Access Guide

Issue 01
Date 2025-01-16

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Security Declaration

Vulnerability

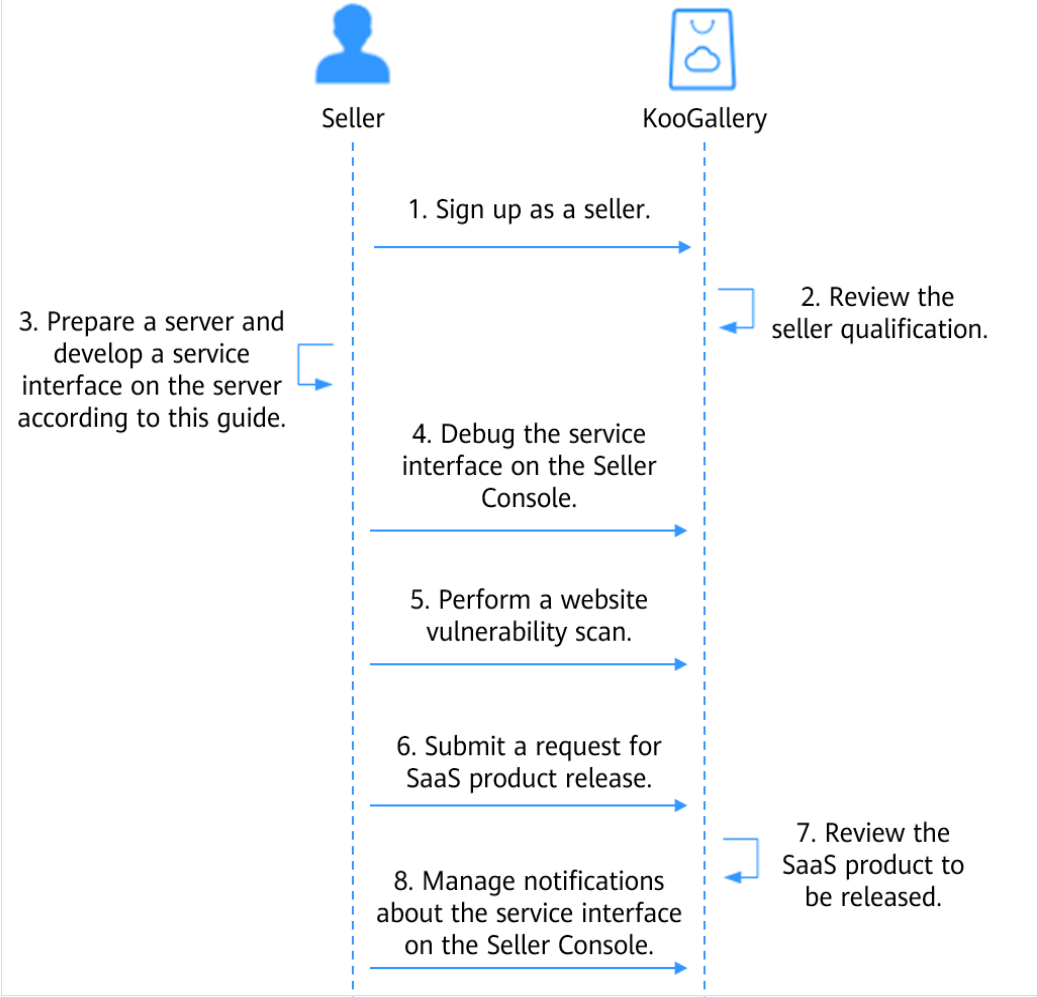
Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process.* For details about this process, visit the following web page:
https://www.huawei.com/en/psirt/vul-response-process
For vulnerability information, enterprise customers can visit the following web page:
https://securitybulletin.huawei.com/enterprise/en/security-advisory

Contents

1 SaaS Access Guide V2.0 (New Products)

1.1 Access Process

The following figure shows the process of SaaS products accessing KooGallery.



The process is as follows:

1. **Register with KooGallery** and become a seller.
2. The KooGallery operations team reviews your company qualification.
3. Prepare a server and develop interfaces on the server based on this guide.
4. Debug the interfaces on the **Seller Console**.
5. Perform vulnerability scans on the **Seller Console**.
6. Apply for releasing a SaaS product on the **Seller Console**.
7. The KooGallery operations team reviews the SaaS product. Once approved, the product is released successfully.
8. Manage interface notifications on the **Seller Console**.

1.2 Interface Description

Before releasing a SaaS product to KooGallery, develop a production interface on your server by referring to this access guide.

The following table lists the SaaS 2.0 interface specifications.

Scenario	SaaS		
	One-Time Payment	Yearly/ Monthly	Pay-per-Use
Creating an instance	Mandatory	Mandatory	Mandatory
Querying instance information	Mandatory	Mandatory	Mandatory
Updating an instance	N/A	Mandatory	Mandatory
Updating the instance status	N/A	Mandatory	Mandatory
Releasing an instance	Mandatory	Mandatory	Mandatory
Upgrading an instance	N/A	Optional	N/A

NOTE

When releasing a product, develop interfaces based on the delivery method and billing mode as listed in the preceding table.

If a product can be upgraded, implement the interface for instance upgrade.

Example: https://www.isvwebsite.com/saasproduce

SaaS 2.0 Instance Dialing Test Rules

After SaaS product release, KooGallery conducts interface address dialing tests every morning and afternoon to monitor the availability for order placement. If a dialing test fails, the system sends a notification to the email address or mobile number bound to your account or as a direct message. Rectify the interface promptly. The following table lists the dialing test rules.

Product Status	Dialing Test
On sale	Required
Discontinued	Skipped
Withdrawn from KooGallery	Skipped

If the dialing test of an interface fails for five consecutive days, the KooGallery operations manager freezes the product transaction and hides the product from customers. According to the **Huawei Cloud KooGallery Partner Product Seller Agreement**, if the fault is not rectified in time, KooGallery has the right to remove the product from the catalog.

Interface Calling Scenario (Order Placement)

- Instance creation: A customer purchases and pays for a product.

a. KooGallery calls the instance creation interface to ask you to create an instance based on the order ID.

b. After receiving the request, your system calls the KooGallery API for querying an order to obtain the subscription information and subscribe to the instance.

c. KooGallery calls the instance query interface to obtain instance information.

- Instance query: A customer queries information about an instance (**instanceId**).

  a. After obtaining the instance ID (**instanceId**) returned by the instance creation interface, KooGallery continuously calls the instance query interface until you successfully return the instance information.

  b. When a customer queries resource information in KooGallery, KooGallery synchronously calls the instance query interface and returns the information.

- Instance update: A customer changes trial use to commercial use, renews resources, or cancels renewals.

  KooGallery calls the instance update interface to notify your system of the resource expiration time. Your system performs the corresponding action and returns the execution result to KooGallery.

- Instance status update: If an instance of a customer expires or the customer violates regulations, the instance will be frozen. After the instance is renewed or the violation is canceled, the instance will be unfrozen.

  KooGallery calls the instance status update interface to ask your system to freeze or unfreeze the corresponding resource and return the execution result to KooGallery.

  NOTE

  When a purchased product expires, the retention period starts. The retention period varies with the customer tier and can be up to 15 days long. During the retention period, the product is frozen and cannot be used. The customer can continue using the product after renewal. Therefore, you need to set a retention period to no less than 15 days for your SaaS products and retain customer data during the retention period.

- Instance release: A customer releases an instance of a purchased product (in scenarios such as no renewal upon expiration and unsubscription).

  a. KooGallery calls the instance release interface to ask your system to release the corresponding resource and return the execution result to KooGallery.

- Instance upgrade: A customer upgrades purchased resources and pays for the upgrade order.

  a. KooGallery calls the instance upgrade interface to ask your system to upgrade the corresponding resource.

  b. Your system calls the KooGallery API for querying an order to obtain the upgrade information, upgrades the instance, and returns the execution result to KooGallery.

  For details about the instance upgrade process, see Upgrade and Billing Rules.

- Tenant synchronization

a. After a customer purchases a joint operations SaaS, logs in to Huawei Cloud console, and binds the app with their enterprise, KooGallery calls this interface to ask you to synchronize the tenant information of the enterprise, save the tenant information, and return the result to KooGallery.

b. If the interface fails to be called, the customer can retry calling. You will be notified of the calling failure via SMS message or email.

c. **When processing an interface request, your server must ensure idempotency.**

For the same instance under a tenant, your server must support multiple requests for adding or deleting the instance. For subsequent additions, no new data should be generated and a success message must be returned. For deletions, you do not need to verify whether the instance generated after the instance creation interface is called exists, but you should return a success message after each deletion.

- App synchronization

a. After a customer purchases a joint operations SaaS, logs in to Huawei Cloud console, and binds the app with their enterprise, KooGallery calls this interface to ask you to synchronize the authentication information of the app, save the app information, and return the result to KooGallery.

b. If the interface fails to be called, the customer can retry calling. You will be notified of the calling failure via SMS message or email.

c. **When processing an interface request, your server must ensure idempotency.**

For the same instance of the same app under a tenant, your server must support multiple requests for adding or deleting the instance. For subsequent additions, no new data should be generated and a success message must be returned. For deletions, you do not need to verify whether the instance generated after the instance creation interface is called exists, but you should return a success message after each deletion.

- App authorization

a. After an administrator is authorized to manage an enterprise, logs in to Huawei Cloud console, and authorizes users in the enterprise to use apps bound to the enterprise, KooGallery asynchronously calls this interface to ask you to synchronize the user authorization information of the apps, save the information, and return the result to KooGallery.

b. If the interface fails to be called, the administrator can retry synchronization. You will be notified of the calling failure via SMS message or email.

c. **When processing an interface request, your server must ensure idempotency.**

For the same instance of the same app and **userName** under a tenant, your server must support multiple requests for adding or deleting the instance. For subsequent additions, no new data should be generated and a success message must be returned. For deletions, you do not need to verify whether the instance generated after the **instance creation interface** is called exists, but you should return a success message after each deletion.

- Organization synchronization (incremental)
  a. After an administrator is authorized to manage an enterprise, logs in to Huawei Cloud console, and creates, edits, or deletes departments for the enterprise, KooGallery calls this interface to ask you to synchronize organization changes, save the organization information, and return the result to KooGallery.
  b. **When processing an interface request, your server must ensure idempotency.**

     For the same instance of the same **orgCode** under a tenant, your server must support multiple requests for adding or deleting the instance. For subsequent additions, no new data should be generated and a success message must be returned. For deletion, you do not need to verify whether the instance generated after the **instance creation interface** is called exists, but you should return a success message after each deletion.
- Organization synchronization (full)
  a. After a customer purchases a joint operations SaaS, logs in to Huawei Cloud console, and binds the app with their enterprise, KooGallery calls this interface to ask you to synchronize all organization information of the enterprise, save the organization information, and return the result to KooGallery.
  b. **When processing an interface request, your server must ensure idempotency.**

     For the same instance of the same **orgCode** under a tenant, your server must support multiple requests. No new data should be generated and a success message must be returned.

Interface Failure Scenarios and Retry Mechanism

If an interface fails to respond, the system sends an email to the email address bound to your KooGallery account. You can query the exception information on the **Service Interface Messages** page of the Seller Console. Handle the exceptions as soon as possible to avoid unsubscription due to order failure.

- When the instance creation interface fails to be called, KooGallery retries for 3 hours. You can click **Restart** in the **Operation** column on the right of the order on the **Service Interface Messages** page to retry calling. If the interface exception persists after 3 hours, the system determines that the order fails and cancels the order.
- When the instance update interface fails to be called, KooGallery retries for an hour. You can query the exception information on the **Service Interface Messages** page. After rectifying the exception, click **Restart** in the **Operation** column on the right of the order on the **Service Interface Messages** page to retry calling.
- When the instance status update interface fails to be called, KooGallery retries for an hour. You can query the exception information on the **Service Interface Messages** page. After rectifying the exception, click **Restart** in the **Operation** column on the right of the order on the **Service Interface Messages** page to retry calling.
- When the instance upgrade interface fails to be called, KooGallery retries for 3 hours. You can click **Restart** in the **Operation** column on the right of the

order on the **Service Interface Messages** page to retry calling. If the interface exception persists after 3 hours, the system determines that the order fails and cancels the order.

NOTE

If an interface fails to respond, an email, SMS message, or direct message will be sent to you. Check the email address and mobile number bound to your account and the Message Center on Huawei Cloud.

If more than five orders of a product failed in a month due to interface failures or an interface fails the dialing test for five consecutive days, KooGallery will remove the product from the catalog.

If an order is automatically canceled due to an interface failure, contact the customer at the earliest to handle the problem.

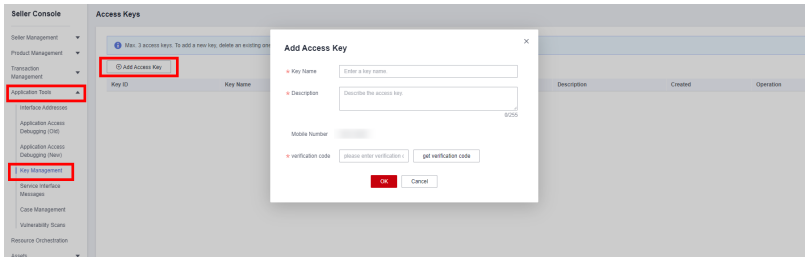
If a customer can still use expired resources due to an interface failure, you shall bear the resource loss incurred.

1.3 Preparations

1.3.1 Obtaining an Access Key (Key ID)

Step 1 Go to the **Seller Console**.

Step 2 In the navigation pane, choose **Application Tools** > **Key Management**. On the displayed page, click **Add Access Key**. After adding a key, obtain the access key.



NOTE

The access key is used to verify the signature of interface requests. Keep it secure.

----End

1.3.2 HTTP Body Signature

Definition

Each time KooGallery calls your interface, KooGallery generates a signature for the request based on certain rules and adds the signature to the URL as a URL parameter. After receiving the request, you need to recalculate the signature for the request body based on the same rules, compare the signature with the signature transferred by KooGallery. If they are the same, the verification succeeds. The following table lists the parameters transferred.

Parameter	Value	Description
signature	String	Encrypted signature, which is generated by signing a request based on certain rules.
timestamp	Long	UNIX timestamp, in seconds. The difference between the timestamp and the current time must not exceed 60 seconds.
nonce	String	Randomly generated by KooGallery each time an interface is called. You can cache it to defend against replay attacks.

Generation Rules

1. Sort request parameters by name (from Z to A). For example, a parameter whose name starts with **a** is placed after a parameter whose name starts with **b**.
2. Obtain the standard request character string.

```
canonicalRequest = accessKey + nonce + timestamp + Lowercase(HexEncode(HMAC_SHA256(RequestPayload)))
```

1. Obtain the **signature** value for the key.

```
signature = HexEncode(HMAC_SHA256(canonicalRequest))
```

Example

The following is an example of the request received by you:

```
curl -X POST -H 'Content-Type: application/json' 'https://www.isvwebsite.com/saasproduce?signature=af71c5a7ef45310b8dc05ab15f7da50189ffa81a95cc284379ebaa5eb61155c0&timestamp=1666677988730&nonce=RLLUammMSInlrNWb' --data '{"activity":"newInstance","buyerInfo":{"customerId":"688055390f3049f283fe9f1aa90f1858","customerName":"CBC_marketplace_mwx616072_01","userId":"1e86066c22754361933f607df834e4fe","userName":"CBC_marketplace_mwx616072_01","mobilePhone":"18652996659","email":"mapengfei8@huawei.com"},"orderInfo":[{"businessId":"8a2c4e6f-405a-4f8d-8e24-f41090522646","orderId":"CS2210101920BWXLK","trialFlag":"0","orderAmount":12.78,"chargingMode":"PERIOD","periodType":"month","periodNumber":5,"provisionType":1,"productInfo":[{"skuCode":"a63ee5c9-4f86-11ed-9f95-fa163e8cb3b2","productId":"OFFI788963615933718528","linearValue":20}],"createTime":"20221024194509","expireTime":"20221224194509","extendParams":[{"name":"emailDomainName","value":"test.xxxx.com"},{"name":"extendParamName","value":"extendParamValue"}]}],"testFlag":"1"}'
```

1.4 Basic Interfaces

1.4.1 Creating an Instance

Description

After a customer purchases and pays for a product, KooGallery calls this interface to ask you to create an instance.

- You must return the unique ID (**instanceId**) of the order. Use **businessId** provided by KooGallery to ensure that **instanceId** is globally unique.

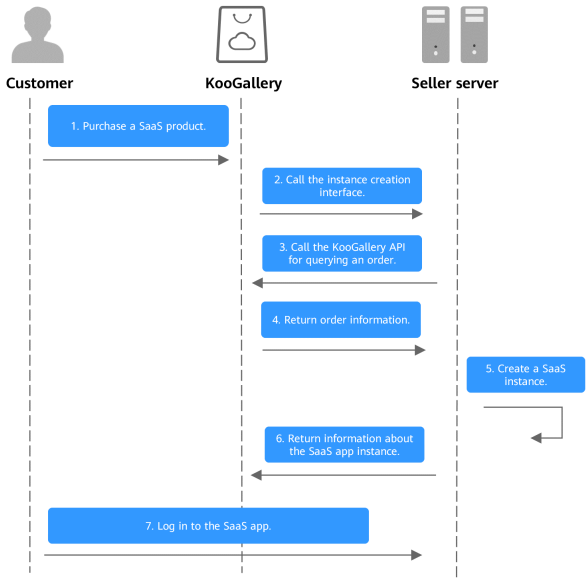
- Do not block this interface. If instance creation is time-consuming, create the instance asynchronously. You can generate an instance ID first and then return a response immediately. KooGallery will use the interface for querying instance information to query the instance provisioning result.
- If pay-per-use specifications and packages are involved, develop the interface for **Pushing the Pay-per-Use Resource Usage (New)**.
- For details about how to obtain order information, see **Querying an Order**.

NOTE

KooGallery may resend a request. For the same order ID (**orderId**) and order line (**orderLineId**), your server should return the same **instanceId** without creating a SaaS instance.

In pay-per-use transactions, ensure idempotency based on the order ID (**orderId**) and product ID (**productId**).

The following figure shows the process of creating an instance.



Request Message

The following table describes the request parameters. In KooGallery, requests are generated based on the subscription mode of products released by you. You need to provide services based on requests.

Request method: POST

Body parameters

Parameter	Mandatory	Type	Maximum Length	Description
activity	Yes	String	20	Request ID, which is used to distinguish the scenario. For new subscriptions, the value is **newInstance**.
orderId	Yes	String	64	KooGallery order ID.
orderLineId	Yes	String	64	KooGallery order line ID.
businessId	Yes	String	64	KooGallery business ID. The value of **businessId** is different for each request.
testFlag	No	String	2	Whether a request is submitted for debugging. ● **1**: debugging request. ● **0**: non-debugging request. The default value is **0**.

Example request:

```
curl -X POST 'https://www.isvwebsite.com/saasproduce?
signature=11C4CD6279191DE931DEF5C51531DFFA9D37969F4E356B8A3A6D8DE4FB357A48&timestamp=16
80508066618&nonce=50D83FDECAED6CCD8EF597F2A577950527928BA287D04E6036E92B2806FD17DA' -H
'Accept:application/json' -H 'Content-Type:application/json;charset=utf8' -
d'{"activity":"newInstance","businessId":"87b94795-0603-4e24-8ae5-69420d60e3c8","orderId":"CS221118181
9B4LVS","orderLineId":"CS2211181819B4LVS-000001","testFlag":"0"}'
```

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	6	Result code. • **000000**: The resource is enabled synchronously. • **000004**: The resource is enabled asynchronously. For details, see **1.6 Result Codes**. **NOTE** Return **0000004** if it takes a long time to create an instance. KooGallery will call the instance information query interface to query the instance provisioning result.
resultMsg	No	String	255	Result message.
instanceId	Yes	String	64	KooGallery business ID.

Example response:

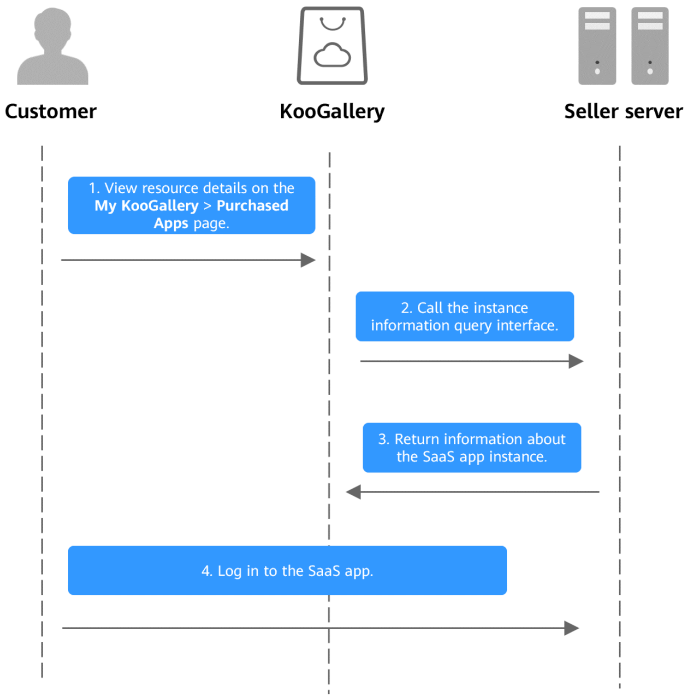
```
{
   "resultCode":"000000",
   "resultMsg":"success." ,
   "instanceId":"03pf80c2bae96vc49b80b917bea776d7"
}
```

1.4.2 Querying Instance Information

Description

After you create an instance, KooGallery queries the instance information based on the instance ID.

The following figure shows the process of querying instance information.



Request Message

The following table describes the request parameters.

Request method: POST

Body parameters

Parameter	Mandatory	Type	Maximum Length	Description
activity	Yes	String	20	Request ID, which is used to distinguish the scenario. In the upgrade scenario, the value is **queryInstance**.
instanceId	Yes	String	100	Instance IDs separated by commas (,). Up to 100 instances can be queried each time.
testFlag	No	String	2	Whether a request is submitted for debugging. • **1**: debugging request. • **0**: non-debugging request. The default value is **0**.

Example request:

```
curl -X POST 'https://www.isvwebsite.com/saasproduce?
signature=9C61F188C3C2889C2DD201B00E42041BDCE4751F31E35805DE412969F0A7829C&timestamp=168
0508237508&nonce=9FB42E04DF4594B1FAA50B304E647AD7154AB9B4F144A65F1168886540A8B24C' -H
'Accept:application/json' -H 'Content-Type:application/json;charset=utf8' -
d'{"activity":"queryInstance","instanceId":"10e758d0-31ad-4c4b-8f1b-81d03469a10e","testFlag":"0"}'
```

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	6	Result code. For details, see **1.6 Result Codes**.
resultMsg	No	String	255	Result message.
info	No	InstanceInfo[]		Instance details.

The following table describes the **InstanceInfo** data structure.

Parameter	Mandatory	Type	Maximum Length	Description
instanceId	Yes	String	64	Instance ID.

Parameter	Mandatory	Type	Maximum Length	Description
appInfo	No	AppInfo	N/A	App instance information. After a customer purchases a product, return a login address (website address) or an address that does not require login for the customer to perform subsequent operations. **NOTE** You must provide customers who purchase your SaaS products with the app usage information, including the addresses, accounts, and passwords. If the usage information can be sent through SMS messages, emails, or other methods, this parameter is not required in the response. Otherwise, the app instance information must be returned in the response. You can use the **memo** parameter to specify usage instructions or other information if any. For details about the **appInfo** data structure, see the following table.

The following table describes the **appInfo** data structure.

Parameter	Mandatory	Type	Maximum Length	Description
frontEndUrl	Yes	String	512	Frontend URL. URL of the website that the customer can access to use the purchased product.
adminUrl	No	String	512	Management URL. URL of the backend website that the customer can access to manage the purchased product.
userName	No	String	128	Administrator account.
password	No	String	128	Initial password of the administrator.
memo	No	String	1,024	Remarks.

Example response:

```
{
   "resultCode": "000000",
   "resultMsg": "success.",
   "Info": [
{
          "instanceId": "huaweitest123",
          "appInfo": {
              "frontEndUrl": "https://www.baidu.com",
              "userName": "zhangsan123",
              "password": "zhangsan123",
              "memo": "Test"
}
          ]
}
```

1.4.3 Updating an Instance

Description

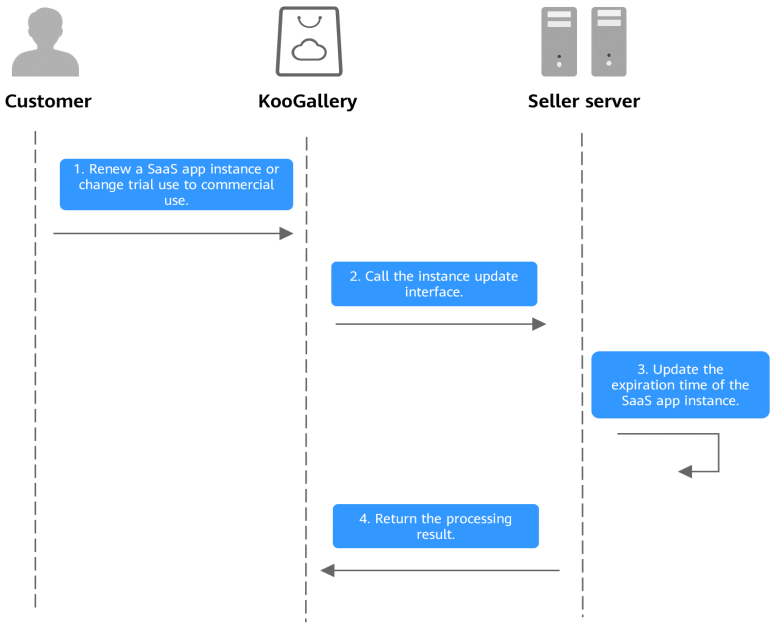
KooGallery calls this interface to update the expiration date of an instance after a customer purchases a product after trial use, renews the product, or cancels the renewal.

- When this interface is called, update the expiration date and return a notification to KooGallery.
- Ensure that this interface is normal. If the interface fails to be called, customer services may be released.

NOTE

- If you receive an email indicating that the interface fails to be called in your email address of customer service or that one bound to your KooGallery account, handle the interface exception in a timely manner.
- KooGallery monitors interface exceptions. If a product has frequent instance exceptions, KooGallery will remove the product from the catalog.

The following figure shows the process of updating an instance.



Request Message

The following table describes the request parameters.

Request method: POST

Body parameters

Parameter	Mandatory	Type	Maximum Length	Description
activity	Yes	String	20	Request ID, which is used to distinguish the scenario. For renewals, the value is **refreshInstance**.
scene	Yes	String	64	Scenario where the instance change is triggered. **TRIAL_TO_FORMAL**: trial use to commercial use. **RENEWAL**: renewal. **UNSUBSCRIBE_RENEWAL_PERIOD**: renewal cancellation.

Parameter	Mandatory	Type	Maximum Length	Description
orderId	Yes	String	64	KooGallery order ID. The commercial order ID is transferred when the customer purchases the product after trial use. The renewal order ID is transferred during renewal. The ID of the renewal order to be cancelled is transferred during renewal cancellation.
orderLineId	Yes	String	64	KooGallery order line ID.
instanceId	Yes	String	64	Instance ID.
productId	No	String	64	Product ID. If a customer renews a product and changes the billing cycle or a customer purchases a product after trial use, a new **productId** is provided.
expireTime	Yes	String	20	Expiration time. Format: yyyyMMddHHmmss
testFlag	No	String	2	Whether a request is submitted for debugging. • **1**: debugging request. • **0**: non-debugging request. The default value is **0**.

Example request:

```
curl -X POST 'https://www.isvwebsite.com/saasproduce?
signature=3F6E6652B7BE26B27ABFC3D11214D04BFD8D2CF8AC21603D85620174FE8DE062&timestamp=16
80509496350&nonce=8BF8496A350E37BDB0E8956D39D433ED417C3FC9459DCFFE7F03BFBF69B12085' -H
'Accept:application/json' -H 'Content-Type:application/json;charset=utf8' -
d'{"activity":"refreshInstance","expireTime":"20221124023618256","instanceId":"10e758d0-31ad-4c4b-8f1b-8
1d03469a10e","orderId":"CS2211181819B4LVS","orderLineId":"CS2211181819B4LVS-000001","productId":"O
FFI461867333479178240","scene":"RENEWAL","testFlag":"0"}'
```

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	6	Result code. For details, see **1.6 Result Codes**.

Parameter	Mandatory	Type	Maximum Length	Description
resultMsg	No	String	255	Result message.

Example response:

```
{
   "resultCode":"000000",
   "resultMsg":"success."
}
```

1.4.4 Updating the Instance Status

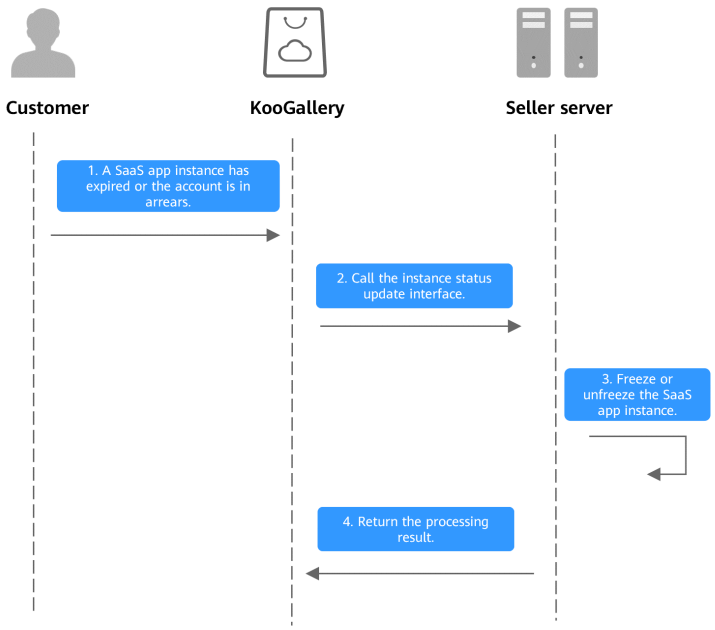
Description

After a customer purchases a yearly/monthly/daily product and the instance expires or the customer violates regulations, KooGallery calls this interface to freeze the instance.

NOTE

- If you receive an email indicating that the interface fails to be called in your email address of customer service or that one bound to your KooGallery account, handle the interface exception in a timely manner.
- KooGallery monitors interface exceptions. If a product has frequent instance exceptions, KooGallery will remove the product from the catalog.

The following figure shows the process of updating the instance status.



The following table describes the request parameters.

Request Message

Request method: POST

Body parameters

Parameter	Mandatory	Type	Maximum Length	Description
activity	Yes	String	32	Request ID, which is used to distinguish the scenario. For instance status updates, the value is **updateInstanceStatus**.
instanceId	Yes	String	64	Instance ID.
status	Yes	String	32	New status. • **FREEZE**: frozen. • **UNFREEZE**: unfrozen.
testFlag	No	String	2	Whether a request is submitted for debugging. • **1**: debugging request. • **0**: non-debugging request. The default value is **0**.

Example request:

```
curl -X POST 'https://www.isvwebsite.com/saasproduce?
signature=95DD9FA6A8C660C9C7F9CFDE97C42535290919BCA3F78B9A254428A692CDF26E&timestamp=16
80509558159&nonce=9F26B85CAEB3A8439221BA293E9250BC5EA689225B523C291EA75CC76B469510' -H
'Accept:application/json' -H 'Content-Type:application/json;charset=utf8' -
d'{"activity":"updateInstanceStatus","instanceId":"10e758d0-31ad-4c4b-8f1b-81d03469a10e","status":"FREEZ
E","testFlag":"1"}'
```

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	6	Result code. For details, see **1.6 Result Codes**.

Parameter	Mandatory	Type	Maximum Length	Description
resultMsg	No	String	255	Result message.

Example response:

```
{
   "resultCode":"000000",
   "resultMsg":"success."
}
```

1.4.5 Releasing an Instance

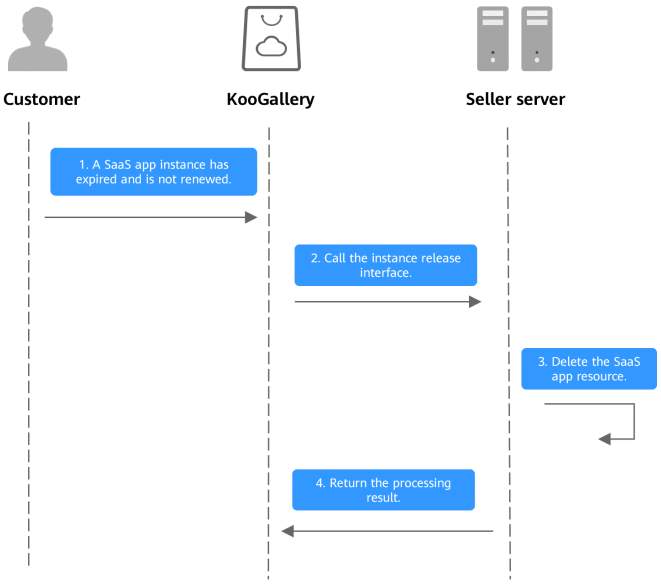
Description

When an instance of a purchased product is released (for example, the instance is not renewed upon expiration or unsubscribed from), KooGallery calls this interface to delete the instance.

NOTE

- If you receive an email indicating that the interface fails to be called in your email address of customer service or that one bound to your KooGallery account, handle the interface exception in a timely manner.
- KooGallery monitors interface exceptions. If a product has frequent instance exceptions, KooGallery will remove the product from the catalog.

The following figure shows the process of expiration.



Request Message

The following table describes the request parameters.

Request method: POST

Body parameters

Parameter	Mandatory	Type	Maximum Length	Description
activity	Yes	String	32	Request ID, which is used to distinguish the scenario. For expiration, the value is **releaseInstance**.
instanceId	Yes	String	64	Instance ID.
orderId	No	String	64	This parameter is required when an instance is released due to unsubscription.
orderLineId	No	String	64	KooGallery order line ID.
testFlag	No	String	2	Whether a request is submitted for debugging. • **1**: debugging request. • **0**: non-debugging request. The default value is **0**.

Example request:

```
curl -X POST 'https://www.isvwebsite.com/saasproduce?
signature=C4E5F264C92F737DEBECB8D27D84684F38BF01D2917880202B59027CEEFC4932&timestamp=168
0509885590&nonce=A49E8F86EE5BCAFBDFD3E53F1E09A29C6D9E8DACC67382EBCDD02CD55CBBB7AE' -H
'Accept:application/json' -H 'Content-Type:application/json;charset=utf8' -
d'{"activity":"releaseInstance","instanceId":"10e758d0-31ad-4c4b-8f1b-81d03469a10e","orderId":"CS2211181
819B4LVS","orderLineId":"CS2211181819B4LVS-000001","testFlag":"0"}'
```

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	6	Result code. For details, see **1.6 Result Codes**.
resultMsg	No	String	255	Result message.

NOTE

- If you receive an email indicating that the interface fails to be called in your email address of customer service or that one bound to your KooGallery account, handle the interface exception in a timely manner.
- KooGallery monitors interface exceptions. If a product has frequent instance exceptions, KooGallery will remove the product from the catalog.

Example response:

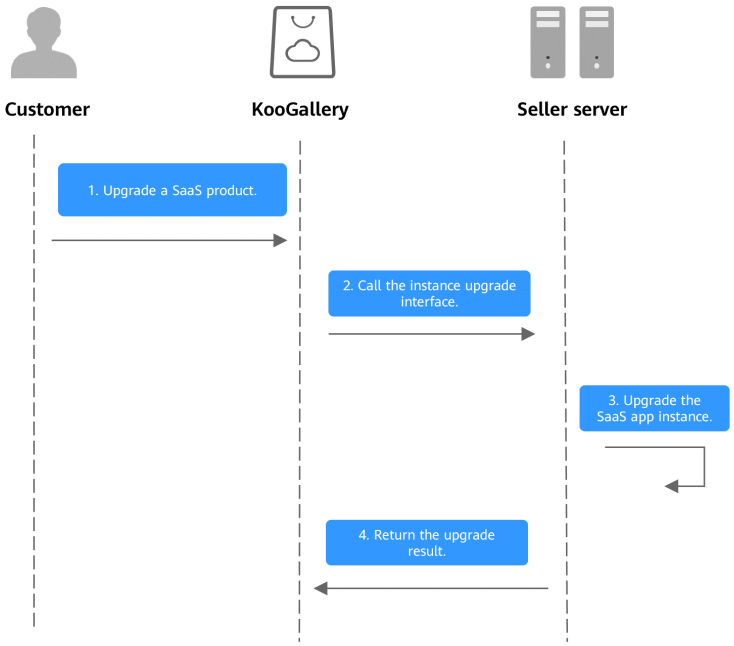
```
{
   "resultCode":"000000",
   "resultMsg":"success."
}
```

1.4.6 Upgrading an Instance

Description

A customer upgrades a purchased resource. After the upgrade order is paid, KooGallery calls this interface to ask you to upgrade the resource and record the upgraded product data.

The following figure shows the process of upgrading a product.



Request Message

The following table describes the request parameters.

Request method: POST

Parameter	Mandatory	Type	Maximum Length	Description
activity	Yes	String	32	Request ID, which is used to distinguish the scenario. For upgrades, the value is **upgradeInstance**.
instanceId	Yes	String	64	Instance ID. **NOTE** The upgrade does not change **instanceId**.
orderId	Yes	String	64	Upgrade order ID. **NOTE** An order is generated for the upgrade.
orderLineId	Yes	String	64	KooGallery order line ID.
testFlag	No	String	2	Whether a request is submitted for debugging. • **1**: debugging request. • **0**: non-debugging request. The default value is **0**.

Example request:

```
curl -X POST 'https://www.isvwebsite.com/saasproduce?
signature=9D49F9BF09D69F7A98B847978D6091A9ADF3B40E07AF95FEE9E5BEF5218DA407&timestamp=16
80510876429&nonce=D8FE86FA6ABE90CA63A72B3256743D3D869648FE99A96354E635F032629F6C21' -H
'Accept:application/json' -H 'Content-Type:application/json;charset=utf8' -
d'{"activity":"upgradeInstance","instanceId":"10e758d0-31ad-4c4b-8f1b-81d03469a10e","orderId":"CS221118
1819B4LVS","orderLineId":"CS2211181819B4LVS-000001","testFlag":"0"}'
```

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	6	Result code. For details, see **1.6 Result Codes**.
resultMsg	No	String	255	Result message.

Example response:

```
{
   "resultCode":"000000",
```



```
    "resultMsg":"success."
}
```

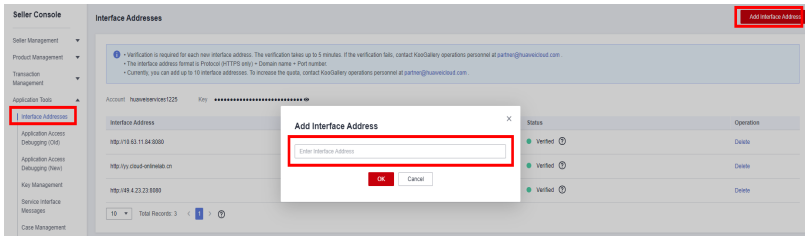
1.5 Interface Debugging

To ensure that SaaS products can be accessed, KooGallery provides a debugging page on the Seller Console. You can debug SaaS interfaces in each scenario.

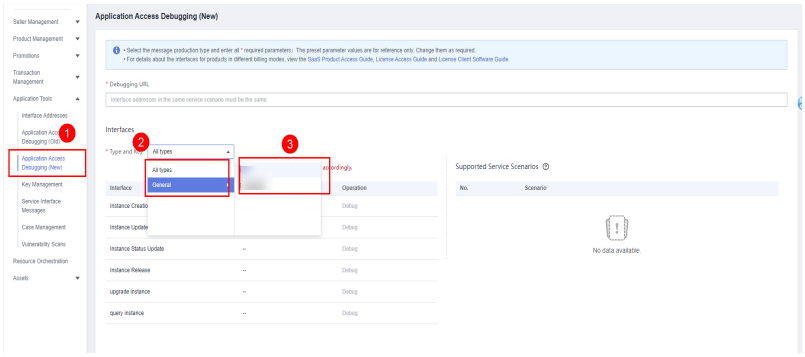
The following uses the interface for creating an instance as an example.

Procedure

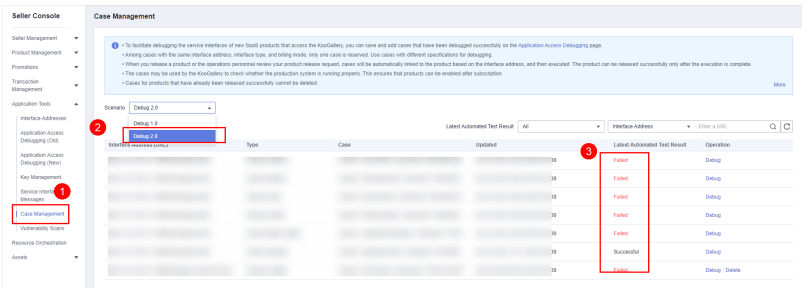
Step 1 Choose **Application Tools > Interface Addresses** in the navigation pane of the Seller Console, add an interface address, and complete the verification.



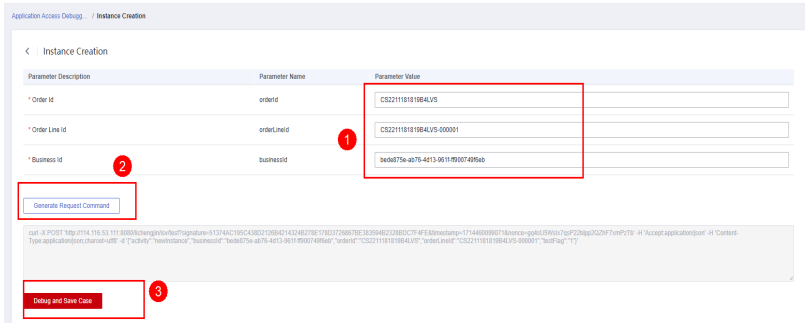
Step 2 Bind the verified interface address to the key. For details about how to obtain the key, see **1.3.1 Obtaining an Access Key (Key ID)**.



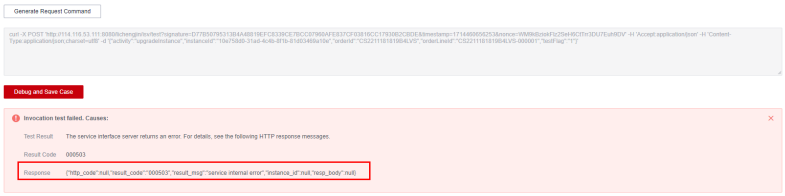
Step 3 Click **Debug** to debug the interface using parameters preset in your system based on the request parameter description of the interface.



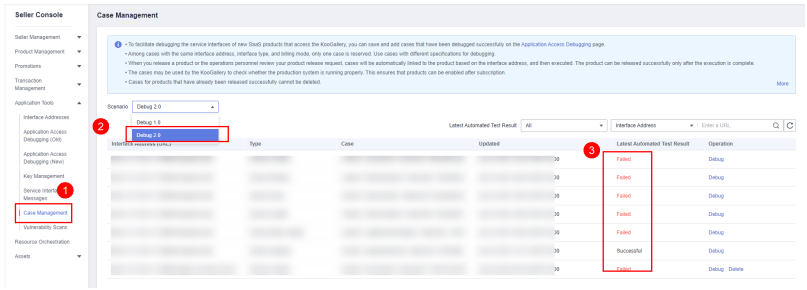
Step 4 On the **Instance Creation** page, enter values of preset parameters and click **Generate Request Command** to generate a request example.



Step 5 Click **Debug and Save Case**. The system calls the interface address to test the interface. If the test is successful, the system displays a message indicating debugging is successful and the case is saved. If the test fails, the error message is displayed in the lower part of the page. You can modify the interface based on the error message.



Step 6 When the debugging is successful, choose **Application Tools > Case Management** in the navigation pane and view the test case.



----End

1.6 Result Codes

Module	Result Code	Description
Common	000000	Succeeded.
	000001	Authentication failed.
	000002	Invalid request parameter.
	000003	The instance ID does not exist. (This result code may be returned when the renewal, expiration, or resource release interface is called.)
	000004	The request is being processed.

Module	Result Code	Description
	000005	Other internal errors.
Instance creation	000100	No instance resource can be allocated.
	000101	Mobile number not specified or already exists.
	000102	Email address not specified or already exists.
	000103	Product already purchased.
	000104	Internal service error.
	000105	Account has already subscribed to benefits.
	000106	Product resources sold out.
	000107	SaaS email domain already exists.

1.7 KooGallery Open APIs

1.7.1 Using APIs

1.7.1.1 Usage

Huawei Cloud provides RESTful APIs.

REST allocates Uniform Resource Identifiers (URIs) to dispersed resources so the resources can be located. Applications on clients use unified resource locators (URLs) to access the resources.

A URL is in the format of https://*Endpoint*/*uri*.

Table 1-1 describes the parameters in a URL.

Table 1-1 URL parameters

Parameter	Description
Endpoint	Entry (URL) for a web service. The KooGallery endpoint is **mkt-intl.myhuaweicloud.com**.
uri	Resource path, that is, the API access path. Obtain this value from the URI of the API, for example, **v1.0/***{partner_id}***/billing/bill-mgr/push-usage-data**. For the user ID parameter in the URI: **partner_id** is used, indicating that the API can be called only using the AK/SK or token of a partner (seller).

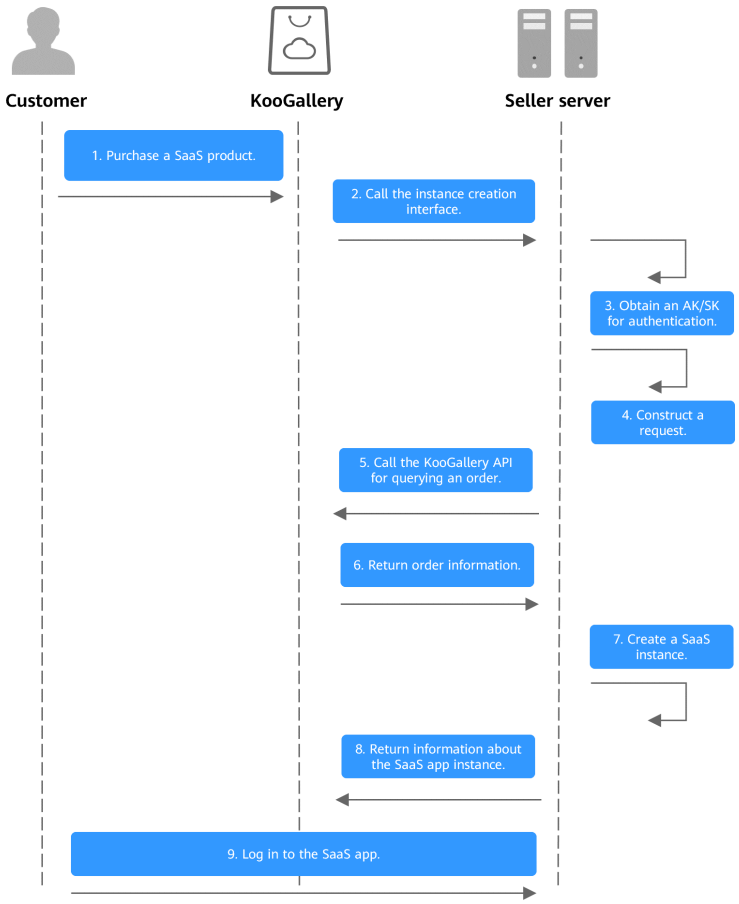
Huawei Cloud APIs use the HTTP transmission protocol and have the following restrictions:

1. Request and response messages are encoded using UTF-8 and in the JSON format.
2. The media type is Application/json.
3. Optional parameters do not need to be encoded in message bodies.
4. UTC time (including the time zone) is used in requests and responses. The format is yyyyMMdd'T'HHmmss'Z'.

   HH ranges from 0 to 23, and *mm* and *ss* range from 0 to 59.

1.7.1.2 Calling

The following figure shows the process of calling an API.



1. Obtain an AK/SK for authentication.

   Request authentication is required for calling an API. After successful authentication, the authentication field is filled into the next method to request for message construction.

   AK/SK authentication is used for calling APIs.

   AK/SK authentication: Requests are encrypted using AK/SK pairs.

When you send requests to underlying services through API Gateway (APIG), the requests must be signed using access key ID (AK) and secret access key (SK).

- AK: access key ID, which is a unique identifier used with a secret access key to sign requests cryptographically.
- SK: secret access key. It is used together with an access key ID to identify a sender who initiates a request and to cryptographically sign requests, preventing the request from being modified.

NOTE

- For details about how to obtain SDKs, see **Calling APIs Through App Authentication**.
- You can obtain an AK/SK on the **Access Keys** page.

2. Construct a request.

   Configure the request parameters to construct a request.
3. Initiate the request.
4. Parse a response.

1.7.1.3 AK/SK Authentication

Generating an AK and SK

1. Register with Huawei Cloud and log in to the management console.
2. Click the username and choose **My Credentials** from the drop-down list.
3. In the navigation pane, choose **Access Keys**.
4. Click **Create Access Key**.
5. Enter the SMS or email verification code and click **OK** to download the access key. Keep the access key secure.

Signing a Request

API requests sent by third-party applications to Huawei Cloud must be authenticated using signatures.

Preparation

1. **Download the APIG signing tool** and decompress it.
2. Create a Java project and reference the decompressed JAR file to the dependency path.

Procedure

1. Create a **com.cloud.sdk.DefaultRequest (JAVA)** request used for signing.
2. Set the destination API URL, HTTPS method, and content of **DefaultRequest**.
3. Sign **DefaultRequest**.
   a. Call **SignerFactory.getSigner(String serviceName, String regionName)** to obtain a signature tool instance.
   b. Call **Signer.sign(Request<?> request, Credentials credentials)** to sign the request created in step 1.

The following code shows the details.

```
// Select the signing algorithm for signing the request.
 Signer signer = SignerFactory.getSigner(serviceName, region);
// Sign the request. The request will change after being signed.
 signer.sign(request, new BasicCredentials(this.ak, this.sk));
```

4. Convert the request signed in the previous step to one that can be used to make an API call and copy the header of the signed request to the new request.
5. For example, for Apache HttpClient, convert **DefaultRequest** into **HttpRequestBase** and copy the header of the signed **DefaultRequest** to **HttpRequestBase**.

1.7.1.4 Constructing a Request

A request consists of three parts: a request line, request header, and request body (optional).

Request Line

A request line starts with the request method, which is followed by the uniform resource identifier (URI) and protocol version. The request method and URI are separated by a space. The request line format is as follows:

```
Method Request-URI HTTP-Version CRLF
```

- **Method**: request method. All methods are capitalized and their meanings are as follows:
  - GET: obtains the resource identified by the Request-URI.
  - POST: suffixes new data to the resource identified by the Request-URI.
  - PUT: stores a resource identified by the Request-URI.
  - DELETE: deletes the resource identified by Request-URI.
- **Request-URI**: uniform resource identifier.

  NOTE

  A combination of different query conditions can be added at the end of the URI by using question marks (?) and ampersands (&). The content contained in **{}** in the URI is the parameters of the URI, where **?** is contained. The part preceding **?** is the path parameter, and the part following **?** is the query parameter. **HTTP-Version**: version of the HTTP protocol used by a request.
- **CRLF**: carriage return and line feed characters. CRLF is placed only at the end of a line. CR and LF must be present at the same time.

Request Header

A request header consists of several fields, each including a domain name, colon (:), and field value. For details, see **1.7.2.1 Common Request Header Fields**.

Request Body

A request body is a JSON-based, nested *key:value* pair. The mandatory and optional fields of an HTTP request body vary depending on the URI.

1.7.1.5 Initiating a Request

You can initiate a request using the constructed request message in either of the following ways:

- cURL

  cURL is a command-line tool used to perform URL operations and transmit information. It serves as an HTTP client that can send HTTP requests to the server and receive response messages. cURL is used for API debugging. For more information about cURL, visit **https://curl.haxx.se/**.
- Encoding

  You can call APIs using code to assemble, send, and process request messages.

1.7.1.6 Parsing a Response

After receiving and interpreting a request message, the server returns an HTTP response message.

A response consists of three parts: status line, response header, and response body.

Status Line

The format of the status line is as follows:

```
HTTP-Version Status-Code Reason-Phrase CRLF
```

- **HTTP-Version**: version of the HTTP protocol used by the server.
- **Status-Code**: status code in the response returned by the server.

  A status code consists of three digits. The first digit defines the class of response. There are five values for the first digit:
  - **1*xx***: informational. The request was received, continuing process.
  - **2*xx***: successful. The request was successfully received, understood, and accepted.
  - **3*xx***: redirection. Further action needs to be taken to complete the request.
  - **4*xx***: client error. The request contains bad syntax or cannot be fulfilled.
  - **5*xx***: server error. The server failed to fulfill an apparently valid request.
- **Reason-Phrase**: text description of a status code.

Response Header

A response header usually contains the response headers listed in **1.7.1.7 Status Codes**.

Response Body

The response body is in JSON format.

1.7.1.7 Status Codes

The following table lists HTTP response status codes.

Table 1-2 HTTP response status codes

Status Code	Message	Description
100	Continue	Continue sending requests. This temporary response is used to inform the client that some requests have been received and not rejected by the server.
101	Switching Protocols	The protocol is switched. The target protocol must be more advanced than the original one. For example, the protocol in use is switched to a later version of HTTP.
201	Created	The request for creating resources has been fulfilled.
202	Accepted	The request has been accepted for processing, but the processing has not been completed.
203	Non-Authoritative Information	The request was successful but the response has been modified by a transforming proxy.
204	No Content	The request has been fulfilled, but the HTTP response does not contain a response body. The status code is returned in response to an HTTP OPTIONS request.
205	Reset Content	The server has fulfilled the request and requires the client to reset the content.
206	Partial Content	The server has fulfilled a range GET request.
300	Multiple Choices	There are multiple options for the requested resource. The response contains a list of resource characteristics and addresses from which the user or user agent (such as a browser) can choose the most appropriate one.
301	Moved Permanently	The requested resource has been assigned with a new permanent URI. This new URI is contained in the response.
302	Found	The requested resource resides temporarily under a different URI.
303	See Other	The server is redirecting the client to a different address. The client should use a GET or POST method to obtain the resource.
304	Not Modified	The requested resource has not been modified. When the server returns this status code, no resource is returned.

Status Code	Message	Description
305	Use Proxy	The requested resource is available only through a proxy.
306	Unused	This HTTP status code is no longer used.
400	Bad Request	Invalid request. The client should not repeat this request without modification.
401	Unauthorized	The authentication information provided by the client is incorrect or invalid.
402	Payment Required	This status code is reserved for future use.
403	Forbidden	The request has been rejected. The server received and understood the request but refused to fulfill it, because the request is set to deny access. The client should not repeat this request without modification.
404	Not Found	The requested resource could not be found. The client should not repeat this request without modification.
405	Method Not Allowed	The method specified in the request is not allowed for the requested resource. The client should not repeat this request without modification.
406	Not Acceptable	The server cannot implement the request based on the content characteristics of the request.
407	Proxy Authentication Required	This status code is similar to 401, but the client must be authenticated using a proxy.
408	Request Time-out	The client does not produce a request within the time that the server was prepared to wait. The client may repeat the request without modifications at any time later.
409	Conflict	The request cannot be processed due to a conflict. The resource that the client attempts to create already exists, or the request fails to be processed because of the update of the conflict request.
410	Gone	The requested resource is no longer available. The requested resource has been deleted permanently.

Status Code	Message	Description
411	Length Required	The server fails to process the request which does not contain the **Content-Length** header field.
412	Precondition Failed	The server does not meet one of the requirements that the requester puts on the request.
413	Request Entity Too Large	The request is larger than that the server can process. The server may close the connection to prevent the client from continuously sending the request. If the server cannot process the request temporarily, the response will contain a **Retry-After** header field.
414	Request-URI Too Large	The Request-URI is too long for the server to process.
415	Unsupported Media Type	The server cannot process the media format in the request.
416	Requested Range Not Satisfiable	The requested range is invalid.
417	Expectation Failed	The server fails to meet the requirements of the **Expect** request header field.
422	Unprocessable Entity	The request is well-formed but is unable to be processed due to semantic errors.
429	Too Many Requests	The client sends too many requests to the server within a given time, exceeding the client's access frequency limit or beyond the server's processing capability. In this case, the client should retry after the time period specified in the **Retry-After** response header.
500	Internal Server Error	The server is able to receive the request but unable to understand it.
501	Not Implemented	The server does not support the function required to fulfill the request.
502	Bad Gateway	The server was acting as a gateway or proxy and received an invalid request from a remote server.
503	Service Unavailable	The requested service is invalid. The client should not repeat this request without modification.

Status Code	Message	Description
504	Server Timeout	The request cannot be fulfilled within a given amount of time. The response will reach the client only if the request carries a timeout parameter.
505	HTTP Version Not Supported	The server does not support the HTTP protocol version used in the request.

1.7.1.8 APIGW Error Codes

For details about error codes starting with **APIGW**, see **Error Codes**.

1.7.1.9 Signature Sample Project Code

For details, see **AK/SK Authentication**.

1.7.2 Common Parameters

1.7.2.1 Common Request Header Fields

Table 1-3 Common request header fields

Name	Description	Mandatory	Example
x-sdk-date	Time when a request is sent. The format is yyyyMMdd'T'HHmmss'Z'. The value is the current GMT time of the system.	No Mandatory for AK/SK authentication	20160629T101459Z
Authorization	Authentication information. It is the result of request signing. For details, see **Signing a Request**.	No Mandatory for AK/SK authentication	-

Name	Description	Mandatory	Example
Host	Information about the requested server, in the *hostname[:port]* format. The value can be obtained from the URL of the service API. If the port number is not specified, the default port is used. The default port number for HTTPS is **443**.	No Mandatory for AK/SK authentication	mkt-intl.myhuaweicloud.com
Content-type	MIME type of the body in the request.	Yes	application/json

1.7.2.2 Common Response Header Fields

Table 1-4 Common response header fields

Name	Description	Example
Date	Standard HTTP header, which indicates the date and time when a message is sent. The value is in the format defined in RFC822.	Mon, 12 Nov 2007 15:55:01 GMT
Server	Standard HTTP header, which contains information about the software that the server uses to process requests.	Nginx
Content-Length	Standard HTTP header, which indicates the representation's data length as a decimal number of octets.	xxx
Content-Type	Standard HTTP header, which indicates the media type of the entity body sent to the recipient.	application/json

1.7.3 APIs

1.7.3.1 Querying an Order

Function

KooGallery sellers can use this API to query all order information.

URI

GET:

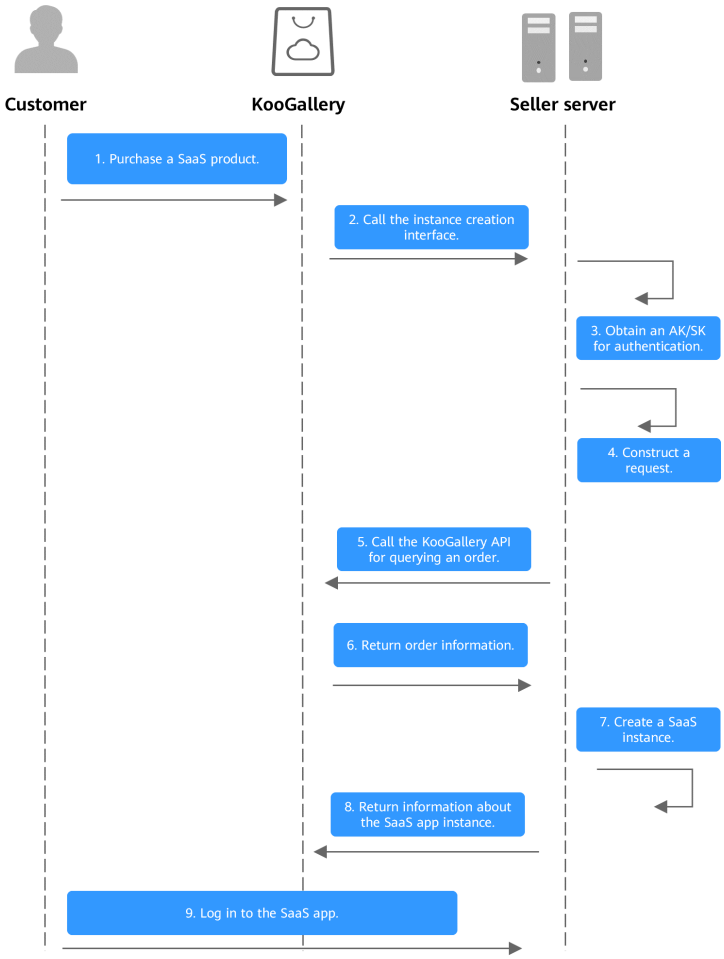
https://mkt-intl.myhuaweicloud.com/api/mkp-openapi-public/global/v1/order/query

Table 1-5 describes the parameters.

NOTE

Only the HTTPS protocol is supported.

The following figure shows the process of querying an order.



The following table lists the order data used for debugging.

Order No.	Order Line ID	Order Type
MOCKPERIODYEARNEW	MOCKPERIODYEARNEW-000001	Subscription - Yearly/Monthly
MOCKONETIMENEW	MOCKONETIMENEW-000001	Subscription - One-time
MOCKONDEMAND	MOCKONDEMAND-000001	Subscription - Pay-per-use specification
MOCKONDEMANDPKG	MOCKONDEMANDPKG-000001	Subscription - Pay-per-use package
MOCKPERIODDAYTRIAL	MOCKPERIODDAYTRIAL-000001	Trial use
MOCKMONTYTRIALTOFORMAL	MOCKMONTYTRIALTOFORMAL-000001	Change from trial use to commercial use
MOCKMONTYUNSUBSCRIBE	MOCKMONTYUNSUBSCRIBE-000001	Unsubscription
MOCKMONTYRENEW	MOCKMONTYRENEW-000001	Renewal
MOCKMONTYCHANGE	MOCKMONTYCHANGE-000001	Change and upgrade

Request Message

Request parameters

Request method: GET

Parameter	Mandatory	Type	Maximum Length	Description
orderId	Yes	String	64	KooGallery order ID.
orderLineId	No	String	64	KooGallery order line ID.

Notes

- Strong verification must be performed on the HTTPS certificate of mkt-intl.myhuaweicloud.com to ensure that the real KooGallery service instead of a forged KooGallery service is called.

Example request

```
GET /api/mkp-openapi-public/global/v1/order/query?
orderId=CS2207261447AUY4H&orderLineId=CS2207261447AUY4H-000001
```



```
Host: Host Server
Content-Type: application/json charset=UTF-8
X-Sdk-Date: request time
Authorization: authorization
```

Table 1-5 Response parameters

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	16	Result code. For details, see **Table 1-7**.
resultMsg	Yes	String	1,024	Result message. For details, see **Table 1-7**.
orderInfo	No	OrderInfo	/	Additional information. **NOTE** For details about the **OrderInfo** data structure, see the following table.

The following table describes the **OrderInfo** data structure.

Parameter	Mandatory	Type	Maximum Length	Description
orderId	Yes	String	64	KooGallery order ID.
orderType	Yes	String	32	Order type. The options are as follows: • **NEW**: new subscription. • **TRIAL**: trial use. • **TRIAL_TO_FORMAL**: commercial use after trial use. • **UNSUBSCRIBE**: unsubscription. • **RENEW**: renewal. • **CHANGE**: change.
createTime	Yes	DateTime	20	Time when an order is created. Format: yyyyMMddHHmmss **NOTE** It is not the time when the order takes effect but the time when the order is placed.

Parameter	Mandatory	Type	Maximum Length	Description
orderLine		List<OrderLine>		Order line information.
buyerInfo	No	BuyerInfo	/	Customer information.

The following table describes the **OrderLine** data structure.

Parameter	Mandatory	Type	Maximum Length	Description
orderLineId	Yes	String	64	KooGallery order line ID.
chargingMode	Yes	String	25	Billing mode. **ON_DEMAND**: pay-per-use. **ONE_TIME**: one-time payment. **PERIOD**: yearly/monthly/daily. **ON_DEMAND_PKG**: pay-per-use package.
expireTime	No	DateTime	20	Expiration time. Format: yyyyMMddHHmmss **NOTE** • This parameter is required for a yearly/monthly/daily product. • This parameter is not required for a product billed by uses. This parameter is determined based on the order creation time and the subscription duration and may differ from the actual expiration time of the order. It is for reference only.
periodType	No	String	2	Period type. **NOTE** This parameter is only required for yearly/monthly/daily subscriptions (**chargingMode** is set to **PERIOD**). Yearly subscription: **year** Monthly subscription: **month**

Parameter	Mandatory	Type	Maximum Length	Description
extendParams	No	List<ExtendParam>	/	Extension parameters. An extension parameter is an array in the key/value format. Example: [{"name":"emailDomainName","value":"test.xxxx.com"}, {"name":"ip","value":"192.168.1.1"}] In the preceding information, **emailDomainName** and **ip** are set during product release.
periodNumber	No	integer	5	Number of periods. **NOTE** This parameter is only required for yearly/monthly/daily subscriptions (**chargingMode** is set to **PERIOD**). Enter a positive integer, for example, **1**, **2**, and **3**.
currency	No	String	64	Order amount. In scenarios such as subscription, renewal, and specification change, the amount is positive. In scenarios such as unsubscription and renewal cancellation, the amount is empty.
currencyAfterDiscount	No	String	25	Order transaction amount, excluding cash coupons and discounts. In scenarios such as subscription, renewal, and specification change, the amount is positive. In scenarios such as unsubscription and renewal cancellation, the amount is negative.
productInfo	Yes	List<ProductInfo>		Product information associated with the order line.

The following table describes the **ProductInfo** data structure.

Parameter	Mandatory	Type	Maximum Length	Description
productId	Yes	String	64	Product ID. The value of **productId** varies between products of different billing modes under the same **skuCode**. For example, when you release a product and add a new specification, an **skuCode** value is generated. After yearly and monthly billing prices are configured, two **productId** values are generated. **NOTE** Log in to the Seller Console. Then, choose **Product Management** > **My Products**. In the row of your product, click **Details** in the **Operation** column. On the product details page, you can view the value of this parameter.
skuCode	Yes	String	64	Specification ID. When renewing the subscription of a yearly/monthly product, a customer can change the billing mode (for example, from monthly to yearly). In this case, **productId** corresponding to **instanceId** of the instance enabled by the customer changes, but the value of **skuCode** does not change. **NOTE** Log in to the Seller Console. Then, choose **Product Management** > **My Products**. In the row of your product, click **Details** in the **Operation** column. On the product details page, you can view the value of this parameter.
linearValue	No	Integer		Linear value that the customer selected when placing the order for a product with the quantity attribute.
productName	Yes	String	64	Product name.

The following table describes the **ExtendParam** data structure.

Table 1-6 Response parameters

Parameter	Mandatory	Type	Maximum Length	Description
name	Yes	String	64	Parameter name.
value	Yes	String	64	Parameter value.

The following table describes the **BuyerInfo** data structure.

Parameter	Mandatory	Type	Maximum Length	Description
customerId	Yes	String	64	Customer ID.
customerName	Yes	String	64	Customer account name, for example, **Sam**.
customerRealName	Yes	String	64	Real customer name, for example, **xxxxx Company**.
customerType	Yes	integer	64	Unknown: **-1**. Individual: **0**. Enterprise: **1**.
mobilePhone	No	String	64	Mobile number of the customer. **NOTE** You can specify whether customers must authorize you to obtain this information when you release a product.
email	No	String	64	Email address of the customer. **NOTE** You can specify whether customers must authorize you to obtain this information when you release a product.
userId	No	String	64	IAM user ID. **NOTE** You can specify whether customers must authorize you to obtain this information when you release a product. This parameter is returned only for new subscriptions (instance creation scenario).

Parameter	Mandatory	Type	Maximum Length	Description
userName	No	String	64	IAM user name. **NOTE** You can specify whether customers must authorize you to obtain this information when you release a product. This parameter is returned only for new subscriptions (instance creation scenario).

Error Codes

Table 1-7 Error codes

HTTP Status Code	resultCode	resultMsg	Description
200	MKT.0000	Success.	Request successful.
500	MKT.0999	System internal error.	Other internal errors.
500	MKT.0100	Failure of input parameter	Input parameter verification failed. Invalid value.
400	MKT.0101	Invalid parameter	Invalid parameter. The parameter is not defined by the API, there are more parameters than required, or a mandatory parameter is missing.
400	MKT.0199	Request parameter error	Incorrect request parameter. Other parameter errors.
401	MKT.0150	Illegal operation	You are trying to perform an unauthorized operation. For example, the product corresponding to **instanceId** is not released by the seller corresponding to the AK/SK.
401	MKT.0151	No authority	Insufficient permissions to access the interface. The token does not belong to a seller.
401	MKT.0154	Illegal token	Authentication failed. Invalid token.

HTTP Status Code	resultCode	resultMsg	Description
406	MKT.0250	Access frequency overlimit	Too many access requests.
500	MKT.9001	Instance ID not found.	The instance ID does not exist. (This result code may be returned when the renewal, expiration, or resource release interface is called.)
500	MKT.9002	Invalid usage entities.	Invalid usage entities.
500	MKT.9003	Usage records extend size limit.	Too many records. Max. records: 100.
500	MKT.9004	Record beginTime extends Limit.	The start time exceeds the validity period (last 21 days).

A failure response contains the **extra_info** parameter only when the value of **resultCode** is **MKT.0100**, **MKT.0150**, **MKT.0250**, **MKT.9001**, **MKT.9002**, **MKT.9004**, or **MKT.9005**. In addition to the parameter description, **resultMsg** in the failure response also contains the failure details. You can locate and rectify the fault based on the failure details and **extra_info** content.

Successful response example

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: length
Date: response time

{
 "resultCode": "MKT.0000",
 "resultMsg": "Success",
 "orderInfo": {
  "orderId": "CS2207261447AUY4H",
  "orderType": "NEW",
  "createTime": "20220726064736",
  "orderLine": [
   {
    "orderLineId": "CS2207261447AUY4H-000001",
    "chargingMode": "PERIOD",
    "periodType": "year",
    "periodNumber": 1,
    "expireTime": "20230726155959",
    "productInfo": [
     {
      "productId": "OFFI758576253042421760",
      "skuCode": "da9b4d34-ee8a-4355-a823-13e034e49986",
      "linearValue": 10,
      "productName": "Test SaaS, Test Specification, Basic Edition, Yearly/Monthly"
     }
    ],
    "extendParams": []
   }
  ],
  "buyerInfo": {
   "mobilePhone": "18699999999",
```



```
    "email": 123@test.com,
    "customerId": "688055390f3049f283fe9f1aa90f7ds3",
    "customerName": "hw1235sd3123"

  }
 }
}
```

Failed response example

```
HTTP/1.1 401 UnauthorizedContent-Type: application/json;charset=UTF-8Content-Length: lengthDate:
response time {
   "resultCode": "CBC.0150",
   "resultMsg": "Illegal operation. param[isvId] and param[instanceId] do not match."  }
```

1.7.3.2 Pushing the Pay-per-Use Resource Usage (New)

Description

After a customer purchases and uses pay-per-use resources in KooGallery, call this API to upload the SDRs of the customer. After obtaining the SDRs, KooGallery charges the customer for the usage.

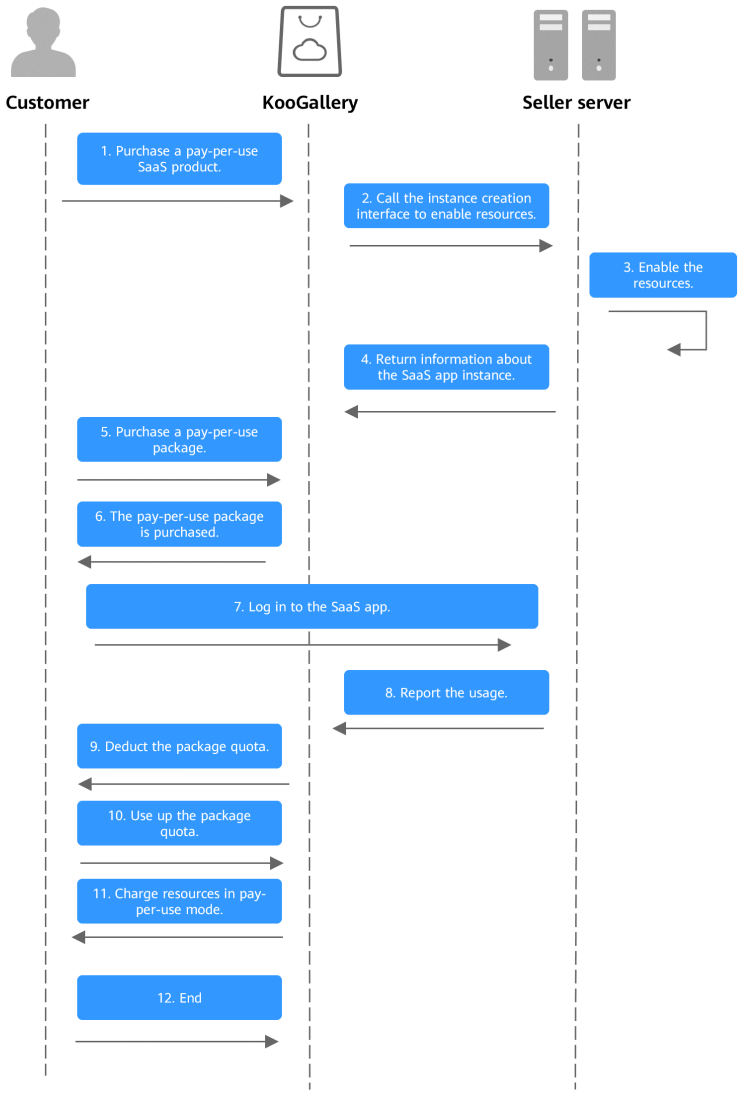
URI

POST https://mkt-intl.myhuaweicloud.com/api/mkp-openapi-public/global/v1/isv/usage-data

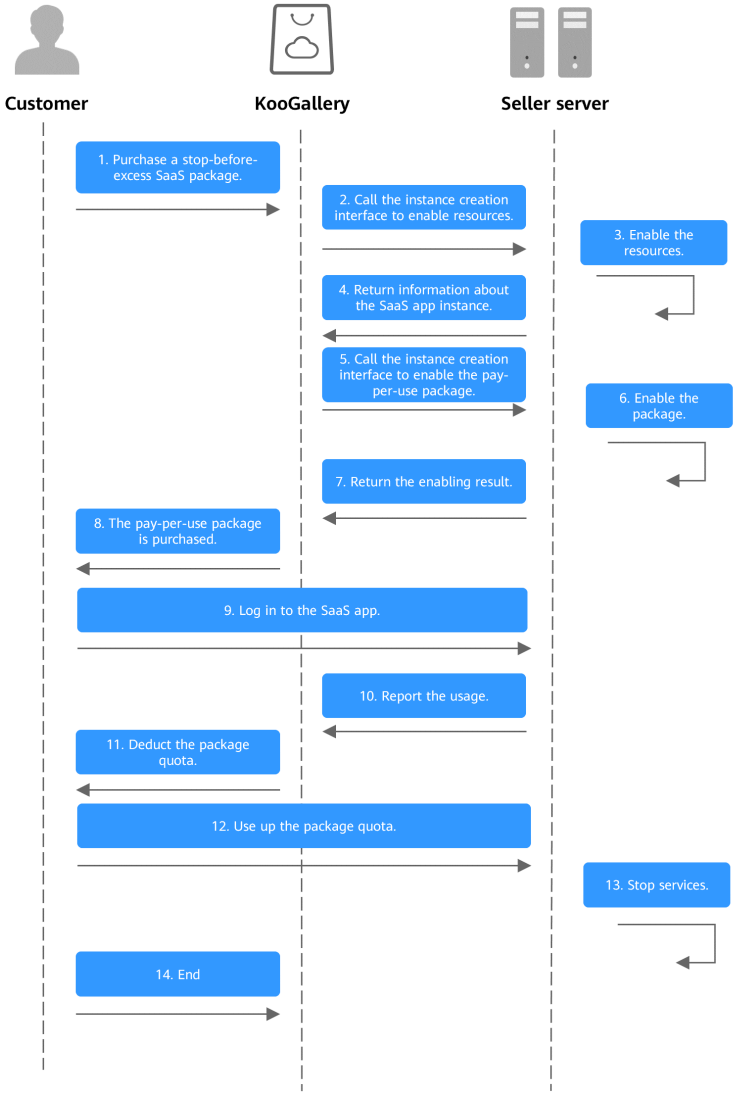
NOTE

If mkt-intl.myhuaweicloud.com is unavailable, try mkt.myhuaweicloud.cn.

1. Pay-per-use specification/package purchase and use



2. Stop-before-excess package purchase and use



Request Message

The following table describes the request parameters.

Request method: POST

Parameter	Mandatory	Type	Maximum Length	Description
signature	Yes	String	1,000	@Header API signature **(base64(hmacSHA256(***Seller interconnection key*,**ts=***{ts}*&**nonce=***{nonce}*&**body=***{body}*))** The body is signed after being sorted naturally.

Parameter	Mandatory	Type	Maximum Length	Description
ts	Yes	String	20	@Header Unix timestamp when an interface request is sent, in milliseconds.
nonce	Yes	String	64	@Header Security random number.
usage_records	Yes	List<UsagePushData>	1,000	SDR list. A list contains up to 1,000 **UsagePushData** records.

Table 1-8 UsagePushData

Parameter	Mandatory	Type	Maximum Length	Description
instance_id	Yes	String	64	Pay-per-use instance ID. Use the instance ID returned by the pay-per-use subscription interface.
record_time	Yes	String	17	Time when a usage record is generated (UTC). Format: yyyyMMdd'T'HHmmss'Z'
begin_time	Yes	String	17	Metering start time (UTC). Format: yyyyMMdd'T'HHmmss'Z'
end_time	Yes	String	17	Metering end time (UTC). Format: yyyyMMdd'T'HHmmss'Z'
usage_value	Yes	Double(12,4)	20	Usage value. The value is a positive number containing up to four significant decimal places.
metering_sn	Yes	String	64	Unique SDR ID. A random code is recommended.
relate_pkg_instance	No	String	64	This parameter is mandatory in SDRs of stop-before-excess packages. The package instance ID needs to be transferred.

Example request:

```
POST {domain}/api/mkp-openapi-public/global/v1/isv/usage-data
Content-Type:application/json
nonce: 6c63c221-1f6b-4141-8ff4-22f5dfe82b65
```

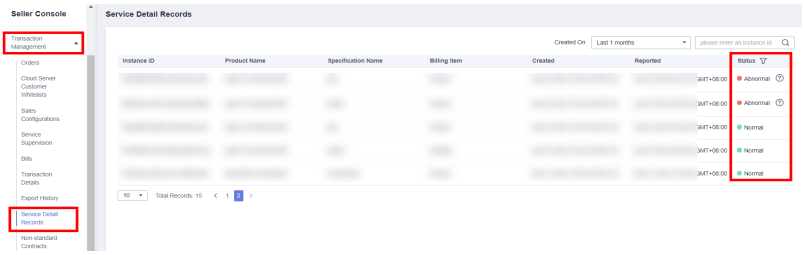


```
ts: 1709690865879
signature: gikLsIgimIscagwSamClFJ1CFT4QprHSDHW...
{
   "usage_records": [
      {
         "instance_id": "7f141bf1-aec8-4859-8323-fb3a8ad50721",
         "record_time": "20220809T091000Z",
         "begin_time": "20220809T080000Z",
         "end_time": "20220809T090000Z",
         "usage_value": "99",
         "metering_sn": "6c75c177b5fe4b8cbb6fc2aa33facfcd"
      },
      {
         "instance_id": "7f141bf1-aec8-4859-8323-fb3a8ad50721",
         "record_time": "20220809T091000Z",
         "begin_time": "20220809T080000Z",
         "end_time": "20220809T090000Z",
         "usage_value": "999",
         "metering_sn": "6c75c177b5fe4b8cbb6fc2aa33facfcb"
      }
   ]
}
```


NOTE

1. During SDR upload, if SDR data is abnormal, no error is reported at the API layer. The backend periodically verifies and processes the uploaded data and generates available SDR data. If the backend fails to process the data, report the data again.

You can view abnormal data on the **Transaction Management** > **Service Detail Records** page of the Seller Console.



2. Requirements for the SDR report period:

- **Hourly billing**

  Report SDRs at least once an hour. It is recommended that SDRs be reported within the first 15 minutes of the next hour after a customer uses the resources. For example, if the customer uses resources at 13:25, report SDRs between 14:00 and 14:15. In this way, the customer can be charged in time. Otherwise, the fee deduction will be delayed. If you cannot report SDRs in real time, report them within 2 hours after resource consumption.

- **Daily billing**

  Report SDRs to KooGallery every hour. If you can only report SDRs once a day, report them from 00:00 to 00:15. SDRs must be reported before 01:00. Otherwise, the fee will be deducted from customers on the next day.

3. Requirements for reporting SDRs:

- **When a resource is not closed:**
- SDR start time (**begin_time**) ≥ Resource start time
- SDR start time (**begin_time**) ≤ SDR end time (**end_time**) ≤ SDR report time
- **When a resource is closed:**
- SDR end time (**end_time**) ≤ Resource close time

4. The time in the reported SDRs is the UTC time.

5. If the values of **begin_time** and **end_time** in a record are the same and the record is reported for multiple times, only the first record is processed. SDRs are collected at 01:00 every day for daily billing and fifteenth minute of every hour for hourly billing. Once SDRs are collected and formal bills are generated, SDRs cannot be corrected.

Duplicate SDRs are regarded as abnormal. You can view abnormal data on the **Transaction Management** > **Service Detail Records** page of the Seller Console.

6. The usage push interface uses the instance ID returned by the pay-per-use subscription interface instead of that returned by the pay-per-use package subscription interface.

7. If multiple SDRs (carrying **usage_value**) of an instance (**instance_id**) are reported during an SDR period, that is, the start time (**begin_time**) and end time (**end_time**) of these SDRs are the same, the SDRs must be identified by different extended parameters (**extend_params**). Or, KooGallery returns a verification failure message.

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
error_code	Yes	String	6	Result code. For details, see **1.6 Result Codes**.
error_msg	No	String	255	Result message.
data	No	AbnormalUsageDataInfo		Information about abnormal SDRs.

Table 1-9 AbnormalUsageDataInfo

Parameter	Mandatory	Type	Maximum Length	Description
abnormal_usage_data	Yes	List<AbnormalUsageData>	1,000	List of abnormal SDRs.

Table 1-10 AbnormalUsageData

Parameter	Mandatory	Type	Maximum Length	Description
metering_sn	Yes	String	64	Unique SDR ID.

Parameter	Mandatory	Type	Maximum Length	Description
error_code	Yes	String	16	SDR-level error code. **001**: The instance does not exist. **002**: Invalid time format. **003**: Abnormal usage. **004**: Missing SDR ID. **005**: Duplicate SDR ID. **006**: The product corresponding to the instance has been removed from the catalog. **007**: The SDR has expired. **009**: The instance does not match the seller. **010**: Duplicate SDR. **011**: Invalid SDR time range. **012**: The instance is not a pay-per-use resource. **013**: The instance resource status is abnormal. **014**: The instance resource has been closed. **015**: The SDR start time is earlier than the resource enabling time. **016**: The instance is being enabled. **017**: In the stop-before-excess scenario, **relate_pkg_instance** is empty. **018**: In the stop-before-excess scenario, **relate_pkg_instance** is invalid or does not match **instance_id**.
error_msg	Yes	String	255	SDR error message.

Error codes

Table 1-11 AbnormalUsageData

HTTP Status Code	Error Code	Error Message	Description
200	MKT.0000	Success	Request successful.

HTTP Status Code	Error Code	Error Message	Description
500	94060001	System error!	Other internal errors.
401	94060002	Auth failed!	Input parameter verification failed. Invalid value.
400	94060004	Param invalid	Invalid parameter. The parameter is not defined by the API, there are more parameters than required, or a mandatory parameter is missing. For example, a value is invalid or there is no instance ID.
400	94060005	Time format error	Incorrect time format.
400	94060006	TimeStamp invalid	Invalid timestamp.
401	94060007	Signature invalid	Signature verification fails.
400	94060008	Replay error	Request replay error.
500	94060009	Failed to report usage data	Report SDR failed.
401	94060010	isv status invalid	Invalid seller status.
200	94060999	Failed	SDR-level error information is returned. For details, see the example response.

If an error code starting with **APIGW** is returned after you call an API, rectify the fault by referring to the instructions provided in **API Gateway Error Codes**.

Example response:

```
{
  "error_code":"MKT.0000",
  "error_msg":"Success"
}

{
   "error_code": "94060999",
   "error_msg": "Failed",
   "data": {
      "abnormal_usage_data": [
         {
            "error_code": "005",
            "error_msg": "METERING_SN_DUPLICATE",
            "metering_sn": "6c75c177b5fe4b8cbb6fc2aa33facfcb"
         }
      ]
```

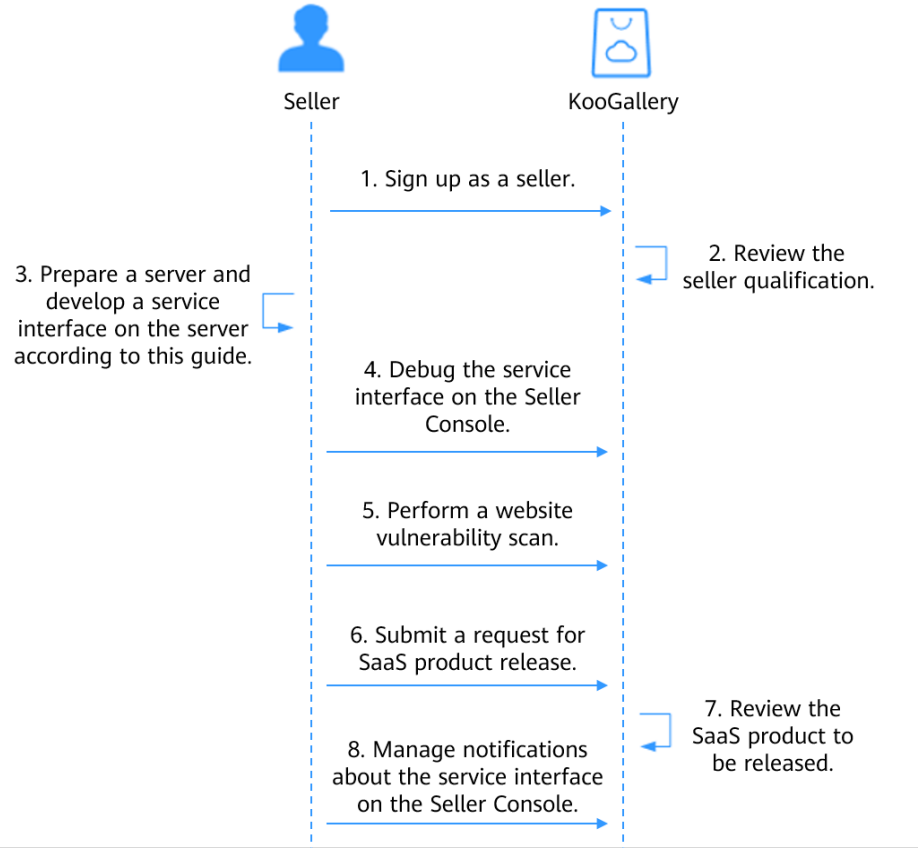


```
  }
}
```

2 SaaS Access Guide V1.0 (Existing Products)

2.1 Access Process

The following figure shows the process of software as a service (SaaS) products accessing KooGallery.



The process is as follows:

1. **Register with KooGallery** and become an independent service vendor (ISV).
2. The KooGallery operations team reviews your company qualification.
3. Prepare a server and develop a service interface on the server based on this guide.
4. Debug the service interface on the **Seller Console**.
5. Perform vulnerability scans on the **Seller Console**.
6. Apply for releasing a SaaS product on the **Seller Console**.
7. The KooGallery operations team reviews the SaaS product. Once approved, the product is released successfully.
8. Manage notifications of the service interface on the **Seller Console**.

2.2 Interface Functions

Before releasing a SaaS product to KooGallery, develop a **service interface** on the ISV server by referring to this access guide.

NOTE

- **Only one service interface for a SaaS product needs to be configured to accommodate different scenarios, including subscription, renewal, expiration, release, and upgrade.**
- If you release a yearly/monthly product, the interface will be called in the subscription, renewal, expiration, and release scenarios.
- If you release a product billed by one-time payment, the interface will be called only in the subscription and release scenarios.
- If you release a pay-per-use product, the interface will be called in the subscription, resource status change, release, and usage push scenarios.
- If the product can be upgraded, the interface will be called in the upgrade scenario.

Functions

- Subscription: After a customer purchases a product and pays for it successfully, KooGallery calls this interface to send you a request containing information about the product and customer. When receiving the request, the ISV server executes product subscription and informs KooGallery about the subscription result.

  NOTE

  When a customer clicks the **View Resource Details** button on the **Purchased Apps** page, KooGallery calls the subscription interface in real time to query the product information. Therefore, the ISV server needs to **perform idempotence processing** when processing requests. KooGallery may resend requests for **a single order**. If receiving a duplicate order, your server needs to return a success response and **the information about the successfully created app instance, rather than create a SaaS instance**.

- Renewal: After a customer places an order for renewal or converts a trial order to a commercial order, KooGallery calls the interface to request you to extend the service. The service interface then updates the expiration date and informs KooGallery about the update result.
- Expiration: When a purchased product expires, KooGallery calls the interface to send you a notification. After receiving an expiration notification, you must freeze the purchased product and inform KooGallery about the freezing result.

  NOTE

  When a purchased product expires, the retention period starts. The retention period varies with the customer tier and can be up to 15 days long. During the retention period, the product is frozen and cannot be used. The customer can continue using the product after renewal. Therefore, you need to set a retention period to no less than 15 days for your SaaS products and retain customer data during the retention period.

- Resource release: If a customer does not renew an expired product in the retention period, or the customer has unsubscribed from the product, KooGallery releases the purchased product and calls the interface to send you a notification. Upon receiving the notification, delete the specified instances and inform KooGallery about the deletion result.
- Upgrade: After a customer places an order for upgrading a purchased product, KooGallery calls the interface to request you to upgrade the product. The ISV server then upgrades the product and informs KooGallery about the upgrade result. The upgrade scenario is optional.

- After a customer purchases a pay-per-use product (or package), when the instance expires, the customer violates regulations, or the customer account is in arrears, KooGallery calls this interface to freeze the instance.

Interface Failure Scenarios and Retry Mechanism

- In subscription and upgrade scenarios, if the service interface fails to respond, KooGallery will retry for 3 hours.

  If the interface exception is rectified, the next call will be successful and the order will be placed successfully. If the exception persists after 3 hours, KooGallery determines that the order fails to be placed and **automatically cancels the order**.

- In the renewal scenario, if the service interface fails to respond, KooGallery will retry for an hour.

  If the interface exception is rectified, the next call will be successful and the order will be placed successfully. If the exception persists after an hour, KooGallery determines that the order fails to be placed. In this case, locate and rectify the exception. **Then go to the Seller Console, locate the order on the Application Tools > Service Interface Messages page, and click Restart Debugging in the same row to call the interface again.**

- In product expiration and resource release scenarios, if the service interface fails to respond, KooGallery will retry for an hour.

  If the interface exception is rectified, the next call will be successful and the order will be placed successfully. If the exception persists after an hour, KooGallery determines that the order fails to be placed. In this case, locate and rectify the exception. **Then go to the Seller Console, locate the order on the Application Tools > Service Interface Messages page, and click Restart Debugging in the same row to call the interface again.**

  If a customer can still use expired resources due to an interface failure, you shall bear the resource loss incurred.

  NOTE

  If the service interface fails to respond, **an email, SMS message, and private message** will be sent to you. Check the email address and mobile number bound to your account and the Message Center on Huawei Cloud.

  If more than five orders failed in a month due to interface failures, KooGallery will remove the product from the catalog.

  If an order is automatically canceled due to an interface failure, contact the customer at the earliest to handle the problem.

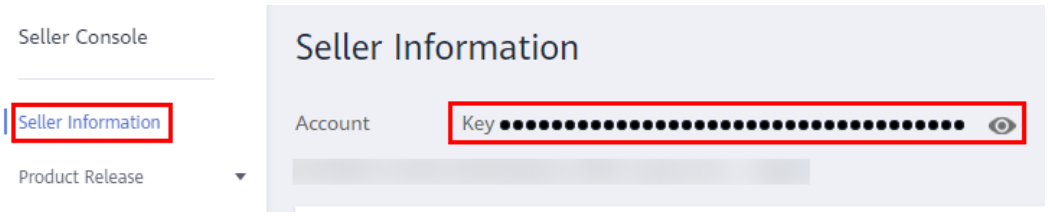
2.3 Preparations

2.3.1 Obtaining the Key

Step 1 Go to the **Seller Console**.

Step 2 In the navigation pane, choose **Seller Info**.

On the **Seller Info** page, click the eye icon to obtain the key.



----End

2.3.2 authToken Value

Definition

The **authToken** parameter is mandatory for verifying the communication security between KooGallery and a seller. It is included in the parameters that KooGallery uses to invoke an interface of a seller.

The seller generates an **authToken** value by following the defined procedure and compares it with the **authToken** value obtained from KooGallery through the interface. If they are identical, the communication security passes the verification.

Generation Procedure

1. Obtain **all** the HTTP GET request parameters excluding the **authToken**.
2. Sort the parameter names in alphabetical order.
3. Use HMAC-SHA256 and the **Key** to encrypt the entire string of the sorted parameter names. The encryption result is adopted as the **authToken** value.

Example

A seller receives an invocation request similar to the following:

```
http://www.isvwebsite.com/saasproduce?
p1=1&p2=2&p3=3&authToken=xxxxxxxxxxxxx&timeStamp=201706211855321
```

1. Obtain all the HTTP GET request parameters p1, p2, p3, and timeStamp.
2. Sort the parameter names in alphabetical order: sort(p1, p2, p3, timeStamp). Assume that the sequence obtained by sorting is p1, p3, p2, and timestamp.
3. Generate an authToken value by encryption: base64_encode(HMAC_SHA256(Key+timeStamp, p1=1&p3=3&p2=2&timeStamp=201706211855321)).

NOTE

All parameter values are URL-encoded in KooGallery. After obtaining the parameter value, the seller needs to decode them.

Example Code

For a code example, see **2.7.1 ISV Server Verifying Requests**.

2.3.3 HTTP Body Signature

A body signature must be contained in the response of each interface. It consists of **sign_type** and **signature**.

Parameter	Value	Description
sign_type	HMAC-SHA256	Current value: HMAC-SHA-256
signature	base64_encode(HMAC_SHA256(key, httpBody))	base64_encode(HMAC_SHA256(key, httpBody)) • **key**: Key value • **httpBody**: The entire HTTP body, including the starting and ending spaces and tab characters

Example of an HTTP response header:

```
Body-Sign: sign_type="HMAC-SHA256", signature= "abcd4567ed03sdfsdfsdfasdfasdgsdfhfgjgkghjllhjkl"
```

NOTE

The format of the header must follow the example. Quotation marks must be added to the values of the **sign_type** and **signature** parameters.

For a code example, see **2.7.2 ISV Server Signing a Response Message Body**.

2.4 Interface Description

2.4.1 Subscription

Description

- After a customer purchases a product and pays for it successfully, KooGallery calls this interface to send you a request containing information about the product and customer. When receiving the request, the ISV server executes product subscription and informs KooGallery about the subscription result.
- A unique instance ID (**instanceId**) should be returned for the order. The instance IDs (**instanceId**) of different subscription orders must be different. Use **businessId** provided by KooGallery to ensure that **instanceId** is globally unique.

  NOTE

  Pay-per-use products are traded by specification. If a specification in an order has multiple pay-per-use billing items, you need to create instances for these billing items separately based on **orderId** (the same order ID) and **productId** (different product IDs). That is, in pay-per-use transactions, an order has multiple instances.
- If the interface fails to respond, KooGallery will notify you by sending an email to the email address bound to your Huawei Cloud account. The interface exception information will be displayed on the **Transaction Management** > **Service Interface Messages** page. Rectify the exception as soon as possible to avoid order cancellation.

  If the subscription interface fails to be called, KooGallery will retry for 3 hours. If the interface exception is rectified, the next call will be successful and the

order will be placed successfully. If the exception persists after 3 hours, KooGallery determines that the order fails to be placed and **automatically cancels the order**. If more than five orders failed in a month due to interface failures, KooGallery will remove the product from the catalog.

NOTE

- Check the email address bound to your Huawei Cloud account and the Message Center. If you receive an email or message about an interface calling failure, rectify the exception as soon as possible.
- KooGallery monitors interface exceptions. If subscriptions to a SaaS product frequently fail due to interface exceptions, KooGallery will remove the product from the catalog.

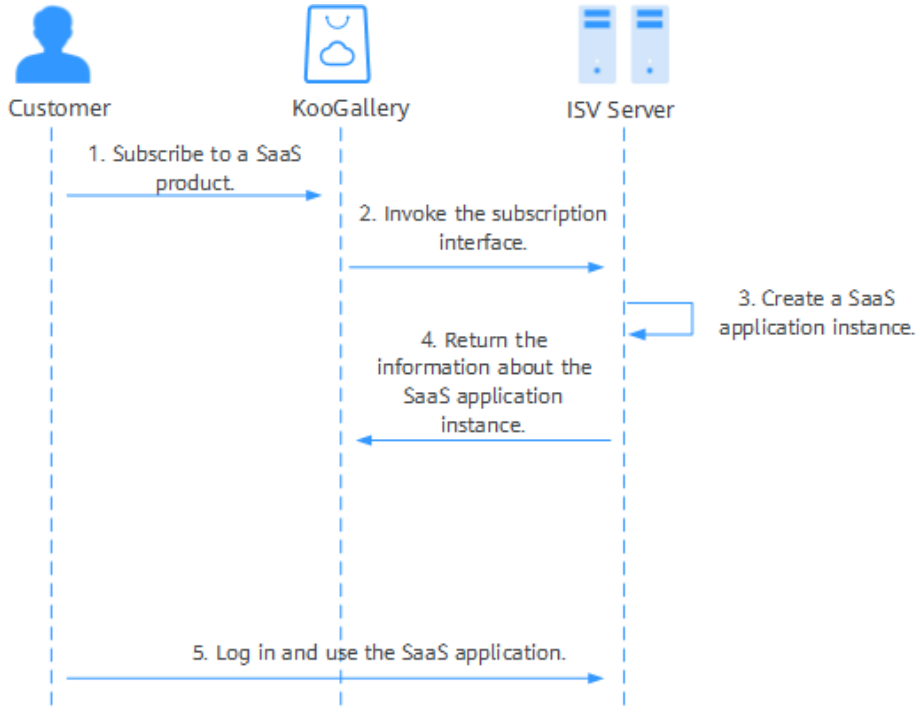
- **When processing an interface request, your server must ensure idempotency.**

  KooGallery may resend requests for **a single order**. If receiving a duplicate order, your server needs to return a success response and **the information about the successfully created app instance, rather than create a SaaS instance**.

  NOTE

  In pay-per-use transactions, ensure idempotency based on the order ID (**orderId**) and product ID (**productId**).

The following figure shows the process of purchasing a product.



Request Message

The following table describes the request parameters. In KooGallery, requests are generated based on the subscription mode of products released by you. You need to provide services based on requests.

Request method: GET

Parameter	Mandatory	Type	Maximum Length	Description
authToken	Yes	String	50	Security verification token. For details about the values, see **2.3.2 authToken Value**.
timeStamp	Yes	String	20	UTC timestamp when a request is initiated. Format: yyyyMMddHHmmssSSS
activity	Yes	String	20	Request ID, which is used to distinguish the scenario. For new subscriptions, the value is **newInstance**.
customerId	Yes	String	100	Unique ID of a customer on Huawei Cloud.
customerName	No	String	64	Customer's username on Huawei Cloud.
userId	No	String	64	Unique ID mapping the username used to log in to the system as an Identity and Access Management (IAM) user. Optional. **If this parameter is required, select To create an account based on IAM username for User Authorization Required when releasing the product.**
userName	No	String	64	Username of the customer used to log in to the system as an IAM user. Optional. **If this parameter is required, select To create an account based on IAM username for User Authorization Required when releasing the product.**

Parameter	Mandatory	Type	Maximum Length	Description
mobilePhone	No	String	256	Customer's mobile number. Optional. **If this parameter is required, select To create an account based on phone number for User Authorization Required when releasing the product. The value is an encrypted mobile number.** The mobile number encryption rules are as follows: The value consists of a 16-bit encryption initialization vector (IV) and a Base-encoded mobile number ciphertext. • iv +base64(AES_CBC(*accessKey*,*mobilePhone*)) • The number of digits to be encrypted is specified by you when you release the product. For details about the example code of mobile number decryption, see **2.7.4 ISV Server Decrypting the Mobile Number and Email Address**. **NOTE** This parameter does not contain the country code. If a customer does not bind the mobile number, the parameter cannot be obtained.

Parameter	Mandatory	Type	Maximum Length	Description
email	No	String	256	Customer's email address. Optional. **If this parameter is required, select To create an account based on email address for User Authorization Required when releasing the product. The value is an encrypted email address.** The email address encryption rules are as follows: The value consists of a 16-bit encryption IV and a Base-encoded email ciphertext. • iv +base64(AES_CBC(*accessKey*,*email*)) • The number of digits to be encrypted is specified by you when you release the product. For details about the example code of email address decryption, see **2.7.4 ISV Server Decrypting the Mobile Number and Email Address**.
businessId	Yes	String	64	KooGallery business ID. The value of **businessId** is different for each request.
orderId	Yes	String	64	KooGallery order ID.
skuCode	No	String	64	Specification ID. When renewing the subscription of a yearly/monthly product, a customer can change the billing mode (for example, from monthly to yearly). In this case, **productId** corresponding to **instanceId** of the instance enabled by the customer changes, but the value of **skuCode** does not change. **NOTE** After a product is approved and successfully released to KooGallery, you can obtain this parameter in the Seller Console. On the **Product Management** > **My Products** page, locate the product and click **Details** in the **Operation** column. The parameter can be obtained on the displayed page.

Parameter	Mandatory	Type	Maximum Length	Description
productId	Yes	String	64	Product ID. The value of **productId** varies between products of different billing modes under the same **skuCode**. For example, when you release a product and add a new specification, an **skuCode** value is generated. After yearly and monthly billing prices are configured, two **productId** values are generated. **NOTE** After a product is approved and successfully released to KooGallery, you can obtain this parameter in the Seller Console. On the **Product Management** > **My Products** page, locate the product and click **Details** in the **Operation** column. The parameter can be obtained on the displayed page.
testFlag	No	String	2	Whether a request is submitted for debugging. • **1**: debugging request. • **0**: non-debugging request. The default value is **0**.
trialFlag	No	String	2	Whether an instance is created for a trial. • **0**: no. • **1**: yes. • **N/A**

Parameter	Mandatory	Type	Maximum Length	Description
expireTime	No	DateTime	20	Expiration time. Format: yyyyMMddHHmmss **NOTE** • This parameter is required for a yearly/monthly/daily product. • This parameter is required for a pay-per-use package. • This parameter is not required for a pay-per-use specification. • This parameter is not required for a product billed by uses. • The expiration time is calculated based on the order creation time and product subscription duration. It may be different from the actual expiration time of the order in the request information and is for reference only. Do not use it for other purposes.
chargingMode	No	Integer	[3]	Billing mode. **0**: pay-per-use. **1**: yearly/monthly/daily. **3**: one-time payment. **5**: pay-per-use package.
saasExtendParams	No	String	2,048	Extension parameters. These parameters are optional. The extension parameters are a JSON string carried in the **url** parameter in the form of **urlEncode(base64(saasExtendParams))**. After obtaining the value of the **saasExtendParams** parameter, your server needs to use **base64Decode(urlDecode(saasExtendParams))** to obtain the JSON string of the extension parameters. For example, **emailDomainName** and **extendParamName** in the JSON string **[{"name":"emailDomainName","value":"test.xxxx.com"}, {"name":"extendParamName","value":"extendParamValue"}]** are the parameter values set during product release.

Parameter	Mandatory	Type	Maximum Length	Description
amount	No	Integer	4	Product attribute of the quantity type. This parameter is optional. Attribute name: quantity (customizable) Unit: none **NOTE** When customers subscribe to multi-SKU SaaS products (billing mode: yearly/monthly or one-time) with specifications that contain the quantity type attribute, they specify or modify the number or usage times. Example: 30 users
diskSize	No	Integer	4	Product attribute of the quantity type. Optional. Attribute name: disk size (customizable) Unit: GB **NOTE** When customers subscribe to multi-SKU SaaS products (billing mode: yearly/monthly or one-time) with specifications that contain the disk size attribute, they specify or modify the disk size. Example: 100 GB
bandWidth	No	Integer	4	Product attribute of the quantity type. Optional. Attribute name: bandwidth (customizable) Unit: Mbit/s **NOTE** When customers subscribe to multi-SKU SaaS products (billing mode: yearly/monthly or one-time) with specifications that contain the bandwidth attribute, they specify or modify the amount of bandwidth. Example: 20 Mbit/s

Parameter	Mandatory	Type	Maximum Length	Description
periodType	No	String	10	Period type. **NOTE** This parameter is only required for yearly/monthly/daily product or pay-per-use package subscriptions (the value of **chargingMode** is set to **1** or **5**). Yearly subscription: **year** Monthly subscription: **month** Daily subscription: **day** If **chargingMode** is set to **3** or **0**, do not pass this parameter.
periodNumber	No	integer	5	Number of periods. **NOTE** This parameter is only required for yearly/monthly/daily product or pay-per-use package subscriptions (the value of **chargingMode** is set to **1** or **5**). Enter a positive integer, for example, **1**, **2**, and **3**.
orderAmount	No	bigdecimal	20	Order amount. **NOTE** This parameter is required only for common product subscriptions. The amount is the actual payment amount, which you can check during reconciliation. The amount is greater than or equal to **0** and can contain a maximum of three decimal places. Unit: USD

Parameter	Mandatory	Type	Maximum Length	Description
provisionType	No	integer	2	Instance provisioning mode. **NOTE** • **Provision upon subscription** (By default, KooGallery calls the **newInstance** interface in polling mode.) • **Provision after acceptance** (The SaaS product involves service supervision.) 1. When a customer purchases the product, KooGallery calls the subscription interface, and you need to return the result code **000004** (request being processed) or **000000** (order created successfully). 2. After you confirm to deliver the product, KooGallery calls the subscription interface, and you need to return the result code **000000**. 3. When the customer accepts the product, KooGallery calls the subscription interface and transfers the acceptance time to you. In this case, return the result code **000000**.
acceptanceTime	No	String	20	Acceptance time. **NOTE** The value is the time when billing for the product starts. If **provisionType** is set to **Provision after acceptance**, this parameter is required. Format: yyyyMMddHHmmssSSS
startTime	No	String	20	Start time. Format: yyyyMMddHHmmss **NOTE** This parameter is transferred only for pay-per-use packages.

NOTE

- On May 12, 2018, interface parameters **trialFlag** and **skuCode** were added.
  - Set these parameters for products released or product specifications added after May 12, 2018. All three values of **trialFlag** must be successfully debugged.
  - If a product was successfully released before May 12, 2018 and does not involve the free trial, the interface debugging is not required.
- On August 9, 2018, the interface for releasing SaaS products whose billing mode is **one-time** was added. If this billing mode is selected for a product release, the interface must be successfully debugged based on the *SaaS Product Access Guide*.
- On September 27, 2019, interface parameters **amount**, **diskSize**, and **bandWidth** were added for attributes of the quantity type.

  If product specifications that are priced using a custom template contain attributes of the quantity type, such as number, bandwidth, and disk size, create the attributes on the product attribute management page, and navigate to the **Application Access Debugging** page to set related parameters and debug the interfaces. After the debugging is successful, you can release the product specifications.
- For details, see **2.6 Interface Debugging**.

Example request:

```
https://isvserver.com/produceAPI?
activity=newInstance&businessId=03pf80c2bae96vc49b80b917bea776d7&customerId=3736bb8ad93b43fca80
12c64a82cec25 &expireTime=20180725000000&orderId=HWS001014ED483AA1E8&productId=
005a8781ef0c4a47a3dbfc4c1e72871e&saasExtendParams=W3sibmFtZSI6ImVtYWlsMTEiLCJ2YWx1ZSI6ImVtY
WlsMTFlbWFpbDExIn0seyJuYW1lIjoiZW1haWwyMiIsInZhbHVlIjoiZW1haWwyMmVtYWlsMjIifV0%3D&timeSt
amp=20170725025113409&testFlag=0&authToken=09lsS5y+KCtxBu+ON4TXv1SrjH5KVYka9sx2MauHrQU=
```

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	6	Result code. For details, see **2.5 Invocation Result Codes**.
resultMsg	No	String	255	Result message.
encryptType	No	String	2	Algorithm for encrypting sensitive information. **1**: AES256_CBC_PKCS5Padding (default) **2**: AES128_CBC_PKCS5Padding **NOTE** If the value of this parameter is **AES256_CBC_PKCS5Padding**, **1** is returned; if the value is **AES128_CBC_PKCS5Padding**, **2** is returned.

Parameter	Mandatory	Type	Maximum Length	Description
instanceId	No	String	64	Instance ID, which is a unique ID provided by a seller. Use **businessId** provided by KooGallery to ensure that **instanceId** is globally unique. **NOTE** The value of **businessId** in each request sent by KooGallery is different. If you use **businessId** as **instanceId**, use **businessId** in the first request sent by KooGallery. If **instanceId** is generated in other ways, for example, using a universally unique identifier (UUID), ensure that it is globally unique. Identical values of **instanceId** will cause a failure of enabling a SaaS app instance.
appInfo	No	AppInfo	N/A	App instance information. After a customer purchases a product, return a login address (website address) or an address that does not require login for the customer to perform subsequent operations. **NOTE** You must provide customers who purchase your SaaS products with the app usage information, including the addresses, accounts, and passwords. If the usage information can be sent through SMS messages, emails, or other methods, this parameter is not required in the response. Otherwise, the app instance information must be returned in the response. You can use the **memo** parameter to specify usage instructions or other information if any. **appInfo** is a JSON string. For details about its data structure, see the following table.

The following table describes the **AppInfo** data structure.

Parameter	Mandatory	Type	Maximum Length	Description
frontEndUrl	Yes	String	512	Frontend URL. URL of the website that the customer can access to use the purchased product.
adminUrl	No	String	512	Management URL. URL of the backend website that the customer can access to manage the purchased product.
userName	No	String	128	Encrypted administrator account. Account (usually an email address or mobile number) used by a customer to access the management backend of the seller after purchasing a product. The value consists of a 16-bit encryption IV and a Base-encoded username ciphertext. • iv +base64(AES_CBC(*accessKey*, *userName*)) • The account is encrypted using the key and encryption algorithm specified by **encryptType**. For details about the example code, see **2.7.3 ISV Server Encrypting the Username and Password After Resource Enabling**.

Parameter	Mandatory	Type	Maximum Length	Description
password	No	String	128	Encrypted initial password of the administrator. Password (usually generated by you) used by a customer to access the management backend. The value consists of a 16-bit encryption IV and a Base-encoded password ciphertext. • iv +base64(AES_CBC(*accessKey*, *pwd*)) • The password is encrypted using the key and encryption algorithm specified by **encryptType**. For details about the example code, see **2.7.3 ISV Server Encrypting the Username and Password After Resource Enabling**.
ip	No	String	64	IP address of the website.
memo	No	String	1,024	Remarks.

NOTE

- For details about how to obtain **accessKey**, see **2.3.1 Obtaining the Key**.
- The lengths of the username and password ciphertexts are verified, which include the IVs.
- When processing an interface request, your server must ensure idempotency.

  KooGallery may resend requests for a single order. If receiving a duplicate order, your server needs to return a success response and the information about the successfully created app instance, rather than create a SaaS instance.
- If a SaaS instance information (for example, the **adminUrl**) changes, call the interface again in KooGallery. When the same **orderId** is provided by KooGallery, your server returns information about the updated SaaS instance information.

  For security purposes, KooGallery does not store SaaS instance information for a long time.

  In the ISV production interface response messages, only the value of the **memo** parameter can include Chinese characters.

Example response:

```
{
    "resultCode":"000000",
    "resultMsg":"success.",
    "instanceId":"03pf80c2bae96vc49b80b917bea776d7",
    "encryptType":"1",
    "appInfo":{
```



```
    "frontEndUrl":"http://www.isvserver.com",
    "adminUrl":"http://www. isvserver.com",
    "userName":"luQg154bx766030TobyT0ghfQRx3tvVEdpwMRg==",
    "password":"7Bx4DyX7980a59T0qbhnpfhCz82Uc5cZQQtExg==",
    "memo":"Test"
  }
}
```

2.4.2 Renewal

Description

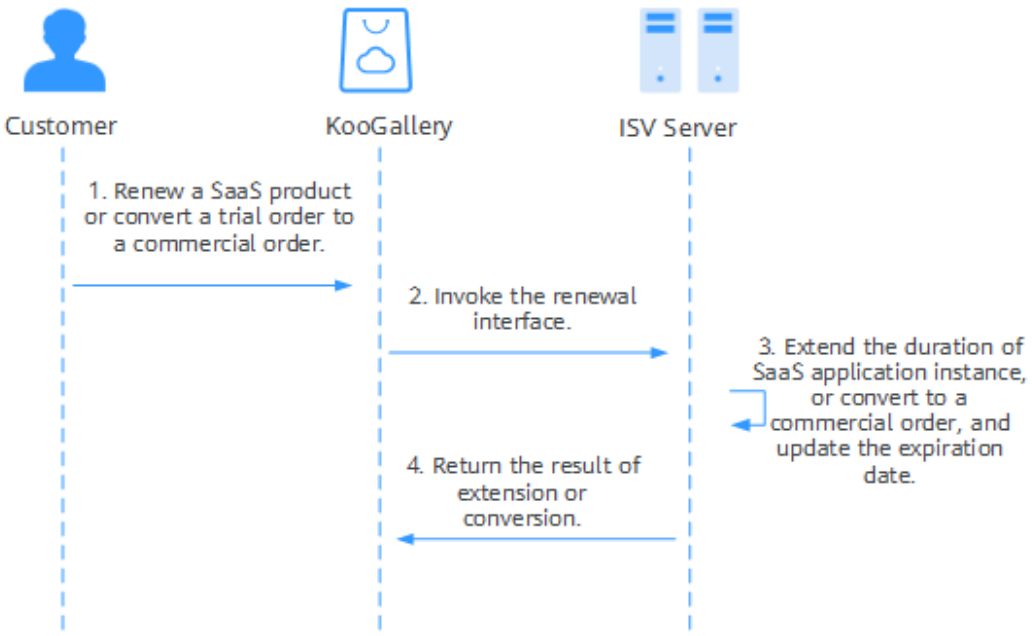
For yearly/monthly products, you must develop the renewal interface.

- After a customer places an order for renewal or converts a trial order to a commercial order, KooGallery calls the interface to request you to extend the service. The service interface then updates the expiration date and informs KooGallery about the update result.
- Ensure that the communication over the interface is normal. If the renewal fails, the service of the user may be terminated.
- If the renewal interface fails to be called, KooGallery will retry for an hour. You can view the interface exception information on the **Application Tools > Service Interface Messages** page. After the exception is solved, ask KooGallery to call the interface again.

  NOTE

  - Check the email address bound to your Huawei Cloud account. If you receive an email about an interface calling failure, rectify the exception as soon as possible.
  - KooGallery monitors interface exceptions. If renewals of a SaaS product frequently fail due to interface exceptions, KooGallery will remove the product from the catalog.

The following figure shows the process of renewing a service.



Request Message

The following table describes the request parameters.

Request method: GET

Parameter	Mandatory	Type	Maximum Length	Description
activity	Yes	String	20	Request ID, which is used to distinguish the scenario. For renewals, the value is **refreshInstance**.
orderId	Yes	String	64	KooGallery order ID. **NOTE** A new order will be generated during the renewal or renewal cancellation and has an ID different from that of a subscription order. Use **instanceId** to identify the resources.
instanceId	Yes	String	64	Instance ID.
productId	No	String	64	Product ID. If a customer renews a product and changes the billing cycle or a customer converts a trial product to a commercial product, a new **productId** is provided.
expireTime	Yes	String	20	Expiration time. Format: yyyyMMddHHmmss
testFlag	No	String	2	Whether a request is submitted for debugging. ● **1**: debugging request. ● **0**: non-debugging request. The default value is **0**.

Parameter	Mandatory	Type	Maximum Length	Description
trialToFormal	No	String	2	Whether a request is submitted to convert a trial product to a commercial product. • Parameter not passed: no • **1**: yes By default, a request is not submitted to convert a trial product to a commercial product. For a request submitted to convert a trial product to a commercial product, it is regarded by default that the instance is not billed in the pay-per-use mode.
authToken	Yes	String	50	Security verification token. For details about the values, see **2.3.2 authToken Value**.
timeStamp	Yes	String	20	UTC timestamp when a request is initiated. Format: yyyyMMddHHmmssSSS
periodType	No	String	10	Period type. **NOTE** This parameter is only required for yearly/monthly product subscriptions (the value of **chargingMode** is set to **1**). Yearly subscription: **year** Monthly subscription: **month**

Parameter	Mandatory	Type	Maximum Length	Description
periodNumber	No	integer	2	Number of periods. **NOTE** This parameter is only required for yearly/monthly product subscriptions (the value of **chargingMode** is set to **1**). Enter a positive integer, for example, **1**, **2**, and **3**.
orderAmount	No	bigdecimal	20	Order amount. **NOTE** The amount is the actual payment amount, which you can check during reconciliation. 1. Renewal: The amount is 0 or a positive number containing up to three decimal places. 2. Renewal cancellation: The amount is a negative number containing up to three decimal places. Unit: USD

Example request:

```
https://isvserver.com/produceAPI?activity=refreshInstance&
expireTime=20180725000000&instanceId=03pf80c2bae96vc49b80b917bea776d7&orderId=HWS001014ED48
3AA1E8&timeStamp=20170725025113409&testFlag=0&authToken=09lsS5y+KCtxBu
+ON4TXv1SrjH5KVYka9sx2MauHrQU=
```

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	6	Result code. For details, see **2.5 Invocation Result Codes**.
resultMsg	No	String	255	Result message.

NOTE

- When processing an interface request, your server must ensure idempotency.
- KooGallery may resend requests for a single order. When receiving a duplicate order, the ISV server needs to return a success response, rather than extend the SaaS instance again.

Example response:

```
{
    "resultCode":"000000",
    "resultMsg":"success."
}
```

2.4.3 Expiration

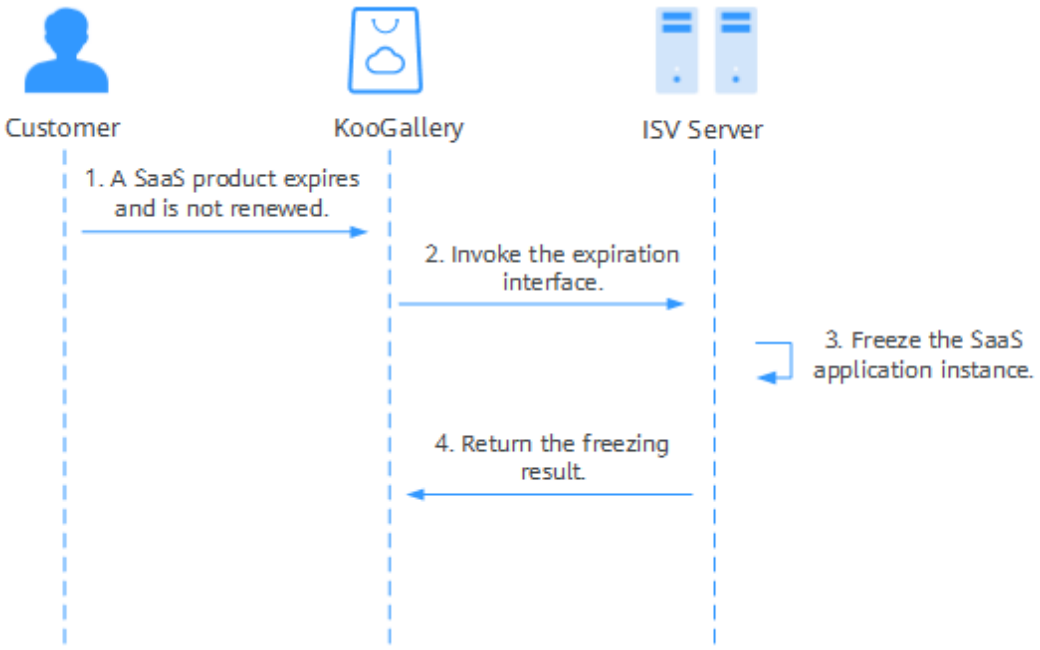
Description

- KooGallery invokes this interface when a purchased product expires. After receiving an expiration notification, you must freeze the purchased product.
- If the expiration interface fails to be called, KooGallery will retry for an hour. You can view the interface exception information on the **Application Tools > Service Interface Messages** page. If the interface exception is rectified, the next call will be successful. If the exception persists after an hour, KooGallery stops calling the interface. In this case, rectify the exception. Then go to the Seller Console, locate the order on the **Application Tools > Service Interface Messages** page, and click **Restart Debugging** in the same row to call the interface again.

  NOTE

  - Check the email address bound to your Huawei Cloud account. If you receive an email about an interface calling failure, rectify the exception as soon as possible.
  - KooGallery monitors interface exceptions. If freezing a SaaS product frequently fails due to interface exceptions, KooGallery will remove the product from the catalog.

The following figure shows the process of expiration.



Request Message

The following table describes the request parameters.

Request method: GET

Parameter	Mandatory	Type	Maximum Length	Description
activity	Yes	String	20	Interface request ID, which is used to distinguish interface request scenarios. For product expiration, the value is **expireInstance**.
instanceId	Yes	String	64	Instance ID.
orderId	Yes	String	64	Same as the ID of the subscription order.
testFlag	No	String	2	Whether a request is submitted for debugging. • **1**: debugging request. • **0**: non-debugging request. The default value is **0**.

Parameter	Mandatory	Type	Maximum Length	Description
authToken	Yes	String	50	Security verification token. For details about the values, see **2.3.2 authToken Value**.
timeStamp	Yes	String	20	UTC timestamp when a request is initiated. Format: yyyyMMddHHmmssSSS

Example request:

```
https://isvserver.com/produceAPI?activity=expireInstance&instanceId=03pf80c2bae96vc49b80b917bea776d7
&timeStamp=20170725025113409&testFlag=0&authToken=09lsS5y+KCtxBu
+ON4TXv1SrjH5KVYka9sx2MauHrQU=
```

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	6	Result code. For details, see **2.5 Invocation Result Codes**.
resultMsg	No	String	255	Result message.

NOTE

- When processing an interface request, your server must ensure idempotency.
- KooGallery may resend requests for a single order. When receiving a duplicate order with the same **instanceId** value, your server needs to return a success response, rather than freeze the instance again.

Example response:

```
{
   "resultCode":"000000",
   "resultMsg":"success."
}
```


2.4.4 Resource Release

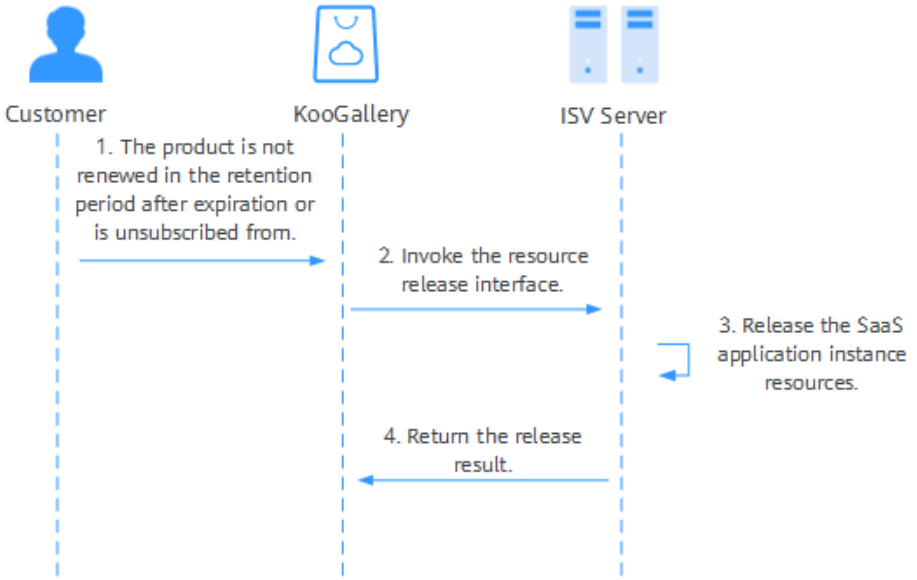
Description

- KooGallery calls this interface to request you to delete a purchased product and sends you a notification. You must delete the instance of the purchased product after receiving the product deletion notification.
- If a customer does not renew an expired product in the retention period, or the customer has unsubscribed from the product, KooGallery releases the purchased product resources.
- If the resource release interface fails to be called, KooGallery will retry for an hour. You can view the interface exception information on the **Application Tools > Service Interface Messages** page. If the interface exception is rectified, the next call will be successful. If the exception persists after an hour, KooGallery stops calling the interface. In this case, rectify the exception. Then go to the Seller Console, locate the order on the **Application Tools > Service Interface Messages** page, and click **Restart Debugging** in the same row to call the interface again.

  NOTE

  - Check the email address bound to your Huawei Cloud account. If you receive an email about an interface calling failure, rectify the exception as soon as possible.
  - KooGallery monitors interface exceptions. If releasing resources of a SaaS product frequently fails due to interface exceptions, KooGallery will remove the product from the catalog.

The following figure shows the process of releasing resources.



Request Message

The following table describes the request parameters.

Request method: GET

Parameter	Mandatory	Type	Maximum Length	Description
activity	Yes	String	20	Interface request ID, which is used to distinguish interface request scenarios. For resource release, the value is **releaseInstance**.
instanceId	Yes	String	64	Instance ID.
orderId	Yes	String	64	Same as the ID of the subscription order.
testFlag	No	String	2	Whether a request is submitted for debugging. • **1**: debugging request. • **0**: non-debugging request. The default value is **0**.
authToken	Yes	String	50	Security verification token. For details about the values, see **2.3.2 authToken Value**.
timeStamp	Yes	String	20	UTC timestamp when a request is initiated. Format: yyyyMMddHHmmssSSS
orderAmount	No	bigdecimal	20	Order amount. **NOTE** The amount is the actual payment amount, which you can check during reconciliation. The amount is greater than or equal to **0** and can contain a maximum of three decimal places. Unit: USD

Example request:

```
https://isvserver.com/produceAPI?
activity=releaseInstance&instanceId=03pf80c2bae96vc49b80b917bea776d7
&timeStamp=20170725025113409&testFlag=0&authToken=09lsS5y+KCtxBu
+ON4TXv1SrjH5KVYka9sx2MauHrQU=
```


Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	6	Result code. For details, see **2.5 Invocation Result Codes**.
resultMsg	No	String	255	Result message.

NOTE

- When processing an interface request, your server must ensure idempotency.
- KooGallery may resend requests for a single order. When receiving a duplicate order with the same **instanceId** value, your server needs to return a success response, rather than release the instance again.

Example response:

```
{
    "resultCode":"000000",
    "resultMsg":"success."
}
```

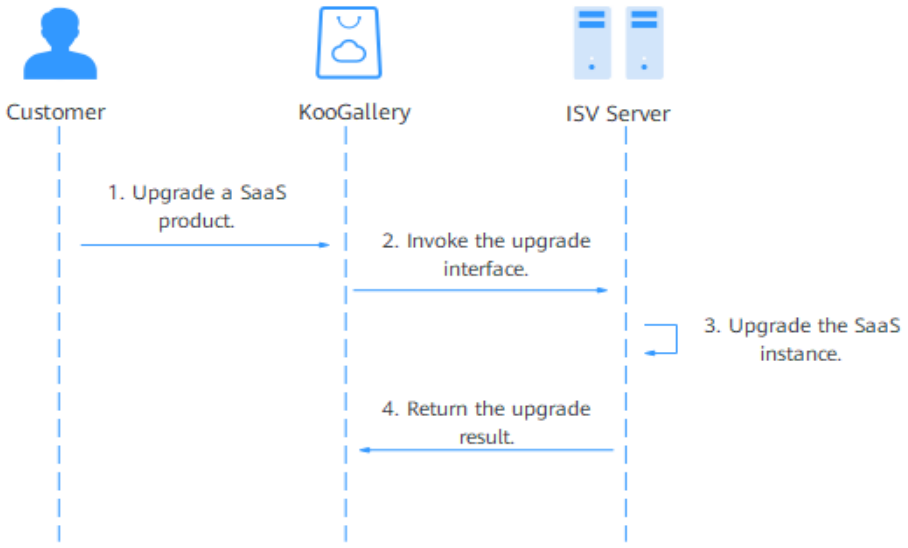
2.4.5 Upgrade

Description

After a customer has successfully paid for an order for upgrading a purchased product, KooGallery calls this interface to request you to upgrade the product. The ISV server needs to upgrade the product and return a notification to KooGallery.

For details about the upgrade rules, see **Upgrade and Billing Rules**.

The following figure shows the process of upgrading a product.



Request Message

The following table describes the request parameters.

Request method: GET

Parameter	Mandatory	Type	Maximum Length of Characters	Description
authToken	Yes	String	50	Security verification token. For details about the value, see **2.3.2 authToken Value**.
activity	Yes	String	20	Interface request ID, which is used to distinguish interface request scenarios. For upgrades, the value is **upgrade**.
instanceId	Yes	String	64	Instance ID. **NOTE** The upgrade does not change **instance ID**.
testFlag	No	String	2	Whether a request is submitted for debugging. • **1**: debugging request. • **0**: non-debugging request. The default value is **0**.
orderId	Yes	String	64	Upgrade order ID. **NOTE** A new order will be generated during the upgrade and has an ID different from that of a subscription order. Use **instanceId** to identify the resources.

Parameter	Mandatory	Type	Maximum Length of Characters	Description
skuCode	Yes	String	64	Product specification ID after the upgrade. **NOTE** A specification with custom attributes will change if the customer selects other attribute values during the upgrade. As a result, the **skuCode** changes. If the customer only expands the capacity by linearly increasing the attribute value, for example, from 10 users to 20 users, the **skuCode** does not change.
productId	Yes	String	64	Product ID after the upgrade. The value of **productId** varies according to the **skuCode**. If the customer only expands the capacity, the value of **productId** does not change.
timeStamp	Yes	String	20	Time (UTC time) when a request is initiated. Format: yyyyMMddHHmmssSSS
amount	No	Integer	4	Product attribute of the quantity type. This parameter is optional. Attribute name: quantity (customizable) Unit: none **NOTE** When customers subscribe to SaaS products (billing mode: yearly/monthly or one-time) with specifications that contain the quantity type attribute, they specify or modify the number or usage times. Example: 30 users

Parameter	Mandatory	Type	Maximum Length of Characters	Description
diskSize	No	Integer	4	Product attribute of the quantity type. This parameter is optional. Attribute name: disk size (customizable) Unit: GB **NOTE** When customers subscribe to SaaS products (billing mode: yearly/monthly or one-time) with specifications that contain the disk size attribute, they specify or modify the disk size. Example: 100 GB
bandWidth	No	Integer	4	Product attribute of the quantity type. This parameter is optional. Attribute name: bandwidth (customizable) Unit: Mbit/s **NOTE** When customers subscribe to SaaS products (billing mode: yearly/monthly or one-time) with specifications that contain the bandwidth attribute, they specify or modify the amount of bandwidth. Example: 20 Mbit/s

Example request:

```
http://isvserver.com/produceAPI?
activity=upgrade&amount=6456&instanceId=huaweitest123456&orderId=CS1906666688ABCDE&productId=0
0301-666688-0-0&saasExtendParams=W3sibmFtZSI6ImlkTnVtIiwidmFsdWUiOiIzNTIyNTU1NTU1NTU2NTYifS
x7Im5hbWUiOiJ1c2VyTmFtZSIsInZhbHVlIjoiaHVhd2VpMTIzIn0seyJuYW1lIjoiY3VzdEVtYWlsIiwidmFsdWUiOiIx
MjNAaHVhd2VpLmNvbSJ9XQ==&skuCode=d0abcd12-1234-5678-
ab90-11ab012aaaa1&testFlag=1&timeStamp=20191216013757582&authToken=a3Bl
+C93xv3ENgm40ngyYvQnYcTS/pgY5ugl20wtzGg=
```

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length of Characters	Description
resultCode	Yes	String	6	Invocation result code. For details, see **2.5 Invocation Result Codes**.
resultMsg	No	String	255	Invocation result description.

NOTE

When processing an interface request, the ISV server must ensure idempotence.

KooGallery may resend requests for a single order. When receiving a duplicate order with the same **orderId** value, the ISV server needs to return a success response, rather than upgrade the instance again.

Example response:

```
{
   "resultCode":"000000",
   "resultMsg":"success."
}
```

2.4.6 Resource Status Change

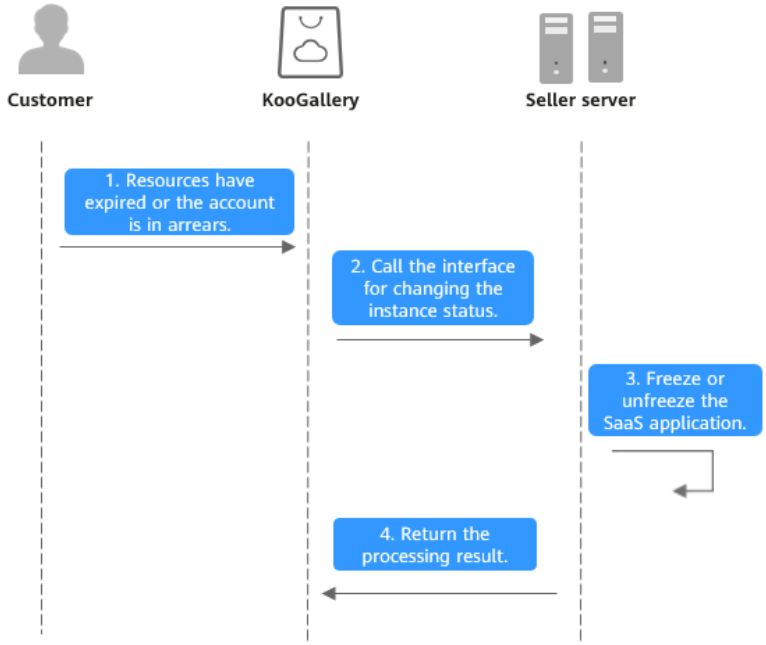
Description

After a customer purchases a pay-per-use product (or package), when the instance expires or the customer violates regulations, KooGallery calls this interface to freeze the instance.

NOTE

- If you receive an email indicating that the interface fails to be called in your email address of customer service or that one bound to your KooGallery account, handle the exception in a timely manner.
- KooGallery monitors interface exceptions. If a product has frequent interface exceptions, KooGallery will remove the product from the catalog.

The following figure shows the process of changing the resource status.



Request Message

The following table describes the request parameters.

Request method: GET

Parameter	Mandatory	Type	Maximum Length of Characters	Description
authToken	Yes	String	50	Security verification token. For details about the value, see **2.3.2 authToken Value**.
activity	Yes	String	32	Interface request ID, which is used to distinguish interface request scenarios. For resource status changes, the value is **instanceStatus**.
instanceId	Yes	String	64	Instance ID. **CAUTION** Use the instance ID returned by the pay-per-use billing interface.
instanceStatus	Yes	String	32	New status. • **FREEZE**: frozen. • **NORMAL**: unfrozen

Parameter	Mandatory	Type	Maximum Length of Characters	Description
timeStamp	Yes	String	20	Time (UTC time) when a request is initiated. Format: yyyyMMddHHmmssSSS
testFlag	No	String	2	Whether a request is submitted for debugging. • **1**: debugging request. • **0**: non-debugging request. The default value is **0**.

Example request:

```
Freezing an instance: https://example.isv.com?
activity=instanceStatus&instanceId=huaweitest123456&instanceStatus=FREEZE&testFlag=1&timeStamp=202
30327070251713&authToken=pqlrW7%2BPHC%2F1JE%2BMEjKxC94GGJreoS6PZHd982auw2o%3D
Unfreezing an instance: https://example.isv.com?
activity=instanceStatus&instanceId=huaweitest123456&instanceStatus=NORMAL&testFlag=1&timeStamp=20
230327070251713&authToken=pqlrW7%2BPHC%2F1JE%2BMEjKxC94GGJreoS6PZHd982auw2o%3D
```

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length of Characters	Description
resultCode	Yes	String	6	Invocation result code. For details, see **2.5 Invocation Result Codes**.
resultMsg	No	String	255	Invocation result description.

Example response:

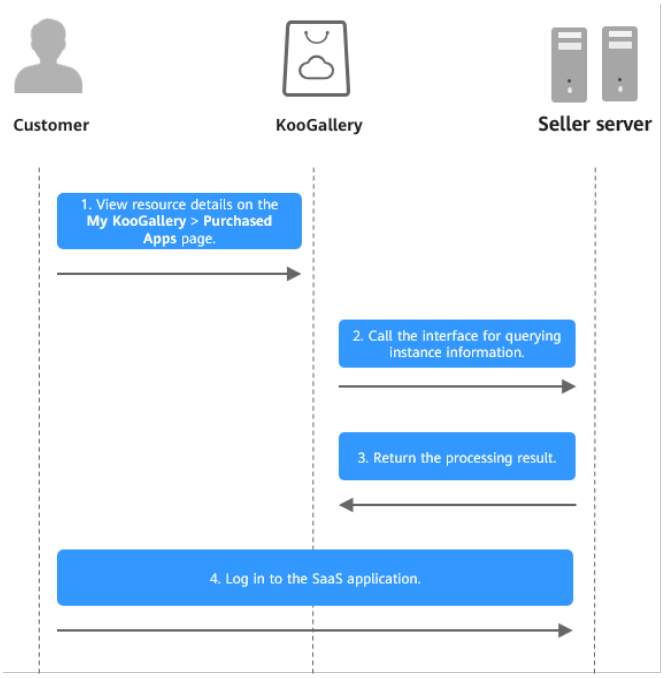
```
{
    "resultCode":"000000",
    "resultMsg":"success."
}
```


2.4.7 Instance Query

Description

- This API is optional for yearly/monthly/one-time products, but it is mandatory for pay-per-use specifications and packages.
- When a customer queries the instance information of a purchased pay-per-use product, you need to return the usage in real time.
- If a customer queries a pay-per-use package, you need to return the package usage through this API.

The following figure shows the process of querying instance information.



Request Message

The following table describes the request parameters.

Request method: GET

Parameter	Mandatory	Type	Maximum Length	Description
authToken	Yes	String	50	Security verification token. For details about the value, see **2.3.2 authToken Value**.
activity	Yes	String	20	Request ID, which is used to distinguish the scenario. For instance query, the value is **queryInstance**.

Parameter	Mandatory	Type	Maximum Length	Description
timeStamp	Yes	String	20	UTC timestamp when a request is initiated. Format: yyyyMMddHHmmssSSS
instanceId	Yes	String	64	Instance IDs separated by commas (,). Up to 100 instances can be queried each time.
testFlag	No	String	2	Whether a request is submitted for debugging. • **1**: debugging request. • **0**: non-debugging request. The default value is **0**.

Example request:

```
https://example.isv.com?activity=queryInstance&instanceId=ebc28eb6-4606-4098-b4bd-c201c99a0654%2Cfe28e27e-1157-4105-8592-24cc9488db10%2C92df74e4-163e-4e0b-a206-d9800d33881b&testFlag=1&timeStamp=20230327065233980&authToken=Eh%2F3Ud%2BR1j3d%2FwOui5CAcvRipM8IuribvgkXfJAsTfE%3D
```

Response Message

The following table describes the response parameters.

Table 2-1 Response parameters

Parameter	Mandatory	Type	Maximum Length	Description
resultCode	Yes	String	6	Result code. For details, see **2.5 Invocation Result Codes**.
resultMsg	No	String	255	Result message.

Parameter	Mandatory	Type	Maximum Length	Description
encryptType	No	String	3	Algorithm for encrypting sensitive information. **1**: AES256_CBC_PKCS5Padding (default) **2**: AES128_CBC_PKCS5Padding **NOTE** If **AES256_CBC_PKCS5Padding** is used, the return value is **1**. If **AES128_CBC_PKCS5Padding** is used, the return value is **2**.
info	No	InstanceInfo[]	/	Instance details.

The following table describes the **InstanceInfo** data structure.

Table 2-2 Response parameters

Parameter	Mandatory	Type	Maximum Length	Description
instanceId	Yes	String	64	Instance ID.

Parameter	Mandatory	Type	Maximum Length	Description
appInfo	No	AppInfo	N/A	App instance information. After a customer purchases a product, return a login address (website address) or an address that does not require login for the customer to perform subsequent operations. **NOTE** You must provide customers who purchase your SaaS products with the app usage information, including the addresses, accounts, and passwords. If the usage information can be sent through SMS messages, emails, or other methods, this parameter is not required in the response. Otherwise, the app instance information must be returned in the response. You can use the **memo** parameter to specify usage instructions or other information if any. For details about the **appInfo** data structure, see the following table.
usageInfo	No	UsageInfo[]	N/A	Usage information associated with an app instance. This parameter is required for pay-per-use specifications and packages. For pay-per-usage packages, the usage information of all billing items associated with the packages needs to be returned.

The following table describes the **AppInfo** data structure.

Table 2-3 Response parameters

Parameter	Mandatory	Type	Maximum Length	Description
frontEndUrl	Yes	String	512	Frontend URL. URL of the website that the customer can access to use the purchased product.

Parameter	Mandatory	Type	Maximum Length	Description
adminUrl	No	String	512	Management URL. URL of the backend website that the customer can access to manage the purchased product.
userName	No	String	128	Encrypted administrator account. Account (usually an email address or mobile number) used by a customer to access the management backend. The value consists of a 16-bit encryption IV and a Base-encoded username ciphertext. iv +base64(AES_CBC(*accessKey*,*userName*)) The account is encrypted using the key and encryption algorithm specified by **encryptType**. For details about the example code, see **2.7.3 ISV Server Encrypting the Username and Password After Resource Enabling**.
password	No	String	128	Encrypted initial password of the administrator. Password (usually generated by you) used by a customer to access the management backend. The value consists of a 16-bit encryption IV and a Base-encoded password ciphertext. iv+base64(AES_CBC(*accessKey*,*pwd*)) The password is encrypted using the key and encryption algorithm specified by **encryptType**. For details about the example code, see **2.7.3 ISV Server Encrypting the Username and Password After Resource Enabling**.
memo	No	String	1,024	Remarks.

The following table describes the **UsageInfo** data structure.

Table 2-4 Response parameters

Parameter	Mandatory	Type	Maximum Length	Description
relatedInstanceId	No	String	64	ID of the associated pay-per-use instance. When a customer queries the usage of resources in a pay-per-use package, the resource instance IDs also need to be returned. For example, if a package has 100 SMS messages and 50 MMS messages, **UsageInfo** and **relatedInstanceId** values of the two message types are returned.
usageValue	Yes	Double(12,4)	20	Usage value, with up to four decimal places. For pay-per-use specifications, the value is the accumulated resource usage. For pay-per-use packages, the value is the used package quota.
statisticalTime	Yes	String	20	UTC time when usage statistics are collected. Format: yyyyMMddHHmmssSSS
dashboardUrl	No	String	512	URL of the dashboard for viewing usage details. After purchasing a pay-per-use specification or pay-per-usage package, customers can view the usage information on this dashboard.

Example response:

```
{
   "resultCode" : "000000",
   "resultMsg" : "success.",
   "encryptType" : "1",
   "info" : [{
          "instanceId" : "ebc28eb6-4606-4098-b4bd-c201c99a0654",
          "appInfo" : {
             "frontEndUrl" : "https://www.baidu.com",
             "adminUrl" : "https://www.baidu.com/admin",
             "userName" : "huawei",
             "password" : "huawei123456",
             "memo" : "Test"
          },
          "usageInfo" : [{
                "relatedInstanceId" : "ebc28eb6-4606-4098-b4bd-c201c99a0654",
                "usageValue" : "0.12",
                "statisticalTime" : "20221101025113409",
                "dashboardUrl" : "https://www.baidu.com/dashboard"
             }
          ]
       }, {
          "instanceId" : "fe28e27e-1157-4105-8592-24cc9488db10",
          "appInfo" : {
```



```
            "frontEndUrl" : "https://www.baidu.com",
            "adminUrl" : "https://www.baidu.com/admin",
            "userName" : "huawei",
            "password" : "huawei123456",
            "memo" : "Test"
        },
        "usageInfo" : [{
            "relatedInstanceId" : "fe28e27e-1157-4105-8592-24cc9488db10",
            "usageValue" : "2042",
            "statisticalTime" : "20221101025113409",
            "dashboardUrl" : "https://www.baidu.com/dashboard"
        }
        ]
    }, {
        "instanceId" : "92df74e4-163e-4e0b-a206-d9800d33881b",
        "appInfo" : {
            "frontEndUrl" : "https://www.baidu.com",
            "adminUrl" : "https://www.baidu.com/admin",
            "userName" : "huawei",
            "password" : "huawei123456",
            "memo" : "Test"
        },
        "usageInfo" : [{
            "relatedInstanceId" : "ebc28eb6-4606-4098-b4bd-c201c99a0654",
            "usageValue" : "3309",
            "statisticalTime" : "20221101025113409",
            "dashboardUrl" : "https://www.baidu.com/dashboard"
        },
        {
            "relatedInstanceId" : "fe28e27e-1157-4105-8592-24cc9488db10",
            "usageValue" : "3309",
            "statisticalTime" : "20221101025113409",
            "dashboardUrl" : "https://www.baidu.com/dashboard"
        }
        ]
    }
  ]
}
```


2.4.8 Pay-per-Use Resource Usage Push

2.4.8.1 Usage Push (New)

Description

After a customer purchases and uses pay-per-use resources in KooGallery, call this interface to upload the service detail records (SDRs) of the customer. After obtaining the SDRs, KooGallery charges the customer for the usage.

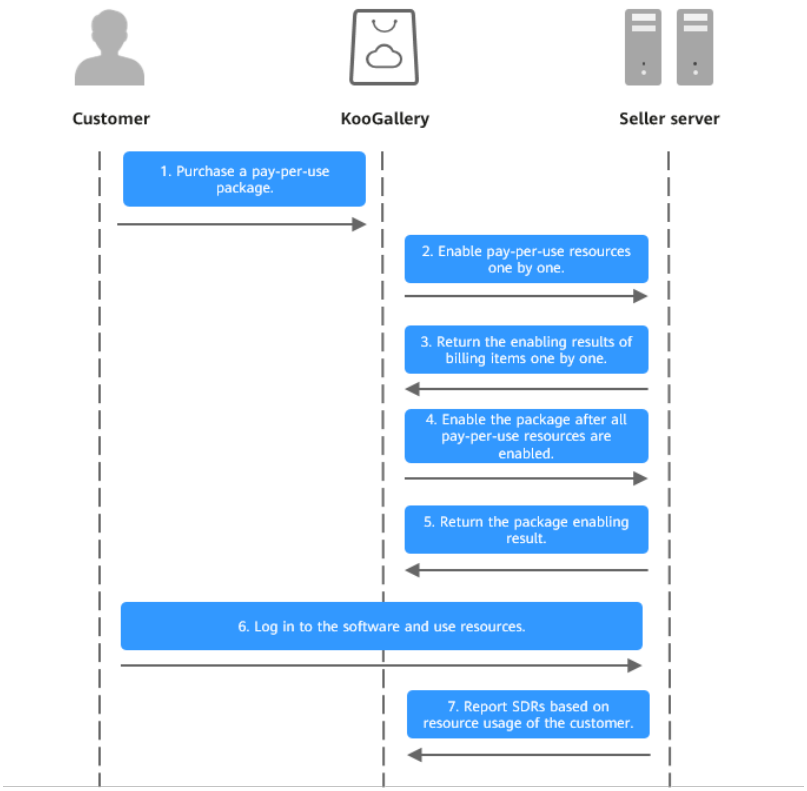
NOTE

For details about how to obtain SDKs, see **Calling APIs Through App Authentication**.

You can obtain an AK/SK on the **Access Keys** page.

URI

POST https://mkt-intl.myhuaweicloud.com/api/mkp-openapi-public/global/v1/isv/usage-data (public network)



Request Message

The following table describes the request parameters.

Request method: POST

Parameter	Mandatory	Type	Maximum Length	Description
signature	Yes	String	1,000	@Header Interface signature **(base64(hmacSHA256(***Seller interconnection key***,ts=***{ts}***&nonce=***{nonce}***&body=***{body}***)))** The body is signed after being sorted naturally.
ts	Yes	String	20	@Header Unix timestamp when an interface request is sent, in milliseconds.
nonce	Yes	String	64	@Header Security random number.

Parameter	Mandatory	Type	Maximum Length	Description
usage_records	Yes	List<UsagePushData>	1,000	SDR list. A list contains up to 1,000 **UsagePushData** records.

Table 2-5 UsagePushData

Parameter	Mandatory	Type	Maximum Length	Description
instance_id	Yes	String	64	Instance ID. Use the instance ID returned by the pay-per-use subscription interface.
record_time	Yes	String	17	Time when a usage record is generated (UTC). Format: yyyyMMdd'T'HHmmss'Z'
begin_time	Yes	String	17	Metering start time (UTC). Format: yyyyMMdd'T'HHmmss'Z'
end_time	Yes	String	17	Metering end time (UTC). Format: yyyyMMdd'T'HHmmss'Z'
usage_value	Yes	String	20	Usage value. The value is a positive number containing up to four significant decimal places.
metering_sn	Yes	String	64	Unique SDR ID. A random code is recommended.
relate_pkg_instance	No	String	64	This parameter is mandatory in SDRs of stop-before-excess packages. The package instance ID needs to be transferred.

Example request:

```
Post {domain}/api/mkp-openapi-public/global/v1/isv/usage-data
Content-Type:application/json
nonce: 6c63c221-1f6b-4141-8ff4-22f5dfe82b65
ts: 1709690865879
signature: gikLsIgimIscagwSamClFJ1CFT4QprHSDHW...
{
    "usage_records": [
        {
            "instance_id": "7f141bf1-aec8-4859-8323-fb3a8ad50721",
            "record_time": "20220809T091000Z",
            "begin_time": "20220809T080000Z",
```



```
        "end_time": "20220809T090000Z",
        "usage_value": "99",
        "metering_sn": "6c75c177b5fe4b8cbb6fc2aa33facfcd"
    },
    {
        "instance_id": "7f141bf1-aec8-4859-8323-fb3a8ad50721",
        "record_time": "20220809T091000Z",
        "begin_time": "20220809T080000Z",
        "end_time": "20220809T090000Z",
        "usage_value": "999",
        "metering_sn": "6c75c177b5fe4b8cbb6fc2aa33facfcb"
    }
  ]
}
```

NOTE

1. During SDR upload, if SDR data is abnormal, no error is reported at the interface layer. The backend periodically verifies and processes the uploaded data and generates available SDR data. If the backend fails to process the data, report the data again.

You can view abnormal data on the **Transaction Management > Service Detail Records** page of the Seller Console.

2. Requirements for the SDR report period:

- **Hourly billing**

  Report service detail records (SDRs) at least once an hour. It is recommended that SDRs be reported within the first 15 minutes of the next hour after a customer uses the resources. For example, if the customer uses resources at 13:25, report SDRs between 14:00 and 14:15. In this way, the customer can be charged in time. Otherwise, the fee deduction will be delayed. If you cannot report SDRs in real time, report them within 2 hours after resource consumption.

- **Daily billing**

  Report SDRs to KooGallery every hour. If you can only report SDRs once a day, report them from 00:00 to 00:15. SDRs must be reported before 01:00. Otherwise, the fee will be deducted from customers on the next day.

3. Requirements for reporting SDRs:

- **When a resource is not closed:**
- SDR start time (**begin_time**) ≥ Resource start time
- SDR start time (**begin_time**) ≤ SDR end time (**end_time**) ≤ SDR report time
- **When a resource is closed:**
- SDR end time (**end_time**) ≤ Resource close time

4. The time in the reported SDRs is the UTC time.

5. If the values of **begin_time** and **end_time** in a record are the same and the record is reported for multiple times, only the first record is processed. SDRs are collected at 01:00 every day for daily billing and fifteenth minute of every hour for hourly billing. Once SDRs are collected and formal bills are generated, SDRs cannot be corrected.

Duplicate SDRs are regarded as abnormal. You can view abnormal data on the **Transaction Management > Service Detail Records** page of the Seller Console.

6. The usage push interface uses the instance ID returned by the pay-per-use subscription interface instead of that returned by the pay-per-use package subscription interface.

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
error_code	Yes	String	6	Result code. For details, see **2.5 Invocation Result Codes**.
error_msg	No	String	255	Result message.
data	No	AbnormalUsageDataInfo		Information about abnormal SDRs.

Table 2-6 AbnormalUsageDataInfo

Parameter	Mandatory	Type	Maximum Length	Description
abnormal_usage_data	Yes	List<AbnormalUsageData>	1,000	List of abnormal SDRs.

Table 2-7 AbnormalUsageData

Parameter	Mandatory	Type	Maximum Length	Description
metering_sn	Yes	String	64	Unique ID of an SDR.

Parameter	Mandatory	Type	Maximum Length	Description
error_code	Yes	String	16	SDR-level error code. **001**: The instance does not exist. **002**: Invalid time format. **003**: Abnormal usage. **004**: Missing SDR ID. **005**: Duplicate SDR ID. **006**: The product corresponding to the instance has been removed from the catalog. **007**: The SDR has expired. **009**: The instance does not match the seller. **010**: Duplicate SDR. **011**: Invalid SDR time range. **012**: The instance is not a pay-per-use resource. **013**: The instance resource status is abnormal. **014**: The instance resource has been closed. **015**: The SDR start time is earlier than the resource enabling time. **016**: The instance is being enabled. **017**: In the stop-before-excess scenario, **relate_pkg_instance** is empty. **018**: In the stop-before-excess scenario, **relate_pkg_instance** is invalid or does not match **instance_id**.
error_msg	Yes	String	255	SDR error message.

Error Code

Table 2-8 AbnormalUsageData

HTTP Status Code	Error Code	Error Message	Description
200	MKT.0000	Success	Request successful.

HTTP Status Code	Error Code	Error Message	Description
500	94060001	System error!	Other internal errors.
401	94060002	Auth failed!	Input parameter verification failed. Invalid value.
400	94060004	Param invalid	Invalid parameter. The parameter is not defined by the interface, there are more parameters than required, or a mandatory parameter is missing. For example, a value is invalid or there is no instance ID.
400	94060005	Time format error	Incorrect time format.
400	94060006	TimeStamp invalid	Invalid timestamp.
401	94060007	Signature invalid	Signature verification fails.
400	94060008	Replay error	Request replay error.
500	94060009	Failed to report usage data	Report SDR failed.
401	94060010	Isv status invalid	Invalid seller status.
200	94060999	Failed	SDR-level error information is returned. For details, see the example response.

If an error code starting with **APIGW** is returned after you call an API, rectify the fault by referring to the instructions provided in **API Gateway Error Codes**.

Example response:

```
{
  "error_code":"MKT.0000",
  "error_msg":"Success"
}

{
   "error_code": "94060999",
   "error_msg": "Failed",
   "data": {
      "abnormal_usage_data": [
         {
            "error_code": "005",
            "error_msg": "METERING_SN_DUPLICATE",
            "metering_sn": "6c75c177b5fe4b8cbb6fc2aa33facfcb"
         }
      ]
```



```
  }
}
```

2.4.8.2 Usage Push (Old)

Description

After a customer purchases and uses pay-per-use resources in KooGallery, call this interface to upload the SDRs of the customer. After obtaining the SDRs, KooGallery charges the customer for the usage.

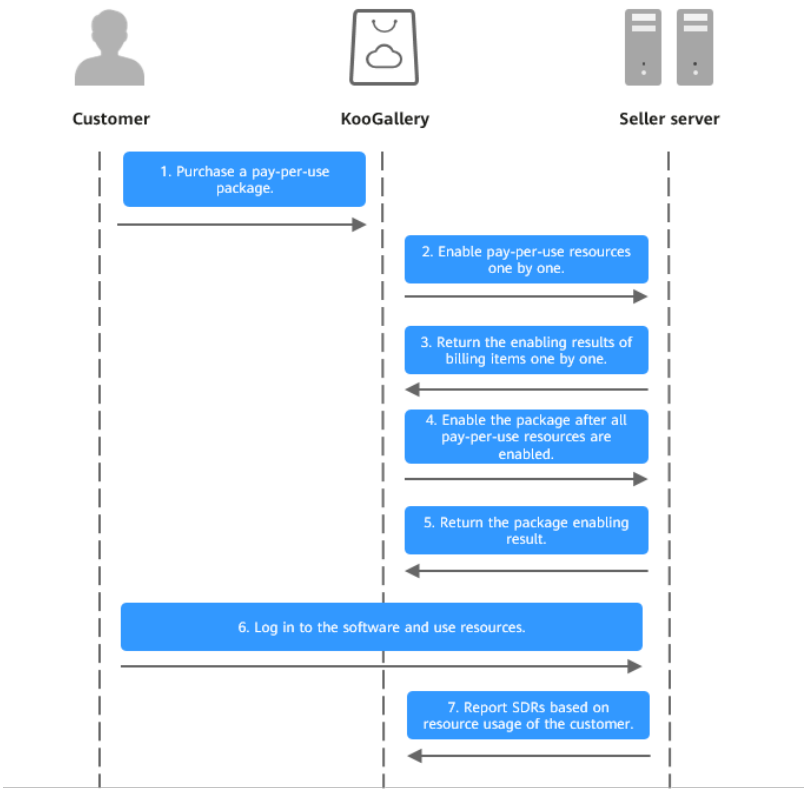
NOTE

For details about how to obtain SDKs, see **Calling APIs Through App Authentication**.

You can obtain an AK/SK on the **Access Keys** page.

URI

POST https://mkt-intl.myhuaweicloud.com/rest/marketplace/v1/isv/usage-data (public network)



Request Message

The following table describes the request parameters.

Request method: POST

Parameter	Mandatory	Type	Maximum Length	Description
usage_records	Yes	List<UsagePushData>	1,000	SDR list. A list contains up to 1,000 **UsagePushData** records.

Table 2-9 UsagePushData

Parameter	Mandatory	Type	Maximum Length	Description
instance_id	Yes	String	64	Pay-per-use instance ID. Use the instance ID returned by the pay-per-use subscription interface.
product_id	Yes	String	64	ID of the product corresponding to the instance.
record_time	Yes	String	17	Time when a usage record is generated (UTC). Format: yyyyMMdd'T'HHmmss'Z'
begin_time	Yes	String	17	Metering start time (UTC). Format: yyyyMMdd'T'HHmmss'Z'
end_time	Yes	String	17	Metering end time (UTC). Format: yyyyMMdd'T'HHmmss'Z'
usage_value	Yes	Double(12,4)	20	Usage value. The value is a positive number containing up to four significant decimal places.
relate_pkg_instance	No	String	64	Package instance ID. This parameter is required in SDRs of stop-before-excess packages.

Example request:

```
{
   "usage_records": [
      {
         "instance_id": "7f141bf1-aec8-4859-8323-fb3a8ad50721",
         "record_time": "20220809T091000Z",
         "begin_time": "20220809T080000Z",
         "end_time": "20220809T090000Z",
         "usage_value": "99"
      },
      {
         "instance_id": "7f141bf1-aec8-4859-8323-fb3a8ad50721",
         "record_time": "20220809T091000Z",
         "begin_time": "20220809T080000Z",
```



```
        "end_time": "20220809T090000Z",
        "usage_value": "999"
    }
  ]
}
```

NOTE

1. During SDR upload, if SDR data is abnormal, no error is reported at the interface layer. The backend periodically verifies and processes the uploaded data and generates available SDR data. If the backend fails to process the data, report the data again.

You can view abnormal data on the **Transaction Management > Service Detail Records** page of the Seller Console.

2. Requirements for the SDR report period:

- **Hourly billing**

  Report SDRs at least once an hour. It is recommended that SDRs be reported within the first 15 minutes of the next hour after a customer uses the resources. For example, if the customer uses resources at 13:25, report SDRs between 14:00 and 14:15. In this way, the customer can be charged in time. Otherwise, the fee deduction will be delayed. If you cannot report SDRs in real time, report them within 2 hours after resource consumption.

- **Daily billing**

  Report SDRs to KooGallery every hour. If you can only report SDRs once a day, report them from 00:00 to 00:15. SDRs must be reported before 01:00. Otherwise, the fee will be deducted from customers on the next day.

3. Requirements for reporting SDRs:

- **When a resource is not closed:**
  - SDR start time (**begin_time**) ≥ Resource start time
  - SDR start time (**begin_time**) ≤ SDR end time (**end_time**) ≤ SDR report time
- **When a resource is closed:**
  - SDR end time (**end_time**) ≤ Resource close time

4. The time in the reported SDRs is the UTC time.

5. If the values of **begin_time** and **end_time** in a record are the same and the record is reported for multiple times, only one record is processed. SDRs are collected at 01:00 every day for daily billing and fifteenth minute of every hour for hourly billing. Once SDRs are collected and formal bills are generated, SDRs cannot be corrected.

- 6. The usage push interface uses the instance ID returned by the pay-per-use subscription interface instead of that returned by the pay-per-use package subscription interface.

Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length	Description
error_code	Yes	String	6	Result code. For details, see the following error codes.
error_msg	No	String	255	Result message.

The following table describes the error codes.

HTTP Status Code	Error Code	Error Message	Description
200	MKT.0000	Success.	Request successful.
500	MKT.0999	System internal error.	Other internal errors.
500	MKT.0100	Failure of input parameter	Input parameter verification failed. Invalid value.
400	MKT.0101	Invalid parameter	Invalid parameter. The parameter is not defined by the interface, there are more parameters than required, or a mandatory parameter is missing.
400	MKT.0102	Invalid body sign	Failed to verify the signature of the request body.
400	MKT.0199	Request parameter error	Incorrect request parameter.
401	MKT.0150	Illegal operation	You are trying to perform an unauthorized operation. For example, the product corresponding to **instance_id** is not released by the seller corresponding to the AK/SK.
401	MKT.0151	No authority	Insufficient permissions to access the API. The token does not belong to a seller.
401	MKT.0154	Illegal token	Authentication failed. Invalid token.
500	MKT.9001	Instance ID not found.	The instance ID does not exist. (This result code may be returned when the renewal, expiration, or resource release interface is called.)
500	MKT.9002	Invalid usage entities.	Invalid usage entities.
500	MKT.9003	Usage records extend size limit.	Too many records. Max. records: 1,000.
500	MKT.9004	Record beginTime extends Limit.	The start time exceeds the validity period (last 21 days).

If an error code starting with **APIGW** is returned after you call an API, rectify the fault by referring to the instructions provided in **API Gateway Error Codes**.

Example response:

```
{
  "error_code":"MKT.0000",
  "error_msg":"success"
}
```

2.5 Invocation Result Codes

Module	Result Code	Description
Common	000000	Succeeded.
	000001	Authentication failed.
	000002	Invalid request parameter.
	000003	The instance ID does not exist. (This result code may be returned when the renewal, expiration, or resource release interface is called.)
	000004	The request is being processed.
	000005	Other internal errors.
Subscription	000100	No instance resource can be allocated.

2.6 Interface Debugging

To ensure that SaaS products can be accessed, KooGallery provides a debugging page on the Seller Console. You can debug SaaS interfaces for subscription, renewal, expiration, and release.

The following uses the subscription interface as an example.

Step 1 Preset parameters in the ISV server according to the **Parameter Description** column in **Request Message**.

Step 2 Go to the **Seller Console**.

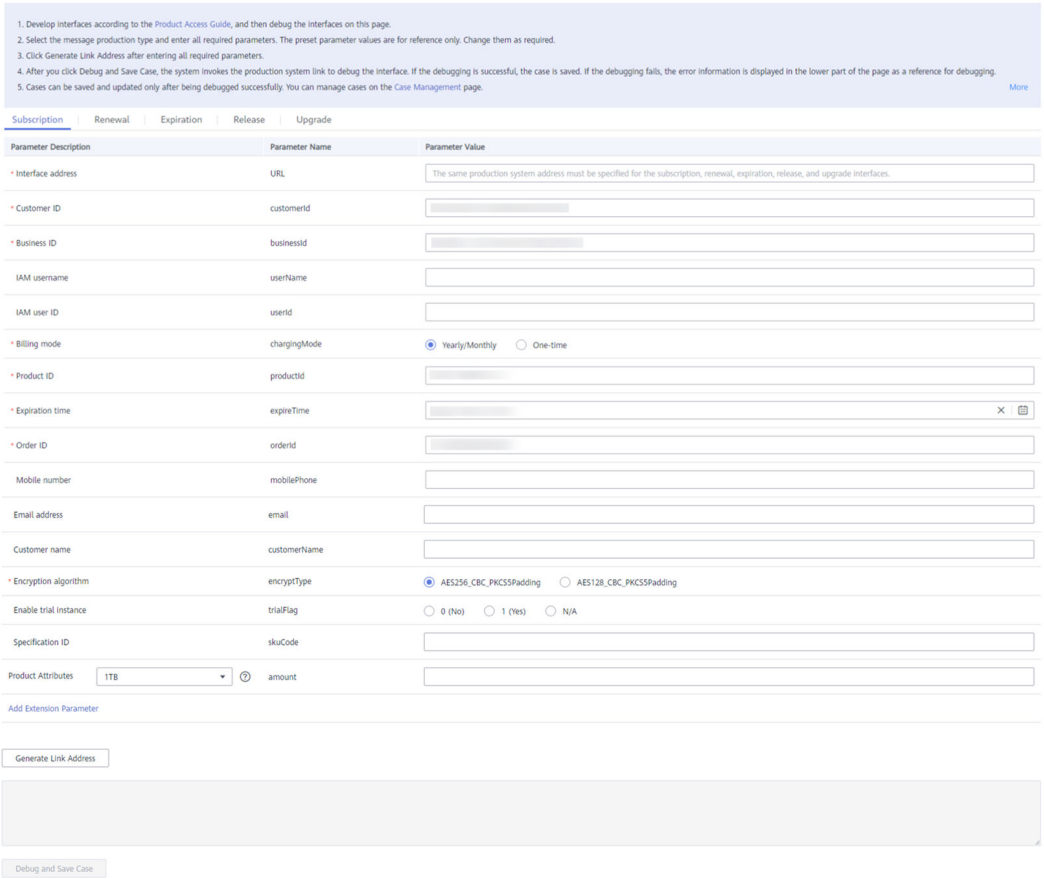
Step 3 In the navigation pane, choose **Application Tools** > **Application Access Debugging**.

Step 4 On the **Subscription** tab page, enter the parameter values preset in **Step 1**, and click **Generate Link Address** to generate an example request. For details about the parameters, see **Interface Description**.

(Optional) If product specifications that are priced using a custom template contain quantity type attributes, such as a number, amount of bandwidth, or disk size, create the attributes on the product attribute management page, and

navigate to the **Application Access Debugging** page to set related parameters and debug the interface. After the debugging is successful, you can release the product specifications.

(Optional) Click **Add Extension Parameter** to add up to three extension parameters that are required for product subscription. Ensure that the interface containing the extension parameters to be added have been debugged successfully. To add a non-default parameter type, send an email to the KooGallery operations manager (partner@huaweicloud.com) to apply for adding the required parameter type. The application result is subject to the KooGallery feedback.



NOTE

- If the SaaS product **does not involve service supervision**, set **Provision Type** to **Provision upon subscription**. If the SaaS product **involves service supervision**, set **Provision Type** to **Provision after acceptance**.
- Extension parameters are optional.
- The extension parameters are a JSON string carried in the **url** parameter in the form of **urlEncode(base64(saasExtendParams))**. After obtaining the value of the **saasExtendParams** parameter, your server needs to use **base64Decode(urlDecode(saasExtendParams))** to obtain the JSON string of the extension parameters.

  For example, **emailDomainName** and **extendParamName** in the JSON string **[{"name":"emailDomainName","value":"test.xxxx.com"}, {"name":"extendParamName","value":"extendParamValue"}]** are the parameter values set during product release.

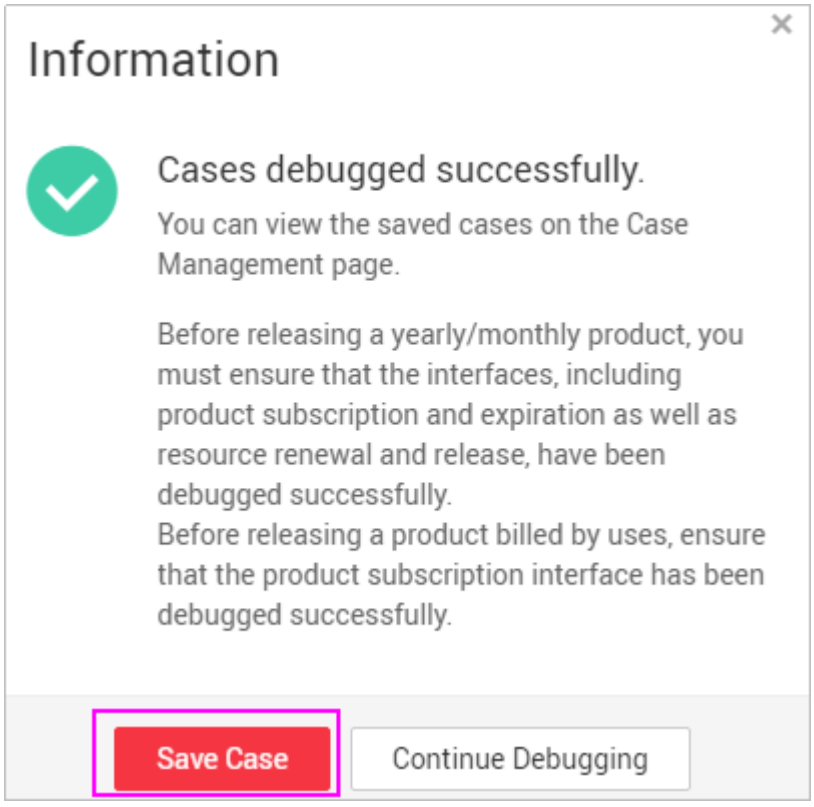
Step 5 Click **Debug and Save Case**. The system calls the interface address to test the interface. If the test is successful, go to **Step 6**. If not, the error message is displayed in the lower part of the page. You can modify the interface based on the error message.

NOTE

- To release a product billed on a yearly/monthly basis, you need to debug the interface for product subscription and expiration as well as resource renewal and release, and save all the cases. On the **Subscription** tab page, set **Billing mode** to **Yearly/Monthly**.
- To release a product billed by one-time payment, you need to debug the interface for product subscription and resource release, and save the cases. On the **Subscription** tab page, set **Billing mode** to **One-time**.
- To release a product that can be billed either on a yearly/monthly basis or by one-time payment, you need to debug the interface for product subscription and expiration as well as resource renewal and release, and save all the cases. Save two cases for product subscription. Set **Billing mode** to **Yearly/Monthly** in one case and to **One-time** in the other.

Step 6 A message is displayed indicating the cases are debugged successfully. Click **Save Case**.

You can choose **Application Tools > Case Management** to view the cases that are successfully debugged.

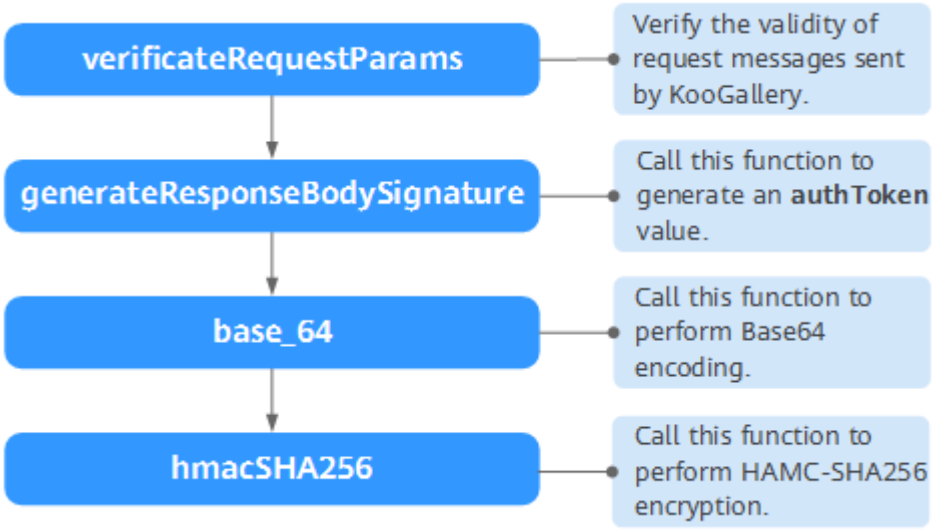


----End

2.7 Code Example (Java)

2.7.1 ISV Server Verifying Requests

The following figure shows the overall process of code invocation for request verification.




```
/**
* Verify the validity of requests.
* @param request --HTTP requests
* @param accessKey --Access key
* @param encryptLength --Length of the encrypted content
* @return --Verification result
*/
public static boolean verificateRequestPar-
ams(javax.servlet.http.HttpServletRequest request,
String accessKey,int encryptLength)
{
// Resolve the URL.
Map<String, String[]> paramsMap = request.getParameterMap();
String timeStamp = null;
String authToken = null;
String[] timeStampArray = paramsMap.get("timeStamp");
if (null != timeStampArray && timeStampArray.length > 0)
{
timeStamp = timeStampArray[0];
}
String[] authTokenArray = paramsMap.remove("authToken");
if (null != authTokenArray && authTokenArray.length > 0)
{
authToken = authTokenArray[0];
}
// Sort the remaining parameters and combine them to form the encrypted
content.
Map<String, String[]> sortedMap = new TreeMap<String, String[]>();
sortedMap.putAll(paramsMap);
StringBuffer strBuffer = new StringBuffer();
Set<String> keySet = sortedMap.keySet();
Iterator<String> iter = keySet.iterator();
while (iter.hasNext())
{
String key = iter.next();
String value = sortedMap.get(key)[0];
strBuffer.append("&").append(key).append("=").append(value);
}
// Rectify the message body by removing the ampersand (&) before the first
parameter.
```



```
String reqParams = strBuffer.toString().substring(1);
String key = accessKey + timeStamp;
String signature = null;
try
{
signature = generateResponseBodySignature(key, reqParams);
}
catch (InvalidKeyException | NoSuchAlgorithmException
| IllegalStateException | UnsupportedEncodingException e)
{
// TODO Auto-generated catch block
}
return authToken.equals(signature);
}
```

```
/**
* Generate an example signature demo of an HTTP response body.
* @param key --Access key obtained on the Seller Console. Log in to the Seller Console to view the access key.
* @param body --HTTP response message body
* @return --Encryption result
* @throws InvalidKeyException
* @throws NoSuchAlgorithmException
* @throws IllegalStateException
* @throws UnsupportedEncodingException
*/
public static String generateResponseBodySignature(String key, String body)
throws InvalidKeyException, NoSuchAlgorithmException,
IllegalStateException, UnsupportedEncodingException
{
return base_64(hmacSHA256(key, body));
}
```



```
/**
*
* HAMC-SHA256 encryption algorithm
* @param macKey --Key
* @param macData --Encryption content, that is, the response message body
* @return --Ciphertext
* @throws NoSuchAlgorithmException
* @throws InvalidKeyException
* @throws IllegalStateException
* @throws UnsupportedEncodingException
*/
public static byte[] hmacSHA256(String macKey, String macData)
throws NoSuchAlgorithmException, InvalidKeyException,
IllegalStateException, UnsupportedEncodingException
{
SecretKeySpec secret =
new SecretKeySpec(macKey.getBytes(), "HmacSHA256");
Mac mac = Mac.getInstance("HmacSHA256");
mac.init(secret);
byte[] doFinal = mac.doFinal(macData.getBytes("UTF-8"));
return doFinal;
}
```

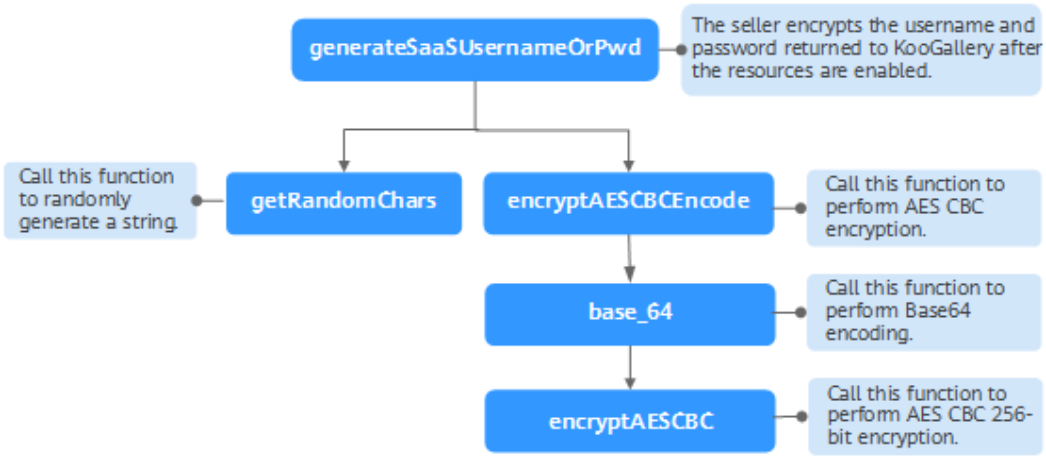
```
/**
*
* Convert the byte array into a string.
* @param bytes --Byte array
* @return --String
*/
public static String base_64(byte[] bytes)
{
return new String(Base64.encodeBase64(bytes));
}
```

2.7.2 ISV Server Signing a Response Message Body

For the example code, see the generateResponseBodySignature method in **2.7.1 ISV Server Verifying Requests**.

2.7.3 ISV Server Encrypting the Username and Password After Resource Enabling

The following figure shows the overall process of code invocation.




```
/**
* Encrypt the username and password returned after the resources are released.
* @param key --Key
* @param str --Original content
* @param encryptLength --Length of the encrypted content
* @return --Encryption result
*/
public static String generateSaaSUsernameOrPwd(String key, String str, int
encryptLength)
{
String iv = getRandomChars(16);
String afterEncryptStr = "";
try
{
afterEncryptStr = encryptAESCBCEncode(str, key, iv, encryptLength);
}
catch (InvalidKeyException | NoSuchAlgorithmException
| NoSuchPaddingException | InvalidAlgorithmParameterException
| IllegalBlockSizeException | BadPaddingException e)
{
//TODO: Troubleshooting
}
System.out
.println(afterEncryptStr);
return iv + afterEncryptStr;
}
```



```
/**
* Randomly generate a string.
* @param length --Length of the randomly generated string
* @return --Random string
*/
public static String getRandomChars(int length)
{
String randomChars = "";
SecureRandom random = new SecureRandom();
for (int i = 0; i < length; i++)
{
// Randomly choose letters and digits.
if (random.nextInt(2) % 2 == 0)
{
// Specify whether uppercase or lowercase letters are output.
int letterIndex = random.nextInt(2) % 2 == 0 ? 65 : 97;
randomChars += (char) (random.nextInt(26) + letterIndex);
}
else
{
randomChars += String.valueOf(random.nextInt(10));
}
}
return randomChars;
}
```

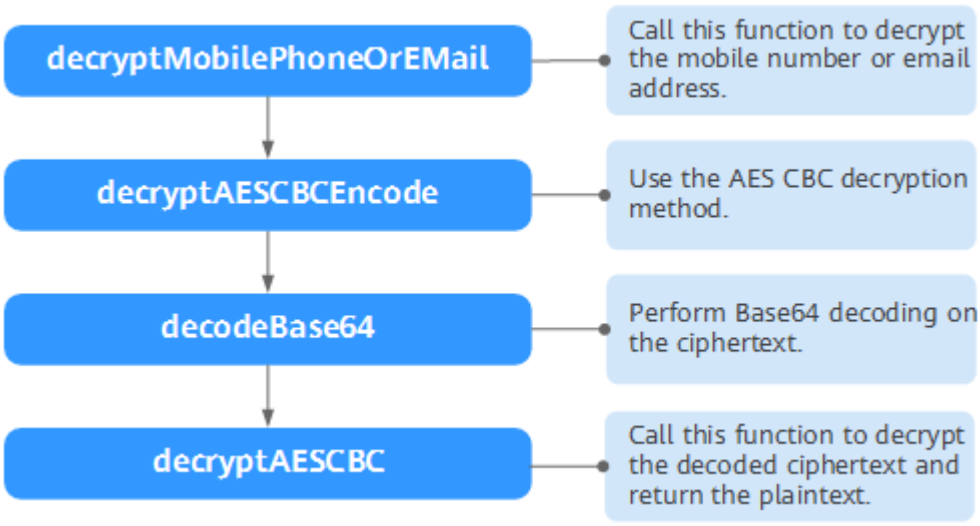
```
/**
* AES CBC encryption
* @param content --Content to be encrypted
* @param key --Encryption key
* @param iv --IV
* @param encryptLength --Only lengths of 128 bits and 256 bits are supported.
* @return --Encryption result
* @throws BadPaddingException
* @throws IllegalBlockSizeException
* @throws InvalidAlgorithmParameterException
* @throws NoSuchPaddingException
* @throws NoSuchAlgorithmException
* @throws InvalidKeyException
*/
public static String encryptAESCBCEncode(String content, String key,
String iv, int encryptLength)
throws InvalidKeyException, NoSuchAlgorithmException,
NoSuchPaddingException, InvalidAlgorithmParameterException,
IllegalBlockSizeException, BadPaddingException
{
if (StringUtils.isEmpty(content) || StringUtils.isEmpty(key)
|| StringUtils.isEmpty(iv))
{
return null;
}
return base_64(
encryptAESCBC(content.getBytes(), key.getBytes(), iv.getBytes(),
encryptLength));
}
```



```
/**
*
* AES CBC 256-bit encryption
* @param content --Byte array of the encrypted content
* @param keyBytes --Encrypted byte array
* @param iv --Byte array of the encrypted IV
* @param encryptLength --Only lengths of 128 bits and 256 bits are supported.
* @return --Decrypted byte content
* @throws NoSuchAlgorithmException
* @throws NoSuchPaddingException
* @throws InvalidKeyException
* @throws InvalidAlgorithmParameterException
* @throws IllegalBlockSizeException
* @throws BadPaddingException
*/
public static byte[] encryptAESCBC(byte[] content, byte[] keyBytes,
byte[] iv, int encryptLength)
throws NoSuchAlgorithmException, NoSuchPaddingException,
InvalidKeyException, InvalidAlgorithmParameterException,
IllegalBlockSizeException, BadPaddingException
{
KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");
SecureRandom secureRandom = SecureRandom.getInstance("SHA1PRNG");
secureRandom.setSeed(keyBytes);
keyGenerator.init(encryptLength, secureRandom);
SecretKey key = keyGenerator.generateKey();
Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
cipher.init(Cipher.ENCRYPT_MODE, key, new IvParameterSpec(iv));
byte[] result = cipher.doFinal(content);
return result;
}
```

2.7.4 ISV Server Decrypting the Mobile Number and Email Address

The following figure shows the code invocation.




```
/**
*
* Decrypt a mobile number or an email address.
* @param key --Key
* @param str --Ciphertext
* @param encryptLength --Length of the encrypted content
* @return --Decryption result
*/
public static String decryptMobilePhoneOrEMail(String key, String str, int
encryptLength)
{
if(null != str && str.length() > 16)
{
String iv = str.substring(0, 16);
String encryptStr = str.substring(16);
String result = null;
try
{
result = decryptAESCBCEncode(encryptStr,
key,
iv,
encryptLength);
}
catch (InvalidKeyException | NoSuchAlgorithmException
| NoSuchPaddingException | InvalidAlgorithmParameterException
| IllegalBlockSizeException | BadPaddingException e)
{
//TODO: Troubleshooting
}
return result;
}
return null;
}
```



```
/**
* Decrypt AES-CBC-encrypted content.
* @param content --Original content
* @param key --Key
* @param iv --IV
* @return --Decryption result
* @throws BadPaddingException
* @throws IllegalBlockSizeException
* @throws InvalidAlgorithmParameterException
* @throws NoSuchPaddingException
* @throws NoSuchAlgorithmException
* @throws InvalidKeyException
*/
public static String decryptAESCBCEncode(String content, String key,
String iv, int encryptType) throws InvalidKeyException, NoSuchAlgorithmException, NoSuchPaddingException, InvalidAlgorithmParameterException, IllegalBlockSizeException, BadPaddingException
{
if (StringUtils.isEmpty(content) || StringUtils.isEmpty(key)
|| StringUtils.isEmpty(iv))
{
return null;
}
return new String(decryptAESCBC(Base64.decodeBase64(content.getBytes()),
key.getBytes(),
iv.getBytes(),encryptType));
}

public static byte[] decryptAESCBC(byte[] content, byte[] keyBytes,
byte[] iv, int encryptType) throws NoSuchAlgorithmException, NoSuchPaddingException, InvalidKeyException, InvalidAlgorithmParameterException, IllegalBlockSizeException, BadPaddingException
{
KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");
SecureRandom secureRandom = SecureRandom.getInstance("SHA1PRNG");
secureRandom.setSeed(keyBytes);
keyGenerator.init(encryptType, secureRandom);
SecretKey key = keyGenerator.generateKey();
Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
cipher.init(Cipher.DECRYPT_MODE, key, new IvParameterSpec(iv));
byte[] result = cipher.doFinal(content);
```



```
return result;
}
```

2.7.5 Java Code Example

```
package com.huawei.cbc.cbcmarketplacecommentservice.ability.jsonutils;

import java.io.UnsupportedEncodingException;
import java.security.InvalidAlgorithmParameterException;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;
import java.util.Set;
import java.util.TreeMap;

import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.KeyGenerator;
import javax.crypto.Mac;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;

import org.apache.commons.codec.binary.Base64;
import org.apache.commons.lang.StringUtils;

public class EncryptTest {
// Encoding format
private static final String CHARSET = "UTF-8";
```

```
// If encryptType is set to AES256_CBC_PKCS5Padding, 1 is transferred. If encryptType is set to AES128_CBC_PKCS5Padding, 2 is transferred.

private static final String ENCRYPT_TYPE_256 = "1";


// Key displayed on the Seller Information page (Replace xxxxxxx with the actual key.)

private static final String ACCESS_KEY = "xxxxxxx";


public static void main(String args[]) throws Exception {

// ------------Request verification----------------

// Convert the request into a map and simulate the operation of obtaining parameters from the request (request.getParameterMap())

Map<String, String[]> paramsMap = getTestUrlMap();

// Encryption type. The value can be AES256_CBC_PKCS5Padding (256-bit encryption) or AES128_CBC_PKCS5Padding (128-bit encryption).

System.out.println("Request verification:" + verificateRequestParams(paramsMap, ACCESS_KEY, 256));


// Mobile number and password to be encrypted

String needEncryptStr = "15905222222";

String encryptStr = generateSaaSUsernameOrPwd(needEncryptStr, ACCESS_KEY, ENCRYPT_TYPE_256);

System.out.println("Mobile number and password to be encrypted:"+ encryptStr);

// Decryption

String decryptStr = decryptMobilePhoneOrEMail(ACCESS_KEY, encryptStr, ENCRYPT_TYPE_256);

System.out.println("Mobile number and password to be decrypted:"+ decryptStr);


// Body signature

String needEncryptBody =

"{\"resultCode\":\"00000\",\"resultMsg\":\"Purchase succeeded\",\"encryptType\":\"1\",\"instanceId\":\"000bd4e1-5726-4ce9-8fe4-fd081a179304\",\"appInfo\"{\"userName\":\"3LQvu8363e5O4zqwYnXyJGWz8y+GAcu0rpM0wQ==\",\"password\":\"RY31aEnR5GMCFmt3iG1hW7UF1HK09MuAL2sgxA==\"}}";

String encryptBody = generateResponseBodySignature(ACCESS_KEY, needEncryptStr);

System.out.println("Body signature:"+ encryptBody);

}


private static Map<String, String[]> getTestUrlMap() {
```



```
// Original request: http://bzapic.natappfree.cc?
activity=newInstance&businessId=61e834ba-7b97-4418-b8f7-
e5345137278c&customerId=68cbc86abc2018ab880d92f36422fa0e&expireTime=20
200727153156&orderId=CS1906666666ABCDE&productId=00301-666666-0--0&tes
tFlag=1&timeStamp=20200727073711903&authToken=Gzbfjf9LHRBcI3bFVi+
+sLinCNOBF6qa7is1fvjEgYQ=

Map<String, String[]> paramsMap = new HashMap<String, String[]>();

paramsMap.put("activity", new String[] {"newInstance"});

paramsMap.put("businessId", new String[] {"61e834ba-7b97-4418-b8f7-
e5345137278c"});

paramsMap.put("customerId", new String[]
{"68cbc86abc2018ab880d92f36422fa0e"});

paramsMap.put("expireTime", new String[] {"20200727153156"});

paramsMap.put("orderId", new String[] {"CS1906666666ABCDE"});

paramsMap.put("productId", new String[] {"00301-666666-0--0"});

paramsMap.put("testFlag", new String[] {"1"});

paramsMap.put("timeStamp", new String[] {"20200727073711903"});

paramsMap.put("authToken", new String[] {"Gzbfjf9LHRBcI3bFVi+
+sLinCNOBF6qa7is1fvjEgYQ="});

return paramsMap;

}


/**

* Verify the validity of the request.

*

* @param accessKey Access key

* @param encryptLength Length of the encrypted content

* @return Verification result

*/

public static boolean verificateRequestParams(Map<String, String[]> paramsMap,
String accessKey,

int encryptLength) {

String timeStamp = null;

String authToken = null;

String[] timeStampArray = paramsMap.get("timeStamp");


if (null != timeStampArray && timeStampArray.length > 0) {

timeStamp = timeStampArray[0];
```



```
}
String[] authTokenArray = paramsMap.get("authToken");
if (null != authTokenArray && authTokenArray.length > 0) {
authToken = authTokenArray[0];
}
// Sort the remaining parameters and combine them to form the encrypted content.
Map<String, String[]> sortedMap = new TreeMap<String, String[]>();
sortedMap.putAll(paramsMap);
sortedMap.remove("authToken");
StringBuffer strBuffer = new StringBuffer();
Set<String> keySet = sortedMap.keySet();
Iterator<String> iter = keySet.iterator();
while (iter.hasNext()) {
String key = iter.next();
String value = sortedMap.get(key)[0];
strBuffer.append("&").append(key).append("=").append(value);
}
// Rectify the message body by removing the ampersand (&) before the first parameter.
String reqParams = strBuffer.toString().substring(1);
String key = accessKey + timeStamp;
String signature = null;
try {
signature = generateResponseBodySignature(key, reqParams);
} catch (InvalidKeyException | NoSuchAlgorithmException | IllegalStateException
| UnsupportedEncodingException e) {
// TODO Auto-generated catch block
}
return authToken.equals(signature);
}

public static String generateResponseBodySignature(String key, String body)
throws InvalidKeyException, NoSuchAlgorithmException, IllegalStateException, UnsupportedEncodingException {
```



```
return base_64(hmacSHA256(key, body));
}

public static byte[] hmacSHA256(String macKey, String macData) {
try {
try {
SecretKeySpec secret = new SecretKeySpec(macKey.getBytes(CHARSET),
"HmacSHA256");
Mac mac = Mac.getInstance("HmacSHA256");
mac.init(secret);
return mac.doFinal(macData.getBytes(CHARSET));
} catch (UnsupportedEncodingException e) {
} catch (InvalidKeyException e) {
}
} catch (NoSuchAlgorithmException e) {
}
return new byte[0];
}

// Body signature
public static String generateSaaSUsernameOrPwd(String isvBody, String
decryptAccessKey, String sEncryptType) {
String iv = getRandomChars(16);
int iEncryptType = 0;
try {
iEncryptType = Integer.parseInt(sEncryptType);
} catch (NumberFormatException exception) {
iEncryptType = 1;
}
int encryptType;
if (1 == iEncryptType) {
encryptType = 256;
} else {
encryptType = 128;
}
```



```
String isvEncryptBody = encryptAESCBCEncode(isvBody, decryptAccessKey, iv,
encryptType);

return iv + isvEncryptBody;

}

/**

* AES CBC 256-bit encryption

*

* @param content Content to be encrypted

* @param key Encryption key

* @param iv Encrypted salt value

* @return Encryption result

*/

public static String encryptAESCBCEncode(String content, String key, String iv, int
encryptType) {

if (StringUtils.isEmpty(content) || StringUtils.isEmpty(key) ||
StringUtils.isEmpty(iv)) {

return null;

}

try {

byte[] encrypContent =

encryptAESCBC(content.getBytes(CHARSET), key.getBytes(CHARSET),
iv.getBytes(CHARSET), encryptType);

if (null != encrypContent) {

return base_64(encrypContent);

} else {

return null;

}

} catch (UnsupportedEncodingException e) {

return null;

}

}

public static byte[] encryptAESCBC(byte[] content, byte[] keyBytes, byte[] iv, int
encryptType) {
```



```
try {
KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");
SecureRandom secureRandom = SecureRandom.getInstance("SHA1PRNG");
secureRandom.setSeed(keyBytes);
keyGenerator.init(encryptType, secureRandom);
SecretKey key = keyGenerator.generateKey();
Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
cipher.init(Cipher.ENCRYPT_MODE, key, new IvParameterSpec(iv));
return cipher.doFinal(content);
} catch (Exception e) {
}
return null;
}

public static String base_64(byte[] bytes) {
try {
return new String(Base64.encodeBase64(bytes), CHARSET);
} catch (UnsupportedEncodingException e) {
return null;
}
}

static String decryptMobilePhoneOrEMail(String accessKey, String encryptStr,
String sEncryptType) {
String iv = encryptStr.substring(0, 16);
int iEncryptType = 1;
try {
iEncryptType = Integer.parseInt(sEncryptType);
} catch (NumberFormatException exception) {
exception.printStackTrace();
}
int encryptType;
if (1 == iEncryptType) {
encryptType = 256;
} else {
```

```
encryptType = 128;
}
String decryptBody = null;
try {
decryptBody = decryptAESCBCEncode(encryptStr.substring(16), accessKey, iv,
encryptType);
} catch (Exception e) {
e.printStackTrace();
return decryptBody;
}
return decryptBody;
}

public static String decryptAESCBCEncode(String content, String key, String iv, int
encryptType)
throws InvalidKeyException, NoSuchAlgorithmException, NoSuchPaddingException,
InvalidAlgorithmParameterException, IllegalBlockSizeException,
BadPaddingException {
if (StringUtils.isEmpty(content) || StringUtils.isEmpty(key) ||
StringUtils.isEmpty(iv)) {
return null;
}
return new
String(decryptAESCBC(org.apache.commons.codec.binary.Base64.decodeBase64(co
ntent.getBytes()),
key.getBytes(), iv.getBytes(), encryptType));
}

public static byte[] decryptAESCBC(byte[] content, byte[] keyBytes, byte[] iv, int
encryptType)
throws NoSuchAlgorithmException, NoSuchPaddingException, InvalidKeyException,
InvalidAlgorithmParameterException, IllegalBlockSizeException,
BadPaddingException {
KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");
SecureRandom secureRandom = SecureRandom.getInstance("SHA1PRNG");
secureRandom.setSeed(keyBytes);
keyGenerator.init(encryptType, secureRandom);
```



```
SecretKey key = keyGenerator.generateKey();
Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
cipher.init(Cipher.DECRYPT_MODE, key, new IvParameterSpec(iv));
byte[] result = cipher.doFinal(content);
return result;
}

/**
* Obtain a random character string.
*
* @param length Character string length
* @return
* @author d00420944
*/
public static String getRandomChars(int length) {
StringBuffer randomCharsBuf = new StringBuffer(1024);

SecureRandom random = new SecureRandom();
for (int i = 0; i < length; i++) {
// Randomly choose letters and digits.
if (random.nextInt(2) % 2 == 0) {
// Specify whether uppercase or lowercase letters are output.
int letterIndex = random.nextInt(2) % 2 == 0 ? 65 : 97;
randomCharsBuf.append((char) (random.nextInt(26) + letterIndex));
} else {
randomCharsBuf.append(String.valueOf(random.nextInt(10)));
}
}

String randomChars = randomCharsBuf.toString();

return randomChars;
}
}
```


2.8 FAQs

1. Why is the following error message displayed when I debug the service interface?
   ```
   http header->bodySign is empty.
   ```
   bodySign is empty is displayed if KooGallery does not obtain the **Body-Sign** message header. Add **Body-Sign** to the response of the interface. For details, see **2.3.3 HTTP Body Signature**.

   When the service interface is invoked, the message header returned by the ISV server must contain **Body-Sign**, which is case sensitive. Otherwise, the message cannot be identified. For example, if the message header is **Body-sign**, it cannot be identified by KooGallery.
2. Why is the error message "Failed to verify the HTTP Body signature. Expected signature value: **********" displayed when I debug the service interface?

   This is because the HTTP body content changes after the signature is obtained and before the message is sent. The possible causes of the change include attribute sequence changes, a blank space added, and a newline character added (\n is added to some output streams). Troubleshoot the issue based on the causes.
3. When the **verificateRequestParams** method is invoked, **authToken** and **signature** are inconsistent, and plus signs (+) become spaces. What should I do?

   Invoke **URLDecoder.decode()** to decrypt **authToken** and **signature**.
4. Why is the following error message displayed when I debug the service interface although the header contains both **sign_type** and **signature**?
   ```
   http header->sign_type is not equals HMAC-SHA256 or signature is empty.
   ```
   The values of **sign_type** and **signature** are not obtained from the **Body-Sign** header. Check whether the two values returned to KooGallery are in the correct format. The correct format is as follows:
   ```
   sign_type="HMAC-SHA256", signature="MkW2WiHUOpT4G/IFYV5kigY7djPRs3U5hOCe/EQrt8g="
   ```
   If the quotation marks ("") are missing, KooGallery cannot retrieve the two values and displays this error message.
5. Why can't a SaaS product customer view the username and password of the product when they click **Manage** on the **My KooGallery Apps** > **Purchased Apps** page?

   Possible causes are as follows:
   - When the instance is enabled, some parameters failed the verification. For example, the length of the password is incorrect. The length of the password ciphertext must not exceed the limit defined in this guide.
   - KooGallery failed to decrypt the username and password. Sellers must use the method provided by KooGallery to encrypt sensitive information. If a seller uses a different programming language, check whether the ciphertext generated using the current language is the same as the ciphertext generated using the code provided by KooGallery. If they are the same, check whether the **encryptType** parameter for encrypting and decrypting sensitive information is correctly transferred to KooGallery. If the algorithm is AES256_CBC_PKCS5Padding, the value must be **1**. If the algorithm is AES128_CBC_PKCS5Padding, the value must be **2**.

6. Why is the **authToken** generated by encrypting the request parameters provided by KooGallery different from the **authToken** provided by KooGallery?

   Parameters in a requested URL are URL-encoded and cannot be directly used to generate the **authToken**. Decode each parameter first, and use the decoded parameters to generate the **authToken**. Note that a customer's mobile number and email address are ciphertexts after being decoded and do not need to be decrypted again.

7. Does the restriction on the username and password lengths apply to the plaintexts or ciphertexts?

   It applies to the ciphertext (including the IV). Sometimes the instance of a product is successfully enabled but the username and password lengths fail the KooGallery verification. As a result, the status of the product is displayed as **Enabling** on the **My Orders** page. To prevent this issue, verify the lengths of the encrypted username and password.

8. After a customer purchases and pays a SaaS product, KooGallery still fails to enable the instance by invoking the subscription interface of the seller when the number of invocation times reaches the maximum. What do I enable the instance?

   Find the reason why the instance fails to be enabled and take measures to rectify the issue. Then, log in to the Seller Center, choose **Application Tools** > **Service Interface Messages** in the navigation pane, and click **Restart Debugging** on the right.

9. What do I do when I encounter a SaaS access problem that cannot be resolved?

   Send an email to **partner@huaweicloud.com**, describing the problem in detail with screenshots. The operations manager will check the email and will respond within two business days.

10. The example code is in Java. What can I do if I use another programming language?

    - Refer to the code provided in this guide to debug your own code.
    - In addition, contact the KooGallery operations team to obtain the example code using the programming language you use to develop instances.

11. After I debug the subscription interface on the **Application Access Debugging** page, do I need to do anything else that should be done?

    To ensure that subsequent service processes such as renewal, expiration, and release are normally performed, debug all of the subscription, renewal, expiration, and release interfaces before releasing a SaaS product to KooGallery.

12. When releasing a SaaS product, where can I specify the algorithm sensitive information encryption performed during application access debugging?

    On the SaaS product release page, after you select **User Authorization Required**, the algorithm for encrypting sensitive information is displayed. By default, **AES256_CBC_PKCS5Padding** is selected. You can choose another encryption algorithm as required.

13. What can I do if the subscription interface is successfully debugged but it cannot be invoked when a customer purchases a product?

a. Log in to the Seller Console, choose **Application Tools** > **Application Access Debugging** in the navigation pane. Use the actual request sent by KooGallery to the customer and the actual message body returned to KooGallery to debug the subscription interface on the **Application Access Debugging** page.

b. If the debugging is successful, choose **Application Tools** > **Service Interface Messages** in the navigation pane of the Seller Console and click **Restart Debugging** on the right of the corresponding record. Otherwise, contact the operations manager.

14. Can multiple production system API URLs be provided for different scenarios (subscription, renewal, expiration, and release) when a product is released?

    No. Currently, only one production system API URL can be configured for a product during release.

3 Automatic Deployment and Access Guide

3.1 Introduction

Huawei Cloud KooGallery allows automatic deployment for **images**.

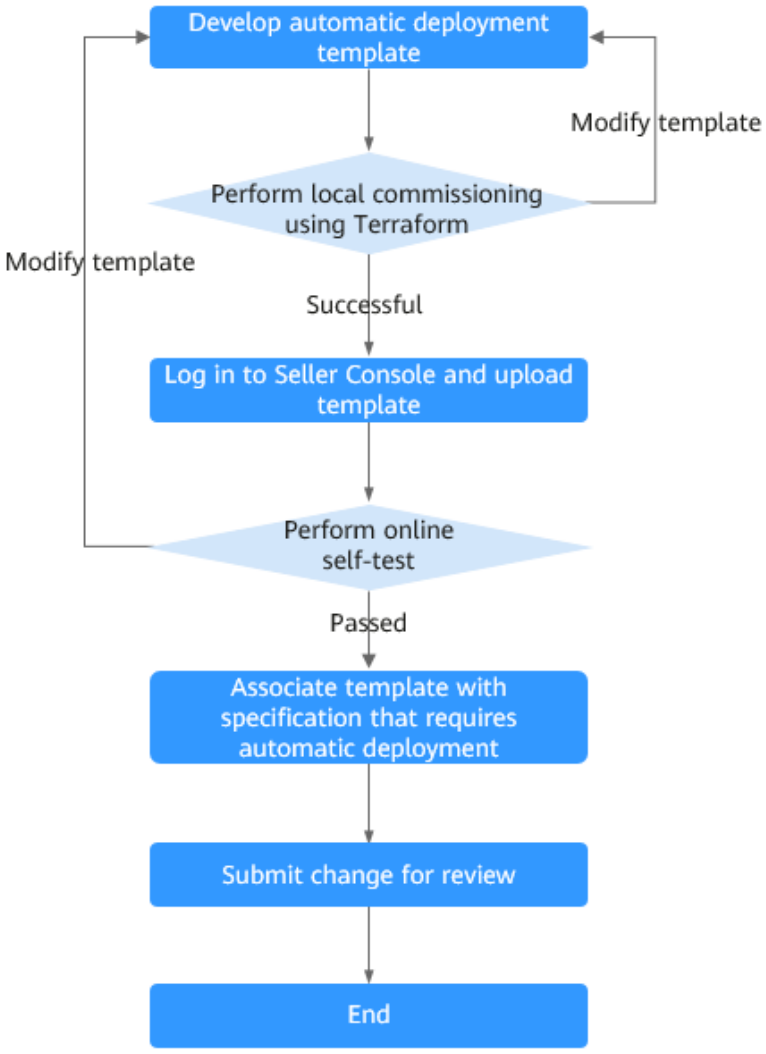
You can develop automatic deployment templates for applications released to KooGallery. After purchasing an application, customers can deploy it on the cloud with one click. Automatic deployment improves the purchase and deployment efficiency of customers and reduces delivery costs of sellers.

Prerequisites

You have become a seller of Huawei Cloud KooGallery. For details about how to register as a seller, see **Registration Process**.

3.2 Image Access Process

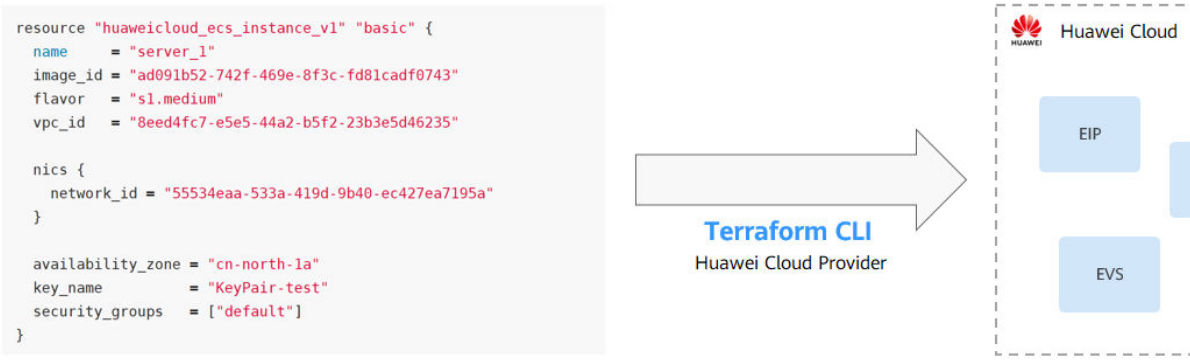
The following figure shows the access process of images with automatic deployment.



3.3 Automatic Deployment

3.3.1 Developing an Automatic Deployment Template

You can develop automatic deployment templates on Terraform. **Terraform** is an open-source automatic resource orchestration tool. The following figure shows the process of using Terraform to manage cloud resources. Before developing a template, install and configure Terraform by referring to **Getting Started with Terraform** and use Terraform to create a Huawei Cloud VPC. In addition, learn how to create resource stacks, create execution plans, and delete resource stacks using **Huawei Cloud Resource Formation Service (RFS)**.



Terraform allows you to describe an application or even an entire data center in configuration files. You can use Terraform to easily create, manage, delete, and version Huawei Cloud resources. For details about Huawei Cloud resources that can be orchestrated by Terraform, see **HuaweiCloud Provider**.

3.3.2 Testing an Automatic Deployment Template

Procedure

Step 1 Install Terraform and use environment variables to configure authentication information for Terraform. For details, see **Getting Started with Terraform**. If your device runs Windows, run the following commands to set environment variables:

```
set HW_REGION_NAME=cn-north-4 [C(1]
set HW_ACCESS_KEY=your ak
set HW_SECRET_KEY=your sk
```

Step 2 Open the command line interface (CLI), go to the template directory, run the following commands, and enter the configuration information as prompted:

```
terraform init [C(1]
terraform plan
terraform apply
```

Step 3 Log in to the Huawei Cloud console and view the created cloud service.

----End

NOTICE

The cloud resources created in this example are charged. If you do not need these resources, run the **terraform destroy** command to delete them in a timely manner.

3.3.3 Sample Code

For details about how to develop an automatic deployment template, see **HuaweiCloud Provider Documentation**.

3.4 Associating an Image Asset with an Automatic Deployment Template

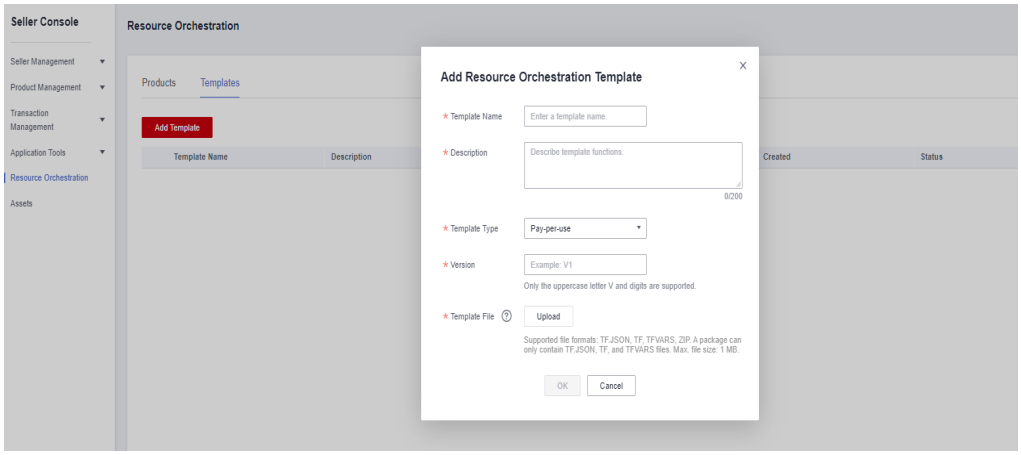
Prerequisites

An image product has been released. For details about how to create a private image and release an image product, see **Image Release Guide** of Huawei Cloud KooGallery.

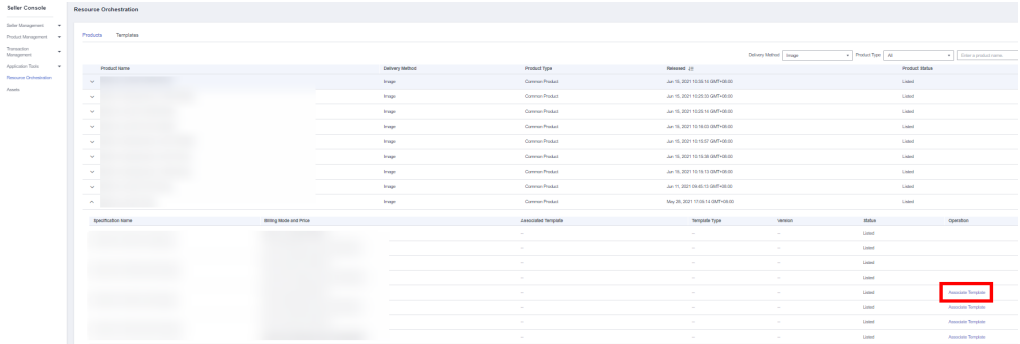
Procedure

Step 1 Log in to the **KooGallery homepage** using a Huawei Cloud account that has enabled resource orchestration and click **Seller Console** on the top to access the **Seller Console**.

Step 2 In the navigation pane, choose **Resource Orchestration**. Click the **Templates** tab and click **Add Template**. Enter the template name, description, and version number, and upload a template file. Click **OK** to create the automatic deployment template.



Step 3 On the **Resource Orchestration** page, click the **Products** tab, locate the specification to be associated with a resource template, and click **Associate Template** in the **Operation** column.



> **CAUTION**
>
> Different specifications of a product can be associated with different resource orchestration templates.

Step 4 Select the automatic deployment template to be associated and click **OK**.

----End

3.5 Releasing and Modifying a Product

For details, see **Image Release Guide**.

3.6 Automatically Deploying a Product

For details about how to automatically deploy images, see **Purchasing and Using an Image**.

