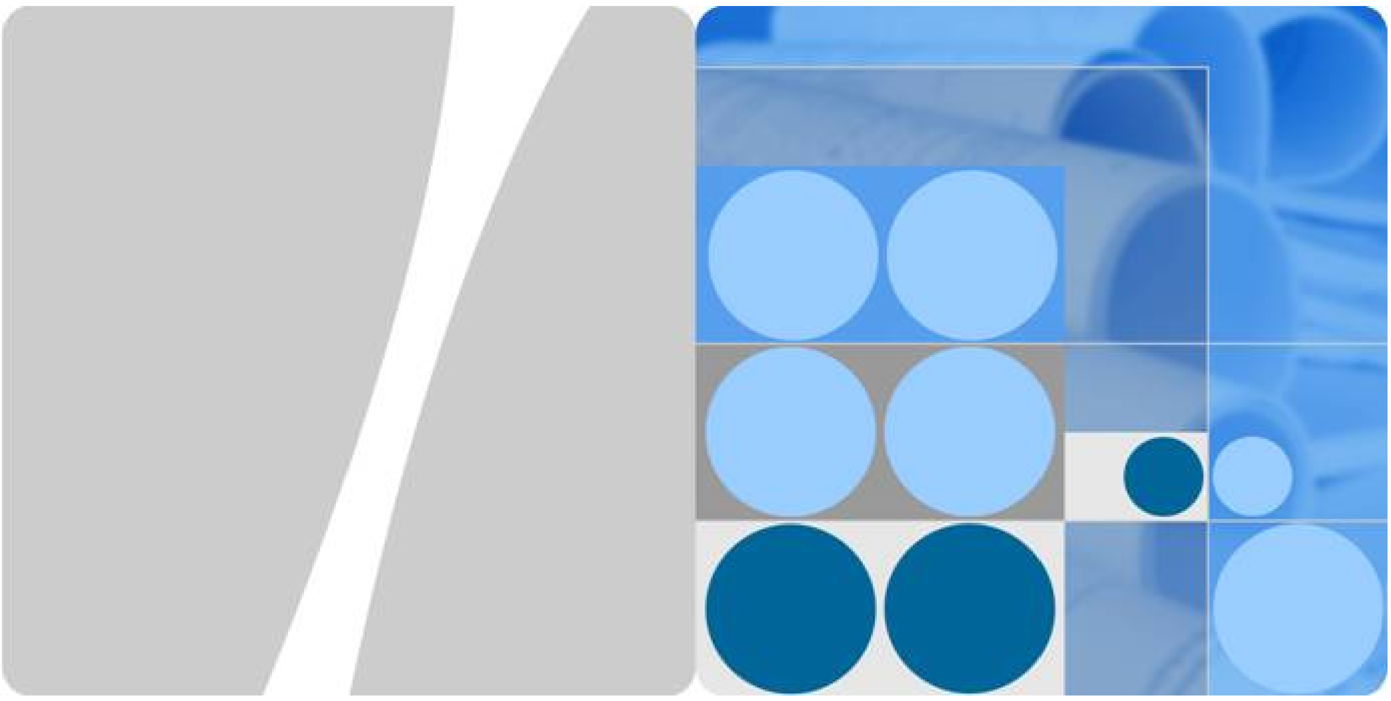


FunctionGraph

API Reference

Date **2021-01-19**

Contents

1 Before You Start

1.1 Overview

Welcome to FunctionGraph. FunctionGraph hosts and computes event-driven functions in a serverless context while ensuring high availability, high scalability, and zero maintenance. All you need to do is write your code and set conditions. You pay only for what you use and you are not charged when your code is not running.

This document describes how to use application programming interfaces (APIs) to perform operations on FunctionGraph resources, such as creating, deleting, query, and executing functions. For details about all supported operations, see **API Overview**.

1.2 API Calling

FunctionGraph supports Representational State Transfer (REST) APIs, allowing you to call APIs using HTTPS. For details about API calling, see **Calling APIs**.

1.3 Endpoints

An endpoint is the **request address** for calling an API. Endpoints vary depending on services and regions. For the endpoints of all services, see **Regions and Endpoints**.

1.4 Constraints

- The number of functions that you can create is determined by your quota. For details, see **Quotas**.
- For more constraints, see API description.

1.5 Concepts

- Account

  An account is created upon successful registration with the cloud system. The account has full access permissions for all of its cloud services and resources. It can be used to reset user passwords and grant user permissions. The account is a payment entity and should not be used directly to perform routine management. For security purposes, create Identity and Access Management (IAM) users and grant them permissions for routine management.

- IAM user

  An IAM user is created using an account to use cloud services. Each IAM user has its own identity credentials (password and access keys).

  The account name, username, and password will be required for API authentication.

- Region

  Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

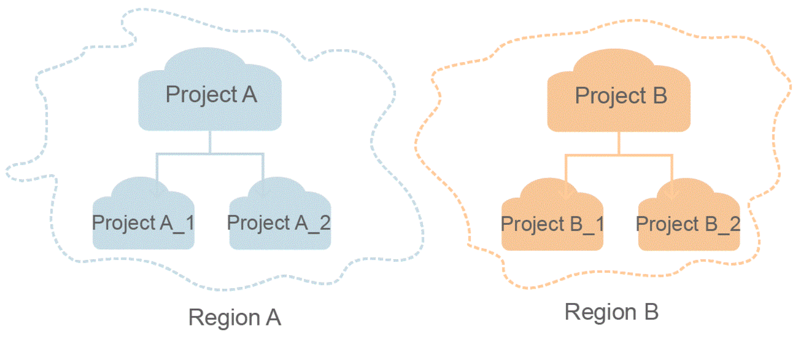
- AZ

  An AZ comprises of one or more physical data centers equipped with independent ventilation, fire, water, and electricity facilities. Computing, network, storage, and other resources in an AZ are logically divided into multiple clusters. AZs within a region are interconnected using high-speed optical fibers to support cross-AZ high-availability systems.

- Project

  Projects group and isolate resources (including compute, storage, and network resources) across physical regions. A default project is provided for each region, and subprojects can be created under each default project. Users can be granted permissions to access all resources in a specific project. For more refined access control, create subprojects under a project and purchase resources in the subprojects. Users can then be assigned permissions to access only specific resources in the subprojects.

Figure 1-1 Project isolating model



2 API Overview

FunctionGraph provides developers and partners with open APIs for development, deployment, hosting, and O&M, helping users quickly implement service innovations at low costs and shorten the rollout period of applications.

FunctionGraph provides the following types of APIs:

Table 2-1 API overview

Type	Description
Function Management Zone APIs	Query, create, modify, and publish functions.
Function Data Zone APIs	Implement synchronous or asynchronous function invocation.

Function Management Zone APIs

Table 2-2 Function management zone APIs

API	Description
Querying a Function List	Query a function list.
Querying the Metadata of a Function	Query the metadata of a function.
Querying the Code of a Function	Query the code of a function.
Creating a Function	Create a function.

API	Description
Deleting a Function or Function Version	Delete a function or function version except the LATEST version.
Modifying the Code of a Function	Modify the code of a function.
Modifying the Metadata of a Function	Modify the metadata of a function.
Publishing a Function Version	Publish a function version.
Querying the Versions of a Function	Query the versions of a function.
Creating an Alias for a Function Version	Create an alias for a function version.
Modifying the Alias Information About a Function Version	Modify the alias information about a function version.
Deleting an Alias of a Function Version	Delete an alias of a function version.
Querying the Alias Information About a Function Version	Query the alias information about a function version.
Querying the Version Alias List of a Function	Query the version alias list of a function.
Querying All Triggers of a Function	Query all triggers of a function.
Querying the Information About a Trigger	Query the information about a trigger.
Deleting All Triggers of a Function	Delete all triggers of a function.
Creating a Trigger	Create a trigger.

API	Description
Deleting a Trigger	Delete a trigger.

Function Data Zone APIs

Table 2-3 Function data zone APIs

API	Description
Implementing Synchronous Function Invocation	Implement synchronous function invocation.
Implementing Asynchronous Function Invocation	Implement asynchronous function invocation.

3 Calling APIs

3.1 Making an API Request

This section describes the structure of a REST API request, and uses the Identity and Access Management (IAM) API for obtaining a user token as an example to demonstrate how to call an API. The obtained token can then be used to authenticate the calling of other APIs.

Request URI

A request URI is in the following format:

{URI-scheme} :// {Endpoint} / {resource-path} ? {query-string}

Although a request URI is included in the request header, most programming languages or frameworks require the request URI to be transmitted separately.

Table 3-1 URI parameters

Parameter	Description
URI-scheme	Protocol used to transmit requests. All APIs use **HTTPS**.
Endpoint	Domain name or IP address of the server bearing the REST service. The endpoint varies between services in different regions. It can be obtained from **Regions and Endpoints**. For example, the endpoint of IAM in the **ae-ad-1** region is **iam.ae-ad-1.myhuaweicloud.com**.
resource-path	Resource path, that is, an API access path. Obtain the path from the URI of an API. For example, the **resource-path** of the API used to obtain a user token is **/v3/auth/tokens**.

Parameter	Description
query-string	Query parameter, which is optional. Query parameter, which is optional. Ensure that a question mark (?) is included before each query parameter that is in the format of "*Parameter name=Parameter value*". For example, **? limit=10** indicates that a maximum of 10 data records will be displayed.

NOTE

To simplify the URI display in this document, each API is provided only with a **resource-path** and a request method. The **URI-scheme** of all APIs is **HTTPS**, and the endpoints of all APIs in the same region are identical.

Request Methods

The HTTP protocol defines the following request methods that can be used to send a request to the server:

- **GET**: requests the server to return specified resources.
- **PUT**: requests the server to update specified resources.
- **POST**: requests the server to add resources or perform special operations.
- **DELETE**: requests the server to delete specified resources, for example, an object.
- **HEAD**: same as GET except that the server must return only the response header.
- **PATCH**: requests the server to update partial content of a specified resource. If the resource does not exist, a new resource will be created.

For example, in the case of the API used to obtain a user token, the request method is POST. The request is as follows:

```
POST https://iam.ae-ad-1.myhuaweicloud.com/v3/auth/tokens
```

Request Header

You can also add additional header fields to a request, such as the fields required by a specified URI or HTTP method. For example, to request for the authentication information, add **Content-Type**, which specifies the request body type.

Common request header fields are as follows:

- **Content-Type**: specifies the request body type or format. This field is mandatory and its default value is **application/json**. Other values of this field will be provided for specific APIs if any.
- **X-Auth-Token**: specifies a user token only for token-based API authentication. The user token is a response to the API used to obtain a user token. This API is the only one that does not require authentication.

NOTE

In addition to supporting token-based authentication, public cloud APIs also support authentication using access key ID/secret access key (AK/SK). During AK/SK-based authentication, an SDK is used to sign the request, and the **Authorization** (signature information) and **X-Sdk-Date** (time when the request is sent) header fields are automatically added to the request.

For more details, see **AK/SK-based Authentication**.

- **X-Project-ID**: specifies a subproject ID. This parameter is mandatory only in multi-project scenarios.
- **X-Domain-ID**: specifies an account ID.

The API used to obtain a user token does not require authentication. Therefore, only the **Content-Type** field needs to be added to requests for calling the API. An example of such requests is as follows:

```
POST https://iam.ae-ad-1.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
```

Request Body

The body of a request is often sent in a structured format as specified in the **Content-Type** header field. The request body transfers content except the request header.

The request body varies between APIs. Some APIs do not require the request body, such as the APIs requested using the GET and DELETE methods.

In the case of the API used to obtain a user token, the request parameters and parameter description can be obtained from the API request. The following provides an example request with a body included. Replace *username*, *domainname*, ******** (login password), and *xxxxxx* (project ID) with the actual values. To learn how to obtain a project ID, see **Regions and Endpoints**.

NOTE

The **scope** parameter specifies where a token takes effect. You can set **scope** to an account or a project under an account. In the following example, the token takes effect only for the resources in a specified project. For more information about this API, see Obtaining a User Token.

```
POST https://iam.ae-ad-1.myhuaweicloud.com/v3/auth/tokens

Content-Type: application/json

{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "username",
                    "password": "********",
                    "domain": {
                        "name": "domainname"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "name": "xxxxxxxxxxxxxxxxxx"
            }
        }
    }
}
```

If all data required for the API request is available, you can send the request to call the API through **curl**, **Postman**, or coding. In the response to the API used to obtain a user token, **x-subject-token** is the desired user token. This token can then be used to authenticate the calling of other APIs.

3.2 Authentication

Requests for calling an API can be authenticated using either of the following methods:

- Token-based authentication: Requests are authenticated using a token.
- AK/SK-based authentication: Requests are authenticated by encrypting the request body using an AK/SK.

Token-based Authentication

NOTE

The validity period of a token is 24 hours. When using a token for authentication, cache it to prevent frequently calling the Identity and Access Management (IAM) API used to obtain a user token.

A token specifies temporary permissions in a computer system. During API authentication using a token, the token is added to requests to get permissions for calling the API.

In **Making an API Request**, the process of calling the API used to obtain a user token is described. After a token is obtained, the **X-Auth-Token** header field must be added to requests to specify the token when other APIs are called. For example, if the token is **ABCDEFJ....**, **X-Auth-Token: ABCDEFJ....** can be added to a request as follows:

```
GET https://iam.ae-ad-1.myhuaweicloud.com/v3/auth/projects
Content-Type: application/json
X-Auth-Token: ABCDEFJ....
```

AK/SK-based Authentication

NOTE

AK/SK-based authentication supports API requests with a body not larger than 12 MB. For API requests with a larger body, token-based authentication is recommended.

In AK/SK-based authentication, AK/SK is used to sign requests and the signature is then added to the requests for authentication.

- AK: access key ID, which is a unique identifier used in conjunction with a secret access key to sign requests cryptographically.

- SK: secret access key used in conjunction with an AK to sign requests cryptographically. It identifies a request sender and prevents the request from being modified.

In AK/SK-based authentication, you can use an AK/SK to sign requests based on the signature algorithm or use the signing SDK to sign requests. For details about how to sign requests and use the signing SDK, see **API Request Signing Guide**.

NOTICE

The signing SDK is only used for signing requests and is different from the SDKs provided by services.

3.3 Response

Status Code

After sending a request, you will receive a response, including a status code, response header, and response body.

A status code is a group of digits, ranging from 1xx to 5xx. It indicates the status of a request. For more information, see **Status Codes**.

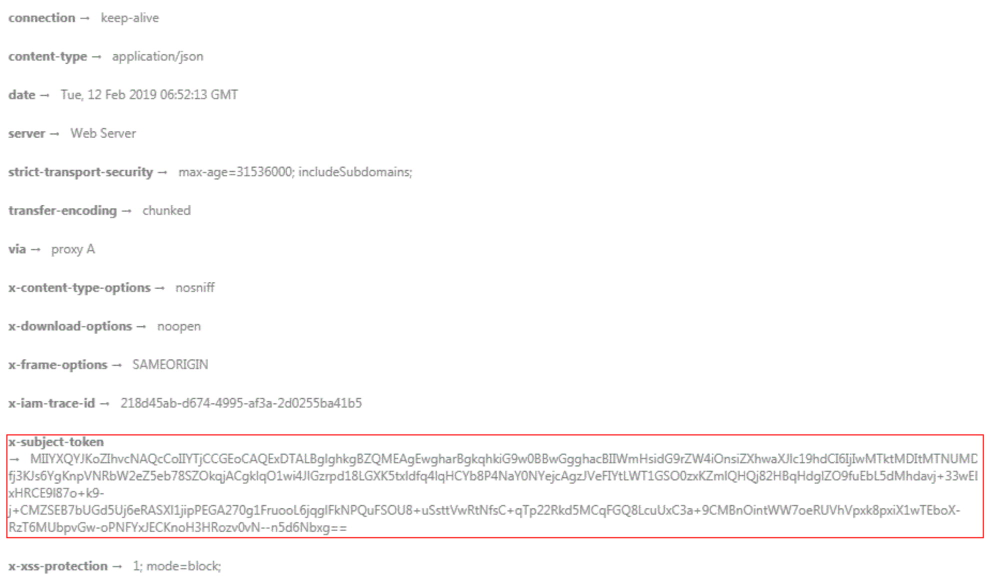
For example, if status code 201 is returned for calling the API used to obtain a user token, the request is successful.

Response Header

Similar to a request, a response also has a header, for example, **Content-Type**.

Figure 3-1 shows the response header fields for the API used to obtain a user token. The **x-subject-token** header field is the desired user token. This token can then be used to authenticate the calling of other APIs.

Figure 3-1 Header fields of the response to the request for obtaining a user token



```
connection → keep-alive

content-type → application/json

date → Tue, 12 Feb 2019 06:52:13 GMT

server → Web Server

strict-transport-security → max-age=31536000; includeSubdomains;

transfer-encoding → chunked

via → proxy A

x-content-type-options → nosniff

x-download-options → noopen

x-frame-options → SAMEORIGIN

x-iam-trace-id → 218d45ab-d674-4995-af3a-2d0255ba41b5

x-subject-token
→ MIIYXQYJKoZIhvcNAQcCoIIYTjCCGEoCAQExDTALBglghkgBZQMEAgEwgharBgkqhkiG9w0BBwGgghacBIIWmHsidG9rZW4iOnsiZXhwaXJlc19hdCI6IjIwMTktMDItMTNUMD
fj3KJs6YgKnpVNRbW2eZ5eb78SZOkqjACgklqO1wi4JIGzrpd18LGXK5txldfq4lqHCYb8P4NaY0NYejcAgzJVeFIYtLWT1GSO0zxKZmIQHQj82HBqHdglZO9fuEbL5dMhdavj+33wEl
xHRCE9I87o+k9-
j+CMZSEB7bUGd5Uj6eRASXl1jipPEGA270g1FruooL6jqglFkNPQuFSOU8+uSsttVwRtNfsC+qTp22Rkd5MCqFGQ8LcuUxC3a+9CMBnOintWW7oeRUVhVpxk8pxiX1wTEboX-
RzT6MUbpvGw-oPNFYxJECKnoH3HRozv0vN--n5d6Nbxg==

x-xss-protection → 1; mode=block;
```

Response Body

The body of a response is often returned in structured format as specified in the **Content-Type** header field. The response body transfers content except the response header.

The following is part of the response body for the API used to obtain a user token.

```
{
    "token": {
        "expires_at": "2019-02-13T06:52:13.855000Z",
        "methods": [
            "password"
        ],
        "catalog": [
            {
                "endpoints": [
                    {
                        "region_id": "XXXXXXXX",
......
```

If an error occurs during API calling, an error code and a message will be displayed. The following shows an error response body.

```
{
  "error_code": "FGS.0111",
  "error_msg": "xxxxxxxxx"
}
```

In the response body, **error_code** is an error code, and **error_msg** provides information about the error.

4 Function Model Definition

4.1 Function Model

Function Model

The function model of FunctionGraph is as follows:

```
{
  "functions": [
   {
    "func_urn": "urn:fss:xxxxxxxxx:7aad83af3e8d42e99ac194e8419e2c9b:function:default:test",
    "func_name": "test",
    "domain_id": "cff01_hk",
    "namespace": "7aad83af3e8d42e99ac194e8419e2c9b",
    "project_name": "xxxxxxxxxx",
    "package": "default",
    "runtime": "Node.js6.10",
    "timeout": 3,
    "handler": "test.handler",
    "memory_size": 128,
    "cpu": 300,
    "code_type": "inline",
    "code_url": "",
    "code_filename": "index.js",
    "code_size": 272,
    "user_data": "",
    "digest":
"decbce6939297b0b5ec6d1a23bf9c725870f5e69fc338a89a6a4029264688dc26338f56d08b6535d
e47f15ad538e22ca66613b9a46f807d50b687bb53fded1c6",
    "version": "latest",
    "image_name": "latest-5qe8e",
    "xrole": "cff",
    "app_xrole": null,
    "description": "111",
    "version_description": "",
    "last_modified": "2018-03-28T11:30:32+08:00",
"func_code": {
  "file": "",
  "link": ""
 },
 "func_vpc":null,
 "mount_config":null,
 "depend_list": null,
 "strategy_config": {
     "concurrency": -1
 },
 "extend_config": "",
 "dependencies": null,
"initializer_handler": "index.initializer",
"initializer_timeout": 3
   }
  ],
  "next_marker": 45
 }
```

Description

Table 4-1 describes the parameters in the function model.

Table 4-1 Parameters in the function model

Parameter	Description
func_urn	Function URN.
func_name	Function name.
domain_id	Tenant name.
namespace	Tenant's project ID.
project_name	Tenant's project name.
package	Group to which the function belongs. This field is defined to group functions.
runtime	Environment for executing the function. FunctionGraph supports Node.js 6.10, Node.js 8.10, Node.js 10.16, Node.js 12.13, Python 2.7, Python 3.6, Java 8, Go 1.8, C# (.NET Core 2.0), C# (.NET Core 2.1), C# (.NET Core 3.1), and PHP 7.3.
timeout	Maximum duration the function can be executed. Value range: 3s–900s.
handler	Handler of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.handler**, the file name is **myfunction.js**, and the entry point function is **handler**.
memory_size	Memory (MB) consumed by the function. Options: 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2560, 3072, 3584, and 4096.

Parameter	Description
cpu	Number of CPU millicores used by the function (1 core = 1000 millicores). The value of this field is proportional to that of **MemorySize**. By default, 100 CPU millicores are required for 128 MB memory. The value is calculated as follows: Memory/128 x 100 + 200 (basic CPU millicores).
code_type	Function code type. Options: ● **inline**: inline code ● **zip**: ZIP file ● **jar**: JAR file (mainly for Java functions) ● **obs**: function code stored in an Object Storage Service (OBS) bucket
code_url	● When **code_type** is set to **obs**, this parameter indicates the address of a function code package in OBS. ● When **code_type** is set to **inline**, **zip**, or **jar**, this parameter is left blank.
code_filename	Function file name. ● When **code_type** is set to **zip** or **jar**, this parameter is required. ● When **code_type** is set to **obs** or **inline**, this parameter is not required.
code_size	Code size in bytes.
user_data	Name/Value information defined for the function. For example, if a function needs to access a host, define Host={host_ip}. You can define a maximum of 20 such parameters, and their total length cannot exceed 4 KB.
digest	SHA512 hash value of function code, which is used to determine whether the function is changed.
version	Function version, which is automatically generated by the system. The version name is in the format of "vYYYYMMDD-HHMMSS" (v+year/month/day-hour/minute/second).
image_name	Internal identifier of a function version.
xrole	Agency used by the function. You need to create an agency on the Identity and Access Management (IAM) console. This field is mandatory when a function needs to access other services.

Parameter	Description
app_xrole	Agency used by the function app. You need to create an agency on the IAM console. This field is mandatory when a function needs to access other services.
description	Description of the function.
version_description	Description of the function version.
last_modified	Time when the function was last updated.
func_code	Function code. See **Table 4-2**.
depend_list	Dependency list.
strategy_config	Function policy configuration. See **Table 4-3**.
extend_config	Function extension configuration.
dependencies	Dependency list. See **Table 4-5**.
initializer_handler	Initializer of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.initializer**, the file name is **myfunction.js**, and the initialization function is **initializer**.
initializer_timeout	Maximum duration the function can be initialized. Value range: 1s–300s.
func_vpc	Virtual Private Cloud (VPC) configuration. See **Table 4-4**.
mount_config	File system configuration. See **Table 4-6**.

Table 4-2 func_code parameters

Parameter	Description
file	Function code. Nothing will be returned.
link	Function code link. Nothing will be returned.

Table 4-3 strategy_config parameter

Parameter	Description
concurrency	● **0**: The function is disabled. ● **-1**: The function is enabled.

Table 4-4 func_vpc parameters

Parameter	Type	Mandatory	Description
vpc_name	String	-	VPC name.
vpc_id	String	Yes when **func_vpc** is not empty.	VPC ID.
subnet_name	String	-	Subnet name.
subnet_id	String	Yes when **func_vpc** is not empty.	Subnet ID.
cidr	String	-	Subnet mask.
gateway	String	-	Gateway.

Table 4-5 dependency parameters

Parameter	Type	Mandatory	Description
owner	String	-	Domain ID of the dependency owner.
link	String	-	URL of the dependency package on OBS.
runtime	String	-	Language of the dependency package (only used for classification purposes).
etag	String	-	MD5 value of the dependency package.
size	Int	-	Size of the dependency package.
name	String	-	Name of the dependency package.
description	String	-	Description of the dependency package.

Parameter	Type	Mandatory	Description
file_name	String	-	File name of the dependency package (ZIP).

Table 4-6 mount_config parameters

Parameter	Type	Mandatory	Description
mount_user	**mount_user**	-	File system user configuration.
func_mounts	**func_mounts**	-	File system list.

Table 4-7 mount_user parameters

Parameter	Type	Mandatory	Description
user_id	Int	Yes when **mount_user** is not empty.	User ID, a non-0 integer from –1 to 65534.
user_group_id	Int	Yes when **mount_user** is not empty.	User group ID, a non-0 integer from –1 to 65534.

Table 4-8 func_mounts parameters

Parameter	Type	Mandatory	Description
mount_type	String	Yes when **func_mounts** is not empty.	Mount type. Options: **sfsTurbo** and **ecs**.
mount_resource	String	Yes when **func_mounts** is not empty.	ID of the mounted resource (corresponding cloud service).
mount_share_path	String	Yes when **mount_type** is set to **ecs**.	Remote mount path. Example: **192.168.0.12:/data**.
local_mount_path	String	Yes when **func_mounts** is not empty.	Function access path.

The format of a function URN is as follows:

```
urn:fss:<region_id>:<project_id>:function:<package>:<function_name>[:<version>|:!<alias>]
```

NOTE

A function URN is divided into eight fields by colons. The value of **region_id** is included in the system configuration. You can set this parameter to the same as that in the backend. The content in the brackets ([]) is a function version or alias. If you enter an alias, add an exclamation mark (!) in front of it for easy identification.

When a function URN is used as an API parameter, you can provide it in a simplified format as follows:

- 1 field: **<function_name>**. **project_id** is obtained from a token, **package** is **default**, and **version** is **latest**.
- 2 fields: **<package>:<function_name>**. **project_id** is obtained from a token, and **version** is **latest**.
- 3 fields: **<project_id>:<package>:<function_name>**. **version** is **latest**.
- 4 fields: **<project_id>:<package>:<function_name>:<Version or Alias>**.
- 7 fields: **urn:fss:<region_id>:<project_id>:function:<package>:<function_name>**. **version** is **latest**.
- 8 fields: **urn:fss:<region_id>:<project_id>:function:<package>:<function_name>:<Version or Alias>**.

4.2 Trigger Management Models

Trigger Type Model

```
{
   "trigger_type_code":"string",
   "display_name":"string",
   "status":"string",
   "event_codes":"array of string",
   "description":"string"
}
```

Table 4-9 describes the parameters in the trigger type model.

Table 4-9 Parameters in the trigger type model

Parameter	Description
trigger_type_code	Trigger type code. Options: SMN, OBS, TIMER, CTS, and kafka.
display_name	Trigger type value.
status	Trigger type status. Options: ● **DISABLED**: The trigger is disabled. ● **TEST**: The trigger is under test and invisible to clients. ● **ACTIVE**: The trigger is available.

Parameter	Description
description	Trigger description.

Trigger Instance Model

```
{
    "trigger_id":"string",
    "trigger_type_code":"string",
    "event_type_code":"string",
    "status":"string",
    "event_data":"json struct",
    "last_updated_time":"string",
    "created_time":"string"
}
```

Table 4-10 describes the parameters in the trigger instance model.

Table 4-10 Parameters in the trigger instance model

Parameter	Description
trigger_id	Trigger ID.
trigger_type_code	Trigger type code. Options: SMN, OBS, TIMER, CTS, and kafka.
event_type_code	Event type code. This parameter is mandatory. It can be a non-null character string. This parameter is not used currently.
status	Trigger status. Options: **ACTIVE** and **DISABLED**.
event_data	Trigger data defined in JSON format.
last_updated_time	Time when the trigger was last updated.
created_time	Time when the trigger was created.

Trigger Instance Data

- The data of a Simple Message Notification (SMN) trigger is as follows:

  ```
  {
      "topic_urn":"string",
      "subscription_status":"string"
  }
  ```

  Table 4-11 describes the parameters of an SMN trigger.

Table 4-11 Parameters of an SMN trigger

Parameter	Description
topic_urn	URN of an SMN topic. This parameter is mandatory when you create an SMN trigger.
subscription_status	Subscription status of a topic. Options: **Unconfirmed** and **Confirmed**.

- The data of an Object Storage Service (OBS) trigger is as follows:

```
{
  "bucket": "yourBucketName",
  "events": ["s3:ObjectCreated:Put"],
  "prefix": "yourPrefix",
  "suffix": "yourSuffix"
}
```

Table 4-12 Parameters of an OBS trigger

Parameter	Description
bucket	Bucket name. This parameter is mandatory.
events	Collection of OBS trigger events. Options: **s3:ObjectCreated:***, **s3:ObjectCreated:Put**, **s3:ObjectCreated:Post**, **s3:ObjectCreated:Copy**, **s3:ObjectCreated:CompleteMultipartUpload**, **s3:ObjectRemoved:***, **s3:ObjectRemoved:DeleteMarkerCreated**, and **s3:ObjectRemoved:Delete**. This parameter is mandatory. Note that **s3:objectcreated:*** includes all events that start with **s3:objectcreated**, and **s3:objectremoved:*** includes all events that start with **s3:objectremoved**.
prefix	Prefix of an OBS object. This parameter is optional.
suffix	Suffix of an OBS object. This parameter is optional.

- The data of a timer trigger is as follows:

```
{
   "name": "string",
   "schedule_type": "string",
   "schedule": "string",
   "user_event": "string"
}
```

Table 4-13 describes the parameters of a timer trigger.

Table 4-13 Parameters of a timer trigger

Parameter	Description
name	Trigger name. This parameter is mandatory.

Parameter	Description
schedule_type	Schedule type. Options: **Rate** or **Cron**. This parameter is mandatory.
schedule	Schedule setting, which varies depending on the schedule type you choose. This parameter is mandatory. When **schedule_type** is set to **Rate**, add unit m, h, or d behind a rate, for example, **3m** for 3 minutes.
user_event	Additional information for calling a function. This parameter is optional.

- The data of a Cloud Trace Service (CTS) trigger is as follows:

```
{
   "name": "eqwrwe",
   "operations": ["AAD:addprotocolrule:addProtocolRule", "BCS:baas-apiserver:scalePeers",
"ARS:ars:setConfigArs"]
}
```

  Table 4-14 describes the parameters of a CTS trigger.

Table 4-14 Parameters of a CTS trigger

Parameter	Description
name	Name of a key notification.
operations	Operation list. The format is "service type:resource type A;resource type B:operation 1;operation 2". Example: ["ECS:ecs;server:restartServer;deleteServer",...].

- The data of a Document Database Service (DDS) trigger is as follows:

```
{
   "instance_id": "string",
       "collection_name": "string",
       "db_name": "string",
   "db_password": string,
       "batch_size": int,
}
```

Table 4-15 Parameters of a DDS trigger

Parameter	Description
instance_id	DB instance ID.
collection_name	Collection name.
db_name	Database name.
db_password	Password for logging in to the database.
batch_size	Batch size.

- The data of a Kafka trigger is as follows:

```
{
        "instance_id": "string",
    "db_name": "string",
        "collection_name": "string",
        "db_user": "string",
    "db_password": string,
        "batch_size": int,
}
```

Table 4-16 Parameters of a Kafka trigger

Parameter	Description
instance_id	Kafka instance ID.
topic_id	Topic ID.
kafka_user	Username.
kafka_password	Password.
kafka_ssl_enable	Whether to enable SSL authentication. If SSL authentication is enabled, the **kafka_user** and **kafka_password** fields are mandatory.
batch_size	Batch size.

5 Function Management Zone APIs

5.1 Querying a Function List

Function

This API is used to query a function list.

URI

GET /v2/{project_id}/fgs/functions?marker={marker}&maxitems={maxitems}

Table 5-1 describes the URI parameter.

Table 5-1 URI parameter

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.

Request

Table 5-2 describes the request parameters.

Table 5-2 Request parameters

Parameter	Type	Mandatory	Description
marker	Int	No	Final record queried last time.

Parameter	Type	Mandatory	Description
maxitems	Int	No	Maximum number of function templates that can be queried each time. The maximum value is 400. If this parameter is not set or is 0 or greater than 400, the default value 400 is used. If this parameter is less than 0, a message indicating a parameter error is returned.

Response

Table 5-3 describes the response parameters.

Table 5-3 Response parameters

Parameter	Type	Description
func_urn	String	Function URN.
func_name	String	Function name.
domain_id	String	Domain ID.
namespace	String	Tenant's project ID.
project_name	String	Tenant's project name.
package	String	Group to which the function belongs. This field is defined to group functions.
runtime	String	Environment for executing the function. FunctionGraph supports Node.js 6.10, Node.js 8.10, Node.js 10.16, Node.js 12.13, Python 2.7, Python 3.6, Java 8, Go 1.8, C# (.NET Core 2.0), C# (.NET Core 2.1), C# (.NET Core 3.1), and PHP 7.3.
timeout	Int	Maximum duration the function can be executed. Value range: 3s–900s.
handler	String	Handler of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.handler**, the file name is **myfunction.js**, and the entry point function is **handler**.
memory_size	Int	Memory (MB) consumed by the function. Options: 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2560, 3072, 3584, and 4096.

Parameter	Type	Description
cpu	Int	Number of CPU millicores used by the function (1 core = 1000 millicores). The value of this field is proportional to that of **MemorySize**. By default, 100 CPU millicores are required for 128 MB memory. The value is calculated as follows: Memory/128 x 100 + 200 (basic CPU millicores).
code_type	String	Function code type. Options: ● **inline**: inline code ● **zip**: ZIP file ● **jar**: JAR file (mainly for Java functions) ● **obs**: function code stored in an Object Storage Service (OBS) bucket
code_url	String	● When **code_type** is set to **obs**, this parameter indicates the address of a function code package in OBS. ● When **code_type** is set to **inline**, **zip**, or **jar**, this parameter is left blank.
code_filename	String	Function file name. ● When **code_type** is set to **zip** or **jar**, this parameter is required. ● When **code_type** is set to **obs** or **inline**, this parameter is not required.
code_size	Int64	Code size in bytes.
user_data	String	Name/Value information defined for the function. For example, if a function needs to access a host, define Host={host_ip}. You can define a maximum of 20 such parameters, and their total length cannot exceed 4 KB.
digest	String	SHA512 hash value of function code, which is used to determine whether the function is changed.
version	String	Function version, which is automatically generated by the system. The version name is in the format of "vYYYYMMDD-HHMMSS" (v+year/month/day-hour/minute/second).
image_name	String	Internal identifier of a function version.

Parameter	Type	Description
xrole	String	Agency used by the function. You need to create an agency on the Identity and Access Management (IAM) console. This field is mandatory when a function needs to access other services.
app_xrole	*String	Agency used by the function app. You need to create an agency on the IAM console. This field is mandatory when a function needs to access other services.
description	String	Description of the function.
version_description	String	Description of the function version.
last_modified	String	Time when the function was last updated.
func_code	String	Function code. See **Table 5-4**.
depend_list	[]String	Dependency list.
strategy_config	String	Function policy configuration. See **Table 5-5**.
extend_config	String	Function extension configuration.
dependencies	[]*String	Dependency code package.
initializer_handler	String	Initializer of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.initializer**, the file name is **myfunction.js**, and the initialization function is **initializer**.
initializer_timeout	Int	Maximum duration the function can be initialized. Value range: 1s–300s.
func_vpc	*String	Virtual Private Cloud (VPC) configuration. See **Table 5-6**.
mount_config	*String	Disk mount configuration. See **Table 5-7**.

Table 5-4 func_code parameters

Parameter	Type	Description
file	String	Function code (deprecated).
link	String	Function code link (code can be downloaded through the OBS SDK).

Table 5-5 strategy_config parameter

Parameter	Type	Description
concurrenc y	Int	● **0**: The function is disabled. ● **-1**: The function is enabled.

Table 5-6 func_vpc parameters

Parameter	Type	Mandatory	Description
vpc_name	String	-	VPC name.
vpc_id	String	-	VPC ID.
subnet_name	String	-	Subnet name.
subnet_id	String	-	Subnet ID.
cidr	String	-	Subnet mask.
gateway	String	-	Gateway.

Table 5-7 mount_config parameters

Parameter	Type	Mandatory	Description
mount_user	*String	Yes	User configuration. See **Table 5-8**.
func_mounts	[]*String	Yes	Function configuration. See **Table 5-9**.

Table 5-8 mount_user parameters

Parameter	Type	Mandatory	Description
user_id	Int	Yes when **mount_user** is not empty.	User ID, a non-0 integer from –1 to 65534.
user_group_id	Int	Yes when **mount_user** is not empty.	User group ID, a non-0 integer from –1 to 65534.

Table 5-9 func_mounts parameters

Parameter	Type	Mandatory	Description
id	String	-	Unique ID that identifies a file system.
mount_type	String	Yes when **func_mounts** is not empty.	Mount type. Options: **sfsTurbo** and **ecs**.
mount_resource	String	Yes when **func_mounts** is not empty.	ID of the mounted resource (corresponding cloud service).
mount_share_path	String	Yes when **mount_type** is set to **ecs**.	Remote mount path. Example: 192.168.0.12:/data.
local_mount_path	String	Yes when **func_mounts** is not empty.	Function access path.
status	String	-	Status. Options: **ACTIVE** and **DISABLED**.

Example

Example request

```
GET /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions?marker=0&maxitems=400
HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200 OK
{
  "functions": [
   {
    "func_urn": "urn:fss:xxxxxxxxx:7aad83af3e8d42e99ac194e8419e2c9b:function:default:test",
    "func_name": "test",
    "user_domain": "cff01_hk",
    "namespace": "7aad83af3e8d42e99ac194e8419e2c9b",
    "project_name": "xxxxxxxx",
    "package": "default",
    "runtime": "Node.js6.10",
    "timeout": 3,
    "handler": "test.handler",
    "memory_size": 128,
    "cpu": 300,
    "code_type": "inline",
     "code_filename": "index.js",
    "code_size": 272,
    "digest":
"decbce6939297b0b5ec6d1a23bf9c725870f5e69fc338a89a6a4029264688dc26338f56d08b6535d
e47f15ad538e22ca66613b9a46f807d50b687bb53fded1c6",
    "version": "latest",
    "image_name": "latest-5qe8e",
    "xrole": "cff",
    "description": "111",
    "last_modified": "2018-03-28T11:30:32+08:00",
    "func_code": {},
    "strategy_config": {
    "concurrency": -1,
    "initializer_handler": "index.initializer",
    "initializer_timeout": 3
    }
 ],
 "next_marker": 45
 }
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 403 Forbidden
{
  "error_code": "FSS.0403",
  "error_msg": "namespace and token mismatch"
 }
```

Status Code

See **Status Codes**.

5.2 Querying the Metadata of a Function

Function

This API is used to query the metadata of a function.

URI

GET /v2/{project_id}/fgs/functions/{function_urn}/config

Table 5-10 describes the URI parameters.

Table 5-10 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**.

Request

None

Response

Table 5-11 describes the response parameters.

Table 5-11 Response parameters

Parameter	Type	Description
func_urn	String	Function URN.
func_name	String	Function name.
domain_id	String	Domain ID.
namespace	String	Tenant's project ID.
project_name	String	Tenant's project name.
package	String	Group to which the function belongs. This field is defined to group functions.
runtime	String	Environment for executing the function. FunctionGraph supports Node.js 6.10, Node.js 8.10, Node.js 10.16, Node.js 12.13, Python 2.7, Python 3.6, Java 8, Go 1.8, C# (.NET Core 2.0), C# (.NET Core 2.1), C# (.NET Core 3.1), and PHP 7.3.
timeout	Int	Maximum duration the function can be executed. Value range: 3s–900s.
handler	String	Handler of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.handler**, the file name is **myfunction.js**, and the entry point function is **handler**.
memory_size	Int	Memory (MB) consumed by the function. Options: 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2560, 3072, 3584, and 4096.
cpu	Int	Number of CPU millicores used by the function (1 core = 1000 millicores). The value of this field is proportional to that of **MemorySize**. By default, 100 CPU millicores are required for 128 MB memory. The value is calculated as follows: Memory/128 x 100 + 200 (basic CPU millicores).

Parameter	Type	Description
code_type	String	Function code type. Options: ● **inline**: inline code ● **zip**: ZIP file ● **jar**: JAR file (mainly for Java functions) ● **obs**: function code stored in an Object Storage Service (OBS) bucket
code_url	String	● When **code_type** is set to **obs**, this parameter indicates the address of a function code package in OBS. ● When **code_type** is set to **inline**, **zip**, or **jar**, this parameter is left blank.
code_filename	String	Function file name. ● When **code_type** is set to **zip** or **jar**, this parameter is required. ● When **code_type** is set to **obs** or **inline**, this parameter is not required.
code_size	Int64	Code size in bytes.
user_data	String	Name/Value information defined for the function. For example, if a function needs to access a host, define Host={host_ip}. You can define a maximum of 20 such parameters, and their total length cannot exceed 4 KB.
digest	String	SHA512 hash value of function code, which is used to determine whether the function is changed.
version	String	Function version, which is automatically generated by the system. The version name is in the format of "vYYYYMMDD-HHMMSS" (v+year/month/day-hour/minute/second).
image_name	String	Internal identifier of a function version.
xrole	String	Agency used by the function. You need to create an agency on the Identity and Access Management (IAM) console. This field is mandatory when a function needs to access other services.
app_xrole	String	Agency used by the function app. You need to create an agency on the IAM console. This field is mandatory when a function needs to access other services.

Parameter	Type	Description
description	String	Description of the function.
version_description	String	Description of the function version.
last_modified	String	Time when the function was last updated.
depend_list	[]String	Dependency list.
strategy_config	String	Function policy configuration. See **Table 5-12**.
extend_config	String	Function extension configuration.
initializer_handler	String	Initializer of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.initializer**, the file name is **myfunction.js**, and the initialization function is **initializer**.
initializer_timeout	Int	Maximum duration the function can be initialized. Value range: 1s–300s.
func_vpc	*String	Virtual Private Cloud (VPC) configuration. See **Table 5-6**.
mount_config	*String	Disk mount configuration. See **Table 5-7**.

Table 5-12 strategy_config parameter

Parameter	Type	Mandatory	Description
concurrency	Int	Yes	● **0**: The function is disabled. ● **-1**: The function is enabled.

Example

Example request

```
GET /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/config HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200 OK
{
  "code_filename": "index.js",
  "code_size": 272,
  "code_type": "inline",
  "cpu": 300,
  "description": "111",
  "digest":
"decbce6939297b0b5ec6d1a23bf9c725870f5e69fc338a89a6a4029264688dc26338f56d08b6535d
e47f15ad538e22ca66613b9a46f807d50b687bb53fded1c6",
  "func_name": "test",
  "func_urn": "urn:fss:xxxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest",
  "handler": "test.handler",
  "image_name": "latest-5qe8e",
  "last_modified": "2018-03-28T11:30:32+08:00",
  "memory_size": 128,
  "namespace": "7aad83af3e8d42e99ac194e8419e2c9b",
  "package": "default",
  "project_name": "xxxxxxxxx",
  "runtime": "Node.js6.10",
  "timeout": 3,
  "user_domain": "cff01_hk",
  "version": "latest",
  "xrole": "cff",
  "strategy_config": {
      "concurrency": -1
  },
 }
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1051",
  "error_msg": "Not found the function"
 }
```

Status Code

See **Status Codes**.

5.3 Querying the Code of a Function

Function

This API is used to query the code of a function.

URI

GET /v2/{project_id}/fgs/functions/{function_urn}/code

Table 5-13 describes the URI parameters.

Table 5-13 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.

Parameter	Type	Mandatory	Description
function_urn	String	Yes	Function URN. See **Function Model**.

Request

None

Response

Table 5-14 describes the response parameters.

Table 5-14 Response parameters

Parameter	Type	Description
func_urn	String	Function URN.
func_name	String	Function name.
domain_id	String	Domain ID.
runtime	String	Environment for executing the function. FunctionGraph supports Node.js 6.10, Node.js 8.10, Node.js 10.16, Node.js 12.13, Python 2.7, Python 3.6, Java 8, Go 1.8, C# (.NET Core 2.0), C# (.NET Core 2.1), C# (.NET Core 3.1), and PHP 7.3.
concurrency	Int	• **0**: The function is disabled. • **-1**: The function is enabled.
code_type	String	Function code type. Options: • **inline**: inline code • **zip**: ZIP file • **jar**: JAR file (mainly for Java functions) • **obs**: function code stored in an Object Storage Service (OBS) bucket
code_url	String	• When **code_type** is set to **obs**, this parameter indicates the address of a function code package in OBS. • When **code_type** is set to **inline**, **zip**, or **jar**, this parameter is left blank.

Parameter	Type	Description
code_filename	String	Function file name. ● When **code_type** is set to **zip** or **jar**, this parameter is required. ● When **code_type** is set to **obs** or **inline**, this parameter is not required.
code_size	String	Code size in bytes.
func_code	String	Function code. See **Table 5-15**.
digest	String	SHA512 hash value of function code, which is used to determine whether the function is changed.
last_modified	String	Time when the function was last updated.
depend_list	String	Dependency list.
strategy_config	String	Function policy configuration. See **Table 5-16**.
func_vpc	func_vpc	Virtual Private Cloud (VPC) configuration. See **Table 5-6**.

Table 5-15 func_code parameters

Parameter	Type	Description
file	String	Function code (deprecated).
link	String	Function code link (code can be downloaded through the OBS SDK).

Table 5-16 strategy_config parameter

Parameter	Type	Description
concurrency	Int	● **0**: The function is disabled. ● **-1**: The function is enabled.

Example

Example request

```
GET /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/code HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200 OK
{
  "code_filename": "index.js",
  "code_size": 272,
  "code_type": "inline",
  "digest":
"decbce6939297b0b5ec6d1a23bf9c725870f5e69fc338a89a6a4029264688dc26338f56d08b6535d
e47f15ad538e22ca66613b9a46f807d50b687bb53fded1c6",
  "func_code": {
   "file": "",
   "link": "https://functionstorage-hk06.obs.xx-xxx.xxxxxxxxcloud.com/xxx/d2b0xxxf6e65/default/
test143/latest/index.zip"
  },
  "func_name": "test",
  "func_urn": "urn:fss:xxxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest",
  "last_modified": "2018-03-28T11:30:32+08:00",
  "runtime": "Node.js6.10",
  "strategy_config": {
      "concurrency": -1
  },
 }
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1052",
  "error_msg": "Not found the function version"
 }
```

Status Code

See **Status Codes**.

5.4 Creating a Function

Function

This API is used to create a function.

URI

POST /v2/{project_id}/fgs/functions

Table 5-17 describes the URI parameter.

Table 5-17 URI parameter

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID

Request

Table 5-18 describes the request parameters.

Table 5-18 Request parameters

Parameter	Type	Mandatory	Description
func_name	String	Yes	Function name.
package	String	Yes	Group to which the function belongs. Default value: **default**. You can customize the value as required.
code_type	String	Yes	Function code type. Options: ● **inline**: inline code ● **zip**: ZIP file ● **jar**: JAR file (mainly for Java functions) ● **obs**: function code stored in an Object Storage Service (OBS) bucket
code_url	String	No	● When **code_type** is set to **obs**, this parameter indicates the address of a function code package in OBS. ● When **code_type** is set to **inline**, **zip**, or **jar**, this parameter is left blank.
description	String	No	Description of the function.
code_filename	String	No	Code file name. ● When **code_type** is set to **zip** or **jar**, this parameter is required. ● When **code_type** is set to **obs** or **inline**, this parameter is not required.
handler	String	Yes	Entry point of the function.
memory_size	Int	Yes	Memory (MB) consumed by the function. Options: 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2560, 3072, 3584, and 4096.
runtime	String	Yes	Environment for executing the function. FunctionGraph supports Node.js 6.10, Node.js 8.10, Node.js 10.16, Node.js 12.13, Python 2.7, Python 3.6, Java 8, Go 1.8, C# (.NET Core 2.0), C# (.NET Core 2.1), C# (.NET Core 3.1), and PHP 7.3.

Parameter	Type	Mandatory	Description
timeout	Int	Yes	Timeout allowed for the function. Value range: 3s–900s.
user_data	String	No	Name/Value information defined for the function.
xrole	String	No	This parameter is mandatory if the function needs to access other cloud services.
func_code.file	String	Yes	Function code. ● This parameter is mandatory when **code_type** is set to **inline**, **zip**, or **jar**. Moreover, the code must be encoded using Base64. ● This parameter is optional when **code_type** is set to **obs**.

Response

Table 5-19 describes the response parameters.

Table 5-19 Response parameters

Parameter	Type	Description
func_urn	String	Function URN.
func_name	String	Function name.
domain_id	String	Domain ID.
namespace	String	Tenant's project ID.
project_name	String	Tenant's project name.
package	String	Group to which the function belongs. This field is defined to group functions.
runtime	String	Environment for executing the function. FunctionGraph supports Node.js 6.10, Node.js 8.10, Node.js 10.16, Node.js 12.13, Python 2.7, Python 3.6, Java 8, Go 1.8, C# (.NET Core 2.0), C# (.NET Core 2.1), C# (.NET Core 3.1), and PHP 7.3.
timeout	Int	Maximum duration the function can be executed. Value range: 3s–900s.

Parameter	Type	Description
handler	String	Handler of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.handler**, the file name is **myfunction.js**, and the entry point function is **handler**.
memory_size	Int	Memory (MB) consumed by the function. Options: 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2560, 3072, 3584, and 4096.
cpu	Int	Number of CPU millicores used by the function (1 core = 1000 millicores). The value of this field is proportional to that of **MemorySize**. By default, 100 CPU millicores are required for 128 MB memory. The value is calculated as follows: Memory/128 x 100 + 200 (basic CPU millicores).
code_type	String	Function code type. Options: ● **inline**: inline code ● **zip**: ZIP file ● **jar**: JAR file (mainly for Java functions) ● **obs**: function code stored in an OBS bucket
code_url	String	● When **code_type** is set to **obs**, this parameter indicates the address of a function code package in OBS. ● When **code_type** is set to **inline**, **zip**, or **jar**, this parameter is left blank.
code_filename	String	Function file name. ● When **code_type** is set to **zip** or **jar**, this parameter is required. ● When **code_type** is set to **obs** or **inline**, this parameter is not required.
code_size	Int64	Code size in bytes.
user_data	String	Name/Value information defined for the function. For example, if a function needs to access a host, define Host={host_ip}. You can define a maximum of 20 such parameters, and their total length cannot exceed 4 KB.

Parameter	Type	Description
digest	String	SHA512 hash value of function code, which is used to determine whether the function is changed.
version	String	Function version, which is automatically generated by the system. The version name is in the format of "vYYYYMMDD-HHMMSS" (v+year/month/day-hour/minute/second).
image_name	String	Internal identifier of a function version.
xrole	String	Agency used by the function. You need to create an agency on the Identity and Access Management (IAM) console. This field is mandatory when a function needs to access other services.
app_xrole	*String	Agency used by the function app. You need to create an agency on the IAM console. This field is mandatory when a function needs to access other services.
description	String	Description of the function.
version_descri ption	String	Description of the function version.
last_modified	String	Time when the function was last updated.
func_vpc	*String	Virtual Private Cloud (VPC) configuration. See **Table 5-6**.
func_code	String	Function code. See **Table 5-4**.
depend_list	[]String	Dependency list.
strategy_confi g	String	Function policy configuration. See **Table 5-20**.
extend_config	String	Function extension configuration.
dependencies	[]*String	Dependency code package.
initializer_han dler	String	Initializer of the function.
initializer_tim eout	Int	Maximum duration the function can be initialized. Value range: 1s–300s.

Table 5-20 strategy_config parameter

Parameter	Type	Mandatory	Description
concurrency	Int	Yes	● **0**: The function is disabled. ● **-1**: The function is enabled.

Example

Example request

```
POST /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions HTTP/1.1
 {
   "func_name": "test",
   "package": "default",
   "description": "",
   "handler": "test.handler",
   "memory_size": 128,
   "timeout": 3,
   "runtime": "Python2.7",
   "user_data": "",
   "code_type": "inline",
   "func_code": {
     "file": 
"aW1wb3J0IGpzb24KZGVmIGhhbmRsZXIgKGV2ZW50LCBjb250ZXh0KToKICAgIG91dHB1dCA9ICdI
ZWxsbyBtZXNzYWdlOiAnICsganNvbi5kdW1wcyhldmVudCkKICAgIHJldHVybiBvdXRwdXQ="
   }
 }
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200 OK
{
  "func_urn": 
"urn:fss:xxxxxxxxx:c3b2459a6d5e4b548e6777e57852692d:function:default:TestCreateFunctionInP
ythonSdk:latest",
  "func_name": "TestCreateFunctionInPythonSdk",
  "user_domain": "FGS_hwx559619",
  "namespace": "c3b2459a6d5e4b548e6777e57852692d",
  "project_name": "xxxxxxxxxx",
  "package": "default",
  "runtime": "Python3.6",
  "timeout": 30,
  "handler": "index.handler",
  "memory_size": 128,
  "cpu": 300,
  "code_type": "inline",
  "code_filename": "index.py",
  "code_size": 110,
  "digest": 
"1c8610d1870731a818a037f1d2adf3223e8ac351aeb293fb1f8eabd2e9820069a61ed8b5d38182e7
60adc33a307d0e957afc357f415cd8c9c3ff6f0426fd85cd",
  "version": "latest",
  "image_name": "latest-0zf5g",
  "last_modified": "2019-03-07T18:37:19+08:00",
  "concurrency": 0,
"strategy_config": {
  "concurrency": -1
 },
}
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 409 Conflict
{
  "error_code": "FSS.1061",
  "error_msg": "The function has existed"
 }
```

Status Code

See **Status Codes**.

5.5 Deleting a Function or Function Version

Function

This API is used to delete a function or function version except the LATEST version. Specifically:

- If the URN contains a function version or alias, the function version or the version corresponding to the specified alias as well as associated triggers will be deleted.
- If the URN does not contain a function version or alias, the entire function as well as all its versions, aliases, and triggers will be deleted.

URI

DELETE /v2/{project_id}/fgs/functions/{function_urn}

Table 5-21 describes the URI parameters.

Table 5-21 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**. **NOTE** The LATEST version of a function cannot be deleted. To delete a function and all its versions, provide a URN without any version or alias. Example: urn:fss:xxxxxxxx:7aad83af3e8d42e99ac194e8419e2c9b:function:default:test.

Request

None

Response

None

Example

Example request

```
DELETE  /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:v20170830-181539 HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 204
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1051",
  "error_msg": "Not found the function"
 }
```

Status Code

See **Status Codes**.

5.6 Modifying the Code of a Function

Function

This API is used to modify the code of a function.

URI

PUT /v2/{project_id}/fgs/functions/{function_urn}/code

Table 5-22 describes the URI parameters.

Table 5-22 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Table 4-1**.

Request

Table 5-23 describes the request parameters.

Table 5-23 Request parameters

Parameter	Type	Mandatory	Description
code_type	String	Yes	Code type of a function. See **Table 4-1**.
code_url	String	No	Enter the address of the function code package in Object Storage Service (OBS). This parameter is mandatory when **code_type** is set to **obs**.
func_code.file	String	No	Function code. ● This parameter is mandatory when **code_type** is set to **inline**, **zip**, or **jar**. Moreover, the code must be encoded using Base64. ● This parameter is optional when **code_type** is set to **obs**.
depend_list	[]*String	No	Dependencies of the function.
code_filename	String	No	Function file name, which consists of a file name and file type.

Response

Table 5-24 describes the response parameters.

Table 5-24 Response parameters

Parameter	Type	Description
func_urn	String	Function URN.
func_name	String	Function name.
domain_id	String	Domain ID.
runtime	String	Environment for executing the function. FunctionGraph supports Node.js 6.10, Node.js 8.10, Node.js 10.16, Node.js 12.13, Python 2.7, Python 3.6, Java 8, Go 1.8, C# (.NET Core 2.0), C# (.NET Core 2.1), C# (.NET Core 3.1), and PHP 7.3.
concurrency	Int	● **0**: The function is disabled. ● **-1**: The function is enabled.

Parameter	Type	Description
code_type	String	Function code type. Options: ● **inline**: inline code ● **zip**: ZIP file ● **jar**: JAR file (mainly for Java functions) ● **obs**: function code stored in an Object Storage Service (OBS) bucket
code_url	String	● When **code_type** is set to **obs**, this parameter indicates the address of a function code package in OBS. ● When **code_type** is set to **inline**, **zip**, or **jar**, this parameter is left blank.
code_filename	String	Function file name. ● When **code_type** is set to **zip** or **jar**, this parameter is required. ● When **code_type** is set to **obs** or **inline**, this parameter is not required.
code_size	String	Code size in bytes.
func_code	String	Function code. See **Table 5-25**.
digest	String	SHA512 hash value of function code, which is used to determine whether the function is changed.
last_modified	String	Time when the function was last updated.
depend_list	[]String	Dependency list.
strategy_config	String	Function policy configuration. See **Table 5-26**.
dependencies	[]dependency	Dependency code package.
func_vpc	func_vpc	Virtual Private Cloud (VPC) configuration. See **Table 5-6**.

Table 5-25 func_code parameters

Parameter	Type	Description
file	String	Function code (deprecated).
link	String	Function code link (code can be downloaded through the OBS SDK).

Table 5-26 strategy_config parameter

Parameter	Type	Mandatory	Description
concurrency	Int	Yes	● **0**: The function is disabled. ● **-1**: The function is enabled.

Example

Example request

```
PUT /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/code HTTP/1.1
{
  "code_type": "inline",
  "func_code": {
    "file":
"aW1wb3J0IGpzb24KZGVmIGhhbmRsZXIoZXZlbnQsIGNvbnRleHQpOgogICAgb3V0cHV0ID0gJ0hl
bGxvIE1zZyBmb3IgbW9kaWZ5OiAnICsganNvbi5kdW1wcyhldmVudCkKICAgIGFrID0gY29udGV4d
C5nZXRBY2Nlc3NLZXkoKQogICAgc2sgPSBjb250ZXh0LmdldFNlY3JldEtleSgpCiAgICB0b2tlbiA9IGN
vbnRleHQuZ2V0VG9rZW4oKQogICAgcHJpbnQgJ2FrOicgKyBhawogICAgcHJpbnQgJ3NrOicgKyBza
wogICAgcHJpbnQgJ3Rva2VuOicgKyB0b2tlbgogICAgcmV0dXJuIG91dHB1dAo=",
  },
  "strategy_config": {
   "concurrency": -1
  },
}
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200 OK
{
  "code_filename": "index.js",
  "code_size": 273,
  "code_type": "inline",
  "code_url": "",
  "digest":
"af40294713c964d24f52fd567022cb7e03373b8acfafc2526bacde08a864e21dd214ad4fe567cd8a6
541822ee76171ca802da6e7d135c07689a6072930e09824",
  "func_code": {
    "file": "",
   "link": "https://functionstorage-hk06.obs.xx-xxx.xxxxxxxxcloud.com/xxx/d2b0xxxf6e65/default/
test143/latest/index.zip"
  },
  "func_name": "test",
  "func_urn": "urn:fss:xxxxxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest",
  "last_modified": "2018-02-26T11:55:41+08:00",
  "runtime": "Node.js6.10",
  "concurrency": 0,
  "depend_list": [],
  "strategy_config": {
      "concurrency": -1
  },
  "dependencies": [],
 "func_vpc": null
}
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1052",
  "error_msg": "Not found the function version"
 }
```

Status Code

See **Status Codes**.

5.7 Modifying the Metadata of a Function

Function

This API is used to modify the metadata of a function.

URI

PUT /v2/{project_id}/fgs/functions/{function_urn}/config

Table 5-27 describes the URI parameters.

Table 5-27 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**.

Request

Table 5-28 describes the request parameters.

Table 5-28 Request parameters

Parameter	Type	Mandatory	Description
runtime	String	No	Environment for executing the function. FunctionGraph supports Node.js 6.10, Node.js 8.10, Node.js 10.16, Node.js 12.13, Python 2.7, Python 3.6, Java 8, Go 1.8, C# (.NET Core 2.0), C# (.NET Core 2.1), C# (.NET Core 3.1), and PHP 7.3.

Parameter	Type	Mandatory	Description
code_type	String	Yes	Code type of a function. See **Table 4-1**.
code_url	String	No	Address of a function code package in Object Storage Service (OBS). This parameter is mandatory when **code_type** is set to **obs**.
description	String	No	Description of the function.
handler	String	No	Entry point of the function. See **Table 4-1**.
memory_size	Int	No	Memory (MB) consumed by the function. Options: 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2560, 3072, 3584, and 4096.
timeout	Int	No	Timeout allowed for the function.
user_data	String	No	Name/Value information defined for the function.
xrole	String	No	This parameter is mandatory if the function needs to access other cloud services.
app_xrole	*String	No	This parameter is mandatory if the function needs to access apps in other cloud services.
initializer_handler	String	No	Initializer of the function.
initializer_timeout	Int	No	Maximum duration the function can be initialized. Value range: 1s–300s.
func_vpc.subnet_id	String	No	Virtual Private Cloud (VPC) subnet ID.
func_vpc.vpc_id	String	No	VPC ID.
mount_config	mount_config	No	File system configuration. See **Table 4-6**.

Response

Table 5-29 describes the response parameters.

Table 5-29 Response parameters

Parameter	Type	Description
func_urn	String	Function URN.
func_name	String	Function name.
domain_id	String	Domain ID.
namespace	String	Tenant's project ID.
project_name	String	Tenant's project name.
package	String	Group to which the function belongs. This field is defined to group functions.
runtime	String	Environment for executing the function. FunctionGraph supports Node.js 6.10, Node.js 8.10, Node.js 10.16, Node.js 12.13, Python 2.7, Python 3.6, Java 8, Go 1.8, C# (.NET Core 2.0), C# (.NET Core 2.1), C# (.NET Core 3.1), and PHP 7.3.
timeout	Int	Maximum duration the function can be executed. Value range: 3s–900s.
handler	String	Handler of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.handler**, the file name is **myfunction.js**, and the entry point function is **handler**.
memory_size	Int	Memory (MB) consumed by the function. Options: 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2560, 3072, 3584, and 4096.
cpu	Int	Number of CPU millicores used by the function (1 core = 1000 millicores). The value of this field is proportional to that of **MemorySize**. By default, 100 CPU millicores are required for 128 MB memory. The value is calculated as follows: Memory/128 x 100 + 200 (basic CPU millicores).
code_type	String	Function code type. Options: ● **inline**: inline code ● **zip**: ZIP file ● **jar**: JAR file (mainly for Java functions) ● **obs**: function code stored in an Object Storage Service (OBS) bucket

Parameter	Type	Description
code_url	String	• When **code_type** is set to **obs**, this parameter indicates the address of a function code package in OBS. • When **code_type** is set to **inline**, **zip**, or **jar**, this parameter is left blank.
code_filename	String	Function file name. • When **code_type** is set to **zip** or **jar**, this parameter is required. • When **code_type** is set to **obs** or **inline**, this parameter is not required.
code_size	Int64	Code size in bytes.
user_data	String	Name/Value information defined for the function. For example, if a function needs to access a host, define Host={host_ip}. You can define a maximum of 20 such parameters, and their total length cannot exceed 4 KB.
digest	String	SHA512 hash value of function code, which is used to determine whether the function is changed.
version	String	Function version, which is automatically generated by the system. The version name is in the format of "vYYYYMMDD-HHMMSS" (v+year/month/day-hour/minute/second).
image_name	String	Internal identifier of a function version.
xrole	String	Agency used by the function. You need to create an agency on the Identity and Access Management (IAM) console. This field is mandatory when a function needs to access other services.
app_xrole	*String	Agency used by the function app. You need to create an agency on the IAM console. This field is mandatory when a function needs to access other services.
description	String	Description of the function.
version_description	String	Description of the function version.
last_modified	String	Time when the function was last updated.
func_code	String	Function code. See **Table 5-4**.

Parameter	Type	Description
depend_list	[]String	Dependency list.
strategy_config	String	Function policy configuration. See **Table 5-30**.
extend_config	String	Function extension configuration.
dependencies	[]*String	Dependency code package.
initializer_handler	String	Initializer of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.initializer**, the file name is **myfunction.js**, and the initialization function is **initializer**.
initializer_timeout	Int	Maximum duration the function can be initialized. Value range: 1s–300s.
func_vpc	func_vpc	Virtual Private Cloud (VPC) configuration. See **Table 5-6**.
mount_config	mount_config	File system configuration. See **Table 4-6**.

Table 5-30 strategy_config parameter

Parameter	Type	Mandatory	Description
concurrency	Int	Yes	**0**: The function is disabled. **-1**: The function is enabled.

Example

Example request

```
PUT
/v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/config HTTP/1.1
{
  "description": "",
  "handler": "test.handler",
  "memory_size": 128,
  "timeout": 3,
  "runtime": "Python",
  "user_data": "",
  "code_type": "inline",
  "func_code": {
    "file":
"aW1wb3J0IGpzb24KZGVmIGhhbmRsZXIgKGV2ZW50LCBjb250ZXh0KToKICAgIG91dHB1dCA9ICdI
ZWxsbyBtZXNzYWdlOiAnICsganNvbi5kdW1wcyhldmVudCkKICAgIHJldHVybiBvdXRwdXQ="
  },
 "xrole": "cffservice"
}
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200
{
    "func_urn": "urn:fss:xxxxxxxx:7aad83af3e8d42e99ac194e8419e2c9b:function:default:test",
    "func_name": "test",
    "user_domain": "cff01_hk",
    "namespace": "7aad83af3e8d42e99ac194e8419e2c9b",
    "project_name": "xxxxxxxx",
    "package": "default",
    "runtime": "Node.js6.10",
    "timeout": 3,
    "handler": "test.handler",
    "memory_size": 128,
    "cpu": 300,
    "code_type": "inline",
    "code_filename": "index.js",
    "code_size": 272,
    "digest": 
"decbce6939297b0b5ec6d1a23bf9c725870f5e69fc338a89a6a4029264688dc26338f56d08b6535d
e47f15ad538e22ca66613b9a46f807d50b687bb53fded1c6",
    "version": "latest",
    "image_name": "latest-5qe8e",
    "xrole": "cff",
    "last_modified": "2018-03-28T11:30:32+08:00",
   "strategy_config": {
      "concurrency": -1
 },
  "initializer_handler": "index.initializer",
  "initializer_timeout": 3
    }
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1051",
  "error_msg": "Not found the function"
 }
```

Status Code

See **Status Codes**.

5.8 Publishing a Function Version

Function

This API is used to publish a function version.

You can publish a version based on the code of the LATEST version, and FunctionGraph automatically generates a version name. A function can have a maximum of 10 versions.

URI

POST /v2/{project_id}/fgs/functions/{function_urn}/versions

Table 5-31 describes the URI parameters.

Table 5-31 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**.

Request

Table 5-32 describes the request parameters.

Table 5-32 Request parameters

Parameter	Type	Mandatory	Description
digest	String	No	Code digest of the function for which you want to publish a version. If this parameter is not specified, a version will be published using the code of the LATEST version.
version	String	No	Name of the version to be published. If this parameter is not specified, the current time in the format of "yyyymmdd-HHMMSS" will be used.
description	String	No	Description of the version to be published.

Response

Table 5-33 describes the response parameters.

Table 5-33 Response parameters

Parameter	Type	Description
func_urn	String	Function URN.
func_name	String	Function name.
domain_id	String	Domain ID.

Parameter	Type	Description
namespace	String	Tenant's project ID.
project_name	String	Tenant's project name.
package	String	Group to which the function belongs. This field is defined to group functions.
runtime	String	Environment for executing the function. FunctionGraph supports Node.js 6.10, Node.js 8.10, Node.js 10.16, Node.js 12.13, Python 2.7, Python 3.6, Java 8, Go 1.8, C# (.NET Core 2.0), C# (.NET Core 2.1), C# (.NET Core 3.1), and PHP 7.3.
timeout	Int	Maximum duration the function can be executed. Value range: 3s–900s.
handler	String	Handler of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.handler**, the file name is **myfunction.js**, and the entry point function is **handler**.
memory_size	Int	Memory (MB) consumed by the function. Options: 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2560, 3072, 3584, and 4096.
cpu	Int	Number of CPU millicores used by the function (1 core = 1000 millicores). The value of this field is proportional to that of **MemorySize**. By default, 100 CPU millicores are required for 128 MB memory. The value is calculated as follows: Memory/128 x 100 + 200 (basic CPU millicores).
code_type	String	Function code type. Options: ● **inline**: inline code ● **zip**: ZIP file ● **jar**: JAR file (mainly for Java functions) ● **obs**: function code stored in an Object Storage Service (OBS) bucket
code_url	String	● When **code_type** is set to **obs**, this parameter indicates the address of a function code package in OBS. ● When **code_type** is set to **inline**, **zip**, or **jar**, this parameter is left blank.

Parameter	Type	Description
code_filename	String	Function file name. ● When **code_type** is set to **zip** or **jar**, this parameter is required. ● When **code_type** is set to **obs** or **inline**, this parameter is not required.
code_size	Int64	Code size in bytes.
user_data	String	Name/Value information defined for the function. For example, if a function needs to access a host, define Host={host_ip}. You can define a maximum of 20 such parameters, and their total length cannot exceed 4 KB.
digest	String	SHA512 hash value of function code, which is used to determine whether the function is changed.
version	String	Function version, which is automatically generated by the system. The version name is in the format of "vYYYYMMDD-HHMMSS" (v+year/month/day-hour/minute/second).
image_name	String	Internal identifier of a function version.
xrole	String	Agency used by the function. You need to create an agency on the Identity and Access Management (IAM) console. This field is mandatory when a function needs to access other services.
app_xrole	*String	Agency used by the function app. You need to create an agency on the IAM console. This field is mandatory when a function needs to access other services.
description	String	Description of the function.
version_description	String	Description of the function version.
last_modified	String	Time when the function was last updated.
func_code	String	Function code. See **Table 5-4**.
depend_list	[]String	Dependency list.
strategy_config	String	Function policy configuration. See **Table 5-34**.
extend_config	String	Function extension configuration.

Parameter	Type	Description
dependencies	[]*String	Dependency code package. See **Table 4-5**.
initializer_handler	String	Initializer of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.initializer**, the file name is **myfunction.js**, and the initialization function is **initializer**.
initializer_timeout	Int	Maximum duration the function can be initialized. Value range: 1s–300s.
func_vpc	*String	Virtual Private Cloud (VPC) configuration. See **Table 5-6**.
mount_config	*String	Disk mount configuration. See **Table 5-7**.

Table 5-34 strategy_config parameter

Parameter	Type	Mandatory	Description
concurrency	Int	Yes	● **0**: The function is disabled. ● **-1**: The function is enabled.

Example

Example request

```
POST /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/versions HTTP/1.1
{
  "digest": "",
  "version": "1.0.0",
  "description": "test publish version"
}
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200
{
        "func_urn": "urn:fss:xxxxxxxxx:7aad83af3e8d42e99ac194e8419e2c9b:function:default:test",
        "func_name": "test",
        "user_domain": "cff01_hk",
        "namespace": "7aad83af3e8d42e99ac194e8419e2c9b",
        "project_name": "xxxxxxxxx",
        "package": "default",
        "runtime": "Node.js6.10",
        "timeout": 3,
        "handler": "test.handler",
        "memory_size": 128,
        "cpu": 300,
        "code_type": "inline",
        "code_filename": "index.js",
        "code_size": 272,
        "digest": 
"decbce6939297b0b5ec6d1a23bf9c725870f5e69fc338a89a6a4029264688dc26338f56d08b6535d
e47f15ad538e22ca66613b9a46f807d50b687bb53fded1c6",
        "version": "latest",
        "image_name": "latest-5qe8e",
        "xrole": "cff",
        "description": "111",
        "last_modified": "2018-03-28T11:30:32+08:00",
  "func_code": {
   },
   "strategy_config": {
        "concurrency": -1
   },
"initializer_handler": "index.initializer",
"initializer_timeout": 3      }
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1051",
  "error_msg": "Not found the function"
 }
```

Status Code

See **Status Codes**.

5.9 Querying the Versions of a Function

Function

This API is used to query the versions of a function.

URI

GET /v2/{project_id}/fgs/functions/{function_urn}/versions?marker={marker}&maxitems={maxitems} HTTP/1.1

Table 5-35 describes the URI parameters.

Table 5-35 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**.
marker	String	Yes	Final record queried last time.

Parameter	Type	Mandatory	Description
maxitems	String	Yes	Maximum number of functions that can be queried each time.

Request

None

Response

Table 5-36 describes the response parameters.

Table 5-36 Response parameters

Parameter	Type	Description
func_urn	String	Function URN.
func_name	String	Function name.
domain_id	String	Domain ID.
namespace	String	Tenant's project ID.
project_name	String	Tenant's project name.
package	String	Group to which the function belongs. This field is defined to group functions.
runtime	String	Environment for executing the function. FunctionGraph supports Node.js 6.10, Node.js 8.10, Node.js 10.16, Node.js 12.13, Python 2.7, Python 3.6, Java 8, Go 1.8, C# (.NET Core 2.0), C# (.NET Core 2.1), C# (.NET Core 3.1), and PHP 7.3.
timeout	Int	Maximum duration the function can be executed. Value range: 3s–900s.
handler	String	Handler of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.handler**, the file name is **myfunction.js**, and the entry point function is **handler**.
memory_size	Int	Memory (MB) consumed by the function. Options: 128, 256, 512, 768, 1024, 1280, 1536, 1792, 2048, 2560, 3072, 3584, and 4096.

Parameter	Type	Description
cpu	Int	Number of CPU millicores used by the function (1 core = 1000 millicores). The value of this field is proportional to that of **MemorySize**. By default, 100 CPU millicores are required for 128 MB memory. The value is calculated as follows: Memory/128 x 100 + 200 (basic CPU millicores).
code_type	String	Function code type. Options: ● **inline**: inline code ● **zip**: ZIP file ● **jar**: JAR file (mainly for Java functions) ● **obs**: function code stored in an Object Storage Service (OBS) bucket
code_url	String	● When **code_type** is set to **obs**, this parameter indicates the address of a function code package in OBS. ● When **code_type** is set to **inline**, **zip**, or **jar**, this parameter is left blank.
code_filename	String	Function file name. ● When **code_type** is set to **zip** or **jar**, this parameter is required. ● When **code_type** is set to **obs** or **inline**, this parameter is not required.
code_size	Int64	Code size in bytes.
user_data	String	Name/Value information defined for the function. For example, if a function needs to access a host, define Host={host_ip}. You can define a maximum of 20 such parameters, and their total length cannot exceed 4 KB.
digest	String	SHA512 hash value of function code, which is used to determine whether the function is changed.
version	String	Function version, which is automatically generated by the system. The version name is in the format of "vYYYYMMDD-HHMMSS" (v+year/month/day-hour/minute/second).
image_name	String	Internal identifier of a function version.

Parameter	Type	Description
xrole	String	Agency used by the function. You need to create an agency on the Identity and Access Management (IAM) console. This field is mandatory when a function needs to access other services.
app_xrole	*String	Agency used by the function app. You need to create an agency on the IAM console. This field is mandatory when a function needs to access other services.
description	String	Description of the function.
version_description	String	Description of the function version.
last_modified	String	Time when the function was last updated.
func_code	String	Function code. See **Table 5-4**.
depend_list	[]String	Dependency list.
strategy_config	String	Function policy configuration. See **Table 5-37**.
extend_config	String	Function extension configuration.
dependencies	[]*String	Dependency code package. See **Table 4-5**.
initializer_handler	String	Initializer of the function in the format of "xx.xx". It must contain a period (.). For example, for Node.js function **myfunction.initializer**, the file name is **myfunction.js**, and the initialization function is **initializer**.
initializer_timeout	Int	Maximum duration the function can be initialized. Value range: 1s–300s.
func_vpc	*String	Virtual Private Cloud (VPC) configuration. See **Table 5-6**.
mount_config	*String	Disk mount configuration. See **Table 5-7**.

Table 5-37 strategy_config parameters

Parameter	Type	Mandatory	Description
concurrency	Int	Yes	• **0**: The function is disabled. • **-1**: The function is enabled.

Example

Example request

```
GET /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/versions?
marker=1&maxtems=10 HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200
{
"versions": [
      {
       "func_urn": "urn:fss:xxxxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test",
       "func_name": "test",
       "user_domain": "cff01_hk",
       "namespace": "7aad83af3e8d42e99ac194e8419e2c9b",
       "project_name": "xxxxxxxxxx",
       "package": "default",
       "runtime": "Node.js6.10",
       "timeout": 3,
       "handler": "test.handler",
       "memory_size": 128,
       "cpu": 300,
       "code_type": "inline",
       "code_filename": "index.js",
       "code_size": 272,
       "digest":
"decbce6939297b0b5ec6d1a23bf9c725870f5e69fc338a89a6a4029264688dc26338f56d08b6535d
e47f15ad538e22ca66613b9a46f807d50b687bb53fded1c6",
       "version": "latest",
       "image_name": "latest-5qe8e",
       "xrole": "cff",
       "description": "111",
       "last_modified": "2018-03-28T11:30:32+08:00",
        "func_code": {
       },
       "strategy_config": {
              "concurrency": -1
       },
       "initializer_handler": "index.initializer",
       "initializer_timeout": 3
      }
  ],
     "next_marker": 1
}
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1051",
  "error_msg": "Not found the function"
 }
```

Status Code

See **Status Codes**.

5.10 Creating an Alias for a Function Version

Function

This API is used to create an alias for a function version.

URI

POST /v2/{project_id}/fgs/functions/{function_urn}/aliases

Table 5-38 describes the URI parameters.

Table 5-38 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**.

Request

Table 5-39 describes the request parameters.

Table 5-39 Request parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Name of the alias.
version	String	Yes	Version corresponding to the alias.
description	String	No	Description of the alias.
additional_version_weights	String	No	Key-value pair in JSON format to respectively indicate an additional version and a weight.

Response

Table 5-40 describes the response parameters.

Table 5-40 Response parameters

Parameter	Type	Description
name	String	Name of the alias.
version	String	Version corresponding to the alias.
description	String	Description of the alias.
last_modified	String	Time when the alias was last modified.
alias_urn	String	URN of the alias.
additional_version_weights	String	Key-value pair in JSON format to respectively indicate an additional version and a weight.

Example

Example request

```
POST /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/aliases HTTP/1.1
{
  "name":"dev",
  "version":"latest" ,
  "additional_version_weights":{"1.0":10 }
}
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200
{
    "name":"dev",
    "version":"latest",
    "description":"",
    "last_modified":"2017-06-26 03:21:10",
    "additional_version_weights ":{"1.0":10 } ,
    "alias_urn":"urn:fss:xxxxxxxxx: 7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:!
dev"
}
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1051",
  "error_msg": "Not found the function"
 }
```

Status Code

See **Status Codes**.

5.11 Modifying the Alias Information About a Function Version

Function

This API is used to modify the alias information about a function version.

URI

PUT /v2/{project_id}/fgs/functions/{function_urn}/aliases/{alias_name}

Table 5-41 describes the URI parameters.

Table 5-41 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**.
alias_name	String	Yes	Function alias.

Request

Table 5-42 describes the request parameters.

Table 5-42 Request parameters

Parameter	Type	Mandatory	Description
version	String	Yes	New version corresponding to the alias to be modified.
description	String	No	Description of the alias to be modified.
additional_version_weights	String	No	Key-value pair in JSON format to respectively indicate an additional version and a weight.

Response

Table 5-43 describes the response parameters.

Table 5-43 Response parameters

Parameter	Type	Description
name	String	Alias to be modified.
version	String	Version corresponding to the alias.
description	String	Description of the alias.
last_modified	String	Time when the alias was last modified.
alias_urn	String	URN of the alias.
additional_version_weights	String	Key-value pair in JSON format to respectively indicate an additional version and a weight.

Example

Example request

```
PUT /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/aliases/dev HTTP/1.1
{
  "version":"v20170725-152305",
  "description": "this is my version alias",
  "additional_version_weights ":{"1.0":10 }
}
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200
{
    "name": "dev",
    "version": "latest",
    "description": "",
"last_modified": "2017-06-26 03:21:10",
   "additional_version_weights ":{"1.0":10 },
    "alias_urn": "urn:fss:xxxxxxxx: 7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:!
dev"
}
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1051",
  "error_msg": "Not found the function"
 }
```

Status Code

See **Status Codes**.

5.12 Deleting an Alias of a Function Version

Function

This API is used to delete an alias of a function version.

NOTE

To delete the additional version of an alias, delete the alias first.

URI

DELETE /v2/{project_id}/fgs/functions/{function_urn}/aliases/{alias_name}

Table 5-44 describes the URI parameters.

Table 5-44 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**.
alias_name	String	Yes	Alias to be deleted.

Request

None

Response

None

Example

Example request

```
DELETE /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/aliases/dev HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 204
```

The format of the response for a failed request is as follows:

```
HTTP / 1.1 404 Not Found
{
    "error_code": "FSS.1053",
    "error_msg": "Not found the function alias"
}
```

Status Code

See **Status Codes**.

5.13 Querying the Alias Information About a Function Version

Function

This API is used to query the alias information about a function version.

URI

GET /v2/{project_id}/fgs/functions/{function_urn}/aliases/{alias_name}

Table 5-45 describes the URI parameters.

Table 5-45 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**.
alias_name	String	Yes	Name of the alias.

Request

None

Response

Table 5-46 describes the response parameters.

Table 5-46 Response parameters

Parameter	Type	Description
name	String	Alias to be obtained.
version	String	Version corresponding to the alias.
description	String	Description of the alias.
last_modified	String	Time when the alias was last modified.

Parameter	Type	Description
additional_version_weights	String	Key-value pair in JSON format to respectively indicate an additional version and a weight.
alias_urn	String	URN of the alias.

Example

Example request

```
GET /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest / aliases/dev HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200
{
    "name":"dev",
    "version":"latest",
    "description":"my dev version",
    "last_modified":"2017-06-26 03:21:10",
    "additional_version_weights":{"1.0":10 },
    "alias_urn":"urn:fss:xxxxxxxxxx: 7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:!
dev"
}
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1053",
  "error_msg": "Not found the function alias"
 }
```

Status Code

See **Status Codes**.

5.14 Querying the Version Alias List of a Function

Function

This API is used to query the version alias list of a function.

URI

GET /v2/{project_id}/fgs/functions/{function_urn}/aliases

Table 5-47 describes the URI parameters.

Table 5-47 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**.

Request

None

Response

Table 5-48 describes the response parameters.

Table 5-48 Response parameters

Parameter	Type	Description
name	String	Alias to be obtained.
version	String	Version corresponding to the alias.
description	String	Description of the alias.
additional_version_weights	String	Key-value pair in JSON format to respectively indicate an additional version and a weight.
last_modified	String	Time when the alias was last modified.
alias_urn	String	URN of the alias.

Example

Example request

```
GET /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/aliases HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200 OK
[
  {
   "alias_urn": "urn:fss:xxxxxxxxx:7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:!
testqq",
"additional_version_weights":{"1.0":10 },
   "last_modified": "2018-03-21T10:06:30+08:00",
   "name": "testqq",
   "version": "latest"
  }
 ]
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1051",
  "error_msg": "Not found the function"
 }
```

Status Code

See **Status Codes**.

5.15 Querying All Triggers of a Function

Function

This API is used to query all triggers of a function.

URI

GET /v2/{project_id}/fgs/triggers/{function_urn}

Request

Table 5-49 describes the request parameters.

Table 5-49 Request parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**.

Response

Table 5-50 describes the response parameters.

Table 5-50 Response parameters

Parameter	Type	Description
trigger_id	String	Trigger ID.
trigger_status	String	Trigger status.

Parameter	Type	Description
trigger_type_code	String	Trigger type code.
event_data	String	Trigger data defined in JSON format. **NOTE** See **Trigger Instance Data**.
last_updated_time	String	Time when the trigger was last updated.
created_time	String	Time when the trigger was created.

Example

Example request

```
GET /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/triggers/urn:fss:xxxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest
HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200 OK
[
 {
  "trigger_id": "0586f1e2-8db2-4d2a-97bd-989f67d9fd8b",
  "trigger_type_code": "TIMER",
  "trigger_status": "ACTIVE",
  "event_data": {
     "name": "Timer-tg0q",
     "schedule": "3m",
     "schedule_type": "Rate"
   }
  "last_updated_time": "2020-04-23T15:02:17+08:00",
  "created_time": "2020-04-23T15:02:17+08:00"
 }
]
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
 "error_code": "FSS.1051",
 "error_msg": "Error getting associated function"
}
```

Status Code

See **Status Codes**.

5.16 Querying the Information About a Trigger

Function

This API is used to query the information about a trigger.

URI

GET /v2/{project_id}/fgs/triggers/{function_urn}/{trigger_type_code}/{trigger_id}

Table 5-51 describes the URI parameters.

Table 5-51 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
function_urn	String	Yes	Function URN. See **Function Model**.
trigger_type_code	String	Yes	Trigger type code. Options: SMN, APIG, OBS, TIMER, CTS, and kafka.
trigger_id	String	Yes	Trigger ID.

Request

None

Response

Table 5-52 describes the response parameters.

Table 5-52 Response parameters

Parameter	Type	Description
trigger_id	String	Trigger ID.
trigger_type_code	String	Trigger type code.
trigger_status	String	Trigger status.
event_data	String	Trigger data defined in JSON format. **NOTE** See **Trigger Instance Data**.
last_updated_time	String	Time when the trigger was last updated.

Parameter	Type	Description
created_time	String	Time when the trigger was created.

Example

Example request

```
GET https://{functiongraph_endpoint}/v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/triggers/
urn:fss:xxxxxxxxxx:7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/TIMER/
9a14fae1-78cf-4185-ac7a-429eb6dc41fb HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
{
  "trigger_id" : "9a14fae1-78cf-4185-ac7a-429eb6dc41fb",
  "trigger_type_code" : "TIMER",
  "trigger_status" : "ACTIVE",
  "event_data" : {
    "name" : "Timer-cpg3",
    "schedule" : "3m",
    "schedule_type" : "Rate"
  },
  "last_updated_time" : "2019-10-29T17:15:53+08:00",
  "created_time" : "2019-10-29T17:15:53+08:00"
}
```

The format of the response for a failed request is as follows:

```
{
  "error_code" : "FS.0019",
  "error_msg" : "Not found the function"
}
```

Status Code

See **Status Codes**.

5.17 Deleting All Triggers of a Function

Function

This API is used to delete all triggers of a function.

- If a non-LATEST version of a function is specified, all triggers of this function version will be deleted.
- If an alias is specified, all triggers corresponding to the alias will be deleted.
- If neither function versions nor aliases are specified or the LATEST version is specified, all triggers of the function (including all versions and aliases) will be deleted.

URI

DELETE /v2/{project_id}/fgs/triggers/{function_urn}

Table 5-53 describes the URI parameters.

Table 5-53 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
function_urn	String	Yes	Function URN. See **Function Model**.

Request

None

Response

None

Example

Example request

```
DELETE /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/triggers/urn:fss:xxxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 204
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404
{"error_code": "FS.0019", "error_msg": "Not found the function"}
```

Status Code

See **Status Codes**.

5.18 Creating a Trigger

Function

This API is used to create a trigger.

URI

POST /v2/{project_id}/fgs/triggers/{function_urn}

Table 5-54 describes the URI parameters.

Table 5-54 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Project ID.
function_urn	String	Yes	Function URN. See **Function Model**.

Request

Table 5-55 describes the request parameters.

Table 5-55 Request parameters

Parameter	Type	Mandatory	Description
trigger_type_code	String	Yes	Trigger type.
event_type_code	String	Yes	Event type.
trigger_status	String	Yes	Trigger status. Options: **ACTIVE** and **DISABLED**.
event_data	String	Yes	Event information.

Response

Table 5-56 describes the response parameters.

Table 5-56 Response parameters

Parameter	Type	Description
trigger_id	String	Trigger ID.
trigger_type_code	String	Trigger type code.
event_type_code	String	Event type code.
trigger_status	String	Trigger status. Options: **ACTIVE** and **DISABLED**.
event_data	String	Trigger data defined in JSON format. **NOTE** See **Trigger Instance Data**.
last_updated_time	String	Time when the trigger was last updated.

Parameter	Type	Description
created_time	String	Time when the trigger was created.

Example

Example request

```
POST /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/triggers/urn:fss:xxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest
{
    "trigger_type_code": "TIMER",
    "event_type_code": "MessageCreated",
    "trigger_status": "ACTIVE",
    "event_data": {
        "name": "Timer-tps7",
        "schedule_type": "Rate",
        "schedule": "3m",
        "user_event": ""
    }
}
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 201 Created
{
  "trigger_id": "be1cb36a-5efd-40ed-8376-7525bfcbe848",
  "trigger_type_code": "TIMER",
  "trigger_status": "ACTIVE",
  "event_data": {
   "name": "Timer-tps7",
   "schedule": "3m",
   "schedule_type": "Rate"
  },
  "last_updated_time": "2020-04-23T15:07:51+08:00",
  "created_time": "2020-04-23T15:07:51+08:00"
 }
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{
  "error_code": "FSS.1051",
  "error_msg": "Error getting associated function"
 }
```

Status Code

See **Status Codes**.

5.19 Deleting a Trigger

Function

This API is used to delete a trigger.

URI

DELETE /v2/{project_id}/fgs/triggers/{function_urn}/{trigger_type_code}/{trigger_id}

Table 5-57 describes the URI parameters.

Table 5-57 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
function_urn	String	Yes	Function URN. See **Function Model**.
trigger_type_code	String	Yes	Trigger type code. Options: SMN, APIG, OBS, TIMER, CTS, and kafka.
trigger_id	String	Yes	Trigger ID.

Request

None

Response

None

Example

Example request

```
DELETE /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/triggers/urn:fss:xxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/CTS/
f4748d95-7ce7-4f9e-9434-67316a828d94 HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 204 No Content
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 400 Bad Request
{
 "error_code": "FSS.0400",
 "error_msg": "Invalid trigger data"
}
```

Status Code

See **Status Codes**.

6 Function Data Zone APIs

6.1 Implementing Synchronous Function Invocation

Function

This API is used to implement synchronous function invocation.

NOTE

For synchronous function invocation, clients must wait for explicit responses to their requests from a function. Responses are returned only after function invocation is complete. For details on how to invoke a function, see **Test Management**.

URI

POST /v2/{project_id}/fgs/functions/{function_urn}/invocations

Table 6-1 describes the URI parameters.

Table 6-1 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
function_urn	String	Yes	Function URN. See **Function Model**.

Header: **X-Cff-Log-Type**. Options: **tail** (4 KB logs will be returned) and **null** (no logs will be returned).

Request

Event in JSON format

Parameter	Type	Mandatory	Description
{Customized_key}	Map<String,String>	No	Function execution request body in JSON format.

Response

Table 6-2 describes the response parameters.

Table 6-2 Response parameters

Parameter	Type	Description
X-Cff-Function-Log	String	Function execution log encoded using Base64.
X-Cff-Invoke-Summary	JSON	Execution summary.
Body	JSON	Function execution result.

Example of X-Cff-Invoke-Summary:

```
{"duration":1.913,"billingDuration":100,"memorySize":128,"memoryUsed":41.51171875}
```

Table 6-3 X-Cff-Invoke-Summary parameters

Parameter	Description
duration	Function execution duration (ms).
billingDuration	Billing duration (ms).
memorySize	Configured memory (MB).
memoryUsed	Used memory (MB).

Example

Example request

```
POST /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/invocations HTTP/1.1
{
"message":"Hello World"
}
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 200 OK
{"message": "hello world from FunctionStage"}
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{"error_code":"FSS.0404","error_msg":"Not found the specified resource"}
```

Status Code

See **Status Codes**.

6.2 Implementing Asynchronous Function Invocation

Function

This API is used to implement asynchronous function invocation.

URI

POST /v2/{project_id}/fgs/functions/{function_urn}/invocations-async

Table 6-4 describes the URI parameters.

Table 6-4 URI parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Tenant's project ID.
function_urn	String	Yes	Function URN. See **Function Model**.

Request

Event in JSON format

Parameter	Type	Mandatory	Description
{Customized_key}	Map<String,String>	No	Function execution request body in JSON format.

Response

Table 6-5 describes the response parameter.

Table 6-5 Response parameter

Parameter	Type	Description
Body	JSON	Request ID.

Example

Example request

```
POST /v2/7aad83af3e8d42e99ac194e8419e2c9b/fgs/functions/urn:fss:xxxxxxxxxxx:
7aad83af3e8d42e99ac194e8419e2c9b:function:default:test:latest/invocations-async HTTP/1.1
```

Example response

The format of the response for a successful request is as follows:

```
HTTP/1.1 202 Accepted
{"request_id": "e834cb5b-1b2b-4c6b-b41c-8bd10fd41826"}
```

The format of the response for a failed request is as follows:

```
HTTP/1.1 404 Not Found
{"error_code":"FSS.0404","error_msg":"function 'test' not exist"}
```

The format of the response for disabling a function is as follows:

```
HTTP / 1.1 429  Disabled
{
    "error_code": "FSS.0429",
    "error_msg": "Function Disabled"
}
```

Status Code

See **Status Codes**.

7 Permissions Policies and Supported Actions

Table 7-1 FunctionGraph actions

Permission	API	Action
Querying a function list	GET /v2/{project_id}/fgs/functions	FunctionGraph:function:list
Querying the metadata of a function	GET /v2/{project_id}/fgs/functions/{function_urn}/config	FunctionGraph:function:getConfig
Querying the code of a function	GET /v2/{project_id}/fgs/functions/{function_urn}/code	FunctionGraph:function:getCode
Creating a function	POST /v2/{project_id}/fgs/functions	FunctionGraph:function:create
Deleting a function or function version	DELETE /v2/{project_id}/fgs/functions/{function_urn}	FunctionGraph:function:delete
Modifying the code of a function	PUT /v2/{project_id}/fgs/functions/{function_urn}/code	FunctionGraph:function:updateCode
Modifying the metadata of a function	PUT /v2/{project_id}/fgs/functions/{function_urn}/config	FunctionGraph:function:updateConfig
Publishing a function version	POST /v2/{project_id}/fgs/functions/{function_urn}/versions	FunctionGraph:function:createVersion
Querying the versions of a function	GET /v2/{project_id}/fgs/functions/{function_urn}/versions	FunctionGraph:function:listVersion

Permission	API	Action
Creating an alias for a function version	POST /v2/{project_id}/fgs/functions/{function_urn}/aliases	FunctionGraph:function:createAlias
Modifying the alias information about a function version	PUT /v2/{project_id}/fgs/functions/{function_urn}/aliases/{alias_name}	FunctionGraph:function:updateAlias
Deleting an alias of a function version	DELETE /v2/{project_id}/fgs/functions/{function_urn}/aliases/{alias_name}	FunctionGraph:function:deleteAlias
Querying the alias information about a function version	GET /v2/{project_id}/fgs/functions/{function_urn}/aliases/{alias_name}	FunctionGraph:function:getAlias
Querying all version aliases of a function	GET /v2/{project_id}/fgs/functions/{function_urn}/aliases	FunctionGraph:function:listAlias
Querying all triggers of a function	GET /v2/{project_id}/fgs/triggers/{function_urn}	FunctionGraph:trigger:listSpecifiedFunctionTriggers
Querying the information about a trigger	GET /v2/{project_id}/fgs/triggers/{function_urn}/{trigger_type_code}/{trigger_id}	FunctionGraph:trigger:get
Deleting all triggers of a function	DELETE /v2/{project_id}/fgs/triggers/{function_urn}	FunctionGraph:trigger:deleteSpecifiedFunctionTriggers
Creating a trigger	POST /v2/{project_id}/fgs/triggers/{function_urn}	FunctionGraph:trigger:create
Deleting a trigger	DELETE /v2/{project_id}/fgs/triggers/{function_urn}/{trigger_type_code}/{trigger_id}	FunctionGraph:trigger:delete
Implementing synchronous function invocation	POST /v2/{project_id}/fgs/functions/{function_urn}/invocations	FunctionGraph:function:invoke
Implementing asynchronous function invocation	POST /v2/{project_id}/fgs/functions/{function_urn}/invocations-async	FunctionGraph:function:invokeAsync

8 Appendix

8.1 Status Codes

Table 8-1 describes the status codes.

Table 8-1 Status codes

Status Code	Message	Description
200	-	The request has succeeded.
204	-	The request has succeeded.
400	Bad Request	The server failed to process the request.
401	Unauthorized	The request requires user authentication.
403	Forbidden	The server understood the request, but is refusing to fulfill it.
404	Not Found	The server has not found anything matching the request URI.
405	Method Not Allowed	The method specified in the request line is not allowed for the resource identified by the request URI.

Status Code	Message	Description
406	Not Acceptable	The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request.
407	Proxy Authentication Required	The client must first authenticate itself with the proxy.
408	Request Timeout	The client did not produce a request within the time that the server was prepared to wait.
409	Conflict	The request could not be completed due to a conflict with the current state of the resource.
500	Internal Server Error	The server encountered an unexpected condition which prevented it from fulfilling the request.
501	Not Implemented	The server does not support the functionality required to fulfill the request.
502	Bad Gateway	The server, while acting as a gateway or proxy, received an invalid response from the upstream server it accessed in attempting to fulfill the request.
503	Service Unavailable	The server is currently unable to handle the request due to a temporary overloading or maintenance of the server.
504	Gateway Timeout	Gateway timed out.

8.2 Error Codes

Table 8-2 Function error codes

Status Code	Error Code	Error Message	Description
400	FSS. 0400	Invalid parameter.	Invalid request parameters.
401	FSS. 0401	Unauthorized access.	Access denied.
403	FSS. 0403	Forbidden	Unauthorized operation.
404	FSS. 0404	The specified resource was not found.	The resource cannot be found.
406	FSS. 0406	Not acceptable.	Incorrect request format. For example, the request body may not be in the required JSON format.
408	FSS. 0408	Request timeout.	Request timed out.
409	FSS. 0409	The specified resource already exists.	The resource already exists.
410	FSS. 0410	The specified resource does not exist.	The specified resource does not exist.
413	FSS. 0413	The request body is too large.	The request body exceeds the maximum allowed limit.
424	FSS. 0424	Invalid dependency.	Invalid dependency.
426	FSS. 0426	An upgrade is required.	Unsupported operation. Perform an upgrade.
428	FSS. 0428	The preconditions have not been met.	The prerequisite has not been met. For example, you must delete related resources before deleting a resource.

Status Code	Error Code	Error Message	Description
429	FSS. 0429	Too many requests.	Too many concurrent requests. Please try again later.
500	FSS. 0500	Internal server error.	The service is temporarily abnormal due to an internal invocation error. Please try again later.
502	FSS. 0502	Bad gateway.	Gateway error.
503	FSS. 0503	Service unavailable.	Service unavailable.
504	FSS. 0504	Gateway timeout.	Gateway timed out.
400	FSS. 1001	Invalid query parameter.	Invalid query parameters.
400	FSS. 1002	Invalid function name.	Invalid function name.
400	FSS. 1003	Invalid function handler.	Invalid handler.
400	FSS. 1004	Invalid Package (function app).	Invalid function package.
400	FSS. 1005	Invalid runtime.	Invalid runtime.
400	FSS. 1006	Invalid function code entry mode.	Invalid code type.
400	FSS. 1007	Invalid function memory.	Invalid memory.
400	FSS. 1008	Invalid function timeout.	Invalid timeout.
400	FSS. 1009	Invalid function UserData.	Invalid environment variables.
400	FSS. 1010	Invalid URL.	Invalid URL.
400	FSS. 1011	Invalid function code.	Invalid function code.
400	FSS. 1012	The function code must be configured.	Function code is required.

Status Code	Error Code	Error Message	Description
400	FSS. 1013	Invalid file type.	Invalid file type.
400	FSS. 1014	Invalid function alias.	Invalid alias.
400	FSS. 1015	Invalid function version.	Invalid version.
400	FSS. 1016	The function cannot be published because no changes have been made since last publication.	The function code cannot be published because it has not been changed since last publication.
400	FSS. 1017	The number of items in the UserData field exceeds the maximum allowed limit (20).	The number of environment variables exceeds 20.
400	FSS. 1018	The total size of the UserData field exceeds the maximum allowed limit (2 KB).	The total size of environment variables exceeds 2 KB.
400	FSS. 1019	The description exceeds the maximum allowed limit.	The maximum length is 512 characters.
400	FSS. 1022	Only one YAML file is allowed.	Only one YAML file is allowed.
400	FSS. 1023	The imported file is too large.	The imported file exceeds the maximum allowed limit.
400	FSS. 1024	Invalid dependency.	Invalid dependency.
400	FSS. 1025	Invalid YAML file.	Invalid YAML file.
400	FSS. 1026	Invalid Concurrency.	Invalid concurrency policy.
400	FSS. 1027	Invalid packageName (app name).	Invalid package or app name.
400	FSS. 1028	The app cannot be deleted because it contains functions.	The app cannot be deleted because it contains functions.
400	FSS. 1029	The default app cannot be deleted.	The default app cannot be deleted.
400	FSS. 1031	The dependency already exists.	The dependency already exists.

Status Code	Error Code	Error Message	Description
400	FSS. 1032	Invalid dependency type.	Invalid dependency type. Currently, only local ZIP packages or packages from OBS can be uploaded.
412	FSS. 1033	The dependency is currently in use.	Failed to delete the dependency because it is in use.
400	FSS. 1034	Invalid image URL.	Invalid image URL.
400	FSS. 1035	The image does not exist.	The image does not exist.
400	FSS. 1036	The VPC does not exist.	The VPC does not exist.
400	FSS. 1037	No subnet matches the specified ID.	No matched subnet found.
400	FSS. 1038	The file system configuration already exists in the function.	The file system configuration already exists in the function.
400	FSS. 1039	The mounting path is invalid.	Invalid mounting path.
403	FSS. 1040	The selected Xrole does not have permissions to mount the resources.	The selected agency does not have permissions to mount the resources.
403	FSS. 1041	The number of functions exceeds the maximum allowed limit.	The number of functions exceeds 400.
403	FSS. 1042	The total code size of functions exceeds the maximum allowed limit.	The total size of functions exceeds 20 GB.
403	FSS. 1043	The number of aliases exceeds the maximum allowed limit.	The number of aliases exceeds the maximum allowed limit.
403	FSS. 1044	The number of apps exceeds the maximum allowed limit (400).	The number of apps exceeds 400.
403	FSS. 1045	The number of dependencies exceeds the maximum allowed limit.	The number of dependencies exceeds the maximum allowed limit.

Status Code	Error Code	Error Message	Description
403	FSS. 1046	The dependency is inaccessible.	The dependency is unavailable.
403	FSS. 1047	The number of bound VPCs exceeds the maximum limit allowed for a tenant.	The number of bound VPCs exceeds the maximum limit allowed for a tenant.
403	FSS. 1048	The number of bound VPCs exceeds the maximum limit allowed for a project.	The number of bound VPCs exceeds the maximum limit allowed for a project.
403	FSS. 1049	The number of file systems mounted to the function exceeds the maximum allowed limit (5).	The number of file systems mounted to the function exceeds 5.
404	FSS. 1050	The mounted resource cannot be found.	The mounted resource cannot be found.
404	FSS. 1051	The function does not exist.	The function cannot be found.
404	FSS. 1052	The version does not exist.	The version cannot be found.
404	FSS. 1053	The alias does not exist.	The alias cannot be found.
404	FSS. 1054	The function app does not exist in OBS.	The specified code package cannot be found in OBS.
404	FSS. 1055	The app does not exist.	The specified function app cannot be found in OBS.
404	FSS. 1056	The dependency does not exist.	The dependency does not exist.
404	FSS. 1057	The function name does not exist in the YAML file.	The function name does not exist in the YAML file.
400	FSS. 1058	The app name and function name cannot be the same in the YAML file.	The combination of the app name and function name cannot be the same in the YAML file.
404	FSS. 1059	The function template does not exist.	The function template does not exist.

Status Code	Error Code	Error Message	Description
404	FSS. 1060	The event template cannot be found.	The event template does not exist.
409	FSS. 1061	The function already exists.	The function already exists.
409	FSS. 1062	The version already exists.	The version already exists.
409	FSS. 1063	The alias already exists.	The alias already exists.
409	FSS. 1064	The app already exists.	The app already exists.
409	FSS. 1065	The dependency already exists.	The dependency already exists.
409	FSS. 1066	The version is already in use by another alias.	The version is already in use by another alias.
409	FSS. 1067	The function template already exists.	The function template already exists.
403	FSS. 1068	The number of events configured for the function exceeds the maximum allowed limit.	The number of events configured for the function exceeds the maximum allowed limit.
403	FSS. 1069	The size of EventData exceeds 4 KB.	The event size exceeds 4 KB.
404	FSS. 1070	The event cannot be found.	The event cannot be found.
413	FSS. 1071	The size of the code package to be uploaded exceeds the maximum allowed limit (50 MB).	The size of the code package to be uploaded exceeds 50 MB.
413	FSS. 1072	The size of the inline code exceeds the maximum allowed limit (10 KB).	The code exceeds 10 KB.
403	FSS. 1073	The function event already exists.	The function event already exists.
400	FSS. 1074	The event field is invalid.	Invalid event field.

Status Code	Error Code	Error Message	Description
400	FSS. 1075	The value of UserId or GroupId must be a non-zero integer from –1 to 65534.	The user ID and user group ID must be a non-0 integer ranging from –1 to 65534.
412	FSS. 1090	The subnet is not in the \"ACTIVE\" state.	The subnet is not in the "ACTIVE" state.
400	FSS. 1091	The additional version is invalid.	Invalid additional version.
400	FSS. 1092	The weight of the additional version is invalid.	The weight of the additional version is invalid.
400	FSS. 1093	The major version and the additional version cannot be the same.	The major version and the additional version cannot be the same.
403	FSS. 1094	The version cannot be deleted because it has been used as the additional version of an alias.	The version cannot be deleted because it has been used as the additional version of an alias.
412	FSS. 1095	The mounted resource is not ready.	The mounted resource is not ready.
403	FSS. 1096	The file sharing protocol of the mounted resource is not NFS.	The file sharing protocol of the mounted resource is not NFS.
400	FSS. 1101	Invalid trigger type.	Invalid trigger type.
400	FSS. 1102	Invalid SMN trigger parameter.	Invalid SMN trigger parameters.
400	FSS. 1106	Invalid OBS trigger parameters.	Invalid OBS trigger parameters.
400	FSS. 1107	Invalid APIG trigger parameters.	Invalid APIG trigger parameters.
403	FSS. 1108	The bucket configuration of the current trigger conflicts with that of an existing OBS trigger.	The bucket configuration of the current trigger conflicts with that of an existing OBS trigger.
400	FSS. 1109	Invalid timer trigger parameters.	Invalid timer trigger parameters.

Status Code	Error Code	Error Message	Description
404	FSS. 1111	The Kafka resource cannot be found.	The Kafka resource cannot be found.
400	FSS. 1112	The Kafka trigger parameters are invalid.	Invalid Kafka trigger parameters.
400	FSS. 1113	The username and password must be specified because Kafka SASL_SSL is enabled.	The username and password must be specified because Kafka SASL_SSL is enabled.
400	FSS. 1114	The subnet of the function must be the same as that of the Kafka instance.	The subnet of the function is different from that of the Kafka instance.
503	FSS. 1115	The network is unreachable.	The network is unreachable.
400	FSS. 1116	Kafka instance configuration error. Please check the username and password.	Kafka instance configuration error. Check the username and password.
400	FSS. 1117	Failed to query messages from the Kafka instance.	Failed to query messages from the Kafka instance.
401	FSS. 1118	Access denied.	Access denied. The user is not in the whitelist.
403	FSS. 1121	Forbidden	Access denied. Check whether the corresponding agency has been configured.
403	FSS. 1122	Forbidden	Access denied. Check whether the corresponding agency has been configured.
403	FSS. 1123	The number of pull triggers exceeds the maximum allowed limit.	The number of pull-mode triggers has reached 10.
403	FSS. 1124	The number of APIs exceeds the maximum allowed limit.	The number of APIs exceeds the maximum allowed limit.
403	FSS. 1125	Forbidden	Access denied.

Status Code	Error Code	Error Message	Description
403	FSS. 1126	You do not have permissions to call the API.	You do not have permissions to call the API.
403	FSS. 1127	The EPS user does not have permissions to call the API.	The EPS user does not have permissions to call the API.
404	FSS. 1131	The trigger does not exist.	The trigger cannot be found.
404	FSS. 1132	The SMN trigger does not exist. View the SMN console.	The SMN trigger cannot be found.
404	FSS. 1136	The OBS trigger does not exist.	The OBS trigger cannot be found.
404	FSS. 1137	Invalid trigger type.	The trigger type cannot be found.
404	FSS. 1138	The APIG trigger does not exist.	The APIG trigger cannot be found.
404	FSS. 1140	The timer trigger does not exist.	The timer trigger cannot be found.
409	FSS. 1141	The SMN trigger already exists.	The SMN trigger already exists.
409	FSS. 1145	The OBS trigger already exists.	The OBS trigger already exists.
409	FSS. 1146	The APIG trigger already exists.	The APIG trigger already exists.
409	FSS. 1147	The request path already exists.	The request path already exists.
409	FSS. 1148	The timer trigger already exists.	The timer trigger already exists.
409	FSS. 1150	The Kafka trigger already exists.	The Kafka trigger already exists.
406	FSS. 1151	The OBS bucket is in a different region.	The region where the OBS bucket is located does not match the current region.
426	FSS. 1152	The selected bucket cannot be used to create a trigger.	The selected OBS bucket cannot be used to create a trigger.

Status Code	Error Code	Error Message	Description
412	FSS. 1153	The triggering conditions have not been met.	The triggering conditions have not been met.
403	FSS. 1154	Aliases of a function bound with triggers cannot be deleted.	The aliases cannot be deleted because they are bound with triggers.
500	FSS. 1162	The operation cannot take effect immediately due to service exception.	The operation cannot take effect immediately because the service is abnormal.
503	FSS. 1169	The network is unreachable.	The network is unreachable.
404	FSS. 1171	The SMN topic does not exist. Create one on the SMN console.	The SMN topic does not exist.
400	FSS. 1172	The database or collection does not exist.	The DB instance cannot be found.
404	FSS. 1174	The Kafka trigger does not exist.	The Kafka trigger cannot be found.
413	FSS. 1201	The request body is too large.	The request body exceeds the maximum allowed limit.
500	FSS. 1202	The response body or callback body is invalid because they do not contain any status code.	Invalid response body.
500	FSS. 1302	Failed to save the data.	Failed to save the trigger data.
400	FSS. 1303	Access denied due to insufficient permissions.	Failed to verify permission. Access denied.
400	FSS. 1306	The number of triggers exceeds the maximum allowed limit.	Trigger threshold reached.
400	FSS. 1307	The trigger name already exists.	The trigger name already exists.
400	FSS. 1308	The operation resource does not exist.	The resource does not exist.
400	FSS. 1309	Invalid function URN.	Invalid function URN.

Status Code	Error Code	Error Message	Description
400	FSS. 1310	Unauthorized user.	Failed to obtain the user token.
400	FSS. 1312	The notification name must be specified.	No key notification name specified.
400	FSS. 1313	The number of operation resources has reached the maximum allowed limit.	The number of operation resources exceeds 100.
400	FSS. 1314	The operation resource must be specified.	No operation resource specified.
400	FSS. 1316	The resource operation has already been selected.	Duplicate operation resource.
400	FSS. 1317	The trigger name is too long.	The trigger name is too long.
400	FSS. 1318	Invalid trigger operation.	Invalid trigger operation.
502	FSS. 1319	Invalid trigger name.	Invalid trigger name.
503	FSS. 1401	Failed to obtain the image information.	Failed to obtain the image information.
503	FSS. 1402	Failed to pull the image to create a container.	Failed to pull the image to create a container.
503	FSS. 1403	Failed to pull the image to delete a container.	Failed to pull the image to delete a container.
400	FSS. 1404	Invalid function initializer.	Invalid function initializer.
400	FSS. 1405	Invalid initialization timeout.	Invalid initialization timeout.

8.3 Obtaining a Project ID

Obtaining a Project ID on the Console

When calling APIs, you need to enter a project ID in some URLs. To obtain a project ID, perform the following steps:

1. Log in to the management console.

2. Click the username and choose **My Credentials** from the drop-down list.

   On the **My Credentials** page, view the project ID.

Obtaining a Project ID by Calling an API

A project ID can also be obtained by calling a specific API. For details, see **Querying Project Information**.

The API used to obtain a project ID is **GET https://*{Endpoint}*/v3/projects**, where *{Endpoint}* indicates the IAM endpoint. You can obtain the IAM endpoint from **Regions and Endpoints**. For details on API calling authentication, see **Authentication**.

The following is an example response. The value of **id** in the **projects** section is the project ID.

```
{
    "projects": [
        {
            "domain_id": "65382450e8f64ac0870cd180d14e684b",
            "is_domain": false,
            "parent_id": "65382450e8f64ac0870cd180d14e684b",
            "name": "xxx",
            "description": "",
            "links": {
                "next": null,
                "previous": null,
                "self": "https://www.example.com/v3/projects/a4a5d4098fb4474fa22cd05f897d6b99"
            },
            "id": "a4a5d4098fb4474fa22cd05f897d6b99",
            "enabled": true
        }
    ],
    "links": {
        "next": null,
        "previous": null,
        "self": "https://www.example.com/v3/projects"
    }
}
```

8.4 FunctionGraph Metrics

Introduction

This section describes the function metrics reported to Cloud Eye.

Their namespace and dimension are also included. You can view monitoring graphs and alarm messages on the Cloud Eye console.

Namespace

SYS.FunctionGraph

Function Metrics

Table 8-3 Function metrics

Metric	Display Name	Description	Unit	Upper Limit	Lower Limit	Recommended Threshold	Value Type	Meaning	Dimension
count	Invocations	Number of times a function is invoked	Count	-	0	-	Int	Number of times a function is invoked	package-functionname
failcount	Errors	Number of errors that occur when a function is invoked	Count	-	0	-	Int	Number of errors that occur when a function is invoked	package-functionname
rejectcount	Throttles	Number of times a function is throttled when invoked	Count	-	0	-	Int	Number of times a function is throttled when invoked	package-functionname
duration	Average Duration	Average time a function is invoked	ms	-	0	-	Int	Average time a function is invoked	package-functionname
maxDuration	Maximum Duration	Maximum time a function is invoked	ms	-	0	-	Int	Maximum time a function is invoked	package-functionname

Metric	Display Name	Description	Unit	Upper Limit	Lower Limit	Recommended Threshold	Value Type	Meaning	Dimension
minDuration	Minimum Duration	Minimum time a function is invoked	ms	-	0	-	Int	Minimum time a function is invoked	package-functionname

Dimension

Table 8-4 Dimension

Key	Value
package-functionname	App_name-Function_name

9 Change History

Table 9-1 Change history

Date	Description
2021-01-19	This issue incorporates the following changes: ● Updated **Function Model**. ● Updated **Querying a Function List**.
2020-10-30	This issue is the first release.