webMethods API Gateway Developers Guide

Version 10.15

October 2022

This document applies to webMethods API Gateway 10.15 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2016-2023 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at https://softwareag.com/licenses/.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at https://softwareag.com/licenses/ and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at https://softwareag.com/licenses/ and/or in the root installation directory of the licensed product(s).

Document ID: YAI-DG-1015-20230519

Table of Contents

About this Documentation

This documentation describes how you can upgrade, and migrate to API Gateway.

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Narrowfont	Identifies service names and locations in the format *folder.subfolder.service*, APIs, Java classes, methods, properties.
Italic	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Product Documentation

You can find the product documentation on our documentation website at https://documentation.softwareag.com.

In addition, you can also access the cloud product documentation via https://www.softwareag.cloud. Navigate to the desired product and then, depending on your solution, go to “Developer Center”, “User Center” or “Documentation”.

Product Training

You can find helpful product training material on our Learning Portal at https://knowledge.softwareag.com.

Tech Community

You can collaborate with Software AG experts on our Tech Community website at https://techcommunity.softwareag.com. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software AG news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories at https://github.com/softwareag and https://hub.docker.com/u/softwareag and discover additional Software AG resources.

Product Support

Support for Software AG products is provided to licensed customers via our Empower Portal at https://empower.softwareag.com. Many services on this portal require that you have an account. If you do not yet have one, you can request it at https://empower.softwareag.com/register. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.
- Add product feature requests.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

1 REST APIs in API Gateway

API Gateway Administration

Do not provide values starting with a dot (.), in any of the fields when making API calls from a REST client as Elasticsearch does not support saving those values.

API Gateway provides the capability to API definitions to administer various functions of the API Gateway.

API Gateway provides the following REST API and the resources to manage API Gateway configuration:

- **GET/rest/apigateway/quiescemode** : Retrieves the quiesce mode setting in API Gateway.
- **PUT/rest/apigateway/quiescemode** : Enables or disables the quiesce mode in API Gateway. Quiesce mode has two block types - `designtime` and `all`. Quiesce mode for `designtime` blocks all the design time API requests to API Gateway server and returns the 503 status code except the GET HTTP method as well as few white-listed APIs like the search API and this API. Quiesce mode for the block type all is an extension of Integration server's Quiesce mode with the addition of flushing of API Gateway in-memory data such as performance metrics, license metrics, and subscription quota to the configured data store. For details about quiesce mode, see *webMethods API Gateway Upgrade and Migration*.
- **GET/rest/apigateway/rule** : Retrieves list of all configured rules in API Gateway.
- **POST/rest/apigateway/rule** : Creates a conditional rule in API Gateway. The API request body must contain the payload for the rule.
- **GET/rest/apigateway/rule/{ruleId}** : Retrieves the details of a configured rule in API Gateway.
- **PUT/rest/apigateway/rule/{ruleId}** : Updates the details of a specified configured rule in API Gateway. The API request body must contain the payload for the updated rule.
- **DELETE/rest/apigateway/rule/{ruleId}** : Deletes the specified rule in API Gateway.
- **PUT/rest/apigateway/rule/{ruleId}/activate** : Activate a rule. This request does not require any request body.
- **PUT/rest/apigateway/rule/{ruleId}/deactivate** : Deactivates a rule. This request does not require any request body.
- **GET/rest/apigateway/is/truststore** : Retrieves all available truststores from API Gateway.
- **POST/rest/apigateway/is/truststore** : Creates a truststore in API Gateway.
- **GET/rest/apigateway/is/truststore/{truststoreName}** : Retrieves an existing truststore matching the given name from API Gateway.
- **POST/rest/apigateway/is/truststore/{truststoreName}** : Updates an existing truststore in API Gateway.
- **DELETE/rest/apigateway/is/truststore/{truststoreName}** : Deletes an existing truststore in API Gateway.

- **GET/rest/apigateway/licenseUsageDetails**: Retrieves the detailed usage information for the transaction based license. The retrieved information contains the maximum number of invocations that is allowed for the current month, the total number of invocations used, and the remaining number of invocations available for the month.

- **GET/rest/apigateway/is/proxyBypass** : Retrieves a list of all host lists for which outbound proxy servers are skipped. Note that the proxyBypass Id is always `proxyBypass`.

- **POST/rest/apigateway/is/proxyBypass** : Updates the proxyBypassAddresses to bypass the outbound proxy servers. The API request body must contain the payload. In the proxyBypassAddresses field, type the fully qualified host and domain name of each server to which you want the Integration Server to issue requests directly. Type the host name and the domain name exactly as they appear in the URLs the server uses. To enter multiple names, separate each with commas. You can use the asterisk (*) to identify several servers with similar names. The asterisk matches any number of characters. For example, if you want to bypass requests made to localhost, www.yahoo.com, home.microsoft.com, and all hosts whose names begin with NYC, you would type: `localhost,www.yahoo.com,home.microsoft.com,NYC*.*`.

- **PUT/rest/apigateway/is/proxyBypass** : Creates the proxyBypassAddresses to bypass the outbound proxy servers.

- **GET/rest/apigateway/portalGateways** : Retrieves API Portal configurations available in API Gateway.

- **POST/rest/apigateway/portalGateways** : Creates API Portal configuration in API Gateway.

- **GET/rest/apigateway/portalGateways/{portalGatewayId}** : Retrieves an API Portal configuration in API Gateway.

- **PUT/rest/apigateway/portalGateways/{portalGatewayId}** : Updates the API Portal configuration in API Gateway.

- **DELETE/rest/apigateway/portalGateways/{portalGatewayId}** : Deletes the API Portal configuration in API Gateway.

- **GET/rest/apigateway/portalGateways/communities**: Retrieves the details about communities in API Portal. An API can be published from API Gateway to any of the communities available in API Portal.

- **GET/rest/apigateway/portalGateways/packages**: Retrieves the details of the published packages that the API is part of.

- **GET/rest/apigateway/is/jmsTriggers** : Retrieves a list of all JMS triggers in API Gateway.

- **PUT/rest/apigateway/is/jmsTriggers** : Updates the JMS trigger in API Gateway.

- **GET/rest/apigateway/is/jmsTriggers/{jmsTriggerId}** : Retrieves the specified JMS trigger in API Gateway.

- **PUT/rest/apigateway/is/jmsTriggers/{jmsTriggerId}/enable** : Enables the specified JMS trigger in API Gateway.

- **PUT/rest/apigateway/is/jmsTriggers/{jmsTriggerId}/disable** : Disables the specified JMS trigger in API Gateway.

- **GET/rest/apigateway/is/jndi**: Retrieves a list of all JNDI configurations in API Gateway.
- **POST/rest/apigateway/is/jndi**: Creates a JNDI configuration in API Gateway. The API request body must contain the payload for the JNDI configuration.
- **PUT/rest/apigateway/is/jndi**: Updates the JNDI configuration in API Gateway.
- **GET/rest/apigateway/is/jndi/{jndiId}**: Retrieves the specified JNDI configuration in API Gateway.
- **DELETE/rest/apigateway/is/jndi/{jndiId}**: Deletes the specified JNDI configuration in API Gateway.
- **GET/rest/apigateway/is/jndi/{jndiId}/test**: Tests the given JNDI configuration in API Gateway.
- **GET/rest/apigateway/is/jndi/template**: Retrieves a list of all JNDI templates in API Gateway.
- **GET/rest/apigateway/is/keystore** : Retrieves all keystores available in API Gateway.
- **POST/rest/apigateway/is/keystore** : Creates a keystore in API Gateway.
- **GET/rest/apigateway/is/keystore/{keyStoreName}** : Retrieves the keystore matching the name specified in API Gateway.
- **POST/rest/apigateway/is/keystore/{keyStoreName}** : Updates an already existing keystore in API Gateway.
- **DELETE/rest/apigateway/is/keystore/{keyStoreName}** : Deletes the keystore matching the name specified in API Gateway.
- **GET/rest/apigateway/is/kerberos** : Retrieves the configured Kerberos settings from API Gateway.
- **PUT/rest/apigateway/is/kerberos** : Persists the configured Kerberos settings in API Gateway.
- **GET/rest/apigateway/apitransactions/archives** : Retrieves the details of existing archive files and response of this method would be the list of archive file names. You can select one of the archive file names returned by this method and use the POST /apitransactions/archives/{fileName} method to restore.
- **POST/rest/apigateway/apitransactions/archives** : Archives the runtime events and metrics. You can additionally scope the archive data using input parameter filters. This method returns the job id as the response which is used to know the status of the job.
- **POST/rest/apigateway/apitransactions/archives/{fileName}** : Restores the runtime data of the archive file that is specified. This method returns the job id as a response to track the status further.
- **GET/rest/apigateway/approvals/{approvalId}** : Retrieves an approval request based on the approvalId.
- **DELETE/rest/apigateway/approvals/{approvalId}** : Deletes an approval request based on the approvalId.
- **GET/rest/apigateway/approvals** : Retrieves all approval requests pending for the user.

- **PUT/rest/apigateway/approvals/{approvalId}/{action}** : Creates an approval request for the specified action.
- **GET/rest/apigateway/apitransactions/typedefinitions** : Retrieves the list of runtime event types. The available event types are transactionalEvents, monitorEvents, errorEvents, performanceMetrics, threatProtectionEvents, lifecycleEvents, and policyViolationEvents. You can use these eventType to scope the archive or purge operation.
- **GET/rest/apigateway/is/jmsConnections** : Retrieves a list of all the JMS connections in API Gateway.
- **POST/rest/apigateway/is/jmsConnections** : Creates a JMS connection in API Gateway. The API request body must contain the payload for the JMS connection.
- **PUT/rest/apigateway/is/jmsConnections** : Updates the JMS connections in API Gateway.
- **GET/rest/apigateway/is/jmsConnections/{jmsConnId}** : Retrieves the specified JMS connection in API Gateway.
- **DELETE/rest/apigateway/is/jmsConnections/{jmsConnId}** : Deletes the JMS connection based on the JMS connection ID that is specified in the path.
- **PUT/rest/apigateway/is/jmsConnections/{jmsConnId}/enable** : Enables the specified JMS connection in API Gateway.
- **PUT/rest/apigateway/is/jmsConnections/{jmsConnId}/disable** : Disables the specified JMS connection in API Gateway.
- **GET/rest/apigateway/licenseNotifications** : Retrieves the latest notification issued for a transaction based license.
- **GET/rest/apigateway/approvalConfigurations**: Retrieves a list of available approval configurations in API Gateway.
- **POST/rest/apigateway/approvalConfigurations**: Creates an approval configuration in API Gateway.
- **GET/rest/apigateway/approvalConfigurations/{id}**: Retrieves the details of a specified approval configuration in API Gateway.
- **PUT/rest/apigateway/approvalConfigurations/{id}**: Updates the details of a specified approval configuration in API Gateway.
- **DELETE/rest/apigateway/approvalConfigurations/{id}**: Deletes the specified approval configuration in API Gateway.
- **POST/rest/apigateway/migration**: Triggers a migration action and immediately returns a 202 status code. The clean action clears the data from the API Data store, the reindex action re-indexes the data from the source Elasticsearch to API Data store, and the transform action transforms the re-indexed assets in the API Data store to be compatible with the current API Gateway version. The clean action should be invoked on target API Gateway server prior to invoking the reindex API for core indices. The current status of the action can be retrieved using /migration/status API. A webhook event with the migration status also would be sent to the subscribed webhook clients.

- **GET/rest/apigateway/migration/status**: Retrieves the current status of the migration action which is invoked in API Gateway.
- **GET/rest/apigateway/masterPassword**: Retrieves the master password properties in API Gateway.
- **PUT/rest/apigateway/masterPassword/setExpiry**: Updates the expiry interval of the master password in API Gateway.
- **PUT/rest/apigateway/masterPassword/update**: Updates the master password in API Gateway. On successful update, all the old passwords available will be encrypted using this new master password.
- **PUT/rest/apigateway/masterPassword/reset**: Resets the master password to the default value in API Gateway. This should be performed when the master password is lost and after a successful reset, Software AG recommends to change the master password again to a secure value.
- **GET/rest/apigateway/is/outboundproxy** : Retrieves the list of all available outbound proxy server aliases in API Gateway.
- **POST/rest/apigateway/is/outboundproxy** : Creates the outbound proxy server alias in API Gateway.
- **PUT/rest/apigateway/is/outboundproxy** : Updates the outbound proxy server alias in API Gateway.
- **DELETE/rest/apigateway/is/outboundproxy/{outboundproxyAlias}** : Deletes the specified outbound proxy server alias from API Gateway.
- **PUT/rest/apigateway/is/outboundproxy/{outboundproxyAlias}/enable**: Enables an already existing outbound proxy server alias in API Gateway.
- **PUT/rest/apigateway/is/outboundproxy/{outboundproxyAlias}/disable**: Disables an already existing outbound proxy server alias in API Gateway.
- **GET/rest/apigateway/is/license** : Retrieves the license details from API Gateway.
- **PUT/rest/apigateway/is/license** : Updates the license details in API Gateway.
- **GET/rest/apigateway/logAggregation/downloadLogs** : Downloads logs from different components used by API Gateway, server configurations, and thread dumps.
- **GET/rest/apigateway/webhooks** : Retrieves the list of all webhooks in API Gateway.
- **POST/rest/apigateway/webhooks** : Creates a webhook in API Gateway. The API request body must contain the payload of the webhook that needs to be saved.
- **GET/rest/apigateway/webhooks/{id}** : Retrieves the details of a webhook in API Gateway.
- **PUT/rest/apigateway/webhooks/{id}** : Updates the details of a specific webhook in API Gateway. The API request body must contain the payload of the webhook that needs to be updated.
- **DELETE/rest/apigateway/webhooks/{id}** : Deletes a webhook resource from API Gateway.

- **GET/rest/apigateway/apitransactions/jobs/{jobId}** : Retrieves the status of a specific job. This method returns the status and file name (in case of archive process) as a response.
- **GET/rest/apigateway/apitransactions/jobs** : Retrieves a list of pending jobs. Every time you initiate archive, restore or purge process you get the job id as a response. You can use the specific job id to query the status of the initiated operation.
- **GET/rest/apigateway/configurations/loadBalancer**: Retrieves information about the load balancer configured.
- **PUT/rest/apigateway/configurations/loadBalancer**: Updates the load balancer configuration information.
- **GET/rest/apigateway/configurations/whiteListingIPs**: Retrieves the details of the whitelisting IPs configuration in API Gateway.
- **PUT/rest/apigateway/configurations/whiteListingIPs**: Updates the details of the whitelisting IPs configuration in API Gateway.
- **GET/rest/apigateway/configurations/settings**: Retrieves the list of the extended settings watt properties from API Gateway.
- **PUT/rest/apigateway/configurations/settings**: Updates or creates a list of the extended settings and watt properties in API Gateway.
- **GET/rest/apigateway/configurations/apiCallBackSettings**: Retrieves the API callback processor settings from API Gateway.
- **PUT/rest/apigateway/configurations/apiCallBackSettings**: Updates or creates API callback processor settings in API Gateway. The user should have Manage general administration configurations privilege to update the API callback processor settings.
- **GET/rest/apigateway/configurations/errorProcessing**: Retrieves the configured error template and the value of the property sendNativeProviderFault, which enables the server to forward the native error as it is.
- **PUT/rest/apigateway/configurations/errorProcessing**: Updates the default error template with any custom templates and the value of the property sendNativeProviderFault.
- **GET/rest/apigateway/configurations/keystore**: Retrieves the details of the default keystore, truststore and alias settings in API Gateway.
- **PUT/rest/apigateway/configurations/keystore**: Updates the details of the default keystore, truststore and alias configurations in API Gateway.
- **GET/rest/apigateway/configurations/gatewayDestinationConfig**: Retrieves the details of the API Gateway destination. API Gateway can publish events and performance metrics data. By default, error events, lifecycle events, policy violation event, and performance data are published to API Gateway.
- **PUT/rest/apigateway/configurations/gatewayDestinationConfig**: Updates the details of the API Gateway destination in API Gateway.

- **GET/rest/apigateway/configurations/auditlogDestinationConfig**: Retrieves the details of the Audit Log destination in API Gateway. Audit log captures the API runtime invocations performed in API Gateway. The audit log data is written to a file or a database based on the configurations. Transactions events are written to the audit log only when the Audit Log is selected as a destination in Log Invocation policy.
- **PUT/rest/apigateway/configurations/auditlogDestinationConfig**: Updates the details of the Audit Log destination in API Gateway.
- **GET/rest/apigateway/configurations/centraSiteDestinationCommunicationConfig** : Retrieves the communication details of the CentraSite destination in API Gateway. API Gateway can publish events and metrics to the configured CentraSite destination.
- **PUT/rest/apigateway/configurations/centraSiteDestinationCommunicationConfig** : Updates the communication details of the CentraSite destination in API Gateway.
- **GET/rest/apigateway/configurations/centraSiteDestinationSNMPConfig** : Retrieves the SNMP details of the CentraSite destination in API Gateway. API Gateway can publish events and metrics to the configured CentraSite destination.
- **PUT/rest/apigateway/configurations/centraSiteDestinationSNMPConfig** : Updates the SNMP details of the CentraSite destination in API Gateway.
- **GET/rest/apigateway/configurations/jdbcDestinationConfig**: Retrieves details of the Database destination in API Gateway. API Gateway can publish events and metrics to the configured database.
- **PUT/rest/apigateway/configurations/jdbcDestinationConfig**: Updates the details of the database destination in API Gateway.
- **GET/rest/apigateway/configurations/desDestinationConfig**: Retrieves details of the Digital Events destination in API Gateway. Digital Event Services (DES) enables API Gateway to communicate by exchanging digital events. Digital events are typed and serialized data structures that are used to convey or record information about the execution of a runtime.
- **PUT/rest/apigateway/configurations/desDestinationConfig**: Updates the details of the Digital Events destination in API Gateway.
- **GET/rest/apigateway/configurations/elasticsearchDestinationConfig**: Retrieves details of the Elasticsearch destination in API Gateway. API Gateway can publish events and metrics to the configured Elasticsearch destination.
- **PUT/rest/apigateway/configurations/elasticsearchDestinationConfig**: Updates the details of the Elasticsearch destination in API Gateway.
- **GET/rest/apigateway/configurations/snmpDestinationConfig**: Retrieves details of the SNMP destination in API Gateway. API Gateway can publish events and metrics to the configured third party SNMP destination.
- **PUT/rest/apigateway/configurations/snmpDestinationConfig**: Updates the details of the SNMP destination in API Gateway.

- **GET/rest/apigateway/configurations/emailDestinationConfig**: Retrieves details of the Email destination in API Gateway. API Gateway can send alerts to the email ID specified either in the Log Invocation, Monitor Performance, Monitor SLA or Traffic Optimization policies through the configured Email destination.
- **PUT/rest/apigateway/configurations/emailDestinationConfig**: Updates the details of the Email destination in API Gateway.
- **GET/rest/apigateway/configurations/apiPortalDestinationConfig**: Retrieves details of the API Portal destination configuration. API Gateway can publish events and performance metrics data. By default, error events, lifecycle events, policy violation event, and performance data are published to API Portal.
- **PUT/rest/apigateway/configurations/apiPortalDestinationConfig**: Updates the details of the API Portal destination in API Gateway.
- **GET/rest/apigateway/configurations/cache**: Retrieves the cache configuration in API Gateway.
- **PUT/rest/apigateway/configurations/cache**: Updates the cache configuration in API Gateway.
- **GET/rest/apigateway/configurations/customContentTypes**: Retrieves the configured custom content types in API Gateway. Custom content types can be defined for base types XML,JSON and Text.These Custom types can be then used for payload processing in policies like Content based routing,Identify and access and Conditional error processing.
- **PUT/rest/apigateway/configurations/customContentTypes**: Updates the configured custom content types in API Gateway. The response is a set of key/value pair where key indicates the custom content type and value indicates the base type. The value can be application/xml or application/json or text/xml.
- **GET/rest/apigateway/configurations/ldapConfig**: Retrieves the LDAP configuration settings configured in API Gateway.
- **PUT/rest/apigateway/configurations/ldapConfig**: Updates the LDAP configuration settings configured in API Gateway.
- **GET/rest/apigateway/configurations/passwordRestrictions**: Retrieves the password restrictions settings configured in API Gateway.
- **PUT/rest/apigateway/configurations/passwordRestrictions**: Saves the password restrictions settings configured in API Gateway.
- **GET/rest/apigateway/configurations/passwordExpiry**: Retrieves the password expiry settings configured in API Gateway.
- **PUT/rest/apigateway/configurations/passwordExpiry**: Saves the password expiry settings configured in API Gateway.
- **GET/rest/apigateway/configurations/denyByIPForFailedAuthConfig**: Retrieves the configuration of global IP access setting for authentication based restrictions in API Gateway.
- **PUT/rest/apigateway/configurations/denyByIPForFailedAuthConfig**: Saves the global IP access setting for authentication based restriction settings in API Gateway.

- **GET/rest/apigateway/configurations/accountLockSettings**: Retrieves the account lock settings configured in API Gateway.
- **PUT/rest/apigateway/configurations/accountLockSettings**: Saves the account lock expiry settings configured in API Gateway.
- **GET/rest/apigateway/configurations/logConfig**: Retrieves the log settings of various components used by API Gateway.
- **PUT/rest/apigateway/configurations/logConfig**: Updates the details of the log configuration in API Gateway.
- **POST/rest/apigateway/assets/owner**: Changes ownership of API Gateway assets.
- **POST/rest/apigateway/assets/team**: Changes the team of API Gateway asset.
- **GET/rest/apigateway/urlaliases**: Retrieves all URL Aliases or a URL Alias with a particular ID in API Gateway (if the query parameter alias is provided).
- **POST/rest/apigateway/urlaliases**: Creates a new URL alias in API Gateway.
- **PUT/rest/apigateway/urlaliases**: Updates an existing URL alias in API Gateway.
- **DELETE/rest/apigateway/urlaliases**: Deletes a URL alias in API Gateway.
- **GET/rest/apigateway/is/cluster** : Retrieves the configured cluster settings from API Gateway.
- **PUT/rest/apigateway/is/cluster** : Updates the cluster settings in API Gateway.
- **GET/rest/apigateway/is/webServiceEndpoints** : Retrieves list of all Webservice endpoints in API Gateway.
- **POST/rest/apigateway/is/webServiceEndpoints** : Creates a Webservice endpoint in API Gateway. The API request body must contain the payload for the Webservice endpoint.
- **PUT/rest/apigateway/is/webServiceEndpoints** : Updates the Webservice endpoint in API Gateway.
- **GET/rest/apigateway/is/webServiceEndpoints/{webServiceEndpointId}** : Retrieves the specified Webservice endpoint in API Gateway.
- **DELETE/rest/apigateway/is/webServiceEndpoints/{webServiceEndpointId}** : Deletes the specified Webservice endpoint in API Gateway.
- **GET/rest/apigateway/apitransactions**: Retrieves the API transactions data. The data to be downloaded is filtered based on the input parameters. The user should be part of API-Gateway-Administrators group or should have Manage purge and restore runtime events privilege to perform this operation.
- **DELETE/rest/apigateway/apitransactions**: Purges the API transactions data and the data to be purged is filtered based on the input parameters. This method returns the job id as response and the job id is used to track the job status.
- **GET/rest/apigateway/configurations/jsonWebToken**: Retrieves the details of the API Gateway JSON Web Token (JWT) configuration. API Gateway can generate a JWT itself or validate the

JWT generated by a trusted third party server. JWT is a JSON-based open standard (RFC 7519) means of representing a set of information to be securely transmitted between two parties. A set of information is the set of claims (claim set) represented by the JWT. A claim set consists of zero or more claims represented by the name-value pairs, where the names are strings and the values are arbitrary JSON values.

- **PUT/rest/apigateway/configurations/jsonWebToken**: Updates the details of the JWT configuration in API Gateway.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayAdministration.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/administration-service/AdministrationService.json.

Alias Management

API Gateway provides the capability to create aliases, retrieve alias information, update alias properties as required, and delete the existing aliases using a REST API.

API Gateway provides the following REST API and the resources to manage aliases:

- **GET/rest/apigateway/alias**: Retrieves the list of all aliases in API Gateway. You can also use this to retrieve details for a particular alias by providing the aliasName.
- **POST/rest/apigateway/alias**: Creates an alias in API Gateway.
- **GET/rest/apigateway/alias/{aliasId}**: Retrieves the details of the specified alias in API Gateway.
- **PUT/rest/apigateway/alias/{aliasId}**: Updates the details of the specified alias in API Gateway.
- **DELETE/rest/apigateway/alias/{aliasId}**: Deletes the specified alias in API Gateway.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayAlias.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/alias-management/AliasManagement.json.

Application Management

API Gateway provides the capability to create applications, retrieve application information, update application properties as required, and delete the existing applications using a REST API. You can use this REST API to register APIs to the application, modify details of the registered APIs for the application, and unregister APIs from the application.

API Gateway provides the following REST API and the resources to manage applications:

- **GET/rest/apigateway/applications**: Retrieves the list of available applications in API Gateway. You can also use this to retrieve details for a particular application by providing the applicationId.
- **POST/rest/apigateway/applications**: Creates an application in API Gateway.
- **DELETE/rest/apigateway/applications**: Deletes the specified application in API Gateway.
- **GET/rest/apigateway/applications/{applicationId}**: Retrieves the details of the specified application in API Gateway.
- **PUT/rest/apigateway/applications/{applicationId}**: Updates the details of the specified application in API Gateway.
- **PATCH/rest/apigateway/applications/{applicationId}**: Suspends the specified application in API Gateway.
- **GET/rest/apigateway/applications/{applicationId}/apis**: Retrieves the list of registered APIs for the specified application in API Gateway.
- **POST/rest/apigateway/applications/{applicationId}/apis**: Registers APIs with the specified application in API Gateway.
- **PUT/rest/apigateway/applications/{applicationId}/apis**: Updates the details of the APIs that are registered with the specified application in API Gateway.
- **DELETE/rest/apigateway/applications/{applicationId}/apis**: Unregisters APIs from the specified application in API Gateway. You can also use this to unregister a particular API by providing the apiIDs.
- **GET/rest/apigateway/strategies**: Retrieves a list of all strategies in API Gateway.
- **POST/rest/apigateway/strategies**: Creates a strategy in API Gateway. The API request body must contain the payload for the strategy.
- **DELETE/rest/apigateway/strategies**: Deletes the specified strategy in API Gateway.
- **GET/rest/apigateway/strategies/{strategyId}**: Retrieves the details of the specified strategy in API Gateway.
- **PUT/rest/apigateway/strategies/{strategyId}**: Updates the details of the specified strategy in API Gateway.
- **PUT/rest/apigateway/strategies/{strategyId}/refreshCredentials**: Refreshes the credentials of the specified strategy in API Gateway.
- **GET/rest/apigateway/applications/{applicationId}/accessTokens**: Retrieves a map of access token endpoints for all the authorization servers configured in API Gateway.
- **POST/rest/apigateway/applications/{applicationId}/accessTokens**: Regenerates the access tokens of an application in API Gateway.
- **PUT/rest/apigateway/applications/{applicationId}/accessTokens**: Updates the access tokens of an application in API Gateway.

- **DELETE/rest/apigateway/applications/{applicationId}/accessTokens**: Deletes the access tokens from a specified application in API Gateway.
- **GET/rest/apigateway/applications/_search**: Retrieves a list of available applications in API Gateway based on the search query parameters.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayApplication.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/application-management/ApplicationManagement.json.

API Gateway Archive

You can import already exported archives of APIs, global policies, and other related assets and re-create them in API Gateway. Each artifact in an archive is associated with a universally unique identifier (UUID) across all API Gateway installations. When importing an archive, the UUID helps in determining whether the corresponding artifact is already available in API Gateway. In such a situation, you can specify whether to overwrite an already existing artifact during the import process.

API Gateway provides the following REST API and the resources to export and import an archive:

- **GET /rest/apigateway/archive**: Retrieves the archive, which is a ZIP file that contains the selected assets and its dependent assets.
- **POST /rest/apigateway/archive**: Imports the API Gateway archive as well as exports the assets as an archive.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayArchive.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/archive-service/ArchiveService.json.

API Gateway Availability

API Gateway provides the capability to monitor the health of API Gateway and report the overall health of API Gateway. Each health check request displays a `status` field as the first entry. The status can have the values `green`, `yellow` or `red` describing the overall status of the components to check. This means that when any of the components signals a problem, then the status is set to `red`.

API Gateway provides the following REST API and the resources to monitor the health of API Gateway:

- **GET /gateway/availability/admin**: Retrieves the availability and health status of the API Gateway administration service (UI, Dashboards, Admin REST API).

- **GET /gateway/availability/engine**: Retrieves the availability and health status of the Gateway policy enforcement engine (ElasticSearch cluster, IS and Terracotta).
- **GET /gateway/availability/externalServices**: Retrieves the availability of external services accessed by API Gateway.
- **GET /gateway/availability/all**: Retrieves the availability of the administration service of the policy enforcement engine and of the external services accessed by API Gateway.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayAvailability.json.

Note:

- To perform the following API Gateway Availability REST calls you must have the *View Administration Configuration* privileges.
  - GET /gateway/availability/externalServices
  - GET /gateway/availability/all
- To perform the following API Gateway Availability REST calls you must be a valid API Gateway user.
  - GET /gateway/availability/admin
  - GET /gateway/availability/engine

You can use the existing health check request GET http://localhost:5555/rest/apigateway/health, without any authentication being set, to retrieve the health of API Gateway that monitors the availability and health status of Kubernetes and Docker containers . This returns a HTTP 200 response without additional data.

Document Management

API Gateway provides the capability to store and manage the documents associated with an API.

API Gateway provides the following REST API and the resources to manage the documents associated with APIs:

- **GET/rest/apigateway/documents/{documentId}**: Retrieves the requested document from API Gateway.
- **PUT/rest/apigateway/documents/{documentId}**: Updates the requested document in API Gateway.
- **DELETE/rest/apigateway/documents/{documentId}**: Deletes the requested document from API Gateway.
- **PATCH/rest/apigateway/documents/{documentId}**: Patches the requested document in API Gateway.
- **POST/rest/apigateway/documents**: Creates and stores the documents in API Gateway.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayDocumentManagement.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/document-mangement-service/DocumentManagementService.json.

Data Center Management

A data center is a facility that shares IT operations and equipment to collect, store, process, and disseminate data and applications in centralized locations. Data centers are an integral part of the enterprise, designed to support business applications and provide services such as data storage, management, backup, and recovery. Hence as part of disaster recovery plan, it is important to deploy multiple data centers in API Gateway.

API Gateway provides the capability to configure data centers, activate data centers in different deployment modes (such as active-active, hot standby, warm, and cold), and switch data centers between different deployment modes.

API Gateway provides the following REST API and the resources to manage the data centers:

- **PUT/rest/apigateway/dataspace/listener**: Configures the GRPC listener in the data center.
- **GET/rest/apigateway/dataspace/listener**: Retrieves the GRPC listener configuration of the associated data center.
- **PUT/rest/apigateway/dataspace/ring**: Configures the data center and establishes the ring configuration with the associated data centers.
- **GET/rest/apigateway/dataspace/ring**: Retrieves the connectivity information of the associated data centers in the ring configuration.
- **PATCH/rest/apigateway/dataspace/ring**: Appends the data center configuration to the ring in API Gateway.
- **PUT/rest/apigateway/dataspace/configure**: Configures multiple data centers and establishes the connection with the associated data centers.
- **PUT/rest/apigateway/dataspace/activate**: Activates a data center configuration in API Gateway.
- **PUT/rest/apigateway/dataspace/activateAll**: Activates multiple data center configuration in API Gateway.

  Use the following query parameters to activate data centers in the required mode:

  - **PUT/rest/apigateway/dataspace/activateAll?mode= ACTIVE_RING**: Activates all the data centers in the active-active mode in API Gateway.
  - **PUT/rest/apigateway/dataspace/activateAll?mode= STANDBY**: Activates all the data centers in the hot standby mode in API Gateway.
  - **PUT/rest/apigateway/dataspace/activateAll?mode= STANDALONE**: Switches the data center from the active-active or hot standby mode to stand alone mode in API Gateway.
- **GET/rest/apigateway/dataspace**: Retrieves the current configuration of the associated data center in API Gateway.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayDataManagement.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/crossdc-management/cross-dc-management-postman-collection.json.

Internal Service

API Gateway provides internal APIs that work on identified applications that are identified based on identifiers such as APi Key, OAuth token, IP address and so on.

API Gateway provides the following REST API and the resources to manage application identification:

- **POST/{apigateway}/security/getJsonWebToken**: Generates JSON Web token with custom claims supplied in the request.
- **POST/{apigateway}/security/exchangeIDToken**: Generate an access token for the given ID Token.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayInternalService.json.

Port Configuration

API Gateway provides the capability to manage port configurations. Each port is associated with a specific type of protocol, HTTP or HTTPS. In addition to these port types, API Gateway also supports the external port, the internal listener port, and the WebSocket listener port. You can specify one or more HTTP or HTTPS ports on which the API Gateway Admin APIs and the deployed APIs are available for consumption. By default, they are available on the primary HTTP port.

API Gateway provides the following REST API and the resources to manage port configuration:

- **GET /rest/apigateway/ports**: Retrieves all port configurations.
- **POST /rest/apigateway/ports**: Creates new port configuration.
- **PUT /rest/apigateway/ports**: Updates an existing port configuration.
- **DELETE /rest/apigateway/ports**: Deletes a port configuration.
- **GET /rest/apigateway/ports/primary**: Retrieves the definition of the primary port.
- **PUT /rest/apigateway/ports/primary**: Sets the primary port to the specified existing port configuration.
- **PUT /rest/apigateway/ports/enable**: Enables the specified port configuration. Only enabled ports can be contacted and can handle server requests.
- **PUT /rest/apigateway/ports/disable**: Disables the specified port configuration. A disabled port cannot be contacted.

- **GET /rest/apigateway/ports/{listenerKey}**: Retrieves the API Gateway port configuration for the specified listener key.
- **GET /rest/apigateway/ports/{listenerKey}/accessMode**: Retrieves the access mode of the API Gateway port configuration for the specified listener key.
- **POST /rest/apigateway/ports/{listenerKey}/accessMode**: Creates an access mode type for the API Gateway port configuration for the specified listener key. You can set the access mode for a port to deny or allow.
- **PUT /rest/apigateway/ports/{listenerKey}/accessMode**: Updates the access mode services of the API Gateway port configuration for the specified listener key. If you want to restrict the allow list, you have to add a PUT call after the POST call.
- **GET /rest/apigateway/ports/{listenerKey}/ipAccessMode**: Retrieves the IP access mode of the API Gateway port configuration for the specified listener key.
- **POST /rest/apigateway/ports/{listenerKey}/ipAccessMode**: Creates the IP access mode type for the API Gateway port configuration for the specified listener key. You can set the IP access mode for a port to deny or allow.
- **PUT /rest/apigateway/ports/{listenerKey}/ipAccessMode**: Updates the IP access mode host list of the API Gateway port configuration for the specified listener key. If you want to restrict the allow list, you have to add a PUT call after the POST call.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayPortManagement.json.

Policy Management

API Gateway provides the capability to retrieve API Gateway policy related data such as policies, parameters, policy stages, policy templates, binding assertion, token assertion and service result cache. You can use this REST API to create, update or delete policies.

API Gateway provides the following REST API and the resources to manage policies:

- **GET/rest/apigateway/denialofservice/deniedIP**: Retrieves the list of denied IPs (IPs that violated the threat protection rules configured).
- **DELETE/rest/apigateway/denialofservice/deniedIP**: Deletes the specified IP from the denied IP list. Once the IP is removed from the list the request from that IP is processed.
- **GET/rest/apigateway/assertions**: Retrieves a list of available assertions in API Gateway.
- **POST/rest/apigateway/assertions**: Creates an assertion in API Gateway. Custom assertions allow the API providers to extend and provide additional security policies that are not available by default in API Gateway. In WS-Security, custom assertions are used for expressing individual security requirements, constraints, or both. The individual policy assertions can be combined to create security policies that ensure secure and reliable exchanges of SOAP messages between a client and a SOAP API.
- **GET/rest/apigateway/assertions/{assertionId}**: Retrieves the specified assertion element.

- **PUT/rest/apigateway/assertions/{assertionId}**: Updates the specified assertion.

- **DELETE/rest/apigateway/tokenAssertion/{assertionId}**: Deletes the specified assertion.

- **GET/rest/apigateway/policyActionTemplates/{policyActionTemplateId}**: Retrieves the template details of the specified policy action.

- **GET/rest/apigateway/policyActionTemplates**: Retrieves all the template detail for list of policy actions. You can also use this to retrieve template details for a particular policy action by providing the policy action template Id.

- **GET/rest/apigateway/policyStages**: Retrieves the list of policy stages available in API Gateway. It also displays the list of policies associated with each stage.

- **GET/rest/apigateway/configurations/mobileApp**: Retrieves the configuration details for the mobile applications for which access has been denied. You can use API Gateway to disable access for certain mobile application versions on a predefined set of mobile platforms. By registering the required devices and applications and disabling access to these versions, you ensure that all users use the latest versions of the applications and take advantage of the latest security and functional updates.

- **PUT/rest/apigateway/configurations/mobileApp**: Updates the details of the mobile applications configuration in API Gateway.

- **GET/rest/apigateway/policyActions**: Retrieves the list of all policy actions from API Gateway. It can also be used to retrieve details for particular set of policy actions by specifying the policy id, policy details for list of policies of a particular policy type.

- **POST/rest/apigateway/policyActions**: Creates policy actions of different types in API Gateway. The result of this request is a policy action payload and is available in the response.

- **GET/rest/apigateway/policyActions/{policyActionId}**: Retrieves the policy action details for a specified policy action based on the id specified in API Gateway.

- **PUT/rest/apigateway/policyActions/{policyActionId}**: Updates the policy action details for a specified policy action based on the id specified in API Gateway.

- **DELETE/rest/apigateway/policyActions/{policyActionId}**: Deletes the policy action based on the id specified in API Gateway.

- **GET/rest/apigateway/policies**: Retrieves the list of all policies from API Gateway. It can also be used to retrieve details for particular set of policies by specifying the policy id, policy details for list of policies of a particular policy type.

- **POST/rest/apigateway/policies**: Creates policies of different types in API Gateway. You can also use this to clone policies.

- **GET/rest/apigateway/policies/{policyId}**: Retrieves the policy details for a specified policy in API Gateway. If policy id is available then the policy details is sent in response.

- **PUT/rest/apigateway/policies/{policyId}**: Updates the policy details for a specified policy in API Gateway. For Global policy user should have API Gateway administrator access to update global policy.

- **DELETE/rest/apigateway/policies/{policyId}**: Deletes the specified policy in API Gateway. This request will automatically delete the associated policy action for this policy.
- **GET/rest/apigateway/policies/{policyId}/apis**: Retrieves the list of applicable APIs for a global policy. An API become applicable API for a global policy only if it satisfies the scope specified in the global policy. By default it will return the basic API details of all the applicable APIs either if the API is active or inactive for a global policy.
- **GET/rest/apigateway/policies/{policyId}/conflicts**: Retrieves the conflicts for the specified global policy.
- **PUT/rest/apigateway/policies/{policyId}/activate**: Activates the specified global policy. This request does not require any request body. This request tries to activate the global policy and if any error occurs during activation it is reported as response or if the global policy is activated then its policy details active flag set to true is sent as response. If the global policy has any conflicts then it cannot be activated and the conflicts are manually resolved.
- **PUT/rest/apigateway/policies/{policyId}/deactivate**: Deactivates the specified global policy. This request does not require any request body. This request tries to deactivate the global policy and if any error occurs during deactivation it is reported as response or if the global policy deactivated the policy details of a global policy with active flag set to false is sent as response. An active global policy cannot have conflicts with other active global policy and hence the deactivation fails only when the conflict occurs between active global policy that is specified and one or more applicable active APIs. This can happen when the applicable active API policy action depends on one or more policy action from the specified global policy. If you deactivate this policy, it would cause the active API to have an unstable state. Hence the deactivation is reported as failed in this case.
- **PUT/rest/apigateway/policies/{policyId}/disable**: Disables the Threat protection policy created in API Gateway. This request does not require any request body. If the threat protection policy is disabled successfully then the policy details of specified policy will be sent as response.
- **PUT/rest/apigateway/policies/{policyId}/enable**: Enables the Threat protection policy created in API Gateway. This request does not require any request body. If the threat protection policy is enabled successfully then the policy details of specified policy is sent as response.
- **PUT/rest/apigateway/policies/{policyId}/movedown**: Moves down the execution order of the Threat protection policy created in API Gateway.
- **PUT/rest/apigateway/policies/{policyId}/moveup**: Moves up the execution order of the Threat protection policy created in API Gateway.
- **GET/rest/apigateway/serviceResultCache/{apiId}**: Retrieves the Service Result Cache size for the specified API accessed using the API Id.
- **DELETE/rest/apigateway/serviceResultCache/{apiId}**: Deletes the Service Result Cache for the specified API accessed using the API Id.
- **GET/rest/apigateway/serviceResultCache**: Retrieves the Service Result Cache size for the specified API accessed using apiName and apiVersion.
- **DELETE/rest/apigateway/serviceResultCache**: Deletes the Service Result Cache for the specified API accessed using apiName and apiVersion.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayPolicyManagement.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/policy-management/PolicyManagement.json.

Promotion Management

API Gateway provides supports staging and promotion of assets. Staging and promotion allows you to promote all the assets across different stages.

API Gateway provides the following REST API and the resources to manage staging and promotion:

- **GET/rest/apigateway/promotion**: Retrieves the promotions history with each promotion entry providing the details such as promotion name, promoted by whom, when it is promoted, and the promoted assets status.
- **POST/rest/apigateway/promotion**: Promote the API Gateway assets from the source machine to destination machine where the destination machine is configured as a stage.
- **GET/rest/apigateway/promotion/{promotionId}**: Retrieves a promotion based on the promotion Id.
- **DELETE/rest/apigateway/promotion/{promotionId}**: Deletes a promotion based on the promotion Id.
- **GET/rest/apigateway/stages**: Retrieves all the configured stages.
- **POST/rest/apigateway/stages**: Configures a stage in the source API Gateway where promotion is initiated.
- **GET/rest/apigateway/stages/{stageId}**: Retrieves a particular stage object based on a stage Id.
- **PUT/rest/apigateway/stages/{stageId}**: Updates a particular stage in the source API Gateway where the promotion is initiated.
- **DELETE/rest/apigateway/stages/{stageId}**: Deletes a particular stage.
- **GET/rest/apigateway/rollback**: Retrieves the list of possible rollbacks from the local (target) API Gateway instance.
- **GET/rest/apigateway/rollback/{rolbackId}**: Retrieves a rollback based on the rollback Id.
- **PUT/rest/apigateway/rollback/{rolbackId}**: Rolls back the assets to the previous state, That is, the state prior to promotion. Rollback should be initiated from the local API Gateway instance.
- **DELETE/rest/apigateway/rollback/{rolbackId}**: Deletes the rollback.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayPromotionManagement.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/promotion-management/PromotionManagement.json.

Public Services

This API allows you to fetch a JWT from API Gateway and also fetch JSON Web key URI of API Gateway.

API Gateway provides the following REST API and the resources to manage public services:

- **GET/rest/pub/apigateway/jwt/getJsonWebToken**: Retrieves JWT from API Gateway. To obtain the JWT from API Gateway the client has to pass the basic authentication credentials.
- **GET/rest/pub/apigateway/jwt/certs**: Retrieves all the public keys of API Gateway, which can be used to validate the JWT generated by API Gateway.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayPublicServices.json.

API Gateway Search

The API Gateway search API allows you to execute a search query in API Gateway and retrieve search results that match the search query.

Remember:
When your search involves a large number of records, the process consumes a considerable memory space from the server, which in turn affects other business transactions. Hence, Software AG recommends that you perform large search operations when you expect lesser business transactions so that the regular business is not affected.

API Gateway provides the following REST API resources:

- **POST/rest/apigateway/search**: Executes a search query in API Gateway and returns the results that match your query. You can perform search across the different objects such as APIs, Applications, Aliases, Assertions, Policies, Administration Settings, Policy properties, Packages, Plans, Subscriptions, Users, User groups, Transactional events, Lifecycle events, Policy violation events, Monitor events, Error events, Threat protection events, and Performance metrics.

  To perform a search operation, specify the following in your REST request:

REST Request Section	Description
Types	Objects for which you want to perform the search operation. You can specify one or more of the listed objects. **Note:**

REST Request Section	Description
	When you specify `Users` and `User Groups` in the **Types** section to return the list of users and user groups from Integration Server respectively, you need not specify any search criteria.
Scope	Search Criteria. You must specify your search attribute and a keyword (value of the attribute) or one of the following as your search criteria: ■ **Time range** - to retrieve results for a date range (from and to values), from a specified date to current date, till a specified date, or since the given amount of time (seconds, minutes, hours, days, weeks, months, quarters, or years). ■ **Value range** - to retrieve results for a integer value range (from and to values), from a given value to the maximum value, and from 0 to the given value. You can specify multiple attributes in this section. **Note:** The search operation is performed based on the search criteria specified in this section for all objects specified in the **Types** section.
Condition	One of the following: ■ **and** - to return results that match all search criteria. ■ **or** - to return results that match any of the given criteria.
Fields	Fields to be returned in the response. You can specify only the required fields, instead of viewing all fields in your response. That is, if you want to view only the API Names and Versions that match your search criteria, you can specify `apiName` and `apiVersion` in this section of your REST request.

- **POST/rest/apigateway/search/_count**: Retrieves the total number of records for the specified scope and types.

  To retrieve the count of records, you can specify the required types and scope similar to the **/search** query. If you do not specify any search criteria in the **Scope** section, then the query returns total number of assets for the objects specified in the **Types** section.

- **POST/rest/apigateway/search/_aggregations**: Executes a search query and groups the results for the specified scope and types.

  To perform an aggregations search, specify the following in your REST request:

REST Request Section	Description
Types	Objects for which you want to perform the search operation. You can specify one or more of the listed objects.
Scope	Search Criteria. You must specify your search attribute and a keyword (value of the attribute) or one of the following as your search criteria: ■ **Time range** - to retrieve results for a date range (from and to values), from a specified date to current date, till a specified date, or since the given amount of time (seconds, minutes, hours, days, weeks, months, quarters, or years). ■ **Value range** - to retrieve results for a integer value range (from and to values), from a given value to the maximum value, and from 0 to the given value.
Condition	One of the following: ■ **and** - to return results that match all search criteria. ■ **or** - to return results that match any of the given criteria.
Aggregations	Values for the following: ■ **Name** - Specify a name used to group the required results. For example, you can specify `Info by API`, if you are grouping the results by APIs. ■ **Type** - One of the following: ■ **group** - to group the results based on the given fields. ■ **timeseries** - to group results based on a given interval value. The interval can be seconds, minutes, hours, days, weeks, months, quarters, or years. ■ **metrics** - to find the average, minimum, maximum and sum of given fields. ■ **Fields** - Fields to be considered for the aggregation. If the type is group and there are multiple fields, separate the field names with commas.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewaySearch.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/search-service/SearchService.json.

Note:

The number of transactions returned for a search is based on the value specified in the **defaultSearchSize** extended setting. If your search result exceeds the value of this setting, then you can navigate through your search results by specifying the range of records that you want to view. For example, the value specified in the **defaultSearchSize** setting is *1000* and the count of your search result is 5000, then only the first 1000 records are displayed. To view the consequent records, you can specify the number of the record from which you want to view, and the number of records that must be displayed. That is, to view the records from 1001 to 2000, you can specify the range as follows:

```
POST http://localhost:5555/rest/apigateway/search
{
   "types": [
        "TRANSACTION_EVENTS"    ],
    "scope": [
        {  "attributeName": "responseCode",
            "keyword": "304"
        },
      ],
    "from": "1001"
    “size”: “1000”
}
```

Server Information

API Gateway provides the capability to retrieve API Gateway server information.

API Gateway provides the following REST API and the resources to retrieve the server information:

- **GET/rest/apigateway/is/serverinfo**: Retrieves API Gateway server information.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayServerInfoSwagger.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/server-information/ServerInformation.json.

Service Management

API Gateway provides the capability to retrieve and manage all APIs in API Gateway and the related information such as applications associated, scopes, versions and so on.

API Gateway provides the following REST API and the resources to manage services:

- **GET/rest/apigateway/apis/{apiId}**: Retrieves an API based on the apiId specified.
- **PUT/rest/apigateway/apis/{apiId}**: Updates an API by importing a file, URL or inline based on the apiId specified.
- **DELETE/rest/apigateway/apis/{apiId}**: Deletes an API based on the apiId specified.
- **PUT/rest/apigateway/apis/{apiId}/activate**: Activates an API so that the API is exposed to consumers.

- **PUT/rest/apigateway/apis/{apiId}/deactivate**: Deactivates an API so that the API is not exposed to consumers.
- **PUT/rest/apigateway/apis/{apiId}/publish**: Publishes API to the registered API Portal.
- **PUT/rest/apigateway/apis/{apiId}/unpublish**: Unpublishes an API from the registered API Portal.
- **PUT/rest/apigateway/apis/{apiId}/mock/enable**: Enables you to mock an API by simulating the native service.
- **PUT/rest/apigateway/apis/{apiId}/mock/disable**: Disables the mocking capability to mock an API.
- **PUT/rest/apigateway/apis/{apiId}/tracing/enable**: Enables tracing for an API.
- **PUT/rest/apigateway/apis/{apiId}/tracing/disable**: Disables the tracing capability to trace an API.
- **POST/rest/apigateway/tracer/archive**: Creates an archive of the tracer events.
- **POST/rest/apigateway/tracer/import**: Imports the traced data from the archive. This API does not import the events in to the storage. It simply reads the archive and returns all the events and their tracing data in the archive.
- **GET/rest/apigateway/tracer/{correlationID}**: Retrieves trace information for an API invocation event specified by its correlationID.
- **GET/rest/apigateway/apis**: Retrieves all APIs or subset of APIs based on the apiIds specified.
- **POST/rest/apigateway/apis**: Creates an API as specified. You can create an API by importing a file, URL, or from scratch.
- **DELETE/rest/apigateway/apis**: Deletes APIs based on the apiIds specified.
- **GET/rest/apigateway/apis/{apiId}/applications**: Retrieves the list of registered applications of an API.
- **GET/rest/apigateway/apis/{apiId}/source**: Retrieves the source file along with the root file name that was used while creating an API.
- **GET/rest/apigateway/apis/{apiId}/globalPolicies**: Retrieves the list of active global policies applicable for the specified API.
- **GET/rest/apigateway/apis/{apiId}/versions**: Retrieves all versions of the specified API.
- **POST/rest/apigateway/apis/{apiId}/versions**: Creates a new version of an API and retains applications if required.
- **GET/rest/apigateway/apis/{apiId}/scopes**: Retrieves the scopes for the specified API.
- **GET/rest/apigateway/apis/{apiId}/scopes/{scopeName}**: Retrieves the scopes for the specified API based on the scope name.

- **PUT/rest/apis/{apiId}/implementation**: Updates the API in API Gateway after its implementation by any API provider tool. This is used by API provider tools to update the API after implementing from their end.
- **GET/rest/apis/{apiId}/providerspecification**: Downloads the provider specification of REST and SOAP based APIs. Provider specification is nothing but, the specification file (in swagger or wsdl format) with out the concrete API Gateway endpoint and contains all resources, methods, and operations irrespective of whether their exposure to consumer.
- **PUT/rest/apigateway/serviceRegistry/unpublish**: Unpublishes one or more APIs from one or more service registries.
- **GET/rest/apigateway/serviceRegistry/publish**: Retrieves the service registry publish information for the API.
- **PUT/rest/apigateway/serviceRegistry/publish**: Publishes one or more APIs from one or more service registries.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayServiceManagement.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/service-management/ServiceManagement.json.

Transaction Data

API Gateway provides the capability to query the API transactions. API Transactions are generated (as events) every time an API invocation happens. API Transactions may contain the details about the invocation such as request and response headers, request and response payloads, consumer applications and so on. API Provider may choose to store these events to one or more destinations by using Log Invocation Policy. API Gateway provides different destination options to the API Provider (like API Gateway's own data store, relational databases, Elasticsearch, and so on) where the events can be stored. By default, API Gateway is chosen as a storage destination for these events. This REST API queries for the transactions data only from the API Gateway's default datastore. There are multiple use cases where you can use this transactions data. For instance, you can integrate this API with your billing system wherein this transactional data can be used to compute the usage history of your API for different consumers for monetization usecases. In other scenarios, the data extracted from this service can be used for custom report generation.

You can search for other events using the API Gateway Search API. For more details, see "API Gateway Search" on page 29.

API Gateway provides the following REST API and the resources to retrieve the transaction events data:

- **GET/rest/apigateway/transactionEvents/_search**: Retrieves the transaction events for a given API, Application, Plan or Package for a specific period of time. Multiple request parameters of this method provide options to specify the request criteria to match the expected result and most of these input parameters support regular expression in their values. Along with the mandatory parameters, fromDate and toDate, any one of the other filter criteria should be passed in the request.

- **GET/rest/apigateway/transactionEvents/_count**: Retrieves the number of transaction events for a given API, Application, Plan or Package for a specific period of time. Multiple request parameters of this method provide options to specify the request criteria to match the expected result and most of these input parameters support regular expression in their values. Along with the mandatory parameters , fromDate and toDate, any one of the other filter criteria should be passed in the request.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayTransactionalEvent.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/transaction-data-service/TransactionDataService.json.

User Management

API Gateway provides the capability to manage Users, Groups and Access profiles in API Gateway.

API Gateway provides the following REST API and the resources to retrieve the User ACL list:

- **GET/rest/apigateway/accessProfiles**: Retrieves a list of all access profiles in API Gateway.
- **POST/rest/apigateway/accessProfiles**: Creates an access profile in API Gateway. The API request body must contain the payload for the access profile.
- **GET/rest/apigateway/accessProfiles/{accessProfileId}**: Retrieves the details of an access profile in API Gateway.
- **PUT/rest/apigateway/accessProfiles/{accessProfileId}**: Updates the details of a specified access profile in API Gateway. The API request body must contain the payload for the updated access profile.
- **DELETE/rest/apigateway/accessProfiles/{accessProfileId}**: Deletes an access profile from API Gateway.
- **GET/rest/apigateway/groups**: Retrieves list of all groups in API Gateway.
- **POST/rest/apigateway/groups**: Creates a group in API Gateway. The API request body must contain the payload for the group.
- **GET/rest/apigateway/groups/{groupId}**: Retrieves the details of a group in API Gateway.
- **PUT/rest/apigateway/groups/{groupId}**: Updates the details of a specified group in API Gateway. The API request body must contain the payload for the updated group.
- **DELETE/rest/apigateway/groups/{groupId}**: Deletes a group from API Gateway.
- **GET/rest/apigateway/users**: Retrieves list of all users in API Gateway.
- **POST/rest/apigateway/users**: Creates an user in API Gateway. The API request body must contain the payload for the user.
- **GET/rest/apigateway/users/{userId}**: Retrieves the details of an user in API Gateway.

- **PUT/rest/apigateway/users/{userId}**: Updates the details of a specified user in API Gateway. The API request body must contain the payload for the updated user.
- **DELETE/rest/apigateway/users/{userId}**: Deletes the a specified user in API Gateway.
- **POST/rest/apigateway/users/authenticate**: Authenticates a user in API Gateway.
- **GET/rest/apigateway/installedLanguages**: Retrieves list of installed language packs in API Gateway.
- **GET/rest/apigateway/is/lockedAccounts**: Retrieves the locked user accounts in API Gateway.
- **POST/rest/apigateway/is/lockedAccounts**: Unlocks the locked user accounts by API Gateway.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayUserManagementSwagger.json.

For details about sample payloads, import Postman collection from the following link in Postman client: https://github.com/SoftwareAG/webmethods-api-gateway/blob/master/apigatewayservices/postmancollections/apis/user-management/UserManagement.json.

Subscription Management

You can manage subscriptions from the REST API provided by API Gateway. This API allows you to create application, view applications, get the application details for a specific package and plan, and so on. Alternatively, you can also use API Portal to manage subscriptions. To use the subscription APIs, you must have the manage application permission.

API Gateway provides the following REST API and the resources to manage subscriptions:

- **POST/rest/apigateway/subscriptions**. Creates a subscription and generates an audit log event. The newly generated event is returned. If the approval is enabled, the application details are returned without the API key. Once the request is approved, user can get subscription details and can view the access key. If the approval is not enabled, then the response contains all the application details, except for the API key. The API key is masked and only the requester can view it.
- **PUT/rest/apigateway/subscriptions/{applicationId}**. Updates the subscription details. You can change the package and plan of a subscription. This API can be used only to update the package and plan details.
- **GET/rest/apigateway/subscriptions**. Retrieves the subscriptions created as applications. The API key is masked for all the subscriptions.
- **GET/rest/apigateway/subscriptions/{applicationId}**. Retrieves the details of a specific application. You must provide the application ID as input parameter.
- **GET/rest/apigateway/subscriptions?packageId={packageId}&planId={planId}**. Retrieves the application details for a specific combination of package and plan.
- **GET/rest/apigateway/subscriptions/usage**. Retrieves the subscription usage details of all the subscriptions for the current cycle of only the existing subscriptions.

- **GET/rest/apigateway/subscriptions/{applicationId}/usage**. Retrieves the usage details for a specific application. You must provide the application ID of the required application, as an parameter.

  GET/rest/apigateway/subscriptions/usage?name={applicationName}& package={packageName}&plan={planName}&from={startingIndexOfSearchResult}& size={numberOfRecordsToFetch}&count={boolean}

  . Retrieves the usage details for a specific application's package and plan. The package name, application name, and plan name are given as input parameters. The from, size, and, count parameters are optional. If you provide the from and sum parameters, the values specified in the from and number of records, specified in the size are fetched. If you set the count parameter to true, the API returns number of records for specified query parameter.

- **DELETE/rest/apigateway/subscriptions/{applicationId}**. Deletes an application. You must provide the application ID of the application to be deleted.

For details about the REST API, see https://github.com/SoftwareAG/webmethods-api-gateway/blob/10.15/apigatewayservices/APIGatewayApplication.json.

Backward compatibility support for REST APIs

All the REST APIs in API Gateway are backward compatible. The backward compatibility handles payload transformation from the previous version to the current version of API Gateway. If you want to use version specific payload then use the corresponding endpoint. For example, if you want to use the 10.1 payload to create an asset, then you have to use `http://`hostname:port`/rest/apigateway/v101/`asset.

With the backward compatibility support, API Gateway exposes the following REST end points with the version number mentioned.

- `http://`hostname:port`/rest/apigateway/v101/`assetsspecificURI

  Use this URI if you want to access the latest API Gateway with 10.1 version specific request and response.

  The following policies have conflicting behavior compared to earlier versions:

  - In 10.1 invokeESB templateKey is used to create Invoke webMethods IS policy that can be used for both request and response transformation stage. From 10.2 version, the invokeESB templateKey is changed to requestInvokeESB and responseInvokeESB for request and response transformation stage respectively. So when you send a payload with older version (10.1), it is not possible to create the correct policy in latest version. To solve this, you have to update the payload, and send the appropriate templateKey in 10.1 payload. For example if you are creating invoke webMethods IS policy for request transformation, then you have to specify requestInvokeESB templateKey instead of invokeESB templateKey.

  - In 10.1 xsltTransformation is used to create XSLT Transformation policy that can be used for both request and response transformation stage. From 10.2 version, the xsltTransformation templateKey is changed to requestTransformation and responseTransformation for request and response transformation stage respectively. To solve this, you have to update the payload, and send the appropriate templateKey in 10.1

payload. For example if you are creating XSLT Transformation policy for request transformation, then you have to specify requestTransformation templateKey instead of xsltTransformation templateKey.

- `http://`hostname:port`/rest/apigateway/v102/`assetsspecificURI

  Use this URI if you want to access the latest API Gateway with 10.2 version specific request and response.

- `http://`hostname:port`/rest/apigateway/v103/`assetsspecificURI

  Use this URI if you want to access the latest API Gateway with 10.3 version specific request and response.

- `http://`hostname:port`/rest/apigateway/v105/`assetsspecificURI

  Use this URI if you want to access the latest API Gateway with 10.5 version specific request and response.

- `http://`hostname:port`/rest/apigateway/`assetsspecificURI

  When there is no version mentioned, the URI, by default, accesses the latest version specific request and response.

Note:
The archive REST endpoint to export assets does not give a version specific archive. It always gives the archive with latest version regardless of the version specified in the REST endpoint.

Troubleshooting Tips: REST APIs

I see error when I search API using the POST/rest/apigateway/search REST API

When I search API using the POST/rest/apigateway/search REST API in a clustered environment, I see the following error message in the server log:

019-05-14 12:37:29 CEST [YAI.0300.0014I] [default][node1.kirsa.pl] Error while retrieving Documents for Index gateway_default, Type apis. Cause: org.apache.http.ContentTooLongException: entity content is too long [105063337] for the configured buffer limit [104857600]

This might be due to insufficient response payload size.

Resolution:

Increase the response payload size by setting the *pg.gateway.elasticsearch.client.http.response.size* property value to a higher value in the *config.properties* file at the `*SAG_Install_Directory*\IntegrationServer\instances\*default*\packages\WmAPIGateway\config\resources\elasticsearch` location and restart the API Gateway for the settings to take effect.