Alibaba Cloud
Apsara Stack Enterprise

User Guide - Middleware and Enterprise Applications

Version: 1901

Issue: 20190528

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.
2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.
3. The content of this document may be changed due to product version upgrades, adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.
4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults" and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.
5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified,

reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Document conventions

Style	Description	Example
	A danger notice indicates a situation that will cause major system changes , faults, physical injuries, and other adverse results.	 **Danger:** Resetting will result in the loss of user configuration data.
	A warning notice indicates a situation that may cause major system changes , faults, physical injuries, and other adverse results.	 **Warning:** Restarting will cause business interruption. About 10 minutes are required to restart an instance.
	A caution notice indicates warning information, supplementary instructions , and other content that the user must understand.	 **Note:** If the weight is set to 0, the server no longer receives new requests.
	A note indicates supplemental instructions, best practices, tips, and other content.	 **Note:** You can use **Ctrl** + **A** to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings** > **Network** > **Set network type**.
Bold	Bold formatting is used for buttons, menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands.	Run the `cd /d C:/window` command to enter the Windows system folder.
`Italic`	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *`Instance_ID`*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig` *`[-all|-t]`*
{} or {a|b}	This format is used for a required value , where only one item can be selected.	`switch` *`{active|stand}`*

Contents

1 API Gateway

1.1 Product overview

API gateway is a complete API hosting service. It helps you use APIs to provide capabilities, services, and data to your partners. You can also publish APIs to the API marketplace for other developers to purchase and use.

- API Gateway provides a range of mechanisms to enhance security and reduce risks arising from APIs. These mechanisms include attack prevention, replay prevention, request encryption , identity authentication, permission management, and request throttling.
- API Gateway provides a full range of API lifecycle management functions, including creating, testing, publishing, and unpublishing APIs. It also generates SDKs and API documentation to improve API management and iteration efficiency.
- API Gateway provides convenient O&M functions to reduce API O&M costs, including monitoring, alarms, and log analysis.

API Gateway maximizes capability multiplexing. It allows enterprises to share capabilities and focus more on their core businesses, which benefits all parties involved.

1.2 Quick start for consumers

1.2.1 Overview

You can use API Gateway to call the API services enabled by other Alibaba Cloud users or third-party service providers. API Gateway provides a series of management and support services.

Call an API based on the following conditions:

- **API:** The API that you call is clearly defined by API parameters.
- **Application:** The application that you use to call the API has a key pair that uniquely identifies you.
- **Authorization relationship between the API and application:** An application can be used to call an API only when the application has been granted the permission to call that API. This permission is granted through authorization.

1.2.2 Step 1: Obtain the API document

API providers need to authorize your applications to use their APIs in the API Gateway console. Provide your application IDs (AppId) to the API providers so that the providers can authorize your

applications. For more information about applications, see *Create an application*. Assume that you have created an application and the API provider has authorized your application to use their APIs.

Procedure

1. Log on to the API Gateway console.
2. Click the **Applications** tab to go to the **Applications** tab page.

   The application that you created is displayed in the application list.
3. Click the application name to go to the application details page.

   The application details page consists of the Basic Information, **AppKey**, and **Callable APIs** areas.

   On the application details page:

   - The **AppKey** area shows the **AppKey** and **AppSecret** of the application. Your API request must contain the AppKey and AppSecret. API Gateway verifies your identity based on this key pair.
   - The **Callable APIs** area shows the APIs that the application has been authorized to use. If the API provider has authorized the application to use their APIs, the corresponding APIs are displayed in the callable API list. In the callable API list, click the management icon in the Actions column corresponding to an API and choose **View Details** from the shortcut menu to view the API details.

1.2.3 Step 2: Create an application

Applications are the identities that you use to call APIs. You can own multiple applications that are authorized to use different APIs based on your service requirements. Instead of user accounts, applications are authorized to use APIs. In the API Gateway console, you can create, change, or delete applications, view application details including callable APIs, and manage keys for applications.

Each application has a key pair made up of an **AppKey** and **AppSecret**. This key pair works similar to the way an account and password works. When calling an API, you must include the **AppKey** as a parameter in the request. The **AppSecret** is used to calculate the signature string. API Gateway verifies your identity based on the key pair. Before using an application to call an API, ensure that the application has been authorized to use the API. Both authorization and verification are performed on the application.

You can log on to the API Gateway console to create applications on the **Applications** tab page.

Procedure

1. Log on to the API Gateway console.
2. Click the **Applications** tab.
3. Click **Create Application**.
4. Set parameters and click **Create**.

   The application name must be globally unique. It can contain English letters, numbers, and underscores (_). It must start with a letter and be 4 to 26 characters in length.

   After an application is created, the system automatically assigns an **Appkey** and an **AppSecret** to it. You need to use the **AppSecret** to calculate the signature string. When calling an API, you must include the string in the request. API Gateway verifies your identity based on the signature.

   On the **Applications** tab page, click the application name to go to the application details page. The **AppKey** and **AppSecret** information is displayed in the lower part of the tab page. If the key pair is lost, you can reset it.

1.2.4 Step 3: Obtain authorization

Authorization is the process of authorizing an application to call an API. Your application must be authorized to call an API.

Provide your application ID (AppID) to the API provider so that the provider can authorize your application. After authorization is complete, log on to the API Gateway console.

You can view the API in the **Callable APIs** list on the application details page.

Only the API provider has the permission to authorize applications to call APIs.

1.2.5 Step 4: Call the API

You can use a self-compiled HTTP or HTTPS request to call the API.

Part 1: Request

Request address

```
http://e710888d3ccb4638a723ff8d03837095-cn-qingdao.aliapi.com/demo/
post
```

Request method

```
POST
```

Request body

```
FormParam1=FormParamValue1&FormParam2=FormParamValue2
//HTTP Request Body
```

Request header

```
Host: e710888d3ccb4638a723ff8d03837095-cn-qingdao.aliapi.com
Date: Mon, 22 Aug 2016 11:21:04 GMT
User-Agent: Apache-HttpClient/4.1.2 (java 1.6)
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
//The request body type. Set the type based on the actual request
content.
Accept: application/json
//The response body type. Some APIs return data based on the specified
 response body type. We recommend that you set this Header parameter
manually. If you do not set this parameter, some HTTP clients set it
to */* by default. This can cause a signature error.
X-Ca-Request-Mode: debug
//Whether to enable the Debug mode. This parameter is case-insensitive
. The Debug mode is disabled by default. It is usually enabled during
API debugging.
X-Ca-Version: 1
//The API version number. Only version 1 is supported. This parameter
is optional. Default value: 1.
X-Ca-Signature-Headers: X-Ca-Request-Mode,X-Ca-Version,X-Ca-Stage,X-Ca
-Key,X-Ca-Timestamp
//Custom Header parameters that are used for signature calculation
. The server reads Header parameters based on this setting during
signature calculation. Content-Type, Accept, Content-MD5, and Date
are part of the basic signature structure, and are not included in
custom Header parameters. For more information, see request signature
instructions.
X-Ca-Stage: RELEASE
//The stage of the requested API. Values: TEST, PRE, and RELEASE. This
 parameter is case-insensitive. An API provider can decide to which
stage the API is to be published. When you call an API that has not
been published, an error message is returned, indicating an invalid
URL or that the API is not found.
X-Ca-Key: 60022326
//The request AppKey. AppKeys are generated in the API Gateway console
. An application can call an API only when the application has been
authorized to use the API.
```

```
X-Ca-Timestamp: 1471864864235
//The request timestamp. It is the current time, in milliseconds. A
timestamp is valid for 15 minutes.
X-Ca-Nonce: b931bc77-645a-4299-b24b-f3669be577ac
//The unique identifier of the request. The combination of AppKey,
API, and Nonce must be unique for requests within 15 minutes. This
parameter must be used together with the timestamp to prevent replay.
X-Ca-Signature: FJleSrCYPGCU7dMlLTG+UD3Bc5Elh3TV3CWHtSKh1Ys=
//The request signature.
CustomHeader: CustomHeaderValue
//Custom Header parameters. The preceding example serves only as a
reference. You can set several custom Header parameters based on the
API definition.
```

Part 2: Response

Status code

```
400
//The response status code. If the value is greater than or equal to
 200 and smaller than 300, the request is successful. If the value
is greater than or equal to 400 and smaller than 500, a client error
 has occurred. If the value is greater than 500, a server error has
occurred.
```

Response header

```
X-Ca-Request-Id: 7AD052CB-EE8B-4DFD-BBAF-EFB340E0A5AF
//The unique ID of the request. After receiving a request, API Gateway
 generates a request ID and returns the request ID to the client
through the response header. We recommend that the request ID be
recorded by both the client and the backend service for troublesho
oting and tracking.
X-Ca-Error-Message: Invalid Url
//The error message returned by API Gateway. When a request fails, API
 Gateway returns the error message to the client through the response
header.
X-Ca-Debug-Info: {"ServiceLatency":0,"TotalLatency":2}
//The Debug message that is returned when the Debug mode is enabled
. The message may be changed in the future and is used only for
reference during debugging.
```

To call an API, you must attach a signature to the HTTP or HTTPS request. For more information about the signature encryption algorithm, see *Request signature instructions*.

1.3 Quick start for providers

1.3.1 Overview

This document guides you through creating and publishing an API.

This document guides you through performing the following operations:

1. Create an API group

2. Bind domain names and certificates
3. Create an API
4. Publish an API
5. Authorize applications to use APIs

1.3.2 Create a group

You can create API groups in the API Gateway console. You can create up to 50 groups.

Procedure

1. Log on to the API Gateway console.
2. Click the **Groups** tab.
3. On the Groups tab page that appears, click **Create Group**.
4. In the Create Group dialog box that appears, set Department, Project, and other required parameters. Click **Create**.

   Group names must be globally unique. A group name must be 4 to 50 characters in length. It can contain English letters, numbers, and underscores (_) and must start with an English letter.

1.3.3 Create an API

Procedure

1. Log on to the API Gateway console.
2. Click the **API** tab.
3. Click **Create API**.
4. Set the basic information of the API and click **Next**.

Parameter	Description
Groups	The group to which the API belongs. APIs are managed by groups. Before you create an API, you must create a group. Select a group from the **Groups** drop-down list.
API Name	The API name.
Authentication Mode	The authentication mode of API requests. Values: **Alibaba Cloud Applications** and **None**. • **Alibaba Cloud Applications**: This authentication mode requires the requester

Parameter	Description
	to pass the application authentication to call this API. • **None**: This authentication mode allows any user who knows the request definition of the API to initiate a request. API Gateway directly forwards the requests to your backend service without verifying the identity of the requesters.
Description	The description of the API.

5. Define API requests. Define the settings of the requests that users can send to call the API, including the related protocols, request paths, HTTP methods, and parameters.

Parameter	Description
Network Protocol	The protocol that can be used to call the API. Both HTTP and HTTPS are supported.
Custom Domain Name	The independent domain name that has been bound to the group to which the API belongs.
Second-Level Domain Name	The second-level domain name of the group to which the API belongs.
URL Suffix	The API request path. It corresponds to the service host. The request path can be different from the backend service path. You can provide any valid and semantically -correct path for users. You can configure dynamic parameters in the request path. API Gateway can map these parameters to Query , Header, or other locations before sending the requests to the backend service.
HTTP Method	The HTTP method supported by the API. Values: PUT, GET, POST, PATCH, DELETE , and HEAD.
Request Parameters	The request parameters of the API. These parameters need to be set by the users. You can define the request parameters in Header, Query, Body, or Path (Parameter Path). If you have defined dynamic parameters in Path, specify how to set these dynamic parameters when your define request parameters. The

Parameter	Description
	following parameter types are supported: String, Number, and Boolean.
Parameter verification rules	Click the management icon in the Actions column corresponding to a request parameter, and choose **Configure Advanced Settings** from the shortcut menu. In the Configure Advanced Settings dialog box that appears, you can configure parameter verification rules, such as Maximum Length and Enumeration. API Gateway pre-verifies requests based on the verification rules. The requests with invalid parameters are not sent to your backend service. This greatly reduces the work load of the backend service.

6. Configure the backend service and click **Next**.

   This section defines mappings between frontend and backend parameters, and specifies the API backend service configurations. The backend service configurations include the backend service address, backend service path, backend response timeout period, parameter mappings, constant parameters, and system parameters. After receiving user requests, API Gateway converts the format of the requests to the format required by the backend service based on the backend service configuration. Then, API Gateway forwards the requests to the backend service.

    **Note:**
   You can enter the following parameters: dynamic parameters in Path, Header parameters, Query parameters, Body parameters (non-binary), constant parameters, and system parameters. Each parameter name must be globally unique. For example, you are not allowed to enter a parameter named **name** in both Header and Query.

   a) Configure basic backend service information.

Parameter	Description
Backend Service URL	The backend service host. It can be a domain name or an address in the format of `http(s)://host:port`.
URL Suffix	The actual request path of your API service on the backend server. If you

Parameter	Description
	have configured dynamic parameters in the backend path, you must specify the corresponding request parameters and their locations by declaring the mapping.
Timeout	The maximum amount of time that API Gateway waits for a response from the backend service. API Gateway sends a request to the backend service and waits for a response. The maximum timeout period is 30 seconds. If API Gateway does not receive a response from the backend service within the timeout period, it stops waiting and returns an error.
Mock	You can mock expected responses to return to API callers during the project development process. For more information , see Mock an API.

b) Configure backend service parameters.

API Gateway can set up mappings between frontend and backend parameters, including names and locations. API Gateway can map a request parameter at any location (Path, Header, Query, or Body) to a backend service parameter at a different location. In this way, you can package your backend services into standard APIs. This part declares the frontend-to-backend API mapping.

Note:
The frontend and backend parameters must be globally unique.

c) Configure constant parameters.

To configure API Gateway to add the **`apigateway`** tag to each request it forwards to your backend service, you can configure the tag as a constant parameter. Constant parameters are invisible to users. After a constant parameter is configured, API Gateway automatically adds this parameter to the specified location of the received requests before sending the requests to your backend service.

d) Configure system parameters.

API Gateway does not send its system parameters to you by default. To obtain these parameters, configure their locations and names in the API. The following table lists the system parameters.

Parameter	Description
CaClientIp	The IP address of the client that sends the request
CaDomain	The domain name that is used to send the request
CaRequestHandleTime	The request time (UTC)
CaAppId	The ID of the application that sends the request
CaRequestId	The request ID
CaApiName	The API name
CaHttpSchema	The protocol that is used to call the API, which is HTTP or HTTPS
CaProxy	The proxy (AliCloudApiGateway)

7. Define the response and click **Create**.

   You can set Response Content Type, Success Response Example, and Error Response Example, and define error codes. API Gateway does not parse responses, but forwards them to the API requester.

1.3.4 Publish an API

After creating an API, you need to debug, test, and publish it.

- When you use a second-level or independent domain name to access an API published to an environment, you must specify the environment to be requested in the request header.
- If you publish an API to the test or release environment where the API already has a version running, the newly published version replaces the running version to take effect in real time. However, all historical versions and definitions are recorded so you can roll the API back to an earlier version.
- You can unpublish an API in the test or release environment. After the API is unpublished , its binding or authorization relationships with the policy, keys, or applications persist and will automatically take effect when the API is published again. Remove these relationships separately if you no longer need them.

Step 1: Test the API

You can create an application and authorize the application to use this API. Then, use this application to simulate real user requests.

You can write code based on real request scenarios or use the SDK samples provided by API Gateway to call the API.

You can publish an API to the test or release environment. If no independent domain name is bound to the group to which the API belongs, you can test or call the API by using the second-level domain name. Specify an environment in your request. If no environment is specified in your request, the API published to the release environment is called by default.

Step 2: Publish the API

After testing the API, you can publish it.

You can use API Gateway to manage versions of APIs in the test or release environment. You can publish or unpublish an API and change its version. The version change takes effect in real time.

1. Log on to the API Gateway console.
2. Click the **API** tab.
3. Locate the API that you want to publish, click the management icon in the Actions column, and choose **Publish** from the shortcut menu.
4. In the dialog box that appears, select an environment, enter a description, and click **OK**.

1.3.5 Authorize applications to call APIs

You need to authorize an application before the application can call your API. After you publish an API to the release environment, you must authorize the applications of users before they can use the API. You can perform an authorization or deauthorization operation to establish or remove an authorization relationship between an API and an application. API Gateway verifies the authorization relationship.

Note:

- You can authorize one or more applications to use one or more APIs.
- If an API has been published to both the test and release environments but only the test environment is selected, applications are authorized to use only the API in the test environment.
- You can locate an application based on its ID or name provided by the user.

- To revoke the authorization from an application under an API, go to the Authorization tab page of the API. Select the application from the application list. Then, click the management icon in the Actions column and choose Deauthorize from the shortcut menu, or click Deauthorize in the upper-right corner.

Applications indicate requesters' identities. When you or your customers test or call an API, you must create an application as the requester's identity and then authorize the application to use the API.

Note:
Authorizations are environment-specific. The same application must be authorized to use the same API in both the test and release environments to avoid errors caused by inconsistency between the authorized environment and requested environment.

Procedure

1. Log on to the API Gateway console.
2. Click the **API** tab.
3. Locate the API that you want to authorize applications to use, click the management icon in the Actions column, and choose **Authorize** from the shortcut menu.
4. In the dialog box that appears, set Environment.
5. Select applications from the left-side list, click **right arrow** to add the selected applications to the right-side list, and click **OK**.

   You can enter an application ID or name to search for applications.

What's next

You can view the authorization information of APIs or revoke the authorization from an application under an API.

In the API list, click the management icon in the **Actions** column corresponding to an API, and choose View Details from the shortcut menu. Click the **Authorization** tab. On the Authorization tab page, you can view the applications that have been authorized to use this API.

You can select one or more application IDs and click **Deauthorize** in the upper-right corner to revoke the authorizations from the selected applications under the API.

1.4 Call an API

1.4.1 Manage applications

1.4.1.1 Create an application

Applications are the identities that you use to call APIs. You can own multiple applications that are authorized to use different APIs based on your service requirements. Instead of user accounts, applications are authorized to use APIs. In the API Gateway console, you can create, change, or delete applications, view application details including callable APIs, and manage keys for applications.

Each application has a key pair made up of an **AppKey** and **AppSecret**. This key pair works similar to the way an account and password works. When calling an API, you must include the **AppKey** as a parameter in the request. The **AppSecret** is used to calculate the signature string. API Gateway verifies your identity based on the key pair. Before using an application to call an API, ensure that the application has been authorized to use the API. Both authorization and verification are performed on the application.

You can log on to the API Gateway console to create applications on the **Applications** tab page.

Procedure

1. Log on to the API Gateway console.
2. Click the **Applications** tab.
3. Click **Create Application**.
4. Set parameters and click **Create**.

   The application name must be globally unique. It can contain English letters, numbers, and underscores (_). It must start with a letter and be 4 to 26 characters in length.

   After an application is created, the system automatically assigns an **Appkey** and an **AppSecret** to it. You need to use the **AppSecret** to calculate the signature string. When calling an API, you must include the string in the request. API Gateway verifies your identity based on the signature.

   On the **Applications** tab page, click the application name to go to the application details page. The **AppKey** and **AppSecret** information is displayed in the lower part of the tab page. If the key pair is lost, you can reset it.

1.4.1.2 View application details

You can view the details of applications.

Procedure

1. Log on to the API Gateway console.
2. Click the **Applications** tab to go to the **Applications** tab page.
3. Click the name of the application that you want to view.

   View the basic information, AppKey, and callable APIs of the application. The callable APIs are the APIs that this application has been authorized to use.

1.4.1.3 Change an application

You can change existing applications.

Procedure

1. Log on to the API Gateway console.
2. Click the **Applications** tab to go to the **Applications** tab page.
3. Locate the application that you want to change, click the management icon in the Actions column, and choose **Change** from the shortcut menu.
4. Change the application information and click **OK**.

1.4.1.4 Delete an application

You can delete existing applications.

Procedure

1. Log on to the API Gateway console.
2. Click the **Applications** tab to go to the **Applications** tab page.
3. Locate the application that you want to delete, click the management icon in the Actions column, and choose **Delete** from the shortcut menu.
4. In the message that appears, click **OK**.

1.4.2 View existing APIs

You can view existing APIs in the API Gateway console.

Procedure

1. Log on to the API Gateway console.
2. Click the **API** tab.

1.4.3 Authorization

Authorization is the process of authorizing an application to call an API. Your application must be authorized to call an API.

Provide your application ID (AppID) to the API provider so that the provider can authorize your application. After authorization is complete, log on to the API Gateway console.

You can view the API in the **Callable APIs** list on the application details page.

Only the API provider has the permission to authorize applications to call APIs.

1.4.4 Encrypt the signature

When you call an API, you need to construct the signature string and place the calculated string in the request header. API Gateway performs symmetric signature calculation to verify the identity of the requester.

- The calculated signature string is attached to the request header.
- You need organize the request parameters into StringToSign based on *request signature instructions*. Then, use the algorithm provided in the SDK sample to calculate the signature. The result is the preceding calculated signature string.
- Both HTTP and HTTPS requests must carry signatures.

For more information about the organization method of StringToSign, see *Request signature instructions*. You only need to change the AppKey and AppSecret in the SDK sample to your own AppKey and AppSecret. Organize StringToSign based on request signature instructions. Then, you can use the signature string to initiate a request.

1.4.5 Request signature instructions

Domain name

- Each API belongs to an API group, and each API group has a unique domain name. The domain names are independent domain names that are bound to API groups by the service provider. API Gateway uses domain names to locate API groups.
- Domain names are in the format of *`www.[independent domain name].com/[Path]?[HTTPMethod]`*.
- API Gateway locates a API group by domain name, and locates the unique API by the combination of Path and HTTPMethod.
- After you purchase an API, you can obtain the API documentation from the **Purchased APIs** list in the API Gateway console. If you have not purchased an API, you need to require

the API provider to authorize your application to call the API. Then, you can obtain the API documentation from the **Callable APIs** list on the application details page.

System-level Header parameters

- (Required) X-Ca-Key: AppKey.
- (Required) X-Ca-Signature: the signature string.
- (Optional) X-Ca-Timestamp: the timestamp passed by the API caller. It is the current time, in milliseconds. A timestamp is valid for 15 minutes.
- (Optional) X-Ca-Nonce: the UUID generated by the API caller. This parameter is used together with the timestamp to prevent replay attacks.
- (Optional) Content-MD5: When the request body is not a form, you can calculate the MD5 value of the body. Then, you can send the value to API Gateway for an MD5 check of the body.
- (Optional) X-Ca-Stage: the stage of the requested API. Values: TEST, PRE, and RELEASE. Default value: RELEASE. If the called API is not in the release environment, you must specify this parameter. If the called API is not in the release environment and you do not specify this parameter, an error is reported.

Signature verification

Organize the strings used for signature calculation

```
String stringToSign=
HTTPMethod + "\n" +
Accept + "\n" +                //Set Accept in Header. If Accept is
 empty, some HTTP clients set the value to */* by default. This can
cause a signature verification failure.
Content-MD5 + "\n"
Content-Type + "\n" +
Date + "\n" +
Headers +
Url
```

The value of HTTPMethod is in all caps, for example, POST.

If Accept, Content-MD5, Content-Type, and Date are empty, add a linefeed `\n`. If Headers is empty, `\n` is not required.

Content-MD5

Content-MD5 indicates the MD5 value of the body. The MD5 value is calculated only when the body is not a form. The calculation formula is as follows:

```
String content-MD5 = Base64.encodeBase64(MD5(bodyStream.getbytes("UTF-8")));
```

bodyStream indicates the byte array.

Headers

It indicates the string constructed by the keys and values of the Header parameters that are used for Headers signature calculation. We recommend that the parameters starting with X-Ca and custom Header parameters be used for signature calculation.

Note:
The following parameters are not used for Headers signature calculation: X-Ca-Signature, X-Ca-Signature-Headers, Accept, Content-MD5, Content-Type, and Date.

Headers organization method:

Sort the keys used for Headers signature calculation **in alphabetical order**. Construct the string based on the following rule: If the value of a Header parameter is empty, use `HeaderKey + ":" + "\n"` for signature calculation. The key and colon (:) must be retained.

```
String headers =
HeaderKey1 + ":" + HeaderValue1 + "\n"\+
HeaderKey2 + ":" + HeaderValue2 + "\n"\+
...
HeaderKeyN + ":" + HeaderValueN + "\n"
```

The keys of Header parameters used for Headers signature calculation are separated by commas (,), and placed in the request header. The key is X-Ca-Signature-Headers.

Url

Url indicates the **`Form`** parameter in **`Path + Query + Body`**. The organization method is as follows: For **`Query + Form`**, sort **`Key`** in alphabetical order and construct the string based on the following rule: If **`Query`** or **`Form`** is empty, `Url = Path` and the question mark (?) is not required. If **`Value`** of a parameter is empty, only **`Key`** is used for signature calculation and the equal sign (=) is not required.

```
String url =
Path +
"?" +
Key1 + "=" + Value1 +
"&" + Key2 + "=" + Value2 +
```

```
...
"&" + KeyN + "=" + ValueN
```

Note:

Note that **Query** or **Form** may have multiple **values**. If there are multiple values, the first value is used for signature calculation.

Calculate the signature

```
Mac hmacSha256 = Mac.getInstance("HmacSHA256");
byte[] keyBytes = secret.getBytes("UTF-8");
hmacSha256.init(new SecretKeySpec(keyBytes, 0, keyBytes.length, "
HmacSHA256"));
String sign = new String(Base64.encodeBase64(Sha256.doFinal(stringToSi
gn.getBytes("UTF-8")),"UTF-8"));
```

`secret` is an AppSecret.

Pass the signature

Place the calculated signature in the request header. The key is X-Ca-Signature.

Signature troubleshooting

If signature verification fails, API Gateway places StringToSign of the server in the HTTP response header and sends the response to the client. The key is X-Ca-Error-Message. Compare StringToSign that is calculated by the client with the one returned by the server.

If the StringToSign values from the client and server are the same, check the AppSecret used for signature calculation.

The HTTP Header does not support linefeeds, so the linefeeds in StringToSign are filtered out. Ignore linefeeds during comparison.

Signature demo

For a detailed demo (Java) of signature calculation, refer to *https://github.com/aliyun/api-gateway-demo-sign-java*.

1.4.6 API call example

You can use a self-compiled HTTP or HTTPS request to call the API.

Part 1: Request

Request address

```
http://e710888d3ccb4638a723ff8d03837095-cn-qingdao.aliapi.com/demo/
post
```

Request method

```
POST
```

Request body

```
FormParam1=FormParamValue1&FormParam2=FormParamValue2
//HTTP Request Body
```

Request header

```
Host: e710888d3ccb4638a723ff8d03837095-cn-qingdao.aliapi.com
Date: Mon, 22 Aug 2016 11:21:04 GMT
User-Agent: Apache-HttpClient/4.1.2 (java 1.6)
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
//The request body type. Set the type based on the actual request
content.
Accept: application/json
//The response body type. Some APIs return data based on the specified
 response body type. We recommend that you set this Header parameter
manually. If you do not set this parameter, some HTTP clients set it
to */* by default. This can cause a signature error.
X-Ca-Request-Mode: debug
//Whether to enable the Debug mode. This parameter is case-insensitive
. The Debug mode is disabled by default. It is usually enabled during
API debugging.
X-Ca-Version: 1
//The API version number. Only version 1 is supported. This parameter
is optional. Default value: 1.
X-Ca-Signature-Headers: X-Ca-Request-Mode,X-Ca-Version,X-Ca-Stage,X-Ca
-Key,X-Ca-Timestamp
//Custom Header parameters that are used for signature calculation
. The server reads Header parameters based on this setting during
signature calculation. Content-Type, Accept, Content-MD5, and Date
are part of the basic signature structure, and are not included in
custom Header parameters. For more information, see request signature
instructions.
X-Ca-Stage: RELEASE
//The stage of the requested API. Values: TEST, PRE, and RELEASE. This
 parameter is case-insensitive. An API provider can decide to which
stage the API is to be published. When you call an API that has not
been published, an error message is returned, indicating an invalid
URL or that the API is not found.
X-Ca-Key: 60022326
//The request AppKey. AppKeys are generated in the API Gateway console
. An application can call an API only when the application has been
authorized to use the API.
```

```
X-Ca-Timestamp: 1471864864235
//The request timestamp. It is the current time, in milliseconds. A
timestamp is valid for 15 minutes.
X-Ca-Nonce: b931bc77-645a-4299-b24b-f3669be577ac
//The unique identifier of the request. The combination of AppKey,
API, and Nonce must be unique for requests within 15 minutes. This
parameter must be used together with the timestamp to prevent replay.
X-Ca-Signature: FJleSrCYPGCU7dMlLTG+UD3Bc5Elh3TV3CWHtSKh1Ys=
//The request signature.
CustomHeader: CustomHeaderValue
//Custom Header parameters. The preceding example serves only as a
reference. You can set several custom Header parameters based on the
API definition.
```

Part 2: Response

Status code

```
400
//The response status code. If the value is greater than or equal to
 200 and smaller than 300, the request is successful. If the value
is greater than or equal to 400 and smaller than 500, a client error
 has occurred. If the value is greater than 500, a server error has
occurred.
```

Response header

```
X-Ca-Request-Id: 7AD052CB-EE8B-4DFD-BBAF-EFB340E0A5AF
//The unique ID of the request. After receiving a request, API Gateway
 generates a request ID and returns the request ID to the client
through the response header. We recommend that the request ID be
recorded by both the client and the backend service for troublesho
oting and tracking.
X-Ca-Error-Message: Invalid Url
//The error message returned by API Gateway. When a request fails, API
 Gateway returns the error message to the client through the response
header.
X-Ca-Debug-Info: {"ServiceLatency":0,"TotalLatency":2}
//The Debug message that is returned when the Debug mode is enabled
. The message may be changed in the future and is used only for
reference during debugging.
```

To call an API, you must attach a signature to the HTTP or HTTPS request. For more information about the signature encryption algorithm, see *Request signature instructions*.

1.5 APIs

1.5.1 Usage limits

Item	Limit
Number of API groups that can be created by a user	50.

Item	Limit
Number of APIs that can be created by a user	10,000. (Each user can create up to 50 API groups, and each group can contain up to 200 APIs.)
Number of independent domain names that can be bound to an API group	5.
Transactions per second (TPS) that can be handled by an API group	500. You can increase this value based on your business requirements.

1.5.2 Manage groups

1.5.2.1 Create a group

You can create API groups in the API Gateway console. You can create up to 50 groups.

Procedure

1. Log on to the API Gateway console.
2. Click the **Groups** tab.
3. On the Groups tab page that appears, click **Create Group**.
4. In the Create Group dialog box that appears, set Department, Project, and other required parameters. Click **Create**.

   Group names must be globally unique. A group name must be 4 to 50 characters in length. It can contain English letters, numbers, and underscores (_) and must start with an English letter.

1.5.2.2 Environment management

To understand environment management, you need to be familiar with two concepts: environment and environment variables.

- An **environment** is a configuration of an API group. You can configure several environments for a group. APIs that are not published are considered as API definitions. After you publish the API to an environment, it starts to provide external services.
- **Environment variables** are environment-specific variables that can be created and managed. For example, you can create an environment variable named `Path` and valued `/stage/release` for the release environment.

In the API definition, you can set `Path` to `#Path#` (which is a variable), and set Parameter Name to `Path`.

When you publish the API to the release environment, the value of `#Path#` in Path is `/stage/release`.

When you publish the API to another environment that does not have the environment variable `#Path#`, the variable in the API fails to obtain the value and the API cannot be called.

You can use environment variables to configure different service addresses for different environments in an API definition. API Gateway calls different backend services based on the environment variable values. Pay attention to the following points:

- Variable names are case-sensitive.
- If you configure a variable in the API definition, you must configure the name and value of the variable for the environment to which the API is published. Otherwise, the variable will not take a value, and the API will not be called.

Create an environment variable

1. Log on to the API Gateway console.
2. Click the **Groups** tab.
3. Locate a group, click the management icon in the Actions column corresponding to the group, and choose View Details from the shortcut menu.
4. On the page that appears, click the **Environment Variables** tab. On the Environment Variables tab page that appears, click Create Variable.
5. In the Create Environment Variable dialog box that appears, set **Variable Name** and **Variable Value**. Click **OK**.

Delete a variable

1. Log on to the API Gateway console.
2. Click the **Groups** tab.
3. Locate a group, click the management icon in the Actions column corresponding to the group, and choose View Details from the shortcut menu.
4. On the page that appears, click the **Environment Variables** tab.
5. On the Environment Variables tab page that appears, select an environment. Locate the variable that you want to delete, click the management icon in the Actions column corresponding to the variable, and choose **Delete** from the shortcut menu.
6. In the message that appears, click **OK**.

1.5.2.3 Delete a group

You can delete an existing group.

Procedure

1. Log on to the API Gateway console.
2. Click the **Groups** tab.
3. Locate the group that you want to delete, click the management icon in the Actions column corresponding to the group, and choose **Delete** from the shortcut menu.
4. In the message that appears, click **OK**.

1.5.3 Create an API

1.5.3.1 Overview

Creating an API is the process of defining the API in the API Gateway console. When creating an API, you need to define the basic information, backend service information, request information, and response information of the API.

- You can configure parameter verification rules. API Gateway pre-verifies API requests based on the verification rules and forwards the requests that contain only valid parameters to the backend service.
- You can configure API Gateway to map a frontend parameter to a backend parameter at any location. For example, you can configure API Gateway to map a **Query** parameter in an API request to a **Header** parameter in a backend service request. In this way, you can package your backend services into standard APIs.
- API Gateway allows you to configure constant parameters and system parameters. These parameters are invisible to users, but API Gateway can attach them to the received requests based on your service requirements before sending the requests to backend services. If you want API Gateway to attach the keyword **apigateway** to each request it forwards to your backend service, you can configure **aligateway** as a constant parameter and specify where it is received.

1.5.3.2 Create an API

Procedure

1. Log on to the API Gateway console.
2. Click the **API** tab.
3. Click **Create API**.

4. Set the basic information of the API and click **Next**.

Parameter	Description
Groups	The group to which the API belongs. APIs are managed by groups. Before you create an API, you must create a group. Select a group from the **Groups** drop-down list.
API Name	The API name.
Authentication Mode	The authentication mode of API requests. Values: **Alibaba Cloud Applications** and **None**. • **Alibaba Cloud Applications**: This authentication mode requires the requester to pass the application authentication to call this API. • **None**: This authentication mode allows any user who knows the request definition of the API to initiate a request. API Gateway directly forwards the requests to your backend service without verifying the identity of the requesters.
Description	The description of the API.

5. Define API requests. Define the settings of the requests that users can send to call the API, including the related protocols, request paths, HTTP methods, and parameters.

Parameter	Description
Network Protocol	The protocol that can be used to call the API. Both HTTP and HTTPS are supported.
Custom Domain Name	The independent domain name that has been bound to the group to which the API belongs.
Second-Level Domain Name	The second-level domain name of the group to which the API belongs.
URL Suffix	The API request path. It corresponds to the service host. The request path can be different from the backend service path. You can provide any valid and semantically -correct path for users. You can configure dynamic parameters in the request path. API Gateway can map these parameters to Query

Parameter	Description
	, Header, or other locations before sending the requests to the backend service.
HTTP Method	The HTTP method supported by the API. Values: PUT, GET, POST, PATCH, DELETE , and HEAD.
Request Parameters	The request parameters of the API. These parameters need to be set by the users. You can define the request parameters in Header, Query, Body, or Path (Parameter Path). If you have defined dynamic parameters in Path, specify how to set these dynamic parameters when your define request parameters. The following parameter types are supported: String, Number, and Boolean.
Parameter verification rules	Click the management icon in the Actions column corresponding to a request parameter, and choose **Configure Advanced Settings** from the shortcut menu. In the Configure Advanced Settings dialog box that appears, you can configure parameter verification rules, such as Maximum Length and Enumeration. API Gateway pre-verifies requests based on the verification rules. The requests with invalid parameters are not sent to your backend service. This greatly reduces the work load of the backend service.

6. Configure the backend service and click **Next**.

   This section defines mappings between frontend and backend parameters, and specifies the API backend service configurations. The backend service configurations include the backend service address, backend service path, backend response timeout period, parameter mappings, constant parameters, and system parameters. After receiving user requests, API Gateway converts the format of the requests to the format required by the backend service based on the backend service configuration. Then, API Gateway forwards the requests to the backend service.

   Note:

You can enter the following parameters: dynamic parameters in Path, Header parameters, Query parameters, Body parameters (non-binary), constant parameters, and system parameters. Each parameter name must be globally unique. For example, you are not allowed to enter a parameter named **name** in both Header and Query.

a) Configure basic backend service information.

Parameter	Description
Backend Service URL	The backend service host. It can be a domain name or an address in the format of `http(s)://host:port`.
URL Suffix	The actual request path of your API service on the backend server. If you have configured dynamic parameters in the backend path, you must specify the corresponding request parameters and their locations by declaring the mapping.
Timeout	The maximum amount of time that API Gateway waits for a response from the backend service. API Gateway sends a request to the backend service and waits for a response. The maximum timeout period is 30 seconds. If API Gateway does not receive a response from the backend service within the timeout period, it stops waiting and returns an error.
Mock	You can mock expected responses to return to API callers during the project development process. For more information , see Mock an API.

b) Configure backend service parameters.

API Gateway can set up mappings between frontend and backend parameters, including names and locations. API Gateway can map a request parameter at any location (Path, Header, Query, or Body) to a backend service parameter at a different location. In this way, you can package your backend services into standard APIs. This part declares the frontend-to-backend API mapping.

Note:
The frontend and backend parameters must be globally unique.

c) Configure constant parameters.

To configure API Gateway to add the **`apigateway`** tag to each request it forwards to your backend service, you can configure the tag as a constant parameter. Constant parameters are invisible to users. After a constant parameter is configured, API Gateway automatically adds this parameter to the specified location of the received requests before sending the requests to your backend service.

d) Configure system parameters.

API Gateway does not send its system parameters to you by default. To obtain these parameters, configure their locations and names in the API. The following table lists the system parameters.

Parameter	Description
CaClientIp	The IP address of the client that sends the request
CaDomain	The domain name that is used to send the request
CaRequestHandleTime	The request time (UTC)
CaAppId	The ID of the application that sends the request
CaRequestId	The request ID
CaApiName	The API name
CaHttpSchema	The protocol that is used to call the API, which is HTTP or HTTPS
CaProxy	The proxy (AliCloudApiGateway)

7. Define the response and click **Create**.

You can set Response Content Type, Success Response Example, and Error Response Example, and define error codes. API Gateway does not parse responses, but forwards them to the API requester.

1.5.3.3 Support HTTPS

Hyper Text Transfer Protocol Secure (HTTPS) is based on the Hyper Text Transfer Protocol (HTTP) and Secure Sockets Layer (SSL) protocols. HTTPS is used to encrypt information and data to secure data transmission. HTTPS is widely used today.

API Gateway supports HTTPS-based encryption of API requests. You can configure APIs to support HTTP, HTTPS, or both.

Perform the following steps to configure your APIs to support HTTPS.

Step 1: Make preparations.

Prepare the following items:

- An independent domain name
- An SSL certificate that has been applied for this domain name

  The SSL certificate contains two parts: XXXXX.key and XXXXX.pem, both of which can be opened in text editors.

  Example:

  KEY

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEA8GjIleJ7rlo86mtbwcDnUfqzTQAm4b3zZEo1aKsfAuwcvCud
....
-----END RSA PRIVATE KEY-----
```

  PEM

```
-----BEGIN CERTIFICATE-----
MIIFtDCCBJygAwIBAgIQRgWF1j00cozRl1pZ+ultKTANBgkqhkiG9w0BAQsFADBP
...
-----END CERTIFICATE-----
```

Step 2: Bind the SSL Certificate to an API group.

After you prepare the preceding items, log on to the API Gateway console and click the **Groups** tab. Locate the API group to which you want to bind the SSL certificate and view the group details.

Bind the independent domain name to the API group before you bind the SSL certificate to the group.

- **Certificate Name**: indicates the name of the certificate.
- **Certificate Content**: indicates the content of the entire certificate. Copy the content in the *XXXXX.pem* file.

- **Private Key**: indicates the private key of the certificate. Copy the content in the *XXXXX.key* file. Click **OK** to bind the SSL certificate to the API group.

Step 3: Adjust the API configuration.

After binding the SSL certificate to the API group, you can configure APIs in the group to support access over HTTP, access over HTTPS, or both. For security considerations, access over HTTPS is recommended.

Locate an API whose configurations you want to adjust on the **API** tab page. Click the management icon in the Actions column and choose **Change** from the shortcut menu. In the Request Basic Settings area on the Define API Request tab page, set Network Protocol.

APIs support the following protocols:

- HTTP: The API supports only access over HTTP.
- HTTPS: The API supports only access over HTTPS.
- HTTP and HTTPS: The API supports both access over both HTTP and HTTPS. If you select HTTPS for Network Protocol, the API supports access over HTTPS.

1.5.3.4 Mock an API

You can mock expected responses to return to API callers during the project development process. This can greatly reduce miscommunication and misunderstanding among team members and significantly improve the development efficiency.

API Gateway supports simple configuration in mock mode.

Configure a mock

On the **Change API** > **Define Backend Service** tab page of an API, configure a mock.

1. Select the Mock type.

   You can set Backend Service Type to Mock and confirm your setting as prompted.

2. Configure the mock response.

   You can enter an actual response in the Mock Response field. Currently, mock responses in the JSON, XML, and text formats are supported. For example:

```
{
"result": {
    "title": " Mock test for API Gateway",
}
```

```
}
```

Save the mock configurations. **Publish** the API to the test or release environment for testing as needed.

1.5.4 API management

1.5.4.1 View and modify an API

You can view APIs and modify them as needed.

> **Note:**
> If you modify an API that has been published to the release environment, you must republish the API for the modifications to take effect in the release environment.

Procedure

1. Log on to the API Gateway console.
2. Click the **API** tab.
3. Locate the API that you want to view.

   View information of the API.
4. Click the management icon in the Actions column corresponding to an API and choose **Change** from the shortcut menu.
5. Set parameters as required and click **Change**.

   The procedure to modify an API is similar to that to create an API. For more information about how to create an API, see *Create an API*.

   If you do not want to continue modifying the API, click **Cancel** in the lower-right corner.

1.5.4.2 Publish an API

After creating an API, you need to debug, test, and publish it.

- When you use a second-level or independent domain name to access an API published to an environment, you must specify the environment to be requested in the request header.
- If you publish an API to the test or release environment where the API already has a version running, the newly published version replaces the running version to take effect in real time. However, all historical versions and definitions are recorded so you can roll the API back to an earlier version.
- You can unpublish an API in the test or release environment. After the API is unpublished , its binding or authorization relationships with the policy, keys, or applications persist and

will automatically take effect when the API is published again. Remove these relationships separately if you no longer need them.

Step 1: Test the API

You can create an application and authorize the application to use this API. Then, use this application to simulate real user requests.

You can write code based on real request scenarios or use the SDK samples provided by API Gateway to call the API.

You can publish an API to the test or release environment. If no independent domain name is bound to the group to which the API belongs, you can test or call the API by using the second-level domain name. Specify an environment in your request. If no environment is specified in your request, the API published to the release environment is called by default.

Step 2: Publish the API

After testing the API, you can publish it.

You can use API Gateway to manage versions of APIs in the test or release environment. You can publish or unpublish an API and change its version. The version change takes effect in real time.

1. Log on to the API Gateway console.
2. Click the **API** tab.
3. Locate the API that you want to publish, click the management icon in the Actions column, and choose **Publish** from the shortcut menu.
4. In the dialog box that appears, select an environment, enter a description, and click **OK**.

1.5.4.3 Authorize applications to call APIs

You need to authorize an application before the application can call your API. After you publish an API to the release environment, you must authorize the applications of users before they can use the API. You can perform an authorization or deauthorization operation to establish or remove an authorization relationship between an API and an application. API Gateway verifies the authorization relationship.

Note:

- You can authorize one or more applications to use one or more APIs.

- If an API has been published to both the test and release environments but only the test environment is selected, applications are authorized to use only the API in the test environment.
- You can locate an application based on its ID or name provided by the user.
- To revoke the authorization from an application under an API, go to the Authorization tab page of the API. Select the application from the application list. Then, click the management icon in the Actions column and choose Deauthorize from the shortcut menu, or click Deauthorize in the upper-right corner.

Applications indicate requesters' identities. When you or your customers test or call an API, you must create an application as the requester's identity and then authorize the application to use the API.

Note:
Authorizations are environment-specific. The same application must be authorized to use the same API in both the test and release environments to avoid errors caused by inconsistency between the authorized environment and requested environment.

Procedure

1. Log on to the API Gateway console.
2. Click the **API** tab.
3. Locate the API that you want to authorize applications to use, click the management icon in the Actions column, and choose **Authorize** from the shortcut menu.
4. In the dialog box that appears, set Environment.
5. Select applications from the left-side list, click **right arrow** to add the selected applications to the right-side list, and click **OK**.

   You can enter an application ID or name to search for applications.

What's next

You can view the authorization information of APIs or revoke the authorization from an application under an API.

In the API list, click the management icon in the **Actions** column corresponding to an API, and choose View Details from the shortcut menu. Click the **Authorization** tab. On the Authorization tab page, you can view the applications that have been authorized to use this API.

You can select one or more application IDs and click **Deauthorize** in the upper-right corner to revoke the authorizations from the selected applications under the API.

1.5.4.4 Delete an API

You can delete existing APIs.

> **Note:**
> Before deleting a published API, you must unpublish it.

Procedure

1. Log on to the API Gateway console.
2. Click the **API** tab.
3. Locate the API that you want to delete, click the management icon in the Actions column, and choose **Delete** from the shortcut menu.
4. In the message that appears, click **OK**.

1.5.4.5 Unpublish an API

You can unpublish a published API.

You can unpublish APIs in the test or release environments. After an API is unpublished, its binding or authorization relationships with policies, keys, or applications still persist. These relationships will take effect if the API is published again. Remove these relationships separately if you no longer need them.

Procedure

1. Log on to the API Gateway console.
2. Click the **API** tab.
3. Locate the API that you want to unpublish, click the management icon in the Actions column, and choose **Take Offline** from the shortcut menu.
4. In the message that appears, click **OK**.

1.5.4.6 View the version history of an API

You can view the version history of an API, including the version number, description, environment, release time, and specific definition of each version.

Procedure

1. Log on to the API Gateway console.

2. Click the **API** tab.
3. Locate the API that you want to view.
4. Click the management icon in the Actions column and choose View Details from the shortcut menu to go to the API details page. Click the **Version History** tab.
5. On the Version History tab page, you can click the management icon in the Actions column corresponding to a version and choose **View** from the shortcut menu to view the API details.

1.5.4.7 Change the version of an API

When viewing the version history of an API, you can select a different version and switch the API to that version. The new version directly replaces the previous one to take effect in real time in the specified environment.

Procedure

1. Log on to the API Gateway console.
2. Click the **API** tab.
3. Locate the API of which you want to change the version.
4. Click the management icon in the Actions column and choose View Details from the shortcut menu to go to the API details page. Click the **Version History** tab.
5. Locate the target version, click the management icon in the Actions column, and choose **Switch to this Version** from the shortcut menu.
6. In the dialog box that appears, enter a description and click **OK**.

1.5.5 Throttling policies

1.5.5.1 Create a throttling policy

You can create throttling policies. Throttling policies are a kind of plug-ins.

Procedure

1. Log on to the API Gateway console.
2. Click the **Plug-ins** tab.
3. Click **Create Plug-in** in the upper-right corner.
4. Set parameters and click **OK**.

1.5.5.2 Bind a throttling policy to APIs

After creating a throttling policy, you need to bind it to an API for the policy to take effect on the bound API.

Context

You can bind a throttling policy to multiple APIs. The limits defined in the throttling policy will be applicable to each bound API. When you bind a throttling policy to an API that is bound with another throttling policy, the new policy takes effect immediately in place of the old policy.

Procedure

1. Log on to the API Gateway console.
2. Click the **Plug-ins** tab.
3. Locate the plug-in that you want to bind to APIs, click the management icon in the Actions column, and choose **Associate API** from the shortcut menu.
4. In the dialog box that appears, set Environment and Groups.
5. Select APIs from the left-side list, click **right arrow** to add the selected APIs to the right-side list, and click**OK**.

1.5.5.3 Delete a throttling policy

You can delete existing throttling policies.

Procedure

1. Log on to the API Gateway console.
2. Click the **Plug-ins** tab.
3. Locate the plug-in that you want to delete, click the management icon in the Actions column, and choose **Delete Plug-in** from the shortcut menu.
4. In the message that appears, click **OK**.