProperties

FullView

G-H

Get_Last_Prompt_Value

Get_Last_Result_String

GetAlarmBackgroundColor Method

GetAlarmForegroundColor Method

GetBoundRect

GetChartEndTime

GetChartStartTime

GetColHeadings

GetColumnInfo

GetConnectionInformation

GetConnectionParameters

GetContinuousUser

GetCurrentDataSet

GetCurrentValue

GetCurrentValueWithQuality

GetDataSetByPosition

GetDeviceRect

GetDuration

GetErrorString

GetEventHandlerIndex

GetFullname

GetGlobalDuration

GetGlobalHistoricalUpdateRate

GetIndirectionInfo

GetInterval

GetKeyMacro

GetKeyMacroIndex

GetLevel

GetNumberOfDataSets

GetObjectInfo

GetPenDataArray

GetPenDataArrayEx

GetPointAt

GetPriorityColor

GetProcedureIndex

GetProperty

GetPropertyAttributes

GetPropertyTargets

GetRibbonView

GetSelectedAlmExt

GetSelectedNodeTag

GetSelectedRow

GetSelectedRowAlarmInfo

GetSelectedRowsAlarmInfo

GetSelectedUserDefFields

GetSignature

GetSignatureAndWriteValue

GetStatusColor

GetStatusFont

GetTimeBeforeNow

GetTimeCursorInfo

GetUserID

GetWindowLocation

GlobalScrollBackFast

GlobalScrollBackSlow

GlobalScrollForwardFast

GlobalScrollForwardSlow

GlobalTimerApply

Group

HiLoDisplay

I-K

ImportToolbar

Initialize

InitializeList

InsertPoint

InteractiveExport

IsColorSelectionVisible

IsConnected

IsEmpty

IsKeyMacroDefined

IsNodeSignEnabled

IsSignatureRequired

IsSignatureRequiredForList

Item

L

ListEvents

ListMethods

ListProperties

ListWindowsGroupNames

Load_TS_List

LoadImage

LoadTagGroupFile

LogicalToPercentage

LogicalToUserFormPoint

M-N

MakeLinesHorizontal

MakeLinesVertical

MakeSameSize

Modify

ModifyColumnLength

Move

O

Open

Open_QT_Pic

Open_QT_Pic_Ex

Open_TCP_Pic

Open_TCP_Pic_Ex

Open_TS_Pic

Open_TS_Pic_Ex

Open_TS_Pic_Type

Open_TS_Pic_Type_Ex

P

ParseConnectionSource

Paste

PasteFromClipboard

PasteSpecial

Pause

PauseAlarmRead

PercentageToLogical

PercentageToPixel

PixelToPercentage

PrintChart

PrintOut

PromptToChangePassword

Q

Quit

R

Read

Refresh

RefreshChartData

Remove

RemoveAll

RemoveAllLevels

RemoveItem

RemoveKeyMacro

RemoveLegendItem

RemoveLevel

RemoveObject

RemovePictureFromStartupList

ReplaceDocument

ReplaceInString

Replace_QT_Pic

Replace_TCP_Pic

Replace_TS_Pic

Replace_TS_Pic_Type

ResetChartData

ResetObjectStats

ResetStats

ResetZoom

ResolveTagGroupFile

Resume

ResumeAlarmRead

RetrieveDefinition

RetrieveTagGroupVariables

Rotate

RunObject

S

Save

Save_TS_List

SaveAsSVG

SaveToHistoryList

ScrollBack

ScrollForward

ScrollTimeBack

ScrollTimeForward

ScrollToPosition

Select

SelectAlarmRow

SelectAll

SelectObject

SendOperatorMessage

SendSignedOperatorMessage

SendToBack

SetContinuousUser

SetCurrentValue

SetDispatch

SetDispid

SetDuration

SetFocusToComboBox

SetAlarmBackgroundColor Method

SetAlarmForegroundColor Method

SetGlobalDuration

SetGlobalEndTimeToCurrent

SetGlobalHistoricalUpdateRate

SetGlobalMovingEndTimeToCurrent

SetIndirectionInfo

SetInterval

SetKeyCombination

SetLegendMask

SetNumericFormat

SetPenDataArray

SetPointAt

SetPriorityColor

SetProperty

SetScriptWindow

SetSource

SetStatusColor

SetStatusFont

SetStringFormat

SetTabSelection

SetTimeBeforeNow

SetTimeCursorTime

SetWindowLocation

ShelveAlarm

ShowAnimations

ShowBrowseDialog

ShowColorBox

ShowColorSelection

ShowCustomPages

ShowPipePreviewDialog

ShowTaskWizard

ShowVBAProcedure

ShowVisualBasicEditor

SilenceAlarmHorn

SnapObjectsToGrid

SpaceEvenly

StartEvent

StartTimer

StickToCursor

StopEvent

StopGlobalPlayBack

StopTimer

Stretch

SwitchLanguage

SwitchMode

SynchronizeSecurity

T

TagGroupSubstitution

TagGroupValue

U

UIActivate

UIDeActivate

Undo

UndoTransaction

UndoZoom Method

UnGroup

UnloadTagGroupFile

UnShelveAlarm

Update_A_Dynamo_By_Name

Update_A_Dynamo_By_Name2

Update_A_Dynamo_By_Ref

Update_A_Dynamo_By_Ref2

UpdateBackgroundObject

UpdateConnectionParameters

UpdateDefinition

UserFormPointToLogical

V-W

ValidateSignature

ValidateSignatureAndWriteValue

ValidateSource

ValueTimeFromXY

Write

X-Y

XYFromValueTime

XYHitTest

Z

Zoom

ZoomToFit

A-B

AboutBox Method

Opens the Help About Box for the Alarm SummaryExpressionEditor , orColor Button , object.

Syntax

object.**AboutBox**

Properties

The **AboutBox** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

AckAlarm Method

Acknowledges the alarm for the specified node and tag.

Syntax

object.**AckAlarm**(*sNode, sTag*)

Properties

The **AckAlarm** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
sNode	String. The nodename specifying which alarm is to be acknowledged.
sTag	String. The tagname specifying which alarm is to be acknowledged.

Return Value

Integer. The status of the alarm acknowledgement. Return value of 0 signifies success, non-zero on failure.

AckAlarmPage Method

Acknowledges the currently displayed page of alarms.

Syntax

object.**AckAlarmPage**()

Properties

The **AckAlarmPage** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

Integer. The status of the alarm acknowledgement. Return value of 0 signifies success, non-zero on failure when at least one alarm was not acknowledged.

AckAlarmPageEx Method

Acknowledges the currently displayed page of alarms and displays the Electronic Signature dialog box if any of the tags on that page require Electronic Signatures.

Syntax

object.**AckAlarmPageEx**()

Properties

The **AckAlarmPageEx** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

Integer. The status of the alarm acknowledgement. Return value of 0 signifies success, non-zero on failure when at least one alarm was not acknowledged.

AckAllAlarms Method

Acknowledges all alarms that match the current filter.

Syntax

object.**AckAllAlarms**()

Properties

The **AckAllAlarms** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

NOTE: If you are using the **AckAllAlarms** method on an Alarm Summary OCX, this method checks to ensure that the Allow Acknowledge All Alarms property is enabled. If the property is disabled, no alarms associated with that Alarm Summary OCX are acknowledged.

Return Value

Integer. The status of the alarm acknowledgement. Return value of 0 signifies success, non-zero on failure when at least one alarm was not acknowledged.

ActivateWorkspaceUI Method

Activates the WorkSpace UI after a DeActivateWorkSpaceUI method call in the Configuration environment.

Syntax

*object.***ActivateWorkspaceUI**

Properties

The **ActivateWorkspaceUI** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

This method must be called when the **DeActivateWorkSpaceUI** method has been called in VBA scripts that do not involve user forms.

Add Method

Adds a new:

- Document in the WorkSpace.
- Event handler to the Procedures collection.
- Line to the Lines collection of the **Procedures** collection.
- DataItem to the **DataItems** collection.
- Group (DataSystem) to the Groups collection.
- Key macro to the collection, copying the definition from the passed Key macro object.

Documents Collection Syntax

*object.***Add**(*[Filename], [DisplayOption]*)

Properties

The **Add** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Filename	String. (Optional) The ProgID for the document type. The default is “FIX.Picture”.
DisplayOption	Long. (Optional) Specifies how the document is displayed. Valid entries: 1 – Load only. 2 – Load and activate the document in a hidden window. 3 – Load and display the document normally. (default)

Document Type	ProgID
Fix Picture	Fix.Picture
Fix Dynamo Set	FixDynamoSetServer.FixDynamoSetServer
Fix Schedule	FixSchedule.FixSchedulerServer
Microsoft Word Document	Word.Document
Microsoft Excel Worksheet	Excel.Sheet
Microsoft Excel Chart	Excel.Chart

Return Value

Object. The dispatch pointer to the added document.

Remarks

Add adds new document to the WorkSpace's application object's documents collection. It is equivalent to selecting New from the File menu.

Procedures Collection Syntax

object.**Add** *lType, bstrProcDecl*

Properties

The **Add** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
lType	Long. Reserved.
bstrProcDecl	String. The header string for the procedure.

Lines Collection Syntax

object.**Add** *bstrNewLine, lLineNumber*

Properties

The **Add** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

bstrNewLine	String. The new line of code to be added.
lLineNumber	Long. The line number to place the code within the event handler.

DataItems and Groups Collection Syntax

*object.***Add** (*bstrName*)

Properties

The **Add** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrName	String. The name of the member to add.

Return Value

Object. The dispatch pointer of the item added to the collection.

AddDataSet Method

Allows you to add a data source to a Line Chart, Histogram Chart, or SPC Bar Chart.

Syntax

*object.***AddDataSet** *bstrSourceName [bUseAnyway]*

Properties

The **AddDataSet** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrSourceName	String. The data source tag that you want to add to the chart as the dataset.
bUseAnyway	Variant. (Optional) Allows you to set an undefined object as the data source. The value should be True or False. True indicates a UseAnyway condition. If the data source does not exist and bUseAnyway is set to False, the AddDataSet method generates an error. The default for the bUseAnyway parameter is False.

Return Value

Object – the dispatch pointer of the Dataset added to the chart.

AddEventHandler Method

Adds a new event handler to the **Procedures** collection.

Syntax

*object.***AddEventHandler** *bstrEventName, bstrLinesOfCode, pIndex*

Properties

The **AddEventHandler** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrEventName	String. The name of the event.
bstrLinesOfCode	String. The code to be written to the event in the form of strings.
pIndex	Long. Numerical index of the procedures position in the existing collection. Note: This index is transient, it will change as procedures are added or deleted.

AddImage Method

Loads the specified image and adds it to the end of the list. This image is always the primary image.

Syntax

*object.***AddImage** *bstrFileName*

Properties

The **AddImage** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrFileName	String. The file name of the image to load.

AddKeyMacro Method

Adds a new KeyMacroObject.

Syntax

AddKeyMacro (*ComboKey, KeyCode*)

Properties

The **AddKeyMacro** method syntax has these parts:

Part	Description
ComboKey	enumCombinationKey. The control shift part of the key combination.
KeyCode	Integer. The ASCII value of the main key of the key combination.

AddLegendItem Method

Adds an item to the Legend of the Chart at the specified location displaying the specified number of characters.

Syntax

*object.***AddLegendItem** *szItem, iColumn, iNumChars*

Properties

The **AddLegendItem** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
szItem	String. The name of the item to be added. Valid entries: Source - Data Source name Description - Data Source's descriptor property Value - Current Value at the time cursor Units - EGU units name Mode - Historical or real time High Limit - High display limit Low Limit - Low display limit Interval - Data point interval High Over - Highest value over the duration Low Over - Lowest value over the duration Avg Over - Average value over the duration USER1 - User defined field USER2 - User defined field USER3 - User defined field USER4 - User defined field USER5 - User defined field USER6 - User defined field USER7 - User defined field USER8 - User defined field USER9 - User defined field USER10 - User defined field
iColumn	Integer. 1-based column index representing where to position the legend item. Column 1 is furthest to the left. Maximum is 22.
iNumChars	Integer. Defines the size of the display width of the column specified by iColumn. The width is calculated by taking the average size character of the font selected multiplied by the number of characters specified by iNumChars. Maximum is 80.

AddLevel Method

Adds a new level to the table (Lookup object).

Syntax

*object.***AddLevel** *pInput1, pOutput1, [pInput2], [pOutput2]*

Properties

The **AddLevel** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pInput1	Variant. The first input parameter. Used as the lookup value when performing exact match lookups, and the minimum value for range comparison lookups.
pOutput1	Variant. The primary output value.
pInput2	Variant. (Optional) The second input parameter. Used as maximum value for a range lookup comparison.
pOutput2	Variant. (Optional) The secondary output value for this level. This value is used by the lookup object as the "blink to" value at this level.

AddObject Method

Adds an object to a **Group**.

Syntax

*object.***AddObject** *dispObject*

Properties

The **AddObject** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pdispObject	Object. The object that you want to add to the group. When an object is added to a group it is put at the top of the group's stacking order (drawing order).

Remarks

We recommend that you build groups either by using the **BuildObject** method on the group object, or using the **Group** method on the **Picture** object to group the selected objects.

AddPen Method

Adds a new **Pen** to the **Chart** object's **Pens** collection.

Syntax

*object.***AddPen***(szSource, [vaUseAnyway])*

Properties

The **AddPen** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
szSource	String. The data source tag or expression to use to fill in the data for the **Pen**. Any valid historical or real-time data source expression is supported. The only restriction is that Historical tags must not be used in an expression.
aUseAnyway	Variant. (Optional) It is used to determine whether to use the data source if the source does not exist. Values should be **True** or **False**. If source does not exist and vaUseAnyway is **False**, AddPen will generate an error. The default for the parameter is **False**.

Return Value

Object. The dispatch pointer of the **Pen** added to the **Chart**.

RefreshChartData must be called after changing the definition of a Pen.

AddPictureToStartupList Method

Adds pictures to the iFIX WorkSpace's startup lists. The startup lists determine the pictures that will be opened automatically when the WorkSpace starts.

Syntax

*object.***AddPictureToStartupList** *bstrPictureName, bMode*

Properties

The **AddPictureToStartupList** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrPictureName	String. Name of the picture to be added to the startup list. You must include the file path and extension
bMode	Boolean. Specifies whether to open the picture when the WorkSpace starts in the Configuration environment or in the Runtime environment. **Valid entries:** 0 – AppConfigurePicturePreferences 1 – AppRunPicturePreferences
plndex	Long. Numerical index of the procedures position in the existing collection. Note: This index is transient, it will change as procedures are added or deleted.

AddPoint Method

Adds a new data point to the object.

Syntax

*object.***AddPoint** *pdispPoint*

Properties

The **AddPoint** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pdispPoint	Object. An OLE object specifying a point object to add to the list of existing data points. The point object has an (x,y) pair that contains the logical coordinate of the data point (see **FixFloatPoint**).

Remarks

A point is an OLE object specifying a point object to add to the list of existing data points. The point object has an (x,y) pair that contains the logical coordinates of the data point.

AddProcedure Method

Adds a new Procedure to the **Procedures** collection.

Syntax

*object.***AddProcedure** *bstrProcName, bstrParamList, bstrLinesOfCode, pIndex*

Properties

The **AddProcedure** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrProcName	String. The name of the Procedure to be added.
bstrParamList	String. The parameter list of the added Procedure.
bstrLinesOfCode	String. The lines of code to be added to the Procedure.
pIndex	Long. The index within the Procedures collection that the new Procedure occupies.

Align Method

Aligns the selected objects.

Syntax

*object.***Align***type*

Properties

The **Align** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
type	Integer. Specifies the align method type. **Valid entries:** 0 – Left 1 – Vertical Center 2 – Right 3 – Top 4 – Horizontal Center 5 – Bottom

Remarks

Align is a Configuration environment method only.

ApplyProperty Method

Reserved for internal purposes.

AutoScaleDisplayLimits Method

Changes the display limits configured for the **Pen** based on the range of the data currently in the pen's data array. If called on the **Chart** object, this method changes the display limits for all pens.

Syntax

*object.***AutoScaleDisplayLimits**

Properties

The **AutoScaleDisplayLimits** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

This method is useful for automatically zooming in or out on a data set to get a full EGU span of the currently displayed data, particularly when the data source's value range varies widely and manual display limit adjustments are not practical for the operator.

BringToFront Method

Moves the selected object to the front of the display stacking order, making it the top object in the stack. It is equivalent to selecting Bring To Front from the Format menu.

Syntax

*object.***BringToFront**

Properties

The **BringToFront** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

BringToFront is a Configuration environment method only.

If you select several objects and call **BringToFront**, the selected objects are placed at the top of the stack, however, they keep their positions relative to one other. The **BringToFront** method is useful for creating complex shapes and using stacking or masking techniques.

BuildObject Method

Adds a new iFIX object to the system. All iFIX objects, pictures, schedules, groups, and shapes can contain other objects. Typically, when creating objects, you would call **BuildObject** on the currently active page object (Application.ActiveDocument.Page). **BuildObject** adds shapes to a graphic, events to a schedule and animations to a shape.

Syntax

object.**BuildObject**(*bstrClassName*)

Properties

The **BuildObject** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrClassName	String. The class name of the object to build. Valid entries: Arc, Bitmap, Chart, Chord, Datalink, Fixevent, Format, Group, Line, Linear, Lookup, OleObject, Oval, Pen, Pie, Polyline, Polygon, Rect, RoundRect, Text, Fixtimer, Variable, LineChart, XYChart, SPCBarChart, and HistogramChart

Return Value

Object. The dispatch pointer of the object that was created.

Remarks

BuildObject inserts the desired object into the **ContainedObjects** collection of the object for which **BuildObject** was called.

BuildObject is a configuration and run mode environment method.

NOTE: The objects that are created in run mode (via scripting) are discarded (not persisted) when the picture is switched back to configure mode.

C

CanConstruct Method

Checks a data source reference for valid syntax.

Syntax

*object.***CanConstruct** *bstrObjectName, bCanConstruct*

Properties

The **CanConstruct** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrObjectName	String. The fully qualified data source reference. For example, Fix32.SCADA1.AI1
bCanConstruct	Boolean. Returns **True** if data source reference is valid, **False** otherwise.

Remarks

This method will always return **False** for datasystems external to the Fix32 datasystem.

The *bCanConstruct* parameter is returned indicating if the *bstrObjectName* reference can be constructed (QuickAdd). If **True**, you can safely call **Construct** to display the appropriate user interface to add the reference in the associated data system.

Currently, you need the FIX 6.15 Integration toolkit to create blocks directly without the Database Manager dialog boxes.

CheckAccountExpiration Method

Checks a Windows user account and returns account password expiration information.

Syntax

object.**CheckAccountExpiration***(bstrUsername, pbExpired, pbCanChangePassword, pnDaysLeft)*

Properties

The **CheckAccountExpiration** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrUserName	String. Specifies the Windows user name.
pbExpired	Boolean. Returns **True** if the Windows user account password is expired, and **False** if it is not.
pbCanChangePassword	Boolean. Returns **True** if the user is allowed to change their password, and **False** if he is not.
pnDaysLeft	Long. Returns the number of days left until the account password expires.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

CheckforDuplicateKeyMacros Method

Checks the passed collection of key macros for duplicates.

Syntax

*object.***CheckforDuplicateKeyMacros** (*KeyMacrosToBeChecked, DuplicateKeyMacros*)

Properties

The **CheckforDuplicateKeyMacros** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
KeyMacrosToBeChecked	FixKeyMacroCollection. Collection of KeyMacro objects to be checked against this object.
DuplicateKeyMacros	FixKeyMacroCollection. Collection of the Duplicate KeyMacro objects found.

CheckSecurityEnabled Method

Checks if iFIX security is enabled on the local node.

Syntax

object.**CheckSecurityEnabled***(pbSecurityEnabled)*

Properties

The **CheckSecurityEnabled** method syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
pbEnabled	Boolean. Returns **True** if security is enabled, **False** if it is not.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

CheckSyntax Method

Tests the syntax of an expression in the Expression Builder.

Syntax

object.**CheckSyntax***szExpression*

Properties

The **CheckSyntax** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
szExpression	String. The expression string to check.

Return Value

Boolean. Returns True if the syntax check was successful.

CheckUserApplicationAccess Method

Checks the user's access to an application feature.

Syntax

object.**CheckUserApplicationAccess***(bstrUserID, nApplicationID, pbAccess, [bQuiet])*

Properties

The **CheckUserApplicationAccess** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrUserID	String. The user ID for the user.
nApplicationID	Long. Numeric identifier associated with the application feature. For a list of application IDs, refer to the FIXcheckApplicationAccess method.
pbAccess	Boolean. Returns **True** if the user has access to the specified application feature, or **False** if the user does not have access.

bQuiet	Boolean. (Optional). If set to **True**, you prevent security violation messages from being sent to the alarm system when this access check fails. If set to **False**, iFIX sends a security violation message. The default value is **False**.

CheckUserAreaAccess Method

Checks a user's access to a security area.

Syntax

object.**CheckUserAreaAccess**(*bstrUserID, bstrSecurityArea, pbAccess, [bQuiet])*

Properties

The **CheckUserAreaAccess** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrUserID	String. The user ID for the user.
bstrSecurityArea	String. Name of the security area to check.
pbAccess	Boolean. Returns **True** if the user has access to the specified security area, or **False** if the user does not have access.
bQuiet	Boolean. (Optional). When set to **True**, you prevent security violation messages from being sent to the alarm system when this access check fails. When set to **False**, iFIX sends a security violation message. **False** is the default value.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

Clear Method

Removes all items out of the collection.

Syntax

*object.***Clear**

Properties

The **Clear** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ClearUndo Method

Removes all currently registered undo transactions from the undo stack.

Syntax

object.**ClearUndo**

Properties

The **ClearUndo** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ClearUndo is a Configuration environment method only.

Close Method

Closes the **Document** or **Window**. When called off the **Documents** collection, this method closes all open documents in the iFIX WorkSpace. When used with the **Document** object, this method closes the **Document** object. This is equivalent to selecting Close from the File menu.

Documents Collection Syntax

object.**Close** *[SaveChanges], [CloseOptions]*

Properties

The **Close** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
SaveChanges	Long. (Optional) **Valid entries:** 1 – Saves the changes without prompting. (default) 2 – Does not save the changes. 3 – Prompts the user to save changes.
CloseOptions	Long. (Optional) **Valid entries:** 1 – Closes all files. (default) 2 – Only closes the files that are not hidden. 3 – Only closes the hidden files.

Remarks

This method removes the document(s) from the **Documents** Collection.
Note that if the **Close** method is called for newly created pictures with a *SaveChanges* option of 1 or 3, the user is prompted to save changes regardless. This is because pictures cannot be saved as their

default name (e.g. “Untitled#”).

Document Object Syntax

*object.***Close** *[SaveChanges], [CloseOptions]*

The **Close** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
SaveChanges	Long. (Optional) **Valid entries:** 1 – Saves the changes without prompting. (default) 2 – Does not save the changes. 3 – Prompts the user to save changes.
CloseOptions	Long. (Optional) **Valid entries:** 1 – Close file. (default) 2 – Close file, only if it is visible. 3 – Close file, only if hidden.

Remarks

If possible, always make this call the last line in your script. Note that when the **Close** method is used, and the document being closed is the document that contains the script, the call *must* be the last line in the script. Otherwise, you may experience unexpected behavior when executing the script.

If the **Close** method is not called from picture being closed and is not the last line in your script, be certain that the operation is complete before the rest of the script continues to execute.

Commit Method

Commits changes made from the object's initialization into the object. This method is usually called after the **BuildObject** method.

Syntax

*object.***Commit**

Properties

The **Commit** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Connect Method

Connects two object's properties together including connecting to an external data source. Once the connection is successfully made, changes in the source's data are automatically transferred to the connected property. Use **Connect** to make animation connections to data source tags. **Connect** supports expressions as the definition of a source.

Syntax

object.**Connect** *bstrPropertyName, bstrSource, iStatus, [fTolerance], [iFlags], [fDeadband], [fUpdateRate]*

Properties

The **Connect** method syntax has the following parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrPropertyName	String. The name of the object's property to animate.
bstrSource	String. The data source reference definition.
iStatus	Long. Returns the error value. Return values are: 0 – OK 1 – Syntax error 2 – Data Undefined 3 – Data type mismatch
fTolerance	Variant. (Optional) Used to evaluate the definition of equal in an expression.
Iflags	Variant. (Optional) Reserved.
fdeadband	Variant. (Optional) Used to determine how much the source needs to change before the exception is transferred to the connected property.
fUpdateRate	Variant. (Optional) The requested maximum update rate for this connection in seconds. This property is used as a hint to the data system for a poll rate to the OPC server.

ConnectDataSet Method

This method is currently unavailable for use in iFIX. It is reserved for future use.

Syntax

*object.***ConnectDataSet ()**

Properties

The **ConnectDataSet** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ConnectedPropertyCount Method

Returns the number of properties that have connections configured.

Syntax

object.**ConnectedPropertyCount** *iConnectedPropertyCount*

Properties

The **ConnectedPropertyCount** method syntax has the following parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
IConnectedPropertyCount	Long. Returns the number of properties that have connections.

Construct Method

Creates an external data source object (tag reference). It displays the QuickAdd user interface that prompts the user for the information needed to create the tag.

Syntax

object.**Construct** *bstrObjectName, iStatus*

Properties

The **Construct** method syntax has the following parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrObjectName	String. The fully qualified data source reference. For example, Fix32.SCADA1.AI1
iStatus	Long. Returns the error value. Return values are: 0 – OK 1 – Syntax error 2 – Data Undefined 3 – Data type mismatch

Convert_A_Group_To_A_Dynamo_By_Name Method

Converts an old Dynamo Instance (Group object) to a new Dynamo object. If you want a converted Dynamo object to become a Master Dynamo, you must also select the “Make Master” option.

NOTE: If you want to be able to use the Dynamo Updater with a converted Dynamo object, you need to call the Update_A_Dynamo_By_Name method after the Convert_A_Group_To_A_Dynamo_By_Name method. Through this process, the Dynamo object obtains the Dynamo_ID and Revision matched with the provided Master Dynamo.

Syntax

*object.***Convert_A_Group_To_A_Dynamo_By_Name** (*bstrGroupName, nOptions, bstrChoiceDialogTitle, pnResultCode*)

Properties

The **Convert_A_Group_To_A_Dynamo_By_Name** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrGroupName	String. The name of the iFIX group object.
nOptions	Long. The specified options as a bitmask: UPDATE_OPTION_ON_MISMATCH_UPDATE_DYNAMO (0x00000001) – When a mismatch is encountered, use the Update / Do not update options. This setting is unused when iFIX finds no mismatches. UPDATE_OPTION_ON_MISMATCH_APPLY_DATA_SOURCES (0x00000002) – When a mismatch is encountered, use the Apply Data Sources / Do not apply data sources options. This setting is unused when iFIX finds no mismatches. UPDATE_OPTION_ON_MISMATCH_PROMT_FOR_CHOICE (0x00000004) – When a mismatch is encountered, ask the user what to do. UPDATE_OPTION_RESIZE_INSTANCE (0x00000008) – Set to True to resize the Dynamo instance to match the Master Dynamo dimensions. Equivalent setting in iFIX 4.5 is always True. UPDATE_OPTION_SAVE_CAPTIONS (0x00000010) – Set to True to save the captions on text objects and button objects. UPDATE_OPTION_UPDATE_ON_CONVERSION (0x00000020) – Set to True to ignore the Dynamo_ID and Revision checking when updating. CONVERT_OPTION_MAKE_MASTER (0x00010000) – Dynamo Converter Options. Convert a group to a Master Dynamo.
bstrChoiceDialogTitle	String. Currently not used, but available in case a future version of iFIX needs to display a choice within a dialog box during the conversion.
pnResultCode	The result code as a bitmask: UPDATER_RESULT_SUCCESS_BIT (0x00000001) – Returns 1 on success, or 0 on failure. UPDATER_RESULT_ALL_DATA_SOURCES_IGNORED_BIT (0x00000002) – Encoded status bit. UPDATER_RESULT_SOME_DATA_SOURCES_IGNORED_BIT (0x00000004) – Encoded status bit.

	UPDATER_RESULT_NOT_ENOUGH_DATA_SOURCES_BIT (0x00000008) – Encoded status bit. UPDATER_RESULT_DYNAMO_NOT_UPDATED_BIT (0x00000010) – Encoded status bit. UPDATER_RESULT_USER_CANCELLED_BIT (0x00000020) – Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_DYNAMO_INSTANCE_BIT (0x00000040) – Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_MASTER_DYNAMO_BIT (0x00000080) – Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_MISTMATCH_OPTION_BIT (0x00000100) – Spare entry, use as needed. UPDATER_RESULT_INVALID_POINTER_BIT (0x00000200) – Spare entry, use as needed. UPDATER_RESULT_INSTANCE_DOESNT_MATCH_MASTER_BIT (0x00000400)– Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_OWNER_BIT (0x00000800) – Spare entry, use as needed. UPDATER_RESULT_TEXT_CAPTIONS_UPDATED_BIT (0x00001000) – Returns 1 if any text captions were updated, or 0 if none were updated. UPDATER_RESULT_TEXT_CAPTIONS_ALL_UPDATED_BIT (0x00002000) – Returns 1 if all captions were updated, or 0 if some (or none) were updated. This field must be 0 if the UPDATER_RESULT_TEXT_CAPTIONS_UPDATED_BIT is also 0. UPDATER_RESULT_INVALID_ARG_GROUP_BIT (0x00004000) – Invaid argument for Convert_A_Group_To_A_Dynamo property.

Convert_A_Group_To_A_Dynamo_By_Ref Method

Converts an old Dynamo Instance (Group object) to a new Dynamo object. If you want a converted Dynamo object to become a Master Dynamo, you must also select the “Make Master” option.

NOTE: If you want to be able to use the Dynamo Updater with a converted Dynamo object, you need to call the Update_A_Dynamo_By_Name method after the Convert_A_Group_To_A_Dynamo_By_Name method. Through this process, the Dynamo object obtains the Dynamo_ID and Revision matched with the provided Master Dynamo.

Syntax

object.**Convert_A_Group_To_A_Dynamo_By_Ref** (*pIGroupDynamo, nOptions, bstrChoiceDialogTitle, pnResultCode*)

Properties

The **Convert_A_Group_To_A_Dynamo_By_Ref** method syntax has these parts:

Part	**Description**
object	An object expression that evaluates to an object in the Applies To list.
pIGroupDynamo	Object. The name of the iFIX Dynamo group object.
nOptions	Long. The specified options as a bitmask: UPDATE_OPTION_ON_MISMATCH_UPDATE_DYNAMO (0x00000001) – When a mismatch is encountered, use the Update / Do not update options. This setting is unused when iFIX finds no mismatches. UPDATE_OPTION_ON_MISMATCH_APPLY_DATA_SOURCES (0x00000002) – When a mismatch is encountered, use the Apply Data Sources / Do not apply data sources options. This setting is unused when iFIX finds no mismatches. UPDATE_OPTION_ON_MISMATCH_PROMT_FOR_CHOICE (0x00000004) – When a mismatch is encountered, ask the user what to do. UPDATE_OPTION_RESIZE_INSTANCE (0x00000008) – Set to True to resize the Dynamo instance to match the Master Dynamo dimensions. Equivalent setting in iFIX 4.5 is always True. UPDATE_OPTION_SAVE_CAPTIONS (0x00000010) – Set to True to save the captions on text objects and button objects. UPDATE_OPTION_UPDATE_ON_CONVERSION (0x00000020) – Set to True to ignore the Dynamo_ID and Revision checking when updating. CONVERT_OPTION_MAKE_MASTER (0x00010000) – Dynamo Converter Options. Convert a group to a Master Dynamo.
bstrChoiceDialogTitle	String. Currently not used, but available in case a future version of iFIX needs to display a choice within a dialog box during the conversion.
pnResultCode	The result code as a bitmask: UPDATER_RESULT_SUCCESS_BIT (0x00000001) – Returns 1 on success, or 0 on failure. UPDATER_RESULT_ALL_DATA_SOURCES_IGNORED_BIT (0x00000002) – Encoded status bit. UPDATER_RESULT_SOME_DATA_SOURCES_IGNORED_BIT (0x00000004) – Encoded status bit. UPDATER_RESULT_NOT_ENOUGH_DATA_SOURCES_BIT (0x00000008) – Encoded status bit.

	UPDATER_RESULT_DYNAMO_NOT_UPDATED_BIT (0x00000010) – Encoded status bit.
	UPDATER_RESULT_USER_CANCELLED_BIT (0x00000020) – Spare entry, use as needed.
	UPDATER_RESULT_INVALID_ARG_DYNAMO_INSTANCE_BIT (0x00000040) – Spare entry, use as needed.
	UPDATER_RESULT_INVALID_ARG_MASTER_DYNAMO_BIT (0x00000080) – Spare entry, use as needed.
	UPDATER_RESULT_INVALID_ARG_MISTMATCH_OPTION_BIT (0x00000100) – Spare entry, use as needed.
	UPDATER_RESULT_INVALID_POINTER_BIT (0x00000200) – Spare entry, use as needed.
	UPDATER_RESULT_INSTANCE_DOESNT_MATCH_MASTER_BIT (0x00000400)– Spare entry, use as needed.
	UPDATER_RESULT_INVALID_ARG_OWNER_BIT (0x00000800) – Spare entry, use as needed.
	UPDATER_RESULT_TEXT_CAPTIONS_UPDATED_BIT (0x00001000) – Returns 1 if any text captions were updated, or 0 if none were updated.
	UPDATER_RESULT_TEXT_CAPTIONS_ALL_UPDATED_BIT (0x00002000) – Returns 1 if all captions were updated, or 0 if some (or none) were updated. This field must be 0 if the UPDATER_RESULT_TEXT_CAPTIONS_UPDATED_BIT is also 0.
	UPDATER_RESULT_INVALID_ARG_GROUP_BIT (0x00004000) – Invaid argument for Convert_A_Group_To_A_Dynamo property.

ConvertPipe Method

Converts the selected lines and/or polylines to pipes.

Syntax

*object.***ConvertPipe**

Properties

The **ConvertPipe** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

ConvertPipe is a Configuration environment method only.

ConvertSecurityAreaNameToNumber Method

Converts the specified security area name to its corresponding area number.

Syntax

object.**ConvertSecurityAreaNameToNumber**(*AreaName*)

Properties

The **ConvertSecurityAreaNameToNumber** method syntax has the following parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
AreaName	String. The area name to be converted.

Return Value

Integer. The security area number.

ConvertSecurityAreaNumberToName Method

Converts the specified security area number to its corresponding area name.

Syntax

object.**ConvertSecurityAreaNumberToName**(*iAreaID*)

Properties

The **ConvertSecurityAreaNumberToName** method syntax has the following parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
iAreaID	Integer. The area number to be converted.

Return Value

String. The security area name.

ConvertToEnhancedCoordinates Method

Converts the x or y coordinate value of a point in legacy Logical Coordinates to the corresponding value in Enhanced Coordinates.

Syntax

*object.***ConvertToEnhancedCoordinates** (*ldInXYValue*, *bIsXValue*)

Properties

The **ConvertToEnhancedCoordinates** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ldInXYValue	Double. The X or Y value of the Logical Coordinate that needs conversion.
bIsXValue	Boolean. When converting a value that is in X axis direction, True is passed for this parameter. When converting value that is in Y axis direction, False is passed for this parameter.

Return Value

Double. The Return value from API is now in postscript points.

Remarks

Use this method to convert hardcoded values in scripts of pictures that have been upgraded to use Enhanced Coordinates.

ConvertToOriginalCoordinates Method

Converts the x or y coordinate value of a point in an Enhanced Coordinate picture to the corresponding value in the legacy Logical Coordinates.

Syntax

*object.***ConvertToOriginalCoordinates** (*ldInXYValue*, *bIsXValue*)

Properties

The **ConvertToOriginalCoordinates** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ldInXYValue	Double. The X or Y value of the Enhanced Coordinate that needs conversion.
bIsXValue	Boolean. When converting a value that is in X axis direction, True is passed for this parameter. When converting value that is in Y axis direction, False is passed for this parameter.

Return Value

Double.

Remarks

Use this method to convert hardcoded values in scripts of pictures that have been upgraded to use Enhanced Coordinates.

Copy Method

Copies the selected objects to the Clipboard in iFIX internal format.

Syntax

object.**Copy**

Properties

The **Copy** method syntax has the following part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Copy is a Configuration environment method only.

CopyAsBitmap Method

Copies the selected objects to the Clipboard in a bitmap format.

Syntax

*object.***CopyAsBitmap**

Properties

The **CopyAsBitmap** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

CopyAsBitmap is a Configuration environment method only.

CopytoClipboard Method

Copies the collection of key macros to the Clipboard.

Syntax

*object.***CopytoClipboard** ()

Properties

The **CopytoClipboard** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Coupled_Activate_Workspace_UI Method

Activates the WorkSpace UI after a Coupled_DeActivate_Workspace_UI method call in the Configuration environment.

Syntax

*object.***Coupled_Activate_Workspace_UI**(*bForce*)

Properties

The **Coupled_Activate_Workspace_UI** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bForce	Boolean. If True, objects can be selected and moved in the currently active document. The right mouse menu and double click events are disabled. In addition, the active document cannot be moved, closed, maximized, or minimized. If False, objects cannot be selected or moved in the currently active document.

Remarks

This method must be called when the **Coupled_DeActivate_Workspace_UI** method has been called in VBA scripts that do not involve user forms.

Coupled_DeActivate_Workspace_UI Method

Determines whether or not a form is modeless in the Configuration environment.

Syntax

*object.***Coupled_DeActivate_Workspace_UI** *bLeaveActiveDocActive*

Properties

The **Coupled_DeActivate_Workspace_UI** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bLeaveActiveDocActive	Boolean. If **True**, objects can be selected and moved in the currently active document. The right mouse menu and double click events are disabled. In addition, the active document cannot be moved, closed, maximized, or minimized. If **False**, objects cannot be selected or moved in the currently active document.

Remarks

User forms are modal by default. To make a form modeless, make the following call on form initialization (in the form's **Activate** event):

```
Application.Coupled_Activate_Workspace_UI True
```

This call must be made from a form that is activated by a toolbar script.

CreateDynamoByGrouping Method

Creates a Dynamo object using the currently selected objects.

Syntax

object.**CreateDynamoByGrouping**

Properties

The **CreateDynamoByGrouping** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

CreateFromDialog Method

Opens the Insert Object dialog box to allow the user to select which ActiveX control to create.

Syntax

*object.***CreateFromDialog**

Properties

The **CreateFromDialog** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Although **CreateFromDialog** applies to many objects in *I*FIX, it is only intended to be used for the **ControlContainer** object.

CreateFromProgID Method

Creates an ActiveX control using its specified ProgID.

Syntax

*object.***CreateFromProgID** *bstrProgId*

Properties

The **CreateFromProgID** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrEventName	String. The name of the event.

Remarks

CreateFromProgID allows the user to create ActiveX controls without requiring any user interface. The ProgIDs can be found in the Insert Object dialog box that can be opened in the WorkSpace by choosing OLE Object off of the Insert menu or by calling the **CreateFromDialog** method.

Although this method applies to many objects in iFIX, it is only intended to be used for the **ControlContainer** object.

CreateWithMouse Method

Allows the user to create the object by using the mouse to set its size and position in the Configuration environment.

Syntax

*object.***CreateWithMouse**

Properties

The **CreateWithMouse** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Cut Method

Removes the selected objects out of the container and puts them on the Clipboard. This is equivalent to selecting Cut from the Edit menu.

Syntax

*object.***Cut**

Properties

The **Cut** method syntax has this part:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.

Remarks

Cut is a Configuration environment method that only works when cutting objects in another document.

D-E

DeActivateWorkspaceUI Method

Determines whether or not a form is modeless in the Configuration environment.

Syntax

*object.***DeActivateWorkspaceUI** *bLeaveActiveDocActive*

Properties

The **DeActivateWorkspaceUI** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bLeaveActiveDocActive	Boolean. If **True**, objects can be selected and moved in the currently active document. The right mouse menu and double click events are disabled. In addition, the active document cannot be moved, closed, maximized, or minimized. If **False**, objects cannot be selected or moved in the currently active document.

Remarks

User forms are modal by default. To make a form modeless, make the following call on form initialization (in the form's **Activate** event):

```
Application.DeActivateWorkspaceUI True
```

This call must be made from a form that is activated by a toolbar script.

DefaultView Method

NOTE: This method only applies to legacy Logical Coordinate System pictures. It does not apply to Enhanced Coordinates.

Restores a document to its default configuration.

Syntax

*object.***DefaultView** *bRedraw*

Properties

The **DefaultView** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bRedraw	Boolean. (Optional) If **True,** repaint the document. (Default) If **False,** make the changes but don't repaint the document.

DelAlarm Method

Deletes the alarm for the requested node and tag.

Syntax

*object.***DelAlarm**(*sNode, sTag*)

Properties

The **DelAlarm** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
sNode	String. The nodename specifying which alarm is to be deleted.
sTag	String. The tagname specifying which alarm is to be deleted.

Return Value

Integer. The status of the alarm deletion. Return value of 0 signifies success, non-zero on failure.

DeleteAllAlarms Method

Deletes all alarms that match the current filter.

Syntax

*object.***DeleteAllAlarms**()

Properties

The **DeleteAllAlarms** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

Integer. The status of the alarm deletion. Return value of 0 signifies success, non-zero on failure when at least one alarm was not deleted.

DeleteAllDataSets Method

Deletes all data sets from the Enhanced Chart.

Syntax

*object.***DeleteAllDataSets ()**

Properties

The **DeleteAllDataSets** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

DeleteDataSet Method

Deletes a specified data set from an Enhanced Chart.

Syntax

*object.***DeleteDataSet** *nDSPosition*

Properties

The **DeleteDataSet** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
nDSPosition	Long. The position of the data set within chart that you want to delete from chart object. Position is 0 based.

DeleteImage Method

Deletes both the primary and secondary images loaded at the specified index.

Syntax

*object.***DeleteImage** *nIndex*

Properties

The **DeleteImage** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

nIndex	Integer. The index of the image to delete. This is a one-based index.

DeletePen Method

Deletes a **Pen** from the **Chart**.

Syntax

*object.***DeletePen** *uiIndex*

Properties

The **DeletePen** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
uiIndex	Integer. Index of the pen to be deleted in the **Pens** collection.

DeletePoint Method

Deletes the point at the given index.

Syntax

*object.***DeletePoint** *lIndex*

Properties

The **DeletePoint** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
lIndex	Long. The position of the point to delete.

DeleteSelectedObjects Method

Deletes the objects in the **SelectedShapes** collection.

Syntax

object.**DeleteSelectedObjects**

Properties

The **DeleteSelectedObjects** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

DeleteSelectedObjects is a Configuration environment method that only works when deleting objects in another document.

DemandFire Method

Forces the object to fire its main event.

Scheduler Object Syntax

object.**DemandFire** *bstrObjectName*

Properties

The **DemandFire** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrObjectName	String. The name of the object in the schedule for which you want to fire an event.

Timer and Event Object Syntax

object.**DemandFire**

Properties

The **DemandFire** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

For the **Scheduler** object, **DemandFire** triggers the event associated with the name of the object in the **Scheduler**. For the **Timer** object, **DemandFire** fires the **OnTimeOut** event. For the **Event** object, the event that the user has configured in the user interface fires. This can be **OnTrue**, **OnFalse**, **WhileTrue**, **WhileFalse**, or **DataChange**.

You can use this method to force the action to trigger, which is useful for debugging or for complex scheduling strategies.

DeselectObject Method

Deselects the object.

Syntax

object.**DeselectObject** *bDeselectAll*

Properties

The **DeselectObject** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bDeselectAll	Boolean. If **True**, all items that were selected are deselected. If **False**, deselects the last selected object.

Remarks

Removes this object in its parent object's **ContainedSelections** collection as well as from the picture's **SelectedShapes** collection.

DestroyObject Method

Deletes the object.

Syntax

object.**DestroyObject**

Properties

The **DestroyObject** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

DisableNonSelectionEvents Method

Reserved for internal purposes.

Disconnect Method

Removes a property connection that was built using **Connect**.

Syntax

object.**Disconnect** *bstrPropertyName*

Properties

The **Disconnect** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrPropertyName	String. The name of the property that has the connection.

Remarks

If more that one connection is built for a single property, this method deletes all of the connections.

DisplaysControlPoints Method

Determines whether or not a shape displays its control points for selecting, resizing, and rotating, etc.

Syntax

*object.***DisplaysControlPoints** *bDisplaysControlPoints*

Properties

The **DisplaysControlPoints** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bDisplaysControlPoints	Boolean. If **True**, the shape displays control points. If **False**, the shape does not display control points.

DoesPropertyHaveTargets Method

Determines if a property has a target object connection. That is, if the local property changes, will this change be sent to any other objects in the system. This list of subscribed connections are called targets.

Syntax

object.**DoesPropertyHaveTargets** *bstrPropertyName, bHasTargets, iNumberOfTargets, iStatus, iIndex*

Properties

The **DoesPropertyHaveTargets** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrPropertyName	String. The name of the local property to check for targets.
bHasTargets	Boolean. Returns **True** if any targets are configured for the property name.
iNumberOfTargets	Long. Returns the number of target objects.
iStatus	Long. Returns the error value. Return values are: 0 – OK

	1 – Syntax error 2 – Data Undefined 3 – Data type mismatch
iIndex	Long. Returns the handle for the passed in property that you can pass to the **GetPropertyTargets** method to obtain the list of targets associated with this property.

DoExtendLines Method

Extends the selected lines to the intersection point(s) of another selected line.

Syntax

*object.***DoExtendLines**

Properties

The **DoExtendLines** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

DoLinesToPolyline Method

Converts the selected lines to a polyline.

Syntax

*object.***DoLinesToPolyline**

Properties

The **DoLinesToPolyline** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

DoMenuCommand Method

Executes the specified menu command.

Syntax

object.**DoMenuCommand** *iMenuCommand*

Properties

The **DoMenuCommand** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
iMenuCommand	Enumeration. The menu command you wish to perform. **Valid entries:** SchSearchReplace SchAboutHelp SchCut SchCopy SchPaste SchDelete SchProperties SchNewTimer SchNewEvent scHREFreshView

DoTrimLines Method

Trims the selected lines to each intersection point.

Syntax

*object.***DoTrimLines**

Properties

The **DoTrimLines** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

DumpProperties Method

Writes the property names and the corresponding values of an object's properties to an ASCII file.

Syntax

object.**DumpProperties** *pbstrDumpFilename, bDumpChildren, bstrDumpProperty, bstrDumpValue*

Properties

The **DumpProperties** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pbstrDumpFilename	String. The name of the file in which to store the output.
bDumpChildren	Boolean. If **True**, the dump file will contain the object's children's properties.
bstrDumpProperty	String. Reserved.
bstrDumpValue	String. Reserved.

Duplicate Method

Copies the selected objects in the container. This is equivalent to selecting Duplicate from the Edit menu.

Syntax

object.**Duplicate**

Properties

The **Duplicate** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

The **Duplicate** method is a configure-time method. **Duplicate** copies any object currently selected. Furthermore, a duplicated object also duplicates its children (contained objects). New names are generated for each new object and all inter-object relationships (connections, containment, internal script object references) are changed to use the new names of the duplicated objects.

NOTE: For Enhanced Charts (LineChart, XY, Histogram, and SPC) the Duplicate method can be used in run mode as well.

EditPicture Method

Opens the Picture's configuration dialog box.

Syntax

object.**EditPicture**

Properties

The **EditPicture** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

EditPicture is a Configuration environment method only.

Enable Method

Enables or disables a **Color Button**.

Syntax

*object.*__Enable__*bEnable*

Properties

The **Enable** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bEnable	Boolean. If **True**, the **Color Button** is enabled. If **False**, the **Color Button** is disabled.

Enumerate_All_Dynamos Method

Returns a list (collection) of Dynamo objects, including those in groups, for a specified collection. The specified collection is usually in picture's contained object collection or that of a Dynamo Set.

Syntax

object.**Enumerate_All_Dynamos** *(pdispObjCollectionToSearch, ppdispObjCollectionOfDynamos)*

Properties

The **Enumerate_All_Dynamos** method syntax has these parts:

Part	Description
object	A FixGeometryHelper object.
pdispObjCollectionToSearch	A collection of objects that you want to search for Dynamo objects.
ppdispObjCollectionOfDynamos	The collection of Dynamo objects that were found in the search.

Enumerate_All_Groups Method

Enumerates the Groups in the supplied list.

Syntax

*object.*__Enumerate_All_Groups__ *(pdispObjCollectionToSearch, ppdispObjCollectionOfDynamos)*

Properties

The **Enumerate_All_Groups** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pdispObjCollectionToSearch	A collection of objects that you want to search for group objects.
ppdispObjCollectionOfDynamos	The collection of group objects that were found in the search.

Enumerate_Top_Level_Dynamos Method

Returns a list (collection) of Dynamo objects, not including Dynamos found within groups, for a specified collection.

Syntax

object.**Enumerate_Top_Level_Dynamos** *(pdispObjCollectionToSearch, ppdispObjCollectionOfDynamos)*

Properties

The **Enumerate_Top_Level_Dynamos** method syntax has these parts:

Part	Description
object	A FixGeometryHelper object.
pdispObjCollectionToSearch	A collection of objects that you want to search for Dynamo objects.
ppdispObjCollectionOfDynamos	The collection of Dynamo objects that were found in the search.

Enumerate_Top_Level_Groups Method

Enumerates the top level groups in the supplied list.

Syntax

*object.***Enumerate_Top_Level_Groups** *(pdispObjCollectionToSearch, ppdispObjCollectionOfDynamos)*

Properties

The **Enumerate_Top_Level_Groups** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pdispObjCollectionToSearch	A collection of objects that you want to search for group objects.
ppdispObjCollectionOfDynamos	The collection of group objects that were found in the search.

Remarks

We recommend that you build groups either by using the **BuildObject** method on the group object, or using the **Group** method on the **Picture** object to group the selected objects.

ExchangePenPositions Method

Sets the location of a **Pen** within the **Pens** collection.

Syntax

*object.***ExchangePenPositions** *iNewPos, iOldPos*

Properties

The **ExchangePenPositions** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
iNewPos	Integer. The new location of the pen.
iOldPos	Integer. The current location of the pen.

Remarks

Note that when this method is called for a **Pen**, all pens starting at that index in the collection are shifted down one (for example, their index is incremented by one).

Execute Method

Used to force execution of the KeyMacro's associated procedure.

Syntax

*object.***Execute**()

Properties

The **Execute** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

Return Type	Description
Boolean	Indication whether or not the key was processed

ExecuteKeyMacro Method

Searches the collection of the passed key combination, if a matching key macro object is found the script associated with that key macro object is executed.

Syntax

*object.***ExecuteKeyMacro** (*ComboKey, KeyCode*)

Properties

The **ExecuteKeyMacro** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ComboKey	enumCombinationKey. The control shift part of the key combination.
KeyCode	Integer. The ASCII value of the main key of the key combination.

ExportData Method

Exports the data in the Enhanced Chart in text format to the clipboard or to a file.

Syntax

*object.***ExportData** *bstrDest, enuExportStyle, [blnIncludingLabels], [blnUseMaximumPrecision]*]

Properties

The **ExportData** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrDest	String. Empty for the system clipboard. To export to a file, use the full path.
enuExportstyle	Enum/Long. Specifies various styles with which the data is exported. It can be either of type enumDataExportStyle or a corresponding numerical value, as follows: DataExportStyle_CommaSeparatedList (0): a comma separated list of data. DataExportStyle_TabSeparatedList (1): a tab separated list of data. DataExportStyle_HorizontalTable (2): a table in which each row has data for one dataset. DataExportStyle_VerticalTable (3): a table in which each column has data for one dataset. **NOTE:** In order to use the enumImageFormat enumerations, you must add the type library file for object to the references of the VBA project. If the type library file is not included in the references, then

only numerical values are accepted. The type libraries for objects in the Applied To list are as follows:

Object	Reference	Type Library File Name
HistogramChart	iFix 2D Histogram Chart Object v1.0 Type Library	Fix2DHistogramChartDll.tlb
LineChart	iFix 2D Line Chart Object v1.0 Type Library	Fix2DLineChartDll.tlb
SPCBarChart	iFix 2D SPC Bar Chart Object v1.0 Type Library	Fix2DSPCBarChartDll.tlb
XYChart	iFix 2D XY Chart Object v1.0 Type Library	Fix2DXYChartDll.tlb

blnIncludingLabels	Boolean. Specifies whether X-axis and Y-axis labels will be exported along with data. Defaults to True if unspecified.
blnUseMaximumPrecision	Boolean. Specifies whether the maximum precision or the current precision should be used. Defaults to False (current precision) if unspecified.

ExportImage Method

Takes a snapshot of the specified Enhanced Chart object and exports the resultant image to a file or the system clipboard. You can specify the file format for the exported image.

Syntax

*object.***ExportImage** *bstrDest, enuFormat, enuSizeUnits, dblWidth, dblHeight, [lngDPI], [blnLargeFont]*

Properties

The **ExportImage** method syntax has these parts:

<table>
<tr><th>Part</th><th>Description</th></tr>
<tr><td><i>object</i></td><td>An object expression that evaluates to an object in the Applies To list.</td></tr>
<tr><td><i>bstrDest</i></td><td>String. Leave empty for the system clipboard. To export to a file, use the full path.</td></tr>
<tr><td><i>enuFormat</i></td><td>Enum/Long. Specifies the desired image format. It can be either of type enumImageFormat or a corresponding numerical value, as follows:
ImageFormat_WMF (0): WMF format
ImageFormat_BMP (1): BMP format
ImageFormat_JPG (2): JPG format
ImageFormat_PNG (3): PNG format
ImageFormat_EMF (5): EMF format
<b>NOTE:</b> In order to use the enumImageFormat enumerations, you must add the type library file for object to the references of the VBA project. If the type library file is not included in the references, then only numerical values are accepted. The type libraries for objects in the Applied To list are as follows:
<table>
<tr><th>Object</th><th>Reference</th><th>Type Library File Name</th></tr>
<tr><td>HistogramChart</td><td>iFix 2D Histogram Chart Object v1.0 Type Library</td><td>Fix2DHistogramChartDll.tlb</td></tr>
<tr><td>LineChart</td><td>iFix 2D Line Chart Object v1.0 Type Library</td><td>Fix2DLineChartDll.tlb</td></tr>
<tr><td>SPCBarChart</td><td>iFix 2D SPC Bar Chart Object v1.0 Type Library</td><td>Fix2DSPCBarChartDll.tlb</td></tr>
<tr><td>XYChart</td><td>iFix 2D XY Chart Object v1.0 Type Library</td><td>Fix2DXYChartDll.tlb</td></tr>
</table></td></tr>
<tr><td><i>enuSizeUnits</i></td><td>Enum/Long. Specifies in what units the image size is specified with <i>dblWidth</i> and <i>dblHeight</i>. It can be either of type enumSizeUnits or a corresponding numerical value, as follows:
<b>NOTE:</b> Not all the formats support all the units.
<table>
<tr><th>Numerical Value</th><th>Format</th><th>Comments</th></tr>
<tr><td>SizeUnits_ Pixels (0): in pixels.</td><td>BMP, JPG, PNG</td><td>This is the only supported unit for these formats.</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><td>SizeUnits_NoSpecificSize (0)</td><td>WMF</td><td>Specifies that the WMF will use the maximum size (8 inches for the longer dimension and 1200 DPI) and dblWidth and dblHeight only determine the aspect ratio of the image. In such cases, dblWidth and dblHeight should fall between 1 and 10000 inclusively.</td></tr>
<tr><td>SizeUnits_Millimeters (1): in millimeters.</td><td>WMF, EMF</td><td>Dimensions must be between 25 and 432 millimeters.</td></tr>
<tr><td>SizeUnits_Inches (2): in inches.</td><td>WMF, EMF</td><td>Dimensions must be between 1 and 17 inches.</td></tr>
<tr><td>SizeUnits_Points (3): in Postscript points (1/72 inches).</td><td>WMF, EMF</td><td>Dimensions must be between 72 and 1224 points.</td></tr>
</table>
<p><b>NOTE:</b> In order to use the enumSizeUnits enumerations, you must add the type library file for object to the references of the VBA project. If the type library file is not included in the references, then only numerical values are accepted. The type libraries for objects in the Applied To list are as follows:</p>
<table>
<tr><th>Object</th><th>Reference</th><th>Type Library File Name</th></tr>
<tr><td>HistogramChart</td><td>iFix 2D Histogram Chart Object v1.0 Type Library</td><td>Fix2DHistogramChartDll.tlb</td></tr>
<tr><td>LineChart</td><td>iFix 2D Line Chart Object v1.0 Type Library</td><td>Fix2DLineChartDll.tlb</td></tr>
<tr><td>SPCBarChart</td><td>iFix 2D SPC Bar Chart Object v1.0 Type Library</td><td>Fix2DSPCBarChartDll.tlb</td></tr>
<tr><td>XYChart</td><td>iFix 2D XY Chart Object v1.0 Type Library</td><td>Fix2DXYChartDll.tlb</td></tr>
</table>
</td></tr>
<tr><td><i>dblWidth</i></td><td>Double. Specifies the width of the image, in the units specified in <i>enuSizeUnits</i>.</td></tr>
<tr><td><i>dblHeight</i></td><td>Specifies the heigth of the image, in the units specified in <i>enuSizeUnits</i>.</td></tr>
<tr><td><i>lngDPI</i></td><td>Long. Specifies the dot density of the image, in dots per inch. Defaults to 300 if left</td></tr>
</table>

	unspecified. It must fall between 50 and 600; otherwise it will be ignored. This parameter does not apply to WMF format if *enuSizeUnits* is set to SizeUnits_NoSpecificSize.
blnLargeFont	Boolean. Specifies whether large font should be used in the chart image. Defaults to False if left unspecified.

Remarks

Regardless of format and dimension, the aspect ratio must be between 0.1 and 10.

ExportLanguageFile Method

Exports the language file for the specified picture or pictures. You can specify the language for an individual picture, or you can specify the language for all pictures.

The method *Object.ExportLanguageFile*, uses the **LanguageDesired** property to export language files.

The method *Object.ExportLanguageFile (xxxx)*, uses the specified language (xxxx) to export language files.

Syntax

*object.***ExportLanguageFile** [*LanguageDesired as Long*]

Properties

The **ExportLanguageFile** method syntax has these parts:

Part	**Description**
object	An object expression that evaluates to an object in the Applies To list.
LanguageDesired Long (Optional)	Default CA_Catalan CS_Czech DA_Danish DE_German EL_Greek EN_English ES_Spanish FI_Finnish HU_Hungarian IT_Italian JA_Japanese KO_Korean NL_Dutch NO_Norwegian PL_Polish RU_Russian SR_Cyrillic

	HR_Croatian SK_Slovak SV_Swedish TH_Thai TR_Turkish IN_Indonesian SL_Slovenian EU_Basque ZHTW_Chinese - Taiwan FR_French PTBR_Brazilian Portuguese PT_Portuguese ZHCH_Chinese PRC FRCA_French Canadian

F

FindAndReplaceDialog Method

Opens the WorkSpace's Find And Replace dialog box.

Syntax

*object.***FindAndReplaceDialog**

Properties

The **FindAndReplaceDialog** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

FindAndReplaceDialog is a Configuration environment method only.

For non-UI find and replace substitutions, see the **FindReplace** object.

FindInString Method

Finds the first occurrence of one string within another.

Syntax

*object.***FindInString** *bstrTargetString, lStartChar, bstrFindString, lFlags, pbstrMarkedMatchString, plFirstChar, plCharCount, pbFound*

Properties

The **FindInString** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrTargetString	String. String to be searched.
lStartChar	Long. One-based index of character in target string at which to start the search.
bstrFindString	String. String to search for.
lFlags	Long. Search modifiers. Valid entries: 0-15 in any of the following combinations: 0 - No modifiers 1 - Match Case 2 - Whole Word Only 4 - Data Source Only 8 - Include Scripts
pbstrMarkedMatchString	String. Returns sub-string extracted from target string with marker characters inserted.
plFirstChar	Long. Returns a one-based index of first character in the target string which is involved in the match with the find string.
plCharCount	Long. Returns the number of characters in the target string which were involved in the match of the find string.
pbFound	Boolean. Returns **True** if a match was found, **False** otherwise. If **False**, the *pbstrMarkedMatchString* is set to null.

Remarks

The *pbstrMarkedMatchString*, *plFirstChar*, and *plCharCount* parameters provide the user with the information needed to perform a subsequent **ReplaceInString** operation.

FindObject Method

Locates an object in the system or in the specified object's **ContainedObjects** collection.

Syntax

system.**FindObject**(*bstrFullyQualifiedName*)

Properties

The **FindObject** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

bstrFullyQualifiedName	String. The name of the object you want to find.

Return Value

Object. The dispatch pointer of the object.

Remarks

FindObject must be called off the **System** object when searching for data items. **FindObject** off any other object looks for the specified object within scope of that object. That is, **FindObject** off a **Group** will only look for the object within that **Group**.

FindReplaceInObject Method

Finds all occurrences of a string in the specified object and replaces them with another string.

Syntax

*object.***FindReplaceInObject** *pdispObject, lFlags, bstrFindString, bstrReplacementString, pbSuccess*

Properties

The **FindReplaceInObject** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pdispObject	Object. Object to be searched.
lFlags	Long. Search modifiers. **Valid entries:** 0-15 in any of the following combinations: 0 - No modifiers 1 - Match Case 2 - Whole Word Only 4 - Data Source Only 8 - Include Scripts
bstrFindString	String. String to be searched for.
bstrReplacementString	String. String to be substituted for all occurrences of *bstrFindString*.
pbSuccess	Boolean. Returns **True** if the operation succeeded, **False** otherwise.

FindReplaceInString Method

Finds the first occurrence of a string within a specified string and replaces it with another.

Syntax

*object.***FindReplaceInString** *pbstrTargetString, lStartChar, bstrFindString, bstrReplacementString, lFlags, plFirstChar, plCharCount, pbstrResultString, pbSuccess*

Properties

The **FindReplaceInString** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pbstrTargetString	String. The name of the event.
lStartChar	String. The code to be written to the event in the form of strings.
bstrFindString	String. The string to search for.
bstrReplacementString	String. The string to be substituted for the first occurrence of the find string.
lFlags	Long. Search modifiers. **Valid entries:** 0-15 in any of the following combinations: 0 - No modifiers 1 - Match Case 2 - Whole Word Only 4 - Data Source Only 8 - Include Scripts
plFirstChar	Long. Returns a one-based index of the first character in the target string which was involved in the match with the find string.
plCharCount	Long. Returns the number of characters in the target string which were involved in the match of the find string.
pbstrResultString	String. Returns the new string with the specified replacement.
pbSuccess	Boolean. Returns **True** if the operation succeeded, **False** otherwise.

FitDocumentToWindow Method

Changes the size of the document so that the entire document can be seen in the window. In instances where the document is bigger than the window and scrollbars are present, the document is shrunk so that it fits within the window.

IMPORTANT: The FitDocumentToWindow property does not apply to documents that use the Enhanced Coordinate system. It is only available for documents using Logical Coordinates.

Syntax

*object.***FitDocumentToWindow** *[bRedraw]*

Properties

The **FitDocumentToWindow** method syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
bRedraw	Boolean. (Optional) If **True**, repaint the document. (Default) If **False**, make the changes but don't repaint the document.

FitWindowToDocument Method

Changes the size of the window (within the limits of the WorkSpace client area) to fit the size of the document.

IMPORTANT: The FitWindowToDocument property does not apply to documents that use the Enhanced Coordinate system. It is only available for documents using Logical Coordinates.

Syntax

object.**FitWindowToDocument** *[bRedraw]*

Properties

The **FitWindowToDocument** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bRedraw	Boolean. (Optional) If **True**, repaint the document. (Default) If **False**, make the changes but do not repaint the document.

FixCheckApplicationAccess Method

Checks to see if the user has access to the specified application.

Syntax

object.**FixCheckApplicationAccess**(*ApplicationID*)

Properties

The **FixCheckApplicationAccess** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ApplicationID	Integer Enumeration. The ID that corresponds to the application the user is interested in.

Return Value

Long. Returns 1 if the user has access to the specified application, 0 otherwise.

FixCheckApplicationAccessQuiet Method

Checks to see if the user has access to the specified application. The **FixCheckApplicationAccessQuiet** method refrains from sending a security violation message if the user does not have access.

Syntax

object.**FixCheckApplicationAccessQuiet**(*ApplicationID*)

Properties

The **FixCheckApplicationAccessQuiet** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ApplicationID	Integer Enumeration. The ID that corresponds to the application the user is interested in.

Return Value

Long. Returns 1 if the user has access to the specified application, 0 otherwise.

FixCheckAreaAccess Method

Checks to see if the user has access to the specified area.

Syntax

object.**FixCheckAreaAccess**(*Area*)

Properties

The **FixCheckAreaAccess** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Area	Integer. The area that the user is trying to access.

Return Value

Long. Returns 1 if the user has access to the specified area, 0 otherwise.

Remarks

The mapping of area number to area name can be seen in the Security Configurator under area configuration.

FixCheckAreaAccessQuiet Method

Checks to see if the user has access to the specified area. The **FixCheckAreaAccessQuiet** method refrains from sending a security violation message if the user does not have access.

Syntax

*object.***FixCheckAreaAccessQuiet**(*Area*)

Properties

The **FixCheckAreaAccessQuiet** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Area	Integer. The area that the user is trying to access.

Return Value

Long. Returns 1 if the user has access to the specified area, 0 otherwise.

Remarks

The mapping of area number to area name can be seen in the Security Configurator under area configuration.

FixCheckSecurityEnabled Method

Checks to see if security is enabled.

Syntax

*object.***FixCheckSecurityEnabled**()

Properties

The **FixCheckSecurityEnabled** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

Long. Returns 1 if the security is enabled, 0 otherwise.

FixGetManualAlmDeleteEnabled Method

Returns information about whether the manual alarm deletion feature is enabled in the System Configuration Utility (SCU).

Syntax

*object.***FixGetManualAlmDeleteEnabled**

Properties

The **FixGetManualAlmDeleteEnabled** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

FixGetUserInfo Method

Returns information about the currently logged in user, including the user name and group name.

Syntax

*object.***FixGetUserInfo** *UserID, UserName, GroupName*

Properties

The **FixGetUserInfo** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
UserID	String. Returns the current user ID.
UserName	String. Returns the current user name.
GroupName	String. Returns the first group to which the user belongs.

FixLogin Method

Logs in the user using the specified user ID and password.

Syntax

*object.***FixLogin** *UserID, Password*

Properties

The **FixLogin** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
UserID	String. The users ID.
Password	String. The users password.

FixLogout Method

Logs out the user.

Syntax

*object.***FixLogout**

Properties

The **FixLogout** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

FontProperties Method

Displays the Font dialog box.

Syntax

*object.***FontProperties**

Properties

The **FontProperties** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

FullView Method

NOTE: This method only applies to legacy Logical Coordinate System pictures. It does not apply to Enhanced Coordinates.

Resize the document to take up the entire screen.

Syntax

*object.***FullView** *[bRedraw]*

Properties

The **FullView** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bRedraw	Boolean. (Optional) If **True**, repaint the document. (Default) If **False**, make the changes but don't repaint the document.

G-H

Get_Last_Prompt_Value Method

Mostly for internal use. Returns the value of the last prompt for choice in the dialog selection. This value is set only during a Dynamo Update process. Advanced users developing their own tools instead of using the Dynamo Updater Wizard or Quick Dynamo Updater may find this method useful.

Syntax

object.**Get_Last_Prompt_Value**

Properties

The **Get_Last_Prompt_Value** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

Integer. Returns the result string of the last operation. The following table lists the possible values.

Value	Description
0	Selection dialog box prompt not displayed.
1	Perform update. Do not apply Data Sources. Apply to selected.
2	Perform update. Do not apply Data Sources. Apply to all.
3	Perform update. Attempt to match single Data Source. Apply to selected.
4	Perform update. Attempt to match Data Source. Apply to all.
5	Do not update. Apply to selected.
6	Do not update. Apply to all.
7	Cancel button clicked.

This string is only set during a Dynamo Updater process.

Get_Last_Result_String Method

Retrieves the last prompt text string for the specified Dynamo object.

Syntax

object.**Get_Last_Result_String**

Properties

The **Get_Last_Result_String** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

String. Returns the last prompt text string for the Dynamo prompt.

GetAlarmBackgroundColor Method

Returns the row background color configured for the specified alarm priority.

Syntax

*object.***GetAlarmBackgroundColor***(PriorityId)*

Properties

The **GetAlarmBackgroundColor** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
PriorityId	Integer. The ID corresponding to the alarm priority. **Valid entries for iFIX 3.5 and earlier:** 0 - High 1 - Medium 2 - Low **Valid entries for iFIX 4.0 and later:** 3 - CRITICAL 4 - HIHI 5 - HIGH 6 - MEDIUM 7 - LOW 8 - LOLO 9 - INFO (INFORMATIONAL)

Return Value

OLE_COLOR. The row background color configured for the specified alarm priority.

GetAlarmForegroundColor Method

Returns the row foreground color configured for alarms with the specified status.

Syntax

*object.***GetAlarmForegroundColor***(StatusId)*

Properties

The **GetAlarmForegroundColor** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
StatusId	Integer. The ID corresponding to the alarm status. Valid entries: 0 - OK 1 - LOLO

	2 - HIHI 3 - LO 4 - HI 5 - RATE 6 - COS 7 - CFN 8 - DEV 9 - FLT 10 - DSAB 11 - ERROR 12 - ANY 13 - NEW 14 - TIME 15 - IOF 16 - OCD 17 - UNDER 18 - OVER 19 - RANGE 20 - COMM 21 - DEVICE 22 - STATION 23 - ACCESS 24 - SQL LOGIN 25 - SQL CMD 26 - DAT MATCH 27 - FLD READ 28 - FLD WRITE 29 - NO DATA 30 - NO XDATA

Return Value

OLE_COLOR. The row foreground color configured for the specified alarm status.

GetBoundRect Method

Returns the shape's current bounding rectangle.

Syntax

*object.*GetBoundRect *pfTop, pfLeft, pfBottom, pfRight*

Properties

The **GetBoundRect** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pfTop	Double. Returns the top position of the bounding rectangle.

pfLeft	Double. Returns the left position of the bounding rectangle.
pfBottom	Double. Returns the bottom position of the bounding rectangle.
pfRight	Double. Returns the right position of the bounding rectangle.

Remarks

The bounding rectangle is represented in Logical or Enhanced Coordinates relative to the picture's upper left hand corner.

GetChartEndTime Method

Returns the end time for the enhanced XY chart.

Syntax

*object.***GetChartEndTime** [= *Date*]

Properties

The **GetChartEndTime** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Date	The ending time for the chart. This takes the regional setting format.

GetChartStartTime Method

Returns the start time for the enhanced XY chart.

Syntax

*object.***GetChartStartTime** [= *Date*]

Properties

The **GetChartStartTime** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Date	The starting time for the chart. This takes the regional setting format.

GetColHeadings Method

Returns the names of all column headings in the Alarm Summary object.

Syntax

object.**GetColHeadings** *bstrColHeadings*

Properties

The **GetColHeadings** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrColHeadings	String. Returns a string containing the column headings of the Alarm Summary object.

Remarks

The **GetColHeadings** method returns the Alarm Summary object's column headings in order, separated by a semicolon.

GetColumnInfo Method

Returns a specific column's name and size information.

Syntax

object.**GetColumnInfo** *iColumn, szItemName, iNumChars*

Properties

The **GetColumnInfo** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
iColumn	Integer. Index to the column you want to access.
szItemName	String. Returns the column name.
iNumChars	Integer. Returns the number of characters used to determine the column size.

GetConnectionInformation Method

Fetches a property connection information set.

Syntax

object.**GetConnectionInformation** *iIndex, bstrProperyName, bstrSource, bstrFullyQualifiedSource, vtSourceObjects, [fTolerance], [fDeadband], [fUpdateRate]*

Properties

The **GetConnectionInformation** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
iIndex	Long. The connection index as returned by **IsConnected**.
bstrProperyName	String. Returns the name of property for this connection index.
bstrSource	String. Returns the data source object name.
bstrFullyQualifiedSource	String. Returns the fully qualified data source name.
tSourceObjects	Variant. Returns the array of tokenized expression parameters.
fTolerance	Variant. (Optional) Returns the equivalency determination factor.
fDeadband	Variant. (Optional) Returns the data change deadband.
fUpdateRate	Variant. (Optional) Returns the refresh rate in seconds for this connection.

GetConnectionParameters Method

Returns the UpdateRate, Deadband and Tolerance for the specified property connection.

Syntax

object.**GetConnectionParameters** *bstrPropertyName, [pvaUpdateRate], [pvaDeadband], [pvaTolerance], [pvaConnectionFlags]*

Properties

The **GetConnectionParameters** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrPropertyName	String. The name of the connected property.
pvaUpdateRate	Variant. (Optional) Returns the refresh rate in seconds for this connection.
pvaDeadband	Variant. (Optional) Returns the data change deadband.
pvaTolerance	Variant. (Optional) Returns the equivalency determination factor.
pvaConnectionFlags	Variant. Reserved.

GetContinuousUser Method

Retrieves the user name of the continuous user.

Syntax

object.**GetContinuousUser***(bstrUserName, lcount)*

Properties

The **GetContinuousUser** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrUserName	String. The user name of the continuous user.
lcount	Long. Returns the value of the continuous user counter. Use the counter to determine if the continuous user has been modified since a previous call to the GetContinuousUser method.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

GetCurrentDataSet Method

Retrieves an object reference to the current data set for the Enhanced Chart.

Syntax

*object.***GetCurrentDataSet**

Properties

The **GetCurrentDataSet** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

Object – the Dataset object retrieved.

GetCurrentValue Method

Returns the value, time stamp, and quality information of the current data associated with the pen object.

Syntax

object.**GetCurrentValue** *CurrentValue, dt, lQual*

Properties

The **GetCurrentValue** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
CurrentValue	Single. Returns the current value of the pen's data source.
dt	Date. Returns the associated time stamp of the value.
lQual	Integer. Returns the OPC quality codes representing the quality of the data being sent.

GetCurrentValueWithQuality Method

Returns the current value, time stamp, and quality information of the dataset.

Syntax

object.**GetCurrentValueWithQuality** *CurrentValue, dt, lQual*

Properties

The **GetCurrentValueWithQuality** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
CurrentValue	Double. Returns the current value of the dataset.
dt	Variant. Returns the associated time stamp of the value if the dataset's X axis is time. Returns a Double for any other data type.
lQual	Long. Returns the OPC quality code representing the quality of the data being retrieved.

GetDataSetByPosition Method

Retrieves the Dataset object by position.

Syntax

*object.***GetDataSetByPosition** *nDSPosition*

Properties

The **GetDataSetByPosition** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
nDSPosition	Long. The position of the data set that you want to retrieve from the chart object. Position

	is 0 based.

Return Value

Object – the dispatch pointer of the Dataset object retrieved by position.

GetDeviceRect Method

Returns the shape's current bounding rectangle in device coordinates.

Syntax

*object.***GetDeviceRect** *plTop, plLeft, plBottom, plRight*

Properties

The **GetDeviceRect** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
plTop	Long. Returns the top position of the bounding rectangle.
plLeft	Long. Returns the left position of the bounding rectangle.
plBottom	Long. Returns the bottom position of the bounding rectangle.
plRight	Long. Returns the right position of the bounding rectangle.

GetDuration Method

Returns the duration for the current object.

Syntax

*object.***GetDuration** *days, hours, minutes, seconds*

Properties

The **GetDuration** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
days	Long. Returns the days portion of the duration time.
hours	Long. Returns the hours portion of the duration time.
minutes	Long. Returns the minutes portion of the duration time.
seconds	Long. Returns the seconds portion of the duration time.

GetGlobalDuration Method

Returns the duration for the current object.

Syntax

*object.***GetGlobalDuration** *days, hours, minutes, seconds*

Properties

The **GetGlobalDuration** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
days	Long. Returns the days portion of the duration time.
hours	Long. Returns the hours portion of the duration time.
minutes	Long. Returns the minutes portion of the duration time.
seconds	Long. Returns the seconds portion of the duration time.

GetErrorString Method

Returns the error string that corresponds to the specified error number.

Syntax

*object.***GetErrorString***(errcode)*

Properties

The **GetErrorString** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
errcode	Long. Returns the error code generated from a method call and/or property setting.

Return Value

String. The text string associated with the specified error code.

GetEventHandlerIndex Method

Returns the index of the Event member in the **Procedures** collection.

Syntax

*object.***GetEventHandlerIndex** *bstrEventName, pIndex, pFound*

Properties

The **GetEventHandlerIndex** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrEventName	String. The name of the event.
pIndex	Long. Numerical index of the procedures position in the existing collection. Note: This index is transient, it will change as procedures are added or deleted.
pFound	Long. Returns 1 if an event procedure is present, 0 if no event procedure is present.

GetFullname Method

Returns the full name of a user.

Syntax

object.**GetFullname***(bstrUsername, bstrFullname)*

Properties

The **GetFullname** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrUserName	String. The user name of the user in Windows security or the login name of the user in iFIX Security.
bstrFullname	String. Returns the full name of the user.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

GetGlobalHistoricalUpdateRate Method

Returns the update rate for historical data sources in run mode.

Syntax

*object.***GetGlobalHistoricalUpdateRate** *hrs, mins, secs*

Properties

The **GetGlobalHistoricalUpdateRate** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
hrs	Long. Returns the hours portion of the update rate.
mins	Long. Returns the minutes portion of the update rate.
secs	Long. Returns the seconds portion of the update rate.

GetIndirectionInfo Method

Reserved for internal use.

GetInterval Method

Returns the interval for the current object.

Syntax

*object.***GetInterval** *days, hours, minutes, seconds*

Properties

The **GetInterval** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
days	Long. Returns the days portion of the interval time.
hours	Long. Returns the hours portion of the interval time.
minutes	Long. Returns the minutes portion of the interval time.
seconds	Long. Returns the seconds portion of the interval time.

GetKeyMacro Method

Returns a key macro object defined by key combination, if one exists.

Syntax

*object.***GetKeyMacro** (*ComboKey, KeyCode*)

Properties

The **GetKeyMacro** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ComboKey	enumCombinationKey. The control shift part of the key combination.
KeyCode	Integer. The ASCII value of the main key of the key combination.

GetKeyMacroIndex Method

Returns the index of the key macro specified by the key combination.

Syntax

*object.***GetKeyMacroIndex** (*ComboKey, KeyCode,Index, Found*)

Properties

The **GetKeyMacroIndex** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ComboKey	enumCombinationKey. The control shift part of the key combination.
KeyCode	Integer. The ASCII value of the main key of the key combination.
Index	Long. Index of KeyMacro, (-1, if not found).
Found	Boolean (optional). **True**, if found. **False** if not found.

GetLevel Method

Gets the level properties for the specified level index of the lookup object.

Syntax

*object.***GetLevel** *iIndex, pInput1, pOutput1, [pInput2], [pOutput2]*

Properties

The **GetLevel** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
iIndex	Long. 1-based index into the array of levels.
pInput1	Variant. Returns the first input parameter. Used as the lookup value for exact match look-ups, and the minimum value for range comparison lookups.
pOutput1	Variant. Returns the primary output value at this level.

pInput2	Variant. (Optional) Returns the second input parameter. Used as the maximum value for range comparison lookups.
pOutput2	Variant. (Optional) Returns the secondary output value for this level. Used by the lookup object as the “blink to” value at this level.

GetNumberOfDataSets Method

Retrieves the number of data sets in an Enhanced Chart.

Syntax

*object.***GetNumberOfDataSets ()**

Properties

The **GetNumberOfDataSets** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

Long – the number of data sets in the chart.

GetObjectInfo Method

Returns a two-dimensional array containing the property values for the specified objects.

Syntax

*object.***GetObjectInfo***(bstrObjectNames, bstrProperties)*

Properties

The **GetObjectInfo** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrObjectNames	String Array. An array of object names the user wants values for.
bstrProperties	String Array. An array of property names the user wants values for.

Return Value

Variant Array. A two-dimensional array consisting of the property values corresponding to the specified property names for the specified objects.

Remarks

GetObjectInfo will return a value of EMPTY if the user has specified an invalid object and/or property name.

GetPenDataArray Method

Fetches the data array for the specified Pen

Syntax

*object.***GetPenDataArray***pNumPts, pVal, ppsa, pQual*

Properties

The **GetPenDataArray** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pNumPts	Long. Number of array points.
pVal	Array of variants. An array of values for each point in the pen.
ppsa	Array of variants. An array of times for each point in the pen.
pQual	Array of variants. An array of qualities for each point in the pen.

Remarks

The **Pen** data array is a set of three arrays containing the value, time stamp, and quality information for each point. The array is ordered as drawn left to right, with the most recent data elements at the end of the array.

GetPenDataArrayEx Method

Fetches the data array for the specified Pen

Syntax

*object.***GetPenDataArrayEx***pNumPts, pVal, ppsa, pQual, pMilliseconds*

Properties

The **GetPenDataArrayEx** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pNumPts	Long. Number of array points.
pVal	Array of variants. An array of values for each point in the pen.
ppsa	Array of variants. An array of times for each point in the pen.

pQual	Array of variants. An array of qualities for each point in the pen.
PMilliseconds	Array of variants. An array containing the millisecond component of the timestamp for each point in the pen.

Remarks

The **Pen** data array is a set of four arrays containing the value, time stamp (excluding milliseconds), quality, and millisecond component of the timestamp information for each point. The array is ordered as drawn left to right, with the most recent data elements at the end of the array.

GetPointAt Method

Returns the point at the given index.

Syntax

*object.***GetPointAt***(lIndex)*

Properties

The **GetPointAt** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
lIndex	Long. The index of the point you want returned.

Return Value

Object. An object of class **FixFloatPoint**, which contains the x and y coordinates.

Remarks

This function allows users to iterate through the data point array. The maximum index is determined by the number of data points in the object.

GetPriorityColor Method

Returns the row background color configured for the specified alarm priority.

NOTE: This method has been deprecated and replaced with the ."GetAlarmBackgroundColor Method " on page 409

Syntax

*object.***GetPriorityColor***(PriorityId)*

Properties

The **GetPriorityColor** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
PriorityId	Integer. The ID corresponding to the alarm priority. **Valid entries for iFIX 3.5 and earlier:** 0 - High 1 - Medium 2 - Low **Valid entries for iFIX 4.0 and later:** 3 - CRITICAL 4 - HIHI 5 - HIGH 6 - MEDIUM 7 - LOW 8 - LOLO 9 - INFO (INFORMATIONAL)

Return Value

OLE_COLOR. The row background color configured for the specified alarm priority.

GetProcedureIndex Method

Returns the index of the Procedure member in the Procedures collection.

Syntax

*object.***GetProcedureIndex** *bstrProcName, pIndex, pFound*

Properties

The **GetProcedureIndex** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrProcName	String. The name of the procedure.
pIndex	Long. Numerical index of the procedures position in the existing collection. **Note:** This index is transient, it will change as procedures are added or deleted.
pFound	Long. Returns 1 if a procedure is present, 0 if no procedure is present.

GetProperty Method

Returns the value of the specified property name.

Syntax

*object.***GetProperty** *bstrPropertyName, vaValue*

Properties

The **GetProperty** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrPropertyName	String. The name of the property.
aValue	Variant. Returns the value of the property.

Remarks

The **GetProperty** and**SetProperty** methods are useful for creating table driven property operations. Generic import and export functions can be written given a list of property names to access (also see **ListProperties**).

GetPropertyAttributes Method

Fetches a list of property attributes. For a tag reference, these are properties such as new alarm status and property range information (EGU limits, list of strings alarm strings (HIHI, LOLO, etc)). Each property queried may have a different set of attributes.

Syntax

object.**GetPropertyAttributes** *bstrFullyQualifiedName, spAttribute, vtResults, vtAttributeNames, iStatus*

Properties

The **GetPropertyAttributes** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrFullyQualifiedName	String. The fully specified object reference. For example, (FIX32.SCADA1.AI1.F_CV).
spAttribute	Integer Enumeration. The attribute of the property that you are interested in. Valid entries: 0 - CurrentValue 1 - Description 2 - Range 3 - LowEGU 4 - HighEGU 5 - CurrentAlarm 6 - AlarmAcknowledge 7 - AlarmLatched
tResults	Variant array. Returns property attribute information. All attribute information except for the range request performs a database read to fetch the information from the process database. For the range attribute, the results vary depending

	on the qualified name passed. If the qualified name ends in A_LAALM or A_CUALM then the results property would return a variant array of strings with the possible alarm strings ("HI", "HIHI", "LO", "LOLO", etc.). For the all other field names (F_CV, A_DESC, etc.) a range request will return the low and high EGU limits of the block that the field specifies.
tAttributesNames	Variant array of strings. This is a return value that is used to help map one fully qualified name to another. For example, if Fix32.SCADA1.AI1.F_CV is entered, and the script needs to also fetch the current alarm status for this tag, calling **GetPropertyAttributes** with an attribute type of CurrentAlarm returns the fully qualified name Fix32.SCADA1.AI1.A_CUALM. This can then be fetched in conjunction with the first item parsing the string, since different OPC servers may map alarm status into different fields.
iStatus	Long. Returns the error status value. Return values are: 0 – OK 1 – Syntax error 2 – Data Undefined 3 – Data type mismatch

GetPropertyTargets Method

Returns a list of object names that have built connections (subscribed) to a property. The targets are the objects that are sent data when the property's value changes.

Syntax

object.**GetPropertyTargets** *iIndex, bstrPropertyName, vtTargets*

Properties

The **GetPropertyTargets** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
iIndex	Long. The index of the connected property list.
bstrPropertyName	String. Returns the property name for the specified connection index.
tTargets	Variant. Returns an array of fully qualified names containing connections to this property.

Remarks

This method can be used to see all objects that are connected to a property. For example, to determine which objects are connected to a specific tag reference, call this method on the tag reference.

The returned targets variant is an array of fully qualified names of objects that have connections to this tag. This list is only valid for those connections currently in memory. (Only for currently loaded objects; not valid for pictures on disk).

GetRibbonView Method

Returns if the iFIX WorkSpace ribbon is enabled.

Syntax

*object.***GetRibbonView**

Properties

The **GetRibbonView** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

Boolean. True indicates that the iFIX WorkSpace ribbon is enabled. False indicates it is not.

GetSelectedAlmExt Method

Returns the alarm extension fields (A_ALMEXT1, A_ALMEXT2) for the tag in the currently selected alarm.

Syntax

object.**GetSelectedAlmExt**(*AlmExt1, AlmExt2*)

Properties

The **GetSelectedAlmExt** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
AlmExt1	String. Returns the string configured in the alarm's first extension field.
AlmExt2	String. Returns the string configured in the alarm's second extension field.

Return Value

Boolean. **True** if an alarm is selected, **False** if no alarm is selected.

Remarks

If no alarm is selected, *AlmExt1* and *AlmExt2* are set to EMPTY.

GetSelectedNodeTag Method

Returns the node and tag name corresponding to the currently selected alarm.

Syntax

object.**GetSelectedNodeTag**(*sNode, sTag*)

Properties

The **GetSelectedNodeTag** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
sNode	String. Returns the node name corresponding to the currently selected alarm.
sTag	String. Returns the tag name corresponding to the currently selected alarm.

Return Value

Boolean. **True** if an alarm is selected, **False** if no alarm is selected.

Remarks

If no alarm is selected, *sNode* and *sTag* are set to EMPTY.

GetSelectedRow Method

Returns the information for the selected alarm in the **Alarm Summary** object. Note that even if a column is not displayed, all information is still returned.

Syntax

object.**GetSelectedRow**(*bAcknowledged, Handle, Area, DateIn, DateLast, TimeIn, TimeLast, Node, TagName, Priority, Status, Description, Value, ExtField1, ExtField2, UserDefField1, UserDefField2*)

Properties

The **GetSelectedRow** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bAcknowledged	Boolean. Returns **True** if the alarm has been acknowledged, **False** otherwise.
Handle	String. Returns the unique identifier for the alarm. Specifically <type><ipn><node-name>.
Area	String. Returns the alarm area(s) for the alarm.
DateIn	String. Returns the date when the block first generated the alarm.
DateLast	String. Returns the date when the block last generated the alarm.
TimeIn	String. Returns the time when the block first generated the alarm.
TimeLast	String. Returns the time when the block last generated the alarm.

Node	String. Returns the name of the Scada server that generated the alarm.
TagName	String. Returns the tagname for the alarm.
Priority	String. Returns the priority for the alarm.
Status	String. Returns the latched alarm for the block that is in alarm.
Description	String. Returns the description field for the block in alarm.
Value	String. Returns the current value of the block in alarm.
ExtField1	String. Returns the first extension field for the block in alarm.
ExtField2	String. Returns the second extension field for the block in alarm.
UserDefField1	String. Returns the current value of the first "A_" field configured for this column.
UserDefField2	String. Returns the current value of the second "A_" field configured for this column.

Return Value

Boolean. **True** if an alarm is selected, **False** if no alarm is selected.

Remarks

If no alarm is selected, all parameters are set to EMPTY.

GetSelectedRowAlarmInfo Method

Returns the information for the selected alarm in the **Alarm Summary** object. Note that even if a column is not displayed, all information is still returned. This function is similar to GetSelectedRow except it includes four additional parameters at the end of its request: Shelvable, ShelveRemDuration, and two Reserved parameters.

Syntax

object.**GetSelectedRowAlarmInfo** *bAcknowledged, Handle, Area, DateIn, DateLast, TimeIn, TimeLast, Node, TagName, Priority, Status, Description, Value, ExtField1, ExtField2, UserDefField1, UserDefField2, Shelvable, ShelveRemDuration, Reserved1, Reserved2*

Properties

The **GetSelectedRowAlarmInfo** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bAcknowledged	Boolean. Returns **True** if the alarm has been acknowledged, **False** otherwise.
Handle	String. Returns the unique identifier for the alarm. Specifically <type><ipn><node-name>.
Area	String. Returns the alarm area(s) for the alarm.
DateIn	String. Returns the date when the block first generated the alarm.
DateLast	String. Returns the date when the block last generated the alarm.
TimeIn	String. Returns the time when the block first generated the alarm.
TimeLast	String. Returns the time when the block last generated the alarm.

Node	String. Returns the name of the Scada server that generated the alarm.
TagName	String. Returns the tagname for the alarm.
Priority	String. Returns the priority for the alarm.
Status	String. Returns the latched alarm for the block that is in alarm.
Description	String. Returns the description field for the block in alarm.
Value	String. Returns the current value of the block in alarm.
ExtField1	String. Returns the first extension field for the block in alarm.
ExtField2	String. Returns the second extension field for the block in alarm.
UserDefField1	String. Returns the current value of the first "A_" field configured for this column.
UserDefField2	String. Returns the current value of the second "A_" field configured for this column.
Shelvable	String.
ShelveRemTime	String.
Reserved1	Not available for use. Reserved.
Reserved2	Not available for use. Reserved.

Return Value

Boolean. **True** if an alarm is selected, **False** if no alarm is selected.

Remarks

If no alarm is selected, all parameters are set to EMPTY.

GetSelectedRowsAlarmInfo Method

Reads the information for the selected rows in the **Alarm Summary** object. Note that even if a column is not displayed, all information is still returned. This function is similar to GetSelectedRow except it includes these additional parameters in its request: NumRows, FlexData1-8, Shelvable, ShelveRemDuration, and two Reserved parameters.

Syntax

object.**GetSelectedRowsAlarmInfo***NumRows, bAcknowledged, Handle, Area, DateIn, DateLast, TimeIn, TimeLast, Node, TagName, Priority, Status, Description, Value, ExtField1, ExtField2, UserDefField1, UserDefField2, FlexData1, FlexData2, FlexData3, FlexData4, FlexData5, FlexData6, FlexData7, FlexData8, Shelvable, ShelveRemDuration, Reserved1, Reserved2*

Properties

The **GetSelectedRowsAlarmInfo** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
NumRows	Long. Returns the number of rows selected.
bAcknowledged	Boolean. Returns **True** if the alarm has been acknowledged, **False** otherwise.
Handle	String. Returns the unique identifier for the alarm. Specifically <type><ipn><nodename>.

Area	String. Returns the alarm area(s) for the alarm.
DateIn	String. Returns the date when the block first generated the alarm.
DateLast	String. Returns the date when the block last generated the alarm.
TimeIn	String. Returns the time when the block first generated the alarm.
TimeLast	String. Returns the time when the block last generated the alarm.
Node	String. Returns the name of the Scada server that generated the alarm.
TagName	String. Returns the tagname for the alarm.
Priority	String. Returns the priority for the alarm.
Status	String. Returns the latched alarm for the block that is in alarm.
Description	String. Returns the description field for the block in alarm.
Value	String. Returns the current value of the block in alarm.
ExtField1	String. Returns the first extension field for the block in alarm.
ExtField2	String. Returns the second extension field for the block in alarm.
UserDefField1	String. Returns the current value of the first "A_" field configured for this column.
UserDefField2	String. Returns the current value of the second "A_" field configured for this column.
FlexData1	Variant. Return the information from the rows as a Variant array.
FlexData2	Variant. Return the information from the rows as a Variant array.
FlexData3	Variant. Return the information from the rows as a Variant array.
FlexData4	Variant. Return the information from the rows as a Variant array.
FlexData5	Variant. Return the information from the rows as a Variant array.
FlexData6	Variant. Return the information from the rows as a Variant array.
FlexData7	Variant. Return the information from the rows as a Variant array.
FlexData8	Variant. Return the information from the rows as a Variant array.
Shelvable	String.
ShelveRemDuration	String.
Reserved1	Not available for use. Reserved.
Reserved2	Not available for use. Reserved.

Return Value

Boolean. **True** if an alarm is selected, **False** if no alarm is selected.

Remarks

If no alarm is selected, all parameters are set to EMPTY.

GetSelectedUserDefFields Method

Returns the current values in the user defined columns configured for the **Alarm Summary** object.

Syntax

object.**GetSelectedUserDefFields**(*UserDefField1, UserDefField2*)

Properties

The **GetSelectedUserDefFields** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
UserDefField1	String. Returns the string configured in the alarm's first extension field.
UserDefField2	String. Returns the string configured in the alarm's second extension field.

Return Value

Boolean. **True** if an alarm is selected, **False** if no alarm is selected.

Remarks

If no alarm is selected and/or there are no user defined columns, *UserDefField1* and *UserDefField2* are set to EMPTY.

GetSignature Method

Displays the Electronic Signature dialog box based on the values passed in. The dialog box validates the signing, but does not send a write request to the database or a signed operator message.

If verification is not required, only the Performed By section of the dialog box displays. If verification is required, both the Performed By and Verified By sections display. If the tag is configured to allow continuous use, the continuous user name appears in the Performed By user name edit box. Default comments display based on the threshold table names passed in. If no threshold table names are passed in, the threshold table names set in the WorkSpace User Preferences, if any, are used to display the comment pick lists in the dialog box.

Syntax

object.**GetSignature***(bstrDescription, bVerify, bAllowContinuousUse, pbValidSig, bstrPerformUser, bstrPerformUserID, bstrPerformComment, [bstrVerifyUser], [bstrVerifyUserID], [bstrVerifyComment], [bCheckTag], [applicationId], [bstrPerformCommentTb]l, [bstrVerifyCommentTbl], [pSecAreas], [bstrDialogCaption], [bPerformCommentRequired])*

Properties

The **GetSignature** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrDescription	String. Displays a message in the Electronic Signature dialog box that describes the action needing a signature.
bVerify	Boolean. Indicates whether or not verification is required.
bAllowContinuousUse	Boolean. Indicates whether or not the Performed By user name should default to the name of the continuous user.
bValidSig	Boolean. Returns **True** if a valid signature was captured, or **False** if not.

bstrPerformUser	String. Returns the user name of the Performed By user.
bstrPerformUserID	String. Returns the user ID (short name) of the Performed By user. This ID is used when sending a signed operator message.
bstrPerformComment	String. Returns the comment entered by the Performed By user.
bstrVerifyUser	String. (Optional). Returns the user name of the Verified By signature.
bstrVerifyUserID	String. (Optional). Returns the user ID (short name) of the Verified By user. This ID is used when sending a signed operator message.
bstrVerifyComment	String. (Optional). Returns the comment entered by the Verified By user.
bCheckTag	Boolean. (Optional). Indicates whether the user access to security areas assigned to the tag should be checked. The default value is **False**. If this is set to **True**, you must call the Initialize method prior to calling this method. If you do not call Initialize() or the tag is not a FIX32 data source, an error is returned when the signature is entered.
applicationId	Integer. (Optional). Contains additional application feature to be checked. Normally, the system checks only the Performed By and Verified By application features during signature validation. For a list of application IDs, refer to the FIXCheckApplicationAccess method. To skip this check, set the value to -1.
bstrPerformCommentTbl	String. (Optional). Contains the name of the comment threshold table to display in the Performed By comment pick list.
bstrVerifyCommentTbl	String. (Optional). Contains the name of the comment threshold table to display in the Verified By comment pick list.
pSecAreas	Variant Array. (Optional). Contains a list of security areas to check when validating the signature.
bstrDialogCaption	String. (Optional). Contains the caption that will display in the Electronic Signature dialog box title bar in place of the default caption "Electronic Signature."
bPerformCommentRequired	Boolean. (Optional). When TRUE, this parameter instructs the Electronic Signature dialog box to require that a perform comment be entered before the electronic signature can be accepted. This parameter is optional and defaults to FALSE if it is not included.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

GetSignatureAndWriteValue Method

Displays the Electronic Signature dialog box based on the values of the properties set in the ESignature object by a prior call to the Initialize(), InitalizeList(), IsSignatureRequired(), or IsSignatureRequiredForList(). Typically, you use this method when performing an electronic signature for FIX32 data sources. The dialog box validates the signing, sends a write request to the database, and then sends a signed operator message to the alarm system.

If verification is not required, only the Performed By section of the Electronic Signature dialog box displays. If verification is required, both the Performed By and Verified By sections display. If the tag is configured to allow continuous use, the continuous user name appears in the Performed By user name edit box. Predefined comments display based on the threshold table names passed in. If no threshold table names are passed in, the threshold table names set in the WorkSpace User Preferences, if any, are used to display the comment pick lists in the dialog box.

Syntax

object.**GetSignatureAndWriteValue***(nAction, pValue, [bReadLabels], [bstrZeroLabel], [bstrNonZeroLabel], [bstrDescription], [bstrPerformCommentTbl], [bstrVerifyCommentTbl], [pbValidSig], [bstrDialogCaption])*

Properties

The **GetSignatureAndWriteValue** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
nAction	Integer. Indicates the type of action to be performed. Valid entries for nAction include: 0 – (WRITE_VAL). After validating the signature(s), the value passed in *pValue* is written to the database and a signed operator message is sent to the alarm system logging the action. 3 – (ACK_OR_REMOVE). After validating the signature(s), the alarm is acknowledged and a signed operator message is sent to the alarm system logging the action. 4 – (ACK_OR_REMOVE_LIST). After validating the signature(s), the alarms in the list are acknowledged and a signed operator message is sent to the alarm system for each alarm that was successfully acknowledged. For a list requiring both perform and verify signatures, if a perform signature fails the security area check for any alarm in the list, the signature fails for the whole list of alarms and no alarms are acknowledged. An error is generated.
pValue	Variant. Contains the value to be written to the database.
bReadLabels	Boolean. (Optional). Indicates whether the zero and nonzero description labels should be read from the database. These labels format the message string that displays in the Electronic Signature dialog box and describe the action that is being signed for. Defaults to **True**.
bstrZeroLabel	String. (Optional). The zero description label to be used when formatting the message string. The message string displays in the Electronic Signature dialog box and describes the action that is being signed for.
bstrNonZeroLabel	String. (Optional). The non-zero description label for the message string. The message string displays in the Electronic Signature dialog box and describes the action that you sign for.
bstrDescription	String. (Optional). Message string that displays in the Electronic Signature dialog box and describes the action that you sign for. This is appended to any description that created by the object based on the nAction parameter.
bstrPerformCommentTbl	String. (Optional). Name of the Performed By comment threshold table to use to display default comments.

bstrVerifyCommentTbl	String. (Optional). Name of the Verified By comment threshold table to use to display default comments.
pbValidSig	Boolean. (Optional). Returns **True** if a valid signature was captured, **False** if not.
bstrDialogCaption	String. (Optional). Contains the caption that will display in the Electronic Signature dialog box title bar in place of the default caption “Electronic Signature.”

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

GetStatusColor Method

Returns the row foreground color configured for alarms with the specified status.

NOTE: This method has been deprecated and replaced with the "GetAlarmForegroundColor Method " on page 410

Syntax

*object.***GetStatusColor***(StatusId)*

Properties

The **GetStatusColor** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
StatusId	Integer. The ID corresponding to the alarm status. Valid entries: 0 - OK 1 - LOLO 2 - HIHI 3 - LO 4 - HI 5 - RATE 6 - COS 7 - CFN 8 - DEV 9 - FLT 10 - DSAB 11 - ERROR 12 - ANY 13 - NEW 14 - TIME 15 - IOF

	16 - OCD 17 - UNDER 18 - OVER 19 - RANGE 20 - COMM 21 - DEVICE 22 - STATION 23 - ACCESS 24 - SQL LOGIN 25 - SQL CMD 26 - DAT MATCH 27 - FLD READ 28 - FLD WRITE 29 - NO DATA 30 - NO XDATA

Return Value

OLE_COLOR. The row foreground color configured for the specified alarm status.

GetStatusFont Method

Returns the row font for alarms that have the specified status.

Syntax

object.**GetStatusFont**(*nStatusID, lpbStrikeout, lpbUnderline, lpbBold, lpbItalic, lpnSize*)

Properties

The **GetStatusFont** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
nStatusID	Integer. The ID corresponding to the alarm status. Valid entries: 0 - OK 1 - LOLO 2 - HIHI 3 - LO 4 - HI 5 - RATE 6 - COS 7 - CFN 8 - DEV 9 - FLT 10 - DSAB 11 - ERROR

	12 - ANY 13 - NEW 14 - TIME 15 - IOF 16 - OCD 17 - UNDER 18 - OVER 19 - RANGE 20 - COMM 21 - DEVICE 22 - STATION 23 - ACCESS 24 - SQL LOGIN 25 - SQL CMD 26 - DAT MATCH 27 - FLD READ 28 - FLD WRITE 29 - NO DATA 30 - NO XDATA
lpbStrikeout	Boolean. Returns whether the text appears with a strikeout through it.
lpbUnderline	Boolean. Returns whether the text is underlined.
lpbBold	Boolean. Returns whether the text is bold or not.
lpbItalic	Boolean. Returns whether the text is italic or not.
lpnSize	Integer. Returns the font size of the text.

Return Value

String. The font name of the text for the specified status.

GetTimeBeforeNow Method

Returns the initial relative start time for the current object.

Syntax

*object.***GetTimeBeforeNow** *hours, minutes, seconds*

Properties

The **GetTimeBeforeNow** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
hours	Long. Returns the hours portion of the start time.
minutes	Long. Returns the minutes portion of the start time.
seconds	Long. Returns the seconds portion of the start time.

GetTimeCursorInfo Method

Returns a set of **Pen** information (time, value, quality) of the trend at the time where the **Pen** crosses the time cursor.

Syntax

*object.***GetTimeCursorInfo** *lPenNum, pDt, pfVal, pQuality*

Properties

The **GetTimeCursorInfo** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
lPenNum	Integer. The index for the pen for which you want the time cursor information for.
pDt	Date. Returns the date represented when the pen crosses the time cursor.
pfVal	Double. Returns the value represented when the pen crosses the time cursor.
pQuality	Long. Returns the quality of the data represented when the pen crosses the time cursor.

GetUserID Method

Returns the user ID (short name) from iFIX security for the given user name and password. This ID is used when sending a signed operator message.

Syntax

object.**GetUserID***(bstrUserName, bstrPassword, bstrUserID)*

Properties

The **GetUserID** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrUserName	String. The name of the event.
bstrPassword	String. The code to be written to the event in the form of strings.
bstrUserID	String. Returns the user ID (short name) of the user in iFIX security.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

GetWindowLocation Method

Retrieves the specified window's size and location in terms of percentage of the screen.

Syntax

object.**GetWindowLocation** *plfTopPct, plfLeftPct, plfHeight, plfWidth*

Properties

The **GetWindowLocation** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
plfTopPct	Double. Returns the top percentage of the window relative to the WorkSpace client area.
plfLeftPct	Double. Returns the left percentage of the window relative to the WorkSpace client area.
plfHeightPct	Double. Returns the percentage of horizontal screen space.
plfWidthPct	Double. Returns the percentage of vertical screen space.

GlobalScrollBackFast Method

Scrolls the Global Time Control time frame forward by the factor specified in the object's GlobalFastScrollRate property. This method applies to historical data sources in run mode. When this method is executed on the Global Time Control in run mode, the time frame specified in the Global Time Control will move forward by the specified slow scroll rate.

NOTE: Tying a script to this method that triggers faster than 5 seconds is not recommended.

Syntax

*object.***GlobalScrollBackFast**

Properties

The **GlobalScrollBackFast** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

GlobalScrollBackSlow Method

Scrolls the Global Time Control time frame forward by the factor specified in the object's GlobalSlowScrollRate property. This method applies to historical data sources in run mode. When this method is executed on the Global Time Control in run mode, the time frame specified in the Global Time Control will move forward by the specified slow scroll rate.

NOTE: Tying a script to this method that triggers faster than 5 seconds is not recommended.

Syntax

*object.***GlobalScrollBackSlow**

Properties

The **GlobalScrollBackSlow** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

GlobalScrollForwardFast Method

Scrolls the Global Time Control time frame forward by the factor specified in the object's GlobalFastScrollRate property. This method applies to historical data sources in run mode. When this method is executed on the Global Time Control in run mode, the time frame specified in the Global Time Control will move forward by the specified slow scroll rate.

NOTE: Tying a script to this method that triggers faster than 5 seconds is not recommended.

Syntax

*object.***GlobalScrollForwardFast**

Properties

The **GlobalScrollForwardFast** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

GlobalScrollForwardSlow Method

Scrolls the Global Time Control time frame forward by the factor specified in the object's GlobalSlowScrollRate property. This method applies to historical data sources in run mode. When this method is executed on the Global Time Control in run mode, the time frame specified in the Global Time Control will move forward by the specified slow scroll rate.

NOTE: Tying a script to this method that triggers faster than 5 seconds is not recommended.

Syntax

*object.***GlobalScrollForwardSlow**

Properties

The **GlobalScrollForwardSlow** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

GlobalTimerApply Method

Applies the configured global time control settings to historical data sources in all open pictures in run mode.

Syntax

*object.***GlobalTimerApply**

Properties

The **GlobalTimerApply** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Group Method

Forms a Group consisting of the currently selected objects. This is equivalent to selecting Group from the Format menu.

Syntax

object.**Group**

Properties

The **Group** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Group is a Configuration environment method only.

The **Group** method assumes that the objects selected are at the top most level. That is, grouping objects that are selected in drill down mode groups the top-most group containing the selected object.

HiLoDisplay Method

Sets the HiDisplay and LoDisplay properties for the Time Axis of a Chart.

Syntax

object.**HiLoDisplay** *HiDisplay, LoDisplay*

Properties

The **HiLoDisplay** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
HiDisplay	Date. The date to set the HiDisplay property.
LoDisplay	Date. The date to set the LoDisplay property.

I-L

ImportToolbar Method

Imports an iFIX toolbar.

Syntax

object.**ImportToolbar***bstrToolbarName, bstrToolbarOwner*

Properties

The **ImportToolbar** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrToolbarName	String. Name of the toolbar file to be imported.
BstrToolbarOwner	String. Name of the newly imported toolbar's owner (*WorkSpace*, *Picture*, or *Scheduler*)

Initialize Method

Sets the object's data variables based on the value of the bstrDataSource parameters. If the method encounters an error (such as bad syntax), a corresponding error code is stored in a member variable of the object for later checking.

If the data source is FIX32 it reads electronic signature fields from the tag, and sets the associated variables in the object with the data that was read from the tag. If an error occurs reading the settings from the tag, a corresponding error code is stored in a member variable of the object for later checking.

Syntax

object.**Initialize***(bstrDataSource)*

Properties

The **Initialize** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrDataSource	String. The fully qualified name of the data source. For example, FIX32.NODE1.-do1.F_CV or, section.node.tag.field format for FIX32 data sources.

Return Value

This method returns an HRESULT, indicating the success or failure of the method call: S_OK for a success or an error code for a failure.

InitializeList Method

Evaluates the data source names in the pDataSourceList parameters and determines if they represent FIX32 data. If all data sources are FIX32, it reads the signature settings from the database for each data source in the list and sets corresponding variables in the object.

If an error occurs reading the settings from the database, a corresponding error code is stored in a member variable of the object for later checking.

Syntax

object.**InitializeList***(pDataSourceList)*

Properties

The **InitializeList** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pDataSourceList	Variant array. The fully qualified names of the data sources.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

InsertPoint Method

Inserts a new point at the given index.

Syntax

*object.***InsertPoint***/Index, pdispPoint*

Properties

The **InsertPoint** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
lIndex	Long. The position at which to insert a point.
pdispPoint	Object. **FixFloatPoint** object containing x,y values.

InteractiveExport Method

Launches the Export dialog box for the Enhanced Chart, even if the chart is non-modifiable at run time.

Syntax

*object.***InteractiveExport**

Properties

The **InteractiveExport** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

IsColorSelectionVisible Method

Determines if the WorkSpace's Color Selection dialog box is open.

Syntax

*object.***IsColorSelectionVisible**

Properties

The **IsColorSelectionVisible** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

Boolean. **True** if the Color Selection dialog box is open, **False** if it is not.

Remarks

IsColorSelectionVisible is a Configuration environment method only.

IsConnected Method

Determines if the specified property has an assigned animation connection.

Syntax

object.**IsConnected***bstrPropertyName, bHasConnection, iIndex, iStatus*

Properties

The **IsConnected** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrPropertyName	String. The name of property.
bHasConnection	Boolean. Returns **True** if the property has a connection, otherwise **False**.
iIndex	Long. Returns the index for the property connection (useful for **GetConnectionInformation**).
iStatus	Long. Returns the error value. Return values are: 0 – OK 1 – Syntax error 2 – Data Undefined 3 – Data type mismatch 4 – Invalid use of property for this type of object 5 - Unexpected Expression Returned

IsEmpty Method

Determines if the **Lookup** object contains any levels.

Syntax

object.**IsEmpty** *pbRefVal*

Properties

The **IsEmpty** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrEventName	Boolean. Returns **True** if the **Lookup** object doesn't contain any levels, **False** if it does.

IsKeyMacroDefined Method

Returns whether or not a key macro defined by the key combination exits.

Syntax

object.**IsKeyMacroDefined** (*ComboKey, KeyCode*)

Properties

The **IsKeyMacroDefined** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ComboKey	enumCombinationKey. The control shift part of the key combination.
KeyCode	Integer. The ASCII value of the main key of the key combination.

IsNodeSignEnabled Method

Checks to see if the node is enabled for electronic signature by determining if the Electronic Signature option is enabled on the node's hardware key, if iFIX security is enabled on the node, and whether bypass signature is in effect. Bypass signature applies when the currently logged in iFIX user has the Electronic Signature-Bypass application feature assigned.

Syntax

object.**IsNodeSignEnabled***(pbEnabled)*

Properties

The **IsNodeSignEnabled** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pbEnabled	Boolean. Returns **True** if signature is enabled on the local node, **False** if it is not.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

IsSignatureRequired Method

Evaluates the variables set by the **Initialize** method and the value of the nAction parameter to determine whether the signature is required for the data source.

Syntax

object.**IsSignatureRequired***(nAction, bSignaturedRequired, [pInfo], [bVerify], [bAllowContinuousUse])*

Properties

The **IsSignatureRequired** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
nAction	Integer. Indicates the type of action associated with this signature. Valid entries for nAction include: 0 – (WRITE_VAL). Writes a value to a single data source. 1 – (ACK). Acknowledges a single alarm. 3 – (ACK_OR_REMOVE). Acknowledges or manually deletes a single alarm.
bSignatureRequired	Boolean. Returns **True** if signature is required for the data source, **False** if it is not.
pInfo	Integer. (Optional). Returns information about why signature is not required for the data source. The return values include: 2– (NO_SIGN). Data source does not require signature. 3– (NO_ACK). Signature is required for writes but not for alarm acknowledgement. 4– (NON_FIX). Data source is not FIX32. 5– (NO_KEY). Electronic Signature option is not enabled on the hardware key on the local or SCADA node. 6– (SEC_NOT_ENAB). Security is not enabled on the local node. 7– (BAD_SYNTAX). Syntax of data source name is bad. 8– (READ_FAIL). Error reading settings from the tag. **NOTE**: You must reference the Electronic Signature type library in VBA to use these enumerations.
bVerify	Boolean. (Optional). Returns **True** if verification is required for the tag, and **False** if it is not.
bAllowContinuousUse	Boolean. (Optional). Returns **True** if continuous use is allowed for the tag, and **False** it is not.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

IsSignatureRequiredForList Method

Evaluates the variables set by the InitializeList() method and the value of the nAction parameter to determine whether a signature is required for the list of data sources.

Syntax

object.**IsSignatureRequiredForList***(nAction, bSignatureRequired, [pInfo], [bVerify], [bAllowContinuousUse])*

Properties

The **IsSignatureRequiredForList** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
nAction	Action Type. Indicates the type of action associated with this signature. Valid entries for nAction include: **ACK_LIST** – Acknowledges multiple alarms. **ACK_OR_REMOVE_LIST** – Acknowledges or manually deletes multiple alarms.
bSignatureRequired	Boolean. Returns **True** if signature is required for the list of data sources, and **False** if it is not. If one tag requires signature all are considered to require signature.
pInfo	Integer. (Optional). Returns information about why signature is not required for the data sources. The return values include: 2 – (NO_SIGN). Data source does not require signature. 3 – (NO_ACK). Signature is required for writes but not for alarm acknowledgement. 4 – (NON_FIX). Data source is not FIX32. 5 – (NO_KEY). Electronic Signature option is not enabled on the hardware key on the local or SCADA node. 6 – (SEC_NOT_ENAB). Security is not enabled on the local node. 7 – (BAD_SYNTAX). Syntax of data source name is bad. 8 – (READ_FAIL). Error reading settings from the tag. **NOTE**: You must reference the Electronic Signature type library in VBA to use these enumerations.
bVerify	Boolean. (Optional). Returns **True** if verification is required for the data source list, **False** if it is not. If one tag requires verification, all require verification.
bAllowContinuousUse	Boolean. (Optional). Returns **True** if continuous use is allowed for the data source list, **False** if it is not. If one tag disallows continuous use, all are considered to disallow continuous use.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

Item Method

Returns the indicated event member in the **Procedures** collection, **Lines** collection. and/or **Sources** collection.

Syntax

object.**Item** (*lIndex*)

Properties

The **Item** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
lIndex	Long. An expression that specifies the position of a member of the **Procedures** collection. The index must be a number from 1 to the value of the Collection's **Count** property.

Return Value

Object. The dispatch pointer to the item object in the collection.

Remarks

If the specified member does not exist, **Item** returns EMPTY.

ListEvents Method

Return a complete list of events for the specified object.

Syntax

object.**ListEvents** *pvEvents, piNumEvents*

Properties

The **ListEvents** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pvEvents	Variant table of strings. Returns a list of event names that have been configured in VBA.
piNumEvents	Integer. Returns the number of configured events.

ListMethods Method

Return a complete list of methods for the specified object.

Syntax

object.**ListMethods** *pvMethods, piNumMethods*

Properties

The **ListMethods** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pvMethods	Variant table of strings. Returns a list of method names that have been configured in VBA.
piNumMethods	Integer. Returns the number of methods returned in the Methods array.

ListProperties Method

Return a complete list of properties and their associated data types for the specified object.

Syntax

object.**ListProperties** *pvProperties, pvDataTypes, piNumProperties*

Properties

The **ListProperties** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pvProperties	Variant array. Returns an array of property names for this object.
pvDataTypes	Variant array. Returns an array of associated property types for the properties.
piNumProperties	Integer. Returns the number of properties in the properties array.

Remarks

ListProperties can be used to get a list of the object's properties, which you can use to call the GetProperty method to extract the contents of the object. Generic import and export functions can be built with these functions.

ListWindowsGroupNames Method

Returns a string array of Windows group names that map to iFIX security privileges.

Syntax

object.**ListWindowsGroupNames** *bNT4NamesOnly*

Properties

The **ListWindowsGroupNames** method syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
bNT4NamesOnly	Boolean. When **False**, returns an array of strings that contains all valid Windows group names. When **True**, returns only Windows group names that do not exceed twenty characters. This satisfies the group name requirements of Windows NT 4.0.

Remarks

ListWindowsGroupNames reads the current iFIX security configuration to generate these names. This method is used primarily by the CreateWindowsGroups.exe tool.

Load_TS_List Method

Loads a tag status list into the active Tag Control Panel Picture. The Tag Control Panel Picture can be viewed when there are multiple tags associated with the selected object(s).

Syntax

*object.***Load_TS_List** *TagList*

Properties

The **Load_TS_List** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
TagList	An array of strings to be loaded into the tag control panel picture.

LoadImage Method

Loads the primary or secondary image at the specified index.

Syntax

*object.***LoadImage** *bPrimary, nIndex, bstrFileName*

Properties

The **LoadImage** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bPrimary	Boolean. If **True**, the primary image at the specified index is loaded. If **False**, the secondary image is loaded.
nIndex	Integer. The index of the image to load. This index is one-based.
bstrFileName	String. The path and file name of the image to load.

LoadTagGroupFile Method

Loads a tag group file into the picture.

Syntax

*object.***LoadTagGroupFile***bstrTagGroupFileName*

Properties

The **LoadTagGroupFile** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrTagGroupFileName	String. The name of the tag group file to load into the picture.

Remarks

If there is already a tag group file associated with the picture, performs an unload. The unload of the tag group would perform any clean up necessary. If the picture has not been resolved against the passed tag group file it peforms a Resolve action – performing substitutions and establishing connections; otherwise, it loads the correct stream from the tag group storage which will contain the data system blob and substituted string information. The load will establish connections between the picture tag group objects and the data source objects. It loops through the contained string list, reads the substitution string from the tag group file and writes the value to the objects properties.

The load will validate the picture version and tag group file version. If either does not match the current versions a re-resolve will occur. It also sends the messages informing linear and pen objects to reset their EGU information and sends the message informing any pen objects to recheck their historical data status.

LogicalToPercentage Method

Converts coordinates in logical units or postscript points and converts them to percentage of screen space available. This is useful if the user wishes to position a picture (whose window location is measured in screen percentages) next to a shape (whose position is measured in logical units or postscript points.).

Syntax

*object.***LogicalToPercentage** *plfTop, plfLeft, [plfHeight], [plfWidth]*

Properties

The **LogicalToPercentage** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

plfTop	Double. Specifies the logical top coordinate and returns the converted screen percentage coordinate.
plfLeft	String. The code to be written to the event in the form of strings.
plfHeight	Double. Specifies the logical left coordinate and returns the converted screen percentage coordinate.
plfWidth	Double. (Optional) Specifies the width of the page in logical units or postscript points and returns the width in screen percentage.

LogicalToUserFormPoint Method

Converts coordinates in logical units or postscript points and converts them to “UserForm Point” coordinate, which is the measure for position VBA user forms on screen.

Syntax

*object.***LogicalToUserFormPoint** *plfTop, plfLeft*

Properties

The **LogicalToUserFormPoint** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
plfTop	Double. Specifies the top coordinate and returns the converted UserForm point coordinate.
plfLeft	Double. Specifies the left coordinate and returns the converted userform point coordinate.

Remarks

Prior to performing the conversion, the **StartUpPosition** property of the form should be changed from *CenterOwner* to either *Manual* or *WindowsDefault*. A setting of *CenterOwner* will result in the form being popped up in the middle of the picture window.

M-P

MakeLinesHorizontal Method

Makes the selected lines horizontal.

Syntax

*object.***MakeLinesHorizontal**

Properties

The **MakeLinesHorizontal** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

MakeLinesVertical Method

Makes the selected lines vertical.

Syntax

*object.***MakeLinesVertical**

Properties

The **MakeLinesVertical** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

MakeSameSize Method

Makes the currently selected object's specified dimensions the same size. It is the equivalent to selecting MakeSameSize from the Format menu.

Syntax

*object.***MakeSameSize** *type*

Properties

The **MakeSameSize** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
type	Integer. The dimension to be used. **Valid entries:** 0 – Height 1 – Width 2 – Both

Remarks

MakeSameSize is a Configuration environment method only.

Modify Method

Displays the Modify Block dialog box for the specified block.

Syntax

*object.***Modify** *bstrFullyQualifiedName, bReadOnly, iStatus*

Properties

The **Modify** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrFullyQualifiedName	String. The fully qualified name of the block you want to modify.
bReadOnly	Boolean. **True** if the block can only be showed, **False** if the block can be modified.
Istatus	Long. Returns the error status value. Return values are: 0 – OK 1 – Syntax error 2 – Data Undefined 3 – Data type mismatch

ModifyColumnLength Method

Modifies the specified column in the Legend to display the number of characters specified.

Syntax

*object.***ModifyColumnLength** *iColumn, iNumChars*

Properties

The **ModifyColumnLength** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
iColumn	Integer. The column to modify.
iNumChars	Integer. The number of characters to display in the column.

Move Method

Moves the object to a new position determined by the X and Y offsets.

Syntax

*object.***Move** *fXOffset, fYOffset*

Properties

The **Move** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
fXOffset	Double. The offset to move the shape in the horizontal direction in postscript points or logical units.
fYOffset	Double. The offset to move the shape in the vertical direction in postscript points or logical units.

Open Method

Opens a **Document** in the WorkSpace. This is equivalent to selecting Open from the File menu.

Syntax

*object.***Open***(Filename, [DisplayOption])*

Properties

The **Open** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Filename	String. The name of the file to open. The file name includes the full path and extension. If a file name is not specified, the user is prompted to enter one.
DisplayOption	Long. (Optional) **Valid entries:** 1 – Load only. 2 – Load and activate the document in a hidden window. This flag is a run-time only option. 3 – (default) Load and display the document normally.

Return Value

Object. The dispatch pointer to the opened **Document**.

Open_QT_Pic Method

Opens a Quick Trend Picture for the selected object(s) with a set of pens based on the first eight (8) valid tags.

NOTE: To allow a user to open multiple instances of the Quick Trend Picture, use the Open_QT_Pic_Ex Method.

Syntax

*object.***Open_QT_Pic**

Properties

The **Open_QT_Pic** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Open_QT_Pic_Ex Method

Opens a Quick Trend Picture for the selected object(s) with a set of pens based on the first eight (8) valid tags. This method allows you to do the same as the Open_QT_Pic Method, but allows for multiple instances.

Syntax

object.**Open_QT_Pic_Ex (*OpenMultipleInstances*)**

Properties

The **Open_QT_Pic_Ex** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
OpenMultipleInstances	Integer. If set to 1 (true) another instance of the picture is opened. If set to 0 (false), no additional instances of this picture are opened.

Open_TCP_Pic Method

Opens a Tag Control Panel Picture that displays up to 20 of the valid tags associated with the currently selected object.

NOTE: To allow a user to open multiple instances of the Tag Control Panel Picture, use the Open_TCP_Pic_Ex Method.

Syntax

object.**Open_TCP_Pic**

Properties

The **Open_TCP_Pic** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Open_TCP_Pic_Ex Method

Opens a Tag Control Panel Picture that displays up to 20 of the valid tags associated with the currently selected object. This method allows you to do the same as the Open_TCP_Pic Method, but allows for multiple instances.

Syntax

*object.***Open_TCP_Pic_Ex (*OpenMultipleInstances*)**

Properties

The **Open_TCP_Pic_Ex** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
OpenMultipleInstances	Integer. If set to 1 (true) another instance of the picture is opened. If set to 0 (false), no additional instances of this picture are opened.

Open_TS_Pic Method

Opens the Tag Status Picture for first found tag for the selected object(s).

NOTE: To allow a user to open multiple instances of the Tag Status Picture, use the Open_TS_Pic_Ex Method.

Syntax

*object.***Open_TS_Pic**

Properties

The **Open_TS_Pic** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Open_TS_Pic_Ex Method

Opens the Tag Status Picture for first found tag for the selected object(s). This method allows you to do the same as the Open_TS_Pic Method, but allows for multiple instances.

Syntax

*object.***Open_TS_Pic_Ex (*OpenMultipleInstances*)**

Properties

The **Open_TS_Pic_Ex** method syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
OpenMultipleInstances	Integer. If set to 1 (true) another instance of the picture is opened. If set to 0 (false), no additional instances of this picture are opened.

Open_TS_Pic_Type Method

Specifies the type of Tag Status picture to open.

NOTE: To allow a user to open multiple instances of a Tag Status picture, use the Open_TS_Pic_Type_Ex Method.

Syntax

*object.***Open_TS_Pic_Type** (*TSPicType, [TagList])*

Properties

The **Open_TS_Pic_Type** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
TSPicType	TS_PIC_TYPE (Optional) The type of Tag Status picture to open: 0 – Single Tag Status picture 1 – Quick Trend picture 2 – Tag Control Panel picture
TagList	Variant. (Optional) An array of strings to be used in the tag status picture.

Open_TS_Pic_Type_Ex Method

Specifies the type of Tag Status picture to open. This method allows you to do the same as the Open_TS_Pic_Type Method, but allows for multiple instances.

Syntax

*object.***Open_TS_Pic_Type** (*TSPicType, [TagList], OpenMultipleInstances*)

Properties

The **Open_TS_Pic_Type_Ex** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
TSPicType	TS_PIC_TYPE (Optional) The type of Tag Status picture to open: 0 – Single Tag Status picture 1 – Quick Trend picture

	2 – Tag Control Panel picture
TagList	Variant. (Optional) An array of strings to be used in the tag status picture.
OpenMultipleInstances	Integer. If set to 1 (true) another instance of the picture is opened. If set to 0 (false), no additional instances of the picture are opened.

ParseConnectionSource Method

Parses the specified Data Source to determine if it is a valid connection.

Syntax

*object.***ParseConnectionSource** *bstrPropertyName, bstrSource, iStatus, pvaValidObjects, pvaUndefinedObjects, bstrFullyQualifiedSource*

Properties

The **ParseConnectionSource** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrPropertyName	String. The name of the property being animated.
bstrSource	String. The Data Source. This can be either a Data System data source or a shape's property.
iStatus	Long. Returns the status of the connection. Return values are: 0 – OK 1 – Invalid Syntax (Data Source could never exist 2 – Undefined Object (Data Source does not exist, could be added) 3 – Data Type Mismatch
pvaValidObjects	Variant. Returns an array of objects that are identified as part of the Data Source and are currently valid and used in the system.
pvaUndefinedObjects	Variant. Returns an array of objects that are identified as part of the Data Source but are not yet defined (Use Anyway objects).
bstrFullyQualifiedSource	Variant. Returns an array of objects that are identified as part of the Data Source but are not yet defined (Use Anyway objects).

Remarks

The object that you apply this method to has a property that may be animated by another object. Parsing checks to see the status of the Data Source and also checks to see if each object that is part of the source is defined or undefined.

ParseConnectionSource also checks complex expressions and returns individual sources. For example, when used on *AI1+AI2*, the method will return *AI1* and *AI2* in the Valid or Undefined object array.

Paste Method

Pastes the contents of the Clipboard into the document.

Syntax

*object.***Paste**

Properties

The **Paste** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Paste is a Configuration environment method only.

PasteFromClipboard Method

Paste KeyMacros from the clipboard into the key macro collection

Syntax

*object.***PasteFromClipboard** (*Overwrite*)

Properties

The **PasteFromClipboard** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Overwrite	Boolean (optional). Indicates whether the to add the key macro objects from the clipboard, or to remove all the existing key macro objects before adding the new key macro objects. **False** will simply add the new key macro objects to the collection, while **True** will first clear the collection before adding the new key macro objects. **Note:** While appending any key combinations found in the existing collection will be overwritten by the key macro object from the clipboard. A prompt appears to state that duplicates were found and asks whether you want to abort or continue.

PasteSpecial Method

Opens the Paste Special dialog box.

Syntax

*object.***PasteSpecial**

Properties

The **PasteSpecial** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

PasteSpecial is a Configuration environment method only.

Pause Method

Pauses the **Chart**.

Syntax

*object.***Pause**

Properties

The **Pause** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

A paused **Chart** can resume displaying data by calling the **Resume** method. If **Resume** is not called, the **Chart** will automatically resume after the time interval specified in the **Timeout** property.

PauseAlarmRead Method

Temporarily disables the alarm refresh rate and instructs the **Alarm Summary** object to stop updating its spreadsheet.

Syntax

object.**PauseAlarmRead**

Properties

The **PauseAlarmRead** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Use the **ResumeAlarmRead** method to resume refreshing the **Alarm Summary** object.

PercentageToLogical Method

Converts coordinates in percentage of screen space available to postscript points or logical units. This is useful if the user wishes to position a picture (whose window location is measured in screen percentages) next to a shape (whose position is measured in postscript points or logical units).

Syntax

*object.***PercentageToLogical** *plfTop, plfLeft, [plfHeight], [plfWidth]*

Properties

The **PercentageToLogical** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
plfTop	Double. Specifies the top screen percentage coordinate and returns the converted coordinate.
plfLeft	Double. Specifies the left screen percentage coordinate and returns the converted coordinate.
plfHeight	Double. (Optional) Specifies the height of the page in screen percentage and returns the height in postscript point or logical unit coordinates.
plfWidth	Double. (Optional) Specifies the width of the page in screen percentage and returns the width in postscript point or logical unit coordinates.

PercentageToPixel Method

Converts a screen percentage (i.e., a window location) to pixel coordinates that are relative to the picture's coordinate system. This is useful if the user is working with an ActiveX control which requires inputs in the form of pixels.

Syntax

*object.***PercentageToPixel** *plfTop, plfLeft, [plfHeight], [plfWidth]*

Properties

The **PercentageToPixel** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
plfTop	Double. Specifies the top screen percentage coordinate and returns the converted pixel coordinate.
plfLeft	Double. Specifies the left screen percentage coordinate and returns the converted pixel coordinate.
plfHeight	Double. (Optional) Specifies the height of the page in screen percentage and returns the height in pixel coordinates.
plfWidth	Double. (Optional) Specifies the width of the page in screen percentage and returns the width in pixel coordinates.

PixelToPercentage Method

Converts pixel coordinates that are relative to the picture's coordinate system to a screen percentage (i.e., a window location).

Syntax

*object.***PixelToPercentage** *plfTop, plfLeft, [plfHeight], [plfWidth]*

Properties

The **PixelToPercentage** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
plfTop	Double. Specifies the pixel top coordinate and returns the converted screen percentage coordinate.
plfLeft	Double. Specifies the pixel left coordinate and returns the converted screen percentage coordinate.
plfHeight	Double. (Optional) Specifies the height of the page in pixel coordinates and returns the height in screen percentage.
plfWidth	Double. (Optional) Specifies the width of the page in pixel coordinates and returns the width in screen percentage.

PrintChart Method

Sends the Enhanced Chart to the printer for output. Causes the Printer Setup dialog box to display.

Syntax

*object.***PrintChart***enuSizeUnits, [dblWidth], [dblHeight]*

Properties

The **PrintChart** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
enuSizeUnits	Enum/Long. Specifies in what units the image size is specified with *dblWidth* and *dblHeight*. It can be either of type enumSizeUnits or a corresponding numerical value, as follows: SizeUnits_FullPage (0). Prints to the full page. SizeUnits_Millimeters (1): in millimeters. SizeUnits_Inches (2): in inches. SizeUnits_Points (3): in Postscript points (1/72 inches). **NOTE:** In order to use the enumSizeUnits enumerations, you must add the type lib-

<table>
<tr><td></td><td>rary file for object to the references of the VBA project. If the type library file is not included in the references, then only numerical values are accepted. The type libraries for objects in the Applied To list are as follows:
<table>
<tr><th>Object</th><th>Reference</th><th>Type Library File Name</th></tr>
<tr><td>HistogramChart</td><td>iFix 2D Histogram Chart Object v1.0 Type Library</td><td>Fix2DHistogramChartDll.tlb</td></tr>
<tr><td>LineChart</td><td>iFix 2D Line Chart Object v1.0 Type Library</td><td>Fix2DLineChartDll.tlb</td></tr>
<tr><td>SPCBarChart</td><td>iFix 2D SPC Bar Chart Object v1.0 Type Library</td><td>Fix2DSPCBarChartDll.tlb</td></tr>
<tr><td>XYChart</td><td>iFix 2D XY Chart Object v1.0 Type Library</td><td>Fix2DXYChartDll.tlb</td></tr>
</table>
</td></tr>
<tr><td><i>dblWidth</i></td><td>Double. Specifies the width of the printed image, in the units specified in <i>enuSizeUnits</i>. This parameter is optional and ignored when SizeUnits_FullPage is specified.</td></tr>
<tr><td><i>dblHeight</i></td><td>Double. Specifies the height of the printed image, in the units specified in <i>enuSizeUnits</i>. This parameter is optional and ignored when SizeUnits_FullPage is specified.</td></tr>
</table>

Remarks

The aspect ratio must be between 0.1 and 10.

PrintOut Method

Prints the **Document** object.

Syntax

*object.***PrintOut**()

Properties

The **PrintOut** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Return Value

Boolean. **True** if printing was successful; **False** otherwise.

PromptToChangePassword Method

Displays the Change Password dialog box and allows the user to change his password.

Syntax

object.**PromptToChangePassword***(bstrUsername, [bstrDomain])*

Properties

The **PromptToChangePassword** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrUserName	String. Name of the user (Windows user name).
bstrDomain	String. (Optional). The Windows domain name where the user account is located. For a local user account, omit or pass an empty string.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

Q-R

Quit Method

Shuts down the WorkSpace, closing all documents and saving them, if specified.

Syntax

object.**Quit** *[SaveChanges]*

Properties

The **Quit** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
SaveChanges	Long. (optional) **Valid entries:** 1 – Save changes; do not prompt. (default) 2 – Do not save changes. 3 – Prompt before saving changes.

Read Method

Reads in the **Value**, **Timestamp**, and **Quality** of the data source represented by the **DataItem**.

Syntax

object.**Read**

Properties

The **Read** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Refresh Method

Repaints an object.

Syntax

object.**Refresh**

Properties

The **Refresh** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

RefreshChartData Method

Refetches and replots data in an Enhanced Chart (HistogramChart, LineChart, SPCBarChart, or XYChart Object. object).

Syntax

object.**RefreshChartData**

Properties

The **RefreshChartData** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remove Method

Removes a member from the specified collection.

Syntax

*object.***Remove** *lIndex*

Properties

The **Remove** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
lIndex	Long. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the Collection's **Count** property.

DataItems and Groups Collection Syntax

*object.***Remove** *vtIndex*

Properties

The **Remove** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
tIndex	Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the Collection's **Count** property. If a string expression, the index is the object name.

RemoveAll Method

Removes all members from a Lines collection.

Syntax

*object.***RemoveAll**

Properties

The **RemoveAll** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

RemoveAllLevels Method

Removes all levels from the **Lookup** table.

Syntax

*object.***RemoveAllLevels**

Properties

The **RemoveAllLevels** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

RemoveItem Method

Removes the specified column from the Legend.

Syntax

*object.***RemoveItem** *iColumn*

Properties

The **RemoveItem** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
iColumn	Integer. The index of the column to remove.

RemoveKeyMacro Method

Deletes the key macro object defined by key combination, if one exists.

Syntax

*object.***RemoveKeyMacro** (*variant*)

Properties

The **RemoveKeyMacro** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
variant	Location or DisplayName of the key macro to be removed.

RemoveLegendItem Method

Removes the specified Legend item.

Syntax

*object.***RemoveLegendItem** *szItem*

Properties

The **RemoveLegendItem** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
szItem	String. The name of the item to be removed. **Valid entries:** Source – Data Source name Description - Data Source's descriptor property Value – Current Value at the time cursor Units – EGU units name Mode – Historical or real time High Limit – High display limit Low Limit – Low display limit Interval – Data point interval High Over – Highest value over the duration Low Over – Lowest value over the duration Avg Over – Average value over the duration USER1 – User defined field USER2 – User defined field USER3 – User defined field USER4 – User defined field USER5 – User defined field USER6 – User defined field USER7 – User defined field USER8 – User defined field USER9 – User defined field USER10 – User defined field

RemoveLevel Method

Removes a level based on the index.

Syntax

*object.***RemoveLevel** *iIndex*

Properties

The **RemoveLevel** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
iIndex	Long. The index of the level to remove.

RemoveObject Method

Removes the specified object from a schedule.

Syntax

*object.***RemoveObject** *bstrObjectName*

Properties

The **RemoveObject** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrObjectName	String. The name of object to be removed.

Remarks

It is necessary to call the **DoMenuCommand** method for the schedule with the *scHREFreshView* parameter in order for the object to appear as removed from the Schedule.

RemovePictureFromStartupList Method

Removes pictures from the iFIX WorkSpace's startup lists. The startup lists determine the pictures that will be opened automatically when the WorkSpace starts.

Syntax

*object.***RemovePictureFromStartupList** *bstrPictureName, bMode*

Properties

The **RemovePictureFromStartupList** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrPictureName	String. Name of the picture to be removed from the startup list. You must include the file path and extension
bMode	Boolean. Specifies whether to remove the picture when the WorkSpace starts in the Configuration environment or in the Runtime environment. Valid entries: 0 – AppConfigurePicturePreferences 1 – AppRunPicturePreferences

ReplaceDocument Method

Replaces the existing document with a new **Document** in the existing document's window.

Syntax

*object.***ReplaceDocument**(*FileName*)

Properties

The **ReplaceDocument** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
FileName	String. The name of the new document to open.

Return Value

Object. The dispatch pointer to the new **Document**.

Remarks

If possible, always make this call the last line in your script. Note that when the **ReplaceDocument** method is used, and the document being replaced is the document that contains the script, the call *must* be the last line in the script. Otherwise, you may experience unexpected behavior when executing the script.

If the **ReplaceDocument** method is not called from the picture being replaced and is not the last line in your script, be certain that the operation is complete before the rest of the script continues to execute.

ReplaceInString Method

Replaces a match occurrence in one string with another string.

Syntax

*object.***ReplaceInString** *pbstrTargetString, bstrReplacementString, bstrMarkedMatchString, lFirstChar, lCharCount, lFlags, pbstrResultString, pbSuccess*

Properties

The **ReplaceInString** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pbstrTargetString	String. Target string to which the replacement is to be made.
bstrReplacementString	String. String to be substituted for the sub-string identified by *lFirstChar* and *lCharCount* in the target string.
bstrMarkedMatchString	String. String returned from a previous **FindInString** method call.
lFirstChar	Long. One-based index of the first character to be replaced in the target string returned from a previous **FindInString** method call.
lCharCount	Long. Number of characters to be replaced in the target string returned from a previous **FindInString** method call.
lFlags	Long. Search modifiers. Valid entries: 0-15 in any of the following combinations: 0 – No modifiers 1 – Match Case

	2 – Whole Word Only 4 – Data Source Only 8 – Include Scripts
pbstrResultString	String. Returns the new string with the specified replacement.
pbSuccess	Boolean. Returns **True** if the operation succeeded, **False** otherwise.

Remarks

Calling **ReplaceInString** after calling **FindInString** produces the same result as calling the **FindReplaceInString** method.

Replace_QT_Pic Method

Replaces the current **picture** with a Quick Trend picture.

Syntax

object.**Replace_QT_Pic**

Properties

The **Replace_QT_Pic** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Replace_TCP_Pic Method

Replaces the current **picture** with a Tag Control Panel picture.

Syntax

object.**Replace_TCP_Pic**

Properties

The **Replace_TCP_Pic** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Replace_TS_Pic Method

Replaces the current **picture** with a Tag Status picture.

Syntax

*object.*__Replace_TS_Pic__

Properties

The **Replace_TS_Pic** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Replace_TS_Pic_Type Method

Replaces the current **picture** with the specified Tag Status picture type.

Syntax

*object.*__Replace_TS_Pic_Type__ *TSPicType; [TagList]*

Properties

The **Replace_TS_Pic_Type** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
TSPicType	TS_PIC_TYPE The type of Tag Status picture to open: 0 – Single Tag Status picture 1 – Quick Trend picture 2 – Tag Control Panel picture
TagList	Variant. (Optional) An array of strings to be used in the tag status picture.

ResetChartData Method

Resets the data displayed in a **Chart**.

Syntax

*object.*__ResetChartData__

Properties

The **ResetChartData** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ResetObjectStats Method

Resets the statistics for the specified objects.

Syntax

*object.***ResetObjectStats** *vObjectNames*

Properties

The **ResetObjectStats** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ObjectNames	Variant. The object(s) for which you want to reset statistics. **Valid entries:** 0 – All Objects 1 – All Timer Objects 2 – All Event Objects A string containing the name of the object to reset. A list of objects to reset.

ResetStats Method

Resets the statistics of the **Timer** or **Event** object to zero, including the number of times fired, and the time stamp of the last time fired.

Syntax

*object.***ResetStats**

Properties

The **ResetStats** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ResetZoom Method

Resets the **Chart** to its default viewing area.

Syntax

*object.***ResetZoom**

Properties

The **ResetZoom** method syntax has this part:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.

Remarks

ResetZoom is used to reset the **Chart** to its default state after calling the **Zoom** method.

ResolveTagGroupFile Method

Resolves the picture against the passed tag group file.

Syntax

*object.***ResolveTagGroupFile** *bstrTagGroupFileName*

Properties

The **ResolveTagGroupFile** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrTagGroupFileName	String. The name of the tag group file to resolve the picture against.

Remarks

This method loops through the list of tag group objects in the picture, performs the necessary substitutions based on the definitions in the tag group file, and establishes connections with these data sources. It then persists identifying information of the picture and tag group file, a list of the complete strings after substitution, and the connection information into a unique stream in the tag group storage. Once this is saved, all the connections between the tag group objects and data sources will be broken.

Resume Method

Resumes the **Chart** after it has been paused.

Syntax

*object.***Resume**

Properties

The **Resume** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Resume is used to resume the chart after calling the **Pause** method.

ResumeAlarmRead Method

Resumes updating the Alarm Summary object after it has been paused.

Syntax

object.**ResumeAlarmRead**

Properties

The **ResumeAlarmRead** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Use the **ResumeAlarmRead** method after calling the PauseAlarmRead method.

RetrieveDefinition Method

Retrieves the definitions contained in a tag group file.

Syntax

object.**RetrieveDefinition** *TagGroupName*, *Count*, *TokenList*, *ReplacementList*, *DescriptionList*

Properties

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
TagGroupName	String. The name of the tag group file in the Picture path to open.
Count	Short. The number of symbols in the tag group file.
TokenList	Variant. The array of symbols. The array index ranges from 0 to Count-1.
ReplacementList	Variant. The array of substitutions. The array index ranges from 0 to Count-1.
DescriptionList	Variant. The array of descriptions associated with each substitution. The array index ranges from 0 to Count-1.

RetrieveTagGroupVariables Method

Retrieves a list of all tag group references in the picture, including connections, string substitutions, and scripts.

Syntax

*object.***RetrieveTagGroupVariables** *intCountOfVars, varTagGroupVars*

Properties

The **RetrieveTagGroupVariables** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
intCountOfVars	Integer. The number of tag group symbols.
arTagGroupVars	Variant. List of all tag group symbols in the picture.

Remarks

The tag group reference list will only contain the tag group symbols, not the full syntax of partial substitutions. The list will have a single entry per tag group symbol regardless of the number of uses.

Rotate Method

Rotates the shape according to the angle, specified in either degrees or radians.

Syntax

*object.***Rotate** *fAngle, bInRadians*

Properties

The **Rotate** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
fAngle	Double. The amount of angle to rotate the shape.
bInRadians	Boolean. If **True**, rotate the object by the specified angle in radians. If **False**, rotate the object by the specified angle in degrees.

RunObject Method

Starts or stops the **Timer** and/or **Event** from running.

Syntax

*object.***RunObject** *vObjectNames, bRunStatus*

Properties

The **RunObject** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ObjectNames	Variant. The object(s) for which you want to reset statistics.

	Valid entries: 0 - All Objects 1 - All Timer Objects 2 - All Event Objects A string containing the name of the object to reset. A list of objects to reset.
bRunStatus	Boolean. If **True**, the event is running. If **False**, the event is stopped.

Remarks

To stop an event that is running, pass in **False** for *bRunStatus*.

S

Save Method

Saves the **Document** to disk. When called off the **Documents** Collection, this method saves all open documents in the iFIX WorkSpace. This is equivalent to selecting SaveAll from the File menu. When used with the **Document** object, this method saves the **Document** object with the optionally specified file name.

NOTE: The Save method is not available in WorkSpace run mode.

DocumentsCollection Object Syntax

*object.***Save** *[Prompt]*

Properties

The **Save** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Prompt	Boolean. (Optional) If **True**, prompt the user to save changes. If **False**, do not prompt the user. (default)

Remarks

Note that if the **Save** method for the **Documents Collection** is called for newly created pictures, the user is prompted to save changes regardless of the value specified in *bPrompt*. This is because pictures cannot be saved as their default name (e.g. “Untitled#”).

Document Object Syntax

*object.***Save** *[Filename], [PromptToSave]*

Properties

The **Save** method syntax has these parts:

Part	Description

object	An object expression that evaluates to an object in the Applies To list.
FileName	String. (Optional) The file name.
PromptToSave	Boolean. (Optional) If **True**, prompt the user before saving. (default)

Remarks

If the **Save** method for the **Document** object is called for newly created pictures, the user is prompted to save changes regardless of the value specified in *PromptToSave*. This is because pictures cannot be saved as their default name (e.g. “Untitled#”). The user must also specify the appropriate extension for the document type in the filename. The following table contains the extensions and their document types.

Extension	Document Type
.fxg	Fix Picture
.fds	Fix Dynamo Set
.evs	Fix Schedule
.doc	Microsoft Word Document
.xls	Microsoft Excel Worksheet
.xls	Microsoft Excel Chart

If the **Save** method for the **Document** object is called for an existing file, the file is overwritten. You may want to check for file existence before calling the Save method.

Note that if the **Save** method is called using the **Save** *[Filename]* syntax, a Save As operation is performed.

Save_TS_List Method

Saves the tag status list to the Tag Status subdirectory of the PIC folder in your iFIX installation location. The tag status list is saved as a ".tags" file.

Syntax

*object.***Save_TS_List**

Properties

The **Save_TS_List** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

SaveAsSVG Method

Saves pictures as Scalable Vector Graphics, with a .svg extension.

NOTE: Portal uses the SVG format for importing picture files.

Syntax

*object.***SaveAsSVG**

Properties

The **SaveAsSVG** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

SaveToHistoryList Method

Places the specified string in the history list of the animation expression control. It will subsequently appear in the history drop-down combo box of the animation dialogs, VBA forms which contain an expression control (such as the animation experts) and chart pen configuration.

Syntax

*object.***SaveToHistoryList** *szHistoryItem*

Properties

The **SaveToHistoryList** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
szHistoryItem	String. The item to save to the history list.

Return Value

Integer. Returns 1 if the operation was successful.

ScrollBack Method

Scrolls back in the **Chart** by the factor specified in the chart's **ScrollPercentage** property.

Syntax

*object.***ScrollBack**

Properties

The **ScrollBack** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ScrollForward Method

Scrolls forward in the Chart by the factor specified in the chart's ScrollPercentage property.

Syntax

*object.***ScrollForward**

Properties

The **ScrollForward** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ScrollTimeBack Method

Scrolls time back in the specified Pen by the factor specified in the pen's ScrollPercentage property.

Syntax

*object.***ScrollTimeBack**

Properties

The **ScrollTimeBack** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ScrollTimeForward Method

Scrolls time forward in the specified Pen by the factor specified in the pen's ScrollPercentage property.

Syntax

*object.***ScrollTimeForward**

Properties

The **ScrollTimeForward** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ScrollToPosition Method

Scrolls the picture to a specified position using X and Y coordinates. For use on a picture using Enhanced Coordinates only.

Syntax

*object.***ScrollToPosition** *IdInXValue*, *IdInYValue*

Properties

The **ScrollToPosition** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
IdInXValue	Double. The number that represents the X coordinate value.
IdInYValue	Double. The number that represents the Y coordinate value.

Remarks

ScrollToPosition is a Run mode environment method only.

Select Method

Selects the specified object.

Syntax

*object.***Select**

Properties

The **Select** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

SetAlarmBackgroundColor Method

Sets the row background color to display for the alarms with the specified alarm priority.

Syntax

*object.***SetAlarmBackgroundColor** *PriorityId, PriorityColor*

Properties

The **SetAlarmBackgroundColor** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
PriorityId	Integer. The ID corresponding to the alarm priority.

	Valid entries for iFIX 3.5 and earlier: 0 - High 1 - Medium 2 - Low Valid entries for iFIX 4.0 and later: 3 - CRITICAL 4 - HIHI 5 - HIGH 6 - MEDIUM 7 - LOW 8 - LOLO 9 - INFO (INFORMATIONAL)
PriorityColor	OLE_COLOR. The row background color to display for the alarms with the alarm priority.

SetAlarmForegroundColor Method

Sets the row foreground color to display for the alarms with the specified status.

Syntax

*object.***SetAlarmForegroundColor** *StatusId, StatusColor*

Properties

The **SetAlarmForegroundColor** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
StatusId	Integer. The ID corresponding to the alarm status. **Valid entries:** 0 – OK 1 – LOLO 2 – HIHI 3 – LO 4 – HI 5 – RATE 6 – COS 7 – CFN 8 – DEV 9 – FLT 10 – DSAB 11 – ERROR 12 – ANY 13 – NEW 14 – TIME 15 – IOF 16 – OCD

	17 – UNDER 18 – OVER 19 – RANGE 20 – COMM 21 – DEVICE 22 – STATION 23 – ACCESS 24 – SQL LOGIN 25 – SQL CMD 26 – DAT MATCH 27 – FLD READ 28 – FLD WRITE 29 – NO DATA 30 – NO XDATA
StatusColor	OLE_COLOR. The color to display for the alarm status.

SelectAlarmRow Method

Selects a row in the **Alarm Summary** object.

Syntax

object.**SelectAlarmRow(***RowNum* **As Integer,** *bSelect* **As Boolean) As Integer**

Properties

The **SelectAlarmRow** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
RowNum	Integer. The row in the **Alarm Summary** object's spreadsheet you want to select. Row numbers start at 1 and you can specify any row even if it is not visible on the screen.
bSelect	Boolean. When True, the row is selected. When False, the row is not selected.

Return Value

Zero. Reserved for future use.

Remarks

Selecting a row that is not displayed on the screen does not cause the spreadsheet to scroll to the selected row.

If you select a row that is not visible, you will not receive the Acknowledgement status, the Latched Alarm status, the Value, and the User Defined Columns when calling the **GetSelectedRow** or **GetSelectedUserDefFields** methods. You can work around this by reading these values after calling the **GetSelectedRow** method.

SelectAll Method

Selects all objects in the document.

Syntax

*object.***SelectAll**

Properties

The **SelectAll** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

SelectObject Method

Selects the specified object and/or group of objects.

Syntax

*object.***SelectObject** *bSingleSelect*

Properties

The **SelectObject** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bSingleSelect	Boolean. If **True**, the user is only allowed to do single object selection. If **False,** the user is allowed to select multiple objects.

Remarks

When *bSingleSelect* is **False**, the **SelectObject** method performs object selection as if the user is holding the CTRL key when clicking on objects to select them.

SendOperatorMessage Method

Sends an event message to the specified node. If no node is specified, it the message is sent to the local node This message is sent to all of the typers including the alarm history window.

Syntax

*object.***SendOperatorMessage** *Text, [NodeName]*

Properties

The **SendOperatorMessage** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Text	String. The message to send.
NodeName	String. (Optional) The fully qualified datasource name.

SendSignedOperatorMessage Method

Sends the signed operator message to the alarm system.

Syntax

object.**SendSignedOperatorMessage***(bstrMessageText, bstrNodeName, bstrTagName, bstrPerformedByUserID, [bstrPerformComment], [bstrVerifiedByUserID], [bstrVerifyComment])*

Properties

The **SendSignedOperatorMessage** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrMessageText	String. Text of the signed operator message.
bstrNodeName	String. Name of the SCADA node the you made the change on.
bstrTagName	String. Name of the tag that has been changed (may be empty string).
bstrPerformedByUserID	String. Performed By user ID.
bstrPerformComment	String. (Optional). Performed By comment.
bstrVerifiedByUserID	String. (Optional). Verified By user ID.
bstrVerifyComment	String. (Optional). Verified By comment.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

SendToBack Method

Moves the selected object to the back of the stack of objects, making it the bottom object in the stack. It is equivalent to selecting Send To Back from the Format menu.

Syntax

object.**SendToBack**

Properties

The **SendToBack** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

SendToBack is a Configuration environment method only.

If you select several objects and call **SendToBack**, the selected objects are placed at the bottom of the stack, however, they keep their positions relative to one other. The **SendToBack** method is useful for creating complex shapes and using stacking or masking techniques.

SetContinuousUser Method

Sets the continuous user.

Syntax

object.**SetContinuousUse***(bstrUserName)*

Properties

The **SetContinuousUser** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrUserName	String. Name of the continuous user.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

SetCurrrentValue Method

Sets the current value, time and quality for a **Pen**.

Syntax

*object.***SetCurrentValue** *cValue, dt, lQual*

Properties

The **SetCurrentValue** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
cValue	Double. The value to set for the **Pen**.
dt	Date. The date at which to set the value for the **Pen**.

lQual	Long. The quality at which to set the value for the **Pen**.

SetDispatch Method

Reserved for internal purposes.

SetDispid Method

Reserved for internal purposes.

SetDuration Method

Sets the length of time to display the **Chart**.

NOTE: Tying a script to this method that triggers faster than 5 seconds is not recommended.

Syntax

object.**SetDuration** *days, hours, minutes, seconds*

Properties

The **SetDuration** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
days	Long. The number of days to display the **Chart**.
hours	Long. The number of hours to display the **Chart**.
minutes	Long. The number of minutes to display the **Chart**.
seconds	Long. The number of seconds to display the **Chart**.

SetGlobalDuration Method

Sets the GlobalDuration property of the the Global Time Control.

Syntax

object.**SetGlobalDuration** *days, hours, minutes, seconds*

Properties

The **SetGlobalDuration** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
days	Long. The number of days to display the object.
hours	Long. The number of hours to display the object.
minutes	Long. The number of minutes to display the object.
seconds	Long. The number of seconds to display the object.

SetGlobalHistoricalUpdateRate Method

Sets the historical update rate for the historical data sources in run mode.

Syntax

*object.***SetGlobalHistoricalUpdateRate** *hrs, mins, secs*

Properties

The **SetGlobalHistoricalUpdateRate** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
hrs	Long. The number of hours at which to set the duration.
mins	Long. The number of minutes at which to set the duration.
secs	Long. The number of seconds at which to set the duration.

SetGlobalMovingEndTimeToCurrent Method

Sets the end time of the Global Time Control to the current time.

NOTE: Tying a script to this method that triggers faster than 5 seconds is not recommended.

Syntax

*object.***SetGlobalMovingEndTimeToCurrent**

Properties

The **SetGlobalMovingEndTimeToCurrent** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

SetFocusToComboBox Method

Sets focus to the ComboBox field of the **ExpressionEditor**.

Syntax

*object.***SetFocusToComboBox**

Properties

The **SetFocusToComboBox** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

SetIndirectionInfo Method

Reserved for internal purposes.

SetInterval Method

Sets the interval of time to elapse between data points in a **Chart**.

Syntax

*object.***SetInterval** *days, hours, minutes, seconds*

Properties

The **SetInterval** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
days	Long. The number of days between data points.
hours	Long. The number of hours between data points.
minutes	Long. The number of minutes between data points.
seconds	Long. The number of seconds between data points.

SetKeyCombination Method

Used to set both the KeyCode and CombinationKey simultaneously.

Syntax

object.**SetKeyCombination** (*ComboKey, KeyCode*)

Properties

The **SetKeyCombination** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ComboKey	enumCombinationKey. The control shift part of the key combination.
KeyCode	Integer. The ASCII value of the main key of the key combination.

SetLegendMask Method

Indicates which legend items to show in the GeneralDataSet Object or RealTimeSPCDataSet object.

Syntax

object.**SetLegendMask***lngMask*

Properties

The **SetLegendMask** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
lngMask	A value that represents the Legend items to show in the data set: LegendMaskSourceName (1) LegendMaskDescription (2) LegendMaskCurrentValue(4) LegendMaskLowLimit (8) LegendMaskHighLimit (16) LegendMaskAvgOverRange (32) LegendMaskLowOverRange (64) LegendMaskHighOverRange (128) LegendMaskQuality (256) **NOTE:** In order to use the lngMask enumerations, you must add the type library file for object to the references of the VBA project. If the type library file is not included in the references, then only numerical values are accepted. The type libraries for objects in the Applied To list are as follows:

Object	Reference	Type Library File Name
GeneralDataSet	iFIX GeneralDataSet Object v1.0 Type Library	FixGeneralDataSetDll.tlb
RealTimeSPCDataSet	iFIX SPCRealTimeDataSet Object v1.0 Type Library	FixSPCRealTimeDataSetDll.tlb

Remarks

Calling this method is functionally equivalent to setting the *LegendMask* property. However, if the type library is included in the project references, VBA's IntelliSense will display a user-friendly list of legend mask symbols while you are typing in the VBA Code Window, and there is no need to memorize or look up for numerical values of the legend mask.

For example, if you are combining multiple mask values using the bitwise OR operation, such as

```
object.SetLegendMask LegendMask HighLimit | LegendMask LowLimit
```

and you want VBA IntelliSens to display the list repeatedly, you need only to type the vertical bar (the OR operator) before the first mask value, move the cursor before it, and invoke the menu item by pressing Ctrl + Shit + J.

SetNumericFormat Method

Sets the format of a numeric value.

Syntax

*object.*__SetNumericFormat__ *[vaWholeDigits], [vaDecimalDigits], [vaJustify]*

Properties

The **SetNumericFormat** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
aWholeDigits	Variant. (Optional) The number of whole digits to display.
aDecimalDigits	Variant. (Optional) The number of decimal digits to display.
aJustify	Variant. (Optional) The justification of the numeric value. **Valid entries:** 0 – Left 1 – Center 2 – Right

SetPenDataArray Method

Creates a static Pen in a Chart with the specified data. Used to add pens from other data sources.

Syntax

*object.***SetPenDataArray** *lNumPoints, pValue, pTime, pQuality*

Properties

The **SetPenDataArray** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
lNumPoints	Long. Number of points in the **Pen**.
pValue	Double array. An array of values for the points in the **Pen**.
pTime	Date array. An array of times for the points in the **Pen**.
pQuality	Long array. An array of qualities for the points in the **Pen**. Use OPC quality values.

SetPointAt Method

Modifies the location of the point at the given index to location specified.

Syntax

*object.***SetPointAt** *lIndex, pdispPoint*

Properties

The **SetPointAt** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
lIndex	Long. The position of the point to modify.
pdispPoint	Object. The position to which to move the point.

Remarks

A point is an OLE object specifying a point object to add to the list of existing data points. The point object has an (x, y) pair that contains the coordinate of the data point (see FixFloatPoint.)

SetPriorityColor Method

Sets the row background color to display for the alarms with the specified alarm priority.

NOTE: This method has been deprecated and replaced with the ."SetAlarmBackgroundColor Method " on page 484

Syntax

*object.***SetPriorityColor** *PriorityId, PriorityColor*

Properties

The **SetPriorityColor** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
PriorityId	Integer. The ID corresponding to the alarm priority. 0 - CRITICAL 1 - HIHI 2 - HIGH 3 - MEDIUM 4 - LOW 5 - LOLO 6 - INFO (INFORMATIONAL)
PriorityColor	OLE_COLOR. The row background color to display for the alarms with the alarm priority.

SetProperty Method

Sets the specified property to the specified value for the given object.

Syntax

*object.***SetProperty** *bstrPropertyName, vaValue*

Properties

The **SetProperty** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrPropertyName	String. The name of the property to set.
aValue	Variant. The value to which to set the property.

Remarks

Calling iOval.SetProperty “ForegroundColor”, 255 is equivalent to executing iOval.ForegroundColor = 255.

SetScriptWindow Method

Instantiates the Visual Basic Editor for the specified event for the currently selected object.

Syntax

*object.***SetScriptWindow** *bCreateIfEmpty,bstrEventName*

Properties

The **SetScriptWindow** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bCreateIfEmpty	Boolean. If **True**, the default event procedure should be prototyped in VBA if there are no procedures present for this object.
bstrEventName	String. The name of the event procedure to display in the code window.

Remarks

SetScriptWindow is a Configuration environment method only.

SetSource Method

Sets up the source connection properties for the animation object. This method is used instead of directly setting the animation object's source property if additional properties have to be specified for the connection.

Syntax

*object.***SetSource** *bstrExpression, [bUseAnyway], [vaUpdateRate], [vaDeadband], [vaTolerance], [vaConnectionFlags*]

Properties

The **SetSource** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrExpression	String. The expression representing the source.
bUseAnyway	Boolean. (Optional) Specifies whether to use a connection if the source doesn't exist. Default = **False**.
aUpdateRate	Float. (Optional) The value at which to set the update rate for the connection.
aDeadband	Float. (Optional) The value at which to set the deadband for the connection.
aTolerance	Float. (Optional) The value at which to set the tolerance for the connection.
aConnectionFlags	Long. (Optional) Reserved Word.

Remarks

The *vaTolerance* parameter only applies in expressions and exact match tables.

SetStatusColor Method

Sets the row foreground color to display for the alarms with the specified status.

NOTE: This method has been deprecated and replaced with the ."SetAlarmForegroundColor Method " on page 485

Syntax

*object.***SetStatusColor** *StatusId, StatusColor*

Properties

The **SetStatusColor** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
StatusId	Integer. The ID corresponding to the alarm status. **Valid entries:** 0 – OK 1 – LOLO 2 – HIHI 3 – LO 4 – HI 5 – RATE 6 – COS 7 – CFN 8 – DEV 9 – FLT 10 – DSAB 11 – ERROR 12 – ANY 13 – NEW 14 – TIME 15 – IOF 16 – OCD 17 – UNDER 18 – OVER 19 – RANGE 20 – COMM 21 – DEVICE 22 – STATION 23 – ACCESS 24 – SQL LOGIN 25 – SQL CMD 26 – DAT MATCH 27 – FLD READ 28 – FLD WRITE 29 – NO DATA 30 – NO XDATA
StatusColor	OLE_COLOR. The color to display for the alarm status.

SetStatusFont Method

Sets the font for alarms with the specified status.

Syntax

object.**SetStatusFont** *nStatusID, lpszFaceName, bStrikeout, bUnderline, bBold, bItalic*

Properties

The **SetStatusFont** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
nStatusID	Integer. The status of the alarm. **Valid values are:** 0 – OK 1 – LOLO 2 – HIHI 3 – LO 4 – HI 5 – RATE 6 – COS 7 – CFN 8 – DEV 9 – FLT 10 – DSAB 11 – ERROR 12 – ANY 13 – NEW 14 – TIME 15 – IOF 16 – OCD 17 – UNDER 18 – OVER 19 – RANGE 20 – COMM 21 – DEVICE 22 – STATION 23 – ACCESS 24 – SQL LOGIN 25 – CMD 26 – DAT MATCH 27 – FLD READ 28 – FLD WRITE 29 – NO DATA 30 – NO XDATA
lpszFaceName	String. The font name to be displayed.
bStrikeout	Boolean. Specifies the Strikeout option for the text. If **True**, the text appears with a line through it.
bUnderline	String. The name of the event.
bBold	Boolean. Specifies whether the text is bold or not.

bItalic	Boolean. Specifies whether the text is italic or not.

Remarks

SetStatusFont is a Configuration environment method only.

SetStringFormat Method

Sets the raw formatting for a string value.

Syntax

object.**SetStringFormat** *[pFormat]*

Properties

The **SetStringFormat** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pFormat	String. The string to be used when formatting the object.

SetTabSelection Method

Specifies which tabs of the expression editor dialog are displayed.

Syntax

object.**SetTabSelection**(*TabIndex*)

Properties

The **SetTabSelection** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
TabIndex	Integer. The value that corresponds to which tabs are displayed.

Return Value

Boolean. Returns **True** if the operation was successful, **False** otherwise.

SetTimeBeforeNow Method

Sets the time for a Chart based on a value that is some time before the current time.

Syntax

*object.***SetTimeBeforeNow** *hours, minutes, seconds*

Properties

The **SetTimeBeforeNow** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
hours	Long. The number of hours before now to set the **Chart** to.
minutes	Long. The number of minutes before now to set the **Chart** to.
seconds	Long. The number of seconds before now to set the **Chart** to.

Remarks

Calling the **SetTimeBeforeNow** method with an hours parameter of 2, minutes parameter of 0 and seconds parameter of 0 causes the **Chart** to display data that occurred 2 hours before the current time. This eliminates the need to calculate the **Chart** time based on the current time.

SetTimeCursorTime Method

Sets the time for the Time Cursor position. The time of the Time Cursor is specified relative to a **Pen**, since pens can have different times on the same **Chart**.

Syntax

*object.***SetTimeCursorTime** *dt, lPenNum*

Properties

The **SetTimeCursorTime** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
dt	Date. The time at which to set the Time Cursor.
lPenNum	Long. The index of the pen in the pen array for which you want to set the Time Cursor.

Remarks

The time of the Time Cursor is specified relative to a **Pen**, since pens can have different times on the same **Chart**.

SetWindowLocation Method

Sets the window's size and location in terms of percentage of the screen.

Syntax

*object.***SetWindowLocation** *lfTopPct, lfLeftPct, lfHeightPct, lfWidthPct, [bRedraw], [bDesiredLocation], [bClampWindow]*

Properties

The **SetWindowLocation** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
LfTopPct	Double. The top percentage of the window relative to the WorkSpace client area.
LfLeftPct	Double. The left percentage of the window relative to the WorkSpace client area.
LfHeightPct	Double. The percentage of horizontal screen space.
LfWidthPct	Double. The percentage of vertical screen space.
bRedraw	Boolean. (Optional) **True** – Redraw the document. (default) **False** – Set the position without redrawing the document.
bDesiredLocation	Boolean. (Optional) **True** – Location is written to disk. **False** – Location is not written to disk. (default)
bClampWindow	Boolean. (Optional) **True** – Sizes the document to the size of the viewport. **False** – Does not resize the document. (default)

Remarks

The *lfTopPct* parameter sets the window's location relative to the WorkSpace's MDI Client area origin and the *lfLeftPct* parameter sets the window's location relative to the absolute screen resolution's origin.

Calling the **SetWindowLocation** will not alter the location of the document that is saved to disk. To alter the window location that is saved to disk, set the **WindowHeightPercentage**, **WindowLeftPercentage**, **WindowTopPercentage**, and **WindowWidthPercentage** properties.

ShelveAlarm Method

Use this method to shelve an alarm in the Alarm Summary object.

Syntax

object.**ShelveAlarm** *DataPoint, ShelveDurationNumber, [intErrorMode], [BsendMsg]*

Properties

The **ShelveAlarm** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
DataPoint	String. The tag name.

ShelveDurationNumber	Long.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
BsendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

Return Values

- Invalid shelve policy name
- Invalid shelve duration
- Alarm disabled
- Alarm shelving not enabled
- Alarm already shelved
- Security access error
- Unknown error

ShowAnimations Method

Opens the Animation dialog box for the currently selected object.

Syntax

*object.***ShowAnimations**

Properties

The **ShowAnimations** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ShowBrowseDialog Method

Opens the ExpressionEditor dialog box.

Syntax

*object.***ShowBrowseDialog**

Properties

The **ShowBrowseDialog** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ShowColorBox Method

Opens the color dialog box for the **Color Button** object.

Syntax

*object.***ShowColorBox**

Properties

The **ShowColorBox** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ShowColorSelection Method

Opens or closes the color selection dialog box for the currently selected object.

Syntax

*object.***ShowColorSelection** *bShow*

Properties

The **ShowColorSelection** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bShow	Boolean. **True** – Opens the dialog box. **False** – Closes the dialog box.

Remarks

ShowColorSelection is a Configuration environment method only.

ShowCustomPages Method

Displays custom pages associated with the object.

Syntax

*object.***ShowCustomPages**

Properties

The **ShowCustomPages** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ShowPipePreviewDialog Method

Displays the Modify Pipe Characteristics dialog box for the selected pipe object.

Syntax

*object.***ShowPipePreviewDialog**

Properties

The **ShowPipePreviewDialog** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ShowTaskWizard Method

Opens the WorkSpace's Task Wizard dialog box.

Syntax

*object.***ShowTaskWizard**

Properties

The **ShowTaskWizard** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ShowVBAProcedure Method

Creates a procedure to receive the focus in the code window by concatenating the script name of the object with the procedure name.

Syntax

*object.***ShowVBAProcedure** (*bstrProcName*, [*objObject*])

Properties

The **ShowVBAProcedure** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrProcName	String. The name of the procedure or event to be located in the VBA code window.
Object	Object. The name of the object to which the procedure or event is associated.

Remarks

If the object is omitted, the procedure name is used by itself. If the procedure is not found, the focus will be set to the top of the script window. For example, to set the code window to Rect2's Mouse Down event, you would make the following call:

ShowVBAProcedure("MouseDown", Rect2)

To find any subroutine within the picture's project, you would make the following call:

ShowVBAProcedure("MySubProcedure")

ShowVisualBasicEditor Method

Opens the WorkSpace's Visual Basic Editor.

Syntax

*object.***ShowVisualBasicEditor**

Properties

The **ShowVisualBasicEditor** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

SilenceAlarmHorn Method

Silences the alarm horn.

Syntax

AlarmHornSilence *([intErrorMode])*

Properties

The **SilenceAlarmHorn** method syntax has this part:

Part	Description
intErrMode	Integer. (Optional). The error mode. 0 (default) = Errors are displayed in the form of a message box. 1 = Errors are not handled so that they can be handled in the calling routine. 2 = Errors are dispatched to the alarm typers using SendOperatorMessage.

Remarks

The alarm horn must be enabled for this method to work. If you call this method and the alarm horn is disabled, no error will be reported.

SnapObjectsToGrid Method

Snaps the currently selected object to the grid. It is equivalent to selecting SnapObjectsToGrid from the Format menu.

Syntax

*object.***SnapObjectsToGrid**

Properties

The **SnapObjectsToGrid** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

SnapObjectsToGrid is a Configuration environment method only.

This method only works when the **GridEnabled** property of the **Picture** or **Dynamo Set** is set to **True**.

SpaceEvenly Method

Positions a group of selected objects so that the amount of horizontal or vertical space between them is equal.

Syntax

*object.***SpaceEvenly** *type*

Properties

The **SpaceEvenly** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
type	Integer. **Valid entries:** 0 – Horizontal 1 – Vertical

Remarks

SpaceEvenly is a Configuration environment method only.

StartEvent Method

Enables the **Event** object to fire its events when data changes.

Syntax

*object.***StartEvent**

Properties

The **StartEvent** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

StartTimer Method

Starts the **Timer** object.

Syntax

*object.***StartTimer**

Properties

The **StartTimer** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

StartTimer is a Run-time environment only method that will only start the **Timer** if its **TimerEnabled** property is set to **True**.

StickToCursor Method

Causes a shape object to stick to the cursor upon creation.

Syntax

*object.***StickToCursor**

Properties

The **StickToCursor** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

StickToCursor is a Configuration environment method only. It allows the user to have functionality for all shapes similar to that which occurs when the user selects CurrentTime from the Insert menu.

StopGlobalPlayBack Method

Stops the historical playback in the iFIX WorkSpace in run mode.

Syntax

*object.***StopGlobalPlayBack**

Properties

The **StopGlobalPlayBack** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

StopEvent Method

Disables the **Event** object from firing its events when data changes.

Syntax

*object.***StopEvent**

Properties

The **StopEvent** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

StopTimer Method

Stops the **Timer**.

Syntax

*object.***StopTimer**

Properties

The **StopTimer** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Stretch Method

Scales the shape according to the percentage of scale entered for the shape's **Height** and **Width**.

Syntax

*object.***Stretch** *fXPercentage, fYPercentage*

Properties

The **Stretch** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
fXPercentage	Double. The scale percentage to apply to the width.
fYPercentage	Double. The scale percentage to apply to the height.

SwitchLanguage Method

Changes the displayed text strings of the specified picture from one language to another.

The method *Object.SwitchLanguage*, uses the LanguageDesired property to switch languages.

The method *Object.SwitchLanguage (xxxx)*, uses the specified language to switch language.

Syntax

*object.***SwitchLanguage** [*LanguageDesired* as Long]

Properties

The **SwitchLanguage** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

LanguageDesired *Long* (Optional)	Default CA_Catalan CS_Czech DA_Danish DE_German EL_Greek EN_English ES_Spanish FI_Finnish HU_Hungarian IT_Italian JA_Japanese KO_Korean NL_Dutch NO_Norwegian PL_Polish RU_Russian SR_Cyrillic HR_Croatian SK_Slovak SV_Swedish TH_Thai TR_Turkish IN_Indonesian SL_Slovenian EU_Basque ZHTW_Chinese - Taiwan FR_French PTBR_Brazilian Portuguese PT_Portuguese ZHCH_Chinese PRC FRCA_French Canadian

SwitchMode Method

Sets the mode of the WorkSpace.

Syntax

object.**SwitchMode** *bMode*

Properties

The **SwitchMode** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bMode	Long. The mode to set the WorkSpace to.

	Valid entries: 1 – Configure 4 – Run

Remarks

If possible, always make this call the last line in your script. Note that when the **SwitchMode** method is used from anywhere other than a toolbar, the call must be the last line in the script. Otherwise, you may experience unexpected behavior when executing the script. Otherwise, you may experience unexpected behavior when executing the script.

If the **SwitchMode** method is made from a toolbar and is not the last line in your script, be certain that the operation is complete before the rest of the script continues to execute.

SynchronizeSecurity Method

Performs the entire security synchronization process based on the property values set before this method is called. This method returns no values.

Syntax

*object.***SynchronizeSecurity**

Properties

The **SynchronizeSecurity** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

The actions performed during the synchronization process include the following:

- Checking that iFIX is running.
- Validating user rights to run SecuritySynchronizer, either through the System Autologin user or the user currently logged in.
- Querying Windows security.
- Modifying the iFIX security configuration, if needed.
- Writing result values to the iFIX database at time of completions, if configured this way.

Depending on the size of your Windows and iFIX security configurations, this method may take longer to complete because it performs a great amount of work.

T

TagGroupSubstitution Method

Returns the substitution string supplied by the user in the tag group file.

Syntax

*object.***TagGroupSubstitution** *bstrTagGroupSymbol*

Properties

The **TagGroupSubstitution** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrTagGroupSymbol	String. The name of the tag group symbol.

Remarks

This method returns the tag group substitution associated with the passed tag group symbol, or an empty string if not found.

TagGroupValue Method

Returns the value of the tag group substitution string of the passed tag group symbol.

Syntax

*object.***TagGroupValue***bstrTagGroupSymbol*

Properties

The **TagGroupValue** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrTagGroupValue	String. The name of the tag group symbol.

Remarks

This method returns a variant containing the value of the item in the substitution string. The variant will be empty if the method fails.

U-Z

UIActivate Method

Sets the object in its custom active mode.

Syntax

*object.***UIActivate**

Properties

The **UIActivate** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

UIDeActivate Method

Sets the object in its custom inactive mode.

Syntax

*object.***UIDeActivate**

Properties

The **UIDeActivate** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Undo Method

Reverses the last action completed in a picture. It is the equivalent of selecting Undo on the Edit menu.

Syntax

*object.***Undo**

Properties

The **Undo** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

Undo is a Configuration environment method only.

UndoTransaction Method

Allows a user to start an undo transaction (a series of nested undoable operations)

Syntax

*object.***UndoTransaction** *evUndoTransactionFlag*

Properties

The **UndoTransaction** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
EvUndoTransactionFlag	Enumeration. *peTransactionFlags*.

Settings

The settings for *peTransactionFlags* are:

Constant	Value	Description
pUndoTransactionStart	1	Start the transaction.
PUndoTransactionEnd	2	End the transaction.

UndoZoom Method

Reverses all of the zooming that the user has done in run mode and reverts the Enhanced Chart to its default coordinates. It is the equivalent of selecting Undo Zoom on the Chart Options right-mouse menu in run mode.

Syntax

*object.***UndoZoom**

Properties

The **UndoZoom** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

UndoZoom can be used whether the chart is Modifiable or not.

UnGroup Method

Disbands the currently selected Group object. It is the equivalent of selecting Ungroup on the Format menu.

Syntax

*object.***UnGroup**

Properties

The **UnGroup** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

When you disband a **Group**, any changes you made to the **Group** that affected a member object are retained by that object. In other words, ungrouping does not return the member objects back to the original state of the objects before you grouped them.

For example, suppose you have a grouped object in your picture that consists of red and black squares. When you change the **Group** color to black, all the squares turn black. If you ungroup the object after making this change, the squares remain black.

UnloadTagGroupFile Method

Unloads a tag group file from the picture.

Syntax

*object.***UnloadTagGroupFile**

Properties

The **UnloadTagGroupFile** method syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

If a tag group file is currently loaded, disconnects all tag group objects from their data sources and sends a message to release historical pens.

UnShelveAlarm Method

Use this method to unshelve an alarm in the Alarm Summary object.

Syntax

object.**UnShelveAlarm***DataPoint, intErrorMode], [BsendMsg]*

Properties

The **UnShelveAlarm** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

DataPoint	String. The tag name.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
BsendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

Return Values

- Alarm not shelved
- Security access error
- Unknown error

Update_A_Dynamo_By_Name Method

Updates a Dynamo by using the fully qualified name of the Master Dynamo and its instance.

Syntax

object.**Update_A_Dynamo_By_Name** *(bstrMasterDynamoName, bstrDynamoInstanceName, nDataSourceMismatchOption, bstrChoiceDialogTitle, pnResultCode)*

Properties

The **Update_A_Dynamo_By_Name** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrMasterDynamoName	String. The fully qualified name of the Master Dynamo.
bstrDynamoInstanceName	String. The fully qualified name of the Dynamo instance.
nDataSourceMismatchOption	Integer. The Dynamo mismatch option code: 0 – Update, but do not apply data sources 1 – Update and attempt to match data sources 2 – Do not update 3 – Prompt for choice
bstrChoiceDialogTitle	String. If Prompt for choice (3) is entered for the *nDataSourceMismatchOption* and a mismatch occurs, the *bstrChoiceDialogTitle* is the caption in the title bar of the Prompt for choice dialog box.
pnResultCode	The result code: 0 – Success

	1 – All data sources ignored 2 – Some data sources ignored 3 – Not enough data sources 101 – Dynamo not updated 201 – Cancelled 202 – Failure 203 – Invalid argument Dynamo Instance 204 – Invalid argument Dynamo Master 205 – Invalid mismatch option argument 206 – Invalid Pointer 207 – Instance does not match Master 208 – An owner of either the Instance or Master is NULL

Update_A_Dynamo_By_Name2 Method

Updates a Dynamo by using the fully qualified name of the Master Dynamo and its instance.

Syntax

object.**Update_A_Dynamo_By_Name2** *(bstrMasterDynamoName, bstrDynamoInstanceName, nDataSourceMismatchOption, bstrChoiceDialogTitle, pnResultCode)*

Properties

The **Update_A_Dynamo_By_Name2** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrMasterDynamoName	String. The fully qualified name of the Master Dynamo.
bstrDynamoInstanceName	String. The fully qualified name of the Dynamo instance.
nDataSourceMismatchOption	Integer. The Dynamo mismatch option as a bitmask: UPDATE_OPTION_ON_MISMATCH_UPDATE_DYNAMO (0x00000001) – When a mismatch is encountered, use the Update / Do not update options. This setting is unused when iFIX finds no mismatches. UPDATE_OPTION_ON_MISMATCH_APPLY_DATA_SOURCES (0x00000002) – When a mismatch is encountered, use the Apply Data Sources / Do not apply data sources options. This setting is unused when iFIX finds no mismatches. UPDATE_OPTION_ON_MISMATCH_PROMT_FOR_CHOICE (0x00000004) – When a mismatch is encountered, ask the user what to do. UPDATE_OPTION_RESIZE_INSTANCE (0x00000008) – Set to True to resize the Dynamo instance to match the Master Dynamo dimensions. Equivalent setting in iFIX 4.5 is always True.

	UPDATE_OPTION_SAVE_CAPTIONS (0x00000010) – Set to True to save the captions on text objects and button objects. UPDATE_OPTION_UPDATE_ON_CONVERSION (0x00000020) – Set to True to ignore the Dynamo_ID and Revision checking when updating. **NOTE:** You can add options like UPDATE_OPTION_ON_MISMATCH_UPDATE_DYNAMO + UPDATE_OPTION_ON_MISMATCH_APPLY_DATA_SOURCES + UPDATE_OPTION_RESIZE_INSTANCE + UPDATE_OPTION_SAVE_CAPTIONS, like this: 0x00000001 + 0x00000002 + 0x00000008 + 0x00000010. Option values express bit by hex, so a decimal expression will be 1 + 2 + 8 + 16.
bstrChoiceDialogTitle	String. If Prompt for choice (0x00000004) is entered for the *nDataSourceMismatchOption* and a mismatch occurs, the *bstrChoiceDialogTitle* is the caption in the title bar of the Prompt for choice dialog box.
pnResultCode	The result code as a bitmask: UPDATER_RESULT_SUCCESS_BIT (0x00000001) – Returns 1 on success, or 0 on failure. UPDATER_RESULT_ALL_DATA_SOURCES_IGNORED_BIT (0x00000002) – Encoded status bit. UPDATER_RESULT_SOME_DATA_SOURCES_IGNORED_BIT (0x00000004) – Encoded status bit. UPDATER_RESULT_NOT_ENOUGH_DATA_SOURCES_BIT (0x00000008) – Encoded status bit. UPDATER_RESULT_DYNAMO_NOT_UPDATED_BIT (0x00000010) – Encoded status bit. UPDATER_RESULT_USER_CANCELLED_BIT (0x00000020) – Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_DYNAMO_INSTANCE_BIT (0x00000040) – Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_MASTER_DYNAMO_BIT (0x00000080) – Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_MISTMATCH_OPTION_BIT (0x00000100) – Spare entry, use as needed. UPDATER_RESULT_INVALID_POINTER_BIT (0x00000200) – Spare entry, use as needed. UPDATER_RESULT_INSTANCE_DOESNT_MATCH_MASTER_BIT (0x00000400)– Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_OWNER_BIT (0x00000800) – Spare entry, use as needed.

	UPDATER_RESULT_TEXT_CAPTIONS_UPDATED_BIT (0x00001000) – Returns 1 if any text captions were updated, or 0 if none were updated. UPDATER_RESULT_TEXT_CAPTIONS_ALL_UPDATED_BIT (0x00002000) – Returns 1 if all captions were updated, or 0 if some (or none) were updated. This field must be 0 if the UPDATER_RESULT_TEXT_CAPTIONS_UPDATED_BIT is also 0.

Update_A_Dynamo_By_Ref Method

Updates a Dynamo by using a reference from the Master Dynamo and its instance.

Syntax

object.**Update_A_Dynamo_By_Ref** *(pIMasterDynamo, pIDynamoInstance, nDataSourceMismatchOption, bstrChoiceDialogTitle, pnResultCode)*

Properties

The **Update_A_Dynamo_By_Ref** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
pIMasterDynamo	String. The fully qualified name of the Master Dynamo.
pIDynamoInstance	String. The fully qualified name of the Dynamo instance.
nDataSourceMismatchOption	Integer. The Dynamo mismatch option code: 0 – Update, but do not apply data sources 1 – Update and attempt to match data sources 2 – Do not update 3 – Prompt for choice
bstrChoiceDialogTitle	String. If Prompt for choice (3) is entered for the *nDataSourceMismatchOption* and a mismatch occurs, the *bstrChoiceDialogTitle* is the caption in the title bar of the Prompt for choice dialog box.
pnResultCode	The result code: 0 – Success 1 – All data sources ignored 2 – Some data sources ignored 3 – Not enough data sources 101 – Dynamo not updated 201 – Cancelled 202 – Failure 203 – Invalid argument Dynamo Instance 204 – Invalid argument Dynamo Master 205 – Invalid mismatch option argument 206 – Invalid Pointer 207 – Instance does not match Master

	208 – An owner of either the Instance or Master is NULL

Update_A_Dynamo_By_Ref2 Method

Updates a Dynamo by using a reference from the Master Dynamo and its instance.

Syntax

object.**Update_A_Dynamo_By_Ref2** *(plMasterDynamo, plDynamoInstance, nDataSourceMismatchOption, bstrChoiceDialogTitle, pnResultCode)*

Properties

The **Update_A_Dynamo_By_Ref2** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
plMasterDynamo	String. The fully qualified name of the Master Dynamo.
plDynamoInstance	String. The fully qualified name of the Dynamo instance.
nDataSourceMismatchOption	Integer. The Dynamo mismatch option as a bitmask: UPDATE_OPTION_ON_MISMATCH_UPDATE_DYNAMO (0x00000001) – When a mismatch is encountered, use the Update / Do not update options. This setting is unused when iFIX finds no mismatches. UPDATE_OPTION_ON_MISMATCH_APPLY_DATA_SOURCES (0x00000002) – When a mismatch is encountered, use the Apply Data Sources / Do not apply data sources options. This setting is unused when iFIX finds no mismatches. UPDATE_OPTION_ON_MISMATCH_PROMT_FOR_CHOICE (0x00000004) – When a mismatch is encountered, ask the user what to do. UPDATE_OPTION_RESIZE_INSTANCE (0x00000008) – Set to True to resize the Dynamo instance to match the Master Dynamo dimensions. Equivalent setting in iFIX 4.5 is always True. UPDATE_OPTION_SAVE_CAPTIONS (0x00000010) – Set to True to save the captions on text objects and button objects. UPDATE_OPTION_UPDATE_ON_CONVERSION (0x00000020) – Set to True to ignore the Dynamo_ID and Revision checking when updating. **NOTE:** You can add options like UPDATE_OPTION_ON_MISMATCH_UPDATE_DYNAMO + UPDATE_OPTION_ON_MISMATCH_APPLY_DATA_SOURCES + UPDATE_OPTION_RESIZE_INSTANCE + UPDATE_OPTION_SAVE_CAPTIONS, like this: 0x00000001 + 0x00000002 + 0x00000008 + 0x00000010. Option

	values express bit by hex, so a decimal expression will be 1 + 2 + 8 + 16.
bstrChoiceDialogTitle	String. If Prompt for choice (0x00000004) is entered for the *nDataSourceMismatchOption* and a mismatch occurs, the *bstrChoiceDialogTitle* is the caption in the title bar of the Prompt for choice dialog box.
pnResultCode	The result code as a bitmask: UPDATER_RESULT_SUCCESS_BIT (0x00000001) – Returns 1 on success, or 0 on failure. UPDATER_RESULT_ALL_DATA_SOURCES_IGNORED_BIT (0x00000002) – Encoded status bit. UPDATER_RESULT_SOME_DATA_SOURCES_IGNORED_BIT (0x00000004) – Encoded status bit. UPDATER_RESULT_NOT_ENOUGH_DATA_SOURCES_BIT (0x00000008) – Encoded status bit. UPDATER_RESULT_DYNAMO_NOT_UPDATED_BIT (0x00000010) – Encoded status bit. UPDATER_RESULT_USER_CANCELLED_BIT (0x00000020) – Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_DYNAMO_INSTANCE_BIT (0x00000040) – Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_MASTER_DYNAMO_BIT (0x00000080) – Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_MISTMATCH_OPTION_BIT (0x00000100) – Spare entry, use as needed. UPDATER_RESULT_INVALID_POINTER_BIT (0x00000200) – Spare entry, use as needed. UPDATER_RESULT_INSTANCE_DOESNT_MATCH_MASTER_BIT (0x00000400)– Spare entry, use as needed. UPDATER_RESULT_INVALID_ARG_OWNER_BIT (0x00000800) – Spare entry, use as needed. UPDATER_RESULT_TEXT_CAPTIONS_UPDATED_BIT (0x00001000) – Returns 1 if any text captions were updated, or 0 if none were updated. UPDATER_RESULT_TEXT_CAPTIONS_ALL_UPDATED_BIT (0x00002000) – Returns 1 if all captions were updated, or 0 if some (or none) were updated. This field must be 0 if the UPDATER_RESULT_TEXT_CAPTIONS_UPDATED_BIT is also 0.

UpdateBackgroundObject Method

Forces changes made to an object in the foreground to be transferred to a Schedule running in the background **FixBackgroundServer** application.

Syntax

*object.***UpdateBackgroundObject** *bstrObjectName, [iUpdateMode], [iRunMode]*

Properties

The **UpdateBackgroundObject** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrObjectName	String. The name of the object in the schedule to be updated in the background task.
iUpdateMode	Long. **Valid entries:** 1 – Add (default) 2 – Modify 3 – Delete
iRunMode	Long. **Valid entries:** 1 – Run (default) 2 – Stop

UpdateConnectionParameters Method

Updates the refresh rate, deadband, and tolerance for an object's connection.

Syntax

*object.***UpdateConnectionParameters** *bstrPropertyName, [vaUpdateRate], [vaDeadband], [vaTolerance], [vaConnectionFlags]*

Properties

The **UpdateConnectionParameters** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
BstrPropertyName	String. The name of the connected property.
aUpdateRate	Variant. (Optional) The value at which to set the refresh rate for the connection.
aDeadband	Variant. (Optional) The value at which to set the deadband for the connection.
aTolerance	Variant. (Optional) The value at which to set the tolerance for the connection.
aConnectionFlags	Variant. (Optional) Reserved Word.

Remarks

Use this method with object to object connections only.

UpdateDefinition Method

Modifies existing definitions, appends new definitions to a tag group file, or creates new tag group files. Before using this method, use **RetrieveDefinition** method to read in the definitions you want to modify or append to.

You should not delete definitions by omitting them from an update. Instead, to delete a tag group definition, you delete the tag group file and then add all the definitions you want to keep.

Syntax

object.**UpdateDefinition***TagGroupName*, *Count*, *TokenList*, *ReplacementList*, *DescriptionList*

Properties

The **UpdateDefinition** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
TagGroupName	String. The name of the tag group file in the Picture path to update.
Count	Short. The number of symbols in the tag group file.
TokenList	Variant. The array of symbols. The array index ranges from 0 to Count-1.
ReplacementList	Variant. The array of substitutions. The array index ranges from 0 to Count-1.
DescriptionList	Variant. The array of descriptions associated with each substitution. The array index ranges from 0 to Count-1.

Remarks

Use the string Trim methods to ensure that there is no whitespace in either the TokenList entries or the Replacement list entries in the UpdateDefinition Tag Group File.

Make sure that Count correctly indicates the size of the array you pass into UpdateDefinition. If it is not correct, you can lose substitutions or you can get a crash.

UserFormPointToLogical Method

Converts “UserForm Point” coordinates to coordinates in logical units or postscript points.. “UserForm Point” coordinates are the measure for position VBA user forms on screen.

Syntax

*object.***UserFormPointToLogical** *plfTop, plfLeft*

Properties

The **UserFormPointToLogical** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
plfTop	Double. Specifies the userform point top coordinate and returns the converted logical unit or postscript point.

plfLeft	Double. Specifies the userform point left coordinate and returns the converted logical unit or postscript point.

Remarks

Prior to performing the conversion, the **StartUpPosition** property of the form should be changed from *CenterOwner* to either *Manual* or *WindowsDefault*. A setting of *CenterOwner* will result in the form being popped up in the middle of the picture window.

ValidateSignature Method

Performs validation of a user name and password. Typically, you call this method when you want to gather signature information through your application, but still use the ESignature object to validate the signature and perform security checks.

Syntax

object.**ValidateSignature***(bstrUsername, bstrPassword, nSigType, pbValidSig, bstrUserID, [bCheckTag])*

Properties

The **ValidateSignature** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrUsername	String. User name to be validated.
bstrPassword	String. Password to be validated.
nSigType	Integer. Indicates the type of signature to be validated. Valid values include: 1. – (PERFORM). Signature is for an action being performed. 2. – (VERIFY). Signature is for an action being verified.
pbValidSig	Boolean. Returns **True** if signature is valid, **False** if not.
bstrUserID	String. Returns the user ID of the user. This ID is used when sending a signed operator message.
bCheckTag	Boolean. (Optional). Indicates whether the user has access to security assigned to the tag. The default value is **False**. If this parameter is **True**, you must call the Initialize() method prior to calling this method. If you do not call Initialize(), or the tag is not a FIX32 data source, the method fails and an error generates.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

ValidateSignatureAndWriteValue Method

Performs validation of both the Performed By and Verified By user names and passwords. Based on the value of the nAction parameter, the **ValidateSignatureAndWriteValue** method writes the value passed in to the tag or acknowledges the alarm or list of alarms, and sends a signed operator message to the alarm system. This method can only write to FIX32 data sources.

Typically, you call this method when you want to gather signature information through the application, but still want to use the ESignature object to perform signature validation, write to FIX32 data source(s), and send the signed operator message.

You must call the Initialize() method prior to calling this method if you are writing values to the database or acknowledging alarms, otherwise the function fails and an error is returned.

Syntax

object.**ValidateSignatureAndWriteValue***(nAction, pValue, bstrPerformUsername, bstrPerformPassword, [bstrPerformComment], [bstrVerifyUsername], [bstrVerifyPassword], [bstrVerifyComment])*

Properties

The **ValidateSignatureAndWriteValue** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
nAction	Integer. Indicates the type of action associated with this signature. Valid entries include: 0 – (WRITE_VAL). Writes a value to a single data source. 3 – (ACK_OR_REMOVE). Acknowledges a single alarm. 4 – (ACK_OR_REMOVE_LIST). Acknowledges multiple alarms.
pValue	Variant. Value to be written to the database.
bstrPerformUsername	String. Performed By user name to be validated.
bstrPerformPassword	String. Performed By password to be validated.
bstrPerformComment	String. (Optional). The performed by comment to be sent with the signed operator message.
bstrVerifyUsername	String. (Optional). Verified By user name to be validated.
bstrVerifyPassword	String. (Optional). Verified By password to be validated.
bstrVerifyComment	String. (Optional). Verified By comment to be sent with the signed operator message.

Return Value

This method returns HRESULT. If the HRESULT is a value other than S_OK, VBA generates an error. You can handle this error using the On Error Statement You can find out more information about the error by using Err Object.

ValidateSource Method

Validates the specified data source.

Syntax

*object.***ValidateSource** *bstrObject, iStatus, ppdispObject, bstrPropertyName*

Properties

The **ValidateSource** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bstrObject	String. The string containing the source to validate.
iStatus	Long. Returns the status for the validity of the source. Return values are: 0 – OK 1 – Syntax error 2 – Data Undefined 3 – Data type mismatch
ppdispObject	Object. Returns the dispatch pointer to the source object if it exists.
bstrPropertyName	String. Returns the name of the actual property of the data source.

ValueTimeFromXY Method

Gets information for a **Pen** based on the X and Y coordinates specified.

Syntax

*object.***ValueTimeFromXY** *x, y, pfVal, pdt, bReal*

Properties

The **ValueTimeFromXY** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
x	Double. The x coordinate.
y	Double. The y coordinate.
pfVal	Double. Returns the value for the **Pen** at the specified x and y coordinates.

pdt	Date. Returns the time information for the **Pen** at the specified x and y coordinates.
bReal	Boolean. Returns **True** if the **(x,y)** point was on a real point and **False** if the **(x,y)** point was on an interpolated point.

Write Method

Writes a value to the data source represented by the DataItem or Group (DataSystem).

DataItem Object Syntax

*object.***Write** *Value*

Properties

The **Write** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Value	Value to be written to the data system.

Group (DataSystem) Object Syntax

*object.***Write**

Properties

The **Write** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

XYFromValueTime Method

Gets the X and Y coordinates for a Pen based on the time and value specified.

Syntax

*object.***XYFromValueTime** *fVal, dt, px, py*

Properties

The **XYFromValueTime** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
fVal	Double. The value for the **Pen**.
dt	Date. The time for the **Pen**.

px	Double. Returns the x coordinate.
py	Double. Returns the y coordinate.

XYHitTest Method

Returns information for a **Pen** based on the specified coordinates. Typically used with the **MouseUp** and **MouseDown** events.

Syntax

*object.***XYHitTest** *lfX, lfY, pDt, pfV, pszPenName, ppPen, lPenNum, pbReal*

Properties

The **XYHitTest** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
x	Double. The X coordinate where the mouse is clicked.
y	Double. The Y coordinate where the mouse is clicked.
pDt	Date. Returns the date for the **Pen** that corresponds to the specified X and Y coordinates.
pfV	Double. Returns the value for the **Pen** that corresponds to the specified X and Y coordinates.
pszPenName	String. Returns the name of the **Pen** that corresponds to the specified X and Y coordinates.
ppPen	Object. Returns the **Pen** object that corresponds to the specified X and Y coordinates.
lPenNum	Long. Returns the index in the **Pens** collection for the **Pen** that corresponds to the specified X and Y coordinates.
pbReal	Boolean. Returns **True** if the hit test was on a real point and **False** if it was on an interpolated point.

Remarks

This method is useful for performing operations based on **Pen** selection, or for creating data annotations. It returns the time and date of the clicked point, the value on the line, the pen's data source, a pointer to the **Pen**, the pen number, and whether the user clicked on a real or interpolated data point.

Zoom Method

Zooms in on an exact position.

Syntax

*object.***Zoom** *fYHi, fYLo, fXHi, fXLo*

Properties

The **Zoom** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
fYHi	Single. The High vertical percentage value.
fYLo	Single. The Low vertical percentage value.
fXHi	Single. The High horizontal percentage value.
fXLo	Single. The Low horizontal percentage value.

ZoomToFit Method

Enables or disables the "Zoom to Fit" feature on a picture using Enhanced Coordinates.

Syntax

object.**ZoomToFit** *bRedraw*

Properties

The **ZoomToFit** method syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
bRedraw	Boolean. If True, the picture zooms to fit the Workspace screen. If False, the picture is not zoomed.

Remarks

ZoomToFit is a Run mode environment method only.

Event Summary

The following list contains the iFIX object events that are available to the Automation Interface. For information on non iFIX events, refer to the appropriate help system.

A-B

Activated

AfterKillFocus

AlarmAck

AlarmAcknowledged

AlarmListChanged

C

Click

Close

ColorChanged

D

DataChange

DblClick

DeActivated

E-H

Edit

EditChange

I-J

Initialize

InitializeConfigure

K

KeyDown

KeyUp

L

LMouseClick

LoadedTagGroup

M-N

MouseDown

MouseMove

MouseUp

MouseUpOffObject

NewAlarm

O

OnChange

OnChartFull

OnChartRefresh

OnFalse

OnPenSelect

OnTimeOut

OnTrue

P-R

RMouseClick

S-V

SelectionChanged

SeverityIncreased

UIDeactivate

W-Z

WhileFalse

WhileTrue

A-D

Activated Event

Occurs when a window gets the user focus. This is defined as the moment when the user's keyboard and mouse focus are directed to that window. The user can tell this is happening by looking at the titlebar (if the document has one).

Syntax

*object*_**Activated( )**

Properties

The **Activated** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

AfterKillFocus Event

Occurs when focus is taken away from the specified **ExpressionEditor** control.

Syntax

*object*_**AfterKillFocus( )**

Properties

The **AfterKillFocus** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

AlarmAck Event

Occurs when an alarm is acknowledged.

Syntax

*object*_**AlarmAck( )**

Properties

The **AlarmAck** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

AlarmAcknowledged Event

Occurs when an alarm is acknowledged by double-clicking an alarm or calling the **AckAlarm**, **AckAlarmPage**, or the **AckAllAlarms** methods. Unlike the AlarmAck event, the AlarmAcknowledged event returns the name of the node, tag, and field that was acknowledged.

Syntax

*object*_**AlarmAcknowledged(***strNode* **As String,** *strTag* **As String,** *strField* **As String)**

Properties

The **AlarmAcknowledged** event syntax has these parts:

Part	Description
Object	An object expression that evaluates to an object in the Applies To list.
strNode	String. The name of the node on which the alarm was acknowledged.
StrTag	String. The name of the tag whose alarm was acknowledged.
StrField	String. The name of the field whose alarm was acknowledged.

Remarks

If you acknowledge a page of alarms, you receive one event for each alarm. If you acknowledge all alarms, the event does not occur.

AlarmListChanged Event

Occurs when the contents or order of information in the **Alarm Summary** object changes. By default, this event does not fire. To trigger the event when information in the **Alarm Summary** object changes, set the **CheckForAlarmListChanged** property to TRUE.

Syntax

*object*_**AlarmListChanged()**

Properties

The **AlarmListChanged** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

This event occurs on the initial receipt of alarms by the **Alarm Summary** object. The event occurs subsequently whenever the contents or order of alarms displayed by the Alarm Summary object changes. For example, the event may also occur when you change the alarm filter if the modified filter changes the content of the object's spreadsheet (for example, if it filters out alarms or adds alarms from another alarm area). The event occurs even if the change in content happens off screen. The following actions may also cause the **AlarmListChanged** event to fire but only if the action changes the content of the object's spreadsheet:

- Changing the sort order.
- Detecting/receiving a new alarm.

You can disable the **AlarmListChanged** event by setting the **CheckForAlarmListChanged** property to FALSE.

Click Event

Occurs when the user releases the left mouse key in the Run-time environment.

Syntax

*object*_**Click( )**

Properties

The **Click** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

This event is the common event used to initiate an operator action. It is also the default event used by the script authoring wizards.

The sequence of mouse-related events is:

- **MouseDown**
- **MouseUp**
- **Click**
- **DblClick**

Close Event

Occurs when a page container is shut down or closed in the Run-time environment.

Syntax

*object*_**Close( )**

Properties

The **Close** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

ColorChanged Event

Occurs when a new color is selected in the Run-time environment.

Syntax

*object*_**ColorChanged(ByVal** *Color* **As Long)**

Properties

The **ColorChanged** event syntax has these parts:

Part	Description
Object	An object expression that evaluates to an object in the Applies To list.
Color	The newly selected color.

DataChange Event

Occurs when a data source's value changes by more than the deadband limit or if the quality or error information associated with the data source changes.

The **DataChange** event occurs at the end of data change processing, therefore, the animation or event object's internal state (properties) will reflect this change. The previous value is not stored in the object.

If the same events occur while the script is executing, the system queues one and only one event for this situation. That is, if multiple data changes occur, the next event firing reflects the newest information.

Syntax

*object*_**DataChange(ByVal** *DataValue* **As Variant, ByVal** *TimeStamp* **as Date, ByVal** *Transition* **As Long, ByVal** *Reserved* **As Variant)**

Properties

The **DataChange** event syntax has these parts:

Part	Description
Object	An object expression that evaluates to an object in the Applies To list.
DataValue	The new data value.
TimeStamp	The time that this data arrived in the system.
Transition	Reserved, always 0.
Reserved	Reserved.

Event Firing Definition

Value Transition	Fire Event Sequence
Uninitialized to **True**	DataChange, OnTrue, WhileTrue
True to **False**	DataChange, OnFalse, WhileFalse
False to **True**	DataChange, OnTrue, WhileTrue

Uninitialized to Error	DataChange
Uninitialized to **False**	DataChange, **OnFalse**,**WhileFalse**

When data changes from an unknown state to a known state, the DataChange event triggers. Therefore, actions such as switching from the Configuration environment to the Run-time environment while an iFIX schedule is open will cause the DataChange event to trigger accordingly.

Using the DataChange Event in a Datalink

If you are attempting to use the DataChange Event in a data link, you cannot create the VBA object by selecting Edit Script from the right-click menu.

▶ To create your own object:

1. Create a data link.
2. Open the VBA editor.
3. Enter the Private Sub *object*_DataChange(ByVal *DataValue* As Variant, ByVal *TimeStamp* as Date, ByVal *Transition* As Long, ByVal *Reserved* As Variant) string and press Enter.
4. Enter any desired code.
5. Enter End Sub at the end of the subroutine.

DblClick Event

Occurs when the user double-clicks the mouse.

Syntax

*object*_**DblClick( )**

Properties

The **DblClick** event syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

The **DblClick** event occurs when multiple left mouse clicks are received in the object.

The sequence of mouse-related events is:

- **MouseDown**
- **MouseUp**
- **Click**
- **DblClick**

Alarm Summary Syntax

*object*_**DblClick(ByVal** *Col* **As Long, ByVal** *Row* **As Long)**

Properties

The **DblClick** event syntax has these parts:

Part	Description
Object	An object expression that evaluates to an object in the Applies To list.
Col	The column in which the user double clicked.
Row	The row in which the user double clicked.

DeActivated Event

Occurs when a window loses the focus. This is fired when another document receives the focus.

Syntax

*object*_**DeActivated( )**

Properties

The **DeActivated** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

When a picture is closed, the **DeActivated** event does not fire.

E-N

Edit Event

Occurs when:

- The user double clicks an object in the Configuration environment.
- The user pastes an object into a page from a **DynamoSet**.

Syntax

*object*_**Edit( )**

Properties

The **Edit** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

This method is the primary mechanism for creating a custom property page for an object that is in the form of a VBA custom form, which is useful for creating Dynamo objects. When an object is modified, the order of execution is:

1. If the object has an edit event, it is fired.
2. If the object has a custom property page (ocxes, chart objects, alarm summary objects), it is fired.
3. Otherwise, the animation dialog box is displayed.

TIP: Put all substitution logic in the **Edit** event and pass all object context into a shared or global form. Object names are automatically modified on a duplicate if the reference is in the event handler. Hard-coded object references in user forms are not modified when an object is duplicated.

EditChange Event

Occurs when the text in the edit box portion of the ExpressionEditor is changed.

Syntax

*object*_**EditChange(ByVal** *bSourceEmpty* **As Boolean)**

Properties

The **EditChange** event syntax has these parts:

Part	Description
Object	An object expression that evaluates to an object in the Applies To list.
bSourceEmpty	Whether the edit box is empty or not. If **True**, the edit box is empty. If **False**, the edit box is not empty.

Initialize Event

Occurs when a document is opened in the Run-time environment.

Syntax

*object*_**Initialize( )**

Properties

The **Initialize** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

InitializeConfigure Event

Occurs when a document is opened in the Configuration environment, or when the user switches to the Configuration environment.

Syntax

*object*_**InitializeConfigure( )**

Properties

The **InitializeConfigure** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

KeyDown Event

Occurs when the user presses a key on the keyboard.

Syntax

*object*_**KeyDown(ByVal** *KeyCode* **As Long, ByVal** *Shift* **As Long, ContinueProcessing As Boolean)**

Properties

The **KeyDown** event syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
KeyCode	An integer that represents the key code of the key that was pressed or released.
Shift	The state of the SHIFT, CTRL, and ALT keys.

Settings

The settings for *Shift* are:

Value	Description
1	SHIFT was pressed.
2	CTRL was pressed.
4	ALT was pressed.

Remarks

Key events are sent to the selected object first. If there is no script tied to the event, the key event is sent to the contained object (group or page). If you have common key events across a set of pictures, it is recommended that you use shared subroutines in the user global page that are called from the picture's key events.

The ContinueProcessing As Boolean parameter is related to the Key Macro scheme. Since key macros can be tied to objects, there is a certain hierarchy. For example, you can have the same key macro run different scripts on an object and in the picture. The processing starts in the object, and the

ContinueProcessing parameter prevents the processing from being passed up to the next level. For example, if F10 runs MacroA on a rectangle, and F10 also runs MacroB in the picture, a false value would keep the picture's macro from firing.

KeyUp Event

Occurs when the user presses a key on the keyboard.

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Syntax

*object*_**KeyUp(ByVal** *KeyCode* **As Long, ByVal** *Shift* **As Long, ContinueProcessing As Boolean)**

Properties

The **KeyUp** event syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
KeyCode	An integer that represents the key code of the key that was pressed or released.
Shift	The state of the SHIFT, CTRL, and ALT keys.

Settings

The settings for *Shift* are:

Value	Description
1	SHIFT was pressed.
2	CTRL was pressed.
4	ALT was pressed.

Remarks

Key events are sent to the selected object first. If there is no script tied to the event, the key event is sent to the contained object (group or page). If you have common key events across a set of pictures, it is recommended that you use shared subroutines in the user global page that are called from the picture's key events.

The ContinueProcessing As Boolean parameter is related to the Key Macro scheme. Since key macros can be tied to objects, there is a certain hierarchy. For example, you can have the same key macro run different scripts on an object and in the picture. The processing starts in the object, and the ContinueProcessing parameter prevents the processing from being passed up to the next level. For example, if F10 runs MacroA on a rectangle, and F10 also runs MacroB in the picture, a false value would keep the picture's macro from firing.

LMouseClick Event

Occurs when the user clicks the left mouse button on the **Alarm Summary** object in either the Configuration or Run-time environment.

Syntax

*object*_**LMouseClick(ByVal** *Col* **As Long, ByVal** *Row* **As Long)**

Properties

The **LMouseClick** event syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Col	The column the user clicked on.
Row	The row the user clicked on.

LoadedTagGroup Event

Occurs when the user loads a tag group in the runtime environment.

Syntax

*object*_**LoadedTagGroup(***TagGroupName* **As String)**

Properties

The **LoadedTagGroup** event syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
TagGroupName	String. The name of the tag group.

MouseDown Event

Occurs when the user presses a mouse button.

Syntax

*object*_**MouseDown(ByVal** *Button* **As Integer, ByVal** *Shift* **As Integer, ByVal** *X* **As Double, byVal** *Y* **As Double)**

Properties

The **MouseDown** event syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Button	An integer value that identifies which mouse button was pressed.

Shift	The state of the SHIFT, CTRL, and ALT keys.
X	The horizontal position, in postscript points or logical units, from the left or top edge of the page where the mouse was pressed.
Y	The vertical position, in postscript points or logical units, from the left or top edge of the page where the mouse was pressed.

Settings

The settings for *Button* are:

Value	Description
1	The left button was pressed.
2	The right button was pressed.
4	The middle button was pressed.

The settings for *Shift* are:

Value	Description
1	SHIFT was pressed.
2	CTRL was pressed.
3	SHIFT and CTRL were pressed.
4	ALT was pressed.
5	ALT and SHIFT were pressed.
6	ALT and CTRL were pressed.
7	ALT, SHIFT, and CTRL were pressed.

Remarks

The sequence of mouse-related events is:

- MouseDown
- MouseUp
- Click
- DblClick

MouseDown or **MouseUp** event procedures specify actions that occur when a mouse button is pressed or released. They enable you to distinguish between the left, right, and middle mouse buttons. You can also write code for mouse-keyboard combinations that use the SHIFT, CTRL, and ALT keyboard modifiers.

Use the Shift argument to identify the state of the SHIFT, CTRL, and ALT keys when the **MouseDown** or **MouseUp** event occurred. For example, if both CTRL and ALT are pressed, the value of Shift is 6.

Mouse events are sent to the selected object first. If there is no event script tied to the selected object, then the event is sent to the object's container (group or page).

MouseMove Event

Occurs when the user moves the mouse over an object.

Syntax

*object*_**MouseMove(ByVal** *Button* **As Integer, ByVal** *Shift* **As Long, ByVal** *X* **As Double, ByVal** *Y* **As Double)**

Properties

The **MouseMove** event syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Button	An integer value that identifies which mouse button was pressed.
Shift	The state of the SHIFT, CTRL, and ALT keys.
X	The horizontal position, in postscript points or logical units, from the left or top edge of the page where the mouse was pressed.
Y	The vertical position, in postscript points or logical units, from the left or top edge of the page where the mouse was pressed.

Settings

The settings for *Button* are:

Value	Description
1	The left button was pressed.
2	The right button was pressed.
4	The middle button was pressed.

The settings for *Shift* are:

Value	Description
1	SHIFT was pressed.
2	CTRL was pressed.
3	SHIFT and CTRL were pressed.
4	ALT was pressed.
5	ALT and SHIFT were pressed.
6	ALT and CTRL were pressed.
7	ALT, SHIFT, and CTRL were pressed.

Remarks

Mouse events are sent to the selected object first. If there is no event script tied to the selected object, then the event is sent to the object's container (group or page).

MouseUp Event

Occurs when the user releases any of the mouse keys.

Syntax

*object*_**MouseUp(ByVal** *Button* **As Integer, ByVal** *Shift* **As Integer, ByVal** *X* **As Double, ByVal** *Y* **As Double)**

Properties

The **MouseUp** event syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Button	An integer value that identifies which mouse button was pressed.
Shift	The state of the SHIFT, CTRL, and ALT keys.
X	The horizontal position, in postscript points or logical units, from the left or top edge of the page where the mouse was pressed.
Y	The vertical position, in postscript points or logical units, from the left or top edge of the page where the mouse was pressed.

Settings

The settings for *Button* are:

Value	Description
1	The left button was pressed.
2	The right button was pressed.
4	The middle button was pressed.

The settings for *Shift* are:

Value	Description
1	SHIFT was pressed.
2	CTRL was pressed.
3	SHIFT and CTRL were pressed.
4	ALT was pressed.
5	ALT and SHIFT were pressed.
6	ALT and CTRL were pressed.
7	ALT, SHIFT, and CTRL were pressed.

Remarks

The sequence of mouse-related events is:

- MouseDown
- MouseUp
- Click
- DblClick

MouseDown or **MouseUp** event procedures specify actions that occur when a mouse button is pressed or released. They enable you to distinguish between the left, right, and middle mouse buttons. You can also write code for mouse-keyboard combinations that use the SHIFT, CTRL, and ALT keyboard modifiers.

Use the Shift argument to identify the state of the SHIFT, CTRL, and ALT keys when the **MouseDown** or **MouseUp** event occurred. For example, if both CTRL and ALT are pressed, the value of Shift is 6.

Mouse events are sent to the selected object first. If there is no event script tied to the selected object, then the event is sent to the object's container (group or page).

MouseUpOffObject Event

Occurs when the left mouse button is pressed on an object, the mouse cursor is moved off the object and then released to the UP position.

Syntax

*object*_**MouseUpOffObject()**

Properties

The **MouseUpOffObject** event syntax has one part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

The **MouseUpOffObject** event is only used with the left mouse button. This event will not apply to the right mouse button.

The **MouseUpOffObject** event can be enabled or disabled by the setting of the AllowMouseUpOffObjectEvent entry in the [AppRunPreferences] section of the FixUserPreferences.ini file. The values of the setting can be 1 for TRUE and 0 for FALSE. The default value is 1 (TRUE).

If the **MouseUpOffObject** event is enabled and there is **MouseUpOffObject** event script created for the object, then the **MouseUpOffObject** event is processed and the script in the event is executed. If **MouseUpOffObject** event is NOT enabled, then the **MouseUp** event is sent to the object's container (group or page).

NewAlarm Event

Occurs when a new alarm is detected after the initial receipt of alarms by the **Alarm Summary** object. By default, this event does not fire. To trigger the event when information in the **Alarm Summary** object changes, set the **CheckForNewAlarms** property to TRUE. The event occurs even if the change in content happens off screen.

For example, in run mode, if a tag goes into a HI alarm and then changes to HIHI, the NewAlarm event fires once, unless the previous HI alarm is acknowledged. The NewAlarm event detects if an alarm is new, not a new severity. If you want to detect a change in severity, use the **CheckForSeverityIncrease** property and **SeverityIncreased** event.

Syntax

*object*_**NewAlarm(***strNode* **As String,** *strTag* **As String)**

Properties

The **NewAlarm** event syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
strNode	String. A list of nodes with new alarms. Example of parsing the list
StrTag	String. A list of tags with new alarms. Example of parsing the list

Remarks

Whenever you change the filter or sort configuration, the **Alarm Summary** object updates its list of alarm. However, this change does not cause the **NewAlarm** event to occur.

If a tag is in alarm and another alarm occurs for this tag, the **NewAlarm** event does not trigger again. For example, if a tag is in HI alarm and it goes to HIHI alarm the **NewAlarm** event fires only once. In order for the event to trigger twice, the HI alarm would have to be acknowledged and return to an OK alarm state prior to the tag going into a HIHI alarm.

O-Z

OnChange Event

Occurs when the **CurrentValue** of the specified **Variable** object is changed.

Syntax

*object*_**OnChange( )**

Properties

The **OnChange** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

OnChartFull Event

Occurs when the data for a **Chart** object scrolls all the way to the right and hits the edge. This event can only occur if the **ScrollDirection** property of the **Chart** is set to *LeftToRight*.

Syntax

*object*_**OnChartFull( )**

Properties

The **OnChartFull** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

This event is useful for handling the appropriate UI action when the data fills up the **Chart**. Possible options are to clear the data and change the start time to now, effectively wiping out the **Chart**, or to scroll the **Chart** over 50%.

OnChartRefresh Event

Occurs when the **Chart** object's data automatically scrolls. The event occurs at an interval defined by the**RefreshRate**property.

Syntax

*object*_**OnChartRefresh( )**

Properties

The **OnChartRefresh** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

OnFalse Event

Occurs when an expression in an **Event** object changes from **True** to **False** (non-zero to zero).

Syntax

*object*_**OnFalse( )**

Properties

The **OnFalse** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

When using an expression such as AI1 > 55.0, the value of this expression is 0 when AI1 is less than or equal to 55.0. When AI1 exceeds 55.0 the value of the expression is 1. The **OnFalse** event is triggered when the value of the expression changes from 1 to 0. If the expression is a single tag, then the value of the tag is evaluated as either 0.0 or non-zero to determine whether to fire the event.

See the complete event firing event transition described in the **DataChange** event.

OnPenSelect Event

Occurs when the user changes the currently active **Pen** by selecting another **Pen**.

Syntax

*object*_**OnPenSelect(ByVal** *lPenNum* **As Long)**

Properties

The **OnPenSelect** event syntax has these parts:

Part	Description
Object	An object expression that evaluates to an object in the Applies To list.
lPenNum	The index in the **Pens** collection that defines the selected **Pen**.

OnTimeOut Event

Occurs depending on the **TriggerType** of the **Timer**:

OneShot - The event is fired at the **StartTime**.

Continuous - The event is fired at the **StartTime** and then at the interval specified for the **Interval** property.

Daily - The event is fired at the **StartTime** for those days that have been configured (see **DaysOfWeek**).

Monthly - The event is fired at the **StartTime** for every day that has been configured (see **DaysOfMonth**).

Syntax

*object*_**OnTimeOut(ByVal** *lTimerId* **As Long)**

Properties

The **OnTimeOut** event syntax has these parts:

Part	Description
Object	An object expression that evaluates to an object in the Applies To list.
lTimerId	Reserved.

OnTrue Event

Occurs when an expression in an Event object changes from **False** to **True** (zero to non-zero).

Syntax

*object*_**OnTrue( )**

Properties

The **OnTrue** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

When using an expression such as AI1 > 55.0, the value of this expression is 0 when AI1 is less than or equal to 55.0. When AI1 exceeds 55.0 the value of the expression is 1. The **OnTrue** event is triggered when the value of the expression changes from 0 to 1. If the expression is a single tag, then the value of the tag is evaluated as either 0.0 or non-zero to determine whether to fire the event.

See the complete event firing event transition described in the DataChange event.

RMouseClick Event

Occurs when the user clicks the right mouse button on the Alarm Summary object in either the Configuration or Run-time environment.

Syntax

*object*_**RMouseClick(ByVal** *Col* **As Long, ByVal** *Row* **As Long, ByVal** *x* **As Long, ByVal** *y* **As Long)**

Properties

The **RMouseClick** event syntax has these parts:

Part	Description
Object	An object expression that evaluates to an object in the Applies To list.
Col	The column the user clicked on.
Row	The row the user clicked on.
X	The horizontal location of the click in device coordinates.
Y	The vertical location of the click in device coordinates.

SelectionChanged Event

Reserved for internal purposes.

SeverityIncreased Event

Occurs when an alarm's status increases in severity. The CheckForSeverityIncrease property must set to TRUE to allow this event to be triggered. The **CheckForSeverityIncrease** property must be set in run mode. The value you enter here is not persisted. In other words, when you switch from run mode to configure mode, the value changes back to FALSE (0), which is the default. If you enter TRUE (1) in configure mode, it switches back to FALSE (0) when you enter run mode. You must set this value in run mode.

Syntax

*object*_**SeverityIncreased( )**

Properties

The **SeverityIncreased** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

UIDeactivate Event

Reserved for internal purposes.

WhileFalse Event

Occurs in the Event object, while the value is zero. This event is continually called back at an interval specified by the Interval property of the **Event** object.

Syntax

*object*_**WhileFalse( )**

Properties

The **WhileFalse** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

See the complete event firing event transition described in the DataChange event.

WhileTrue Event

Occurs in the **Event** object, while the value is non-zero. This event is continually called back at an interval specified by the **Interval** property of the **Event** object.

Syntax

*object*_**WhileTrue( )**

Properties

The **WhileTrue** event syntax has this part:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.

Remarks

See the complete event firing event transition described in the **DataChange** event.

Subroutine Summary

The following list contains the iFIX subroutines that are available to the Automation Interface. For information on non iFIX objects, refer to the appropriate help system.

A-B

AcknowledgeAllAlarms

AcknowledgeAnAlarm

AlarmHornEnabled

AlarmHornEnabledToggle

AlarmHornSilence

C

CloseDigitalPoint

ClosePicture

D

DisableAlarm

E

EnableAlarm

F-K

FetchLimits

FindDataSource

FindLocalObject

GeneratePicture

GetAllConnections

GetDecimalSeparator

GetFormDynamoColor

GetFormNumeric

GetFormPushbutton

GetFormRamp

GetFormSlider

GetLocaleInfoA

GetUserDefaultLCID

HandleError

IsUserFxg

L-N

LocateObject

LogIn

O

OffScan

OnScan

OpenDigitalPoint

OpenPicture

OpenTGDPicture

P-Q

PictureAlias

PrintReport

QuickAdd

R

RampValue

ReadValue

RegCloseKey

RegOpenKeyEx

ReplacePicture

ReplaceTGDPicture

S

SetAuto

SetManual

SetSymbolValues

ShellExecute

T

ToggleDigitalPoint

ToggleManual

[ToggleScan](#)

U-Z

[WriteValue](#)

A-F

AcknowledgeAllAlarms Subroutine

Acknowledges alarms for all tags in the specified **Picture**. If any of the alarms in the picture require an electronic signature, acknowledgement depends on the setting of the Unsigned Writes options in each alarm's block:

Accept Enabled – The subroutine will also acknowledge the alarm associated with this block.

Reject Enabled – The subroutine will acknowledge other alarms, but not the alarm associated with this block.

Syntax

AcknowledgeAllAlarms*[Picture], [intErrorMode]*

Properties

The **AcknowledgeAllAlarms** subroutine syntax has these parts:

Part	Description
Picture	String. (Optional) The file name of the **Picture** for which you want to acknowledge all alarms. If no picture is specified, the current picture is used.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.

Remarks

If the only object in a picture is an Alarm Summary object that has Allow Acknowledge All Alarms disabled, this subroutine will not acknowledge any alarms. The order of logic in the subroutine is as follows:

1. Look for an Alarm Summary Object in the picture.
2. Check for whether the Allow Acknowledge All Alarms option is enabled.

- If enabled, run the AckAllAlarms method.
- If disabled, go to next step.

3. Check the other objects in the picture.

If you are using the AcknowledgeAllAlarms subroutine on an Alarm Summary OCX, this subroutine checks to ensure that the Allow Acknowledge All Alarms property is enabled. If the property is disabled, no alarms associated with that Alarm Summary OCX are acknowledged.

AcknowledgeAnAlarm Subroutine

Acknowledges new alarms for the specified block.

Syntax

AcknowledgeAnAlarm*[DataPoint], [intErrorMode], [BsendMsg]*

Properties

The **AcknowledgeAnAlarm** subroutine syntax has these parts:

Part	Description
DataPoint	String. (Optional) The name of the database block for which you want to acknowledge alarms. If no block is specified, alarms for the database block associated with the selected object are acknowledged.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
BsendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

AlarmHornEnabled Subroutine

Gets or sets the alarm horn enabled status.

Syntax

AlarmHornEnabled *([blnNewValue], [intErrorMode])*

Properties

The **AlarmHornEnabled** subroutine syntax has these parts:

Part	Description
blnNewValue	Boolean. (Optional). The value to which you want to set the alarm horn enable property.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Errors are displayed in the form of a message box. 1 – Errors are not handled so that they can be handled in the calling routine. 2 – Errors are dispatched to the alarm destinations using SendOperatorMessage.

Return Value

Boolean. The status of the AlarmHornEnable after the call is completed.

True = The horn will sound on any new alarm.

False = The horn will not sound for any new alarms.

AlarmHornEnabledToggle Subroutine

Toggles the system's AlarmHornEnabled status.

Syntax

AlarmHornEnabledToggle *([intErrorMode])*

Properties

The **AlarmHornEnabledToggle** subroutine syntax has this part:

Part	Description
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Errors are displayed in the form of a message box. 1 – Errors are not handled so that they can be handled in the calling routine. 2 – Errors are dispatched to the alarm destinations using SendOperatorMessage.

Return Value

Boolean. The status of the AlarmHornEnable after the call is completed.

True = The status was toggled to True. The horn will sound on any new alarm.

False = The status was toggled to False. The horn will not sound for any new alarms.

AlarmHornSilence Subroutine

Silences the alarm horn.

Syntax

AlarmHornSilence *([intErrorMode])*

Properties

The **AlarmHornSilence** subroutine syntax has this part:

Part	Description
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Errors are displayed in the form of a message box. 1 – Errors are not handled so that they can be handled in the calling routine. 2 – Errors are dispatched to the alarm destinations using SendOperatorMessage.

Remarks

The alarm horn must be enabled for this to work. If you call this routine and the alarm horn is disabled, no error will be reported.

If a new alarm comes in after this routine is called, the alarm horn will sound again. When alarms come in at a rapid rate, it may seem as though the AlarmHornSilence subroutine is not working because the new alarms keep retriggering the horn.

CloseDigitalPoint Subroutine

Closes, or sends a value of 1, to the specified digital block.

Syntax

CloseDigitalPoint*[DigitalPoint], [intErrorMode], [bSendMsg]*

Properties

The **CloseDigitalPoint** subroutine syntax has these parts:

Part	Description
DigitalPoint	String. (Optional). The name of the digital block that you want to close. If no block is specified, it closes the digital point associated with the selected object.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bSendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

Remarks

If no block is specified and the selected object does not have an associated digital point, an error message appears informing the user that the write was not successful.

ClosePicture Subroutine

Closes the specified **Picture**. If there are multiple instances of a picture open, all instances of that picture are closed. If a different alias is assigned to each instance, you can close one instance using an alias.

Syntax

ClosePicture*[Picture], [intErrorMode]*

Properties

The **ClosePicture** subroutine syntax has these parts:

Part	Description
Picture	String. (Optional). The file name or alias of the picture you want to close. If no file name is specified, the currently active picture is closed.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used.

DisableAlarm Subroutine

Disables alarm limit checking for the specified block.

Syntax

DisableAlarm*[DataPoint], [intErrorMode], [bSendMsg]*

Properties

The **DisableAlarm** subroutine syntax has these parts:

Part	Description
DataPoint	String. (Optional) The name of the database block for which you want to disable alarming. If no block is specified, alarming is disabled for the block associated with the selected object.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used.

	1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bSendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

EnableAlarm Subroutine

Enables alarm limit checking for the specified block.

Syntax

EnableAlarm*[DataPoint], [intErrorMode], [bSendMsg]*

Properties

The **EnableAlarm** subroutine syntax has these parts:

Part	Description
DataPoint	String. (Optional). The name of the database block for which you want to enable alarming. If no block is specified, alarming is enabled for the database block associated with the selected object.
intErrorMode	Integer. (Optional). The error mode. 0 - (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bSendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

FetchLimits Subroutine

Returns the High and Low EGUs of the specified data source.

Syntax

FetchLimits*DataSource, HiLimit, LoLimit, ret, [intErrorMode]*

Properties

The **FetchLimits** subroutine syntax has these parts:

Part	Description
DataSource	String. The string name for the data source. For example, FIX32.MYNODE.AI1.F_CV
HiLimit	Single. Returns the high EGU limit for the data source.
LoLimit	Single. Returns the low EGU limit for the data source.
ret	Integer. Returns the status of the call: 0 = Successful 1 = Syntax error 2 = Data source does not exist 3 = Data type mismatch
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 3 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.

FindDataSource Subroutine

Returns the string name of the data source that is connected to a property of the specified object if the data source is an iFIX database tag or animation object.

Syntax

FindDataSource (*Object, [strProperty]*)

Properties

The **FindDataSource** function syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
strProperty	String. (Optional). The property connected to the data source. If no property is specified, the function returns the connection to the first property if finds that is connected to a data source.

Return Value

String. The string name of the data source object.

Remarks

This function does not return a value for items within groups. For example, if you have a group of data links all connected to different data sources but, the group itself is not connected to a data source, this function will return an empty string. It only returns the name of the data source that is connected to the object you pass in for the Object parameter.

FindLocalObject Subroutine

Finds an object inside a group based on the object's partial name. The group could be a **Picture**, **DynamoSet** or a **Group** of shapes.

Syntax

FindLocalObject (*StartObject, PartialName*)

Properties

The **FindLocalObject** subroutine syntax has these parts:

Part	Description
StartObject	Object. The name of the **Picture** or **Group** where the object you are looking for is contained.
PartialName	String. A partial name for the object to be found. For example, if the object's full name is PipeColorAnim1, you can pass in “PipeColorA”, or “PipeC”.

Return Value

Object. The first object in the **Group** whose name contains what is entered for *PartialName*.

Remarks

For example, if, through scripting, you want to get an object inside a **Group** in order to animate that particular object's vertical fill, use **FindLocalObject** with the group's name and just a partial name of the object to fill.

FindLocalObject is typically used for Dynamo sets where a Dynamo objects share common names for all of their contained objects - the only difference being the numeric ending. Forms and subroutines that call this subroutine make use of the partial name to operate on all similar Dynamo objects so that all similar Dynamo objects in a picture can use the same subroutines and forms. This assumes that user creating the Dynamo objects uses a naming convention for the objects inside of the Dynamo object.

G-I

GeneratePicture Subroutine

Creates a new **Picture**.

Syntax

GeneratePicture (*aPicInfo*)

Properties

The **GeneratePicture** subroutine syntax has this part:

Part	Description
aPicInfo	The PictureInfo structure specifies the properties of the picture to be generated.

The members of the PictureInfo structure are as follows:

Member	Description
lfTopPct	The preferred top window edge location of the picture being generated. The location Unit, pixel or percent, is determined by bPixels. Type is Double.
lfLeftPct	The preferred left window edge location of the picture being generated. The location Unit, pixel or percent, is determined by bPixels. Type is Double.
lfHeightPct	The preferred window height of the picture being generated. The dimension Unit, pixel or percent, is determined by bPixels. Type is Double.
lfWidthPct	The preferred window width of the picture being generated. The dimension Unit, pixel or percent, is determined by bPixels. Type is Double.
lBkColor	Color for background color of the picture. Type is Long.
szName	The name of the picture. Type is String.
bPixels	Determines whether the window location units are pixel or percent. Type is Boolean.
bTitlebar	Defines whether or not the picture window has a title bar. Type is Boolean.
bSystemMenu	Defines whether or not the picture window has a system menu. Type is Boolean.
bResizable	Defines whether or not the picture window is resizable. Type is Boolean.
bAlwaysOnTop	Defines whether or not the picture window is always on top of other picture windows. Type is Boolean.
bRuntimeVisible	Defines whether or not the picture is visible at run time. Type is Boolean.

GetAllConnections Subroutine

Retrieves all data blocks connected to objects in a **Picture** or in a **Group**.

Syntax

GetAllConnections (*Obj*)

Properties

The **GetAllConnections** subroutine syntax has this part:

Part	Description
Obj	Object. Usually the page object that represents the **Picture** whose connections you want to retrieve. This can also be a **Group** object.

Return Value

Collection. The Collection of all the fully qualified names of database items in the **Picture**. These items can be either data links or data items that animate objects.

GetDecimalSeparator Subroutine

Reserved method used internally for Experts.

Syntax

GetDecimalSeparator (*[intErrorMode]*)

Properties

The **GetDecimalSeparator** subroutine syntax has this part:

Part	Description
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.

GetFormDynamoColor Subroutine

Reserved method used internally for iFIX Dynamos.

Syntax

GetFormDynamoColor *DynColor*

GetFormNumeric Subroutine

Reserved method used internally for Experts.

Syntax

GetFormNumeric ()

GetFormPushbutton Subroutine

Reserved method used internally for Experts.

Syntax

GetFormPushbutton ()

GetFormRamp Subroutine

Reserved method used internally for Experts.

Syntax

GetFormRamp ()

GetFormSlider Subroutine

Reserved method used internally for Experts.

Syntax

GetFormSlider ()

GetLocaleInfoA Subroutine

Retrieves information relating to a specific locale.

Syntax

GetLocaleInfoA (*Locale, LCType, lpLCData, cchData*)

Properties

The **GetLocaleInfoA** subroutine syntax has these parts:

Part	Description
Locale	Long. The locale ID from where you want to get information.
LCType	Long. The type of information to retrieve. Refer to the API32.TXT file for constants with the LOCALE_ prefix.
lpLCData	String. Buffer to load with the information. Make sure this string is initialized to the appropriate length.
cchData	Long. The length of the lpLCData buffer , or zero to get the buffer length.

Return Value

Long. The number of characters loaded into the buffer. Zero on error.

Remarks

GetLocaleInfoA is a Windows API call.

GetUserDefaultLCID Subroutine

Retrieves the default locale for the current user.

Syntax

GetUserDefaultLCID ()

Return Value

Long. The default locale ID for the current user.

Remarks

GetUserDefaultLCID is a Windows API call.

HandleError Subroutine

The global error handler routine. It displays the error number and its description when an error occurs.

Syntax

HandleError(*[intErrorMode]*)

Properties

The **HandleError** subroutine syntax has this part:

Part	Description
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.

IsUserFxg Subroutine

Determines whether the active document is the User.fxg.

Syntax

IsUserFxg ()

Return Value

Boolean. **True** if the active picture is the User.fxg. **False** if it is not.

Remarks

This subroutine is useful when cycling through the collection of open **Documents**. Typically, a user would not want to perform the same actions that you do on User.fxg that he/she would on a picture or schedule. Thus, if **IsUserFxg** returns **True**, the user can write their code to skip User.fxg.

L-R

LocateObject Subroutine

Searches a **Picture** for the specified object and selects the object.

Syntax

LocateObject*LinkName, bRelative, [intErrorMode]*

Properties

The **LocateObject** subroutine syntax has these parts:

Part	Description
LinkName	String. The name of the object for which you want to search.
bRelative	Boolean. If **True**, the current picture is searched; if **False**, all open pictures are searched.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 2 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 3 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.

LogIn Subroutine

The **LogIn** subroutine:

- Executes the standard Login program.
- Logs out the current user upon accepting a new login and saves the current user information which is restored after the execution of a subsequent **Login** subroutine.
- Updates the user ID information so that all operator messages and SCADA node security checking reflects the correct user.

Syntax

LogIn(*[intErrorMode]*)(*[bPushCurrentUser As Boolean]*)

Properties

The **LogIn** subroutine syntax has these parts:

Part	Description
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bPushCurrentUser	Boolean. (Optional). False (default) – By default, this feature is disabled. True – Pushes the current user (UserA) to the stack when logging in a new user (UserB). This parameter allows you to logout UserA after accepting the UserB login but saves the UserA information. The saved information is restored after the execution of a subsequent LogOut subroutine.

OffScan Subroutine

Places the specified block off scan.

Syntax

OffScan*[DataPoint], [intErrorMode], [bSendMsg], [BsendMsg]*

Properties

The **OffScan** subroutine syntax has these parts:

Part	Description
DataPoint	String. (Optional). The name of the database block that you want to turn off scan. If no block is specified, the database block for the selected object is turned off scan.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed

	in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bSendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.
BsendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

OnScan Subroutine

Places the specified block on scan.

Syntax

OnScan*[DataPoint], [intErrorMode], [bSendMsg]*

Properties

The **OnScan** subroutine syntax has these parts:

Part	Description
DataPoint	String. (Optional). The name of the database block that you want to put on scan. If you do not specify a name, the database block for the selected object is put on scan.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bSendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

OpenDigitalPoint Subroutine

Opens, or sends a value of 0, to the specified digital block.

Syntax

OpenDigitalPoint*[DigitalPoint], [intErrorMode], [bSendMsg]*

Properties

The **OpenDigitalPoint** subroutine syntax has these parts:

Part	Description
DigitalPoint	String. (Optional). The name of the digital block that you want to open. If no block is specified, it opens the digital point associated with the selected object.
bstrEventName	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bSendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

Remarks

If no block is specified and the selected object does not have an associated digital point, an error message will occur informing the user that the write was not successful.

OpenPicture Subroutine

Opens the specified **Picture**.

Syntax

OpenPicture*[Picture], [PictureAlias], [TopPosition], [LeftPosition], [intErrorMode], [CallingPicture], [TSPicType], [TagList], [bNewInstance]*

Properties

The **OpenPicture** subroutine syntax has these parts:

Part	Description
Picture	String. (Optional) The file name of the **Picture** you want to open. If no file name is specified, the Open dialog is launched allowing the user to select a picture to open.
PictureAlias	String. (Optional) The alias that you want to assign to the **Picture**.
TopPosition	Variant. (Optional) The position at which you want to place the top of the **Picture**. Default = 0.
LeftPosition	Variant. (Optional) The position at which you want to place the left side of the **Picture**. Default = 0.

intErrorMode	Integer. (Optional) The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
Calling Picture	Object. (Optional) The picture whose tag status functionality will be accessed.
TSPicType	TS_PIC_TYPE (Optional) The type of Tag Status picture to open: 0 – Single Tag TS picture 1 – Quick Trend 2 – Tag Control Panel
TagList	Variant. (Optional) An array of strings to be used in the tag status picture.
bNewInstance	Boolean. (Optional) Describes whether the picture is a new instance. This field is useful when opening multiple pictures. True – Opens a new instance of the same picture. False – Does not open a new instance of the picture. This is the default setting.

Remarks

The alias is a user-defined generic picture name. By using aliases, you can control the opening and closing of multiple pictures without concern for the currently displayed picture.

Tag status functionality is available only when no Picture string has been provided and if the CallingPicture and TSPicType parameters are provided. If the CallingPicture parameter is not provided, the call to OpenPicture is treated as a regular OpenPicture request. If the CallingPicture parameter is provided, but the TSPicType parameter is not, the call to OpenPicture generates an error message.

If a string has been provided via the Picture parameter, the standard OpenPicture functionality is invoked. In this case, the CallingPicture, TSPicType, and TagList parameters are ignored.

In run mode, you can modify the VBA scripting in a new instance of a picture. However, modifications to scripts in an instance of a picture will not be saved. Saving changes to scripts in run mode is not allowed. To make changes to a script in the main picture, switch to configure mode and edit the script in the main picture (not the instance).

OpenTGDPicture Subroutine

Opens the specified **Picture** with the specified tag group file.

Syntax

OpenTGDPicture*[Picture], [PictureAlias], [TopPosition], [LeftPosition], [TagGroupName], [intErrorMode], [bNewInstance]*

Properties

The **OpenTGDPicture** subroutine syntax has these parts:

Part	Description
Picture	String. (Optional) The file name of the **Picture** you want to open. If no file name is specified, the Open dialog is launched allowing the user to select a picture to open.
PictureAlias	String. (Optional) The alias that you want to assign to the **Picture**.
TopPosition	Variant. (Optional) The position at which you want to place the top of the **Picture**. Default = 0.
LeftPosition	Variant. (Optional) The position at which you want to place the left side of the **Picture**. Default = 0.
TagGroupName	String (Optional) The tag group file name. Do not specify a path for this parameter.
intErrorMode	Integer. (Optional) The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bNewInstance	Boolean. (Optional) Describes whether the picture is a new instance. This field is useful when opening multiple pictures. True – Opens a new instance of the same picture. False – Does not open a new instance of the picture. This is the default setting.

Remarks

The alias is a user-defined generic picture name. By using aliases, you can control the opening and closing of multiple pictures without concern for the currently displayed picture.

In run mode, you can modify the VBA scripting in a new instance of a picture. However, modifications to scripts in an instance of a picture will not be saved. Saving changes to scripts in run mode is not allowed. To make changes to a script in the main picture, switch to configure mode and edit the script in the main picture (not the instance).

PictureAlias Subroutine

Assigns an alias, or nickname, to the current **Picture**.

Syntax

PictureAlias*PictureAlias, [intErrorMode]*

Properties

The **PictureAlias** subroutine syntax has these parts:

Part	Description
PictureAlias	String. The alias that you want to assign to the current picture.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.

PrintReport Subroutine

Prints the specified Crystal Report.

Syntax

PrintReport *Report, [Prompt],[Copies],[Coll],[StartNo],[EndNo], [intErrorMode]*

Properties

The **PrintReport** subroutine syntax has these parts:

Part	Description
Report	String. The name of the report to print, including the file's full path.
Prompt	Boolean. (Optional) If **True**, the user is prompted with the open report dialog, allowing the user to select the report to print. If **False**, the report specified is printed. This option is ignored in Crystal XI.
Copies	Long. (Optional) The number of copies to print. Default = 1. This option is ignored in Crystal XI.
Coll	Boolean. (Optional) If **True**, the reports are collated. If **False**, they are not. This option is ignored in Crystal XI.
StartNo	Long. (Optional) The first page of the report to print. This option is ignored in Crystal XI.
EndNo	Long. (Optional) The last page of the report to print. This option is ignored in Crystal XI.
intErrorMode	Integer. (Optional). The error mode. This option is ignored in Crystal XI. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used.

	1 - Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.

QuickAdd Subroutine

Launches the Quick Add dialog box for adding a data block.

Syntax

QuickAdd (*DataSource, [intErrorMode]*)

Properties

The **QuickAdd** subroutine syntax has these parts:

Part	Description
DataSource	String. The name of the database block you want to add.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.

Return Value

Integer. The status of the database block.

0 = OK (data block already exists)

1 = Invalid Syntax

2 = Undefined (Quick Add dialog is launched)

3 = Data Type Mismatch

4 = User chose Use Anyway

5 = User chose not to add the block

RampValue Subroutine

Manually ramps the specified database block by increasing or decreasing its current value by a specified percentage of the engineering units (EGU) range.

Syntax

RampValue*RampValue, ByPercent, [DataPoint], [intErrorMode]*

Properties

The **RampValue** subroutine syntax has these parts:

Part	Description
RampValue	String. The value you want to use to ramp the database block.
ByPercent	Boolean. If **True**, the block is ramped by the percentage of the EGU range. If **False**, the data point is ramped by the *RampValue*.
DataPoint	String. (Optional). The name of the database block that you want to ramp. If no block is specified, the value of the data source associated with the selected object is ramped.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.

ReadValue Subroutine

Reads the value of the specified block.

Syntax

ReadValue*[DataPoint], [intErrorMode]*

Properties

The **ReadValue** subroutine syntax has these parts:

Part	Description
DataPoint	String. (Optional). The database block that you want to read. If no block is specified, the data source associated with the selected object is read.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling.

	2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage. To view your enabled alarm destinations, open the System Configuration Utility (SCU.EXE) and on the Configure menu, click Alarm. The Alarm Configuration dialog box appears where you can access this information.

Return Value

ReadValue returns the value of the specified block.

RegCloseKey Subroutine

Closes a key in the system registry. FactoryGlobals provides the Declare statement in GlobalSubroutines so you can use this Windows API call without declaring it in your procedures.

NOTE: For Crystal XI users, use the PrintReport subroutine instead of using the RegOpenKeyEx and RegCloseKeyEx subroutines. The PrintReport subroutine will do all of the registry entries for you.

Syntax

RegCloseKey (*hKey*)

Properties

The **RegCloseKey** subroutine syntax has this part:

Part	Description
hKey	Long. The key to close.

Return Value

Long. Zero on success. All other values indicate an error.

Remarks

RegCloseKey is a Windows API call.

RegOpenKeyEx Subroutine

Opens an existing registry key. FactoryGlobals provides the Declare statement in GlobalSubroutines so you can use this Windows API call without declaring it in your procedures.

NOTE: For Crystal XI users, use the PrintReport subroutine instead of using the RegOpenKeyEx and RegCloseKeyEx subroutines. The PrintReport subroutine will do all of the registry entries for you.

Syntax

RegOpenKeyEx (*hKey, lpSubKey, ulOptions, samDesired, phkResult*)

Properties

The **RegOpenKeyEx** subroutine syntax has these parts:

Part	Description
hKey	Long. Handle of an open key or one of the standard key names. HKEY_CLASSES_ROOT HKEY_CURRENT_USER HKEY_LOCAL_MACHINE HKEY_USERS HKEY_CURRENT_CONFIG HKEY_DYN_DATA Use the VB5.0 API Viewer to get the correct hex number for the key.
lpSubKey	String. Name of the key to open.
ulOptions	Long. Unused. Set to zero.
samDesired	Long. One or more constants with the prefix KEY_ that describes which operation are allowed for this key.
phkResult	Long. Variable to load with a handle to the open key.

Return Value

Long. Zero on success. All other values indicate an error.

Remarks

RegOpenKeyEx is a Windows API call.

ReplacePicture Subroutine

Closes a **Picture** and replaces it with another **Picture**. If a picture with a specified tag group file is already open in the WorkSpace, another instance of that picture with the same tag group file opens in the WorkSpace. The picture being opened will stretch or skew to fit the document height and width of the picture being replaced. The DisableAutoScale setting does not affect this behavior.

Syntax

ReplacePicture *NewPicture, [OldPicture], [TagGroupFileName], [intErrorMode], [bShowPictureNotOpenErrors]; [CallingPicture]; [TSPicType]; [TagList]; [bNewInstance]*

Properties

The **ReplacePicture** subroutine syntax has these parts:

Part	Description
NewPicture	String. The file name of the **Picture** you want to open.
OldPicture	String. (Optional) The file name of the **Picture** you want to replace. If no picture is specified, the currently active picture is replaced.
TagGroupFileName	String. (Optional) The file name of the Tag Group File you want to load with the new picture.
intErrorMode	Integer. (Optional) The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for

	the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bShowPictureNotOpenErrors	Boolean. (Optional). If this value is set to TRUE, an error displays when the picture to be replaced is not open and no replace occurs. The default is FALSE.
CallingPicture	Object. (Optional) The picture whose tag status functionality will be accessed.
TSPicType	TS_PIC_TYPE (Optional) The type of Tag Status picture to open: 0 – Single Tag Status picture 1 – Quick Trend picture 2 – Tag Control Panel picture
TagList	Variant. (Optional) An array of strings to be used in the tag status picture.
bNewInstance	Boolean. (Optional). Describes whether the picture is a new instance. This field is useful when opening multiple pictures. If the value is set to TRUE, a new instance of the same picture is opened. If the value is set to FALSE, then a new instance of the picture is not opened. The default is FALSE.

Remarks

Tag status functionality is available only when no NewPicture string has been provided and if the CallingPicture and TSPicType parameters are provided. If the CallingPicture parameter is not provided, the call to ReplacePicture is treated as an invalid ReplacePicture request. If the CallingPicture parameter is provided, but the TSPicType parameter is not, the call to ReplacePicture is treated as an invalid ReplacePicture request.

If a string has been provided via the NewPicture parameter, the standard ReplacePicture functionality is invoked. In this case the CallingPicture, TSPicType, and TagList parameters are ignored.

If possible, always make this call the last line in your script. Note that when the **ReplacePicture** subroutine is used, and the document being replaced is the document that contains the script, the call *must* be the last line in the script. Otherwise, you may experience unexpected behavior when executing the script.

If the **ReplacePicture** subroutine is not in the picture being replaced and is not the last line in your script, be certain that the operation is complete before the rest of the script continues to execute.

ReplaceTGDPicture Subroutine

Closes a **Picture** and replaces it with another **Picture** with the specified tag group file. If a picture with that specified tag group file is already open in the WorkSpace, another instance of that picture with the same tag group file opens in the WorkSpace.

Syntax

ReplaceTGDPicture *NewPicture, TagGroupName, [OldPicture], [bNewInstance]*

Properties

The **ReplaceTGDPicture** subroutine syntax has these parts:

Part	Description
NewPicture	String. The file name of the **Picture** you want to open.
TagGroupName	String. Tag group file name. Do not specify a path.
OldPicture	String. (Optional) The file name of the **Picture** you want to replace. If no picture is specified, the currently active picture is replaced.
bNewInstance	Boolean. (Optional). Describes whether the picture is a new instance. This field is useful when opening multiple pictures. If the value is set to TRUE, a new instance of the same picture is opened. If the value is set to FALSE, then a new instance of the picture is not opened. The default is FALSE.

Remarks

If possible, always make this call the last line in your script. Note that when the ReplaceTGDPicture subroutine is used, and the document being replaced is the document that contains the script, the call must be the last line in the script. Otherwise, you may experience unexpected behavior when executing the script.

If the ReplaceTGDPicture subroutine is not in the picture being replaced and is not the last line in your script, be certain that the operation is complete before the rest of the script continues to execute.

S-Z

SetAuto Subroutine

Sets the specified block to automatic mode.

Syntax

SetAuto*[DataPoint], [intErrorMode], [bSendMsg]*

Properties

The **SetAuto** subroutine syntax has these parts:

Part	Description
DataPoint	String. (Optional) The name of the database block that you want to set to automatic mode. If no block is specified, the database block associated with the selected object is set to automatic mode.

intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bSendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

SetManual Subroutine

Sets the specified block to manual mode.

Syntax

SetManual*[DataPoint], [intErrorMode], [bSendMsg]*

Properties

The **SetManual** subroutine syntax has these parts:

Part	Description
DataPoint	String. (Optional) The name of the database block that you want to set to manual mode. If no block is specified, the database block associated with the selected object is set to manual mode.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bSendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

SetSymbolValues Subroutine

Sets the symbol substitutions for model context.

Syntax

SetSymbolValues *(vtSymbol)*

Properties

The **SetSymbolValues** subroutine syntax has this part:

Part	Description
vtSymbol	2 dimensional array. The first column is the index and second is a string parameter that holds the symbol. For use with the model.

ShellExecute Subroutine

Finds the file name of the program that is associated with a specified file and either runs the program for the file or prints the file. FactoryGlobals provides the Declare statement in GlobalSubroutines so you can use this Windows API call without declaring it in your procedures.

Syntax

ShellExecute (*hwnd, lpOperation, lpFile, lpParameters, lpDirectory, nShowCmd*)

Properties

The **ShellExecute** subroutine syntax has these parts:

Part	Description
hwnd	Long. A handle to a window.
lpOperation	String. The string “Open” to open the lpFile document or “Print” to print it. You can use vbNullString to default to “Open”.
lpFile	String. A program name or the name of a file to print or open using the associated program.
lpParameters	String. A string with parameters to pass to the exe file if lpszFile is an executable file. VbNullString if lpszFile refers to a document file or if on parameters are used.
lpDirectory	String. The full path of the default directory to use.
nShowCmd	Long. A constant value specifying how to show the launched program.

Return Value

Long. The success of the **ShellExecute** call. A value greater than 32 signifies success.

Remarks

ShellExecute is a Windows API call.

ToggleDigitalPoint Subroutine

Toggles the current state of the digital block between open and closed.

Syntax

ToggleDigitalPoint*[DigitalPoint], [intErrorMode], [bSendMsg]*

Properties

The **ToggleDigitalPoint** subroutine syntax has these parts:

Part	Description
DigitalPoint	String. (Optional). The name of the digital block that you want to toggle. If no block is specified, it opens the digital point associated with the selected object.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bSendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

ToggleManual Subroutine

Toggles the mode status of the specified block between manual and automatic modes.

Syntax

ToggleManual*[DataPoint], [intErrorMode], [bSendMsg]*

Properties

The **ToggleManual** subroutine syntax has these parts:

Part	Description
DataPoint	String. (Optional). The name of the database block whose state you want to toggle. If no name is specified, the database block associated with the selected object is toggled.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.

bSendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

ToggleScan Subroutine

Toggles the scan status of the specified block.

Syntax

ToggleScan *[DataPoint], [intErrorMode], [bSendMsg]*

Properties

The **ToggleScan** subroutine syntax has these parts:

Part	Description
DataPoint	String. (Optional). The name of the database block whose scan status you want to toggle. If you do not specify a name, the scan status of the database block associated with the selected object is toggled.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
bSendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

WriteValue Subroutine

Sets the value of the current value of the specified database block.

Syntax

WriteValue*Value , [DataPoint], [intErrorMode] [BsendMsg]*

Properties

The **WriteValue** subroutine syntax has these parts:

Part	Description
Value	String. The value that you want to write to the specified database block.
DataPoint	String. (Optional). The database block to which you want to write the specified value. If

	no block is specified, the data source associated with the selected object is write.
intErrorMode	Integer. (Optional). The error mode. 0 (default) – Allows the subroutines to provide the error messages. Errors are displayed in the form of a message box. If no entry is made for the intErrorMode parameter, the default is used. 1 – Allows the user to handle the error messages. Errors in the subroutines are passed back to the calling routine for handling. 2 – Writes errors to all destinations. No error messages display. Instead, the errors are written to all iFIX destinations, including the Alarm History window destinations using SendOperatorMessage.
BsendMsg	Boolean. (Optional). If TRUE, the message "value changed byxxx" will be sent to the alarm destinations. If FALSE, the message will be suppressed.

Database Functions Summary

The following list contains the iFIX Database Functions that are available to the Automation Interface. For information on non iFIX objects, refer to the appropriate help system.

- eda_add_block
- eda_delete_block
- eda_get_pdb_name
- eda_reload_databse
- eda_save_database
- eda_type_to_index
- FixGetMyName
- NlsGetText

eda_add_block Function

Add a new block of the specified type on the specified node.

Syntax

Function **eda_add_block**(*NodeName, TagName, BlkType*)

Properties

The **eda_add_block** function syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
NodeName	String. The node name on which to add the block.
TagName	String. The name to identify the tag (maximum of 256 characters).
BlkType	Integer. The block type index.

Return Value

Integer. FE_OK if data is valid

Remarks

Note that the type is passed as an integer. A type string such as “AI” can be converted to a type index by calling the **eda_type_to_index** function as listed below. The new tag will be initialized to the same default values that appear when a new tag is created with DatabaseManager. These values can be written using the various other functions mentioned here.

eda_delete_block Function

Deletes the specified database block from the database.

Syntax

Function **eda_delete_block** (*NodeName, TagName*)

Properties

The **eda_delete_block** function syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
NodeName	String. The node name on which to delete the block.
TagName	String. The name of the block to be deleted.

Return Value

FE_OK if successful; FE_XXX if error.

Remarks

Deletes the specified block from the database. If this block is part of a chain, then the previous and next blocks of the chain will be reconnected. It is good practice to place the block OFFSCAN before deleting it.

eda_get_pdb_name Function

Retrieves the PDB file name from the database.

Syntax

Function **eda_get_pdb_name** As Integer (*NodeName, DatabaseName, BufSize*)

Properties

The **eda_get_pdb_name** function syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
NodeName	String. The name of the node on which the database resides.
DatabaseName	String. Returns the name of the database for *NodeName*.
BufSize	Integer. The maximum size, in bytes, to return in *DatabaseName*.

Return Value

Integer. FE_OK if no error.

Remarks

This is the name of the current PDB file loaded by the system (for example, by the DBB RELOAD function or by the eda_reload_database function). The name is returned **without** the path or extension.

The *DatabaseName* parameter must be declared either as fixed-length strings of 260 characters or as variable-length strings that are initialized as 260 characters before calling **eda_get_pdb_name**. The *NodeName* parameter must be declared either as fixed-length strings of 9 characters or as variable-length strings that are initialized as 9 characters before calling **eda_get_pdb_name**. For example, you could initialize the NodeName parameter in either of the following ways:

```
Dim NodeName As String * 9
or
Dim NodeName As String
NodeName = " "
```

eda_reload_database Function

Loads the specified PDB to a file into the specified node.

Syntax

Function **eda_reload_database** (*NodeName, DatabaseName*)

Properties

The **eda_reload_database** function syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
NodeName	String. The name of the node on which to re-load the database.
DatabaseName	String. The name of the database file that is to be loaded into memory.

Return Value

Integer. FE_OK if no error

Remarks

This is equivalent to the DBB Reload... function. The name passed becomes the new name of the database. The name should be passed **without** a path or extension.

NOTE: The previous database in memory is replaced. No check is made to see if the database has been modified.

eda_save_database Function

Saves the current active database to a file.

Syntax

Function **eda_save_database** (*NodeName, DatabaseName*)

Properties

The **eda_save_database** function syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
NodeName	String. The name of the node whose database is to be saved.
DatabaseName	String. The name of the file to which the current database is to be saved.

Return Value

Long. Some typical errors would be:

Value	Description
FE_OK	Successful.
FE_SEC_ACCESSS	Unauthorized access attempted.
FE_BAD_FILENAME	*DatabaseName* specified is bad.
FE_READ_ONLY	Database is read only.

Remarks

This is equivalent to the DBB Save As... function. The name passed becomes the new name of the database. The name should be passed **without** a path or extension.

NOTE: This file will only exist on the node whose database is being saved. This may be different than the node that is making the **eda_save_database** call.

eda_type_to_index Function

Retrieves the type index for the specified string containing a block type.

Syntax

Function **eda_type_to_index** (*NodeName, BlockType*)

Properties

The **eda_type_to_index** function syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
NodeName	String. Name of node on which the block is located.
BlockType	String. The block type whose block index is required.

Return Value

Integer. The database block type (a value between 1 and 150). Returns 0 if the type name is not recognized.

Remarks

The block index returned by this function should be used when adding blocks to the database via the **eda_add_block** function.

FixGetMyname Function

Get the node name of this node. This function works whether or not the FIX is running.

Syntax

Function **FixGetMyname** (*Myname, MaxSize*)

Properties

The **FixGetMyname** function syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
Myname	String. Returns the name of the current node.
MaxSize	Long. The maximum size, in bytes, to return in *Myname*.

Return Value

Long.

Value	Description
FTK_OK	Successful.
FTK_BAD_LENGTH	String length too small.
FTK_NODENAME_NOT_DEFINED	No node name defined for this node.
FTK_BAD_MHANDLE	Pointer to non-writable memory passed in.
FTK_NO_MESSAGE	No message exists for error.
FTK_BAD_LENGTH	String length too small.

Remarks

The maximum size for MyName (MaxSize) is NODE_NAME_SIZE.

The *MyName* parameter must be declared either as a fixed-length string of 9 characters before calling **FixGetMyName**. For example, initialize the *MyName* parameter in the following way:

```
Dim Myname As String * 9 'init variable
Myname = " " 'clear variable before usage
```

When the length of the *Myname* string (the node name) does not fill the maximum size (9 characters), the returned string contains NULL. To remove the NULL from the VBA string, you can use the following code snippet:

```
rtn = FixGetMyname(Myname, 9)
StrMyNode = ""
For i = 1 To len(Myname)
rtn = Mid(Myname, i, 1)
If Asc(rtn) >= 65 And (Asc(rtn) <= 95) Then 'check if I'm Alpha
StrMyNode = StrMyNode & rtn
Else
If Asc(rtn) >= 48 And (Asc(rtn) <= 57) Then 'Check if I'm numeric
StrMyNode = StrMyNode & rtn
End If
End If
Next i
```

In addition, you may want to check for characters such as the underscore (_) and other valid characters used in a node name. The previous example assumes that your node name only contains the characters A to Z, or the numbers 1 to 9.

NlsGetText Function

Translates an error number returned as a string.

Syntax

Function **NlsGetText** (*ErrCode, MsgString, MaxLength*)

Properties

The **NlsGetText** function syntax has these parts:

Part	Description
object	An object expression that evaluates to an object in the Applies To list.
ErrCode	Integer. The error code returned from a previous eda function call.
MsgString	String. The error message that corresponds to *ErrCode*.
MsgString	Integer. The maximum size, in bytes, to return in *MsgString*.

Return Value

Long.

Part	Description
FTK_OK	Successful.
FTK_BAD_MHANDLE	Pointer to non-writable memory passed in.
FTK_NO_MESSAGE	No message exists for error.
FTK_BAD_LENGTH	String length too small.

Remarks

The Error originates from iFIX. If the message is longer than specified in MaxLength, then string will be truncated. If no message exists for Error, FTK_NO_MESSAGE will be returned.

The *MsgString* parameter must be declared as a fixed-length string of 255 characters before calling **NlsGetText**. For example, you could initialize the *MsgString* parameter like so:

Dim MsgString As String * 255

Examples

The following list contains available examples. For information on non-iFIX methods, refer to the appropriate help system.

A

AboutBox Method Example

AckAlarm Method Example

AckAlarmPage Method Example

AckAlarmPageEx Method Example

AckAllAlarms Method Example

AcknowledgeAllAlarms Subroutine Example

AcknowledgeAnAlarm Subroutine Example

ActivateWorkspaceUI Method Example

Add Method Example

AddDataSet Method Example

AddEventHandler Method Example

AddImage Method Example

AddLegendItem Method Example

AddLevel Method Example

AddObject Method Example

AddPen Method Example

AddPictureToStartupList Method Example

AddPoint Method Example

AddProcedure Method Example

AlarmHornEnabled Subroutine Example

AlarmHornEnabledToggle Subroutine Example

AlarmHornSilence Subroutine Example

Align Method Example

ApplyProperty Method Example

AutoScaleDisplayLimits Method Example

B

BringToFront Method Example

BuildObject Method Example

C

CanConstruct Method Example

CheckAccountExpiration Method Example

CheckSecurityEnabled Method Example

CheckSyntax Method Example

CheckUserApplicationAccess Method Example

CheckUserAreaAcess Method Example

Clear Method Example

ClearUndo Method Example

Close Method Example

CloseDigitalPoint Subroutine Example

ClosePicture Subroutine Example

Commit Method Example

Connect Method Example

ConnectDataSet Method Example

ConnectedPropertyCount Method Example

Construct Method Example

ConvertPipe Method Example

Convert_A_Group_To_A_Dynamo_By_Name Method Example

Convert_A_Group_To_A_Dynamo_By_Ref Method Example

ConvertPipe Method Example

ConvertSecurityAreaNameToNumber Method Example

ConvertSecurityAreaNumberToName Method Example

ConvertToEnhancedCoordinates Method Example

ConvertToOriginalCoordinates Method Example

Copy Method Example

Coupled_Activate_Workspace_UI Method Example

Coupled_DeActivate_Workspace_UI Method Example

CopyAsBitmap Method Example

CreateDynamoByGrouping Method Example

CreateFromDialog Method Example

CreateFromProgID Method Example

CreateWithMouse Method Example

Cut Method Example

D

DeActivateWorkspaceUI Method Example

DefaultView Method Example

DelAlarm Method Example

DeleteAllAlarms Method Example

DeleteAllDataSets Method Example

DeleteDataSet Method Example

DeleteImage Method Example

DeletePen Method Example

DeletePoint Method Example

DeleteSelectedObjects Method Example

DemandFire Method Example

DeselectObject Method Example

DestroyObject Method Example

DisableAlarm Subroutine Example

DisableNonSelectionEvents Method Example

Disconnect Method Example

DisplaysControlPoints Method Example

DoesPropertyHaveTargets Method Example

DoExtendLines Method Example

DoLinesToPolyline Method Example

DoMenuCommand Method Example

DoTrimLines Method Example

DumpProperties Method Example

Duplicate Method Example

E

EditPicture Method Example

Enable Method Example

EnableAlarm Subroutine Example

Enumerate_All_Dynamos Method Example

Enumerate_All_Groups Method Example

Enumerate_Top_Level_Dynamos Method Example

Enumerate_Top_Level_Groups Method Example

Esignature Object Example

ExchangePenPositions Method Example

ExportData Method Example

ExportImage Method Example

ExportLanguageFile Method Example

F

FetchLimits Subroutine Example

FindAndReplaceDialog Method Example

FindDataSource Subroutine Example

FindInString Method Example

FindLocalObject Subroutine Example

FindObject Method Example

FindReplaceInObject Method Example

FindReplaceInString Method Example

FitDocumentToWindow Method Example

FitWindowToDocument Method Example

FixCheckApplicationAccess Method Example

FixCheckApplicationAccessQuiet Method Example

FixCheckAreaAccess Method Example

FixCheckAreaAccessQuiet Method Example

FixCheckSecurityEnabled Method Example

FixGetManualAlmDeleteEnabled Method Example

FixGetUserInfo Method Example

FixLogin Method Example

FixLogout Method Example

FontProperties Method Example

FullView Method Example

G

GeneratePicture Subroutine Example

Get_Last_Prompt_Value Method Example

Get_Last_Result_String Method Example

GetAlarmBackgroundColor Method Example

GetAlarmForegroundColor Method Example

GetAllConnections Subroutine Example

GetBoundRect Method Example

GetChartEndTime Method Example

GetChartStartTime Method Example

GetColHeadings Method Example

GetColumnInfo Method Example

GetConnectionInformation Method Example

GetConnectionParameters Method Example

GetContinuousUser Method Example

GetCurrentDataSet Method Example

GetCurrentValue Method Example

GetCurrentValueWithQuality Method Example

GetDataSetByPosition Method Example

GetDecimalSeparator Subroutine Example

GetDeviceRect Method Example

GetDuration Method Example

GetErrorString Method Example

GetEventHandlerIndex Method Example

GetFormDynamoColor Subroutine Example

GetFormNumeric Subroutine Example

GetFormPushbutton Subroutine Example

GetFormRamp Subroutine Example

GetFormSlider Subroutine Example

GetFullname Method Example

GetIndirectionInfo Method Example

GetInterval Method Example

GetNumberOfDataSets Method Example

GetLevel Method Example

GetLocaleInfoA Subroutine Example

GetObjectInfo Method Example

GetPenDataArray Method Example

GetPenDataArrayEx Method Example

GetPointAt Method Example

GetPriorityColor Method Example

GetProcedureIndex Method Example

GetProperty Method Example

GetPropertyAttributes Method Example

GetPropertyTargets Method Example

GetSelectedAlmExt Method Example

GetSelectedNodeTag Method Example

GetSelectedRow Method Example

GetSelectedRowAlarmInfo Method Example

GetSelectedRowsAlarmInfo Method Example

GetSelectedUserDefFields Method Example

GetSignature Method Example

GetSignatureAndWriteValue Method Example

GetStatusColor Method Example

GetStatusFont Method Example

GetTimeBeforeNow Method Example

GetTimeCursorInfo Method Example

GetUserDefaultLCID Subroutine Example

GetUserID Method Example

GetWindowLocation Method Example

GlobalScrollBackFast Method Example

GlobalScrollBackSlow Method Example

GlobalScrollForwardFast Method Example

GlobalScrollForwardSlow Method Example

GlobalTimerApply Method Example

Group Method Example

H

HandleError Subroutine Example

HiLoDisplay Method Example

I-K

ImportToolbar Method Example

Initialize Method Example

InitializeList Method Example

InsertPoint Method Example

InteractiveExport Method Example

IsColorSelectionVisible Method Example

IsConnected Method Example

IsEmpty Method Example

IsNodeSignEnabled Method Example

IsSignatureRequired Method Example

IsSignatureRequiredForList Method Example

IsUserFxg Subroutine Example

Item Method Example

L

ListEvents Method Example

ListMethods Method Example

ListProperties Method Example

ListWindowsGroupNames Method Example

Load_TS_List Method Example

LoadImage Method Example

LoadTagGroupFile Method Example

LocateObject Subroutine Example

LogicalToPercentage Method Example

LogicalToUserFormPoint Method Example

LogIn Subroutine Example

M-N

MakeLinesHorizontal Method Example

MakeLinesVertical Method Example

MakeSameSize Method Example

Modify Method Example

ModifyColumnLength Method Example

Move Method Example

NewAlarm Event Example

O

OffScan Subroutine Example

OnScan Subroutine Example

Open Method Example

Open_QT_Pic Method Example

Open_QT_Pic_Ex Method Example

Open_TCP_Pic Method Example

Open_TCP_Pic_Ex Method Example

Open_TS_Pic Method Example

Open_TS_Pic_Ex Method Example

Open_TS_Pic_Type Method Example

Open_TS_Pic_Type_Ex Method Example

OpenDigitalPoint Subroutine Example

OpenPicture Subroutine Example

OpenTGDPicture Subroutine Example

P-Q

ParseConnectionSource Method Example

Paste Method Example

PasteSpecial Method Example

Pause Method Example

PauseAlarmRead Example

PercentageToLogical Method Example

PercentageToPixel Method Example

PictureAlias Subroutine Example

PixelToPercentageMethod Example

PrintChart Method Example

PrintOut Method Example

PrintReport Subroutine Example

PromptToChangePassword Method Example

QuickAdd Subroutine Example

Quit Method Example

R

RampValue Subroutine Example

Read Method Example

ReadValue Subroutine Example

Refresh Method Example

RefreshChartData Method Example

RegCloseKey Subroutine Example

RegOpenKeyEx Subroutine Example

Remove Method Example

RemoveAll Method Example

RemoveAllLevels Method Example

RemoveItem Method Example

RemoveLegendItem Method Example

RemoveLevel Method Example

RemoveObject Method Example

ReplacePicture Subroutine Example

RemovePictureFromStartupList Method Example

Replace_QT_Pic Method Example

Replace_TCP_Pic Method Example

Replace_TS_Pic_Type Method Example

Replace_TS_Pic Method Example

ReplaceDocument Method Example

ReplaceInString Method Example

ReplaceTGDPicture Subroutine Example

ResetChartData Method Example

ResetObjectStats Method Example

ResetStats Method Example

ResetZoom Method Example

ResolveTagGroupFile Method Example

Resume Method Example

ResumeAlarmRead Method Example

RetrieveDefinition Method Example

RetrieveTagGroupVariables Method Example

Rotate Method Example

RunObject Method Example

S

Save Method Example

Save_TS_List Method Example

SaveAsSVG Method Example

SaveToHistoryList Method Example

ScrollBack Method Example

ScrollForward Method Example

ScrollTimeBack Method Example

ScrollTimeForward Method Example

ScrollToPosition Method Example

Select Method Example

SelectAlarmRow Method Example

SelectAll Method Example

SelectObject Method Example

SendOperatorMessage Method Example

SendSignedOperatorMessage Method Example

SendToBack Method Example

SetAlarmBackgroundColor Method Example

SetAlarmForegroundColor Method Example

SetAuto Subroutine Example

SetContinuousUser Method Example

SetCurrentValue Method Example

SetDispatch Method Example

SetDispid Method Example

SetDuration Method Example

SetFocusToComboBox Method Example

SetGlobalMovingEndTimeToCurrent Method Example

SetIndirectionInfo Method Example

SetInterval Method Example

SetLegendMask Method Example

SetManual Subroutine Example

SetNumericFormat Method Example

SetPenDataArray Method Example

SetPointAt Method Example

SetPriorityColor Method Example

SetProperty Method Example

SetScriptWindow Method Example

SetSource Method Example

SetStatusColor Method Example

SetStatusFont Method Example

SetStringFormat Method Example

SetSymbolValues Subroutine Example

SetTabSelection Method Example

SetTimeBeforeNow Method Example

SetTimeCursorTime Method Example

SetWindowLocation Method Example

ShellExecute Subroutine Example

ShelveAlarm Method Example

ShowAnimations Method Example

ShowBrowseDialog Method Example

ShowColorBox Method Example

ShowColorSelection Method Example

ShowCustomPages Method Example

ShowPipePreviewDialog Method Example

ShowTaskWizard Method Example

ShowVBAProcedure Method Example

ShowVisualBasicEditor Method Example

SilenceAlarmHorn Method Example

SnapObjectsToGrid Method Example

SpaceEvenly Method Example

StartEvent Method Example

StartTimer Method Example

StickToCursor Method Example

StopEvent Method Example

StopGlobalPlayBack Method Example

StopTimer Method Example

Stretch Method Example

SwitchLanguage Method Examples

SwitchMode Method Example

SynchronizeSecurity Method Example

T

TagGroupSubstitution Method Example

TagGroupValue Method Example

ToggleDigitalPoint Subroutine Example

ToggleManual Subroutine Example

ToggleScan Subroutine Example

U

UIActivate Method Example

UIDeActivate Method Example

Undo Method Example

UndoTransaction Method Example

UndoZoom Method Example

UnGroup Method Example

UnShelveAlarm Method Example

UnloadTagGroupFile Method Example

Update_A_Dynamo_By_Name Method Example

Update_A_Dynamo_By_Name2 Method Example

Update_A_Dynamo_By_Ref Method Example

Update_A_Dynamo_By_Ref2 Method Example

UpdateBackgroundObject Method Example

UpdateConnectionParameters Method Example

UpdateDefinition Method Example

UserFormPointToLogical Method Example

V-Z

ValidateSignature Method Example

ValidateSignatureAndWriteValue Method Example

ValidateSource Method Example

ValueTimeFromXY Method Example

WriteValue Subroutine Example

Write Method Example

XYFromValueTime Method Example

XYHitTest Method Example

Zoom Method Example

ZoomToFit Method Example

A

AboutBox Method Example

The following example opens the Help About Box for the **Alarm Summary** object *AlarmSummaryOCX1*.

```
AlarmSummaryOCX1.AboutBox
```

AckAlarm Method Example

The following example acknowledges an alarm generated for the block titled *AI1* on a node titled *NODE1* in the Alarm Summary object *AlarmSummaryOCX1*.

```
Dim iRVal as integer
iRVal = AlarmSummaryOCX1.AckAlarm ("NODE1", "AI1")
```

AckAlarmPage Method Example

The following example acknowledges the currently displayed page of alarms in the Alarm Summary object *AlarmSummaryOCX1*.

```
Dim iRVal as integer
iRVal = AlarmSummaryOCX1.AckAlarmPage
```

AckAlarmPageEx Method Example

The following example acknowledges the currently displayed page of alarms in the Alarm Summary object *AlarmSummaryOCX1*.

```
AlarmSummaryOCX1.AckAlarmPageEx
```

AckAllAlarms Method Example

The following example acknowledges all alarms in the Alarm Summary object *AlarmSummaryOCX1*.

```
Dim iRVal as integer
iRVal = AlarmSummaryOCX1.AckAllAlarms
```

AcknowledgeAllAlarms Subroutine Example

The following example acknowledges alarms for all blocks for Picture *TestPicture.*

```
AcknowledgeAllAlarms "TestPicture"
```

AcknowledgeAnAlarm Subroutine Example

The following example acknowledges alarm for block *AI1*.

```
AcknowledgeAnAlarm "AI1"
```

ActivateWorkspaceUI Method Example

The following example activates the WorkSpace UI.

```
Application.ActivateWorkspaceUI
```

Add Method Example

The following example adds a picture to the WorkSpace by default.

```
Application.Documents.Add
```

The following example adds a Microsoft Word Document to the WorkSpace.

```
Application.Documents.Add "Word.Document"
```

The following example adds the subroutine *MySub* to *Rect1*'s **Procedures** collection.

```
Rect1.Procedures.Add 1, "Private Sub MySub(x as integer, y as integer)"
```

The following example adds a line of code to the fifth line in *Rect1*'s first event's event handler using the **Lines** collection.

```
Rect1.Procedures.Item(1).Lines.Add "Msgbox 2", 5
```

The following example adds a data **Group** to the **Groups** of the **FixDataSystem** *FDS*.

```
FDS.Groups.Add ("DataGroup1")
```

AddDataSet Method Example

The following example adds a data set to an object named LineChart1 (a Line Chart). An undefined object can be accepted as the data source. (True indicates a UseAnyway condition.)

```
Dim objDS As Object
Set objDS = LineChart1.AddDataSet("Fix32.Fix.AI1.F_CV")
```

This next example adds a data set to an object named HistogramChart1 (a Histogram Chart). Be aware that only a Histogram block can be used with a HistogramChart object, and only the T_DATA field can be used with this block to show the data. (T_DATA is array of 16-bit integers. T_DATA is supported in the HistogramChart object, while T_DATA2 is not. T_DATA2 is an array of 32-bit integers.)

```
Dim objDS As Object
Set objDS = HistogramChart1.AddDataSet("Fix32.GCMSA01.HS1.T_DATA", True)
```

AddEventHandler Method Example

The following example adds the event handler "*OpenPicture*" to the **Click** for the object *CurrentObject*.

```
Dim lIndex As Long
CurrentObject.Procedures.AddEventHandler "Click", "OpenPicture", lIndex
```

The resulting procedure is as follows:

```
private sub CurrentObject_Click()
OpenPicture
end sub
```

AddImage Method Example

The following example adds the image *CustomButton10* to the **Bitmap** object *Bitmap1* and then displays it by setting it to be the current image.

```
Bitmap1.AddImage "C:\Program Files (x86)\Proficy\iFIX\Local\CustomButton10.bmp"
Bitmap1.CurrentImage = Bitmap1.ImageCount
```

AddLegendItem Method Example

The following example adds the legend item *High OverRange* in column 5 of the legend of **Chart** object *TestChart*, displaying 10 characters.

```
TestChart.AddLegendItem "High Over", 5, 10
```

AddLevel Method Example

The following example adds a level to the **Lookup** object *Lookup1*, using a range comparison. This level will have an output of 100 for any input value between 10 and 20.

```
Lookup1.AddLevel 10, 100, 20
```

The following example adds a level to the **Lookup** object *Lookup1*, using an exact match comparison. For each input of 40, this level will display an output of 75.

```
Lookup1.AddLevel 40, 75
```

AddObject Method Example

The following example adds the **Oval** object *TestOval* to the **Group** *TestGroup.*

```
TestGroup.AddObject TestOval
```

AddPen Method Example

The following example adds a pen to the **Chart** *Chart1* using a datasource of *AI1.*

```
Dim iPen As Object
Set iPen = Chart1.AddPen("AI1")
```

AddPictureToStartupList Example

The following example adds a Picture named *pic1.grf* (in the D:\Program Files (x86)\Proficy\iFIX\PIC directory) to the Runtime environment's startup list of the iFIX WorkSpace.

```
Dim lErr as Long
lErr = Application.UserPreferences.AddPictureToStartupList_
("D:\Program Files (x86)\Proficy\iFIX\pic\pic1.grf", True)
```

Note that the full path is required for the picture. Also note that the second paramater, when set to TRUE, adds the picture to the Runtime startup list. When set to FALSE, it adds the picture to the Configuration environment's startup list.

AddPoint Method Example

The following example adds a point to the Polygon object *Polygon1*.

```
Dim iPoint as FixFloatPoint
Set iPoint = New FixFloatPoint
iPoint.x = 50.5
iPoint.y = 60.1
Polygon1.AddPoint iPoint
```

AddProcedure Method Example

This example generates the following code in the ActiveDocument's project:

```
Public Sub Rect2_DisplayMessage(strMessage As String)
MsgBox strMessage
End Sub
Dim o As Object
Dim lIndex As Long
Set o = Application.ActiveDocument.Page.FindObject("Rect2")
o.Procedures.AddProcedure "DisplayMessage", "strMessage as String", _
" MsgBox strMessage", lIndex
```

AlarmHornEnabled Example

In Example 1, blnReturn contains the status of the system's AlarmHornEnabled property

Example 1

```
Dim blnReturn As Boolean
blnReturn = AlarmHornEnabled
In Example 2, True passed to enable the alarm horn, then
 checks to see if it was set.
```

Example 2

```
Dim blnReturn As Boolean
blnReturn = AlarmHornEnabled(True)
If Not blnReturn Then
MsgBox ("Horn could not be enabled")
End If
```

AlarmHornEnabledToggle Example

This example shows how to toggle the alarm horn and read the return value that indicates how the status was changed.

```
Dim blnReturn As Boolean
blnReturn = AlarmHornEnabledToggle
If blnReturn Then
MsgBox ("The horn is now Enabled")
Else
MsgBox ("The horn is now Disabled"
End If
```

AlarmHornSilence Example

This example silences the alarm horn.

```
Private Sub Button1_Click ()
AlarmHornSilence
End Sub
```

Align Method Example

The following example aligns the tops of the **Oval** and **Polygon** objects, *Oval1* and *Polygon1*.

```
Oval1.SelectObject False
Polygon1.SelectObject False
untitled1.Align 3
```

ApplyProperty Method Example

Reserved for internal purposes.

AutoScaleDisplayLimits Method Example

The following example changes the display limits for all pens in the **Chart**, *Chart1* based on the range of the data currently in the pens' data array

```
Chart1.AutoScaleDisplayLimits
```

B

BringToFront Method Example

The following example selects the **Oval** object *Oval1* contained within the **Picture** *TestPicture* and brings it to the front of the stacking order.

```
Oval1.Select
TestPicture.BringToFront
```

BuildObject Method Example

The following example adds a **Rectangle** to the **Picture***TestPicture.*

```
Dim iRect As Object
Set iRect = TestPicture.BuildObject("Rect")
iRect.HorizontalPosition = 10.0
iRect.VerticalPosition = 30.0
iRect.EdgeWidth = 5
iRect.FillStyle = 1
iRect.Commit
```

C

CanConstruct Method Example

The following example determines whether the datasource *AI1* for *NODE1* has valid syntax for the **Picture** *TestPicture.*

```
Dim bCanConstruct As Boolean
TestPicture.CanConstruct "FIX32.NODE1.AI1", bCanConstruct
```

CheckAccountExpiration Method Example

The following example checks the user's Windows account expiration status. If the account expired, this example prompts the user to change his password (if he has rights to do so). If the account has not expired, a message appears stating how many days are left until the password expires.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim UserName As String
Dim PassWord As String
Dim UsrId As String
Dim bExpired As Boolean
Dim daysLeft As Long
Dim bCanChangePassword As Boolean
'user name of an iFix user using Windows security
UserName = "expire"
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignature")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Check account status and user's right to change password
```

```
ESig.CheckAccountExpiration UserName, bExpired, bCanChangePassword, daysLeft
If bExpired <> False Then
If bCanChangePassword <> False Then
ESig.PromptToChangePassword UserName
End If
Else
'password is not expired
MsgBox "Password is due to expire in " & daysLeft & " days."
End If
Else
'the hardware key is not enabled for signature, security is disabled, or bypass signature is in effect
MsgBox “Signature is not enabled on this node.”
End If
```

CheckSecurityEnabled Method Example

The following example creates the ESignature Object and checks to see if security is enabled on the current node.

```
Dim ESig As Object
Dim bEnabled As Boolean
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignature")

'Check if security is enabled
ESig.CheckSecurityEnabled bEnabled
MsgBox "Security enabled: " & bEnabled
```

CheckSyntax Method Example

The following example returns the syntax check for the **ExpressionEditor** object *ExpressionEditor1*.

```
Dim bSuccess As Boolean
Dim strExpression As String
strExpression = "AI1.F_CV + AI2.F_CV"
bSuccess = ExpressionEditor1.CheckSyntax(strExpression)
```

CheckUserApplicationAccess Method Example

The following example creates the ESignature object, checks to see if the node has electronic signature enabled, obtains the user ID, checks to see if the user has access to the Electronic Signature Bypass application feature, and checks to see if the user has access to a security area.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim UserName As String
Dim PassWord As String
Dim result As Boolean
Dim UsrId As String
'valid iFix user name and password
UserName = "username"
PassWord = "password"
```

```
result = False
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignature")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Get the user id
ESig.GetUserid UserName, PassWord, UsrId
'Check if user has access to Electronic Signature Bypass application feature
ESig.CheckUserApplicationAccess UsrId, 74, result
MsgBox "User: " & UserName & vbCr & "Has rights to Bypass Signature? " & result
'Check if user has access to security area B
ESig.CheckUserAreaAccess UsrId, "B", result
MsgBox "User: " & UserName & vbCr & "Has rights to Security Area B? " & result
Else
'the hardware key is not enabled for signature, security is disabled, or bypass signature is in effect
MsgBox ‟Signature is not enabled on this node.”
End If
```

CheckUserAreaAccess Method Example

The following example creates the ESignature object, checks to see if the node has electronic signature enabled, obtains the user ID, checks to see if the user has access to the Electronic Signature Bypass application feature, and checks to see if the user has access to a security area (in this example, B).

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim UserName As String
Dim PassWord As String
Dim result As Boolean
Dim UsrId As String
'valid iFix user name and password
UserName = "username"
PassWord = "password"
result = False
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignature")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Get the user id
ESig.GetUserid UserName, PassWord, UsrId
'Check if user has access to Electronic Signature Bypass application feature
ESig.CheckUserApplicationAccess UsrId, 74, result
MsgBox "User: " & UserName & vbCr & "Has rights to Bypass Signature? " & result
'Check if user has access to security area B
ESig.CheckUserAreaAccess UsrId, "B", result
MsgBox "User: " & UserName & vbCr & "Has rights to Security Area B? " & result
Else
'the hardware key is not enabled for signature, security is disabled, or bypass signature is in effect
MsgBox ‟Signature is not enabled on this node.”
End If
```

Clear Method Example

The following example removes all members from the Groups collection of **FixDataSystem** *FDS*.

```
FDS.Groups.Clear
```

ClearUndo Method Example

The following example removes all currently registered undo transactions from the undo stack for *TestPicture*.

```
TestPicture.ClearUndo
```

Close Method Example

The following example closes all open non-hidden documents, saving changes without prompting the user.

```
Application.Documents.Close 1, 2
The following example opens the DocumentTestPicture and then closes it without saving any changes.
Dim iDoc As Object
Set iDoc = Application.Documents.open("C:\Program Files (x86)\Proficy\iFIX\pic\testpicture.grf")
iDoc.Close 2, 1
```

CloseDigitalPoint Subroutine Example

The following example closes the block *DO1.*

```
CloseDigitalPoint "DO1"
```

ClosePicture Subroutine Example

The following example closes the **Picture** *TestPicture.*

```
ClosePicture "TestPicture"
```

Commit Method Example

The following example creates a **Rectangle** *iRect*, customizes it, then commits these changes.

```
Dim iRect As Object
Set iRect = TestPicture.BuildObject("Rect")
iRect.HorizontalPosition = 10.0
iRect.VerticalPosition = 30.0
iRect.EdgeWidth = 5
iRect.FillStyle = 1
iRect.Commit
```

Connect Method Example

The following example ties the value of the *TEMP1* output string to the **Caption** property of the **Text** object *Text1*.

A connection is a one-way mechanism. Therefore, changes to the caption property directly (via OLE automation) will not be written to the connected source. Also, it may be necessary to transform the data from one type or range to another to create the appropriate user interface (i.e. numeric to string, numeric to color, numeric to screen coordinates). To accomplish this, an animation object should be used (**Linear**, **Lookup**, **Format**).

```
Dim objAnim as Object
Dim lngStatus as Long
'Create the animation object

Set objAnim = Rect1.BuildObject("Linear")
'Initialize the Linear object's
 properties
objAnim.Source = "Fix32.Scada1.AI1.F_CV"
'Set up transformation in and
 out range parameters
objAnim.AutoFetch = True 'Set it to fetch limits
objAnim.UseDelta = True 'Set it for relative movement
objAnim.LoOutValue = 0 'Set low output value
objAnim.HiOutValue = 15 'Set high output value
'Connect the animation object's
 output to the rectangle's position
Rect1.Connect "HorizontalPosition", objAnim.FullyQualifiedName
 & ".OutputValue", lngStatus
```

ConnectDataSet Method Example

This method is currently unavailable for use in iFIX. It is reserved for future use.

ConnectedPropertyCount Method Example

The following example determines the number of properties that have connections configured for the **Oval** object *Oval1.*

```
Dim lCount As Long
Oval1.ConnectedPropertyCount lCount
```

Construct Method Example

The following example displays the QuickAdd user interface that prompts the user for the information needed to create the tag *NewAI1* for the **Oval** object *Oval1* on node *NODE1.*

```
Dim lStatus As Long
```

```
Oval1.Construct "FIX32.NODE1.NEWAI", lStatus
```

Convert_A_Group_To_A_Dynamo_By_Name Method Example

There is currently no example available at this time.

Convert_A_Group_To_A_Dynamo_By_Ref Method Example

The following example shows the Convert_A_Group_To_A_Dynamo_By_Ref method example using the methods in PlugandSolve – modDynamoConverter.

```
Public Function ConvertADynamo(objGroup As Fix2DGroup.Fix2DGroup, objMasterDynamo As Fix2DDynamo.Fix2DDynamo, ByR
    Dim strDIName As String
    Dim strDMName As String
    Dim iDataSourceOption As DynamoDataSourceOption
    Dim iResult As Long 'UpdateDynamoResult
    Dim iPrompt As Long
    Dim iUpdateOptions As Long ' update options
    Dim bProceedConvert As Boolean

    iDataSourceOption = g_WizardConfig.iDataSourceOption
    If g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE Then
        If g_ReturnFromPromptForChoiceOnConvert = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_BUT_NOT_APPLY
        ElseIf g_ReturnFromPromptForChoiceOnConvert = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_AND_APPLY
        ElseIf g_ReturnFromPromptForChoiceOnConvert = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_NOT_UPDATE
        End If
    End If

    ' Convert option parameters
    iUpdateOptions = &H0
    If iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE Then
        iUpdateOptions = UPDATE_OPTION_ON_MISMATCH_PROMT_FOR_CHOICE
    ElseIf iDataSourceOption = DYNAMO_UPDATE_AND_APPLY Then
        iUpdateOptions = UPDATE_OPTION_ON_MISMATCH_UPDATE_DYNAMO + UPDATE_OPTION_ON_MISMATCH_APPLY_DATA_SOURCES
    ElseIf iDataSourceOption = DYNAMO_UPDATE_BUT_NOT_APPLY Then
        iUpdateOptions = UPDATE_OPTION_ON_MISMATCH_UPDATE_DYNAMO
    End If

    ' set options (it includes update option in case)
    If g_WizardConfig.bKeepSize = True Then
        iUpdateOptions = iUpdateOptions + UPDATE_OPTION_RESIZE_INSTANCE
    End If
    If g_WizardConfig.bKeepCaption = True Then
        iUpdateOptions = iUpdateOptions + UPDATE_OPTION_SAVE_CAPTIONS
    End If

    ' if FDS Converter, set Make Master option
    If g_iDynamoToolType = DYN_FDS_CONVERTER Then
        iUpdateOptions = iUpdateOptions + CONVERT_OPTION_MAKE_MASTER
    End If

    ' call Update A Dynamo
    PlugandSolve.GeometryHelperObj.Convert_A_Group_To_A_Dynamo_By_Ref objGroup, iUpdateOptions, mobjStrMgr.GetNLS
    ' get a result string
    strReturnMsg = PlugandSolve.GeometryHelperObj.Get_Last_Result_String

    ' don't take the last prompt for choice
    ' get the last user choice from the prompt
    'If (g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE) And _
    '    (g_ReturnFromPromptForChoiceOnConvert = PROMPT_DLG_SEL_NONE) Then
    '    iPrompt = PlugandSolve.GeometryHelperObj.Get_Last_Prompt_Value
    '    If (iPrompt = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL) Or _
    '        (iPrompt = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL) Or _
    '        (iPrompt = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL) Then
    '        g_ReturnFromPromptForChoiceOnConvert = iPrompt
    '    End If
    'End If

    ' convert result code
    If (iResult And UPDATER_RESULT_SUCCESS_BIT) > 0 Then
        ConvertADynamo = DYNAMO_UPDATED
    End If
    If (iResult And UPDATER_RESULT_DYNAMO_NOT_UPDATED_BIT) > 0 Then
        ConvertADynamo = DYNAMO_NOTUPDATED
    End If
    If (iResult And UPDATER_RESULT_USER_CANCELLED_BIT) > 0 _
        Or (iResult And UPDATER_RESULT_SUCCESS_BIT) = 0 Then
        ConvertADynamo = DYNAMO_UPDATE_ABORTED
    End If
End Function
```

ConvertPipe Method Example

The following example converts selected line(s) and polyline(s) to a pipe object in an active picture.

```
Set ObjHelper = BuildObject("GeometryHelper")
Call ObjHelper.ConvertPipe
```

ConvertSecurityAreaNameToNumber Method Example

The following example converts security area *D* to its corresponding area number.

```
Dim iAreaID As Integer
iAreaID = System.ConvertSecurityAreaNameToNumber("D")
```

ConvertSecurityAreaNumberToName Method Example

The following example converts security area *11* to its corresponding area name.

```
Dim sAreaName As String
sAreaName = System.ConvertSecurityAreaNumberToName(11)
```

ConvertToEnhancedCoordinates Method Example

ConvertToEnhancedCoordinates is a Run mode method. The following example uses a command button click event, and converts the hardcoded Width and Height values in to Enhanced Coordinates (postscript points).

```
Private Sub CommandButton5_Click()
LineChart1.Width = Me.ConvertToEnhancedCoordinates(57.9,True)
End sub
Private Sub CommandButton6_Click()
LineChart1.Height = Me.ConvertToEnhancedCoordinates(34.9,False)
End sub
```

NOTE: For more information on Enhanced Coordinates, refer to the Picture Coordinate Systems topic in the Creating Pictures e-book.

ConvertToOriginalCoordinates Method Example

ConvertToOriginalCoordinates is a Run mode method. The following example runs a command button click event, and converts the hardcoded Width and Height values to Logical Coordinates.

```
Private Sub CommandButton5_Click()
LineChart1.Width = Me.ConvertToOriginalCoordinates(500,True)
End sub
Private Sub CommandButton6_Click()
LineChart1.Height = Me.ConvertToOriginalCoordinates(300,False)
End sub
```

NOTE: For more information on Enhanced Coordinates, refer to the Picture Coordinate Systems topic in the Creating Pictures e-book.

Copy Method Example

The following example selects the **Oval** object *Oval1* and copies it to the clipboard.

```
Oval1.Select
TestPicture.Copy
```

Coupled_Activate_Workspace_UI Method Example

The following example activates the WorkSpace UI.

```
Application.Coupled_Activate_Workspace_UI
```

Coupled_DeActivate_Workspace_UI Method Example

The following example de-activates the WorkSpace UI, allowing a form to act modally. This call would be made on the form's initialization.

```
Application.Coupled_DeActivate_Workspace_UI False
```

The following example de-activates the WorkSpace UI, allowing a form to act modeless. This call would be made on the form's initialization.

```
Application.Coupled_DeActivate_Workspace_UI True
```

CopyAsBitmap Method Example

The following example selects the Oval object *Oval1* and copies it to the clipboard as a bitmap.

```
Oval1.Select
TestPicture.CopyAsBitmap
```

CreateDynamoByGrouping Method Example

The following code provides an example of the CreateDynamoByGrouping method:

```
Private Sub btnCreateADynamoByGrouping_Click()
' As long as there's selected objects a Dynamo will be created using those selected objects.
    Me.Create_Dynamo_By_Grouping
End Sub
```

CreateFromDialog Method Example

The following example creates the FixControlContainer object and then displays the Insert Object dialog box to allow the user to select which ActiveX control to create.

```
Dim iActiveX As Object
Set iActiveX = Application.ActiveDocument.Page.BuildObject("oleobject")

iActiveX.CreateFromDialog
```

CreateFromProgID Method Example

The following example creates the FixControlContainer and then the AlarmSummary control by specifying its ProgID.

```
Dim iActiveX As Object
Set iActiveX = Application.ActiveDocument.Page.BuildObject("oleobject")

iActiveX.CreateFromProgId "ALARMSUMMARYOCX.AlarmSummaryOCXCtrl.2"
```

CreateWithMouse Method Example

The following example creates an **Oval** by allowing the user to specify its size and position by creating it with the mouse.

```
Dim iOval As Object
Set iOval = Application.ActiveDocument.Page.BuildObject("oval")

iOval.CreateWithMouse
```

Cut Method Example

The following example selects the **Pie** object *Pie1* and removes it from its **Picture** *TestPicture* and copies it to the clipboard.

```
Pie1.Select
TestPicture.Cut
```

D

DeActivateWorkspaceUI Method Example

The following example de-activates the WorkSpace UI, allowing a form to act modally. This call would be made on the form's initialization.

```
Application.DeActivateWorkspaceUI False
```

The following example de-activates the WorkSpace UI, allowing a form to act modeless. This call would be made on the form's initialization.

```
Application.DeActivateWorkspaceUI True
```

DefaultView Method Example

NOTE: This method only applies to legacy Logical Coordinate System pictures. It does not apply to Enhanced Coordinates.

The following example sets the **Picture** *TestPicture* to its default window location.

```
TestPicture.DefaultView
```

The following example sets the **Picture** *TestPicture* to its default window location, preventing the repainting of the picture.

```
TestPicture.DefaultView False
```

DelAlarm Method Example

The following example deletes an alarm generated for the block titled *AI1* on a node titled *NODE1* in the **AlarmSummary** object *AlarmSummaryOCX1*.

```
Dim iRval As Integer
iRval = AlarmSummaryOCX1.DelAlarm("NODE1", "AI1")
```

DeleteAllAlarms Method Example

The following example deletes all alarms in the **AlarmSummary** object *AlarmSummaryOCX1*.

```
Dim iRval As Integer
iRval = AlarmSummaryOCX1.DeleteAllAlarms
```

DeleteAllDataSets Method Example

The following example shows an example of the DeleteAllDataSets method with the LineChart object.

```
LineChart1.DeleteAllDataSets
```

DeleteDataSet Method Example

The following example shows an example of the DeleteDataSet method with the LineChart object.

```
LineChart1.DeleteDataSet 0
```

DeleteImage Method Example

The following example deletes the primary and secondary images at the index of 1 for the **Bitmap** object *Bitmap1.*

Bitmap1.DeleteImage 1

DeletePen Method Example

The following example uses the Count property of the **Pens** Collection for the **Chart**, *Chart1,* as an index to delete the last pen added to the **Chart**.

```
Dim x As Integer
x = Chart1.Pens.Count
Chart1.DeletePen x
```

The following example deletes a pen in a Chartnamed*Chart1*,and then replaces it with another one using the AddPen method, if it does not already exist.

```
Private Sub Chart1_Click()
'Select a pen on user Click
'Check if user wants to delete it
Dim Strtn as string
Dim Inti as integer
Strtn = Chart1.CurrentPen
Msgprompt = "You have selected Pen : " & Chart1.Pens.Item(Strtn).Source & vbCrLf & "Do you want to remove it ?"
user_reponse = MsgBox(Msgprompt, vbYesNo, "Removing Pen")
If user_reponse = 6 Then
 Chart1.DeletePen (Strtn)
End If
End Sub

Private Sub DELPEN_Click()
'delete all pen from the pen collection under a specific chart
Dim Inti as integer
If Chart1.Pens.Count <> 0 Then
 For Inti = Chart1.Pens.Count To 1 Step -1
  Chart1.DeletePen (Inti)
 Next Inti
End If
End Sub

Private Sub PBADDREALPEN_Click()
'Add pen to the pen collection for a given Chart
'after checking the pens are not already inside the pen collection
If Chart1.Pens.Count <> 0 Then
' first time the chart is used the collection contains a default pen
 Chart1.DeletePen (1)
End If
 checkifalreadyexist ("Fix32.Alice.MyTag1")
 checkifalreadyexist ("Fix32.Alice.MyTag2")
 checkifalreadyexist ("Fix32.Alice.MyTag3")
End Sub

Function checkifalreadyexist(Tagname As String)
'check if the pen if not already inside the collection
Dim loc_tagname As String
loc_tagname = Tagname & ".F_CV"
Egu_tagname = Tagname
If Chart1.Pens.Count = 0 Then
 Chart1.AddPen (loc_tagname)
 hiEGU = Readvalue(Egu_tagname & ".A_Ehi")
 loEGU = Readvalue(Egu_tagname & ".A_Elo")

 Else
 Tag_found = False
  For i = 1 To Chart1.Pens.Count
   If UCase(loc_tagname) = UCase(Chart1.Pens.Item(i).Source) Then
Tag_found = True
   Else
 End If
  Next i
  If Tag_found = False Then
     Chart1.AddPen (loc_tagname)
     Tag_found = False
   End If
 End If
End Function
```

DeletePoint Method Example

The following example deletes the point at index 3 for the **Polyline** object *PolyLine1*.

```
PolyLine1.DeletePoint 3
```

DeleteSelectedObjects Method Example

The following example selects the **Chord** and **Polygon** objects *Chord1* and *Polygon1* and then deletes them from the **Picture** *TestPicture*.

```
Chord1.SelectObject False
Polygon1.SelectObject False
TestPicture.DeleteSelectedObjects
```

DemandFire Method Example

The following example fires the **OnTimeOut** configured for the **Timer***MyTimer* object in the **Schedule***TestSchedule.*

```
TestSchedule.DemandFire "MyTimer"
```

The following example fires the **OnTimeOut** event for the **Timer***MyTimer*.

```
MyTimer.DemandFire
```

The following example fires whatever event is configured for the **Event** object *MyEvent*.

```
MyEvent.DemandFire
```

DeselectObject Method Example

The following example deselects all objects currently selected in the **Picture***TestPicture.*

```
TestPicture.DeselectObject True
```

DestroyObject Method Example

The following example deletes the **Chord** object *Chord1* from its container.

```
Chord1.DestroyObject
```

DisableAlarm Subroutine Example

The following example disables alarm limit checking for block *AI1.*

```
DisableAlarm "AI1"
```

DisableNonSelectionEvents Method Example

Reserved for internal purposes.

Disconnect Method Example

The following example disconnects the VerticalFillPercentage property of the Oval object *Oval1* from its source.

```
Oval1.Disconnect "VerticalFillPercentage"
```

DisplaysControlPoints Method Example

The following example prohibits the Oval object *Oval1* from displaying its control points.

```
Oval1.DisplaysControlPoints False
```

DoesPropertyHaveTargets Method Example

The following example determines if the VerticalFillPercentage property of the Oval object *Oval1* has a target object connection, and if so, retains information about the target(s).

```
Dim bHasTargets As Boolean
Dim lNumOfTargets As Long
Dim lStatus As Long
Dim lIndex As Long
Oval1.DoesPropertyHaveTargets "VerticalFillPercentage", bHasTargets, lNumOfTargets, lStatus, lIndex
```

DoExtendLines Method Example

The following exampleshows that the selected line object(s) will be extended to the intersection when the ExtendType property is set to shorter than 30 pixels.

```
Set ObjHelper = BuildObject("GeometryHelper")
ObjHelper.ExtendType = 3 'set extend type max
ObjHelperExtendMaxSpace = 30 ' set max pixels to extend
Call ObjHelper.DoExtendLines
```

DoLinestoPolyline Method Example

The following exampleshows that the selected **line objects** will be converted to **polyline object(s)**.

```
Set ObjHelper = BuildObject ("GeometryHelper")
Call ObjHelper.DoLinesToPolyline
```

DoMenuCommand Method Example

The following example instantiates the New Timer dialog box for the **Schedule***TestSchedule* as if the user had selected Add New Timer Entry from the Insert menu.

```
TestSchedule.DoMenuCommand schNewTimer
```

The following example refreshes the **Schedule** after the **Event** object *MyEvent* has been removed from the **Schedule** *TestSchedule.*

```
TestSchedule.RemoveObject "MyEvent"
TestSchedule.DoMenuCommand scHREFreshView
```

DoTrimLines Method Example

The following exampleshows that the selected **line object(s)** will be trimmed to the extension point when the TrimType property is set to shorter than 30 pixels.

```
Set ObjHelper = BuildObject("GeometryHelper")
ObjHelper.TrimType = 3 ' set trim type max
ObjHelper.TrimMaxLength = 30 ' set max trim pixels
Call ObjHelper.DoTrimLines
```

DumpProperties Method Example

The following example writes all of the properties and their corresponding values for the **Picture** *TestPicture* and its contained objects to the CSV file *TestPicture.txt.*

```
Dim sRes1 As String
Dim sRes2 As String
TestPicture.DumpProperties "C:\Temp\TestPicture.txt",
 True, sRes1, sRes2
```

Duplicate Method Example

The following example selects the **Oval** object *Oval1* and then duplicates it.

```
Oval1.Select
TestPicture.Duplicate
```

E

EditPicture Method Example

The following example opens the configuration dialog box for the **Picture***TestPicture*.

```
TestPicture.EditPicture
```

Enable Method Example

The following example disables the **Color Button** object *ColorButton1*.

```
ColorButton1.Enable False
```

EnableAlarm Subroutine Example

The following example enables alarm limit checking for block *AI1.*

```
EnableAlarm "AI1"
```

Enumerate_All_Dynamos Method Example

The following code provides an example of the Enumerate_All_Dynamos method:

```
Private Sub btnEnumAllDynamos_Click()
    ' Useful to add a reference to iFix Geometry Helper & iFIX Collection
    Dim GH As FixGeometryHelper.FixGeometryHelper ' Geometry Helper object
    Dim DynamoCollection As Object ' Returned Collection of Dynamo Objects

    Set GH = Me.BuildObject("GeometryHelper")


    GH.Enumerate_All_Dynamos Me.ContainedObjects, DynamoCollection

    ' Now do something useful with the returned Dynamos
    Dim aDynamo As Fix2DDynamo.Fix2DDynamo
    Dim aStr As String

    For Each aDynamo In DynamoCollection
        aStr = aDynamo.FullyQualifiedName ' Do something useful here.
    Next aDynamo

    GH.DestroyObject
End Sub
```

Enumerate_All_Groups Method Example

The following code provides an example of the Enumerate_All_Groups method:

```
Public Function GetAllGroupList(objPic As CFixPicture, ByRef colDIList As Collection) As Boolean
    Dim colPIC As CFixObjectCollection
    Dim colList As CFixObjectCollection
    Dim objDI As Object

    If TypeName(objPic) = "Nothing" Or objPic.Category <> "Pictures" Then
        GetAllGroupList = False
        GoTo CLEAN_UP
    End If

    'If objHelper.ClassName <> "GeometryHelper" Then
    '    GetMasterDynamoList = False
    '    GoTo CLEAN_UP
    'End If

    ' clear dynamo instance list
    Do While colDIList.Count > 0
        colDIList.Remove (1)
    Loop

    ' get all Master Dynamos on a specified DynamoSets
    Set colPIC = objPic.ContainedObjects
    Call PlugandSolve.GeometryHelperObj.Enumerate_All_Groups(colPIC, colList)
    If colList.Count <= 0 Then
        GetAllGroupList = True
        GoTo CLEAN_UP
    End If

    For Each objDI In colList
        colDIList.Add objDI
    Next objDI

    GetAllGroupList = True
CLEAN_UP:
    On Error Resume Next
    Set colList = Nothing
    Set colPIC = Nothing
    Set objDI = Nothing

End Function
```

Enumerate_Top_Level_Dynamos Method Example

The following code provides an example of the Enumerate_Top_Level_Dynamos method:

```
Private Sub btnEnumTopLevelDynamos_Click()
    ' Useful to add a reference to iFix Geometry Helper & iFIX Collection
    Dim GH As FixGeometryHelper.FixGeometryHelper ' Geometry Helper object
    Dim DynamoCollection As Object ' Returned Collection of Dynamo Objects

    Set GH = Me.BuildObject("GeometryHelper")

    GH.Enumerate_Top_Level_Dynamos Me.ContainedObjects, DynamoCollection

    ' Now do something useful with the returned Dynamos
    Dim aDynamo As Fix2DDynamo.Fix2DDynamo
    Dim aStr As String
    For Each aDynamo In DynamoCollection
        aStr = aDynamo.FullyQualifiedName ' Do something useful here.
    Next aDynamo

    GH.DestroyObject
End Sub
```

Enumerate_Top_Level_Groups Method Example

The following code provides an example of the Enumerate_Top_Level_Groups method:

```
Public Function GetOldDynamoList(objFDS As CFixDynamoSet, ByRef colDMList As Collection) As Boolean
    Dim colFDS As CFixObjectCollection
    Dim colList As CFixObjectCollection
    Dim objDM As Object

    If TypeName(objFDS) = "Nothing" Or objFDS.Category <> "Dynamo Sets" Then
        GetOldDynamoList = False
        GoTo CLEAN_UP
    End If

    'If objHelper.ClassName <> "GeometryHelper" Then
    '    GetMasterDynamoList = False
    '    GoTo CLEAN_UP
    'End If

    ' clear dynamo instance list
    Do While colDMList.Count > 0
        colDMList.Remove (1)
    Loop

    ' get all Master Dynamos on a specified DynamoSets
    Set colFDS = objFDS.ContainedObjects
    Call PlugandSolve.GeometryHelperObj.Enumerate_Top_Level_Groups(colFDS, colList)
    If colList.Count <= 0 Then
        GetOldDynamoList = True
        GoTo CLEAN_UP
    End If

    For Each objDM In colList
        colDMList.Add objDM
    Next objDM

    GetOldDynamoList = True
CLEAN_UP:
    On Error Resume Next
    Set colList = Nothing
    Set colFDS = Nothing
    Set objDM = Nothing

End Function
```

ESignature Object Example

To use the ESignature object from VBA, declare an object variable and create the object by calling CreateObject( ). You can then access the ESignature object's methods through the object variable.

The sample code below illustrates how to use the ESignature object in VBA applications.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bSigRequired As Boolean
Dim Value as Variant
'Create the ESignature object
Set ESig = CreateObject (“ElectronicSignature.ESignature")

'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Check if data source requires electronic signature
ESig.Initialize ("Fix32. THISNODE.DO1.F_CV")
ESig.IsSignatureRequired 0, bSigRequired
If bSigRequired = True Then
'Capture the signature and write the new value
Value = 1
ESig.GetSignatureAndWriteValue 0, Value
Else
'Signature not required for this data source, just write
 the value
End If
Else
'Signature not enabled on this node, just write the value

End If
```

ExchangePenPositions Method Example

The following example sets the location of the Pen in position 1 to position 3 in the Pens collection.

```
Chart1.ExchangePenPositions 3, 1
```

ExportData Method Example

The following example exports the file to the location C:\TEMP\DATA.TXT, in table format.

```
Private Sub CommandButton6_Click()
LineChart1.ExportData "C:\TEMP\DATA.TXT", DataExportStyle_VerticalTable
End Sub
```

ExportImage Method Example

The following example exports the image to the clipboard, in the PNG file format, in the size of 800 X 600 pixels.

```
Private Sub CommandButton8_Click()
LineChart1.ExportImage "", ImageFormat_PNG, SizeUnits_Pixels, 800, 600
End Sub
```

ExportLanguageFile Method Example

The following example exports a Spanish language file.

```
pic.LanguageDesired = ES_Spanish
pic.ExportLanguageFile
```

- OR -

```
pic.ExportLanguageFile (ES_Spanish)
```

F

FetchLimits Subroutine Example

The following example gets the high and low EGUs for the data item FIX32.NODE1.AI1.F_CV and enters them in two text fields.

```
Dim sngHi As Single
Dim sngLo As Single
Dim intRet As Integer
Call FetchLimits ("FIX32.NODE1.AI1.F_CV", sngHi,
 sngLo, intRet)
If intRet = 0 Then
txtHighEGU.Caption = sngHi
txtLowEGU.Caption = sngLo
End If
```

FindAndReplaceDialog Method Example

The following example opens the Find And Replace dialog for the Picture*TestPicture*.

```
TestPicture.FindAndReplaceDialog
```

FindDataSource Subroutine Example

The following example finds the datasource for the VerticalFillPercentage property of the object *Tank1,* which is contained in the Picture *TestPicture.*

```
Dim iTank As Object
Dim sDataSource As String
Set iTank = untitled1.Tank1
sDataSource = FindDataSource(iTank, "VerticalFillPercentage")
```

FindInString Method Example

The following example finds each occurrence of the string *AI1* with the string *FIX32.NODE1.AI1.F_CV*. The operation is specified to include scripts in the search.

```
Dim lFirst As Long
Dim lCount As Long
Dim sMatchString As String
Dim bFound As Boolean
FindReplace.FindInString "FIX32.NODE1.AI1.F_CV", 1, "AI1", 8, sMatchString, lFirst, lCount, bFound
```

FindLocalObject Subroutine Example

The following example finds the object *PipeColorA* within the Group*Pipe*.

```
Dim PipeSectObj As Object
Set PipeSectObj = FindLocalObject(Pipe, "PipeColorA")
```

FindObject Method Example

The following example returns the Picture object *TestPicture*.

```
Dim iPic as object
Set iPic = System.FindObject("TestPicture")
The following example returns a data item reference for the block AI1 on node NODE1.
Dim iDataItem as object
Set iDataItem = System.FindObject("FIX32.NODE1.AI1.F_CV")
The following example returns an animation object for the Rectangle object Rect4 in PictureTestPicture.
Dim iLinear as object
Set iLinear = System.FindObject("TestPicture.Rect4.AnimatedHorizontalFill")
```

FindReplaceInObject Method Example

The following example replaces each instance of the string *NODE1* with the string *NODE2* within the Group object *Group1*. The operation is specified to be case sensitive.

```
Dim bFound as Boolean
FindReplace.FindReplaceInObject Group1, 1, "NODE1",
 "NODE2", bFound
```

FindReplaceInString Method Example

The following example finds the first occurrence of the string *AI1* within the string *FIX32.NODE1.AI1.F_CV* and replaces it with *AI2*, returning the new string in *sReplacement*. The operation is specified to include scripts in the search.

```
Dim lFirst As Long
Dim lCount As Long
Dim bsuccess As Boolean
Dim sTarget As String
Dim sReplacement As String
sTarget = "FIX32.NODE1.AI1.F_CV"
FindReplace.FindReplaceInString sTarget, 1, "AI1", "AI2", 8, lFirst, lCount, sReplacement, bsuccess
```

FitDocumentToWindow Method Example

IMPORTANT: The FitDocumentToWindow Method Example does not apply to documents that use the Enhanced Coordinate system. It is only available for documents using Logical Coordinates

The following example changes the size of the *TestPicture***Document** to fit within the window.

```
TestPicture.FitDocumentToWindow
```

The following example changes the size of the *TestPicture***Document** to fit within the window, preventing the repainting of the **Picture**.

```
TestPicture. FitDocumentToWindow False
```

FitWindowToDocument Method Example

IMPORTANT: The FitWindowToDocument Method Example does not apply to documents that use the Enhanced Coordinate system. It is only available for documents using Logical Coordinates.

The following example changes the size of the window to fit the size of the *TestPicture***Document**.

```
TestPicture.FitWindowToDocument
```

The following example changes the size of the window to fit the size of the *TestPicture***Document**, preventing the repainting of the **Picture**.

```
TestPicture. FitWindowToDocument False
```

FixCheckApplicationAccess Method Example

The following example checks to see if the user has access to the specified application.

```
Dim lAccess as long
lAccess = System.FixCheckApplicationAccess(DatabaseManager)
```

FixCheckApplicationAccessQuiet Method Example

The following example checks to see if the user has access to the specified application.

```
Dim lAccess as long
```

```
lAccess = System.FixCheckApplicationAccessQuiet(DatabaseManager)
```

FixCheckAreaAccess Method Example

The following example checks to see if the user has access to the security area 11.

```
Dim lAccess as long
lAccess = System.FixCheckAreaAccess(11)
```

FixCheckAreaAccessQuiet Method Example

The following example checks to see if the user has access to the security area 11.

```
Dim lAccess as long
lAccess = System.FixCheckAreaAccessQuiet(11)
```

FixCheckSecurityEnabled Method Example

The following example checks to see if security is enabled.

```
Dim lEnabled as long
lEnabled = System.FixCheckSecurityEnabled()
```

FixGetManualAlmDeleteEnabled Method Example

This example shows how to enable manual alarm deletions.

```
Dim bManlAlmDel As Boolean
System.FixGetManualAlmDeleteEnabled bManlAlmDel
If bManlAlmDel = True Then
MsgBox "Manual alarm deletion is enabled"
End If
```

FixGetUserInfo Method Example

The following example retrieves login information about the current user, including the user ID, user name and group name.

```
Dim sUserID as string
Dim sUserName as string
Dim sGroupName as string
System.FixGetUserInfo sUserID, sUserName, sGroupName
```

FixLogin Method Example

The following example logs in the user *TestUser* with a password of *MyPassword*.

```
System.FixLogin "TestUser", "MyPassword"
```

FixLogout Method Example

The following example logs out the first user.

```
System.FixLogout
```

FontProperties Method Example

The following example opens the font dialog for the **Text** object *Text1*.

```
Text1.FontProperties
```

FullView Method Example

NOTE: The FullView Method Example applies only to legacy Logical Coordinates. The FullView Method Example does not apply to Enhanced Coordinates.

The following example resizes the *TestPicture* **Document** to take up the entire screen.

```
TestPicture.FullView
```

G

GeneratePicture Subroutine Example

```
Dim aPicInfo As PictureInfo
aPicInfo.lfTopPct = TopPct
aPicInfo.lfLeftPct = LeftPct
aPicInfo.lfHeightPct = 100
aPicInfo.lfWidthPct = 100
aPicInfo.lBkColor = RGB(255,0,0)
aPicInfo.szName = "Tank1"
aPicInfo.bPixels = True
aPicInfo.bTitlebar = True
aPicInfo.bSystemMenu = True
aPicInfo.bResizable = True
aPicInfo.bAlwaysOnTop = False
aPicInfo.bRuntimeVisible = True
Dim retVal
retVal = GeneratePicture(aPicInfo)
```

Get_Last_Prompt_Value Method Example

The following code provides an example of the Get_Last_Prompt_Value method that appears in the mod-DynamoUpdater module of the Project_PlugandSolve VBA project.

```
Public Function UpdateADynamo(objDynamoInstance As Fix2DDynamo.Fix2DDynamo, objMasterDynamo As Fix2DDynamo.Fix2DD
    Dim strDIName As String
    Dim strDMName As String
    Dim iDataSourceOption As DynamoDataSourceOption
    Dim iResult As Long 'UpdateDynamoResult
    Dim iPrompt As Long

    iDataSourceOption = g_WizardConfig.iDataSourceOption
    If g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE Then
        If g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_BUT_NOT_APPLY
        ElseIf g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_AND_APPLY
        ElseIf g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_NOT_UPDATE
        End If
    End If

'////// actual update call
    ' call Update A Dynamo
    PlugandSolve.GeometryHelperObj.Update_A_Dynamo_By_Ref objMasterDynamo, objDynamoInstance, iDataSourceOption,
    ' get a result string
    strReturnMsg = PlugandSolve.GeometryHelperObj.Get_Last_Result_String
    ' get the last user choice from the prompt
    If (g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE) And _
        (g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_NONE) Then
        iPrompt = PlugandSolve.GeometryHelperObj.Get_Last_Prompt_Value
        If (iPrompt = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL) Or _
            (iPrompt = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL) Or _
            (iPrompt = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL) Then
            g_ReturnFromPromptForChoice = iPrompt
        End If
    End If
'////// end of actual update call
'////// dummy for test
'    g_testcount = g_testcount + 1
'    strDIName = objDynamoInstance.Name
'    strDMName = objMasterDynamo.Name
'    If g_testcount = 3 Then
'        strReturnMsg = ">>>" & strDIName & " was not updated  with " & strDMName
'        iResult = DYNAMO_NOTUPDATED
'    ElseIf g_testcount = 150 Then
'        strReturnMsg = ">>> User canceled"
'        iResult = DYNAMO_UPDATE_ABORTED
'    Else
'        strReturnMsg = strDIName & " was updated successfully with " & strDMName
'        iResult = DYNAMO_UPDATED
'    End If
'////// end of dummy for test

    If iResult <= DYNAMO_UPDATED Then
```

```
        UpdateADynamo = DYNAMO_UPDATED
    ElseIf iResult <= DYNAMO_NOTUPDATED Then
        UpdateADynamo = DYNAMO_NOTUPDATED
    Else
        UpdateADynamo = DYNAMO_UPDATE_ABORTED
    End If
End Function
```

▶ To view this code in context:

1. In Classic view, from the WorkSpace menu, select Visual Basic Editor.

   -Or-

   In Ribbon view, on the Home tab, in the WorkSpace group, click Visual Basic Editor.
2. In the tree view, double-click the Project_PlugandSolve folder, and then the Modules folder, and finally the modDynamoUpdater.
3. Search for UpdateADynamo to locate this code.

Get_Last_Result_String Method Example

The following code provides an example of the Get_Last_Result_String method that appears in the modDynamoUpdater module of the Project_PlugandSolve VBA project.

```
Public Function UpdateADynamo(objDynamoInstance As Fix2DDynamo.Fix2DDynamo, objMasterDynamo As Fix2DDynamo.Fix2DD
    Dim strDIName As String
    Dim strDMName As String
    Dim iDataSourceOption As DynamoDataSourceOption
    Dim iResult As Long 'UpdateDynamoResult
    Dim iPrompt As Long

    iDataSourceOption = g_WizardConfig.iDataSourceOption
    If g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE Then
        If g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_BUT_NOT_APPLY
        ElseIf g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_AND_APPLY
        ElseIf g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_NOT_UPDATE
        End If
    End If

'////// actual update call
    ' call Update A Dynamo
    PlugandSolve.GeometryHelperObj.Update_A_Dynamo_By_Ref objMasterDynamo, objDynamoInstance, iDataSourceOption,
    ' get a result string
    strReturnMsg = PlugandSolve.GeometryHelperObj.Get_Last_Result_String
    ' get the last user choice from the prompt
    If (g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE) And _
        (g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_NONE) Then
        iPrompt = PlugandSolve.GeometryHelperObj.Get_Last_Prompt_Value
        If (iPrompt = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL) Or _
            (iPrompt = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL) Or _
            (iPrompt = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL) Then
            g_ReturnFromPromptForChoice = iPrompt
        End If
    End If
'////// end of actual update call
```

```
'////// dummy for test
'    g_testcount = g_testcount + 1
'    strDIName = objDynamoInstance.Name
'    strDMName = objMasterDynamo.Name
'    If g_testcount = 3 Then
'        strReturnMsg = ">>>" & strDIName & " was not updated  with " & strDMName
'        iResult = DYNAMO_NOTUPDATED
'    ElseIf g_testcount = 150 Then
'        strReturnMsg = ">>> User canceled"
'        iResult = DYNAMO_UPDATE_ABORTED
'    Else
'        strReturnMsg = strDIName & " was updated successfully with " & strDMName
'        iResult = DYNAMO_UPDATED
'    End If
'////// end of dummy for test

    If iResult <= DYNAMO_UPDATED Then
        UpdateADynamo = DYNAMO_UPDATED
    ElseIf iResult <= DYNAMO_NOTUPDATED Then
        UpdateADynamo = DYNAMO_NOTUPDATED
    Else
        UpdateADynamo = DYNAMO_UPDATE_ABORTED
    End If
End Function
```

▶ To view this code in context:

1. In Classic view, from the WorkSpace menu, select Visual Basic Editor.

   -Or-

   In Ribbon view, on the Home tab, in the WorkSpace group, click Visual Basic Editor.
2. In the tree view, double-click the Project_PlugandSolve folder, and then the Modules folder, and finally the modDynamoUpdater.
3. Search for UpdateADynamo to locate this code.

GetAlarmBackgroundColor Method Example

The following example retrieves the color for each alarm priority in the AlarmSummary object *AlarmSummaryOCX1*.

```
Dim lLow As OLE_COLOR
Dim lMedium As OLE_COLOR
Dim lHigh As OLE_COLOR
lLow = AlarmSummaryOCX1.GetAlarmBackgroundColor(7)
lMedium = AlarmSummaryOCX1.GetAlarmBackgroundColor(6)
lHigh = AlarmSummaryOCX1.GetAlarmBackgroundColor(5)
```

GetAlarmForegroundColor Method Example

The following example retrieves the color for alarms with a HIHI status in the AlarmSummary object *AlarmSummaryOCX1*.

```
Dim lHiHi As OLE_COLOR
```

```
lHiHi = AlarmSummaryOCX1.GetAlarmForegroundColor(2)
```

GetAllConnections Subroutine Example

The following example shows how to use a click event on rectangle object to retrieve all data items within a picture and then display them in a message box.

```
Private Sub Rect2_Click()
   Dim obj
   Dim str As String
   GetAllConnections Me
   For Each obj In AllConnectionsCollection
     str = str & CStr(obj) & vbCr
   Next
   MsgBox str, , Me.Name
End Sub
```

GetBoundRect Method Example

The following example retrieves the points of the bounding rectangle of the Oval object *Oval1*.

```
Dim dTop As Double
Dim dLeft As Double
Dim dBottom As Double
Dim dRight As Double
Oval1.GetBoundRect dTop, dLeft, dBottom, dRight
```

GetChartEndTime Method Example

The following example returns the chart's end time, which is displayed in a text object.

First, insert a object. Then, add an XYChart (XYChart1). Next, add a pushbutton and add the following script to its onclick event. In run mode, click on the pushbutton and the end date will be displayed in the text box.

```
Text1.Caption = XYChart1.GetChartEndTime
```

GetChartStartTime Method Example

The following example returns the chart's start time, which is displayed in a text object.

First, insert a object. Then, add an XYChart (XYChart1). Next, add a pushbutton and add the following script to its onclick event. In run mode, click on the pushbutton and the start date will be displayed in the text box.

```
Text1.Caption = XYChart1.GetChartStartTime
```

GetColHeadings Method Example

The following example retrieves the column headings of the alarm summary object. In this example, the column headings will be returned in the variable *bstrColHeadings*.

```
Dim lErr as Long
Dim bstrColHeadings As String
lErr = AlarmSummaryOCX1.GetColHeadings(bstrColHeadings)
```

GetColumnInfo Method Example

The following example retrieves the item name and number of characters displayed in column *3* for the **Chart** *Chart1.*

```
Dim sItemName As String
Dim iNumChar As Integer
Chart1.GetColumnInfo 3, sItemName, iNumChar
```

GetConnectionInformation Method Example

The following example retrieves the connection information for the first connection for the **Oval** object *Oval1*.

```
Dim sPropName As String
Dim sSource As String
Dim sFullQualSource As String
Dim vSourceObjs As Variant
Dim vTolerance As Variant
Dim vDeadBand As Variant
Dim vUpdateRate As Variant
Oval1.GetConnectionInformation 1, sPropName, sSource, sFullQualSource, vSourceObjs, vTolerance, vDeadBand, vUpdat
```

GetConnectionParameters Method Example

The following example returns the UpdateRate, Deadband and Tolerance for the **ForegroundColor** property connection for the **Oval** object *Oval1.*

```
Dim vTolerance As Variant
Dim vDeadBand As Variant
Dim vUpdateRate As Variant
Dim vFlags As Variant
Oval1.GetConnectionParameters "ForegroundColor",
 vUpdateRate, vDeadBand, vTolerance, vFlags
```

GetContinuousUser Method Example

The following example ensures that the User Name entered is valid, creates the ESignature object, checks that the node is enabled for electronic signature, and sets and gets the name of the continuous user.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bSigRequired As Boolean
Dim UserName As String
Dim SetName As String
'valid user name
SetName = "TestUser"
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignature")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Set the name of the continuous user to "TestUser"
ESig.SetContinuousUser SetName
'Get the name of the continuous user
ESig.GetContinuousUser UserName
MsgBox “Continuous User: ” & UserName
Else
'the hardware key is not enabled for signature, security is disabled, or bypass signature is in effect
MsgBox “Signature is not enabled on this node.”
End If
```

GetCurrentDataSet Method Example

The following example retrieves the object reference to the current dataset for the Line/Multiline chart.

```
Private Sub CommandButton2_Click()
Dim objDataSet As Object
Set objDataSet = LineChart1.GetCurrentDataSet()
' Set the color of the dataset to its complement.
objDataSet.DataSetColor = objDataSet.DataSetColor Xor (-1)
End Sub
```

GetCurrentValueWithQuality Method Example

The following example displays the current value of x, y, and the data quality of the Enhanced Chart in a message box, which are returned using the GetCurrentValueWithQuality method.

```
Private Sub CommandButton11_Click()
Dim objDataSet As Object ' FixRealTimeDataSet.FixRealTimeDataSet
Dim varXValue As Variant, dblYValue As Double, lngQuality As Long
Set objDataSet = LineChart1.GetCurrentDataSet()
objDataSet.GetCurrentValueWithQuality varXValue, dblYValue, lngQuality
MsgBox "AssignedID is " & CStr(objDataSet.AssignedID) & vbCrLf & _
"ResolvedSourceName is " & CStr(objDataSet.ResolvedSourceName) & vbCrLf & _
"Current X Value is " & CStr(varXValue) & vbCrLf & _
"Current X Value is " & CStr(dblYValue) & vbCrLf & _
"Current Quality is " & CStr(lngQuality)
End Sub
```

GetCurrentValue Method Example

The following example retrieves the value, timestamp, and quality information for **Pen***Pen1.*

```
Dim dCurVal As Double
Dim dt As Date
Dim lQual As Long
Pen1.GetCurrentValue dCurVal, dt, lQual
```

GetDataSetByPosition Method Example

The following example shows an example of the GetDataSetByPosition method with the LineChart object.

```
    Dim objDS As Object
    Set objDS = LineChart1.GetDataSetByPosition(2)
Return Value
Object. The dispatch pointer of the Dataset object retrieved by position.
```

GetDecimalSeparator Subroutine Example

The following example retrieves the decimal separator set for the local machine.

```
Dim sDecimal As String
sDecimal = GetDecimalSeparator
```

GetDeviceRect Method Example

The following example retrieves the points of the bounding rectangle of the **Oval** object *Oval1* in device coordinates.

```
Dim dTop As Long
Dim dLeft As Long
Dim dBottom As Long
Dim dRight As Long
Oval1.GetDeviceRect dTop, dLeft, dBottom, dRight
```

GetDuration Method Example

The following example retrieves the duration for the **Chart** *Chart1.*

```
Dim lDays As Long
Dim lHours As Long
Dim lMinutes As Long
Dim lSeconds As Long
Chart1.GetDuration lDays, lHours, lMinutes, lSeconds
```

GetErrorString Method Example

The following is an example of how a user might use GetErrorString to handle errors that occur. In this example, setting Bitmap1.CurrentImage = 4 causes an error, forcing the error handler to take effect.

```
Dim sErrString As String
On Error GoTo errhand
Bitmap1.CurrentImage = 4
errhand:
sErrString = System.GetErrorString(Err.Number)
```

GetEventHandlerIndex Method Example

The following example gets the index number for the **Click** of the object *CurrentObject* and, if a procedure exists, removes it.

```
CurrentObject.Procedures.GetEventHandlerIndex "Click",
 lIndex, lFound
If lFound = 1 Then
CurrentObject.Procedures.Remove lIndex
End If
```

GetFormDynamoColor Subroutine Example

The following example retrieves an instance of the form named DynamoColorBy.

```
Dim frmDynamoColor as Object
GetFormDynamoColor frmDynamoColor
```

GetFormNumeric Subroutine Example

Reserved for internal purposes.

GetFormPushbutton Subroutine Example

Reserved for internal purposes.

GetFormRamp Subroutine Example

Reserved for internal purposes.

GetFormSlider Subroutine Example

Reserved for internal purposes.

GetFullname Method Example

The following example creates the ESignature object, checks to see if the node is enabled for electronic signature, displays the Electronic Signature dialog box, validates the signature and sends a message to the audit trail.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bVerify As Boolean
Dim bContinuousUse As Boolean
Dim bValidSig As Boolean
Dim PerformUserName As String
Dim PerformUserID As String
Dim PerformComment As String
Dim PerformFullName As String
Dim VerifyUserName As String
Dim VerifyUserID As String
Dim VerifyComment As String
Dim VerifyFullName As String
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignatureFactory")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
bVerify = True ' verification is not required
bContinuousUse = True ' allow continuous user, if any, to be displayed in the dialog box
bValidSig = False ' will be set to TRUE by GetSignature if signature is captured successfully
'Display the Electronic Signature dialog box
ESig.GetSignature "Action Description", bVerify, bContinuousUse, bValidSig, Ucase(PerformUserName), Ucase(Perform
If bValidSig = True Then
'Send a message to the audit trail
ESig.SendSignedOperatorMessage "Action Description", "", "", Ucase (PerformUserID), PerformComment, Ucase(VerifyU
'Get the full names of the signers
ESig.GetFullname Ucase(PerformUserID), Ucase(PerformFullName)
ESig.GetFullname Ucase(VerifyUserID), Ucase(VerifyFullName)
'Show Results
MsgBox "Action performed by " + PerformUserName + " (" + PerformFullName + ") " + PerformComment + " and verifie
Else
MsgBox "Signature was not captured."
End If
Else
MsgBox "Signature is not enabled on this node."
End If
```

GetIndirectionInfo Method Example

Reserved for internal purposes.

GetInterval Method Example

The following example retrieves the interval for the **Pen** *Pen1.*

```
Dim lDays As Long
Dim lHours As Long
Dim lMinutes As Long
Dim lSeconds As Long
Pen1.GetInterval lDays, lHours, lMinutes, lSeconds
```

GetNumberOfDataSets Method Example

The following example shows an example of the GetNumberOfDataSets method with the LineChart object.

```
Dim nNumOfDS As Long
nNumOfDS = LineChart1.GetNumberOfDataSets
```

GetLevel Method Example

The following example retrieves the level properties for the **Lookup** object *iLookup* at the level index 1 for the **Oval** object *Oval1*.

```
Dim iLookup As Object
Dim vIn1 as Variant
Dim vOut1 as Variant
Dim vIn2 as Variant
dim vOut2 as Variant
Set iLookup = System.FindObject("TestPicture.Oval1.AnimatedForegroundColor")

iLookup.GetLevel 1, vIn1, vOut1, vIn2, vOut2
```

GetLocaleInfoA Subroutine Example

The following example uses the **GetLocaleInfoA** call to get the decimal separator specified for the local machine. We will also use the API call **GetUserDefaultLCID** to obtain the default locale of the local user. This gives us the first parameter for **GetLocaleInfoA**.

```
Public Function GetDecimalSeparator()
Dim intCountChar As Integer
Dim lngHolder As Long
Dim strDecChar As String
On Error GoTo ErrorHandler
'Set the buffer for the return value
strDecChar = Space$(255)
'Get the decimal separator and the count of characters for
 the thousand 'separator
lngHolder = GetLocaleInfoA(GetUserDefaultLCID(), LOCALE_SDECIMAL,
 strDecChar, Len(strDecChar) + 1)
'Now set the return value to just the decimal separator.

GetDecimalSeparator = Left$(strDecChar, lngHolder - 1)
Exit Function
ErrorHandler:
```

```
HandleError
End Function
```

GetObjectInfo Method Example

The following example retrieves a two dimensional array consisting of the values for *StartTime* and *Interval* for the Timer objects *MyTimer* and *MyTimer1*.

```
Dim iSched As Object
Dim Objects(1) As String
Dim Properties(1) As String
Dim Objinfo As Variant
Objects(0) = "MyTimer"
Objects(1) = "MyTimer1"
Properties(0) = "StartTime"
Properties(1) = "Interval"
Set iSched = System.FindObject("TestSchedule")

objinfo = iSched.GetObjectInfo(objects, properties)
```

GetPenDataArray Method Example

The following example generate a report containing all data contained inside a standard chart, Chart1, into a file.

```
Dim lNumPts As Long
Dim vVal As Variant
Dim vPsa As Variant
Dim vQual As Variant
Dim vMill As Variant
Dim toto As Object
Dim Mypath As String
Dim myfile As String
Dim RTN As String

RTN = ""

If Chart1.Pens.Count <> 0 Then
Mypath = System.BasePath
myfile = Mypath & "\app\ChartReport.txt"
RTN = Dir(myfile)
 If RTN <> "" Then
  'File do exist
  'Delete Output file
  Kill myfile
 End If
Open myfile For Output As #1
For i = 1 To Chart1.Pens.Count
 Chart1.Pens.Item(i).GetPenDataArrayEx lNumPts, vVal, vPsa, vQual, vMill
  For j = 0 To (lNumPts - 1)
   Value = vVal(j)
   Time = vPsa(j)
   Quality = vQual(j)
   Mill = vMill(j)
   Write #1, Chart1.Pens.Item(i).Source; " "; Time; " "; Value; ""
  Next j
 Next i
 Close #1
End If
```

GetPenDataArrayEx Method Example

The following example fetches the data array for the **Pen** *Pen1.*

```
Dim lNumPts As Long
Dim vVal As Variant
Dim vPsa As Variant
Dim vQual As Variant
Dim vMill As Variant
Pen1.GetPenDataArrayEx lNumPts, vVal, vPsa, vQual, vMill
```

GetPointAt Method Example

The following example retrieves the point object at the point index 2 for the **Polygon** object *Polygon1.*

```
Dim iPoint As Object
Dim dX as double
Dim dY as double
Set iPoint = Polygon1.GetPointAt(2)
dX = iPoint.x
dY = iPoint.y
```

GetPriorityColor Method Example

The following example retrieves the color for each alarm priority in the **AlarmSummary** object *AlarmSummaryOCX1*.

```
Dim lLow As OLE_COLOR
Dim lMedium As OLE_COLOR
Dim lHigh As OLE_COLOR
lLow = AlarmSummaryOCX1.GetPriorityColor(7)
lMedium = AlarmSummaryOCX1.GetPriorityColor(6)
lHigh = AlarmSummaryOCX1.GetPriorityColor(5)
```

GetProcedureIndex Example

The following example finds the index of the procedure Rect2_DisplayMessage in the ActiveDocument's project.

```
Dim o As Object
Dim oProc As Object
Dim lIndex As Long
Dim lFound As Long
Set o = Application.ActiveDocument.Page.FindObject("Rect2")

o.Procedures.GetProcedureIndex "DisplayMessage",
 lIndex, lFound
If (lFound) Then
Set oProc = o.Procedures.Item(lIndex)
End If
```

GetProperty Method Example

The following example retrieves the value of the **Caption** property of the **Text** object *Text1.*

```
Dim vValue As Variant
Text1.GetProperty "Caption", vValue
The following example retrieves the value of the FillStyle property of the Rectangle object Rect1.
Dim vValue As Variant
Rect1.GetProperty "FillStyle", vValue
```

GetPropertyAttributes Method Example

The following example fetches the attribute information for the *HighEGU* attribute of the *AI1* block on node *NODE1.*

```
Dim vtResults
Dim vtAttributes
Dim lStatus As Long
Dim strLoEGU as String
Dim LoEGUval
Oval1.GetPropertyAttributes "FIX32.NODE1.AI1.F_CV", 3, vtResults, vtAttributes, lStatus
strLoEGU = vtAttributes(0)
LoEGUval = vtResults(0)
```

In the above example, the variable **strLoEGU** will now hold the string "FIX32.NODE1.AI1.A_ELO" and the variable **LoEGUval** will hold tag AI1's low EGU value.

GetPropertyTargets Method Example

The following example determines which objects are connected to a specific tag reference.

```
Dim sPropName As String
Dim vtTargets As Variant
Oval1.GetPropertyTargets 1, sPropName, vtTargets
```

GetSelectedAlmExt Method Example

The following example retrieves the alarm extensions configured for the currently selected alarm in the AlarmSummary object *AlarmSummaryOCX1*.

```
Dim sExt1 As String
Dim sExt2 As String
AlarmSummaryOCX1.GetSelectedAlmExt sExt1, sExt2
```

GetSelectedNodeTag Method Example

The following example retrieves the node and tag name corresponding to the currently selected alarm in the AlarmSummary object *AlarmSummaryOCX1*.

```
Dim sNode As String
Dim sTag As String
AlarmSummaryOCX1.GetSelectedNodeTag sNode, sTag
```

GetSelectedRow Method Example

The following example retrieves the information for the selected alarm in the AlarmSummary object *AlarmSummaryOCX1*.

```
Dim bAck As Boolean
Dim sHandle As String
Dim sArea As String
Dim sDateIn As String
Dim sDateLast As String
Dim sTimeIn As String
Dim sTimeLast As String
Dim sNode As String
Dim sTag As String
Dim sPriority As String
Dim sStatus As String
Dim sDesc As String
Dim sValue As String
Dim sExt1 As String
Dim sExt2 As String
Dim sUser1 As String
Dim sUser2 As String
AlarmSummaryOCX1.GetSelectedRow bAck, sHandle, sArea, sDateIn,
 sDateLast, sTimeIn, sTimeLast, sNode, sTag, sPriority, sStatus, sDesc,
 sValue, sExt1, sExt2, sUser1, sUser2
```

GetSelectedRowAlarmInfo Method Example

The following example obtains the alarm information (including shelving) for the selected alarm in the Alarm Summary object, *AlarmSummaryOCX1*. It then displays a message with the Shelve Status and Remaining Duration.

```
Public Sub HandleGetRowAlarmShelveInfo()
Dim AppObj As Object
Dim PictureObj As Object
Dim CurrentObj As Object
Dim bAck As Boolean
Dim sHandle As String
Dim sArea As String
Dim sDateIn As String
Dim sDateLast As String
Dim sTimeIn As String
Dim sTimeLast As String
Dim sNode As String
Dim sTag As String
Dim sPriority As String
Dim sStatus As String
Dim sDesc As String
Dim sValue As String
Dim sExt1 As String
Dim sExt2 As String
Dim sUser1 As String
Dim sUser2 As String
Dim Shelvable As String
Dim ShelveRemDuration As String
Dim Reserved1 As String
Dim Reserved2 As String
If TypeName(Application) = "CFixApp" Then
Set AppObj = Application
Else
Set AppObj = App
If AppObj Is Nothing Then
Exit Sub
End If
End If
Set PictureObj = AppObj.ActiveDocument
For Each CurrentObj In PictureObj.Page.ContainedObjects
If TypeName(CurrentObj) = "AlarmSummaryOCX" Then
If CurrentObj.Name = "AlarmSummaryOCX1" Then
CurrentObj.GetSelectedRowAlarmInfo bAck, sHandle, sArea, sDateIn, sDateLast, sTimeIn, sTimeLast, sNode, sTag, sPr
MsgBox "Shelvable Status: " & Shelvable
MsgBox "Remaining Duration: " & ShelveRemDuration
End If
End If
Next
End Sub
```

GetSelectedRowsAlarmInfo Method Example

The following example obtains the alarm information (including shelving) for the for the selected alarms in the **Alarm Summary** object, *AlarmSummaryOCX1*. It then displays a message with the Shelve Status and Remaining Duration for each alarm.

```
Public Sub HandleMultipleRowAlarmShelveInfo()
Dim AppObj As Object
Dim PictureObj As Object
Dim CurrentObj As Object

Dim NumRows As Long
Dim bAck As Variant
Dim sHandle As Variant
Dim sArea As Variant
Dim sDateIn As Variant
Dim sDateLast As Variant
Dim sTimeIn As Variant
Dim sTimeLast As Variant
Dim sNode As Variant
Dim sTag As Variant
Dim sPriority As Variant
Dim sStatus As Variant
Dim sDesc As Variant
Dim sValue As Variant
Dim sExt1 As Variant
Dim sExt2 As Variant
Dim sUser1 As Variant
Dim sUser2 As Variant
Dim FlexData1 As Variant
Dim FlexData2 As Variant
Dim FlexData3 As Variant
Dim FlexData4 As Variant
Dim FlexData5 As Variant
Dim FlexData6 As Variant
Dim FlexData7 As Variant
Dim FlexData8 As Variant
Dim Shelvable As Variant
Dim ShelveRemDuration As Variant
Dim Reserved1 As Variant
Dim Reserved2 As Variant
If TypeName(Application) = "CFixApp" Then
Set AppObj = Application
Else
Set AppObj = App
If AppObj Is Nothing Then
Exit Sub
End If
End If
Set PictureObj = AppObj.ActiveDocument
For Each CurrentObj In PictureObj.Page.ContainedObjects
If TypeName(CurrentObj) = "AlarmSummaryOCX" Then
If CurrentObj.Name = "AlarmSummaryOCX1" Then
CurrentObj.GetSelectedRowsAlarmInfo NumRows, bAck, sHandle, sArea, sDateIn, sDateLast, sTimeIn, sTimeLast, sNode,
'                MsgBox "Shelvable Status: " & Shelvable
'                MsgBox "Remaining Duration: " & ShelveRemDuration
Exit Sub
End If
End If
Next
End Sub
```

GetSelectedUserDefFields Method Example

The following example retrieves value of the "A_" fields configured for the currently selected alarm in the AlarmSummary object *AlarmSummaryOCX1*.

```
Dim sUserDef1 As String
Dim sUserDef2 As String
AlarmSummaryOCX1.GetSelectedUserDefFields sUserDef1, sUserDef2
```

GetSignature Method Example

The following example creates the ESignature object, checks to see if the node is enabled for electronic signature, displays the Electronic Signature dialog box, validates the signature and sends a message to the audit trail.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bVerify As Boolean
Dim bContinuousUse As Boolean
Dim bValidSig As Boolean
Dim PerformUserName As String
Dim PerformUserID As String
Dim PerformComment As String
Dim PerformFullName As String
Dim VerifyUserName As String
Dim VerifyUserID As String
Dim VerifyComment As String
Dim VerifyFullName As String
Dim bPerformCommentRequired As Boolean

'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignatureFactory")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
bVerify = True ' verification is not required
bContinuousUse = True ' allow continuous user, if any, to be displayed in the dialog box
bValidSig = False ' will be set to TRUE by GetSignature if signature is captured successfully
bPerformCommentRequired = True  'require that a perform comment be entered before the electronic signature can be
'Display the Electronic Signature dialog box
ESig.GetSignature "Action Description", bVerify, bContinuousUse, bValidSig, PerformUserName, PerformUserID, Perfo

If bValidSig = True Then
'Send a message to the audit trail
ESig.SendSignedOperatorMessage "Action Description", "", "", Ucase (PerformUserID), PerformComment, Ucase(VerifyU
ESig.GetFullname Ucase(PerformUserID), Ucase(PerformFullName)
ESig.GetFullname Ucase(VerifyUserID), Ucase(VerifyFullName)
'Show Results
MsgBox "Action performed by " + PerformUserName + " (" + PerformFullName + ") " + PerformComment + " and verified
Else
MsgBox "Signature was not captured."
End If
Else
MsgBox "Signature is not enabled on this node."
End If
```

GetSignatureAndWriteValue Method Example

The following example creates the ESignature object, checks to see if the node is enabled for electronic signature, determines if the tag FIX32.thisnode.D01.F_CV requires electronic signature, captures the signature, writes the new value, and sends a message to the audit trail.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bSigRequired As Boolean
Dim NewValue As Variant
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignatureFactory")

'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
```

```
If bNodeSignEnabled = True Then
'Check if tag requires electronic signature
ESig.Initialize "Fix32.thisnode.DO1.F_CV"
ESig.IsSignatureRequired 0, bSigRequired
If bSigRequired = True Then
'Capture the signature, write the new value and send a message
 to the audit trail
NewValue = 1
ESig.GetSignatureAndWriteValue 0, NewValue
Else
MsgBox "Signature is not required for this tag."

End If
Else
MsgBox "Signature is not enabled on this node."

End If
```

GetStatusColor Method Example

The following example retrieves the color for alarms with a HIHI status in the AlarmSummary object *AlarmSummaryOCX1*.

```
Dim lHiHi As OLE_COLOR
lHiHi = AlarmSummaryOCX1.GetStatusColor(2)
```

GetStatusFont Method Example

The following example retrieves the font properties for alarms with a LO status in the AlarmSummary object *AlarmSummaryOCX1*.

```
Dim sFontName As String
Dim bStrike As Boolean
Dim bUnderline As Boolean
Dim bBold As Boolean
Dim bItalic As Boolean
Dim iSize As Integer
sFontName = AlarmSummaryOCX1.GetStatusFont(3, bStrike, bUnderline,
 bBold, bItalic, iSize)
```

GetTimeBeforeNow Method Example

The following example retrieves the initial relative start time for the Chart object *Chart1.*

```
Dim lHours As Long
Dim lMinutes As Long
Dim lSeconds As Long
Chart1.GetTimeBeforeNow lHours, lMinutes, lSeconds
```

GetTimeCursorInfo Method Example

The following example fetches the time, value and quality of the trend at the time where the **Pen** at index 1 for the **Chart***Chart1* crosses the time cursor.

```
Dim Dt As Date
Dim dVal As Double
Dim lQual As Long
Chart1.GetTimeCursorInfo 1, Dt, dVal, lQual
```

GetUserDefaultLCID Subroutine Example

The following example uses the **GetLocaleInfoA** call to get the decimal separator specified for the local machine. We will also use the API call **GetUserDefaultLCID** to obtain the default locale of the local user. This gives us the first parameter for **GetLocaleInfoA**.

```
Public Function GetDecimalSeparator()
Dim intCountChar As Integer
Dim lngHolder As Long
Dim strDecChar As String
On Error GoTo ErrorHandler
'Set the buffer for the return value
strDecChar = Space$(255)
'Get the decimal separator and the count of characters for the thousand 'separator
lngHolder = GetLocaleInfoA(GetUserDefaultLCID(), LOCALE_SDECIMAL, strDecChar, Len(strDecChar) + 1)
'Now set the return value to just the decimal separator.
GetDecimalSeparator = Left$(strDecChar, lngHolder - 1)
Exit Function
ErrorHandler:
HandleError
End Function
```

GetUserID Method Example

The following example creates the ESignature object, checks to see if the node has electronic signature enabled, obtains the user ID, checks to see if the user has access to the Electronic Signature Bypass application feature, and checks to see if the user has access to a security area.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim UserName As String
Dim PassWord As String
Dim result As Boolean
Dim UsrId As String
'valid iFix user name and password
UserName = "username"
PassWord = "password"
result = False
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignature")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Get the user id
ESig.GetUserid UserName, PassWord, UsrId
MsgBox "UserID: " & UsrId
'Check if user has access to Electronic Signature Bypass application feature
```

```
ESig.CheckUserApplicationAccess UsrId, 74, result
MsgBox "User: " & UserName & vbCr & "Has rights to Bypass Signature? " & result
'Check if user has access to security area B
ESig.CheckUserAreaAccess UsrId, "B", result
MsgBox "User: " & UserName & vbCr & "Has rights to Security Area B? " & result
Else
'the hardware key is not enabled for signature, security is disabled, or bypass signature is in effect
MsgBox “Signature is not enabled on this node.”
End If
```

GetWindowLocation Method Example

The following example retrieves the window size and location of the Picture*TestPicture* in terms of percentage of the screen.

```
Dim dTopPct As Double
Dim dLeftPct As Double
Dim dHeightPct As Double
Dim dWidthPct As Double
TestPicture.GetWindowLocation dTopPct, dLeftPct, dHeightPct,
 dWidthPct
```

GlobalScrollBackFast Method Example

The following example sets the System.GlobalStartTime and System.GlobalEndTime properties and scrolls backward by the amount specified in the GlobalFastScrollRate property.

```
Private Sub FastLeftButton_Click()
System.GlobalScrollBackFast
End Sub
```

GlobalScrollBackSlow Method Example

The following example sets the System.GlobalStartTime and System.GlobalEndTime properties and scrolls backward by the amount specified in the GlobalSlowScrollRate property.

```
Private Sub LeftButton_Click()
System.GlobalScrollBackSlow
End Sub
```

GlobalScrollForwardFast Method Example

The following example sets the System.GlobalStartTime and System.GlobalEndTime properties and scrolls forward by the amount specified in the GlobalFastScrollRate property.

```
Private Sub FastRightButton_Click()
System.GlobalScrollForwardFast
End Sub
```

GlobalScrollForwardSlow Method Example

The following example sets the System.GlobalStartTime and System.GlobalEndTime properties and scrolls forward by the amount specified in the GlobalSlowScrollRate property.

```
RightButton_Click()
System.GlobalScrollForwardSlow
End Sub
```

GlobalTimerApply Method Example

The following example shows how to use the Global Time Control and Playback properties.

```
Private Sub DTPicker7_Change()
System.GlobalStartTime = DTPicker7.Value
System.GlobalDuration = 3600
System.GlobalPlayBack = True
System.GlobalPlayBackFrameSize = 600
System.GlobalTimerApply
'The above settings will enable playback of 1 hour data showing 10 minutes of data each time.
End Sub
```

Group Method Example

The following example forms a group consisting of the Oval and Polygon objects *Oval1* and *Polygon1.*

```
Oval1.SelectObject False
Polygon1.SelectObject False
TestPicture.Group
```

H

HandleError Subroutine Example

The following example demonstrates how the **HandleError** subroutine would be used in trapping errors.

```
Public Function GetDecimalSeparator()
Dim intCountChar As Integer
Dim lngHolder As Long
Dim strDecChar As String
On Error GoTo ErrorHandler
'Set the buffer for the return value
strDecChar = Space$(255)
'Get the decimal separator and the count of characters for
 the thousand 'separator
lngHolder = GetLocaleInfoA(GetUserDefaultLCID(), LOCALE_SDECIMAL,
 strDecChar, Len(strDecChar) + 1)
'Now set the return value to just the decimal separator.

GetDecimalSeparator = Left$(strDecChar, lngHolder - 1)
```

```
Exit Function
ErrorHandler:
HandleError
End Function
```

HiLoDisplay Method Example

The following example sets the HiDisplay and LoDisplay properties of the TimeAxis object for the Pen*Pen1* to 11/15/98 and 10/17/98 respectively.

```
Dim iTimeAxis as object
set iTimeAxis = Pen1.TimeAxis
iTimeAxis.HiLoDisplay #11/15/98 12:00:00AM#, #10/17/98 12:00:00AM#
```

I-K

ImportToolbar Method Example

The following example imports a ToolBar named *Toolbar1* that is owned by the iFIX WorkSpace.

```
Dim lErr as Long
lErr = Application.ToolbarManager.ImportToolbar(“Toolbar1”,
 “WorkSpace”)
```

Initialize Method Example

The following example creates the ESignature object, checks to see if the node is enabled for electronic signature, and determines if a specified tag (FIX32.thisnode.D01.F_CV) requires electronic signature. If required, the example validates the signature, writes the new value, and sends a message to the audit trail.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bSigRequired As Boolean
Dim bVerify As Boolean
Dim bContinuousUse As Boolean
Dim nInfo As Integer
Dim NewValue As Variant
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignatureFactory")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Check if tag requires electronic signature
ESig.Initialize "Fix32.thisnode.DO1.F_CV"
ESig.IsSignatureRequired 0, bSigRequired, nInfo, bVerify, bContinuousUse
If bSigRequired = True Then
'Validate the signature, write the new value and send a message to the audit trail
NewValue = 1
```

```
If bVerify = False Then
ESig.ValidateSignatureAndWriteValue 0, NewValue, "admin", "admin", "Perform Comment Example"
Else
ESig.ValidateSignatureAndWriteValue 0, NewValue, "admin", "admin", "Perform Comment Example", "supervisor1", "GE
End If
Else
MsgBox "Signature is not required for this tag."
End If
Else
MsgBox "Signature is not enabled on this node."
End If
```

InitializeList Method Example

The following example creates the ESignature object, checks to see if the node is enabled for electronic signature and determines if a set of tags require electronic signature.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bSigRequired As Boolean
Dim bValidSig As Boolean
Dim info As Integer
'Create the list of data sources
Dim DataSources As Variant
ReDim DataSources(2) As String
DataSources(0) = "Fix32.THISNODE.TAG1.F_CV"
DataSources(1) = "Fix32.THISNODE.TAG2.F_CV"
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignature")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'initialize the object and set data sources
ESig.InitializeList DataSources
'Is signature required for data sources
ESig.IsSignatureRequiredForList 4, bSigRequired, info
If bSigRequired Then
'signature is required for this list
MsgBox “Signature is required.”
Else
'signature is not required for this list, check info parameter for reason
MsgBox “Signature is not required.”
End If
Else
'the hardware key is not enabled for signature, security is disabled, or bypass signature is in effect
MsgBox “Signature is not enabled on this node.”
End If
```

InsertPoint Method Example

The following example inserts the point 55,75 at an index of 3 for the **Polyline** object *Polyline1*.

```
Dim iPoint As FixFloatPoint
Set iPoint = New FixFloatPoint
iPoint.x = 55
iPoint.y = 75
PolyLine1.InsertPoint 3, iPoint
```

InteractiveExport Method Example

The following example launches the Export dialog box for the Line/Multiline chart.

```
Private Sub CommandButton4_Click()
LineChart1.InteractiveExport
End Sub
```

IsColorSelectionVisible Method Example

The following example determines whether the WorkSpace's Color Selection dialog box is open.

```
Dim bIsVisible As Boolean
bIsVisible = TestPicture.IsColorSelectionVisible
```

IsConnected Method Example

The following example determines whether the **VerticalFillPercentage** property of the **Oval** object *Oval1* has a connection assigned to it.

```
Dim bIsConnected As Boolean
Dim lIndex As Long
Dim lStatus As Long
Oval1.IsConnected "VerticalFillPercentage", bIsConnected,
 lIndex, lStatus
```

IsEmpty Method Example

The following example determines if the **Lookup** object *iLookup* is empty.

```
Dim bIsEmpty As Boolean
iLookup.IsEmpty bIsEmpty
```

IsNodeSignEnabled Method Example

The following example creates the ESignature object, checks to see if the node is enabled for electronic signature, and determines if a specified tag (FIX32.thisnode.D01.F_CV) requires electronic signature. If required, the example validates the signature, writes the new value, and sends a message to the audit trail.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bSigRequired As Boolean
Dim bVerify As Boolean
Dim bContinuousUse As Boolean
Dim nInfo As Integer
Dim NewValue As Variant
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignatureFactory")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Check if tag requires electronic signature
ESig.Initialize "Fix32.thisnode.DO1.F_CV"
ESig.IsSignatureRequired 0, bSigRequired, nInfo, bVerify, bContinuousUse
If bSigRequired = True Then
'Validate the signature, write the new value and send a message to the audit trail
NewValue = 1
If bVerify = False Then
ESig.ValidateSignatureAndWriteValue 0, NewValue, "admin", "admin", "Perform Comment Example"
Else
ESig.ValidateSignatureAndWriteValue 0, NewValue, "admin", "admin", "Perform Comment Example", "supervisor1", "GE
End If
Else
MsgBox "Signature is not required for this tag."
End If
Else
MsgBox "Signature is not enabled on this node."
End If
```

IsSignatureRequired Method Example

The following example creates the ESignature object, checks to see if the node is enabled for electronic signature, and determines if a specified tag (FIX32.thisnode.D01.F_CV) requires electronic signature. If required, the example validates the signature, writes the new value, and sends a message to the audit trail.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bSigRequired As Boolean
Dim bVerify As Boolean
Dim bContinuousUse As Boolean
Dim nInfo As Integer
Dim NewValue As Variant
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignatureFactory")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Check if tag requires electronic signature
ESig.Initialize "Fix32.thisnode.DO1.F_CV"
ESig.IsSignatureRequired 0, bSigRequired, nInfo, bVerify, bContinuousUse
If bSigRequired = True Then
'Validate the signature, write the new value and send a message to the audit trail
NewValue = 1
If bVerify = False Then
ESig.ValidateSignatureAndWriteValue 0, NewValue, "admin", "", "Perform Comment Example"
Else
ESig.ValidateSignatureAndWriteValue 0, NewValue, "admin", "", "Perform Comment Example", "supervisor1", "GEF", "
End If
Else
MsgBox "Signature is not required for this tag."
End If
```

```
Else
MsgBox "Signature is not enabled on this node."
End If
```

IsSignatureRequiredForList Method Example

The following example creates the ESignature object, checks to see if the node is enabled for electronic signature and determines if a set of tags require electronic signature.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bSigRequired As Boolean
Dim bValidSig As Boolean
Dim info As Integer
'Create the list of data sources
Dim DataSources As Variant
ReDim DataSources(2) As String
DataSources(0) = "Fix32.THISNODE.TAG1.F_CV"
DataSources(1) = "Fix32.THISNODE.TAG2.F_CV"
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignature")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'initialize the object and set data sources
ESig.InitializeList DataSources
'Is signature required for data sources
ESig.IsSignatureRequiredForList 4, bSigRequired, info
If bSigRequired Then
'signature is required for this list
MsgBox “Signature is required.”
Else
'signature is not required for this list, check info parameter for reason
MsgBox “Signature is not required.”
End If
Else
'the hardware key is not enabled for signature, security is disabled, or bypass signature is in effect
MsgBox “Signature is not enabled on this node.”
End If
```

IsUserFxg Subroutine Example

The following example determines if the active document is the **UserGlobals** page.

```
Dim bUserFxg As Boolean
bUserFxg = IsUserFxg
```

Item Method Example

The following example retrieves the Item object associated with index 2 in the **Procedures** collection for the object *CurrentObject*.

```
Dim iItem As Object
Set iItem = CurrentObject.Procedures.Item(2)
```

The following example retrieves the Item object associated with index 3 in the **Lines** collection for the second item in the **Procedures** collection for the object *CurrentObject*.

```
Dim iItem As Object
Set iItem = CurrentObject.Procedures.Item(2).Lines.Item(3)
```

L

ListEvents Method Example

The following example retrieves the number and the corresponding list of events for the **Oval** object *Oval1*.

```
Private Sub Oval1_Click()
Dim strMsg As String
Dim pvEvents As Variant
Dim iNumEvents As Integer
strMsg = "This object has the following events: "
Oval1.ListEvents pvEvents, iNumEvents
For Each i In pvEvents
    strMsg = strMsg + i + ","
    Next
MsgBox strMsg, vbOKOnly, "Message"
End Sub
```

ListMethods Method Example

The following example retrieves the number and the corresponding list of methods for the **Oval** object *Oval1.*

```
Dim strMsg As String
Dim pvMethods As Variant
Dim iNumMethods As Integer
Oval1.ListMethods pvMethods, iNumMethods
strMsg = "This object has the following Methods: "
For Each i In pvMethods
   strMsg = strMsg + i + ","
  Next
MsgBox strMsg, vbOKOnly, "Message"
```

ListProperties Method Example

The following example retrieves the number and the corresponding list of properties and their datatypes for the **Oval** object *Oval1.*

```
Dim strMsg As String
Dim pvProperties As Variant
Dim pvDataTypes As Variant
Dim iNumProps As Integer
Oval1.ListProperties pvProperties, pvDataTypes, iNumProps
strMsg = "This object has the following Properties: "
For Each i In pvProperties
   strMsg = strMsg + i + ","
  Next
MsgBox strMsg, vbOKOnly, "Message"
```

ListWindowsGroupNames Method Example

The following example retrieves a string aray of Windows group names for the SecuritySynchronizer object *objSecSynch.*

```
Dim objSecSynch As SecuritySynchronizer
Set objSecSynch = New SecuritySynchronizer
Dim bNT4NamesOnly As Boolean
bNT4NamesOnly = True
objSecSynch.ListWindowsGroupNames bNT4NamesOnly
```

Setting the *bNT4NamesOnly* variable to **True** returns only Windows group names that do not exceed twenty characters.

Load_TS_List Method Example

This example loads the tag status list *TAGSTATUS* into the current picture when a button is pressed.

```
Private Sub Button_LoadTSList_Click()

    Me.Load_TS_List TAGSTATUS

End Sub
```

LoadImage Method Example

The following example loads the image *CustomButton10* as the primary image at index 1 for the Bimtap object *Bitmap1*.

```
Bitmap1.LoadImage True, 1, "C:\Program Files (x86)\Proficy\iFIX\Local\CustomButton10.bmp"
```

LoadTagGroupFile Example

The following example loads the tag group file *mytaggroup.tgd* into the iFIX picture *pic1*.

```
Private Sub CommandButton1_Click()
pic1.LoadTagGroupFile "mytaggroup.tgd"
```

```
End Sub
```

LocateObject Subroutine Example

The following example finds the object *Tank1* in all open documents.

```
LocateObject "Tank1", False
```

LogicalToPercentage Method Example

NOTE: The LogicalToPercentage Method Example applies to both Enhanced and Logical Coordinates.

The following example converts the Top, Left, Height and Width coordinates from 30, 30, 100, 150 in logical units to percentage of screen space available for the **Picture***TestPicture*.

```
Dim dTop As Double
Dim dLeft As Double
Dim dHeight As Double
Dim dWidth As Double
dTop = 30
dLeft = 30
dHeight = 100
dWidth = 150
TestPicture.LogicalToPercentage dTop, dLeft, dHeight, dWidth
```

LogicalToUserFormPoint Method Example

NOTE: The LogicalToUserFormPoint Method Example applies to both Enhanced and Logical Coordinates.

The following example converts the Top, Left coordinates from 30, 50 in logical units or postscript points to "UserForm Point" coordinates for the **Picture***TestPicture*.

```
Dim dTop As Double
Dim dLeft As Double
dTop = 30
dLeft = 50
TestPicture.LogicalToUserFormPoint dTop, dLeft
```

LogIn Subroutine Example

The following example opens the login dialog, allowing the user to perform login/logout procedures.

```
LogIn (0) , (False) (by default)
```

M-N

MakeLinesHorizontal Method Example

The following example shows how to convert the selected **line object** into horizontal line(s).

```
Set ObjHelper = BuildObject("GeometryHelper")
Call ObjHelper.MakeLinesHorizontal
```

MakeLinesVertical Method Example

The following example shows how to convert the selected **line object(s)** into vertical line(s).

```
Set ObjHelper = BuildObject("GeometryHelper")
Call ObjHelper.MakeLinesVertical
```

MakeSameSize Method Example

The following example selects the **Polygon** and **Oval** objects, *Polygon1* and *Oval1,* and sets them to have the same width.

```
Polygon1.SelectObject False
Oval1.SelectObject False
TestPicture.MakeSameSize 1
```

Modify Method Example

The following example displays the Modify Block dialog box for *AI1* of the **Bitmap** object *Bitmap1*.

```
Dim bReadOnly As Boolean
Dim iStatus As Long
Bitmap1.Modify "Fix32.NODE1.AI1", bReadOnly, iStatus
```

ModifyColumnLength Method Example

The following example sets the number of characters to be displayed in column 1 of the **Chart** *Chart1* to 7.

```
Chart1.ModifyColumnLength 1, 7
```

Move Method Example

The following example moves the object *Tank1* in the horizontal direction by an offset of 10 and in the vertical direction by an offset of 25.

```
Tank1.Move 10, 25
```

NewAlarm Event Example

The following example parses a list of nodes and tags for the NewAlarm event:

```
' parses through the list of new alarms (there could be more
 than one)
Private Sub AlarmSummaryOCX1_NewAlarm(strNode As String,
 strTag As String)
Dim lngNumAlarms&, lngLoop&, strThisTag$, strThisNode$

' compute the number of new alarms being sent to us
lngNumAlarms = Len(strTag) / 30
For lngLoop = 1 To lngNumAlarms&
' get the next node and tag
strThisNode = Mid(strNode, ((lngLoop - 1) * 8) + 1, 8)
strThisNode = Trim(strThisNode)
strThisTag = Mid(strTag, ((lngLoop - 1) * 30) + 1, 30)
strThisTag = Trim(strThisTag)
Next lngLoop
End Sub
```

O

OffScan Subroutine Example

The following example places the block *AI1* off scan.

```
OffScan "AI1"
```

OnScan Subroutine Example

The following example places the block *AI1* on scan.

```
OnScan "AI1"
```

Open Method Example

The following example opens the **Picture** *TestPicture*, displaying it normally.

```
Dim iDoc As Object
Set iDoc=Application.Documents.Open("C:\Program Files (x86)\Proficy\iFIX\Pic\TestPicture.grf",3)
```

Open_QT_Pic Method Example

The following example opens a Quick Trend Picture when a button is pressed. The tags that are displayed are retrieved from the currently selected objects.

```
Private Sub Button_OpenQTPic_Click()

    Me.Open_QT_Pic

End Sub
```

Open_QT_Pic_Ex Method Example

The following example opens a Quick Trend Picture when a button is pressed. The tags that are displayed are retrieved from the currently selected objects.

```
Private Sub Button_OpenQTPic_Click()

    Me.Open_QT_Pic_Ex (1)

End Sub
```

Open_TCP_Pic Method Example

The following example opens the Tag Control Panel Picture when a button is pressed. The tags that are displayed are retrieved from the currently selected objects.

```
Private Sub Button_OpenTCPPic_Click()

    Me.Open_TCP_Pic

End Sub
```

Open_TCP_Pic_Ex Method Example

The following example opens the Tag Control Panel Picture when a button is pressed. The tags that are displayed are retrieved from the currently selected objects.

```
Private Sub Button_OpenTCPPic_Click()

    Me.Open_TCP_Pic_Ex (1)

End Sub
```

Open_TS_Pic Method Example

The following example opens the Tag Status Picture for the first found tag for the object *Object1* when a button is pressed. The tags that are displayed are retrieved from the currently selected objects.

```
Private Sub Button_OpenTSPic_Click()

    Object1.Open_TS_Pic

End Sub
```

Open_TS_Pic_Ex Method Example

The following example opens the Tag Status Picture for the first found tag for the object *Object1* when a button is pressed. The tags that are displayed are retrieved from the currently selected objects.

```
Private Sub Button_OpenTSPic_Click()

    Object1.Open_TS_Pic_Ex (1)

End Sub
```

Open_TS_Pic_Type Method Example

The following example opens the Tag Status Picture *TAGSTATUS* and displays the tag list AI, AO, and DI when a button is pressed.

```
Private Sub Button_OpenTSPicType_TagStatus_Click()
    Dim SomeStrings(2) As String
    Dim OpenedPic As Object

    SomeStrings(0) = "FIX32.THISNODE.AI.A_NAME"
    SomeStrings(1) = "FIX32.THISNODE.AO.A_NAME"
    SomeStrings(2) = "FIX32.THISNODE.DI.A_NAME"

    Set OpenedPic = Me.Open_TS_Pic_Type(TAGSTATUS, SomeStrings)

End Sub
```

The following example opens the Quick Trend Picture *QUICKTREND* into the current picture and displays the tag list AI, AO, and DI when a button is pressed.

```
Private Sub Button_OpenTSPicType_QT_Click()
    Dim SomeStrings(2) As String

    SomeStrings(0) = "FIX32.THISNODE.AI.A_NAME"
    SomeStrings(1) = "FIX32.THISNODE.AO.A_NAME"
    SomeStrings(2) = "FIX32.THISNODE.DI.A_NAME"

    Me.Open_TS_Pic_Type QUICKTREND, SomeStrings

End Sub
```

The following example opens the Tag Control Panel Picture *TAGCONTROLPANEL* into the current picture and displays the tag list AI, AO, and DI when a button is pressed.

```
Private Sub Button_OpenTSPicType_TCP_Click()
    Dim SomeStrings(2) As String
    SomeStrings(0) = "FIX32.THISNODE.AI.A_NAME"
    SomeStrings(1) = "FIX32.THISNODE.AO.A_NAME"
    SomeStrings(2) = "FIX32.THISNODE.DI.A_NAME"

    Me.Open_TS_Pic_Type TAGCONTROLPANEL, SomeStrings

End Sub
```

Open_TS_Pic_Type_Ex Method Example

The following example opens the Tag Status Picture *TAGSTATUS* and displays the tag list AI, AO, and DI when a button is pressed. If one instance of this picture is already open, another instance displays.

```
Private Sub Button_OpenTSPicType_TagStatus_Click()
    Dim SomeStrings(2) As String
    Dim OpenedPic As Object

    SomeStrings(0) = "FIX32.THISNODE.AI.A_NAME"
    SomeStrings(1) = "FIX32.THISNODE.AO.A_NAME"
    SomeStrings(2) = "FIX32.THISNODE.DI.A_NAME"

    Set OpenedPic = Me.Open_TS_Pic_Type_Ex(TAGSTATUS, SomeStrings, 1)

End Sub
```

The following example opens the Quick Trend Picture *QUICKTREND* into the current picture and displays the tag list AI, AO, and DI when a button is pressed. If one instance of this picture is already open, another instance displays.

```
Private Sub Button_OpenTSPicType_QT_Click()
    Dim SomeStrings(2) As String

    SomeStrings(0) = "FIX32.THISNODE.AI.A_NAME"
    SomeStrings(1) = "FIX32.THISNODE.AO.A_NAME"
    SomeStrings(2) = "FIX32.THISNODE.DI.A_NAME"

    Me.Open_TS_Pic_Type_Ex QUICKTREND, SomeStrings, 1

End Sub
```

The following example opens the Tag Control Panel Picture *TAGCONTROLPANEL* into the current picture and displays the tag list AI, AO, and DI when a button is pressed. If one instance of this picture is already open, another instance displays.

```
Private Sub Button_OpenTSPicType_TCP_Click()
    Dim SomeStrings(2) As String

    SomeStrings(0) = "FIX32.THISNODE.AI.A_NAME"
    SomeStrings(1) = "FIX32.THISNODE.AO.A_NAME"
    SomeStrings(2) = "FIX32.THISNODE.DI.A_NAME"

    Me.Open_TS_Pic_Type_Ex TAGCONTROLPANEL, SomeStrings, 1

End Sub
```

OpenDigitalPoint Subroutine Example

The following example opens the block *DO1.*

```
OpenDigitalPoint "DO1"
```

OpenPicture Subroutine Example

The following example opens the **Picture** *TestPicture*, giving it an alias of *Test* and positioning its top left corner at (75,75).

```
OpenPicture "TestPicture" ,"Test", 75, 75
```

The following example opens another instance of a picture if it is already open in the iFIX WorkSpace:

```
OpenPicture "TestPicture", , , , , , , , True
```

The following example opens another instance of a picture if it is already open in the iFIX WorkSpace, but with a different tag group file (named taglist3):

```
OpenPicture "TestPicture", , , , , , , taglist3, True
```

The next example shows the OpenPicture subroutine using the optional intErrormode parameter, with a value of 0:

```
OpenPicture "BadPic", , , , 0
```

When you use 0 for the intErrorMode, if you try to open a picture that does not exist, a message box appears whose title is the name of the picture that made the erroneous call and whose contents are the error number and error description. This is the default. If no entry is made for the intErrorMode parameter, the default is used.

If you enter a 1 for intErrorMode the error is raised for you to handle:

```
OpenPicture "BadPic", , , , 1
```

Your error handling code would have to look something like this:

```
On Error Goto Errorhandler
OpenPicture "BadPic", , , , 1
End Sub
Errorhandler:
Msgbox "my error message" + Chr(13) + Cstr(Err.Number) + Chr(13) + Err.Description, , Err.Source
```

If you enter a 2 for intErrorMode, the error is sent to all typers, including the Alarm History window using the SendOperatorMessage method:

```
OpenPicture "BadPic", , , , 2
```

When you use 2 for the intErrorMode, you provide for silent error tracking.

The following example allows the calling **Picture***Test* to open a Quick Trend Picture and display the tag list *Strings1*.

```
OpenPicture , , , , , Test, QuickTrend, Strings1
```

The following example allows the calling picture *Test* to open a Tag Status Picture and display the tag list *Strings2*.

```
OpenPicture , , , , , Test, TagStatus, Strings2
```

The following example allows the calling picture *Test* to open a Tag Control Panel Picture and display the tag list *Strings3*.

```
OpenPicture , , , , , Test, TagControlPanel, Strings3
```

OpenTGDPicture Subroutine Example

This example opens the picture *TestPicture*, gives it an alias of *Test*, and then open the taggroup file *TestTGD*. The coordinates of the top-left side of the picture are 75, 75.

```
OpenTGDPicture "TestPicture", "Test",
 75, 75, "TestTGD"
```

This next example opens the picture *TestPicture*, allowing for multiple instances. The coordinates of the top-left side of the picture are 75, 75.

```
OpenTGDPicture "TestPicture", , 75, 75, , , true
```

This next example opens the picture *TestPicture*, allowing for multiple instances. It opens the instance with the taggroup file *TestTGD*. The coordinates of the top-left side of the picture are 75, 75.

```
OpenTGDPicture "TestPicture", , 75, 75, "TestTGD", , true
```

P-Q

ParseConnectionSource Method Example

The following example parses the *AI1.F_CV* source to the **VerticalFillPercentage** property of **Oval***Oval1* to determine the validity of the data source.

```
Dim iStatus As Long
Dim validObjs As Variant
Dim UndObjs As Variant
Dim FQSource As String
Oval1.ParseConnectionSource "VerticalFillPercentage", "AI1.F_CV", iStatus, validObjs, UndObjs, FQSource
```

Paste Method Example

The following example pastes the **Oval** object *Oval1* to the **Picture** *TestPicture* after copying it to the clipboard.

```
Oval1.Select
TestPicture.Copy
TestPicture.Paste
```

PasteSpecial Method Example

The following example opens the Paste Special dialog box for the **Picture** *TestPicture.*

```
TestPicture.PasteSpecial
```

Pause Method Example

The following example pauses the **Chart** *Chart1*.

```
Chart1.Pause
```

PauseAlarmRead Method Example

The following example pauses the **Alarm Summary** object prior to copying alarms to list.

```
' pause alarm read so nothing is added, deleted or moved

AlarmSummaryOCX1.PauseAlarmRead
lngTotalFiltered = AlarmSummaryOCX1.TotalFilteredAlarms
ListBox1.Clear
For lngLoop = 1 To lngTotalFiltered
AlarmSummaryOCX1.SelectAlarmRow lngLoop, True
AlarmSummaryOCX1.GetSelectedNodeTag strNode, strTag
ListBox1.AddItem strNode & "." & strTag

AlarmSummaryOCX1.SelectAlarmRow lngLoop, False
Next lngLoop
' resume alarm read
AlarmSummaryOCX1.ResumeAlarmRead
```

PercentageToLogical Method Example

The following example converts the Top, Left, Height and Width coordinates from 30, 30, 100, 150 in percentage of screen space available to postscript points or logical units for the **Picture** *TestPicture*.

```
Dim dTop As Double
Dim dLeft As Double
Dim dHeight As Double
Dim dWidth As Double
dTop = 30
dLeft = 30
dHeight = 100
dWidth = 150
TestPicture.PercentageToLogical dTop, dLeft, dHeight, dWidth
```

PercentageToPixel Method Example

The following example converts the Top, Left, Height and Width coordinates from 30, 30, 100, 150 in percentage of screen space available to pixels for the **Picture** *TestPicture*.

```
Dim dTop As Double
Dim dLeft As Double
Dim dHeight As Double
Dim dWidth As Double
dTop = 30
dLeft = 30
dHeight = 100
dWidth = 150
TestPicture.PercentageToPixel dTop, dLeft, dHeight, dWidth
```

PictureAlias Subroutine Example

The following example assigns the alias *TestPicAlias* to the current **Picture**.

```
PictureAlias "TestPicAlias"
ClosePicture "TestPicAlias"
```

PixelToPercentage Method Example

The following example converts the Top, Left, Height and Width coordinates from 30, 30, 100, 150 in pixels to percentage of screen space available for the **Picture** *TestPicture*.

```
Dim dTop As Double
Dim dLeft As Double
Dim dHeight As Double
Dim dWidth As Double
dTop = 30
dLeft = 30
dHeight = 100
dWidth = 150
TestPicture.PixelToPercentage dTop, dLeft, dHeight, dWidth
```

PrintChart Method Example

The following example prints the Line/Multiline chart to the size of a full page of paper.

```
Private Sub CommandButton4_Click()
LineChart1.PrintChart SizeUnits_FullPage
End Sub
```

PrintOut Method Example

The following example opens the print dialog for the active **Document**.

```
Dim bPrinted as boolean
bPrinted = Application.ActiveDocument.PrintOut
```

PrintReport Subroutine Example

The following example prints 3 copies of pages 1 through 5 of the report *TestReport*, collating each copy.

```
PrintReport "TestReport", False, 3, True, 1, 5
```

PromptToChangePassword Method Example

The following example checks the user's Windows account expiration status. If the account expired, this example prompts the user to change his password (if he has rights to do so). If the account has not expired, a message appears stating how many days are left until the password expires.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim UserName As String
Dim PassWord As String
Dim UsrId As String
Dim bExpired As Boolean
Dim daysLeft As Long
Dim bCanChangePassword As Boolean
'user name of an iFix user using Windows security
UserName = "expire"
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignature")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Check account status and user's right to change password
ESig.CheckAccountExpiration UserName, bExpired, bCanChangePassword, daysLeft
If bExpired <> False Then
If bCanChangePassword <> False Then
ESig.PromptToChangePassword UserName
End If
Else
'password is not expired
MsgBox "Password is due to expire in " & daysLeft & " days."
End If
Else
'the hardware key is not enabled for signature, security is disabled, or bypass signature is in effect
MsgBox “Signature is not enabled on this node.”
End If
```

QuickAdd Subroutine Example

The following example gets the status of the block *NewBlock* to the database. If *NewBlock* doesn't exist, the QuickAdd dialog will be opened, allowing the user to configure the block on the fly.

```
Dim iStatus As Integer
iStatus = QuickAdd("NewBlock1")
```

Quit Method Example

The following example shuts down the WorkSpace, prompting the user to save changes made to any open documents.

```
Application.Quit 3
```

R

RampValue Subroutine Example

The following example ramps the block *AO1* by 25 percent of its EGU range.

```
RampValue "25", True, "AO1"
```

The following example ramps the block *AO1* by a value of 15.

```
RampValue "15", False, "AO1"
```

Read Method Example

The following example:

- Creates a data system ocx *FDS*;
- Adds a data **Group***DataGroup1* to the **Groups** collection;
- Adds a **DataItem** to the **DataItems** collection;
- Reads the **Group** *DataGroup1*;
- Reads the **DataItem**

```
'Create the Data System OCX
Dim FDS As Object
Set FDS = CreateObject("FixDataSystems.Intellution FD
 Data System Control")
'Add a group to the Groups collection
FDS.Groups.Add ("DataGroup1")
FDS.Groups.Item("DataGroup1").DataItems.Add("Fix32.THISNODE.AI1.F_CV")

'Read DataGroup1
FDS.Groups.Item("DataGroup1").Read
'Read the DataItem
FDS.Groups.Item("DataGroup1").DataItems.Item(1).Read
```

ReadValue Subroutine Example

The following example reads the value of the block *AI1*.

```
Dim lValue As Variant
lValue = ReadValue("AI1")
```

Refresh Method Example

The following example repaints the **Rectangle** object *Rect1.*

```
Rect1.Refresh
```

RefreshChartData Method Example

The following simple example refreshes the data being displayed in the *Chart1*, which is a HistogramChart, LineChart, or SPCBarChart object.

```
Chart1.RefreshChartData
```

The following example shows a refresh of an Enhanced Chart, LineChart1, after scroll forward of 50% is performed on the Enhanced Chart.

```
    Dim dtTime As Variant
    Dim dtDate As Variant
    Dim dInterval As Long

    ' set scroll percentage
    dInterval = LineChart1.Duration
    dInterval = dInterval * 0.5 ' 50%

    'scroll time
    dtTime = GeneralDataset1.FixedTime
    dtTime = DateAdd("s", dInterval, dtTime)
    HistoricalDataset1.FixedTime = dtTime

    ' scroll date
    dtDate = GeneralDataset1.FixedDate
    dtDate = DateAdd("s", dInterval, dtDate)
    HistoricalDataset1.FixedDate = dtDate

    'refresh chart
    LineChart1.RefreshChartData
```

The second example above requires a reference to the historical data set named “iFIX GeneralDataSet Object v 1.0 Type Library” in your VBA project.

RegCloseKey Subroutine Example

The following example is the actual PrintReport subroutine from FactoryGlobals global subroutines. Here, we implement late binding to run Crystal Reports. First, find out if Crystal Reports is installed on the user's machine with **RegOpenKeyEx**. If it is, we create an instance of it. Then, we close the registry key with **RegCloseKey**.

NOTE: This example does not apply to Crystal XI; it applies to an earlier version, such as Crystal 7. For Crystal XI, use the PrintReport subroutine instead. The PrintReport subroutine will do all of the registry entries for you.

```
Public Sub PrintReport(ByVal Report As String, Optional Prompt As Boolean, Optional ByVal Copies As Long, Optiona
Dim CrystalApplication As Object
Dim CrystalReport As Object
Dim lngResult As Long
Dim lngRes As Long
On Error GoTo ErrorHandler
'Check if Crystal Reports is installed.
lngResult = RegOpenKeyEx(&H80000000, "CrystalReports", &O0, &H20000, lngRes)
'If it is, create an instance of it.
If lngResult = 0 Then
Set CrystalApplication = CreateObject("Crystal.CRPE.Application")
'If not, send the user a message.
Else
MsgBox "You do not have Crystal Reports installed."
End
End If
'Close the registry key.
lngResult = RegCloseKey(&H80000000)
Set CrystalReport = CrystalApplication.OpenReport(Report)
CrystalReport.PrintOut Prompt, Copies, Coll, StartNo, EndNo
Exit Sub
ErrorHandler:
HandleError
End Sub
```

RegOpenKeyEx Subroutine Example

The following example is the actual PrintReport subroutine from FactoryGlobals global subroutines. Here, we implement late binding to run Crystal Reports. First, find out if Crystal Reports is installed on the user's machine with **RegOpenKeyEx**. If it is, we create an instance of it. Then, we close the registry key with **RegCloseKey**.

NOTE: This example does not apply to Crystal XI; it applies to an earlier version, such as Crystal 7. For Crystal XI, use the PrintReport subroutine instead. The PrintReport subroutine will do all of the registry entries for you.

```
Public Sub PrintReport(ByVal Report As String, Optional Prompt As Boolean, Optional ByVal Copies As Long, Optiona
Dim CrystalApplication As Object
Dim CrystalReport As Object
Dim lngResult As Long
Dim lngRes As Long
On Error GoTo ErrorHandler
'Check if Crystal Reports is installed.
lngResult = RegOpenKeyEx(&H80000000, "CrystalReports", &O0, &H20000, lngRes)
'If it is, create an instance of it.
If lngResult = 0 Then
Set CrystalApplication = CreateObject("Crystal.CRPE.Application")
'If not, send the user a message.
Else
MsgBox "You do not have Crystal Reports installed."
```

```
End
End If
'Close the registry key.
lngResult = RegCloseKey(&H80000000)
Set CrystalReport = CrystalApplication.OpenReport(Report)
CrystalReport.PrintOut Prompt, Copies, Coll, StartNo, EndNo
Exit Sub
ErrorHandler:
HandleError
End Sub
```

Remove Method Example

The following example gets the index number for the **Click** of the object *CurrentObject* and, if a procedure exists, removes it.

```
Dim lIndex As Long
Dim lFound As Long
CurrentObject.Procedures.GetEventHandlerIndex "Click", lIndex, lFound
If lFound = 1 Then
CurrentObject.Remove lIndex
End If
The following example removes the GroupDataGroup1 from the Groups collection of the FixDataSystemFDS.
FDS.Groups.Remove("DataGroup1")
The following example removes the first Group from the Groups collection of the FixDataSystemFDS.
FDS.Groups.Remove(1)
```

RemoveAll Method Example

The following example removes all the lines of code from the Click event of the object Rect2 in the active document.

```
Dim o As Object
Dim oProc As Object
Dim lIndex As Long
Dim lFound As Long
Set o = Application.ActiveDocument.Page.FindObject("Rect2")

o.Procedures.GetEventHandlerIndex "Click", lIndex,
 lFound
If (lFound) Then
Set oProc = o.Procedures.Item(lIndex)
oProc.Lines.RemoveAll
End If
```

RemoveAllLevels Method Example

The following example removes all levels in the **Lookup** object *iLookup.*

```
iLookup.RemoveAllLevels
```

RemoveItem Method Example

The following example removes the first column from the **Chart***Chart1.*

```
Chart1.RemoveItem 1
```

RemoveLegendItem Method Example

The following example removes the value column from the **Chart***Chart1.*

```
Chart1.RemoveLegendItem "Value"
```

RemoveLevel Method Example

The following example removes level 4 from the **Lookup** object *Lookup1*.

```
Lookup1.RemoveLevel 4
```

RemoveObject Method Example

The following example removes the **Timer** object *MyTimer* from the **Schedule** *TestSchedule* and then refreshes the **Schedule** so that the **Timer** object is no longer displayed.

```
TestSchedule.RemoveObject "MyTimer"
TestSchedule.DoMenuCommand scHREFreshView
```

ReplacePicture Subroutine Example

The following example replaces the **Picture** *TestPicture* with *TestPicture1*.

```
ReplacePicture "TestPicture1", "TestPicture"
```

The following example allows the calling picture *TestPicture* to replace the active Quick Trend Picture with *QuickTrend1* and display the tag list *Strings1*.

```
ReplacePicture , , , , , TestPicture, QuickTrend1, Strings1
```

The following example allows the calling picture *TestPicture* to replace the active Tag Status Picture with *TagStatus1* and display the tag list *Strings2*.

```
ReplacePicture , , , , , TestPicture, TagStatus1, Strings2
```

The following example allows the calling picture *TestPicture* to replace the active Tag Control Panel Picture with *TagControlPanel1* and display the tag list *Strings3*.

```
ReplacePicture , , , , , TestPicture, TagControlPanel1, Strings3
```

RemovePictureFromStartupList Example

The following example removes the Picture named pic1.grf (in the D:\Program Files (x86)\Proficy\iFIX\PIC directory) from the Configuration environment's startup list of the iFIX WorkSpace.

```
Dim lErr as Long
lErr = Application.UserPreferences.RemovePictureFromStartupList_

("D:\Program Files (x86)\Proficy\iFIX\pic\pic1.grf", False)
```

Note that the full path is required for the picture. Also note that the second paramater, when set to FALSE, removes the picture from the Configuration environment's startup list. When set to TRUE, it removes the picture from the Runtime environment's startup list.

Replace_QT_Pic Method Example

The following example replaces the current picture with a Quick Trend picture when a button is pressed. The tags that are displayed are retrieved from the currently selected objects.

```
Private Sub Button_ReplaceQTPic_Click()

    Me.Replace_QT_Pic

End Sub
```

Replace_TCP_Pic Method Example

The following example replaces the current picture with a Tag Control Panel picture when a button is pressed. The tags that are displayed are retrieved from the currently selected objects.

```
Private Sub Button_ReplaceTCPPic_Click()

    Me.Replace_TCP_Pic

End Sub
```

Replace_TS_Pic_Type Method Example

The following example replaces the current picture *Pic1* with the Quick Trend picture *QTPic1* and the tag list *Strings1*.

```
Pic1.Replace_TS_Pic_Type QTPic1, Strings1
```

The following example replaces the current picture *Pic1* with the Tag Status picture *TSPic1* and the tag list *Strings2*.

```
Pic1.Replace_TS_Pic_Type TSPic1, Strings2
```

The following example replaces the current picture *Pic1* with the Tag Control Panel picture *TCPPic1* and the tag list *Strings3.*

```
Pic1.Replace_TS_Pic_Type TCPPic1, Strings3
```

Replace_TS_Pic Method Example

The following example replaces the current picture with a Tag Status picture when a button is pressed. The tags that are displayed are retrieved from the currently selected objects.

```
Private Sub Button_ReplaceTSPic_Click()

    Me.Replace_TS_Pic

End Sub
```

ReplaceDocument Method Example

The following example replaces the active **Document** with *TestPicture.*

```
Application.ActiveWindow.ReplaceDocument "C:\Program Files (x86)\Proficy\iFIX\Pic\TestPicture.grf"
```

ReplaceInString Method Example

The following example finds each occurrence of the string *AI1* with the string *FIX32.NODE1.AI1.F_CV* by calling **FindInString** and then uses the lFirst, lCount parameter return values to replace *AI1* with *AI2*, returning the new string in the *sReplacement* parameter. The operation is specified to include scripts in the search.

```
Dim lFirst As Long
Dim lCount As Long
Dim sMatchString As String
Dim bFound As Boolean
Dim bsuccess As Boolean
Dim sTarget As String
Dim sReplacement As String
sTarget = "FIX32.NODE1.AI1.F_CV"
FindReplace.FindInString "FIX32.NODE1.AI1.F_CV", 1, "AI1", 8, sMatchString, lFirst, lCount, bFound
FindReplace.ReplaceInString sTarget, "AI2", sMatchString, lFirst, lCount, 8, sReplacement, bsuccess
```

ReplaceTGDPicture Subroutine Example

This example replaces the open picture *TestPicture* with *TestPicture1* and then opens the tag group file *TestTGD*.

```
ReplaceTGDPicture "TestPicture1", "TestTGD",
 "TestPicture"
```

ResetChartData Method Example

The following example resets the data being displayed for the **Chart***Chart1.*

```
Chart1.ResetChartData
```

ResetObjectStats Method Example

The following example resets the statistics for all **Event** objects in the **Schedule***TestSchedule.*

```
TestSchedule.ResetObjectStats 2
```

The following example resets the statistics for the **Timer** *MyTimer* in the **Schedule** *TestSchedule.*

```
TestSchedule.ResetObjectStats "MyTimer"
```

The following example resets the statistics for the **Timer** objects *TestTimer, TestTimer1, TestTimer2* in the **Schedule** *TestSchedule.*

```
Dim objects(2) As Variant
objects(0) = "TestTimer"
objects(1) = "TestTimer1"
objects(2) = "TestTimer2"
TestSchedule.ResetObjectStats objects
```

ResetStats Method Example

The following example resets the statistics for the **Timer** object *iTimer* to zero.

```
iTimer.ResetStats
```

ResetZoom Method Example

The following example resets the **Chart** *Chart1* to display its default viewing area.

```
Chart1.ResetZoom
```

ResolveTagGroupFile Example

The following example resolves the tag group file *mytaggroup.tgd* in the iFIX picture *pic1*.

```
pic1.ResolveTagGroupFile "mytaggroup.tgd"
```

The ResolveTagGroupFile method is a Configuration environment method that loops through the tag groups referenced in the picture, perform the substitutions, and persist the information. Using this method provides you with a faster load time.

Resume Method Example

The following example resumes the **Chart** *Chart1.*

```
Chart1.Resume
```

ResumeAlarmRead Method Example

The following example resumes updating the **Alarm Summary** object after to copying alarms to list.

```
' pause alarm read so nothing is added, deleted or moved

AlarmSummaryOCX1.PauseAlarmRead
lngTotalFiltered = AlarmSummaryOCX1.TotalFilteredAlarms
ListBox1.Clear
For lngLoop = 1 To lngTotalFiltered
AlarmSummaryOCX1.SelectAlarmRow lngLoop, True
AlarmSummaryOCX1.GetSelectedNodeTag strNode, strTag
ListBox1.AddItem strNode & "." & strTag

AlarmSummaryOCX1.SelectAlarmRow lngLoop, False
Next lngLoop
' resume alarm read
AlarmSummaryOCX1.ResumeAlarmRead
```

RetrieveDefinition Method Example

The following example retrieves the definitions contained in a tag group file.

```
Dim sTokenList() as String, TokenList as Variant
Dim sReplacementList() as String, ReplacementList as Variant

Dim sDescriptionList() as String, DescriptionList as Variant

' Create the tag group file object
Dim TGD As Object
Set TGD = CreateObject("TagGroupDefinitionInterfaceDll.TagGroupDefinitionInterface")

TGD.RetrieveDefinition "Test", 2, TokenList, ReplacementList,
 DescriptionList
```

RetrieveTagGroupVariables Method Example

Used to retrieve all of the tag group symbols referenced in the picture. This does include scripts but does not include Forms. An array of strings is returned.

```
Dim iCount As Integer
Dim vaSymbols As Variant
pic1.RetrieveTagGroupVariables iCount, vaSymbols
```

Rotate Method Example

The following example rotates the **Polygon** object *Polygon1* by 45 degrees.

```
Polygon1.Rotate 45, False
```

RunObject Method Example

The following example runs all **Timer** objects in the **Schedule***TestSchedule.*

```
TestSchedule.RunObject 1
```

The following example runs the **Timer** *MyTimer* in the **Schedule** *TestSchedule.*

```
TestSchedule. RunObject "MyTimer"
```

The following example runs the **Timer** objects *TestTimer, TestTimer1, TestTimer2* in the **Schedule** *TestSchedule.*

```
Dim objects(2) As Variant
objects(0) = "TestTimer"
objects(1) = "TestTimer1"
objects(2) = "TestTimer2"
TestSchedule. RunObject objects
```

S

Save Method Example

The following example saves all open documents in the WorkSpace, prompting the user for each **Document**.

```
Dim iStatus As Long
iStatus = Application.Documents.Save(True)
The following example opens the DocumentTestPicture and saves it as TestPicture1 without prompting the user.
Dim iDoc As Object
Set iDoc = Application.Documents.open("C:\Program Files (x86)\Proficy\iFIX\pic\testpicture.grf")
iDoc.Save "TestPicture1.grf", False
```

Save_TS_List Method Example

The following example saves the tag status list displayed in the current picture when a button is pressed.

```
Private Sub Button_SaveTSList_Click()

    Me.Save_TS_List

End Sub
```

SaveAsSVG Method Example

The following example opens the Document TestPicture and saves it as TestPicture.svg.

```
Private Sub CommandButton1_Click()
Dim iDoc As CFixFileLink
Set iDoc = Application.Documents.open("C:\Program Files (x86)\Proficy\iFIX\pic\TestPicture.grf")
Application.Documents.SaveAsSVG "C:\Program Files (x86)\Proficy\iFIX\pic\TestPicture.svg", iDoc
End Sub
```

SaveToHistoryList Method Example

The following example saves the item "AI1.F_CV" to the history list for the **ExpressionEditor** object *ExpressionEditor1*.

```
Dim iIndex As Integer
iIndex = ExpressionEditor1.SaveToHistoryList("AI1.F_CV")
```

ScrollBack Method Example

The following example scrolls back in the **Chart** *Chart1* by the current value of its **ScrollPercentage** property.

```
Chart1.ScrollBack
```

ScrollForward Method Example

The following example scrolls forward in the **Chart** *Chart1* by the current value of its **ScrollPercentage** property.

```
Chart1.ScrollForward
```

ScrollTimeBack Method Example

The following example scrolls the **Pen** *Pen1* back by the current value of its **ScrollPercentage** property.

```
Pen1.ScrollTimeBack
```

ScrollTimeForward Method Example

The following example scrolls the **Pen** *Pen1* forward by the current value of its **ScrollPercentage** property.

```
Pen1.ScrollTimeForward
```

ScrollToPosition Method Example

ScrollToPosition is a Run mode method that can be used for pictures with Enhanced Coordinates enabled. The following example uses the ScrollToPosition method after zooming in on an object.

```
Option Explicit
Private Declare Function FindWindow Lib "user32" Alias "FindWindowA" (ByVal lpClassName As String, ByVal lpWindow
Private Declare Function FindWindowEx Lib "user32" Alias "FindWindowExA" (ByVal hWnd1 As Long, ByVal hWnd2 As Lon
Private Declare Function GetClientRect Lib "user32" (ByVal hWnd As Long, lpRect As RECT) As Long
Private Type RECT
    Left As Long
    Top As Long
    Right As Long
    Bottom As Long
End Type
Private Sub CFixPicture_MouseUp(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Double, ByVal Y As Do
If Button > 1 Then
unzoom
Else
'this will center the zoom on a mouse click
ZoomtoObject X, Y
End If
End Sub
Private Sub ZoomtoObject(xpos As Double, ypos As Double)
 Dim lZoom As Double
    Dim scrollx, windowHeightPixel, pixelx As Long
    Dim scrolly, windowWidthPixel, pixely As Long
    Dim Top, windowWidthLogical  As Double
    Dim Left, windowHeightLogical As Double
    Dim lhwndWS As Long
    Dim lhwndParent As Long
    Dim lhwndPic As Long
    Dim lRet As Long
    Dim sBuf, sName As String
    Dim buffer As String
    sBuf = vbNullString
    sName = Me.PictureName
    windowWidthPixel = 0
    windowHeightPixel = 0
    'Find the Workspace window
    lhwndWS = FindWindow("WorkSpaceClass", sBuf)
    If lhwndWS > 0 Then
        lhwndParent = FindWindowEx(lhwndWS, 0, "MDIClient", vbNullString)
        If lhwndParent <> 0 Then
            lhwndPic = FindWindowEx(lhwndParent, 0, vbNullString, sName)
```

```
                'get client rect of picture
                If lhwndPic <> 0 Then
                   Dim aRect2 As RECT
                   lRet = GetClientRect(lhwndPic, aRect2)
                   If lRet <> 0 Then
                       Top = 0
                       Left = 0
                       windowHeightPixel = aRect2.Bottom - aRect2.Top
                       windowWidthPixel = aRect2.Right - aRect2.Left
                   End If
               End If
           End If
       End If

       Me.PixelToLogical windowWidthPixel, windowHeightPixel, windowWidthLogical, windowHeightLogical
       lZoom = Me.Zoom
       lZoom = lZoom + lZoom / 2

       scrollx = (windowWidthLogical / lZoom) / 2
       scrolly = (windowHeightLogical / lZoom) / 2

       If lZoom > 1 Then
           scrollx = xpos - scrollx
           scrolly = ypos - scrolly
       Else
           scrollx = xpos
           scrolly = ypos
       End If

       Me.Zoom = lZoom
       Me.LogicalToPixel scrollx, scrolly, pixelx, pixely
       Me.ScrollToPosition pixelx * lZoom, pixely * lZoom
   End Sub
   Private Sub unzoom()
   Me.Zoom = 1
   End Sub
```

NOTE: For more information on Enhanced Coordinates, refer to the Picture Coordinate Systems topic in the Creating Pictures e-book.

Select Method Example

The following example selects the the **Oval** *Oval1*.

```
Oval1.Select
```

SelectAlarmRow Method Example

The following example shows how to select and deselect a row of the **Alarm Summary** object.

```
' pause alarm read so nothing is added, deleted or moved

AlarmSummaryOCX1.PauseAlarmRead
lngTotalFiltered = AlarmSummaryOCX1.TotalFilteredAlarms
ListBox1.Clear
For lngLoop = 1 To lngTotalFiltered
AlarmSummaryOCX1.SelectAlarmRow lngLoop, True
```

```
AlarmSummaryOCX1.GetSelectedNodeTag strNode, strTag
ListBox1.AddItem strNode & "." & strTag

AlarmSummaryOCX1.SelectAlarmRow lngLoop, False
Next lngLoop
' resume alarm read
AlarmSummaryOCX1.ResumeAlarmRead
```

SelectAll Method Example

The following example selects all of the objects in the **Picture***TestPicture*.

```
TestPicture.SelectAll
```

SelectObject Method Example

The following example selects both the **Polyline** object, *Polyline1,* and the **Oval** object, *Oval1*.

```
PolyLine1.SelectObject False
Oval1.SelectObject False
The following example selects only Oval1.
PolyLine1.SelectObject True
Oval1.SelectObject True
```

SendOperatorMessage Method Example

The following example sends a message to node *NODE1*.

```
System.SendOperatorMessage "Something has occurred",
 "NODE1"
```

SendSignedOperatorMessage Method Example

The following example creates the ESignature object, checks to see if the node is enabled for electronic signature, displays the Electronic Signature dialog box, validates the signature and sends a message to the audit trail.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bVerify As Boolean
Dim bContinuousUse As Boolean
Dim bValidSig As Boolean
Dim PerformUserName As String
Dim PerformUserID As String
Dim PerformComment As String
Dim PerformFullName As String
Dim VerifyUserName As String
Dim VerifyUserID As String
Dim VerifyComment As String
Dim VerifyFullName As String
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignatureFactory")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
bVerify = True ' verification is not required
bContinuousUse = True ' allow continuous user, if any, to be displayed in the dialog box
bValidSig = False ' will be set to TRUE by GetSignature if signature is captured successfully
'Display the Electronic Signature dialog box
ESig.GetSignature "Action Description", bVerify, bContinuousUse, bValidSig, Ucase(PerformUserName), Ucase(Perform
If bValidSig = True Then
'Send a message to the audit trail
ESig.SendSignedOperatorMessage "Action Description", "", "", Ucase (PerformUserID), PerformComment, Ucase(Verify
'Get the full names of the signers
ESig.GetFullname Ucase(PerformUserID), Ucase(PerformFullName)
ESig.GetFullname Ucase(VerifyUserID), Ucase(VerifyFullName)
'Show Results
MsgBox "Action performed by " + PerformUserName + " (" + PerformFullName + ") " + PerformComment + " and verified
Else
MsgBox "Signature was not captured."
End If
Else
MsgBox "Signature is not enabled on this node."
End If
```

SendToBack Method Example

The following example selects the **Oval** object *Oval1* contained within the **Picture** *TestPicture* and sends it to the back of the stacking order.

```
Oval1.Select
TestPicture.SendToBack
```

SetAlarmBackgroundColor Method Example

The following example sets the color to display for each alarm priority in the **AlarmSummary** object *AlarmSummaryOCX1*.

```
Dim lGreen As OLE_COLOR
Dim lBlue As OLE_COLOR
Dim lRed As OLE_COLOR
lGreen = 57344
lBlue = 16722988
lRed = 725759
AlarmSummaryOCX1.SetPriorityColor 7, lGreen
AlarmSummaryOCX1.SetPriorityColor 6, lBlue
AlarmSummaryOCX1.SetPriorityColor 5, lRed
```

SetAlarmForegroundColor Method Example

The following example sets the color to display for alarms with a LOLO status in the **AlarmSummary** object *AlarmSummaryOCX1*.

```
Dim lGreen As OLE_COLOR
lGreen = 57344
AlarmSummaryOCX1.SetAlarmForegroundColor 1, lGreen
```

SetAuto Subroutine Example

The following example sets the block *AI1* to automatic mode.

```
SetAuto "AI1"
```

SetContinuousUser Method Example

The following example ensures that the User Name entered is valid, creates the ESignature object, checks that the node is enabled for electronic signature, and sets and gets the name of the continuous user.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bSigRequired As Boolean
Dim UserName As String
Dim SetName As String
'valid user name
SetName = "TestUser"
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignature")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Set the name of the continuous user to "TestUser"
ESig.SetContinuousUser SetName
'Get the name of the continuous user
ESig.GetContinuousUser UserName
MsgBox “Continuous user: ” & UserName
Else
'the hardware key is not enabled for signature, security is disabled, or bypass signature is in effect
MsgBox “Signature is not enabled on this node.”
End If
```

SetCurrentValue Method Example

The following example sets the current value for the **Pen** *Pen1*.

```
Pen1.SetCurrentValue 25, #4/13/98 1:15:00 PM#, 192
```

SetDispatch Method Example

Reserved for internal purposes.

SetDispid Method Example

Reserved for internal purposes.

SetDuration Method Example

The following example sets the **Duration** of the **Chart***Chart1* to be 1 hour and 35 minutes.

```
Chart1.SetDuration 0, 1, 35, 0
```

SetFocusToComboBox Method Example

The following example sets the focus to the combobox portion of the **ExpressionEditor** object, *ExpressionEditor1*.

```
ExpressionEditor1.SetFocusToComboBox
```

SetGlobalMovingEndTimeToCurrent Method Example

The following example sets the System.GlobalEndTime property to the current system time.

```
Private Sub Group4_Click()
System.SetGlobalMovingEndTimeToCurrent
End Sub
```

SetIndirectionInfo Method Example

Reserved for internal purposes.

SetInterval Method Example

The following example sets the interval of the **Chart** *Chart1* to be 5 minutes and 30 seconds.

```
Chart1.SetInterval 0, 0, 5, 30
```

SetManual Subroutine Example

The following example sets the block *AI1* to manual mode.

```
SetManual "AI1"
```

SetLegendMask Method Example

The following example causes all of the potential legend items for the Enhanced Chart to display.

```
Private Sub CommandButton12_Click()
Dim objDataSet As Object ' FixRealTimeDataSet.FixRealTimeDataSet
Set objDataSet = LineChart1.GetCurrentDataSet()
With objDataSet
.SetLegendMask IIf(0 = .DSLegendMask, LegendMask_All, 0)
End With
End Sub
```

SetNumericFormat Method Example

The following example formats the display of the **Format** object *Format1*, displaying 3 whole digits, 3 decimal digits and setting the justification to Right.

```
Format1.SetNumericFormat 3, 3, 2
```

SetPenDataArray Method Example

The following is an example on how to pass in arrays of user defined data to create a static **Pen** in a **Chart**.

```
Dim iWrkSpace As Workspace
Dim db_var_name As Database
Dim record_var As Recordset
Dim iCount As Integer
Dim dVal As Variant
Dim dtDate As Variant
Dim lQual As Variant
Dim iResult As Integer
Set iWrkSpace = CreateWorkspace("", "admin",
 "", dbUseJet)
Set db_var_name = iWrkSpace.OpenDatabase("Chart.mdb")

Set record_var = db_var_name.OpenRecordSet("Data Query",
 dbOpenDynaset)
record_var.MoveLast
iCount = record_var.RecordCount
record_var.MoveFirst
Dim iRow As Integer
Dim iCol As Integer
Dim Value(500) As Double
Dim Times(500) As Date
Dim Quality(500) As Long
Dim i As Integer
For i = 0 To iCount - 1
```

```
Value(i) = record_var.Fields("Value").Value
Times(i) = record_var.Fields("Time").Value
Quality(i) = record_var.Fields("Quality").Value

'Quality(i) should be set to
 192 (good data quality) in order to show data
record_var.MoveNext
Next i
db_var_name.Close
dVal = Value
dtDate = Times
lQual = Quality
Call Pen1.SetPenDataArray(iCount, dVal, dtDate, lQual)
```

SetPointAt Method Example

The following example modifies the location of the point at index 2 to the point 75,10 for the Polygon object *Polygon1*.

```
Dim iPoint As FixFloatPoint
Set iPoint = New FixFloatPoint
iPoint.x = 75
iPoint.y = 10
Polygon1.SetPointAt 2, iPoint
```

SetPriorityColor Method Example

The following example sets the color to display for each alarm priority in the AlarmSummary object *AlarmSummaryOCX1*.

```
Dim lGreen As OLE_COLOR
Dim lBlue As OLE_COLOR
Dim lRed As OLE_COLOR
lGreen = 57344
lBlue = 16722988
lRed = 725759
AlarmSummaryOCX1.SetPriorityColor 7, lGreen
AlarmSummaryOCX1.SetPriorityColor 6, lBlue
AlarmSummaryOCX1.SetPriorityColor 5, lRed
```

SetProperty Method Example

The following example sets the FillStyle property of the Oval object *Oval1* to *3*.

```
Oval1.SetProperty "FillStyle", 3
```

SetScriptWindow Method Example

The following example instantiates the Visual Basic editor for the Oval object *Oval1* contained within the Picture*TestPicture*, displaying the prototype for its MouseDown event.

```
Oval1.Select
TestPicture.SetScriptWindow True, "MouseDown"
```

SetSource Method Example

The following example sets the data source for the Linear object *Linear1* to AI1, specifying to use the data source if it doesn't exist in the database, setting the UpdateRate to 1 and the Deadband to 5.

```
Linear1.SetSource "ai1.f_cv", True, 1, 5
```

SetStatusColor Method Example

The following example sets the color to display for alarms with a LOLO status in the AlarmSummary object *AlarmSummaryOCX1*.

```
Dim lGreen As OLE_COLOR
lGreen = 57344
AlarmSummaryOCX1.SetStatusColor 1, lGreen
```

SetStatusFont Method Example

The following example sets the font to display for alarms with a HIHI status in the AlarmSummary object *AlarmSummaryOCX1*. It sets the font to be displayed as Bold Garamond with the Strikethrough option set to True.

```
AlarmSummaryOCX1.SetStatusFont 2, "Garamond", True,
 False, True, False
```

SetStringFormat Method Example

The following example sets the raw formatting of the Format object for the Datalink*Datalink1* contained in Picture*TestPicture*.

```
Dim iFormat As Object
Set iFormat = TestPicture.FindObject("Datalink1.Format1")

iFormat.SetStringFormat "Value = %s"
```

SetSymbolValues Subroutine Example

In this example, we pass in two symbols with their corresponding substitution strings.

```
SetSymbolValues
Private Sub Oval1_Click()
Dim t(1, 1) As String
t(0, 0) = "AC" 'Symbol to substitute
t(0, 1) = "AIRCONDITIONER2" 'Actual substitution string
t(1, 0) = "TEST" 'Symbol to substitute
t(1, 1) = "Fix32.FXPR.AIRCONDITIONER2>MOTOR>TEMP.F_CV" 'Actual substitution string
Me.SetSymbolValues (t)
End Sub
```

SetTabSelection Method Example

The following example displays the Fix32Database, Pictures and Globals for the ExpressionEditor *ExpressionEditor1*.

```
Dim bResult as boolean
bResult = ExpressionEditor1.SetTabSelection(11)
```

SetTimeBeforeNow Method Example

The following example sets the time for the Chart *Chart1* to be 2 hours and 30 minutes before the current time.

```
Chart1.SetTimeBeforeNow 2, 30, 0
```

SetTimeCursorTime Method Example

The following example sets the time cursor time for the first Pen in the Pens collection of the Chart*Chart1* to be 11/15/98 at 1:30PM.

```
Chart1.SetTimeCursorTime #11/15/98 1:30:00 PM#, 1
```

SetWindowLocation Method Example

The following example sets the location of the Picture *TestPicture* by setting the top percentage to 10, the left percentage to 10 and setting the height and width percentages both to 200, without redrawing the document after setting the window's location.

```
TestPicture.SetWindowLocation 10, 10, 200, 200, False
```

ShellExecute Subroutine Example

The following example opens Notepad.

```
Dim lResult As Long
lResult = ShellExecute(hWnd, "Open", "Notepad.exe",
 vbNullString, "c:\temp", 4)
```

ShelveAlarm Method Example

The following example shelves an alarm in the **Alarm Summary** object, *AlarmSummaryOCX1*, with a given duration.

```
Public Sub ShelveAnAlarm(NodeName As String, TagName As String, ShelveDuration As Long)
Dim AppObj As Object
Dim PictureObj As Object
Dim CurrentObj As Object

nShelveTime = 100
If TypeName(Application) = "CFixApp" Then
' running in the workspace
Set AppObj = Application
Else
Set AppObj = App
If AppObj Is Nothing Then
Exit Sub
End If
End If
Set PictureObj = AppObj.ActiveDocument
For Each CurrentObj In PictureObj.Page.ContainedObjects
' Search for alarm ocx control in the display
If TypeName(CurrentObj) = "AlarmSummaryOCX" Then
If CurrentObj.Name = "AlarmSummaryOCX1" Then
CurrentObj.ShelveAlarm NodeName, TagName, ShelveDuration
Exit Sub
End If
End If
Next
End Sub
```

ShowAnimations Method Example

The following example opens the Animation dialog for the **Oval** object *Oval1*.

```
Oval1.Select
TestPicture.ShowAnimations
```

ShowBrowseDialog Method Example

The following example opens Expression Editor dialog box for the **ExpressionEditor** object *ExpressionEditor1*.

```
ExpressionEditor1.ShowBrowseDialog
```

ShowColorBox Method Example

The following example opens the color dialog box for the **ColorButton** object *ColorButton1*.

```
ColorButton1.ShowColorBox
```

ShowColorSelection Method Example

The following example opens the color selection dialog box for the **Picture***TestPicture*.

```
TestPicture1.ShowColorSelection True
```

ShowCustomPages Method Example

The following example opens the custom configuration dialog for the **Chart***Chart1.*

```
Chart1.ShowCustomPages
```

ShowPipePreviewDialog Method Example

This example opens the Modify Pipe Characteristics dialog box for the selected **pipe object(s)**.

```
Set ObjHelper = BuildObject("GeometryHelper")
Call ObjHelper.ShowPipePreviewDialog
```

ShowTaskWizard Method Example

The following example opens the Workspace's Task Wizard dialog box.

```
Application.ShowTaskWizard
```

ShowVBAProcedure Method Example

This example sets the focus of the VBA script window to the Click event of Rect2 when Rect2 is contained in the active document.

```
Dim oPic As Object
Dim oRect As Object
Set oPic = Application.ActiveDocument.Page
Set oRect = oPic.FindObject("Rect2")
oPic.ShowVBAProcedure "Click", oRect
```

ShowVisualBasicEditor Method Example

The following example opens the Workspace's Visual Basic Editor.

```
Application.ShowVisualBasicEditor
```

AlarmHornSilence Example

This example silences the alarm horn.

```
Private Sub Button1_Click ()
AlarmHornSilence
End Sub
```

SnapObjectsToGrid Method Example

The following example snaps the **Oval** and **Datalink** objects, *Oval1* and *Datalink1,* to grid.

```
Oval1.SelectObject False
DataLink1.SelectObject False
TestPicture.SnapObjectsToGrid
```

SpaceEvenly Method Example

The following example spaces the **Oval**, **Polygon**, and **RoundRectangle**objects *Oval1*, *Polygon1* and *RoundRect1* with equal horizontal spacing between them.

```
Oval1.SelectObject False
Polygon1.SelectObject False
RoundRect1.SelectObject False
TestPicture.SpaceEvenly 0
```

StartEvent Method Example

The following example starts the firing of **Event** object *FixEvent1* of the **Schedule** *TestSchedule*.

```
Dim iEvent As Object
Set iEvent = System.FindObject("TestSchedule.FixEvent1")

iEvent.StartEvent
```

StartTimer Method Example

The following example starts the firing of Timer object *FixTimer1* of the Schedule *TestSchedule*.

```
Dim iTimer As Object
Set iTimer = System.FindObject("TestSchedule.FixTimer1")

iTimer.StartTimer
```

StickToCursor Method Example

The following example creates a Rectangle object and specifies that the object stick to the cursor upon creation.

```
Dim iRect As Object
Set iRect = Application.ActiveDocument.Page.BuildObject("rect")

iRect.HorizontalPosition = 3#
iRect.VerticalPosition = 3#
iRect.StickToCursor
```

StopEvent Method Example

The following example stops the firing of Event object *FixEvent1* of the Schedule *TestSchedule*.

```
Dim iEvent As Object
Set iEvent = System.FindObject("TestSchedule.FixEvent1")

iEvent.StopEvent
```

StopGlobalPlayBack Method Example

The following example stops the historical playback option:

```
Private Sub Button1_Click ()
System.StopGlobalPlayback
End Sub
```

StopTimer Method Example

The following example stops the firing of event object *FixTimer1* of the Schedule*TestSchedule*.

```
Dim iTimer As Object
```

```
Set iTimer = System.FindObject("TestSchedule.FixTimer3")

iTimer.StopTimer
```

Stretch Method Example

The following example scales the **Oval** *Oval1* by a horizontal scale percentage of 10 and a vertical scale percentage of 20.

```
Oval1.Stretch 10, 20
```

SwitchLanguage Method Examples

The following example sets the language of the displayed text to *Spanish* for the active picture.

```
pic.LanguageDesired = 1034
pic.SwitchLanguage
```

- OR -

```
pic.SwitchLanguge (ES_Spanish)
```

NOTE: For a list of the ID numbers representing each language, refer to the Language Letter Acronyms table in the Exporting and Importing Language Files topic.

The following example shows how you to switch the language on only the AlarmSummary object, by clicking a rectangle in run mode.

```
Private Sub Rect1_Click()
    AlarmSummaryOCX1.SwitchLanguage RU_Russian

End Sub
```

SwitchMode Method Example

The following example sets the mode of the WorkSpace to run mode.

```
Application.SwitchMode 4
```

SynchronizeSecurity Method Example

Before you can use the following example, you must first add a VBA reference to SecuritySynchronizerDLL.DLL.

To add a VBA reference to SecuritySynchronizerDLL.DLL:

1. In the Visual Basic Editor, from the Tools menu, select References. The VBA References dialog box appears.
2. Click the Browse button and select the iFIX installation directory (normally C:\Program Files (x86)\Proficy\iFIX).
3. Select the SecuritySynchronizerDLL.dll file and click Open. A checked SecuritySynchronizerDLL reference is added in the VBA References dialog box.
4. Click OK, then exit the Visual Basic Editor.

This example creates the **SecuritySynchronizer** object and calls the **SynchronizeSecurity** method.

```
Dim objSecSynch as SecuritySynchronizer
Set objSecSynch = New SecuritySynchronizer
objSecSynch.UseLocalSecurity = True
objSecSynch.SynchronizeSecurity
```

This method runs the security synchronization process.

Before calling the **SynchronizeSecurity** method, you must set either one or both of the following properties to **True**:

- **UseLocalSecurity**
- **UseDomainSecurity**

If you set the UseDomainSecurity property to True, you must also set the **Domain** property to a valid Windows domain name.

T

TagGroupSubstitution Method Example

This example retrieves the substitution value of the tag group symbol @*tag1*@.

```
Dim szSubstitution As String
pic1.TagGroupSubstitution "@tag1@", szSubstitution
```

This method looks in the currently loaded tag group file for the tag and retrieves the substitution. The TagGroupSubstitution method is only available in the Runtime environment. If you pass it a string that is not a tag group symbol, it will return the same string.

TagGroupValue Method Example

This example retrieves the value of the tag group substitution of the tag group symbol @*tag1*@.

```
Dim vaValue As Variant
pic1.TagGroupValue "@tag1@", vaValue
```

This Runtime-only method looks in the currently-loaded tag group file to find the substitution for the passed tag and reads the current value. Calling this method is similar to calling the TagGroupSubstitution method, finding the Object based on the returned string, and then reading the value. If you pass it a string that is not a tag group symbol, it will return the same string.

ToggleDigitalPoint Subroutine Example

The following example toggles the block *DO1* between open and closed.

```
ToggleDigitalPoint "DO1"
```

ToggleManual Subroutine Example

The following example toggles the block *AI1* between manual and automatic modes.

```
ToggleManual "AI1"
```

ToggleScan Subroutine Example

The following example toggles the scan status of the block *AI1.*

```
ToggleScan "AI1"
```

U

UIActivate Method Example

The following example sets *CommandButton1* to its custom active mode.

```
CommandButton1.UIActivate
```

UIDeActivate Method Example

The following example sets *CommandButton1* to its custom inactive mode.

```
CommandButton1.UIDeActivate
```

Undo Method Example

The following example undoes the last action completed in the **Picture***TestPicture*.

```
TestPicture.Undo
```

UndoTransaction Method Example

The following example starts an undo transaction for the **Picture***TestPicture*.

```
TestPicture.UndoTransaction pUndoTransactionStart
```

UndoZoom Method Example

The following example restores the Line/Multiline chart to its default size.

```
Private Sub CommandButton9_Click()
LineChart1.UndoZoom
End Sub
```

UnGroup Method Example

The following example ungroups the selected group in the **Picture***TestPicture*.

```
TestPicture.UnGroup
```

UnloadTagGroupFile Method Example

The following example unloads the currently loaded tag group file from the iFIX picture *pic1*.

```
Private Sub CommandButton1_Click()
pic1.UnloadTagGroupFile
End Sub
```

UnShelveAlarm Method Example

The following example unshelves an alarm in the **Alarm Summary** object, *AlarmSummaryOCX1*.

```
Public Sub UnShelveAnAlarm(NodeName As String, TagName As String)
Dim AppObj As Object
Dim PictureObj As Object
Dim CurrentObj As Object
If TypeName(Application) = "CFixApp" Then
Set AppObj = Application
Else
Set AppObj = App
```

```
If AppObj Is Nothing Then
Exit Sub
End If
End If
' Search alarm summary object and unshelve alarm
Set PictureObj = AppObj.ActiveDocument
For Each CurrentObj In PictureObj.Page.ContainedObjects
If TypeName(CurrentObj) = "AlarmSummaryOCX" Then
If CurrentObj.Name = "AlarmSummaryOCX1" Then
CurrentObj.UnShelveAlarm NodeName, TagName
Exit Sub
End If
End If
Next
End Sub
```

Update_A_Dynamo_By_Name Method Example

The following code provides an example of the Update_A_Dynamo_By_Name method. This example is similar to the one that appears in the Update_A_Dynamo_By_Ref Method Example.

```
Public Function UpdateADynamo(DynamoInstanceFullyQualifiedName As String, MasterDynamoFullyQualifiedName As Strin
    Dim iDataSourceOption As DynamoDataSourceOption
    Dim iResult As Long 'UpdateDynamoResult
    Dim iPrompt As Long

    iDataSourceOption = g_WizardConfig.iDataSourceOption
    If g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE Then
        If g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_BUT_NOT_APPLY
        ElseIf g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_AND_APPLY
        ElseIf g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_NOT_UPDATE
        End If
    End If

'////// actual update call
    ' call Update A Dynamo
    PlugandSolve.GeometryHelperObj.Update_A_Dynamo_By_Name MasterDynamoFullyQualifiedName, DynamoInstanceFullyQua
    ' get a result string
    strReturnMsg = PlugandSolve.GeometryHelperObj.Get_Last_Result_String
    ' get the last user choice from the prompt
    If (g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE) And _
        (g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_NONE) Then
        iPrompt = PlugandSolve.GeometryHelperObj.Get_Last_Prompt_Value
        If (iPrompt = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL) Or _
            (iPrompt = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL) Or _
            (iPrompt = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL) Then
            g_ReturnFromPromptForChoice = iPrompt
        End If
    End If
'////// end of actual update call
'////// dummy for test
'    g_testcount = g_testcount + 1
'    If g_testcount = 3 Then
'        strReturnMsg = ">>>" & DynamoInstanceFullyQualifiedName & " was not updated  with " & MasterDynamoFullyQ
'        iResult = DYNAMO_NOTUPDATED
'    ElseIf g_testcount = 150 Then
'        strReturnMsg = ">>> User canceled"
'        iResult = DYNAMO_UPDATE_ABORTED
'    Else
'        strReturnMsg = DynamoInstanceFullyQualifiedName  & " was updated successfully with " & MasterDynamoFully
'        iResult = DYNAMO_UPDATED
'    End If
'////// end of dummy for test

    If iResult <= DYNAMO_UPDATED Then
        UpdateADynamo = DYNAMO_UPDATED
    ElseIf iResult <= DYNAMO_NOTUPDATED Then
        UpdateADynamo = DYNAMO_NOTUPDATED
    Else
        UpdateADynamo = DYNAMO_UPDATE_ABORTED
    End If
End Function
```

Update_A_Dynamo_By_Name2 Method Example

Refer to the Update_A_Dynamo_By_Ref2 Method Example. Update_A_Dynamo_By_Ref2 allows you to specify a Dynamo name, while Update_A_Dynamo_By_Name2 allows you to specify the Dynamo object's dispatch pointer.

Update_A_Dynamo_By_Ref Method Example

The following code provides an example of the Update_A_Dynamo_By_Ref method that appears in the modDynamoUpdater module of the Project_PlugandSolve VBA project.

```
Public Function UpdateADynamo2(objDynamoInstance As Fix2DDynamo.Fix2DDynamo, objMasterDynamo As Fix2DDynamo.Fix2DDynamo
    Dim strDIName As String
    Dim strDMName As String
    Dim iDataSourceOption As DynamoDataSourceOption
    Dim iResult As Long 'UpdateDynamoResult
    Dim iPrompt As Long
    Dim iUpdateOptions As Long ' update options

    iDataSourceOption = g_WizardConfig.iDataSourceOption
    If g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE Then
        If g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_BUT_NOT_APPLY
        ElseIf g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_AND_APPLY
        ElseIf g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_NOT_UPDATE
        End If
    End If

    ' Convert option parameters
    iUpdateOptions = &H0
    If iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE Then
        iUpdateOptions = UPDATE_OPTION_ON_MISMATCH_PROMT_FOR_CHOICE
    ElseIf iDataSourceOption = DYNAMO_UPDATE_AND_APPLY Then
        iUpdateOptions = UPDATE_OPTION_ON_MISMATCH_UPDATE_DYNAMO + UPDATE_OPTION_ON_MISMATCH_APPLY_DATA_SOURCES
    ElseIf iDataSourceOption = DYNAMO_UPDATE_BUT_NOT_APPLY Then
        iUpdateOptions = UPDATE_OPTION_ON_MISMATCH_UPDATE_DYNAMO
    End If
    If g_WizardConfig.bKeepSize = True Then
        iUpdateOptions = iUpdateOptions + UPDATE_OPTION_RESIZE_INSTANCE
    End If
    If g_WizardConfig.bKeepCaption = True Then
```

```
        iUpdateOptions = iUpdateOptions + UPDATE_OPTION_SAVE_CAPTIONS
    End If

    ' Add new option if Dynamo instance conversion
    If g_iDynamoToolType = DYN_QUICK_CONVERTER Or _
        g_iDynamoToolType = DYN_CONVERTER_WIZARD Then
        iUpdateOptions = iUpdateOptions + UPDATE_OPTION_UPDATE_ON_CONVERSION
    End If

    ' call Update A Dynamo
    PlugandSolve.GeometryHelperObj.Update_A_Dynamo_By_Ref2 objMasterDynamo, objDynamoInstance, iUpdateOptions, mobjStrM
    ' get a result string
    strReturnMsg = PlugandSolve.GeometryHelperObj.Get_Last_Result_String
    ' get the last user choice from the prompt
    If (g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE) And _
        (g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_NONE) Then
        iPrompt = PlugandSolve.GeometryHelperObj.Get_Last_Prompt_Value
        If (iPrompt = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL) Or _
            (iPrompt = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL) Or _
            (iPrompt = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL) Then
            g_ReturnFromPromptForChoice = iPrompt
        End If
    End If
'////// end of actual update call

'    If iResult <= DYNAMO_UPDATED Then
'        UpdateADynamo = DYNAMO_UPDATED
'    ElseIf iResult <= DYNAMO_NOTUPDATED Then
'        UpdateADynamo = DYNAMO_NOTUPDATED
'    Else
'        UpdateADynamo = DYNAMO_UPDATE_ABORTED
'    End If

    If (iResult And UPDATER_RESULT_SUCCESS_BIT) > 0 Then
        UpdateADynamo2 = DYNAMO_UPDATED
    End If
    If (iResult And UPDATER_RESULT_DYNAMO_NOT_UPDATED_BIT) > 0 Then
        UpdateADynamo2 = DYNAMO_NOTUPDATED
    End If
    If (iResult And UPDATER_RESULT_USER_CANCELLED_BIT) > 0 _
        Or (iResult And UPDATER_RESULT_SUCCESS_BIT) = 0 Then
        UpdateADynamo2 = DYNAMO_UPDATE_ABORTED
    End If
End Function
```

To view this code in context:

1. In Classic view, from the WorkSpace menu, select Visual Basic Editor.

   -Or-

   In Ribbon view, on the Home tab, in the WorkSpace group, click Visual Basic Editor.
2. In the tree view, double-click the Project_PlugandSolve folder, and then the Modules folder, and finally the modDynamoUpdater.
3. Search for UpdateADynamo to locate this code.

Update_A_Dynamo_By_Ref2 Method Example

The following code provides an example of the Update_A_Dynamo_By_Ref2 method that appears in the modDynamoUpdater module of the Project_PlugandSolve VBA project.

```
Public Function UpdateADynamo(objDynamoInstance As Fix2DDynamo.Fix2DDynamo, objMasterDynamo As Fix2DDynamo.Fix2DDynamo,
    Dim strDIName As String
    Dim strDMName As String
    Dim iDataSourceOption As DynamoDataSourceOption
    Dim iResult As Long 'UpdateDynamoResult
    Dim iPrompt As Long

    iDataSourceOption = g_WizardConfig.iDataSourceOption
    If g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE Then
        If g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_BUT_NOT_APPLY
        ElseIf g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_UPDATE_AND_APPLY
        ElseIf g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL Then
            iDataSourceOption = DYNAMO_NOT_UPDATE
        End If
    End If

'////// actual update call
    ' call Update A Dynamo
    PlugandSolve.GeometryHelperObj.Update_A_Dynamo_By_Ref2 objMasterDynamo, objDynamoInstance, iDataSourceOption, mobjS
    ' get a result string
    strReturnMsg = PlugandSolve.GeometryHelperObj.Get_Last_Result_String
    ' get the last user choice from the prompt
    If (g_WizardConfig.iDataSourceOption = DYNAMO_PROMPT_FOR_CHOICE) And _
        (g_ReturnFromPromptForChoice = PROMPT_DLG_SEL_NONE) Then
        iPrompt = PlugandSolve.GeometryHelperObj.Get_Last_Prompt_Value
        If (iPrompt = PROMPT_DLG_SEL_UPDATE_NO_DS_APPLY_TO_ALL) Or _
            (iPrompt = PROMPT_DLG_SEL_UPDATE_ATTEMPT_MATCH_APPLY_TO_ALL) Or _
            (iPrompt = PROMPT_DLG_SEL_DO_NOT_UPDATE_APPLY_TO_ALL) Then
            g_ReturnFromPromptForChoice = iPrompt
        End If
    End If
'////// end of actual update call
'////// dummy for test
'    g_testcount = g_testcount + 1
'    strDIName = objDynamoInstance.Name
'    strDMName = objMasterDynamo.Name
'    If g_testcount = 3 Then
'        strReturnMsg = ">>>" & strDIName & " was not updated  with " & strDMName
'        iResult = DYNAMO_NOTUPDATED
'    ElseIf g_testcount = 150 Then
'        strReturnMsg = ">>> User canceled"
'        iResult = DYNAMO_UPDATE_ABORTED
'    Else
'        strReturnMsg = strDIName & " was updated successfully with " & strDMName
'        iResult = DYNAMO_UPDATED
'    End If
'////// end of dummy for test

    If iResult <= DYNAMO_UPDATED Then
        UpdateADynamo = DYNAMO_UPDATED
    ElseIf iResult <= DYNAMO_NOTUPDATED Then
        UpdateADynamo = DYNAMO_NOTUPDATED
    Else
        UpdateADynamo = DYNAMO_UPDATE_ABORTED
    End If
End Function
```

▶ **To view this code in context:**

1. In Classic view, from the WorkSpace menu, select Visual Basic Editor.

   -Or-

   In Ribbon view, on the Home tab, in the WorkSpace group, click Visual Basic Editor.
2. In the tree view, double-click the Project_PlugandSolve folder, and then the Modules folder, and finally the modDynamoUpdater.
3. Search for UpdateADynamo to locate this code.

UpdateBackgroundObject Method Example

The following example creates the object *TimerTest* in the **Schedule***TestSchedule* and transfers it to the corresponding schedule running in the background **FixBackgroundServer** application.

```
Dim TestSchedule As Object
Dim iTimer As Object
Set TestSchedule = System.FindObject("TestSchedule")

Set iTimer = TestSchedule.BuildObject("FixTimer")

iTimer.Name = "TimerTest"
iTimer.TriggerType = 1
iTimer.Interval = 2000
TestSchedule.UpdateBackgroundObject "TimerTest",
 bkAdd, bkRun
```

UpdateConnectionParameters Method Example

The following example updates the UpdateRate to 30.0, Deadband to 5.0 and Tolerance 15.0 to for the **Oval** object *Oval2*.

```
Oval2.UpdateConnectionParameters "VerticalFillPercentage",
 30.0, 5.0, 15.0
```

UpdateDefinition Method Example

The following example updates the definitions contained in a tag group file.

```
Dim sTokenList(4) as String, TokenList as Variant
Dim sReplacementList(4) as String, ReplacementList as Variant

Dim sDescriptionList(4) as String, DescriptionList as Variant

Dim TGD As Object
Set TGD = CreateObject("TagGroupDefinitionInterfaceDll.TagGroupDefinitionInterface")
```

```
TGD.RetrieveDefinition "Test", 4, TokenList, ReplacementList,
 DescriptionList
TokenList(2) = "Tag3"
TokenList(3) = "Tag4"
ReplacementList (2) = "FIX32.NODE2.AI1.F_CV"
ReplacementList (3) = "FIX32.NODE2.AI2.F_CV"
DescriptionList (2) = "Temperature for Node 2"

DescriptionList (3) = "Pressure for Node 2"
' Create the tag group file object
Set TGD = CreateObject("TagGroupDefinitionInterfaceDll.TagGroupDefinitionInterface")

TGD.UpdateDefinition "Test", 4, TokenList, ReplacementList,
 DescriptionList
Set TGD = Nothing
```

UserFormPointToLogical Method Example

NOTE: The UserFormPointsToLogical Method Example applies to both Enhanced and Logical Coordinates.

The following example converts the Top, Left coordinates from 30, 50 in "UserForm Point" coordinates to logical units or postscript points for the **Picture** *TestPicture*.

```
Dim dTop As Double
Dim dLeft As Double
dTop = 30
dLeft = 50
TestPicture.UserFormPointToLogical dTop, dLeft
```

V-Z

ValidateSignature Method Example

The following example creates the ESignature object, checks to see if the node is enabled for electronic signature, validates the signature, sends a message to the audit trail, and retrieves the full name of the signer.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bValidSig As Boolean
Dim UserID As String
Dim FullName As String
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignatureFactory")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
bValidSig = False ' will be set to TRUE by ValidateSignature if signature is valid
'Validate the signature
ESig.ValidateSignature "admin", "admin", 1, bValidSig, Ucase(UserID)
If bValidSig = True Then
'Send a message to the audit trail
ESig.SendSignedOperatorMessage "Action Description", "", "", Ucase(UserID), "Comment"
'Get the full name of the signer
ESig.GetFullname Ucase(UserID), Ucase(FullName)
```

```
'Show Results
MsgBox "Action performed by admin (" + FullName + ") " + "Comment"
Else
MsgBox "Invalid Signature."
End If
Else
MsgBox "Signature is not enabled on this node."
End If
```

ValidateSignatureAndWriteValue Method Example

The following example creates the ESignature object, checks to see if the node is enabled for electronic signature, and determines if a specified tag (FIX32.thisnode.D01.F_CV) requires electronic signature. If required, the example validates the signature, writes the new value, and sends a message to the audit trail.

```
Dim ESig As Object
Dim bNodeSignEnabled As Boolean
Dim bSigRequired As Boolean
Dim bVerify As Boolean
Dim bContinuousUse As Boolean
Dim nInfo As Integer
Dim NewValue As Variant
'Create the ESignature object
Set ESig = CreateObject("ElectronicSignature.ESignatureFactory")
'Check if node is enabled for electronic signature
ESig.IsNodeSignEnabled bNodeSignEnabled
If bNodeSignEnabled = True Then
'Check if tag requires electronic signature
ESig.Initialize "Fix32.thisnode.DO1.F_CV"
ESig.IsSignatureRequired 0, bSigRequired, nInfo, bVerify, bContinuousUse
If bSigRequired = True Then
'Validate the signature, write the new value and send a message to the audit trail
NewValue = 1
If bVerify = False Then
ESig.ValidateSignatureAndWriteValue 0, NewValue, "admin", "admin", "Perform Comment Example"
Else
ESig.ValidateSignatureAndWriteValue 0, NewValue, "admin", "admin", "Perform Comment Example", "supervisor1", "GE
End If
Else
MsgBox "Signature is not required for this tag."
End If
Else
MsgBox "Signature is not enabled on this node."
End If
```

ValidateSource Method Example

The following example validates the *AI1* source for the **Oval** *Oval1*.

```
Dim iStatus As Long
Dim iObj As Object
Dim sPropName As String
Oval2.ValidateSource "AI1", iStatus, iObj, sPropName
```

ValueTimeFromXY Method Example

The following example retrieves the value, date and type which corresponds to the coordinates 10,25 for the **Pen** *Pen1*.

```
Dim dValue As Double
Dim dDate As Date
Dim bReal As Boolean
Pen1.ValueTimeFromXY 10, 25, dValue, dDate, bReal
```

WriteValue Subroutine Example

The following example writes the value 1 to the block *DO1*.

```
WriteValue 1, "DO1"
```

Write Method Example

The following example:

- Creates a data system ocx *FDS*;
- Adds a data **Group***DataGroup1* to the **Groups** collection;
- Adds a **DataItem** to the **DataItems** collection;
- Writes the value of the **Group***DataGroup1*;
- Writes a value of 45 to the **DataItem**

```
'Create the Data System OCX
Dim FDS As Object
Set FDS = CreateObject("FixDataSystems.Intellution FD
 Data System Control")
'Add a group to the Groups collection
FDS.Groups.Add ("DataGroup1")
FDS.Groups.Item("DataGroup1").DataItems.Add("Fix32.THISNODE.AI1.F_CV")

'Write DataGroup1
FDS.Groups.Item("DataGroup1").Write
'Write the DataItem
FDS.Groups.Item("DataGroup1").DataItems.Item(1).Write(45)
```

XYFromValueTime Method Example

The following example retrieves the x and y coordinates for the value of 10 at the time 11:05 PM for the **Pen** *Pen1*.

```
Dim dX As Double
Dim dY As Double
Pen1.XYFromValueTime 10, #11:05:00 PM#, dX, dY
```

XYHitTest Method Example

The following example retrieves the information for the pen at the coordinate x,y for the chart **Chart** *Chart1*. Note that the x and y parameters come from**MouseUp** and/or **MouseDown** event prototypes.

```
Dim dDate As Date
Dim dValue As Double
Dim sPenName As String
Dim iPen As Object
Dim lPenNum As Long
Dim bReal As Boolean
Chart1.XYHitTest x, y, dDate, dValue, sPenName, iPen, lPenNum,
 bReal
```

Zoom Method Example

The following example zooms in on the **Chart** with a high and low vertical value of 50 and 10 and a high and low horizontal value of 75 and 35 for *Chart1*.

```
Chart1.Zoom 50, 10, 75, 35
```

ZoomToFit Method Example

By default, Zoom to Fit is enabled for Run mode for all pictures with Enhanced Coordinates enabled. The following example shows how to disable this "Zoom to Fit" for a picture. In this example, the picture initialization code sets "ME.ZoomToFit False." Then, it sets the Zoom factor to 1 (100%) with the "ME.Zoom = 1" line:

```
Private Sub CFixPicture_Initialize()
Me.ZoomToFit False
Me.Zoom = 1
End Sub
```

NOTE: For more information on Enhanced Coordinates, refer to the Picture Coordinate Systems topic in the Creating Pictures e-book.

Index

A

access 404-405

acknowledge alarms 555-556

activate 354, 513, 532

active 40-41

ActiveX 7, 232, 380

add 354, 357-361

AddDataSet Method 356

AdvancedGraphics property 41

alarm 74-75, 219-220, 406, 546

Alarm Summary 5

alarmhorn 42, 506

AlarmHornEnabled 556

AlarmHornSilence 506, 607

AlarmHornSilence subroutine 557

AlarmHornToggle 557

alarms 42-43, 114-115, 132, 310, 318, 352-353, 383, 410, 424, 428-432, 436-437, 463, 477, 485-486, 495, 497, 502, 516, 533-534, 555-557, 559-560, 607

alias 572

align 361

alignment 43

always on top 223

angle 47, 120, 245, 278

animations 11-13, 503

application 5, 48

arc 6

ascending 274

attributes 426

author 49

automatic mode 579, 582

AutoMinMaxPaddingY property 50

autoscale 362

autoupdate rate 52

axis 45, 53, 257, 266, 269, 293, 319-320

B

backdrop 54-59

background 54, 59-61, 522

background color 223

bars 265

BarVal property 62

base 62

BestFitWithCenter 63

bitmap 6, 378

BitmapGradientMode property 63

blend 55, 64

blink 64-65, 218-219, 246

block 585

border 55-56

BorderTypes property 65

bottom 66-67

bounding 411

bounding rectangle 68, 417

build 363

button 68-70

C

CacheEnabled 69

caps 122, 279

caption 71

category 71

center 66, 72-73, 176, 245, 303

change 536

characters 74, 199, 206

chart 6, 12-13, 52

chartdata 52

ChartFontSize property 74

CheckForSeverityIncrease 75

CheckSyntax Method 366

chord 6

class 76

click 119, 535, 537, 541, 543, 545, 550

close 368, 535, 558-559

collection 14, 275

collections 82, 102, 222

color 53, 55-56, 59-60, 77, 111, 137, 163, 173, 179, 219-220, 247, 295, 306, 310, 317, 323, 424, 445, 484-485, 495, 497, 504, 536, 564

colorbutton 6

column 413, 456

CombinationKey 77

comments 78

commit 369

configure 539

ConnectDataSet method 370

connections 80, 130, 370-371, 388, 413-414, 445, 461, 523, 563

construct 364, 371

containment 81

contextID 82

control container 6

control points 388

Convert_A_Group_To_A_Dynamo_By_Name method 371

Convert_A_Group_To_A_Dynamo_By_Ref method 373

ConvertToEnhancedCoordinates method 376

ConvertToOriginalCoordinates method 377

coordinates 375, 453-454, 464-465, 524

copy 378, 391

count 82

Coupled_Activate_Workspace_UI method 379

Coupled_DeActivate_Workspace_UI method 379

create 363, 369, 381, 508

CreateDynamoByGrouping Method 380

cursor 267, 295

cut 381

D

data 7, 83, 88, 93, 158, 191, 237, 275, 319, 423, 468, 475, 494, 536

data entry 79

data source 94, 159, 254, 561

database 585

database functions:summaries 584

datalink 7

dataservers 89

DataSetColor property 89

DataShadows property 90

datasystem 10

date 83-84, 86-87, 133, 259, 279

daylightsavingtime 90

days 84, 91-92

deactivate 382, 514, 538, 551

default 93-94, 223-226, 382

delete 383-385, 387, 471

DeleteDataSet method 384

delta 314

demand 386

descending 274

description 95, 178, 220

deselect 386

DeskColor property 96

destroy 387

device 417

digital point 559, 570, 581

digits 92, 329

DigitsOfPrecision property 97

direction 251

disable 387

disable alarms 559

DisableAutoScale property 97

display 157, 188, 388, 442

DisplayShelvedAlarms 99

displaystring 100

document 40, 101-103, 403-404, 472

DocumentHeightEx property 102

documents 7-8, 10, 12-13, 17-18, 102

DocumentWidthEx property 103

doubleclick 537

drilldown 44

DSDescription property 105

DSLegendAvgerageOverRangeColWidth property 105

DSLegendCurrentValColWidth property 105

DSLegendDescriptionColWidth property 106

DSLegendEngUnitsColWidth property 106

DSLegendHighLimitColWidth property 106

DSLegendHighOverRangeColWidth property 107

DSLegendLowLimitColWidth property 107

DSLegendLowOverRangeColWidth property 107

DSLegendMask property 108

DSLegendQualityColWidth property 109

DSLegendSourceColWidth property 109

DSPosition property 109

duplicate 391

duration 110, 417, 490

dynamo 8, 564

Dynamo_Description Property 110

Dynamo_ID Property 110

DynamoSet object 8

E

edge 60-61, 111-112

edit 113, 391, 538-539

editor 506

egu 131, 203

elbows 113

empty 446

enable 116, 153, 158, 297, 392

enable alarms 406, 560

enabled 42, 406

EngUnits property 124

EnhancedCoordinates property 124

Enumerate_All_Dynamos Method 392

Enumerate_All_Groups method 392

Enumerate_Top_Level_Dynamos Method 393

Enumerate_Top_Level_Groups method 393

Error Handling 418

errors 125, 418

esignature 8, 114

event 9, 238, 508-509

events 126, 356, 418, 450, 531

exact 127

Expandable property 127

ExportLanguageFile 399

expred 92

expressioneditor 9

extend 128, 389

F

fadecolor 57, 129

fadetype 57, 129

false 548, 551

fetch 49

FetchDataSetLimits property 130

file 131

fill 162, 322

filter 132

find 10, 400-402, 473, 561-562, 567

fire 386

FixGeometryHelper object 10

FixGetMyname 589

focus 492, 533

font 135-136, 284, 408, 437, 498

ForceVerticalPoints property 136

form 564-566

format 11, 139, 240, 494, 500

front 362

G

Get_Last_Prompt_Value Method 408

Get_Last_Result_String Method 409

GetEventHandlerIndex Method 418

GetNumberOfDataSets method 422

GetProcedureIndex 425

GetSignature Method 433

GetSignatureAndWriteValue Method 434

gradient 58, 151

Gradient property 150

GraphBackColor property 151

GraphForeColor property 151

GraphPlusTable property 152

GraphPlusTableMenu property 152

grid 153-155, 163, 207, 209-210, 212, 251, 261-262, 270, 273, 323, 507

GridInFront property 153

GridLinesToShow property 154

GridStyle property 154

group 11, 442, 515

groups 11, 155

H

HDA 590

headers 261

headings 413

height 101, 156, 214, 224, 227, 250, 326, 330

help 82, 156, 352

helper functions 588, 590

hide 157

highlight 158-159

HistDatalink object 12

HistMode property 160

HistogramChart object 12

historical 161, 262

HistUpdateRate property 162

I

image 165, 253, 272, 285

images 85, 104, 357, 384, 452

import 443

index 166, 425

indirection 420, 492

initialize 443-444, 539

input 167

insert 444

interval 154, 168, 179, 420, 492

intervalmilliseconds 168

IsNodeSignEnabled Method 447

IsSignatureRequired Method 447

IsSignatureRequiredForList Method 448

item 449

items 207

J

Justification Property 172

K

key macro collection 10

key macros 10

keydown 540

keyup 541

L

LabelBold property 173

LabelFont property 174

LabelItalic property 174

labels 173, 210, 293, 320

LabelUnderline property 175

layer 175

layers 98

LCL property 176

left 66, 176, 303, 326, 330

legend 12, 177-185, 263, 358, 470

length 53, 456

levels 358, 421, 469, 471

limit 190

limits 131, 362, 560

line 220-221, 263

linear 12, 185

LineChart object 12

LineConnector object 13

lines 12-13, 16, 80, 186, 199, 207, 209-210, 212, 261, 389-390

LineType property 186

load 453, 542

Load_TS_List method 452

location 501

lockstarttime 187

log in 188-189, 407, 567

log out 407

lookup 13

LWL property 191

M

MainTitle property 192

MainTitleBold property 192

MainTitleFont property 192

MainTitleItalic property 193

MainTitleUnderline property 193

manager 301

manual mode 580, 582

ManualMaxX property 194

ManualMaxY property 194

ManualMinX property 195

ManualMinY property 195

ManualScaleControlY property 196

MarkDataPoints property 197

markers 197, 316

Master Property 198

Max_Dynamo_Desc_Length Property 198

MaxXAxisLabels property 200

menu 289

menus 389

message 487

method 55, 57-58, 64, 129, 150, 365, 370, 378, 388, 452, 614

methods 341, 450

milliseconds 98

mode 180, 279, 281, 333, 511

modify 455

MonoDeskColor property 201

MonoGraphBackColor property 201

MonoGraphForeColor property 201

MonoShadowColor property 202

MonoTableBackColor property 202

MonoTableForeColor property 202

MonoTextColor property 203

mouse 119, 381

MouseUpOffObject event 546

move 456, 543

MyNodeName 204

N

name 140, 189-190, 204-205, 227, 331, 419

names 131, 139, 332, 572

next 205

NlsGetText 590

NoSaveOnClose property 206

NumOfPoints property 209

NumPointsToGraph property 209

NumRandomSubsets property 211

NumScrollingSubsets property 211

O

object 212, 359, 363, 471, 487

objects 2, 81

off scan 568

ole 232, 380-381

on scan 569

opc 213-214

open 457-460, 569-571

origin 215-216

output 144, 216

output value 94

oval 13

owner 216

P

page 217

parameters 414

paste 461-462

path 62, 102, 134, 205, 218, 227, 250, 301

pause 463

pause indicator 218-220

pen 13, 85, 220, 359, 385, 394

picture 14, 85, 223-228, 360, 391, 453, 457-460, 472, 474-475, 516, 559, 562, 570-571, 577-578

pictures:securing 253

pie 14, 228

Pipe object 14

PipeConnector object 14

pipes 375, 505

PlottingMethod property 229

point 10, 280, 333, 335

points 72-73, 122, 200, 208, 211, 361, 385, 424, 444, 495

PointType property 229

polygon 14

polyline 14, 389

position 164, 295, 324

print 466, 573

procedure 361

procedures 13-14, 16, 231-232, 356, 425, 505

ProjectPath Property 233

PromptToChangePassword Method 467

properties 20, 390, 451

property 362, 425, 496

PutBackdropGradAngle 340

Q

queue 238

QuickConfigure property 238

QuickStyle property 239

quit 467

R

radius 240

ramp 574

rate 242, 300

read 467

read values 575-576

RealTimeSPCDataSet object 15

rectangle 15

refresh 88, 468, 548

remove 468-472

replace 10, 400, 402, 472-475, 577-578

report 573

reset 242, 475-476

resizable 224

resize 243

resolve 477

restricting access to pictures 253

resume 477

retrieve 478

revision 244

Revision Property 244

right 67, 244-245, 303

RightAngleLineConnector object 15

rotate 72, 245, 479

rotation 48

rounded rectangle 15

row 67, 264, 304

run 479

run indicator 246-247, 317

runtime 119

S

save 248, 480-482

SaveThumbnail property 248

scale 164-165, 249, 311, 324-325, 510

scan 568-569, 583

scheduler 15

screen 140, 214-215, 250

script 496

scripting 13, 15-16, 166

scroll 251-252, 265, 482-483

ScrollToPosition method 483

search 9, 400-403, 473

securing pictures 253

security 253, 376, 404-407

security areas 253

SecuritySynchronizer 16, 46, 78, 96, 512

SelectedDatasource Property 254

SelectedFieldName Property 254

SelectedNodeName Property 254

SelectedTagName Property 255

selection 82, 170-171, 385, 387, 484, 486-487, 549-550

selections 81, 255

separator 564

servers 95

SetSymbolValues subroutine 580

SeverityIncreased 551

ShadowColor property 256

shapes 255

shell 581

show 505

ShowDSLegend property 259

ShowTimeStamp property 268

ShowXAxis property 270

ShowYAxis property 271

silencealarmhorn 506

size 51, 455

sort 116, 274

source 130, 275-276, 300, 461, 497, 527

sources 16, 275

space 507

SPCBarChart object 16

SPCChartType property 276

SPCInterval property 277

SPCType property 277

start 333, 508

startup list 360, 472

startup mode 333

state 68, 331

statistics 476

status 283

status bar 100, 265, 284

stop 509

stretch 510

StretchMode property 286

strikethrough 286

style 54, 56, 60-61, 69, 111, 132, 138, 163, 221, 323 subroutine 563, 607

subroutines 553

substitution 512-513

SubTitle property 287

SubTitleBold property 287

SubTitleFont property 288

SubTitleItalic property 288

SubTitleUnderline property 288

summaries 1, 19, 340, 530, 552

SwitchLanguage 510

symbol 513

system 16, 289

system menu 225

system tree 101

T

table 77

TableBackColor property 290

TableFont property 290

TableForeColor property 291

tag groups 17, 453, 477-478, 512-513, 516, 523, 542, 571, 578

tags 180

targets 208, 388, 427

task wizard 505

text 17, 113

TextColor property 291

tge 571, 578

thickness 291

Thumbnail property 292

ticks 212, 294, 320

time 86, 118, 122, 133, 204, 266, 280-282, 295, 298, 438-439, 483, 500-501, 527-528

time axis 17, 45, 294

TimeCursorStyle property 296

timeout 255, 297

timer 17, 297, 508-509

timezonebiasexplicit 298

timezonebiasrelative 299

title 53, 266, 269, 299, 320

title bar 226

titles 294

toggle 144, 149, 300, 581, 583

toggle state 582

tolerance 300

toolbar 17, 301, 443

toolbar manager 301

TooltipOption property 302

tooltips 120, 267, 296

top 46, 302-304, 327, 332

TotalFilteredAlarms 304

TranslateOnOpen 305

transparent 305-306

TreatSinglePointsAsLines property 307

trend 284

trigger 307

trim 308, 390

true 550, 552

TruncateTitles property 309

type 126, 222, 279, 282, 309

U

UCL property 310

underline 310

undo 368, 514

ungroup 515

units 47, 181, 312

unload 516, 542

update 522, 524

update rate 94

Update_A_Dynamo_By_Name Method 517

Update_A_Dynamo_By_Name2 method 518

Update_A_Dynamo_By_Ref Method 520

Update_A_Dynamo_By_Ref2 method 521

UpdateRate property 312

UseDefaultYAxisSettings property 313

UseDSLimits property 315

user 407, 414, 439, 489, 566

user globals 17

User Preferences 18, 318

UWL property 318

V

validate 276, 525, 527

value 87, 159-160, 167, 185, 190-191, 216, 269, 319, 415, 489, 513, 527

value axis 18, 45

values 528

variable 547

variables 18, 321, 478

VBA 505

VBA object 115

version 321

view 382, 408

ViewingStyle property 325

viewport 326-327

visibility 59, 247, 328

visible 67, 225, 304

visibleunacknowledgedalarms 328

W

width 61, 103, 112, 138, 155, 215, 221, 226, 228, 249-250, 327, 329, 332

window 41, 403-404, 439, 501

windows 330-332

wizards 332

WorkSpace 333, 382

write 528

write point 583

WriteValue Subroutine 583

X

XAxisLabel property 334

XAxisType property 335

Y

YAxesStyle property 336

YAxisAlwaysVisible property 336

YAxisLabel property 337

YAxisLongTicks property 337

YAxisScaleControl property 337

YAxisTitle property 338

Z

zoom 338-339, 476, 529

ZoomToFit Method 530

ZoomType property 339