Proficy Historian 9.1

REST APIs Reference Manual

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

Chapter 1. Historian REST APIs

Introduction to Historian REST APIs

Historian APIs

Historian is a high performance data archiving system designed to collect, store, and retrieve time-based information at extremely high speed efficiently. The Historian environment provides a set of REST APIs to query data from the archives.

This document provides links for setting up your development environment, as well as information for getting started with the Historian services and their associated APIs.

Important: Starting Historian 8.0, using a port number in request URL is not supported. Hence, do not include the port number 8443 while working with REST APIs.

Also, the default admin client name is changed from admin to hostname.admin

Example:

```
curl -u admin:adminsecret https://<nodename>:8443/uaa/oauth/token -d
       'grant_type=client_credentials'
```

should be replaced with

```
curl -u hostname.admin:adminsecret https://<nodename>/uaa/oauth/token -d
       'grant_type=client_credentials'
```

See the following topics for more information:

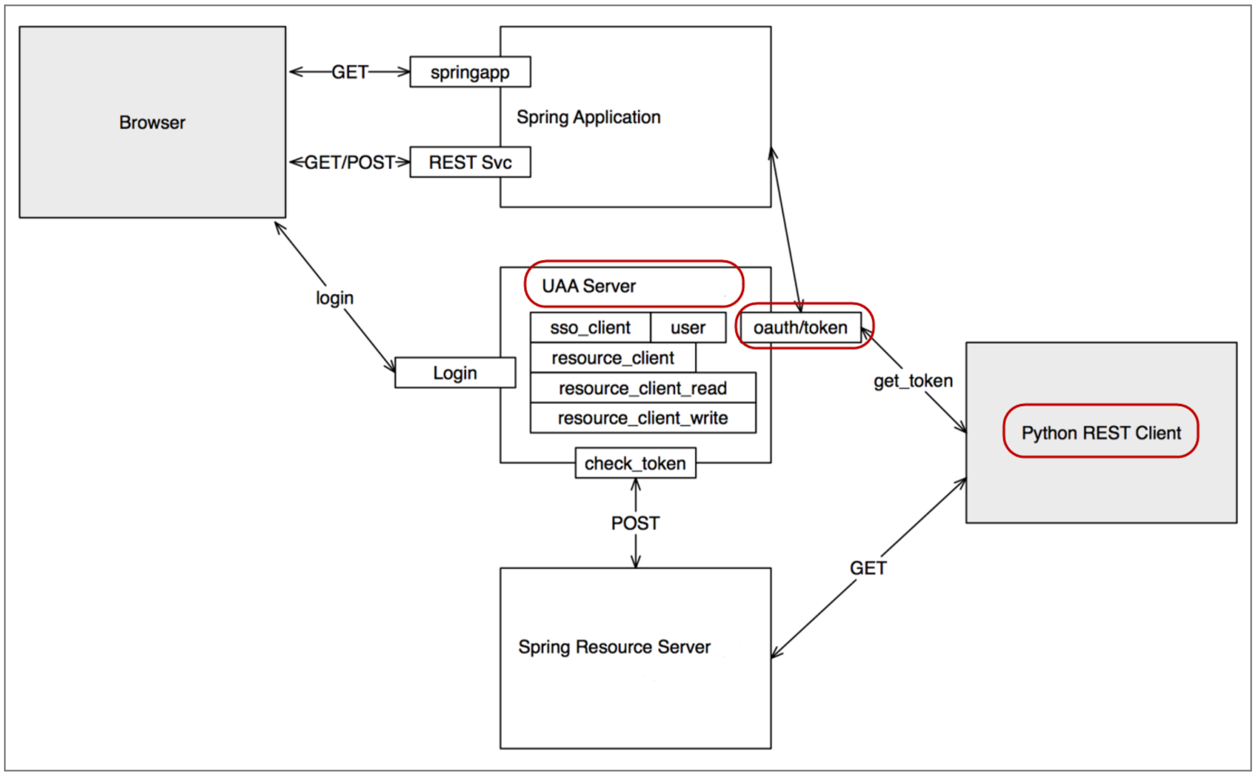
- About Security and Authentication *(page 3)*
- Standards *(page 5)*
- API Methods *(page 6)*
- API Status Messages *(page 6)*

About Security and Authentication

For security purposes, Historian uses the User Account and Authentication (UAA) service as a trusted source of tokens issued for authentication. The UAA is a multi-tenant identity management service, used in Cloud Foundry, but also available as a standalone OAuth2 server. Its primary role is as an OAuth2 provider, issuing tokens for client applications to use when they act on behalf of Cloud Foundry users. It can also authenticate users with Cloud Foundry credentials, and can act as an SSO service using those credentials, or others. It contains endpoints for managing user accounts, registering OAuth2 clients, and other management functions.

The following diagram shows how the UAA Server functions with a Python REST client:

Figure: UAA Server and Python REST Client



Authorization

For exchanging data between the client-server system, user authentication is required. Once you have provided your client credentials, an access or bearer token is generated. This token is used for the REST APIs.

cURL command format for generating an oauth token: `curl -u <Client-Id>:<Client-secret> https://<nodename>:8443/uaa/oauth/token -d 'grant_type=client_credentials'`

Sample cURL command: `curl -u hostname.admin:adminsecret https://<nodename>:8443/uaa/oauth/token -d 'grant_type=client_credentials'`

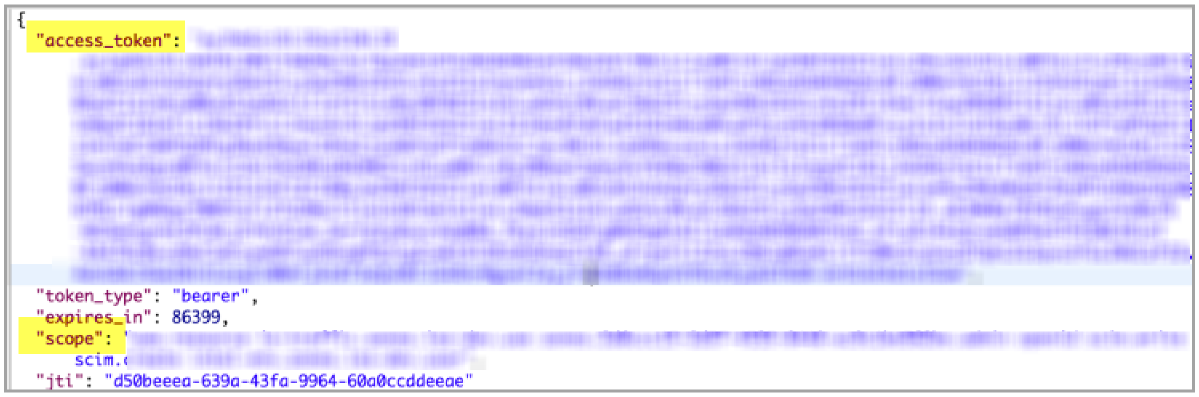
If 'grant_type=client_credentials' does not work, use 'grant_type=password'.

Example:

```
curl -u user1:mypassword http://WIN-2T4JG45H2TI:8080/uaa/oauth/token -d
 grant_type=password
```

In the following image, the actual token text is blurred for security concerns.

Figure: OAuth Access Token Sample



```
{
  "access_token": [redacted]
  "token_type": "bearer",
  "expires_in": 86399,
  "scope": [redacted]
      scim.[redacted]
  "jti": "d50beeea-639a-43fa-9964-60a0ccddeeae"
```

Client applications can access data using service REST API endpoints. Your application makes an HTTP request and parses the response. You can use any web-development language to access the APIs.

Standards

Historian APIs use a REST application architecture constrained by Hypermedia as the Engine of Application State (HATEOAS) that distinguishes it from most other network application architectures. Therefore, a client interacts with a network application entirely through hypermedia provided dynamically by application servers. The REST client doesn't need prior knowledge about how to interact with a particular application or server beyond a basic understanding of hypermedia.

As defined by the query parameters, the Historian APIs use "search" functions to access raw data using cURL and HTTP, while responses are in JSON format.

cURL is a command-line utility used to transfer data from or to a server, using one of the supported protocols, such as DICT, FILE, FTP, FTPS, GOPHER, HTTP, HTTPS, IMAP, IMAPS, LDAP,LDAPS, POP3, POP3S, RTMP, RTSP, SCP, SFTP, SMTP, SMTPS, TELNET and TFTP. The command is designed to work without user interaction.

cURL offers many useful functions such as proxy support, user authentication, FTP upload, HTTP post, SSL connections, cookies, file transfer resume, user and password authentication, and more.

You can run the sample commands provided in this document from Bash on Windows in the Windows operating system, and also in Linux Shell in the Linux operating system.

As a prerequisite, make sure you install cURL on your system, if it is not already installed. Run the `curl --version` command on Windows Bash or Linux shell to check if cURL is installed on your system.

Important: Do not create your own URIs. Instead, use the links in this document and in the responses to navigate between resources.

API Methods

The Historian APIs use GET, POST, PUT, and DELETE methods.

Method	Usage
GET	Retrieves a resource.
POST	Creates (or adds) a resource.
PUT	Updates a resource.
DELETE	Removes a resource.

API Status Messages

In its use of the following HTTP status codes, the Historian API services adhere as closely as possible to standard HTTP and REST conventions.

Status Code	Usage
200 OK	Success message. The request has completed.
201 Created	Success message. A new resource has been created. The resource URI is available from the location header in the response.
204 No Content	Success message. An update to an existing resource has been applied.
400 Bad Request	Error message. The request was malformed. The response body provides additional information.
401 Unauthorized	Error message. Either you are not authenticated, or the authentication is incorrect. You must re-authenticate and try again.
403 Forbidden	Error message. You do not have permission to access this resource.
404 Not Found	Error message. The requested resource does not exist.
500 Internal Error	Error message. The server encountered an unexpected condition that prevented it from fulfilling the request.

Common API Parameters

Overview of Commonly Used API Parameters

The Historian REST service provides various REST API calls to retrieve the current tags list and query data with different sampling modes. Most of these API calls use the following common parameters:

TagNames Parameter

By default, the Historian REST service provides support to read samples for multiple tags. Multiple tag names are separated by semicolons (**;**). For example, "tagname1;tagname2;tagname3".

```
https://<historianservername>:8443/historian-rest-api
/v1/datapoints/currentvalue?tagNames=tagName1;tagname2;tagname3
```

Encode the semicolon as `%3B` if using the URI format, as the semicolon is also a valid character for a Historian name, and the web service parses the tag names incorrectly if a tag name contains a semicolon.

Start and End Timestamps Parameter

For the Start and End Timestamps parameter, the Timestamp format in the URI must be in ISO data format, such as `YYYY-MM-DDTHH:mm:ss.SSSZ`.

EPOCH time (standard base time) is only valid in the JSON-format request body or response body, such as `\/Date(928167600000-0500)\/`. If you use the same timestamp for start and end timestamps, the request returns a single result.

All timestamps passed to the REST service must be formatted as UTC timestamps.

Object Name	Description
StartTime	Start time of the query. This represents the earliest timestamp for any tag contained in the query. If no StartTime is specified, the start time is two hours prior to running the query.
EndTime	End time of the query. This represents the latest timestamp for any tag contained in the query. If no EndTime is specified, the end time is the time that the query runs.

TagSamples Parameter

The TagSamples parameter is the output from the REST API calls.

Property Name	Property Type	Description
TagName	String	Name of the tag.

Property Name	Property Type	Description
DataType	String	**Tag Data Type Value:** • **Blob** – Stores tags as binary large objects. The Blob datatype generally refers to undetermined binary data types, such as an Excel spreadsheet, a PDF file, or a Word file. • **Boolean** (one byte) – Stores boolean values. Valid values for the boolean data type are 0=FALSE and 1=TRUE. If the user sends zero, the value is taken as zero. Anything other than zero, the value is treated as one. • **Byte** (one byte) – Stores integer values. Valid values for the byte data type are -128 to +127. • **SingleFloat** (four bytes) – Stores decimal values up to six places. Valid ranges for the single float data type are 1.175494351e-38F to 3.402823466e+38F • **DoubleFloat** (eight bytes) – Stores decimal values up to 15 places. Valid values for the double float data type are 2.2250738585072014e-308 to 1.7976931348623158e+308. • **SingleInteger** (two bytes) – Stores whole numbers, without decimal places. Valid values for the single integer data type are -32767 to +32767. • **DoubleInteger** (four bytes) – Stores whole numbers, without decimal places. Valid values for the double integer data type are -2147483648 to +2147483648. • **FixedString** (Configured by user) – Stores string data of a fixed size. Valid values are between 0 and 255 bytes. • **Float** – Single float. • **Integer** – Single integer. • **MultiField** – Stores string data that has multiple words. • **QuadInteger** (eight bytes) – Stores whole numbers without decimal places. Valid values for the quad integer data type are -9,223,372,036,854,775,808 (negative nine quintillion) to +9,223,372,036,854,775,807 (positive nine quintillion). • **Scaled** (two bytes) – Lets you store a four-byte float as a twobyte integer in the Historian archive. The scaled data type saves disk space but sacrifices data precision as a result. • **Time** – Returns or sets the type of time stamping applied to data at collection time. • **UDoubleInteger (Unsigned Double Integer)** (four bytes) – Stores whole numbers without decimal places. Valid values for the unsigned double integer data type are 0 to 4,294,967, 295 (4.2 billion). • **Undefined** – Data type is not defined. • **UQuadInteger (Unsigned Quad Integer)** (eight bytes) – Stores whole numbers without decimal places. Valid values for the unsigned quad integer data type are 0 to 18,446,744,073,709,551,615 (19 quintillion). • **USingleInteger (Unsigned Single Integer)** (two bytes) – Stores whole numbers without decimal places. Valid values for the unsigned single integer data type are 0 to 65535. • **VariableString** (No fixed size) – Stores string values of undetermined size. This data type is useful if you cannot rely on a constant string length from your data source. • **Array** – Returns an array of tags from your data source. You can specify orientation, size, and number of rows returned in the array.
ErrorCode	Error Code	Error Code Definition See Error Code Definition *(page 22)* for more information.

Property Name	Property Type	Description
Samples	Data Sample	See DataSample Parameter *(page 10)* for more information.

DataSample Parameter

The DataSample Parameter specifies the number of data samples to retrieve from the archive. Samples are evenly spaced within the time range defined by start time and end time for most sampling modes.

Property Name	Property Type	Description
Value	String	Format for a multi-field tag like `{ "field1":"1","field2":"1000.0" }` (user-defined type tag). JavaScript code can parse the value string as a JSON object. All field values are string.
TimeStamp	DateTime	Start and end times of the query. If no start time is specified, the start time is two hours prior to running the query. If no EndTime is specified, the end time is the time the query runs.
Quality	Integer (Enumerated value of DataQuality.StatusType)	Data type consisting of a set of named values called elements, members or enumerators of the type. Property values reflect quality as "quality is good" or " quality is bad". **Value and Status** • **0** – Bad • **1** – Uncertain • **2** – NA • **3** – Good

SamplingModeType Parameter

The SamplingModeType parameter is the mode of sampling data from the archive. The default setting for the Sampling Mode is `Calculated`.

Properties	Description	Value
Undefined	Sampling mode is not defined.	0
CurrentValue	Retrieves the current value. The time- interval criteria are ignored.	1

Properties	Description	Value
Interpolated	Retrieves evenly-spaced, interpolated values based on interval or NumberOfSamples and the time-frame criteria.	2
Trend	Returns the raw minimum and raw maximum value for each specified interval. Use the Trend sampling mode to maximize performance when retrieving data points for plotting. For the Trend sampling mode, if the sampling interval does not evenly divide by the interval length, Historian ignores any leftover values at the end, rather than putting them into a smaller interval.	3
RawByTime	Retrieves raw archive values based on time-frame criteria.	4
RawByNumber	Retrieves raw archive values based on the StartTime criteria, the NumberOfSamples, and Direction criteria. The EndTime criteria is ignored for this sampling mode.	5
Calculated	Retrieves evenly spaced calculated values based on NumberOfSamples, interval, the time frame criteria, and the CalculationMode criteria.	6
Lab	Returns actual collected values without interpolation.	7
InterpolatedtoRaw	Provides raw data in place of interpolated data when the number of samples are fewer than the available samples.	8
TrendtoRaw	The TrendtoRaw sampling mode almost always produces the same results as the Trend sampling mode. However, when more samples are requested than there are raw data points, the TrendtoRaw sampling mode returns all available raw data points with no further processing. Use TrendtoRaw in place of Trend when this condition exists.	9
LabtoRaw	Provides raw data for the selected calculated data, when NumberOfSamples is less than the available samples.	10
RawByFilterToggle	Returns filtered time ranges using the following values: • 1 – true • 0 – false This sampling mode is used with the time range and filter tag conditions. The response string starts with a starting time stamp and ends with an ending timestamp.	11

Direction Parameter

The Direction Parameter specifies the direction (Forward or Backward from the starting time) of data sampling from the archive. The default value is Forward.

Direction	Value
Forward	0
Backward	1

CalculationModeType Parameter

The CalculationModeType parameter is only applied if the Sampling Mode is set to Calculated. It represents the type of calculation to use on the archive data. The default Calculation Mode, if none is specified, is Average.

Calculation Mode Type	Description	Value
Undefined	Calculation mode is not defined.	0
Average	Retrieves the time-weighted average for each calculation interval.	1
StandardDeviation	Retrieves the time-weighted standard deviation for each calculation interval.	2
Total	Retrieves the time-weighted rate total for each calculation interval. Use rate totals when working with a continuous measurement. Time weighting takes into account that compressed data is not evenly spaced in time. A factor must be applied to the total value to convert into appropriate engineering units. As a rate total, the default is Units/Day. If the actual units of the continuous measurement are Units/Minute, you would multiply the results by 1440 (minutes per day) to convert the total into appropriate engineering units.	3
Minimum	Retrieves the minimum value for each calculation interval.	4
Maximum	Retrieves the maximum value for each calculation interval.	5

Calculation Mode Type	Description	Value
Count	Counts the number of raw samples found with good quality in the interval. Value is the count of raw samples with good quality in the interval. The values of each sample are ignored. The Count does not include any samples of bad quality, including the start and end of collection markers. For Quality, the percentage of good samples is always 100, even if the interval does not contain any raw samples, or contains only bad quality samples. Count is useful for analyzing the distribution of the raw data samples to determine the effect of compression deadbands. It is also useful to determine which tags are consuming the most archive space.	6
RawAverage	Retrieves the arithmetic average of all good quality raw samples for each calculation interval. Value is the sum of all good quality samples in the interval, divided by the number of good quality samples in the interval. All bad quality samples are ignored. That is RawAverage is equivalent to RawTotal divided by the Count. For Quality, if there are no raw samples in the interval or if they all are bad quality, then the percentage of good is 0. Otherwise, the percentage of good is always 100, even if the interval contains bad quality samples. RawAverage is useful for calculating an accurate average when a sufficient number of raw samples are collected.	7
RawStandardDeviation	Retrieves the arithmetic standard deviation of raw values for each calculation interval. For Value, any raw point of bad data quality is ignored. For Quality, if there are no raw samples in the interval or they all have bad quality, then the percentage of good is 0. Otherwise, the percentage of good is always 100, even if the interval contains bad quality samples. RawStandardDeviation is useful for calculating an accurate standard deviation when a sufficient number of raw samples are collected.	8

Calculation Mode Type	Description	Value
RawTotal	Retrieves the arithmetic total (sum) of sampled values for each interval. Value is the sum of the good quality values of all raw samples in the interval. All bad quality samples are ignored. For Quality, the percentage of good samples is always 100, even if the interval does not contain any raw samples or it contains only bad quality samples. If the same start and end times are used, and the time span is treated as a single interval, then all values are added together. RawTotal is useful for calculating an accurate total when a sufficient number of raw samples are collected. Note that unlike ihTotal, this is a simple sum with no assumption that the values are rate values.	9
MinimumTime	Retrieves the timestamp of the minimum value found within each calculation interval. It can be a raw or an interpolated value. The minimum must be a good data quality sample.	10
MaximumTime	Retrieves the timestamp of the maximum value found within each calculation interval. It can be a raw or an interpolated value. The maximum must be a good data quality sample.	11
TimeGood	Retrieves the amount of time (milliseconds) during the interval when the data is of good quality and the filter condition is met.	12
StateCount	Retrieves the amount of time a tag uses to transition to another state from a previous state during a time interval.	13
StateTime	Retrieves the duration that a tag stayed in a given state within an interval.	14
OPCQAnd	Retrieves the OPCQAND, bit-wise AND operation of all the 16-bit OPC qualities of the raw samples stored in the specified interval. Note that OPC Quality is a subfield for Quality-Value-Timestamp (QVT), so when this calculation mode is used, OPC Quality is considered for calculation.	15

Calculation Mode Type	Description	Value
OPCQOr	Retrieves the OPCQOR, bit-wise OR operation of all the 16-bit OPC qualities of the raw samples stored in the specified interval. Note that OPC Quality is a subfield for Quality-Value-Timestamp (QVT), so when this calculation mode is used, OPC Quality is considered for calculation.	16
FirstRawValue	Retrieves the first good raw sample value for a given interval. Value is the value of the raw sample, or zero if there are no good raw samples in the interval. For Quality, if there are not good raw samples in the interval, then the percentage of good is 0. Otherwise, the percentage of good is always 100, even if the interval contains bad quality samples. Note that Quality is the same for FirstRawValue and FirstRawTime. The Raw sample has a quality of Good, Bad, or Uncertain, and that is converted to a 0 or 100 percent.	17
FirstRawTime	Retrieves the first good raw timestamp for a given interval. Value is the timestamp of the sample or the year 1969 if there are no good raw samples in the interval. For Quality, if there are not good raw samples in the interval, then the percentage of good is 0. Otherwise, the percentage of good is always 100, even if the interval contains bad quality samples. Note that Quality is the same for FirstRawValue and FirstRawTime. The Raw sample has a quality of Good, Bad, or Uncertain, and that is converted to a 0 or 100 percent.	18

Calculation Mode Type	Description	Value
LastRawValue	Retrieves the last good raw sample value for a given time interval. Value is the value of the raw sample or zero if there are no good raw samples in the interval. For Quality, if there are no good raw samples in the interval, the percentage of good samples is 0. Otherwise, the percentage of good is always 100, even if the interval contains bad samples. Note that Quality is the same for LastRawValue and LastRawTime. The Raw sample has a quality of Good, Bad, or Uncertain, and that is converted to a 0 or 100 percent.	19
LastRawTime	Retrieves the last good timestamp of the last value for a given time interval. Value is the timestamp of the sample or the year 1969 if there are no good raw samples in the interval. For Quality, if there are no good raw samples in the interval, the percentage of good samples is 0. Otherwise, the percentage of good is always 100, even if the interval contains bad samples. Note that Quality is the same for LastRawValue and LastRawTime. The Raw sample has a quality of Good, Bad, or Uncertain, and that is converted to a 0 or 100 percent.	20
TagStats	Retrieves the statistics for a tag from the archive stored in the specified interval.	21

FilterModeType Parameter

The FilterModeType parameter defines how time periods before and after transitions in the filter condition should be handled.

When the FilterModeType parameter is applied, then the Start time and End time are specified as:

- ExactTime
- BeforeTime
- AfterTime
- BeforeAndAfterTime

For example, AfterTime indicates that the filter condition should be True starting at the timestamp of the archive value that triggered the True condition, and leading up to the timestamp of the archive value that triggered the False condition.

Properties	Description	Value
ExactTime	Retrieves data for the exact times that the filter condition is True.	1
BeforeTime	Retrieves data from the timestamp of the last False filter condition to the timestamp of the next True condition.	2
AfterTime	Retrieves data from the timestamp of the True filter condition to the timestamp of the next False condition.	3
BeforeAndAfterTime	Retrieves data from the timestamp of the last False filter condition to the timestamp of the next False condition.	4

ReturnDataFields Parameter

The ReturnDataFields bitwise parameter specifies which data fields are returned in a query. Using it in a query returns data such as TimeStamp, and each field returns a Boolean value.

Each time-series data sample contains QVT (quality, value, and timestamp) values. If ReturnDataFields is not provided, then the default value of 0 is considered, and all QVT values are returned for each data sample. To return one of the data field properties, such as TimeStamp, use the TimeStamp option as shown in the table.

Properties	Description	Field value (Boolean)
All Fields	Specifies that all data fields are returned in the query.	0 (0000)
TimeStamp	The time stamp of the collected sample or an interval time stamp. When specified in the query, returns the TimeStamp property.	1 (0001)
Value	The collected value or sampled value; the data type of the value will be the same data type as the tag's raw data.	2 (0010)

Properties	Description	Field value (Boolean)
Quality	When specified in the query, returns the Quality property. Each sample in Current Value and Raw query retrieval has a quality of: • Good (3) • Not Available (2) • Uncertain (1) • Bad (0) Interpolated and Lab Retrieval express quality as "percent good".	4 (0100)

Payload Parameter

This parameter queries for the tag properties requested from the server.

Use the Payload parameter to query for all the tag properties to return from the server. In the Update Tag Configuration API, you must provide the actual tag property value. However, in the Get Tag Properties API, you must provide the property name and value of 1 (true), so the property can be read from the server and returned.

The properties listed in the following table are valid in APIs that use the Payload parameter, unless otherwise specified. For Property Names used in the Get Tag Properties API, the property name is always a Boolean (true/false) value, while it can be a string or integer for other APIs.

Property Name	Property Type	Description
AllFields	Boolean	Used for Get Tag Properties API.
Name	Boolean, String	Used for the Get Tag Properties API, Add Single Tag API, and Add Bulk Tags API.
Description	String	
EngineeringUnits	String	
Comment	String	

Property Name	Property Type	Description

DataType : ihDataType	SignedIntegral	Type definition is an enumerated type "ihDataType".
		`{`
		`ihDataTypeUndefined = 0,`
		`ihScaled,`
		`ihFloat,`
		`ihDoubleFloat,`
		`ihInteger,`
		`ihDoubleInteger,`
		`ihFixedString,`
		`ihVariableString,`
		`ihBlob,`
		`ihTime,`
		`ihInt64,`
		`ihUInt64,`
		`ihUInt32,`
		`ihUInt16,`
		`ihByte,`
		`ihBool,`
		`ihMultiField,`
		`ihArray,`
		`ihMaxDataType`
		`} ihDataType;`

FixedStringLength	UnsignedChar	
CollectorName	String	
SourceAddress	String	

CollectionType : ihCollectionType	SignedIntegral	Type definition is an enumerated type "ihCollectionType".
		`{`
		`ihUnsolicited = 1,`
		`ihPolled`
		`} ihCollectionType;`

CollectionInterval	SignedIntegral	
CollectionOffset	UnsignedLong	
LoadBalancing	Boolean	

TimeStampType : ihTimeStampType	SignedIntegral	Type definition is an enumerated type "ihTimeStampType".
		`{`
		`ihSource = 1,`
		`ihInterface,`
		`} ihTimeStampType;`

HiEngineeringUnits	Double	
LoEngineeringUnits	Double	
InputScaling	Boolean	
HiScale	Double	
LoScale	Double	
CollectorCompression	Boolean	

Property Name	Property Type	Description
CollectorDeadbandPercentRange	Float	
ArchiveCompression	Boolean	
ArchiveDeadbandPercentRange	Float	
General1	String	
General2	String	
General3	String	
General4	String	
General5	String	
ReadSecurityGroup	String	
WriteSecurityGroup	String	
AdministratorSecurityGroup	String	
LastModified	Boolean	Used for Get Tag Properties API.
LastModifiedUser	Boolean	Used for Get Tag Properties API.
InterfaceType	Boolean	Used for Get Tag Properties API.

CollectorType : ihInterfaceType	SignedIntegral	Type definition is an enumerated type "ihInterfaceType".
		`{`
		`ihInterfaceUndefined = 0,`
		`ihIFix,`
		`ihRandom,`
		`ihOPC,`
		`ihFile,`
		`ihIFixLabData,`
		`ihManualEntry,`
		`ihOther,`
		`ihCalcEngine,`
		`ihServerToServer,`
		`ihPI,`
		`ihOPCAE,`
		`ihCIMPE,`
		`ihPIDistributor,`
		`ihCIMME,`
		`ihPerfTag,`
		`ihCustom,`
		`ihServerToServerDistributor,`
		`ihWindowsPerfMon,`
		`} ihInterfaceType;`

UTCBias	SignedIntegral	
AverageCollectionTime	Boolean	Used for Get Tag Properties API.
CalculationDependencies	StringArray	
CollectionDisabled	Boolean	
ArchiveCompressionTimeout	UnsignedLong	
CollectorCompressionTimeout	UnsignedLong	

Property Name	Property Type	Description
SpikeLogic	Boolean	
SpikeLogicOverride	Boolean	
CollectorAbsoluteDeadbanding	Boolean	
CollectorAbsoluteDeadband	Double	
ArchiveAbsoluteDeadbanding	Boolean	
ArchiveAbsoluteDeadband	Double	
StepValue	Boolean	

TimeResolution : ihTimeResolution	SignedIntegral	Type definition is an enumerated type "ihTimeResolution".
		`{`
		`ihSeconds = 0,`
		`ihMilliseconds,`
		`ihMicroseconds,`
		`ihNanoseconds`
		`} ihTimeResolution;`

ConditionCollectionEnabled	Boolean	
ConditionCollectionTriggerTag	String	

ConditionCollectionComparison : ihConditionCollectionComparison	SignedIntegral	Type definition is an enumerated type "ihConditionCollectionComparison".
		`{`
		`ihConditionComparisonUndefined = 0,`
		`ihConditionComparisonEqual,`
		`ihConditionComparisonLessThan,`
		`ihConditionComparisonLessThanEqual,`
		`ihConditionComparisonGreaterThan,`
		`ihConditionComparisonGreaterThanEqual,`
		`ihConditionComparisonNotEqual`
		`} ihConditionCollectionComparison;`

ConditionCollectionCompareValue	String	
ConditionCollectionMarkers	Boolean	
Calculation	String	When the Calculation field is used, then two more conditions are required. Calculation is not a specific field for a tag property. If the tag's collector or interface type is Server-to-server and the Calculation field is set (not Null), then the field value is set to the source address.
TagId	Boolean	Used for Get Tag Properties API.
EnumeratedSetName	String	
DataStoreName	String	
DefaultQueryModifiers	Long Long	
UserDefinedTypeName	String	
NumberOfElements	SignedIntegral	

Property Name	Property Type	Description

DataDensity : ihTagDataDensity	SignedIntegral	Type definition is an enumerated type "ihTagDataDensity".
		`{`
		`ihDataDensityUndefined = 0,`
		`ihDataDensityMinimum = 1,`
		`ihDataDensityMedium = 4,`
		`ihDataDensityMaximum = 7`
		`} ihTagDataDensity;`

CalcType : ihTagCalcType	SignedIntegral	Type definition is an enumerated type "ihCalcType".
		`{`
		`ihRawTag = 0,`
		`ihAnalyticTag = 1,`
		`ihPythonExprTag = 2`
		`} ihTagCalcType;`

HasAlias	Boolean	Used for Get Tag Properties API.
IsStale	Boolean	Used for Get Tag Properties API.

Error Code Definitions

The following table provides the values and definitions for the ErrorCode parameter.

Table 1. Error Code Definitions

Error Code Value:	Error Code Definition
Success = 0	Operation successful.
Failed = -1	Operation failed.
Timeout = -2	Operation failed due to timeout.
NotConnected = -3	Not connected to Historian server.
CollectorNotFound = -4	The given collector does not exist on the server.
NotSupported = -5	Operation not supported.
DuplicateData = -6	Attempt to overwrite an existing data sample.
InvalidUsername = -7	Bad user name or password.
AccessDenied = -8	Insufficient permissions for operation.
WriteInFuture = -9	Attempted data write too far in the future.
WriteArchiveOffline = -10	Attempted data write to an offline archive.
WriteArchiveReadonly = -11	Attempted data write to a read-only archive.
WriteOutsideActiveRange = -12	Attempted data write beyond the configured active range.
WriteNoArchiveAvailable = -13	Attempted data write with no available archives.

Error Code Value:	Error Code Definition
InvalidTagname = -14	The requested tag was not found.
LicensedTagCountExceeded = -15	Number of licensed tags exceeded.
LicensedConnectionCountExceeded = -16	Number of licensed server connections exceeded.
InternalLicenseError = -17	Internal license error.
NoValue = -18	No available tag data.
DuplicateCollector = -19	The given collector name already exists on the server.
NotLicensed = -20	Server or feature is not licensed.
CircularReference = -21	Circular reference detected in calculation.
BackupInsufficientSpace = -22	Insufficient disk space to perform backup.
InvalidServerVersion = -23	Operation unsupported due to server version.
QueryResultSizeExceeded = -24	Upper limit on query results exceeded.
DeleteOutsideActiveRange = -25	Attempted data delete outside allowed modification interval.
AlarmArchiverUnavailable = -26	Alarms and Events subsystem unreachable.
ArgumentException = -27	A supplied argument is invalid.
ArgumentNullException = -28	A supplied argument is NULL.
ArgumentOutOfRangeException = -29	A supplied argument is outside the valid range.
InvalidEnumeratedSet = -30	The requested Enumerated Set was not found.
InvalidDataStore = -31	The requested data store was not found.
NotPermitted = -32	Operation not permitted.
InvalidCustomDataType = -33	The Custom data type is not supported.
ihSTATUS_EXISTING_USERDEF_REFERENCES = -34	N/A
ihSTATUS_INVALID_TAGNAME_DELETEDTAG = -35	N/A
ihSTATUS_INVALID_DHS_NODENAME = -36	N/A
ihSTATUS_DHS_SERVICE_IN_USE = -37	N/A
ihSTATUS_DHS_STORAGE_IN_USE = -38	N/A
ihSTATUS_DHS_TOO_MANY_NODES_IN_MIRROR = -39	N/A
ihSTATUS_ARCHIVE_IN_SYNC = -40	N/A
InvalidArchiveName= -41	N/A
InvalidSession = 1	Session id is invalid.
SessionExpired = 2	Session has expired.

Error Code Value:	Error Code Definition
UnknownError = 3	Unknown error, please check server log.
NoValidClientBufferManager= 4	No valid client buffer manager.
NoValueInDataSet = 5	No value in returned data set.
TagNotExisting = 6	Tag doesn't exist.
ClientBufferManagerCommunicationError = 7	Service call to central buffer server fail.
TagTypeNotSupported=8	Tag type is not supported.
ValueTypeNotMatchTagDataType = 9	Value type doesn't match tag data type.
InvalidParameter=10	Invalid query parameter.
TagSearchResultIsHuge = 11	Tag Search Criteria result was more than 5000.
InvalidHistorianServer=12	No valid server or historian server name isn't in the server list.
ihSTATUS_INVALID_INTERFACETYPE = -49	The collector type is not valid. For a list of collector types, refer to Collector Type and Subtype *(page 53)*.
ihSTATUS_INTERFACE_START_FAIL = -50	Starting the collector has failed.
ihSTATUS_INTERFACE_STOP_FAIL = -51	Stopping the collector has failed.

Historian REST APIs

Overview of the Historian REST APIs

Historian provides REST APIs to manage Historian systems, collectors, data stores, and tags. In addition, it provides APIs to install and manage collector instances.

Important: Port 8443 is used in examples and sample code. If you copy and paste the sample code from Help, you must change this port to your installed port.

Managing Systems

The Get DHS Machines API

Using the Get DHS Machines API, you can view the list of DHS machines in a location.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/dhsmachines?storageName=`

SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/dhsmachines?storageName=xx`
SAMPLE RESPONSE	`{ "ErrorCode": 0, "ErrorMessage": null, "Data": [ { "NodeName": "xyz", "IsAlreadyAdded": true } ] } }`
SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" https://<historianservername>/historian-rest-api/v1/dhsmachines?storageName=xxx`

Table 2. Query Parameters

Parameter	Description	Required?	Values
`storageName`	The value of the location whose DHS machines you want to view.	Yes	String

Table 3. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Get DHS Services API

Using the Get DHS Services API, you can view the list of DHS services in a data store.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/dhsservices?dHSServiceMask=&withReason=false`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/dhsservices?dHSServiceMask =*&withReason=false`

SAMPLE RESPONSE	see below

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Data": [
        {
            "LogicalName": "ConfigManager_NPI212611749M1",
            "NodeName": "NPI212611749M1",
            "ServiceType": 4,
            "Status": 1,
            "TCPPort": 14002
        },
        {
            "LogicalName": "DataArchiver_NPI212611749M1",
            "NodeName": "NPI212611749M1",
            "ServiceType": 2,
            "Status": 1,
            "TCPPort": 14001
        },
        {
            "LogicalName": "ClientManager_NPI212611749M1",
            "NodeName": "NPI212611749M1",
            "ServiceType": 3,
            "Status": 1,
            "TCPPort": 14000
        },
        {
            "LogicalName": "DiagnosticsManager_NPI212611749M1",
            "NodeName": "NPI212611749M1",
            "ServiceType": 5,
            "Status": 1,
            "TCPPort": 14003
        },
        {
            "LogicalName": "DataArchiver_distmachine2",
            "NodeName": "distmachine2",
            "ServiceType": 2,
            "Status": 0,
            "TCPPort": 14001
        },
        {
```

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" https://<historianservername>/historian-rest-api/v1/dhsservices?dHSServiceMask=*&withReason=false`

Table 4. Query Parameters

Parameter	Description	Required?	Values
`withReason`	Indicates whether the reason must be retrieved in the API response.	Yes	Boolean
`dHSServiceMask`	The value of the DHS service mask.	Yes	String

Table 5. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Get Server Properties API

Using the Get Server Properties API, you can view the list of properties of a server.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/serverproperties`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/serverproperties`

SAMPLE RESPONSE	(see code below)

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Data": {
        "Storages": [
            {
                "StorageName": "System
 Storage",
                "StorageType": 2,
                "NumberOfDataStores": 1,
                "NumberOfArchivers": 0,
                "DataStores": [
                    "System"
                ],
                "Id":
 "861C2743-72E0-46FC-9B31-90E28CC39B8D",
                "IsDefault": false,
                "LastModifiedUser": null,
                "LastModifiedTime":
 "1970-01-01T00:00:00.000Z",
                "ArchiverServices": []
            },
            {
                "StorageName": "xyz",
                "StorageType": 0,
                "NumberOfDataStores": 3,
                "NumberOfArchivers": 1,
                "DataStores": [
                    "ScadaBuffer",
                    "DHSSystem",
                    "User"
                ],
                "Id":
 "5F267DF3-879A-4222-8A0E-D31EDEA83C14",
                "IsDefault": true,
                "LastModifiedUser": null,
                "LastModifiedTime":
 "1970-01-01T00:00:00.000Z",
                "ArchiverServices": [
                    {
                        "LogicalName":
 "DataArchiver_xyz",
                        "NodeName": "xyz",
                        "ServiceType": 2,
                        "IsAlreadyAdded":
 true,
```

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" https://<historianservername>/historian-rest-api/v1/serverproperties`

Table 6. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Get System Statistics API

Using the Get System Statistics API, you can view the statistics of a system.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/systemstats`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/systemstats`

SAMPLE RESPONSE

```json
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Data": {
        "Utilization": {
            "WriteCacheHitRatio": "0.499",
            "SpaceConsumptionRate": "",
            "CompressionRatio": "0.199",
            "ReadQueueSize": "0",
            "WriteQueueSize": "0",
            "MsgQueueSize": "0",
            "ReadQueueProcessRate": "3",
            "WriteQueueProcessRate": "0",
            "MsgQueueProcessRate": "0",
            "MemoryVMUsage": "62",
            "OutOfOrderRate": "0",
            "ReadThreadUsage": "0",
            "WriteThreadUsage": "0",
            "FailedWriteRate": "0",
            "DiskFreeSpace": "59828"
        },
        "AlarmEvents": {
            "AverageAlarmRate": ""
        },
        "TotalCollectors": {
            "TotalCollectors": 1,
            "RunningCollectors": 1,
            "StoppedCollectors": 0,
            "UnknownCollectors": 0
        },
        "Licence": {
            "ActualDataStores": 3,
            "MaxDataStores": 200,
            "ActualTags": 0,
            "MaxTags": 2147483647,
            "ActualUsers": 0,
            "MaxUsers": 1000
        }
    }
}
```

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" https://<historianservername>/historian-rest-api/systemstats`

Table 7. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Get Read Sample and Receive Rate API

Using the Get Read Sample and Receive Rate API, you can view the read rate and receive rate of a system.

METHOD	GET

URI	Read Sample Rate
	`https://<historianservername>/historian-rest-api/v1/performancecounter/perftagdata/PerfTag_AverageEventRate/-/-/starttime/endtime/interval`
	Receive Rate
	`https://<historianservername>/historian-rest-api/v1/performancecounter/perftagdata/PerfTag_AverageReadRawRate/-/-/starttime/endtime/interval`

SAMPLE GET URI	`https://<historianservername>/historian-rest-api/v1/performancecounter/perftagdata/PerfTag_AverageEventRate/-/-/2020-12-15T11:19:01.719Z/2020-12-15T12:19:`

SAMPLE RESPONSE	
	```
	{
	    "ErrorCode": 0,
	    "ErrorMessage": null,
	    "Data": [
	        {
	            "TagName": "PerfTag_AverageEventRate",
	            "ErrorCode": 0,
	            "DataType": "DoubleFloat",
	            "Samples": [
	                {
	                    "TimeStamp": "2020-11-18T05:35:22.612Z",
	                    "Value": "0",
	                    "Quality": 0
	                },
	                {
	                    "TimeStamp": "2020-11-18T05:47:22.612Z",
	                    "Value": "0",
	                    "Quality": 0
	                },
	                {
	                    "TimeStamp": "2020-11-18T05:53:22.612Z",
	                    "Value": "0",
	                    "Quality": 0
	                },
	                {
	                    "TimeStamp": "2020-11-18T06:11:22.612Z",
	                    "Value": "0",
	                    "Quality": 0
	                },
	                {
	                    "TimeStamp": "2020-11-18T06:29:22.612Z",
	                    "Value": "0",
	                    "Quality": 0
	                }
	            ]
	        }
	    ]
	}
	```

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” https://<historianservername>/historian-rest-api/v1/performancecounter/perftagdata/PerfTag_AverageEventRate/-/-/2020-12-15T11:19:01.719Z/2020-12-15T12:19:`

Table 8. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Get Storages API

Using the Get Storages API, you can view the list of locations in a system.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/storages?storageMask=`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/storages?storageMask=*`

SAMPLE RESPONSE

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Data": [
        {
            "StorageName": "System Storage",
            "StorageType": 2,
            "NumberOfDataStores": 1,
            "NumberOfArchivers": 0,
            "DataStores": [
                "System"
            ],
            "Id": "861C2743-72E0-46FC-9B31-90E28CC39B8D",
            "IsDefault": false,
            "LastModifiedUser": null,
            "LastModifiedTime": "1970-01-01T00:00:00.000Z",
            "ArchiverServices": []
        },
        {
            "StorageName": "srinivaswin10",
            "StorageType": 0,
            "NumberOfDataStores": 3,
            "NumberOfArchivers": 1,
            "DataStores": [
                "ScadaBuffer",
                "DHSSystem",
                "User"
            ],
            "Id": "5F267DF3-879A-4222-8A0E-D31EDEA83C14",
            "IsDefault": true,
            "LastModifiedUser": null,
            "LastModifiedTime": "1970-01-01T00:00:00.000Z",
            "ArchiverServices": [
                {
                    "LogicalName": "DataArchiver_xyz",
                    "NodeName": "xyz",
                    "ServiceType": 2,
                    "TCPPort": 14001
                }
            ]
```

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” https://<historianservername>/historian-rest-api/v1/storages?storageMask=*`

Table 9. Query Parameters

Parameter	Description	Required?	Values
`storageMask`	The value of the location mask.	No	String

Table 10. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Add Machine API

Using the Add Machine API, you can add a server in a Historian system.

METHOD	POST
URI	`https://<historianservername>/historian-rest-api/v1/machine`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/machine`
	
	`Payload`
	
	`{`
	`"nodeName": "node1"`
	`}`

SAMPLE RESPONSE	`{`
	
	`    "ErrorCode": 0,`
	
	`    "ErrorMessage": null`
	
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>”`
	`-d “{ \”nodeName \”:\”name\”}” -X POST https://<historianservername>/historian-rest-api/v1/machine`

Table 11. Query Parameters

Parameter	Description	Required?	Values
`Payload`	Contains the machine name of the server that you want to add.	Yes	Multiple

Table 12. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Delete Machine API

Using the Delete Machine API, you can remove a server from a Historian system.

METHOD	DELETE
URI	`https://<historianservername>/historian-rest-api/v1/machine`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/machine`
	
	`Payload`
	
	`{`
	`"nodeName": "",`
	
	`}`

SAMPLE RESPONSE	`{`
	
	`    "ErrorCode": 0,`
	
	`    "ErrorMessage": null`
	
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \”nodeName \”:\”name\”}” -X DELETE https://<historianservername>/historian-rest-api/v1/machine`

Table 13. Query Parameters

Parameter	Description	Required?	Values
`Payload`	Contains the name of the machine that you want to remove.	Yes	Multiple

Table 14. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Create Mirror Group API

Using the Create Mirror Group API, you can create a mirror group.

METHOD	POST
URI	`https://<historianservername>/historian-rest-api/v1/mirrorgroup`

SAMPLE URI	
	```
	https://<historianservername>/historian-rest-api/v1/mirrorgroup

	Payload

	{
	"mirrorStorageName": "storagename",

	"nodes": "node1;node2"

	}
	```

SAMPLE RESPONSE	
	```
	{

	    "ErrorCode": 0,

	    "ErrorMessage": null

	}
	```

SAMPLE cURL COMMAND	```curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -d "{ \" mirrorStorageName \":\"name\",\" nodes \": \"xx;yy\"}" -X POST https://<historianservername>/historian-rest-api/v1/mirrorgroup```

Table 15. Query Parameters

Parameter	Description	Required?	Values
`Payload`	Contains the mirror group name and the servers you want to add to the group.	Yes	Multiple

Table 16. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Add Mirror Machine API

Using the Add Mirror Machine API, you can add a server to a mirror group.

METHOD	POST

URI	`https://<historianservername>/historian-rest-api/v1/mirrormachine`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/mirrormachine`
	`Payload`
	`{`
	`"mirrorStorageName": "Mirror2",`
	`"mirrorMachineName": "distmachine1"`
	`}`

SAMPLE RESPONSE	`{`
	`"ErrorCode": 0,`
	`"ErrorMessage": null`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -d “{ \” mirrorStorageName \”:\”name\”,\" nodes \": \"xx;yy\"}” -X POST https://<historianservername>/historian-rest-api/v1/mirrormachine`

Table 17. Query Parameters

Parameter	Description	Required?	Values
`Payload`	Contains the machine name of the server that you want to add.	Yes	Multiple

Table 18. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Mirror Group Update API

Using the Mirror Group Update API, you can update the name of a mirror group.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/mirrorgroup`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/mirrorgroup`
	
	`Payload`
	
	`{`
	`"mirrorStorageName": "Mirror2",`
	
	`"mirrorStorageNewName": "Mirror3"`
	
	`}`

SAMPLE RESPONSE	`{`
	
	`    "ErrorCode": 0,`
	
	`    "ErrorMessage": null`
	
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -d "{ \"mirrorStorageName \":\"name\", \" mirrorStorageNewName \": \"sname\"}" -X PUT https://<historianservername>/historian-rest-api/v1/mirrorgroup`

Table 19. Query Parameters

Parameter	Description	Required?	Values
`Payload`	Contains the existing and new names of the mirror group that you want to rename.	Yes	Multiple

Table 20. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Delete Mirror Machine API

Using the Delete Mirror Machine API, you can remove a server from a mirror group.

METHOD	DELETE
URI	`https://<historianservername>/historian-rest-api/v1/mirrormachine`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/mirrormachine`
	
	`Payload`
	
	`{`
	
	`"mirrorStorageName": "Mirror 2",`
	
	`"mirrorMachineName": "distmachine1"`
	`}`

SAMPLE RESPONSE	`{`
	
	`    "ErrorCode": 0,`
	
	`    "ErrorMessage": null`
	
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -d "{ \"mirrorStorageName \":\"name\", \" mirrorMachineName\":\"name\"}" -X DELETE https://<historianservername>/historian-rest-api/v1/mirrormachine`

Table 21. Query Parameters

Parameter	Description	Required?	Values
`Payload`	Contains the name of the machine that you want to remove and the name of the mirror group.	Yes	Multiple

Table 22. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Delete Mirror Group API

Using the Delete Mirror Group API, you can delete a mirror group.

METHOD	DELETE
URI	`https://<historianservername>/historian-rest-api/v1/mirrorgroup`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/mirrorgroup`
	
	`Payload`
	
	`{`
	`"mirrorStorageName": "Mirror3",`
	`}`

SAMPLE RESPONSE	`{`
	
	`    "ErrorCode": 0,`
	
	`    "ErrorMessage": null`
	
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -d "{ \"mirrorStorageName \":\"name\"}" -X DELETE https://<historianservername>/historian-rest-api/v1/mirrorgroup`

Table 23. Query Parameters

Parameter	Description	Required?	Values
`Payload`	Contains the name of the machine that you want to remove.	Yes	Multiple

Table 24. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Local OPC Servers API

Using the Local OPC Servers API, you can view the list of OPC servers installed on a specified machine.

METHOD	GET
URI	`http://<historianservername>/v1/localopcservers/<machine name>`
SAMPLE QUERY PARAM GET URL	`http://<historianservername>/v1/localopcservers/<machine name>`

SAMPLE RESPONSE	`{ "ErrorCode": 0, "ErrorMessage": null, "ServerIDs": [ "ID1", "ID2 " ] }`
SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" https://<historianservername>/historian-rest-api/v1/localopcservers/xyz`

Table 25. Query Parameters

Parameter	Description	Required?	Values
`machine name`	The machine name of the OPC server.	Yes	String

Table 26. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Local OPC AE Servers API

Using the Local OPC AE Servers API, you can view the list of OPC Alarms and Events servers installed on a specified machine.

METHOD	GET
URI	`http://<historianservername>/v1/localopcaeservers/<machine name>`
SAMPLE QUERY PARAM GET URL	`http://<historianservername>/v1/localopcaeservers/<machine name>`
SAMPLE RESPONSE	`{ "ErrorCode": 0, "ErrorMessage": null, "ServerIDs": [ "ID1", "ID2 " ] }`
SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" https://<historianservername>/historian-rest-api/v1/localopcaeservers/abc`

Table 27. Query Parameters

Parameter	Description	Required?	Values
`machine name`	The machine name of the OPC Alarms and Events server.	Yes	String

Table 28. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Local OPC HDA Servers API

Using the Local OPC HDA Servers API, you can view the list of OPC HDA servers installed on a specified machine.

METHOD	GET
URI	`http://<historianservername>/v1/localopchdaservers/<machine name>`
SAMPLE QUERY PARAM GET URL	`http://<historianservername>/v1/localopchdaservers/<machine name>`
SAMPLE RESPONSE	`{ "ErrorCode": 0, "ErrorMessage": null, "ServerIDs": [ "ID1", "ID2 " ] }`
SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” https://<historianservername>/historian-rest-api/v1/localopchdaservers/xyz`

Table 29. Query Parameters

Parameter	Description	Required?	Values
`machine name`	The machine name of the OPC HDA server.	Yes	String

Table 30. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The DHS Service Port Update API

Using the DHS Service Port Update API, you can change the port and other details of a DHS service.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/dhsservice/<DHS service name>`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/dhsservice/ DataArchiver_xxx`
	`{`
	`   "LogicalName": "DataArchiver_xxx",`
	`   "NodeName": "xxx",`
	`   "ServiceType": 2,`
	`   "Status": 1,`
	`   "TCPPort": 14005`
	`}`

SAMPLE RESPONSE	`{`
	`    "ErrorCode": 0,`
	`    "ErrorMessage": null,`
	`    "Data": {`
	`        "LogicalName": "DataArchiver_xxx",`
	`        "NodeName": "xxx",`
	`        "ServiceType": 2,`
	`        "Status": 1,`
	`        "TCPPort": 14005`
	`    }`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \” LogicalName \”:\” DataArchiver_xxx \”, \" NodeName \": \"xxx\"\”,\" ServiceType \": 2,\" Status\": 1, \" TCPPort \": 14005}” -X PUT https://<historianservername>/historian-rest-api/v1/dhsservice/DataArchiver_xxx`

Table 31. Query Parameters

Parameter	Description	Required?	Values
`Payload`	Contains the values of the attributes of the data store that you want to change.	Yes	Multiple

Table 32. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

Managing Collector Instances

The Create Collector Instance API

Using the Create Collector Instance API, you can create a collector instance.

METHOD	POST
URI	`https://<historianservername>/historian-rest-api/v1/collector/createnewinstance`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/collector/createnewinstance`
	Payload
	`{`
	`"mode":1,`
	`"CollectorSystemName":"xyz",`
	`"InterfaceDescription":"xyz_Simulation_<IP address>_2",`
	`"DataPathDirectory":"C:\\Proficy Historian Data",`
	`"CollectorDestination":"Historian",`
	`"winUserName":"","winPassword":"",`
	`"InterfaceSubType":"",`
	`"DestinationHistorianUserName":"abc",`
	`"DestinationHistorianPassword":"password",`
	`"DestinationHistorian":"<IP address>",`
	`"General1":"",`
	`"General2":"",`
	`"General3":"123",`
	`"General4":"",`
	`"General5":"",`
	`"Type":2,`
	`"InterfaceName":"`*`<source server>_<type of the collector>_<destination server>`*`"`
	`}`
	Note:
	• The DestinationHistorian parameter will not have a value for offline collector configuration.
	• To connect to MQTT destinations such as AWS IoT and Google Cloud Platform (GCP), you must provide an encrypted password.

SAMPLE RESPONSE	`{`
	`    "ErrorCode": 0,`
	`    "ErrorMessage": null`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \"mode\":1,\"CollectorSystemName\": \"xyz\", \"InterfaceDescription\": \"xyz_Simulation_<IP address>_2\", \"DataPathDirectory\":\"C:\\Proficy Historian Data\", \"CollectorDestination\":\"Historian\", \"winUserName\":\"\",\"winPassword\":\"\", \"InterfaceSubType\":\"\", \"DestinationHistorianUserName\":\"abc\", \"DestinationHistorianPassword\":\"password \", \"DestinationHistorian\":\"<IP address>\", \"General1\":\"\", \"General2\":\"\",\"General3\":\"xyz\", \"General4\":\"\",\"General5\":\"\", \"Type\":2,\"InterfaceName\": \"<source server>_<type of the collector>_<destination server>\"}” -X POST https://<historianservername>/historian-rest-api/v1/collector/createnewinstance`

Table 33. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Get Collector Instance Details API

Using the Get Collector Instance Details API, you can view the details of a collector instance.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/collector/instancedetails/<interface name>`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/collector/instancedetails/<interface name>`

SAMPLE RESPONSE	`{`
	`"ErrorCode":0,`
	`"ErrorMessage":null,`
	`"Data":`
	`{`
	`"CloudDestination":"",`
	`"InterfaceSubType":"",`
	`"CollectorSystemName":"xyz",`
	`"Type":2,`
	`"DefaultCompression":false,`
	`"CloudInformationLogLevel":0,`
	`"InterfaceDataDir":"C:\\Proficy Historian Data",`
	`"SourceServer":"",`
	`"Username":"",`
	`"Password":"",`
	`"DestinationType":"Historian",`
	`"DestinationServer":"abc",`
	`"DebugLogLevel":0,`
	`"InterfaceInstallDrive":"C",`
	`"ConnnectionString":"xyz"`
	`}`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" https://<historianservername>/historian-rest-api/v1/collector/instancedetails/xyz_Simulation_<IP address>_2`

Table 34. Query Parameters

Parameter	Description	Required?	Values
`interface name`	The interface name of the collector whose details you want to view.	Yes	String

Table 35. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Edit Collector Instance API

Using the Edit Collector Instance API, you can modify the cloud parameters of a collector instance. The collector instance will be restarted after you make changes.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/collector/editinstance`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/collector/editinstance`
	Payload
	`{"interfaceName":"<source server>_<type of the collector>_<destination server>", "messageCompression":0, "azureLogLevel":1, "debugMode":0, "CollectorDestination":"Predix", "DestinationHistorian":"abc", "mode":1, "CloudDestinationAddress":"wss://def.run.abc.ice.predix.io/v1/stream/messages", "IdentityIssuer":"https://1234.predix-uaa.run.abc.ice.predix.io/oauth/token", "ClientID":"xyz", "ClientSecret":"123", "ZoneID":"1234", "Proxy":"http://1.2.3.4:80", "ProxyUserName":"", "ProxyPassword":"", "DatapointAttribute1":"", "DatapointAttribute2":"", "DatapointAttribute3":"", "DatapointAttribute4":""}`
	Note:
	• The DestinationHistorian parameter will not have a value for offline collector configuration.
	• To connect to MQTT destinations such as AWS IoT and Google Cloud Platform (GCP), you must provide an encrypted password.

SAMPLE RESPONSE	`{ "ErrorCode": 0, "ErrorMessage": null }`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{\"interfaceName\": \"<source server>_<type of the collector>_<destination server>\", \"InterfaceName\":\"<source server>_<type of the collector>_<destination server>\", \"messageCompression\":0,\"azureLogLevel \":1, \"debugMode\":0,\"CollectorDestination\": \"Predix\",\"DestinationHistorian\":\"abc \",\"mode\":1, \"CloudDestinationAddress\":\"wss://wss:// def.run.abc.ice.predix.io/v1/stream/ messages\", \"IdentityIssuer\":\"https://1234.predix- uaa.run.abc.ice.predix.io/oauth/token/", \"ClientID\":\"HistQA\",\"ClientSecret\": \"Gei321itc\",\"ZoneID\":\"1234\", \"Proxy\":\"http://1.2.3.4:80\", \"ProxyUserName\":\"\",\"ProxyPassword\": \"\", \"DatapointAttribute1\":\"\", \"DatapointAttribute2\":\"\", \"DatapointAttribute3\":\"\", \"DatapointAttribute4\":\"\"}” -X PUT https://<historianservername>/historian- rest-api/v1/collector/editinstance`

Table 36. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Azure Log Level API

Using the Azure Log Level API, you can set the debug information log level for destination - Azure IoT Hub. You can set a value ranging from 0 to 4.

METHOD	PUT
URI	`https://<historianservername>/historian- rest-api/v1/collector/azureloglevel`

SAMPLE URI	`https://<historianservername>/historian- rest-api/v1/collector/azureloglevel`
	Payload
	`{"interfaceName":""InterfaceName":"<source server>_<type of the collector>_<destination server>"", "azureLogLevel":1,}`

SAMPLE RESPONSE	`{`
	`    "ErrorCode": 0,`
	`    "ErrorMessage": null`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -d “{\"interfaceName\":\"<source server>_<type of the collector>_<destination server>\", \"azureLogLevel\":1}” -X PUT https://<historianservername>/historian-rest-api/v1/collector/azureloglevel`

Table 37. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Install Component Details API

Using the Install Component Details API, you can view the install component details from the collector machine.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/installcomponentdetails/collectorType/collectorSubType/machine`
SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/installcomponentdetails/2/-/abc`

SAMPLE RESPONSE	`{`
	`"ErrorCode":0,`
	`"ErrorMessage":null,`
	`"Data":`
	`{`
	`"InterfaceInstallDrive":"C",`
	`"InterfaceDataDir":"C:\\Proficy Historian Data",`
	`"CertPathDir":"NONE"`
	`}`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” https://<historianservername>/historian-rest-api/v1/installcomponentdetails/2/-/abc`

Table 38. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Message Compression API

Using the Message Compression API, you can enable or disable message compression.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/collector/messagecompression`
SAMPLE REQUEST	`{ "interfaceName":"<source server>_<type of the collector>_<destination server>", "messageCompression":1 }`
SAMPLE RESPONSE	`{ "ErrorCode":0, "ErrorMessage":null, }`
SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{\"interfaceName\":\"<source server>_<type of the collector>_<destination server>\", \"messageCompression\":1}” -X PUT https://<historianservername>/historian-rest-api/v1/collector/messagecompression`

Table 39. Query Parameters

Parameter	Description	Required?	Values
`interface name`	The interface name of the collector whose message compression you want to enable or disable.	Yes	String
`messagecompression`	Identifies whether you want to enable or disable message compression. The valid values are 0 and 1.	Yes	

Table 40. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Delete Instance API

Using the Delete Instance API, you can delete a collector instance.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/collector/deleteinstance`

SAMPLE REQUEST	`https://<historianservername>/historian-rest-api/v1/collector/deleteinstance`
	`Payload`
	`{`
	`"interfaceName":"<source server>_<type of the collector>_<destination server>",`
	`"deleteTags":true`
	`}`

SAMPLE RESPONSE	`{`
	`"ErrorCode":0,`
	`"ErrorMessage":null,`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -d "{\"interfaceName\": \"<source server>_<type of the collector>_<destination server>\", \"deleteTags\":true}" -X PUT https://<historianservername>/historian-rest-api/v1/collector/deleteinstance`

Table 41. Query Parameters

Parameter	Description	Required?	Values
`interface name`	The interface name of the collector whose details you want to delete.	Yes	String
`deleteTags`	Identifies whether you want to delete the tags. The valid values are true and false.	Yes	Boolean

Table 42. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

Collector Type and Subtype

The following table provides a list of collector type and subtype for each collector, which you will provide in APIs.

Collector	Collector Type	Collector Subtype
The Calculation collector	8	
The Cygnet collector	16	Cygnet
The File collector	4	
The iFIX Alarms and Events collector	11	iFixAE
The iFIX collector	1	
The MQTT collector	16	MQTT
The ODBC collector	16	ODBC
The OPC Classic Alarms and Events collector	11	
The OPC Classic DA collector	3	
The OPC Classic HDA collector	16	OPCHDA
The OPC UA DA collector	16	OPCUA
The OSI PI collector	10	
The OSI PI distributor	13	
The Server-to-Server collector	9	
The Server-to-Server distributor	17	
The Simulation collector	2	
The Windows Performance collector	18	
The Wonderware collector	16	Wonderware

Managing Collectors ApIs

The Start Collector API

Using the Start Collector API, you can start a collector.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/collector/start`

<table>
<tr><td>SAMPLE URI</td><td>Sample URI for the service mode:
<pre>
https://<historianservername>/historian-
rest-api/v1/collector/start

Payload

{
  "interfaceName":"<i><source server>_<type of
 the collector>_<destination server></i>",
  "mode":1
}
</pre>
Sample URI for the command line mode:
<pre>
https://<historianservername>/historian-
rest-api/v1/collector/start

Payload

{
  "interfaceName":"<i><source server>_<type of
 the collector>_<destination server></i>",
  "mode":2,
  “winUserName”:””,
  “winPassword”:””

}
</pre></td></tr>
<tr><td>SAMPLE RESPONSE</td><td><pre>
{
    "ErrorCode": 0,
    "ErrorMessage": ""
    "Data": "Collector Start Initiated"
}
</pre></td></tr>
<tr><td>SAMPLE cURL COMMAND</td><td><pre>
curl -i -H "Accept: application/json" -i -H
 "Content-Type: application/json"
-H "Authorization: Bearer <TOKEN>” -d
“{ \”interfaceName\”:\”<i><source
 server>_<type of the
 collector>_<destination server></i>\”,
\"mode\": 1}” -X PUT https://
<historianservername>/historian-rest-api/
v1/collector/start
</pre></td></tr>
</table>

Query parameters include the Payload parameter, which is a JSON file, which contains the following properties.

Table 43. Query Parameters

Parameter	Description	Required?	Values
`interfaceName`	The interface name of the collector.	Yes	String
`mode`	The mode to use to manage the collector.	Yes	• 1: service mode • 2: command-line mode
`winUserName`	The Windows username.	Yes (only if you want to use the command-line mode)	String

Parameter	Description	Required?	Values
`winPassword`	The Windows password	Yes (only if you want to use the command-line mode)	String

Table 44. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.
`Data`	String	No	Indicates if the task has been initiated.

The Stop Collector API

Using the Stop Collector API, you can stop a collector.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/collector/stop`

SAMPLE URI	
	```
	https://<historianservername>/historian-
	rest-api/v1/collector/stop

	Payload

	{
	“interfaceName":"<source server>_<type of
	 the collector>_<destination server>"
	“winUserName”:”TestAdmin”,
	“winPassword”:”TestAdminPassword”

	}
	```

SAMPLE RESPONSE	
	```
	{
	    "ErrorCode": 0,
	    "ErrorMessage": null,
	    "Data": "Collector Stop Initiated"
	}
	```

SAMPLE cURL COMMAND	
	```
	curl -i -H "Accept: application/json" -i -H
	 "Content-Type: application/json"
	-H "Authorization: Bearer <TOKEN>” -d
	“{ \”interfaceName\”:\”<source
	 server>_<type of the
	 collector>_<destination server>\”}”
	-X PUT https://<historianservername>/
	historian-rest-api/v1/collector/stop
	```

Query parameters include the Payload parameter, which is a JSON file, which contains the following properties.

Table 45. Query Parameters

Parameter	Description	Required?	Values
`interfaceName`	The interface name of the collector.	Yes	String
`winUserName`	The Windows username.	Yes (only if you want to use the command-line mode)	String
`winPassword`	The Windows password	Yes (only if you want to use the command-line mode)	String

Table 46. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.
`Data`	String	No	Indicates if the task has been initiated.

The Restart Collector API

Using the Restart Collector API, you can restart a collector.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/collector/restart`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/collector/restart`
	
	`Payload`
	`{`
	`  “interfaceName":"<source server>_<type of the collector>_<destination server>"`
	`   "winUserName":"",`
	`   "winPassword":""`
	`}`

SAMPLE RESPONSE	`{`
	`    "ErrorCode": 0,`
	`    "ErrorMessage": null,`
	`    "Data": "Collector Restart Initiated"`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -d "{ \"interfaceName\":\"<source server>_<type of the collector>_<destination server>\"}" -X PUT https://<historianservername>/historian-rest-api/v1/collector/restart`

Query parameters include the Payload parameter, which is a JSON file, which contains the following properties.

Table 47. Query Parameters

Parameter	Description	Required?	Values
`interfaceName`	The interface name of the collector.	Yes	String
`winUserName`	The Windows username.	Yes	String
`winPassword`	The Windows password	Yes	String

Table 48. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.
`Data`	String	No	Indicates if the task has been initiated.

The Pause Data Collection API

Using the Pause Data Collection API, you can pause the data collection of a collector.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/collector/pausecollection/{interfaceName}`
SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/collector/pausecollection/RSSERVER2012-02_Simulation`
SAMPLE RESPONSE	`{ "ErrorCode": 0, "ErrorMessage": "" }`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -X PUT https://<historianservername>/historian-rest-api/v1/collector/pausecollection/RSSERVER2012-02_Simulation`

Table 49. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.
`Data`	String	No	Indicates if the task has been initiated.

The Resume Data Collection API

Using the Resume Data Collection API, you can resume the data collection of a collector.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/collector/resumecollection/{interfaceName}`
SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/collector/resumecollection/RSSERVER2012-02_Simulation`
SAMPLE RESPONSE	`{ "ErrorCode": 0, "ErrorMessage": "" }`
SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -X PUT https://<historianservername>/historian-rest-api/v1/collector/resumecollection/RSSERVER2012-02_Simulation`

Table 50. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.
`Data`	String	No	Indicates if the task has been initiated.

The Add Tag Comment API

Using the Add Tag Comment API, you can add a comment to a tag.

METHOD	POST
URI	`https://<historianservername>/historian-rest-api/v1/tags/addcomment`

SAMPLE URI	
	```
	https://<historianservername>/historian-
	rest-api/v1/tags/addcomment

	Payload
	{

	 "tagName":"rsserver2012-02.Simulation00003",
	  "comment":"Retest",
	  "timeStamp":"2020-04-22T00:00:00.000Z"
	}
	```

SAMPLE RESPONSE	
	```
	{
	               "ErrorCode": 0,
	               "ErrorMessage": null,
	               "Data": ""
	              }
	```

SAMPLE cURL COMMAND	
	```
	curl -i -H "Accept: application/json"
	 -i -H "Content-Type: application/
	json" -H "Authorization: Bearer
	 <TOKEN>" -d "{ \"tagName\":\"
	 rsserver2012-02.Simulation00003\",
	\"comment\":\"Test10\",\"timeStamp\":
	 \"2020-04-22T00:00:00.000Z \"}" -X POST
	 https://<historianservername>:8443/
	historian-rest-api/v1/tags/addcomment
	```

Query parameters include the Payload parameter, which is a JSON file, which contains the following properties.

Table 51. Query Parameters

Parameter	Description	Required?	Values
`tagName`	The name of the tag.	Yes	String
`timestamp`	The timestamp of the comment.	Yes	String
`comment`	The comment.	Yes	String

Table 52. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Get Tag Comment API

Using the Get Tag Comment API, you can view the comments added to a tag.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/tags/comments/{tagNames}/{start}/{end}`
SAMPLE QUERY PARAM GET URI	`https://<historianservername>/historian-rest-api/v1/tags/comments/?tagNames=rsserver2012-02.Simulation00003;rsserver2012-02.Simulation00004&start=2020-04-19T00:00:00.000Z&end=2020-04-24T00:00:00.000Z` **Note:** The query parameter supports multiple tags.
SAMPLE RESPONSE	see below

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Data": [
        {
            "TagName": "Motor Temperature",
            "ErrorCode": 0,
            "Comments": [
                {
                    "TimeStamp":
 "2020-04-22T00:00:00.000Z",
                    "Comment": "Heat run
 test: Starting temperature"
                },
                {
                    "TimeStamp":
 "2020-04-22T00:00:00.000Z",
                    "Comment": "Heat run
 test: Temperature of the stator"
                },
                {
                    "TimeStamp":
 "2020-04-22T00:00:00.000Z",
                    "Comment": "Heat run
 test: Temperature of the rotor"
                },
                {
                    "TimeStamp":
 "2020-04-22T00:00:00.000Z",
                    "Comment": "Heat run
 test: Temperature of the shaft"
                },
                {
                    "TimeStamp":
 "2020-04-22T00:00:00.000Z",
                    "Comment": "Heat run
 test: Temperature of the endshield"
                }
            ]
        }
    ]
}
```

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” http://<historianservername>/historian-rest-api/v1/tags/comments/<tagNames>/<start>/<end>`

Query parameters include the Payload parameter, which is a JSON file, which contains the following properties.

Table 53. Query Parameters

Parameter	Description	Required?	Values
`tagNames`	The names of the tag as a semicolon-separated list. For example: HISTWIN20161.ctag1; HISTWIN20161.ctag2	Yes	String
`start`	The start time of the query, in ISO data format (YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime
`end`	The end time of the query, in ISO format (YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime

Table 54. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Set Debug Mode API

Using the Set Debug Mode API, you can set the debug mode of a collector.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/collector/setdebugmode`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/collector/setdebugmode`
	
	`Payload`
	`{`
	`  “interfaceName":"<source server>_<type of the collector>_<destination server>",`
	`  “debugMode”:255`
	`}`

SAMPLE RESPONSE	`{ "ErrorCode": 0, "ErrorMessage": "" }`
SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -d "{ \"interfaceName\":\"<source server>_<type of the collector>_<destination server>\", \"debugMode\"}" -X PUT https://<historianservername>/historian-rest-api/v1/collector/setdebugmode`

Query parameters include the Payload parameter, which is a JSON file, which contains the following properties.

Table 55. Query Parameters

Parameter	Description	Required?	Values
`interfaceName`	The interface name of the collector.	Yes	String
`debugMode`	The debug log level for the collector.	Yes	• 0: Normal log level • 255: Debug log level

Table 56. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Buffer File Control API

Using the Buffer File Control API, you can delete or move the buffer files. It is recommended to move the buffer files to a new folder within the same drive.

Note: Moving files to a network shared drive is not supported.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/collector/buffercontrol`

SAMPLE URI	Sample URI for moving buffer files:
	`https://<historianservername>/historian-rest-api/v1/collector/buffercontrol`
	
	`Payload`
	`{`
	`  “interfaceName":"<source server>_<type of the collector>_<destination server>",`
	`  “bufferMode”:2,`
	`     "winUserName":"Administrator",`
	`        "winPassword":"xxxxxxx",`
	`  "bufferPath": "C:\\Users\\bufffiles"`
	`}`
	Sample URI for deleting buffer files:
	`https://<historianservername>/historian-rest-api/v1/collector/buffercontrol`
	
	`Payload`
	`{`
	`  “interfaceName":"<source server>_<type of the collector>_<destination server>",`
	`  “bufferMode”:1`
	` "winUserName":"Administrator",`
	`        "winPassword":"xxxxxxx",`
	`}`

SAMPLE RESPONSE	Sample response for moving buffer files:
	`{`
	`    "ErrorCode": 0,`
	`    "ErrorMessage": null,`
	`    "Data": "BufferFiles Move Initiated. Collector is in the Stopped state."`
	`}`
	Sample response for deleting buffer files:
	`{`
	`    "ErrorCode": 0,`
	`    "ErrorMessage": null,`
	`    "Data": "BufferFiles Delete Initiated. Collector is in the Stopped state."`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \”interfaceName\”:\”<source server>_<type of the collector>_<destination server>\”, \”bufferMode \”:1,\”bufferPath \”:\” C:\\Users\\bufffiles\”}” -X PUT https://<historianservername>/historian-rest-api/v1/collector/buffercontrol`

Query parameters include the Payload parameter, which is a JSON file, which contains the following properties.

Table 57. Query Parameters

Parameter	Description	Required?	Values
`interfaceName`	The interface name of the collector.	Yes	String
`bufferMode`	Indicates whether you want to move or delete the files.	Yes	• 1: Indicates that the buffer files will be deleted. • 2: Indicates that the buffer files will be moved to the location specified in the bufferPath parameter.
`bufferPath`	The directory to which you want to move the buffer files. For example: `C:\\Data\\NewBufferFilesLocation` or `C:/Data/NewBufferPathLocation` **Note:** The double slash (\\) is required in the JSON format.	Yes (only if you want to move the buffer files)	String
`winUserName`	The Windows username.	Yes (only if you want to use the command-line mode)	String
`winPassword`	The Windows password	Yes (only if you want to use the command-line mode)	String

Table 58. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.
`Data`	String	No	Indicates if the task has been initiated (and if the collector is in the stopped state).

The Server Node Change API

Using the Server Node Change API, you can change the server node of a collector to a machine that has Historian 8.1 installed on it.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/collector/historiannodechange`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/collector/historiannodechange`
	
	`Payload`
	`{`
	`  “interfaceName":"<source server>_<type of the collector>_<destination server>",`
	`  “mode”:2,`
	`  "winUserName":"TestAdministrator",`
	`  "winPassword":"TestPassword",`
	`  "historianNode":"VMHISTWEBAUTO",`
	`  "historianUserName":" TestAdministrator2 ",`
	`  "historianpassword":" TestPassword2"`
	`}`

SAMPLE RESPONSE	`{`
	`    "ErrorCode": 0,`
	`    "ErrorMessage": ""`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \”interfaceName\”:\”<source server>_<type of the collector>_<destination server>\”, \”userName winUserName\”:\”tesrt\”, \”winpPassword \”:\”password\”, \”historianNode \”:\”nodename\”,\” historianUserName \”:\”husername\”, \” historianpassword \”:\”hpassword\”}” -X PUT https://<historianservername>/historian-rest-api/v1/collector/ historiannodechange`

Query parameters include the Payload parameter, which is a JSON file, which contains the following properties.

Table 59. Query Parameters

Parameter	Description	Required?	Values
`interfaceName`	The interface name of the collector.	Yes	String
`mode`	The mode to use to manage the collector.	Yes	• 1: service mode • 2: command-line mode
`winUserName`	The Windows username.	Yes (only if you want to use the command-line mode)	String

Parameter	Description	Required?	Values
`winPassword`	The Windows password.	Yes (only if you want to use the command-line mode)	String
`historianNode`	The host name of the new Historian destination machine. The destination machine must have Historian 8.1.	Yes	String
`historianUserName`	The Windows username of the new Historian destination machine.	Yes (only if you want to use the command-line mode)	String
`historianPassword`	The Windows password of the new Historian destination machine.	Yes (only if you want to use the command-line mode)	String

Table 60. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Get Collector Version API

Using the Get Collector Version API, you can view the version number of a collector.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/collector/version/{interfaceName}`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/collector/version/?interfaceName=<source server>_<type of the collector>_<destination server>`
SAMPLE RESPONSE	`{ "ErrorCode": 0, "ErrorMessage": null, "Data": { "Version": "8.1.2068.0" } }`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” http://<historianservername>/historian-rest-api/v1/collector/version/RSSERVER2012-02_Simulation`

Table 61. Query Parameters

Parameter	Description	Required?	Values
`interfaceName`	The interface name of the collector.	Yes	String

Table 62. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.
`Data`	String	Yes	Returns the version of the collector.

The Get Collector Status API

Using the Get Collector Status API, you can view the status of a collector.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/collector/status/{interfaceName}`
SAMPLE GET URI	`https://<historianservername>/historian-rest-api/v1/collector/status/RSSERVER2012-02_Simulation`
SAMPLE RESPONSE	`{ "ErrorCode": 0, "ErrorMessage": "null" "Data":{ "Status":"Running" } }`
SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” https://<historianservername>/historian-rest-api/v1/collector/status/RSSERVER2012-02_Simulation`

Table 63. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.

Parameter	Data Type	Required?	Description
`ErrorMessage`	String	Yes	For example, NULL.
`Data`	String	Yes	Returns the status of the collector.

The Collector Manager List API

Using the Collector Manager List API, you can view the list of collector agents machines associated with the Historian server.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/collectormanagerlist`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/collectormanagerlist`
SAMPLE RESPONSE	see below
SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" https://<historianservername>/historian-rest-api/v1/collectormanagerlist`

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Data": [
        {
            "Name": "CollectorManager::abc",
            "IPAddress": "[::ffff :<IP address>]",
            "Status": 1,
            "ComputerName": "abc "
        },
        {
            "Name": "CollectorManager::xyz",
            "IPAddress": "[::ffff:<IP address>]",
            "Status": 1,
            "ComputerName": "xyz"
        },
        {
            "Name": "CollectorManager::abc",
            "IPAddress": "[::ffff:<IP address>]",
            "Status": 1,
            "ComputerName": "abc"
        },
        {
            "Name": "CollectorManager::123",
            "IPAddress": "[::ffff:<IP address>]",
            "Status": 1,
            "ComputerName": "123"
        }
    ]
}
```

Table 64. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Collector Mode API

Using the Collector Mode API, you can view the running mode of a collector.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/collector/mode/<collector interface name>`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/collector/mode/<collector interface name>`
SAMPLE RESPONSE	`{ "ErrorCode": 0, "ErrorMessage": null, "Data": { "RunningMode": "Service Mode" } }`
SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” https://<historianservername>/historian-rest-api/v1/collector/mode/<host name>_Simulation_<IP address>_2`

Table 65. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Collector Details API

Using the Collector Details API, you can view the details of a collector.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/collector/details`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/collector/details`

SAMPLE RESPONSE	`{`
	`"ErrorCode":0,`
	`"ErrorMessage":null,`
	`"Data":`
	`[`
	`{`
	`"Name":"<value>",`
	`"ComputerName":"<value>",`
	`"Status":"Running",`
	`"ReportRate":0,`
	`"MaximumEventRate":0,`
	`"MinimumEventRate":0,`
	`"OutOfOrderEvents":0,`
	`"Overruns":0,`
	`"OverrunsPercent":0,`
	`"TotalEventsCollected":0,`
	`"TotalEventsReported":0,`
	`"LastDataValue":"\\`
	`\"1970-01-01T00:00:00.000Z\\\"",`
	`"Redundency":"",`
	`"Comments":"<username>--test2--\\`
	`\"2020-12-15T07:19:42.000Z\\\";",`
	`"Version":"9.0.4326.0",`
	`"CollectorCompression":0,`
	`"TagsCount":0`
	`}`
	`]`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” https://<historianservername>/historian-rest-api/v1/collector/details`

Table 66. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Offline Collectors API

Using the Offline Collectors API, you can view a list of offline collectors.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/offlinecollectors`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/offlinecollectors`

SAMPLE RESPONSE	`{`
	`"ErrorCode": 0,`
	`"ErrorMessage": null,`
	`"Data": [`
	`{`
	`"Name": "DISTMACHINE1_Cygnet",`
	`"ComputerName": "DISTMACHINE1",`
	`"Status": "Stopped"`
	`},`
	`{`
	`"Name": "NPI212611749M1_Cygnet",`
	`"ComputerName": "NPI212611749M1",`
	`"Status": "Stopped"`
	`},`
	`{`
	`"Name": "NPI212611749M1_Mqtt",`
	`"ComputerName": "NPI212611749M1",`
	`"Status": "Stopped"`
	`}`
	`]`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" https://<historianservername>/historian-rest-api/v1/offlinecollectors`

Table 67. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

Managing Data Stores

The Get Data Stores API

Using the Get Data Stores API, you can view the list of data stores in a system.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/datastores?dataStoreMask=`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-rest-api/v1/datastores?dataStoreMask=*`

SAMPLE RESPONSE

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Data": [
        {
            "Description": "The System Data Store.",
            "Id": "D3C23639-81CD-40F7-9CB0-37484FC5190D",
            "IsDefault": false,
            "IsSystem": true,
            "Name": "System",
            "NumberOfTags": 0,
            "State": 2,
            "DHSStorageName": "System Storage",
            "StorageType": 0,
            "Links": [
                {
                    "Rel": "self",
                    "Href": "/datastore/System"
                }
            ]
        },
        {
            "Description": "The Scada Buffer Data Store.",
            "Id": "39B39D42-DC7A-4048-9BA8-E4BAB4644B0C",
            "IsDefault": false,
            "IsSystem": false,
            "Name": "ScadaBuffer",
            "NumberOfTags": 0,
            "State": 2,
            "DHSStorageName": "xyz",
            "StorageType": 1,
            "Links": [
                {
                    "Rel": "self",
                    "Href": "/datastore/ScadaBuffer"
                }
            ]
        },
        {
```

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” https://<historianservername>/historian-rest-api/v1/datastores?dataStoreMask=*`

Table 68. Query Parameters

Parameter	Description	Required?	Values
`dataStoreMask`	The value of the data store mask.	No	String

Table 69. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Get Data Stores of Storage API

Using the Get Data Stores of Storage API, you can view the list of data stores in a location.

METHOD	GET
URI	`https://<historianservername>/historian-rest-api/v1/storage/datastores?storageName=`
SAMPLE QUERY PARAM GET URL	`https://<historianservername>/historian-resr-api/v1/storage/datastores?storageName=xx`

SAMPLE RESPONSE

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Data": [
        {
            "Description": "The Scada
 Buffer Data Store.",
            "Id": "39B39D42-DC7A-4048-9BA8-
E4BAB4644B0C",
            "IsDefault": false,
            "IsSystem": false,
            "Name": "ScadaBuffer",
            "NumberOfTags": 0,
            "State": 2,
            "DHSStorageName": "xyz",
            "StorageType": 1,
            "Links": [
                {
                    "Rel": "self",
                    "Href": "/datastore/
ScadaBuffer"
                }
            ]
        },
        {
            "Description": "The DHS System
 Data Store.",
            "Id": "56C1DFE9-D0BF-427F-B5D8-
B127E38B5C11",
            "IsDefault": false,
            "IsSystem": false,
            "Name": "DHSSystem",
            "NumberOfTags": 0,
            "State": 2,
            "DHSStorageName": "xyz",
            "StorageType": 0,
            "Links": [
                {
                    "Rel": "self",
                    "Href": "/datastore/
DHSSystem"
                }
            ]
        },
        {
            "Description": "The User Data
```

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” https://<historianservername>/historian-rest-api/v1/storage/datastores?storageName=xx`

Table 70. Query Parameters

Parameter	Description	Required?	Values
`storageName`	The name of the location whose data stores you want to view.	Yes	String

Table 71. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Add Datastore API

Using the Add Datastore API, you can create a data store in a Historian server.

METHOD	POST
URI	`https://<historianservername>/historian-rest-api/v1/datastoretostorage`

SAMPLE PATH PARAM GET URI	`https://<historianservername>/historian-rest-api/v1/datastoretostorage`
	
	`Payload`
	
	`{`
	
	`"dataStoreName": "abc",`
	
	`"storageName": "storage1",`
	
	`"description": "test",`
	
	`"isDefault": true`
	
	`}`

SAMPLE RESPONSE	`{`
	
	`    "ErrorCode": 0,`
	
	`    "ErrorMessage": null`
	
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \”dataStoreName \”:\”name\”,\" storageName \": \"sname\",\"description \":\" des\",\" isDefault \":false}” -X POST https://<historianservername>/historian-rest-api/v1/datastoretostorage`

Table 72. Query Parameters

Parameter	Description	Required?	Values
`Payload`	Contains the details of the data store in the JSON format.	Yes	Multiple

Table 73. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Delete Data Store API

Using the Delete Data Store API, you can delete a data store.

METHOD	DELETE
URI	`https://<historianservername>/historian-rest-api/v1/datastore`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/datastore`
	
	`Payload`
	
	`{`
	
	`"dataStoreName": ""`
	
	`}`

SAMPLE RESPONSE	`{`
	
	`    "ErrorCode": 0,`
	
	`    "ErrorMessage": null`
	
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \”dataStoreName \”:\”name\”}” -X DELETE https://<historianservername>/historian-rest-api/v1/datastore`

Table 74. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Data Store Update API

Using the Data Store Update API, you can modify a data store.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/dataStore/<data store name>`
SAMPLE URI	see below
SAMPLE RESPONSE	see below

SAMPLE URI

```
https://<historianservername>/historian-
rest-api/v1/dataStore/mirror1DS1
Payload

{

            "Description": "testing",

            "Id": "5761BCBF-A04D-494F-
AE6E-30F8652F4B96",

            "IsDefault": true,

            "IsSystem": false,

            "Name": "mirror1DS1",

            "NumberOfTags": 0,

            "State": 2,

            "DHSStorageName": "mirror1",

            "StorageType": 0,

            "Links": [

                {

                    "Rel": "self",

                    "Href": "/datastore/
mirror1DS1"

                }

            ]

        }
```

SAMPLE RESPONSE

```
{

    "ErrorCode": 0,

    "ErrorMessage": null

}
```

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \” Description\”:\”des\”,\"IsDefault \": true, \” IsSystem \”:false, \” Name\”:\” mirror1DS1\”,\”NumberOfTags \”:0,\”State \”:2, \”DHSStorageName\”:\”mirror1\”, \”StorageType \”:0,\”Links\”: [{\"Rel\": \"self\", \"Href\": \"/datastore/mirror1DS1\" }]}” -X PUT https://<historianservername>/historian-rest-api/v1/dataStore/mirror1DS1`

```
curl -i -H "Accept: application/json" -i -H
 "Content-Type: application/json"
-H "Authorization: Bearer <TOKEN>” -d “{ \”
 Description\”:\”des\”,\"IsDefault \":
 true,
\” IsSystem \”:false, \” Name\”:\”
 mirror1DS1\”,\”NumberOfTags \”:0,\”State
\”:2,
\”DHSStorageName\”:\”mirror1\”,
\”StorageType \”:0,\”Links\”: [{\"Rel\":
\"self\",
\"Href\": \"/datastore/mirror1DS1\" }]}”
 -X PUT https://<historianservername>/
historian-rest-api/v1/dataStore/mirror1DS1
```

Table 75. Query Parameters

Parameter	Description	Required?	Values
`Payload`	Contains the values of the attributes of the data store that you want to change.	Yes	Multiple

Table 76. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

The Default Data Store Update API

Using the Default Data Store Update API, you can change the default data store.

METHOD	PUT
URI	`https://<historianservername>/historian-rest-api/v1/storage/<location name>`

SAMPLE URI	`https://<historianservername>/historian-rest-api/v1/storage/NPI212611749M1`
	
	`Payload`
	
	`{`
	`"StorageName": "NPI212611749M1",`
	`"StorageType": 0,`
	`"NumberOfDataStores": 5,`
	`"NumberOfArchivers": 1,`
	`"DataStores": [`
	`"User",`
	`"testDS1",`
	`"ScadaBuffer",`
	`"testDS2",`
	`"DHSSystem"`
	`],`
	`"Id": "9CD06AFB-1566-4CE6-99D4-B2F65857F33A",`
	`"IsDefault": true,`
	`"LastModifiedUser": null,`
	`"LastModifiedTime": "1970-01-01T00:00:00.000Z",`
	`"ArchiverServices": [`
	`"DataArchiver_NPI212611749M1", "DataArchiver_distamchine1"`
	`]`
	`}`
	`}`

SAMPLE RESPONSE	`{`
	`"ErrorCode": 0,`
	`"ErrorMessage": null,`
	`}`

SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \”StorageName\”:\”name\”,\"StorageType\": 0, \”NumberOfDataStores\”:5,\” NumberOfArchivers\”:1, \”IsDefault\”:true,\”ArchiverServices\”: [\"DataArchiver_NPI212611749M1\", \"DataArchiver_distamchine1"\"]}” -X PUT https://<historianservername>/historian-rest-api/v1/storage/NPI212611749M1`

Table 77. Query Parameters

Parameter	Description	Required?	Values
`Payload`	Contains the values of the attributes of the default data store that you want to change.	Yes	Multiple

Table 78. Response Parameters

Parameter	Data Type	Required?	Description
`ErrorCode`	Integer	Yes	For example, ErrorCode = 0 implies the operation was successful.
`ErrorMessage`	String	Yes	For example, NULL.

Managing Tags

The Tags API

The Tags API retrieves the qualified tag name list by a given `nameMask`.

Note: URI format supports asterisks (*) and question marks (?).

METHOD	GET
URI	`https://<historianservername>:8443/historian-rest-api/v1/tags/{nameMask}/{maxNumber}`
SAMPLE URI	`https://<historianservername>:8443/historian-rest-api/v1/tags?nameMask=*&maxNumber=100`
SAMPLE cURL COMMAND	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” https://<nodename>:8443/ historian-rest-api/v1/tags?nameMask=*&maxNumber=<Number_Of_Tags>`

Table 79. Query Parameters

Parameter	Description	Required?	Values
`nameMask`	Tagmask that searches for all tags that match the mask and applies the remaining criteria to retrieve data. The mask can include wildcards, such as asterisks (*).	Optional	String
`maxNumber`	Maximum tag number provides the limit while returning the results (0 by default). This means that for a query, if using 0, all tags are returned. If a negative number is used, then 0 is used for the maxNumber. If a positive number is used, then that number of tags is returned. In addition, an error number of +14 notifies the user that there are more than the requested number of tags in the system.	Optional	Integer 0 by default

Table 80. Response Parameters

Parameter	Data Type	Required?	Description
ErrorCode	Number	Yes	For example, ErrorCode = 0, which means the operation was successful.
ErrorMessage	String	Yes	For example, NULL.
tags	String	Yes	Includes the following: • ALT_SENSOR • tagName1 • tagName2

The Tagslist API

The Tags List API `GET` method retrieves the list of tags.

When retrieving large tag lists from Historian, you can paginate the response, allowing you to get the next page, go the end, go back one page, and go to the beginning.

Request Parameters

You can use wildcards (*, &,?) with string parameters for pattern matching. Results are sorted in ascending tag names. All parameters use the `AND` operator. The `OR` operator is not supported.

All request parameters are optional.

When there are NO wildcard characters (*, &,?) with string parameters for pattern matching, then search would be a `contains` search

Example: “dog” pattern will match “dog1”, “dog2”,”dogs”, “dogx”, “dog12”, “dogs are faithful”, “1dog1” and so on. When wildcards (*,&, ?) are used in the search string parameters for pattern matching, then they work as per the wildcard character definition.

? - Single character matching

* - Multi character matching

Eg1: “dog?” pattern will match: “dog1”, “dog2”,”dogs”, “dogx” and so on but not “dog12” or “dogs are faithful”

Eg2: “dog*” pattern will match “dog1”, “dog2”,”dogs”, “dogx”, “dog12”, “dogs are faithful” and so on but not “1dog1”

Parameter Name	Data Type	Default	Description
`calctype`	Integer	`-1`	Returns exact match of calc type (0,1,2).
`collectiondisabled`	Boolean	If ignored, all types considered.	Must be only true / false, else error out.
`collectioninterval`	Integer	0 – means all intervals	If `collectorinterval` = 0 consider all intervals, else exact match.
`collectorcompression`	Boolean	`*`	Returns exact match of collector compression (true/false).
`collectorname`	String	`*`	Default `*` means consider all.
`collectortype`	Integer	0 – means consider all collector types	Returns exact match of collector type.
`comment`	String	`*`	Default `*` means consider all.
`data storename`	String	`*`	Default `*` means consider all.
`datatype`	Integer	`0` – means consider all data types	Returns exact match of data type.
`description`	String	`*`	Default `*` means consider all.
`egudescription`	String	`*`	Default `*` means consider all.
`enumeratedset`	String	`*`	Default `*` means consider all.
`hasalias`	Boolean	If ignored, all types considered.	Must be only true / false, else error out.
`isstale`	Boolean	If ignored, all types considered.	Must be only true / false, else error out.

Parameter Name	Data Type	Default	Description
`lastmodified`	String	`1970-01-01T00:00:00.00Z`	`>=` is applied so that last modified tag is returned in the result set.
`lastmodifieduser`	String	`*`	Default `*` means consider all.
`numberofelements`	Integer	0	If 0, ignore this parameter else returns exact match of number of elements.
`pageno`	Integer	`1` Must be > `1`	If invalid, no data is returned.
`pagesize`	Integer	`128` Max `512` Min `2`	If out of range, returns error.
`sourceaddress`	String	`*`	Default `*` means consider all.
`tagname`	String	`*`	Default `*` means consider all.
`userdefinedtypename`	String	`*`	Default `*` means consider all.

The Tags List Pagination Parameters

When retrieving large tag lists from Historian, you can paginate the response, allowing you to get the next page, go the end, and go back on page and to the beginning. Results with no errors return these pagination parameters:

Parameter	Value
`pagesize`	Current page size.
`pageno`	Current page number
`totalcount`	Total result other than current page.
Links to URLs	All URLs are part of the HTTP response headers. • `first` – First page `tags list` URL (can be null if count is 0). • `last` - Last page `tags list` URL (can be null if count is 0). • `prev` – Previous page `tags list` URL (can be null if current page is 1). • Next - Next page `tags list` URL (can be null if current page is last page).

Table 81. Sample cURL commands

METHOD	GET
`SAMPLE cURL COMMAND: [lastmodified]`	curl -i -H "Accept: application/json" -H "Authorization:Bearer <TOKEN>" http://<nodename>:8443/historian-rest-api/v1/tagslist?lastmodified=2017-05-01T00:00:00.00Z

METHOD	GET
`SAMPLE cURL COMMAND: [pageno=0]`	curl -i -H "Accept: application/json" -H "Authorization:Bearer <TOKEN>" http://<nodename>:8443/historian-rest-api/v1/tagslist?pageno=0
`SAMPLE cURL COMMAND: [pageno=1]`	curl -i -H "Accept: application/json" -H "Authorization:Bearer <TOKEN>" http://<nodename>:8443/historian-rest-api/v1/tagslist?pageno=1
`SAMPLE cURL COMMAND: [complete tagslist]`	curl -i -H "Accept: application/json" -H "Authorization:Bearer <TOKEN>" http://<nodename>:8443/historian-rest-api/v1/tagslist

Example Queries

The following request returns first page as `pageno` is ignored and `pagesize` is defaulted to 128, all tags are considered:

```
<baseurl>/v1/tagslist
```

The following request returns first page as `pageno` is ignored and `pagesize` is defaulted to 128, all tags are considered that are modified after `2017-05-01T00:00:00.00Z`.

```
<baseurl>/v1/tagslist?lastmodified=2017-05-01T00:00:00.00Z
```

Example Results

The following info is returned for each tag from the criteria provided in the request as an array of tag info.

- `tagid` - String
- `tagname` - String
- `description` - String
- `datatype` - Integer
- `collectorname` - String
- `collectortype` - Integer
- `data storename` - String
- `egudescription` - String
- `comment` - String
- `sourceaddress` - String
- `sourceaddress` - String
- `collectioninterval` - Integer
- `collectorcompression` - Boolean
- `lastmodifieduser` - String
- `enumeratedset` - String
- `userdefinedtypename` - String
- `calctype` - Integer
- `isstale` - Boolean
- `lastmodified` - Long

- `lastmodified` - Long
- `lastmodifiedString` – String – In readable format
- `has alias` - Boolean
- `numberofelements` - Integer
- `collectiondisabled` - Boolean

Example:

```
{
 "TotalCount": 1031,
 "Page": 1,
 "PageSize": 4,
 "Tags": [
  {
   "Tagid": "adb70ebf-978f-46dd-ac6f-5e863cdb0739",
   "Tagname": "-anilgwxb.Constant",
   "Description": "anilgwxb.Constant",
   "DataType": 3,
   "CollectorName": "ANILGWXB_Simulation",
   "CollectorType": 2,
   "DataStoreName": "User",
   "EngineeringUnits": "",
   "Comment": "",
   "SourceAddress": "$Constant",
   "CollectionInterval": 1000,
   "CollectorCompression": false,
   "LastModifiedUser": null,
   "EnumeratedSetName": "",
   "UserDefinedTypeName": "",
   "CalcType": 0,
   "IsStale": false,
   "HasAlias": false,
   "NumberOfElements": 0,
   "CollectionDisabled": false,
   "LastModified": 1496992712,
   "LastModifiedString": "2017-06-09T07:18:32Z"
  },
  {
   "Tagid": "88e1f448-643f-465a-95c2-d2bd08870547",
   "Tagname": "anilgwxb.Constant_1%Noise",
   "Description": "anilgwxb.Constant_1%Noise",
   "DataType": 3,
   "CollectorName": "ANILGWXB_Simulation",
   "CollectorType": 2,
   "DataStoreName": "User",
   "EngineeringUnits": "",
   "Comment": "",
   "SourceAddress": "$Constant_1%Noise",
   "CollectionInterval": 1000,
   "CollectorCompression": false,
   "LastModifiedUser": null,
   "EnumeratedSetName": "",
    "UserDefinedTypeName": "",
    "CalcType": 0,
    "IsStale": false,
    "HasAlias": false,
    "NumberOfElements": 0,
    "CollectionDisabled": false,
    "LastModified": 1496992712,
    "LastModifiedString": "2017-06-09T07:18:32Z"
    },
<SNIP>
 ],
 "Links": {
  "first": "https://anilgwxb:8443/historian-rest-api/v1/
tagslist?pageno=1&pagesize=4",
  "last": "https://anilgwxb:8443/historian-rest-api/v1/tagslist?
pageno=258&pagesize=4",
  "prev": null,
  "next": "https://anilgwxb:8443/historian-rest-api/v1/tagslist?
pageno=2&pagesize=4"
 }
}
```

The Raw Data API

The Raw Data API queries raw data, such as a number of samples or the time range for a list of tags. If the count is not zero, then the API service returns the number of raw samples taken beginning from the start time. If the count is zero, then the service returns the raw samples taken between the start time and the end time.

METHOD:	GET, POST

URI:	**GET**
	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/raw/{tagNames}/{start}/{end}/{direction}/{count}`
	POST
	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/raw/{start}/{end}/{direction}/{count}`

SAMPLE GET URI:	**Raw By Number**
	Count value is a non-zero positive number, and end time is greater than start time.
	`https://<historianservername>:8443/historian-rest-api/datapoints/raw/tagName1/2013-10-02T11:30:00.111Z/2013-10-02T11:31:11.111Z/0/100https://<historianservername>:8443/historian-rest-api/datapoints/raw?tagNames=tagName1&start=2013-10-02T11:30:00.111Z&end=2013-10-02T11:31:11.111Z&count=100&direction=0`
	Raw By Time
	The count value equals 0.
	`https://<historianservername>:8443/historian-rest-api/datapoints/raw/tagName1/2013-10-02T11:30:00.111Z/2013-10-02T11:31:11.111Z/0/0https://<historianservername>:8443/historian-rest-api/datapoints/raw?tagNames=tagName1&start=2013-10-02T11:30:00.111Z&end=2013-10-02T11:31:11.111Z&count=0&direction=0`

SAMPLE POST URI:	**Raw By Number**
	Count value is a non-zero positive number, and end time is greater than start time.
	`https://<historianservername>:8443/historian-rest-api/datapoints/raw/2013-10-02T11:30:00.111Z/2013-10-02T11:31:11.111Z/0/100https://<historianservername>:8443/historian-rest-api/datapoints/raw?start=2013-10-02T11:30:00.111Z&end=2013-10-02T11:31:11.111Z&count=100&direction=0`
	Raw By Time
	The count value equals 0.
	`https://<historianservername>:8443/historian-rest-api/datapoints/raw/2013-10-02T11:30:00.111Z/2013-10-02T11:31:11.111Z/0/0https://<historianservername>:8443/historian-rest-api/datapoints/raw?start=2013-10-02T11:30:00.111Z&end=2013-10-02T11:31:11.111Z&count=0&direction=0`

SAMPLE cURL COMMAND (GET): [Raw By Number]	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” http://<nodename>:8443/ historian-rest-api/v1/ datapoints/raw/<tagName>/<start time>/<end time>/<direction>/<count>`
SAMPLE cURL COMMAND (GET): [Raw By Time]	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” http://<nodename>:8443/ historian-rest-api/v1/ datapoints/raw/<tagName>/<start time>/<end time>/<direction>/0`
SAMPLE cURL COMMAND (POST): [Raw By Number]	`curl –X POST -i -H "Content-Type: application/json" -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” -d “{\”tagNames\”:\”<tagName>;<tagName>\”}” http:// <nodename>/ historian-rest-api/v1/datapoints/raw/ <start time>/<end time>/<direction>/<count>`
SAMPLE cURL COMMAND (POST): [Raw By Time]	`curl –X POST -i -H "Content-Type: application/json" -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” -d “{\”tagNames\”:\”<tagName>;<tagName>\”}” http:// <nodename>/ historian-rest-api/v1/ datapoints/raw? start=<start time>&end=<end time>&direction=<direction>&count=<count>`

Table 82. Query Parameters

Parameter	Description	Required?	Values
TagNames	Queries the specified tag names.	Yes	String

Parameter	Description	Required?	Values
Start	Start time of the query, in ISO data format (such as YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime
End	End time of the query, in ISO data format (such as YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime
Direction	Specifies the direction (Forward or Backward from the starting time) of data sampling from the archive. The default value is Forward (0).	Yes	Integer, with a value such as 0.
Count	Count of archive values within each calculation interval.	Yes	Integer, with a value such as 100.

Table 83. Response Parameters

Parameter	Data Type	Required?	Description
ErrorCode	Integer	Yes	For example, 0.
ErrorMessage	String	Yes	For example, NULL.
Data	String	Yes	The object container for the following parameters: **DataType** DoubleFloat, which stores decimal values up to 15 places. **ErrorCode** Value is 0, which means the operation was successful. **TagName** Example: TagName1. **Samples** Provides TimeStamp, Value and Quality for each sample. For example, TimeStamp = 2013-10-02T11:30:00.111Z, Value = 34.26155, and Quality = 3.

The Interpolated Data API

The Interpolated Data API queries interpolated values for a list of tags. If the start time equals the end time, the request returns one sample.

METHOD:	GET, POST

URI:	**GET** `https://<historianservername>:8443/historian-rest-api/v1/datapoints/interpolated/{tagNames}/{start}/{end}/{count}/{intervalMs}` **POST** `https://<historianservername>:8443/historian-rest-api/v1/datapoints/interpolated/{start}/{end}/{count}/{intervalMs}`
SAMPLE GET URI:	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/interpolated/tagName1/2013-10-02T11:30:00.111111Z/2013-10-02T11:31:11.111Z/100/10000`
SAMPLE POST URI:	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/interpolated/2013-10-02T11:30:00.111111Z/2013-10-02T11:31:11.111Z/100/10000`
SAMPLE cURL COMMAND (GET):	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” http://<nodename>:8443/ historian-rest-api/v1/ datapoints/interpolated/<tagName>/<start time>/<end time>/<count>/<intervalMS>`
SAMPLE cURL COMMAND (POST):	`curl -i –X POST -H "Content-Type: application/json" -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” -d “{\”tagNames\”:\”<tagName>\”}” http://<nodename>:8443/ historian-rest-api/v1/datapoints/interpolated/<start time>/<end time>/<count>/<intervalMS>`

Table 84. Query Parameters

Parameter	Description	Required?	Values
TagName	Queries the tag names specified.	Yes	String
Start	Start time of the query, in ISO data format (such as YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime
End	End time of the query, in ISO data format (such as YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime
Count	Count of archive values within each calculation interval.	Yes	Integer, with a value such as 100.
intervalMS	Interval in milliseconds.	Yes	64-bit signed integer, with a value such as 10000.

Table 85. Response Parameters

Parameter	Data Type	Required?	Description
ErrorCode	Integer	Yes	For example, 0.
ErrorMessage	String	Yes	For example, NULL.

Parameter	Data Type	Required?	Description
Data	String	Yes	The object container for the following parameters: **DataType** DoubleFloat, which stores decimal values up to 15 places. **ErrorCode** Value is 0, which means the operation was successful. **TagName** Example is TagName1. **Samples** Provides TimeStamp, Value and Quality for each sample. For example, TimeStamp = 2013-10-02T11:30:00.111Z, Value = 34.26155, and Quality = 3.

The Current Value API

The Current Value API queries the current value data and reads the current values for a list of tags. If the start time is equal to end time, the request returns one sample.

METHOD:	GET, POST

URI:	**GET**
	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/raw/{tagNames}/{start}/{end}/{direction}/{count}`
	POST
	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/currentvalue`

SAMPLE GET URI:	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/currentvalue?tagNames=tagName1`
SAMPLE POST URI:	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/currentvalue`
SAMPLE cURL COMMAND (GET):	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" http://<nodename>:8443/ historian-rest-api/v1/ datapoints/currentvalue/<tagName>`
SAMPLE cURL COMMAND (POST):	`curl -i –X POST -H "Content-Type: application/json" -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" -d "{\"tagNames\":\"<tagName>\"}" http://<nodename>:8443/ historian-rest-api/v1/datapoints/currentvalue`

Table 86. Query Parameters

Parameter	Description	Required?	Values
TagNames	Queries the specified tag names.	Yes	String

Table 87. Response Parameters

Parameter	Data Type	Required?	Description
ErrorCode	Integer	Yes	For example, 0.
ErrorMessage	String	Yes	For example, NULL.
Data	String	Yes	The object container for the following parameters: **DataType** DoubleFloat, which stores decimal values up to 15 places. **ErrorCode** Value is 0, which means the operation was successful. **TagName** Example is TagName1. **Samples** Provides TimeStamp, Value and Quality for each sample. For example, TimeStamp = 2014-01-01T12:00:00Z, Value = 34.26155, and Quality = 3.

The Calculated Data API

The Calculated Data API queries the calculated data for a list of tags. Data can be requested using a number of samples or a time range for a list of tags. If the count is not zero, the service returns the number of raw samples beginning from the start time. If the count is zero, the services uses the interval, start time, and end time to calculate the required sample number.

METHOD:	GET, POST

URI:	**GET**
	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/calculated/{tagNames}/{start}/{end}/{calculationMode}/{count}/{intervalMs}`
	POST
	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/calculated/{start}/{end}/{calculationMode}/{count}/{intervalMs}`

SAMPLE GET URI:	**Number of Samples**
	`https://<historianservername>:8443/`
	`historian-rest-api/v1/datapoints/calculated/`
	`tagName1/2013-10-02T11:30:00.111Z/2013-10-02T11:31:11.111Z/1/100/1000`
	Time Range for List of Tags
	`https://<historianservername>:8443/`
	`historian-rest-api/v1/datapoints/calculated?`
	`tagNames=tagName1&start=2013-10-02T11:30:00.111Z&end=2013-10-02T11:31:11.111Z&count=100&calcula`

SAMPLE POST URI:	**Number of Samples**
	`https://<historianservername>:8443/`
	`historian-rest-api/v1/datapoints/`
	`calculated/2013-10-02T11:30:00.111Z/2013-10-02T11:31:11.111Z/1/100/1000`
	Time Range for List of Tags
	`https://<historianservername>:8443/`
	`historian-rest-api/v1/datapoints/calculated?`
	`start=2013-10-02T11:30:00.111Z&end=2013-10-02T11:31:11.111Z&count=100&calculationMode=1&interva`

SAMPLE cURL COMMAND (GET):	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>”`
	`http://<nodename>:8843/ historian-rest-api/v1/ datapoints/calculated/`
	`<tagName>/<start time>/<end time>/<count>/<calculation mode>/<intervalMS>`

SAMPLE cURL COMMAND (POST):	`curl -i –X POST -H "Content-Type: application/json" -H "Accept:`
	`application/json" -H "Authorization: Bearer <TOKEN>” -d “{\”tagNames\”:`
	`\”<tagName>\”}” http://<nodename>:8843/ historian-rest-api/v1/ datapoints/`
	`calculated/<start time>/<end time>/<count>/<calculationmode>/<intervalMS>`

Table 88. Query Parameters

Parameter	Description	Required?	Values
TagNames	GE identifier for a location.	Yes	1000000106
Start	Start time of the query, in ISO data format (such as YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime
End	End time of the query, in ISO data format (such as YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime
Count	Count of archive values within each calculation interval.	Yes	Integer, with a value such as 100.
Calculation Mode	End time in milliseconds.	Yes	Integer, with a value such as 1.
IntervalMS	Interval in milliseconds.		64-bit signed integer, with a value such as 1000.

Table 89. Response Parameters

Parameter	Data Type	Required?	Description
ErrorCode	Integer	Yes	For example, 0.
ErrorCode	String	Yes	For example, NULL.
Data	String	Yes	The object container for the following parameters: **DataType** DoubleFloat, which stores decimal values up to 15 places. **ErrorCode** Value is 0, which means the operation was successful. **TagName** Example is TagName1. **Samples** Provides TimeStamp, Value and Quality for each sample. For example, TimeStamp = 2013-10-02T11:30:00.111Z, Value = 34.26155, and Quality = 3.

The Sampled Data API

The Sampled Data API queries the sampled data for a list of tags. Data can be requested using a number of samples or a time range for a list of tags. If the count is not zero, the service returns the number of raw samples beginning from the start time. If the count is zero, the services uses the interval, start time, and end time to calculate the required sample number.

Note: For the query, you can also use optional parameters such as FilterMode and ReturnDataFields. Unused parameters can be omitted.

METHOD:	GET, POST
URI:	**GET** `https://<historianservername>:8443/historian-rest-api/v1/datapoints/sampled` **POST** `https://<historianservername>:8443/historian-rest-api/v1/datapoints/sampled`
SAMPLE GET URI:	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/sampled?tagNames=tagName1&start=2013-10-02T11:30:00.111Z&end=2013-10-02T11:31:11.111Z&samplingMode=1&calc`
SAMPLE POST URI:	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/sampled`

SAMPLE cURL COMMAND (GET):	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" http://<nodename>:8443/ historian-rest-api/v1/ datapoints/ sampled/<tagName>/<start time>/<end time>/<direction>/<count>/ <intervalMS>`
SAMPLE cURL COMMAND (POST):	`curl -i –X POST -H "Content-Type: application/json" -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" -d "{ \"tagNames \":\"<tagName>\", \"start\": \"<start>\", \"end\": \"<end> \", \"samplingMode\": <samplingMode>, \"calculationMode\": <calculationMode>, \"direction\": <direction>, \"count\": <count>, \"returnDataFields\": <returnDataFields>, \"intervalMs\": <intervalMs>, \"queryModifier\": <queryModifier>, \"filterMode\": <filterMode>, \"filterExpression\": \"<filterExpression>\"}" http:// <nodename>:8443/historian-rest-api/v1/datapoints/sampled`

Table 90. Query Parameters

Parameter	Description	Required?	Values
TagNames	Queries the tag names specified.	Yes	String
Start	Start time of the query, in ISO data format (such as YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime
End	End time of the query, in ISO data format (such as YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime
SamplingMode	Also known as SamplingModeType.	Optional	Integer, with a value such as 1.
CalculationMode	Also known as CalculationModeType.	Optional	Integer, with a value such as 1.
Direction	Specifies the direction (Forward or Backward from the starting time) of data sampling from the archive. The default value is Forward (0).	Optional	Integer, with a value such as 0.
Count	The count of archive values within each calculation interval.	Optional	Integer, with a value such as 0.
IntervalMS	Interval in milliseconds.	Optional	64-bit signed integer, with a value such as 1000.

Table 91. Response Parameters

Parameter	Data Type	Required?	Description
ErrorCode	Integer	Yes	For example, 0.
ErrorCode	String	Yes	For example, NULL.

Parameter	Data Type	Required?	Description
Data	String	Yes	The object container for the following parameters: **DataType** DoubleFloat, which stores decimal values up to 15 places. **ErrorCode** Value is 0, which means the operation was successful. **TagName** Example is TagName1. **Samples** Provides TimeStamp, Value and Quality for each sample. For example, TimeStamp = 2013-10-02T11:30:00.111Z, Value = 34.26155, and Quality = 3.

The Trend Data API

The Trend Data API queries the trend data for a list of tags.

Note: For the query, you can also use optional parameters such as FilterMode and StatisticsItemFilter. Unused parameters can be omitted.

METHOD:	GET, POST
URI:	**GET** `https://<historianservername>:8443/historian-rest-api/v1/datapoints/trend` **POST** `https://<historianservername>:8443/historian-rest-api/v1/datapoints/trend`
SAMPLE GET URI:	`https://<historianservername>:8443/historian-rest-api /v1/datapoints/trend?tagNames=tagName1&start=2013-10-02T11: 30:00.111Z&end=2013-10-02T11:31:11.111Z&samplingMode=1&calculationMode=1 &direction=0&count=0&intervalMs=1000`
SAMPLE POST URI:	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/trend`
SAMPLE cURL COMMAND (GET):	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” http://<nodename>:8443/ historian-rest-api/v1/datapoints/trend/<tagName>/<start time>/<end time>/<samplingMode>/<calculationMode>/<direction>/<count>/<intervalMS>`

SAMPLE cURL COMMAND (POST):	`curl -i –X POST -H "Content-Type: application/json" -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \”tagNames \”:\”<tagName>\”, \”start\”: \”<start>\”, \”end\”: \”<end> \”, \”samplingMode\”: <samplingMode>, \”calculationMode\”: <calculationMode>, \”direction\”: <direction>, \”count\”: <count>, \”returnDataFields\”: <returnDataFields>, \”intervalMs\”: <intervalMs>, \”queryModifier\”: <queryModifier>, \”filterMode\”: <filterMode>, \”filterExpression\”: \”<filterExpression>\”}” http://<nodename>:8443/historian-rest-api/v1/datapoints/trend`

Table 92. Query Parameters

Parameter	Description	Required?	Values
TagNames	Queries the tag names specified.	Yes	String
Start	Start time of the query, in ISO data format (such as YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime
End	End time of the query, in ISO data format (such as YYYY-MM-DDTHH:mm:ss.SSSZ).	Yes	DateTime
SamplingMode	Also known as SamplingModeType.	Optional	Integer, with a value such as 1.
CalculationMode	Also known as CalculationModeType.	Optional	Integer, with a value such as 1.
Direction	Specifies the direction (Forward or Backward from the starting time) of data sampling from the archive. The default value is Forward (0).	Optional	Integer, with a value such as 0.
Count	The count of archive values within each calculation interval.	Optional	Integer, with a value such as 0.
IntervalMS	Interval in milliseconds.	Optional	64-bit signed integer, with a value such as 1000.

Table 93. Response Parameters

Parameter	Data Type	Required?	Description
ErrorCode	Integer	Yes	For example, 0.
ErrorMessage	String	Yes	For example, NULL.

Parameter	Data Type	Required?	Description
Data	String	Yes	The object container for the following parameters: **TagName** Name of the tag, such as ahistfile.Simulation00001. **TagSource** Location where tags are being searched for. **DataType** Float, which stores decimal values up to 6 places. **Trend** Provides TimeStamp, Value and Quality for each sample. For example, TimeStamp = 2016-03-15T04:53:17.000Z, Value = 170903.6563, and Quality = True.

The Add Single Tag API

For the Add Single Tag API, you can add a new tag to Historian, and the tag name and data type must be provided in the payload (parameter) of the method. All other tags are optional. If a property is provided, the respective validation is performed at the server end. If the tag exists, then any new properties provided in the payload are applied to the existing tag.

METHOD:	POST
URI:	`https://<historianservername>:8443/historian-rest-api/v1/tags/addtag`

SAMPLE DELETE URI:	`https://<historianservername>:8443/historian-rest-api/v1/tags/addtag`
	
	`Payload:`
	
	`{`
	`    "Name" : "SampleTag",`
	`    "DataType" : 3`
	`}`

SAMPLE cURL COMMAND:	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \”Name\”:\”Sampletag\”, \"DataType\":3}” -X POST https://<historianservername>:8443/historian-rest-api/v1/tags/addtag`

Table 94. Query Parameters

Parameter	Description	Required?	Values
Payload	JSON array of PropertyName and PropertyValue.	Yes. "Name" and "DataType" properties are required. All other properties are optional.	Multidata types. See Payload Parameter *(page 18)* for a list of tag properties used to update a tag configuration.

Sample Response

Parameter	Data Type	Required?	Description
ErrorCode	Integer	Yes	For example, 0.
ErrorMessage	String	Yes	For example, NULL.

The Add Bulk Tags API

For the Add Bulk Tags API, you can add new tags to Historian using an array, and the tag names and data types must be provided in the payload (parameter) of the method. All other tags are optional. If a property is provided, the respective validation is performed at the server end. If the tags exist, then any new properties provided in the payload are applied to the existing tags. The payload is be an array of tags defined.

METHOD:	POST
URI:	`https://<historianservername>:8443/historian-rest-api/v1/tags/addtags`

SAMPLE DELETE URI:

```
https://<historianservername>:8443/historian-rest-api/v1/tags/addtags

Payload:

[
{
    "Name" : "SampleTag1",
    "DataType" : 3
},
{
    "Name" : "SampleTag2",
    "DataType" : 3
}
]
```

SAMPLE cURL COMMAND:	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -d “[{ \”Name\”:\”Sampletag1\”}, { \”Name\”:\”Sampletag2\”}]” -X POST https://<historianservername>:8443/historian-rest-api/v1/tags/addtags`

Table 95. Query Parameters

Parameter	Description	Required?	Values
Payload	JSON array tags with individual tags of PropertyName and PropertyValue.	Yes. "Name" and "DataType" properties are required. All other properties are optional.	Multidata types. See Payload Parameter *(page 18)* for a list of tag properties used to update a tag configuration.

Table 96. Response Parameters

Parameter	Data Type	Exists?	Description
TagName	String	Yes	Tag name.
ErrorCode	Integer	Yes	For example, 0.
ErrorMessage	String	Yes	For example, NULL.

The Update Tag Configuration API

The Update Tag Configuration API allows you to set or modify any tag property values. You cannot, however, rename a tag using this API.

METHOD:	PUT
URI:	`https://<historianservername>:8443/historian-rest-api/v1/tags/properties/tagName`

SAMPLE DELETE URI:	`https://<historianservername>:8443/historian-rest-api/v1/tags/properties/tagName`
	
	`Payload:`
	`{`
	`    "PropertyName" : "PropertyValue"`
	`}`

SAMPLE cURL COMMAND:	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \”Description\”:\”SampleDesc\”}” -X PUT https://<historianservername>:8443/historian-rest-api/v1/tags/properties/tagName`

Table 97. Query Parameters

Parameter	Description	Required?	Values
tagName	Tag name for which properties need to be set or modified.	Yes	String
Payload	JSON array of PropertyName and PropertyValue.	At least one property must be provided.	Multidata types. See Payload Parameter *(page 18)* for a list of tag properties used to update a tag configuration.

Table 98. Response Parameters

Parameter	Data Type	Required?	Description
ErrorCode	Integer	Yes	For example, 0.
ErrorMessage	String	Yes	For example, NULL.

The Get Tag Properties API

You can use this API to specify which properties are required for retrieval. If no property names are provided, then all properties are retrieved. When using the Get Tag Properties method, requesting a non-existent tag name returns an error.

METHOD:	GET / POST
URI: (GET)	`https://<historianservername>:8443/historian-rest-api/v1/tags/properties/tagName` This URI returns all tag properties.

URI: (POST)	`https://<historianservername>:8443/historian-rest-api/v1/tags/properties/tagName`
	`Payload`
	`{`
	`"PropertyName1" : 1,`
	`"PropertyName2" : 1`
	`}`

SAMPLE GET URI:	`https://<historianservername>:8443/historian-rest-api/v1/tags/properties/tagName`

SAMPLE POST URI:	`https://<historianservername>:8443/historian-rest-api/v1/tags/properties/tagName`
	`Payload:`
	`{`
	`    "Description" : 1`
	`}`

SAMPLE cURL GET COMMAND:	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -X GET https://<historianservername>:8443/historian-rest-api/v1/tags/properties/tagName`
SAMPLE cURL POST COMMAND:	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>" -d "{ \"Description\": 1}" -X POST https://<historianservername>:8443/historian-rest-api/v1/tags/properties/tagName`

Table 99. Query Parameters

Parameter	Description	Required?	Values
tagName	Tag name for which properties need to be retrieved.	Yes	String

Parameter	Description	Required?	Values
Payload	JSON array of PropertyName and boolean (true/false).	At least one property must be provided.	Multi data types. See Payload Parameter *(page 18)* for a list of tag properties used to update a tag configuration.

Note: The query payload contains all the tag properties you want returned from the server. In the Update Tag Config method, you need to provide the actual tag property value. However, in the Get Tag Properties method, you need to provide the property and a value of 1 (true), to allow it to be read from the server and returned.

Table 100. Response Parameters

Parameter	Data Type	Required?	Description
ErrorCode	Integer	Yes	For example, 0.
ErrorMessage	String	Yes	For example, NULL.
Name	String	Optional	If no error, then the tag name of query is returned and all requested parameters.

The Delete Tag API

The Delete Tag API provides the ability to delete an existing tag from the Historian server.

Its URI format supports question marks (?).

METHOD:	DELETE
URI:	`https://<historianservername>:8443/historian-rest-api/v1/tags/tagName?{permanentDelete}`
SAMPLE DELETE URI:	`https://<historianservername>:8443/historian-rest-api/v1/tags/tagName?permanentDelete=true`
SAMPLE CURL COMMAND:	`curl -i -H "Authorization: Bearer <TOKEN>" -X DELETE https://<historianservername>:8443/historian-rest-api/v1/tags/tagName?permanentDelete=<true|false>`

Table 101. Query Parameters

Parameter	Description	Required?	Values
tagName	Name of the tag to be deleted.	Yes	String
permanentDelete	Deletes the tag permanently from the Historian server if the value passed in is true. If the parameter is not provided, then permanentDelete is assumed to be false.	Optional (false is default)	Boolean, true or false

Table 102. Response Parameters

Parameter	Data Type	Required?	Description
ErrorCode	Number	Yes	For example, ErrorCode=0, which means the operation was successful.
ErrorMessage	String	Yes	For example, NULL.

The Query Results API

The Query Results API enables you to include the number of samples required, by providing an end point to configure query results.

The minimum number of samples should be 1000.

METHOD:	PUT
URI:	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/configuration/{maxDataQueryResultSize}`
SAMPLE URI:	`https://<historianservername>:8443/historian-rest-api/v1/datapoints/configuration?maxDataQueryResultSize=6000`
SAMPLE CURL COMMAND:	`curl -i -H "Accept: application/json" -H "Authorization: Bearer <TOKEN>" https://<nodename>:8443/ historian-rest-api/v1/datapoints/configuration?maxDataQueryResultSize=<Number_Of_Query_Results>`

Table 103. Query Parameters

Parameter	Description	Required?	Values
maxDataQueryResultSize	Maximum samples that should be configured as part of Query Results	Yes	Integer

Maximum DataQueryResultSize set to 6000

The Tag Rename API

This API allows the administrator to rename tags.

METHOD:	**PUT**
URI	`https://<historianservername>:8443/historian-rest-api/v1/tags/tagrename/oldtagname/newtagname?{truerename}`
SAMPLE URI	`https://<historianservername>:8443/historian-rest-api/v1/tags/tagrename/GDW14NV2E.Simulation0000101/GDW14NV2E.Simulation0000101newname?truerename= <true | false>`
SAMPLE CURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json"-H "Authorization: Bearer <TOKEN> -X PUT https://<historianservername>:8443/historian-rest-api/v1/tags/tagrename/<oldtagname>/<newtagname>?truerename=<true | false>`

Table 104. Query Parameters

Parameter	Description	Required?	Values
oldtagname	Tag which is to be renamed.	Yes	String
newtagname	New name for the selected tag.	Yes	String
truerename	Renames the tag permanently if the value entered is true. Creates an alias if the value entered is false.	Optional (false is default)	Boolean (true or false)

Table 105. Response Parameters

Parameter	Data Type	Required?	Description
Error Code	Integer	Yes	For example, 0.
Error Message	String	Yes	For example, NULL.
Data	List	Yes	Returns all the properties of the tag.

The Write Tag API

Write Tag Data API enables you to create data for tags. You can write data to a tag for different data types such as integer, float, array, multifield and so on. Once created, you can view the data using other end points. Only REST API Administrator and users with write permission can perform this operation.

Method	POST
URI	`https://<historianservername>:8443/historian-rest-api /v1/datapoints/create`
SAMPLE URI	see below

```
https://<historianservername>:8443/
historian-rest-api   /v1/datapoints/create

   Payload

   {
   "TagName": "GDW14NV2E.Simulation00015",
   "samples": [
                {

                "TimeStamp":
 "2019-09-17T15:58:00.000Z",

                "Value":   "1",

                "Quality": 3
                }
           ]
   }
```

Method	POST

SAMPLE RESPONSE	`{`
	`"ErrorCode": 0,`
	`"ErrorMessage": ""`
	`}`

SAMPLE CURL COMMAND	`curl -i -H "Accept: application/json" -i -H "Content-Type: application/json" -H "Authorization: Bearer <TOKEN>” -d “{ \”TagName\”:\”GDW14NV2E.Simulation00015\”, \"samples\":[{\"TimeStamp\":\ "2019-09-17T15:58:00.000Z\",\"Value\": \"1\",\"Quality\": 3}]}” -X POST https://<historianservername>:8443/historian-rest-api/v1/datapoints/create`

Table 106. Query Parameters

Parameter	Description	Required?	Values
Payload	JSON format of Property Name and Property Value.	Yes	Multi-data types. It can have integer, float, array, multifield data types.

Table 107. Response Parameters

Parameter	Data Type	Required?	Description
Error Code	Integer	Yes	For example, ErrorCode = 0, which means the operation was successful.
Error Message	String	Yes	For example, NULL.