

**Access Guide**

# **Access Guide**

**Issue**            01  
**Date**             2024-03-05



**Copyright © Huawei Technologies Co., Ltd. 2024. All rights reserved.**

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

## **Trademarks and Permissions**



HUAWEI 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

---

<b>1 SaaS Product Access Guide.....</b>	<b>1</b>
1.1 Access Process.....	1
1.2 Interface Functions.....	2
1.3 Preparations.....	4
1.3.1 Obtaining the Key.....	4
1.3.2 authToken Value.....	5
1.3.3 HTTP Body Signature.....	5
1.4 Interface Description.....	6
1.4.1 Subscription.....	6
1.4.2 Renewal.....	20
1.4.3 Expiration.....	25
1.4.4 Resource Release.....	28
1.4.5 Upgrade.....	30
1.4.6 Resource Status Change.....	34
1.4.7 Usage Push.....	37
1.5 Invocation Result Codes.....	42
1.6 Interface Debugging.....	42
1.7 Code Example (Java).....	45
1.7.1 ISV Server Verifying Requests.....	45
1.7.2 ISV Server Signing a Response Message Body.....	48
1.7.3 ISV Server Encrypting the Username and Password After Resource Enabling.....	49
1.7.4 ISV Server Decrypting the Mobile Number and Email Address.....	53
1.7.5 Java Code Example.....	57
1.8 FAQs.....	66
<b>2 Automatic Deployment Access Guide.....</b>	<b>69</b>
2.1 Introduction.....	69
2.2 Image Access Process.....	69
2.3 Automatic Deployment.....	70
2.3.1 Developing an Automatic Deployment Template.....	70
2.3.2 Testing an Automatic Deployment Template.....	71
2.3.3 Sample Code.....	71
2.4 Associating an Image Asset with an Automatic Deployment Template.....	72
2.5 Releasing and Modifying a Product.....	73

---

2.6 Automatically Deploying a Product..... 73

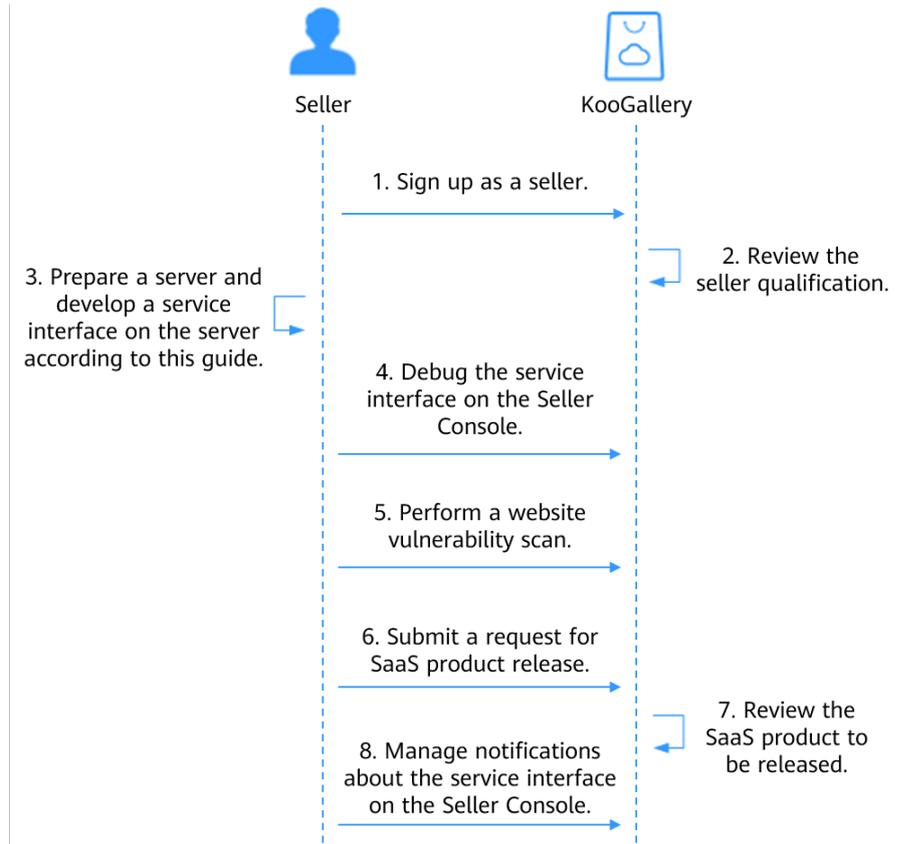
# 1 SaaS Product Access Guide

---

- [1.1 Access Process](#)
- [1.2 Interface Functions](#)
- [1.3 Preparations](#)
- [1.4 Interface Description](#)
- [1.5 Invocation Result Codes](#)
- [1.6 Interface Debugging](#)
- [1.7 Code Example \(Java\)](#)
- [1.8 FAQs](#)

## 1.1 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**.

## 1.2 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, the ISV server needs to return a success response and **the information about the successfully created application instance, rather than create a new 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 to call it 60 times (once every 3 minutes).

If the interface exception is resolved, the next call will be successful and the order is placed successfully. Otherwise, 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 to call it 60 times (once every minute).

If the interface exception is resolved, the next call will be successful and the order is placed successfully. Otherwise, KooGallery determines that the order fails to be placed. Locate and rectify the exception and **send an email to partner@huaweicloud.com to notify KooGallery to call the interface again**.

- In product expiration and resource release scenarios, if the service interface fails to respond, KooGallery will retry to call it 60 times (once every minute).

If the interface exception is resolved, the next call will be successful and the order is placed successfully. Otherwise, KooGallery determines that the order fails to be placed. Locate and rectify the exception. **Then log in to the Seller Console, choose Application Tools > Service Interface Messages in the navigation pane, and click Restart Debugging in the Operation column containing the order that fails to be placed to call the interface again**.

If the product can still be used after it expires due to the interface response failure, you shall bear the loss incurred therefrom.

### 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.

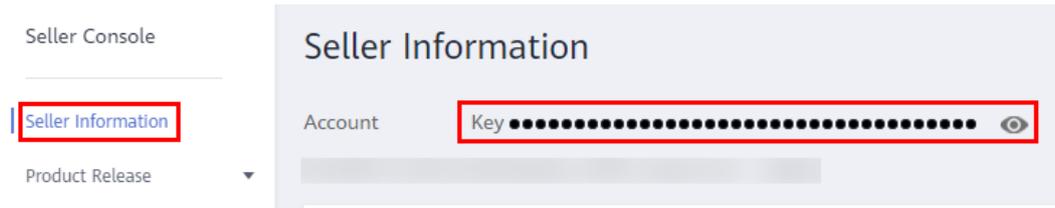
## 1.3 Preparations

### 1.3.1 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

## 1.3.2 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=xxxxxxxxxxxx&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 [1.7.1 ISV Server Verifying Requests](#).

## 1.3.3 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)) <ul style="list-style-type: none"> <li>• <b>key</b>: Key value</li> <li>• <b>httpBody</b>: The entire HTTP body, including the starting and ending spaces and tab characters</li> </ul>

Example of an HTTP response header:

**Body-Sign:** sign\_type="HMAC-SHA256", signature="abcd4567ed03sdfdsdfasdfsdfgdsdfhfgjgkghjllhjdkl"

 **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 [1.7.2 ISV Server Signing a Response Message Body](#).

## 1.4 Interface Description

### 1.4.1 Subscription

#### Description

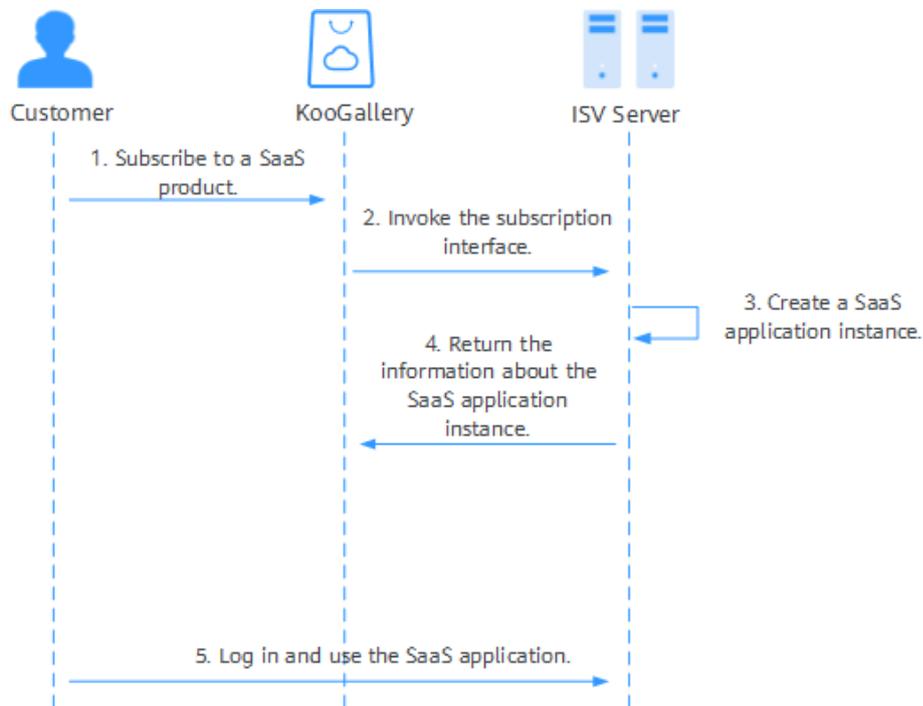
- After a customer purchases a product and pays for it successfully, KooGallery calls this interface to send you a request containing information about the product and customer. When receiving the request, the ISV server executes product subscription and informs KooGallery about the subscription result.
- A unique instance ID (**instanceld**) should be returned for the order. The instance IDs (**instanceld**) of different subscription orders must be different. You are advised to use the **businessId** provided by KooGallery to ensure that the **instanceld** is globally unique.
- 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 to call it 60 times (once every three minutes). If the interface exception is resolved, the next call will be successful and the order is placed successfully. Otherwise, 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, resolve 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, the ISV server must ensure idempotence.**  
KooGallery may resend requests for **a single order**. If receiving a duplicate order, the ISV server needs to return a success response and **the information about the successfully created application instance, rather than create a new SaaS instance**.

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 sellers. You need to provide services based on requests.

**Request method: GET**

Parameter	Mandatory	Type	Maximum Length of Characters	Description
authToken	Yes	String	50	Security verification token. For details, see <a href="#">1.3.2 authToken Value</a> .
timeStamp	Yes	String	20	Time (UTC time) when a request is initiated. Format: yyyyMMddHHmmssSSS
activity	Yes	String	20	Interface request ID, which is used to distinguish interface request scenarios. For new subscriptions, the value is <b>newInstance</b> .
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. <b>If this parameter is required, select To create an account based on IAM username for User Authorization Required when releasing the product.</b>
userName	No	String	64	Username of the customer used to log in to the system as an IAM user. Optional. <b>If this parameter is required, select To create an account based on IAM username for User Authorization Required when releasing the product.</b>

Parameter	Mandatory	Type	Maximum Length of Characters	Description
mobilePhone	No	String	256	<p>Customer's mobile number.</p> <p>Optional. <b>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.</b></p> <p>The mobile number encryption rules are as follows:</p> <p>The value consists of a 16-bit encryption initialization vector (IV) and a Base-encoded mobile number ciphertext, as follows:</p> <ul style="list-style-type: none"> <li>• iv +base64(AES_CBC(accessKey,mobilePhone))</li> <li>• The number of digits to be encrypted is specified by the seller when releasing the product.</li> </ul> <p>For an example of the mobile number decryption code, see <a href="#">1.7.4 ISV Server Decrypting the Mobile Number and Email Address</a>.</p> <p><b>NOTE</b> This parameter does not contain the country code. If a customer does not bind the mobile number, the parameter cannot be obtained.</p>

Parameter	Mandatory	Type	Maximum Length of Characters	Description
email	No	String	256	<p>Customer's email address.</p> <p>Optional. <b>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.</b></p> <p>The email address encryption rules are as follows:</p> <p>The value consists of a 16-bit encryption IV and a Base-encoded email ciphertext, as follows:</p> <ul style="list-style-type: none"> <li>iv +base64(AES_CBC(accessKey,email))</li> <li>The number of digits to be encrypted is specified by the seller when releasing the product.</li> </ul> <p>For an example of the email address decryption code, see <a href="#">1.7.4 ISV Server Decrypting the Mobile Number and Email Address</a>.</p>
businessId	Yes	String	64	<p>KooGallery business ID.</p> <p>The value of <b>businessId</b> is different for each request.</p>
orderId	Yes	String	64	<p>KooGallery order ID.</p>

Parameter	Mandatory	Type	Maximum Length of Characters	Description
skuCode	No	String	64	<p>Product 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, the <b>productId</b> corresponding to the <b>instanceId</b> of the instance enabled by the customer changes, but the value of <b>skuCode</b> does not change.</p> <p><b>NOTE</b> After a product is approved and successfully released to KooGallery, you can obtain this parameter in the Seller Console. On the <b>Product Management &gt; My Products</b> page, locate the product and click <b>Details</b> in the <b>Operation</b> column. The parameter can be obtained on the displayed page.</p>
productId	Yes	String	64	<p>Product ID. The value of <b>productId</b> varies between products of the same <b>skuCode</b> depending on the billing mode.</p> <p>For example, when you release a product and add a new specification, an <b>skuCode</b> value is generated. After yearly and monthly billing prices are configured, two <b>productId</b> values are generated.</p> <p><b>NOTE</b> After a product is approved and successfully released to KooGallery, you can obtain this parameter in the Seller Console. On the <b>Product Management &gt; My Products</b> page, locate the product and click <b>Details</b> in the <b>Operation</b> column. The parameter can be obtained on the displayed page.</p>
testFlag	No	String	2	<p>Whether a request is submitted for debugging.</p> <ul style="list-style-type: none"> <li>● <b>1</b>: debugging request.</li> <li>● <b>0</b>: non-debugging request.</li> </ul> <p>The default value is <b>0</b>.</p>

Parameter	Mandatory	Type	Maximum Length of Characters	Description
trialFlag	No	String	2	Whether an instance is created for a trial. <ul style="list-style-type: none"> <li>• <b>0</b>: no.</li> <li>• <b>1</b>: yes.</li> <li>• <b>N/A</b></li> </ul>
expireTime	No	DateTime	20	Expiration time. Format: yyyyMMddHHmmss <b>NOTE</b> <ul style="list-style-type: none"> <li>• This parameter is requested if the product is billed by yearly/monthly.</li> <li>• This parameter is not requested if the product is billed by one-time payment.</li> <li>• 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.</li> </ul>
chargingMode	No	Integer	[3]	Billing mode. <b>0</b> : pay-per-use. <b>1</b> : yearly/monthly/daily. <b>3</b> : one-time payment.

Parameter	Mandatory	Type	Maximum Length of Characters	Description
saasExtendParams	No	String	2048	<p>Extension parameters. These parameters are optional.</p> <p>The extension parameters are a JSON string carried in the <b>url</b> parameter in the form of <b>urlEncode(base64(saasExtendParams))</b>. After obtaining the value of the <b>saasExtendParams</b> parameter, the ISV server needs to use <b>base64Decode(urlDecode(saasExtendParams))</b> to obtain the JSON string of the extension parameters.</p> <p>For example, <b>emailDomainName</b> and <b>extendParamName</b> in the JSON string <b>[{"name":"emailDomainName","value":"test.xxxx.com"}, {"name":"extendParamName","value":"extendParamValue"}]</b> are the parameter values set during product release.</p>
amount	No	Integer	4	<p>Product attribute of the quantity type. This parameter is optional.</p> <p>Attribute name: quantity (customizable)</p> <p>Unit: none</p> <p><b>NOTE</b></p> <p>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.</p> <p>Example: 30 users</p>

Parameter	Mandatory	Type	Maximum Length of Characters	Description
diskSize	No	Integer	4	Product attribute of the quantity type. Optional. Attribute name: disk size (customizable) Unit: GB <b>NOTE</b> 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 <b>NOTE</b> 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
periodType	No	String	10	Period type. <b>NOTE</b> This parameter is only required for yearly/monthly product subscriptions (the value of <b>chargingMode</b> is set to 1). Yearly subscription: <b>year</b> Monthly subscription: <b>month</b> Daily subscription: <b>day</b> If <b>chargingMode</b> is set to 3, do not transfer this parameter.
periodNumber	No	Integer	5	Number of periods. <b>NOTE</b> This parameter is only required for yearly/monthly product subscriptions (the value of <b>chargingMode</b> is set to 1). Enter a positive integer, for example, 1, 2, and 3.

Parameter	Mandatory	Type	Maximum Length of Characters	Description
orderAmount	No	bigdecimal	20	Order amount. <b>NOTE</b> 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 <b>0</b> and can contain a maximum of three decimal places. Unit: USD
provisionType	No	Integer	2	Instance provisioning mode. <b>NOTE</b> <ul style="list-style-type: none"> <li>• <b>Provision upon subscription</b> (By default, KooGallery calls the <b>newInstance</b> interface in polling mode.)</li> <li>• <b>Provision after acceptance</b> (The SaaS product involves service supervision.) <ol style="list-style-type: none"> <li>1. When a customer purchases the product, KooGallery calls the subscription interface, and you need to return the result code <b>000004</b> (request being processed) or <b>000000</b> (order created successfully).</li> <li>2. After you confirm to deliver the product, KooGallery calls the subscription interface, and you need to return the result code <b>000000</b>.</li> <li>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 <b>000000</b>.</li> </ol> </li> </ul>
acceptanceTime	No	String	20	Acceptance time. <b>NOTE</b> The value is the time when billing for the product starts. If <b>provisionType</b> is set to <b>Provision after acceptance</b> , this parameter is required. Format: yyyyMMddHHmmssSSS

Parameter	Mandatory	Type	Maximum Length of Characters	Description
startTime	No	String	20	Start time. Format: yyyyMMddHHmmss <b>NOTE</b> 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 [1.6 Interface Debugging](#).

Example request:

```
https://isvserver.com/produceAPI?activity=newInstance&businessId=03pf80c2bae96vc49b80b917bea776d7&customerId=3736bb8ad93b43fca8012c64a82cec25 &expireTime=20180725000000&orderId=HWS001014ED483AA1E8&productId=005a8781ef0c4a47a3dbfc4c1e72871e&saasExtendParams=W3sibmFtZSI6ImVtYWlsMTEiLCJ2YWx1ZSI6ImVtYWlsMTFlbWFpbDEuIn0seyJuYW1lIjoizW1haWwyaWlsInZhbHVlIjoizW1haWwyaWlsMjliIj0%3D&timeStamp=20170725025113409&testFlag=0&authToken=09ls5y+KCtxBu+ON4TXv1SrjH5KVYka9sx2MauHrQU=
```

## 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 <a href="#">1.5 Invocation Result Codes</a> .
resultMsg	No	String	255	Invocation result description.
encryptType	No	String	2	Algorithms for encrypting sensitive information. <b>1:</b> AES256_CBC_PKCS5Padding (default) <b>2:</b> AES128_CBC_PKCS5Padding <b>NOTE</b> If the value of this parameter is <b>AES256_CBC_PKCS5Padding</b> , <b>1</b> is returned; if the value is <b>AES128_CBC_PKCS5Padding</b> , <b>2</b> is returned.
instanceId	No	String	64	Instance ID, which is a unique ID provided by a seller. You are advised to use the <b>businessId</b> provided by KooGallery to ensure that the <b>instanceId</b> is globally unique. <b>NOTE</b> The <b>businessId</b> in each request sent by KooGallery is different. If you use the <b>businessId</b> as the <b>instanceId</b> , use the <b>businessId</b> in the first request sent by KooGallery.  If <b>instanceId</b> is generated in other ways, for example, using a universally unique identifier (UUID), ensure that it is globally unique. Identical values of <b>instanceId</b> will cause a failure of enabling a SaaS application instance.

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

The following table describes the data structure of **applInfo**.

Parameter	Mandatory	Type	Maximum Length of Characters	Description
frontEndUrl	Yes	String	512	<p>Frontend URL.</p> <p>URL of the website that the customer can access to use the subscribed product.</p>
adminUrl	No	String	512	<p>Management address.</p> <p>URL of the backend website that the customer can access to manage the subscribed product.</p>

Parameter	Mandatory	Type	Maximum Length of Characters	Description
userName	No	String	128	<p>Encrypted administrator account.</p> <p>The account (usually an email address or mobile number) used by the 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, as follows:</p> <ul style="list-style-type: none"> <li>iv +base64(AES_CBC(accessKey, userName))</li> <li>Encrypt the account using the <b>Key</b>. The encryption algorithm is specified by the <b>encryptType</b> parameter. For example code, see <a href="#">1.7.3 ISV Server Encrypting the Username and Password After Resource Enabling</a>.</li> </ul>
password	No	String	128	<p>Encrypted initial password of the administrator.</p> <p>The password (usually generated by the seller) used by the customer to access the administration backend of the seller after purchasing a product. The value consists of a 16-bit encryption IV and a Base-encoded password ciphertext, as follows:</p> <ul style="list-style-type: none"> <li>iv +base64(AES_CBC(accessKey, pwd))</li> <li>Encrypt the password using the <b>Key</b>. The encryption algorithm is specified by the <b>encryptType</b> parameter. For example code, see <a href="#">1.7.3 ISV Server Encrypting the Username and Password After Resource Enabling</a>.</li> </ul>

Parameter	Mandatory	Type	Maximum Length of Characters	Description
ip	No	String	64	IP address of the website.
memo	No	String	1024	Remarks.

#### NOTE

- For details about how to obtain the **accessKey**, see [1.3.1 Obtaining the Key](#).
- The lengths of the username and password ciphertexts are verified, which include the IVs.
- When processing an interface request, the ISV server must ensure idempotence. KooGallery may resend requests for a single order. If receiving a duplicate order, the ISV server needs to return a success response and the information about the successfully created application instance, rather than create a new SaaS instance.
- If a SaaS instance information (for example, the **adminUrl**) changes, invoke the interface again in KooGallery. When the same **orderId** is provided by KooGallery, the ISV 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=="
  }
}
```

## 1.4.2 Renewal

### Description

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

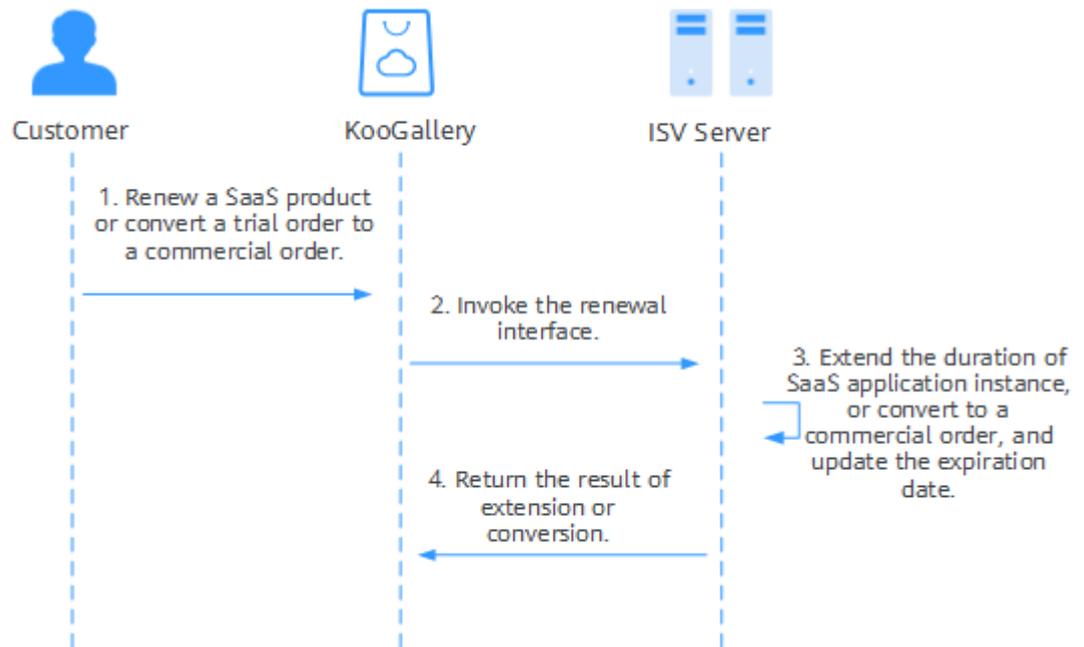
- After a customer places an order for renewal or converts a trial order to a commercial order, KooGallery calls the interface to request you to extend the service. The service interface then updates the expiration date and informs KooGallery about the update result.
- Ensure that the communication over 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 call the interface again. You can view the interface exception information on the [Application Tools > Service Interface Messages](#) page. After the exception is solved, notify 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, resolve 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 of Characters	Description
activity	Yes	String	20	Interface request ID, which is used to distinguish interface request scenarios. For renewals, the value is <b>refreshInstance</b> .

Parameter	Mandatory	Type	Maximum Length of Characters	Description
orderId	Yes	String	64	KooGallery order ID. <b>NOTE</b> A new order will be generated during the renewal and has an ID different from that of a subscription order. Use <b>instanceId</b> 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 <b>productId</b> is provided.
expireTime	Yes	String	20	Expiration time. Format: yyyyMMddHHmmss
testFlag	No	String	2	Whether a request is submitted for debugging. <ul style="list-style-type: none"> <li>• <b>1</b>: debugging request.</li> <li>• <b>0</b>: non-debugging request.</li> </ul> The default value is <b>0</b> .

Parameter	Mandator y	Type	Maximum Length of Characters	Description
trialToFormal	No	String	2	<p>Whether a request is submitted to convert a trial product to a commercial product.</p> <ul style="list-style-type: none"> <li>Parameter not passed: no</li> <li>1: yes</li> </ul> <p>By default, a request is not submitted to convert a trial product to a commercial product.</p> <p>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.</p>
authToken	Yes	String	50	<p>Security verification token.</p> <p>For details about the values, see <a href="#">1.3.2 authToken Value</a>.</p>
timeStamp	Yes	String	20	<p>Time (UTC time) when a request is initiated.</p> <p>Format: yyyyMMddHHmmssSSS</p>
periodType	No	String	10	<p>Period type.</p> <p><b>NOTE</b> This parameter is only required for yearly/ monthly product subscriptions (the value of <b>chargingMode</b> is set to <b>1</b>).</p> <p>Yearly subscription: <b>year</b> Monthly subscription: <b>month</b></p>

Parameter	Mandator y	Type	Maximum Length of Characters	Description
periodNumber	No	integer	2	Number of periods. <b>NOTE</b> This parameter is only required for yearly/ monthly product subscriptions (the value of <b>chargingMode</b> is set to <b>1</b> ). Enter a positive integer, for example, <b>1</b> , <b>2</b> , and <b>3</b> .
orderAmount	No	bigdecimal	20	Order amount. <b>NOTE</b> The amount is the actual payment amount, which you can check during reconciliation. The amount is greater than or equal to <b>0</b> and can contain a maximum of three decimal places. Unit: USD

Example request:

```
https://isvserver.com/produceAPI?activity=refreshInstance&
expireTime=20180725000000&instanceld=03pf80c2bae96vc49b80b917bea776d7&orderId=HWS001014ED48
3AA1E8&timeStamp=20170725025113409&testFlag=0&authToken=09ls55y+KCtxBu
+ON4TXv1SrjH5KVYka9sx2MauHrQU=
```

## Response Message

The following table describes the response parameters.

Parameter	Mandator y	Type	Maximum Length of Characters	Description
resultCode	Yes	String	6	Invocation result code. For details, see <a href="#">1.5 Invocation Result Codes</a> .
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, the ISV server needs to return a success response, rather than extend the SaaS instance again.

Example response:

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

## 1.4.3 Expiration

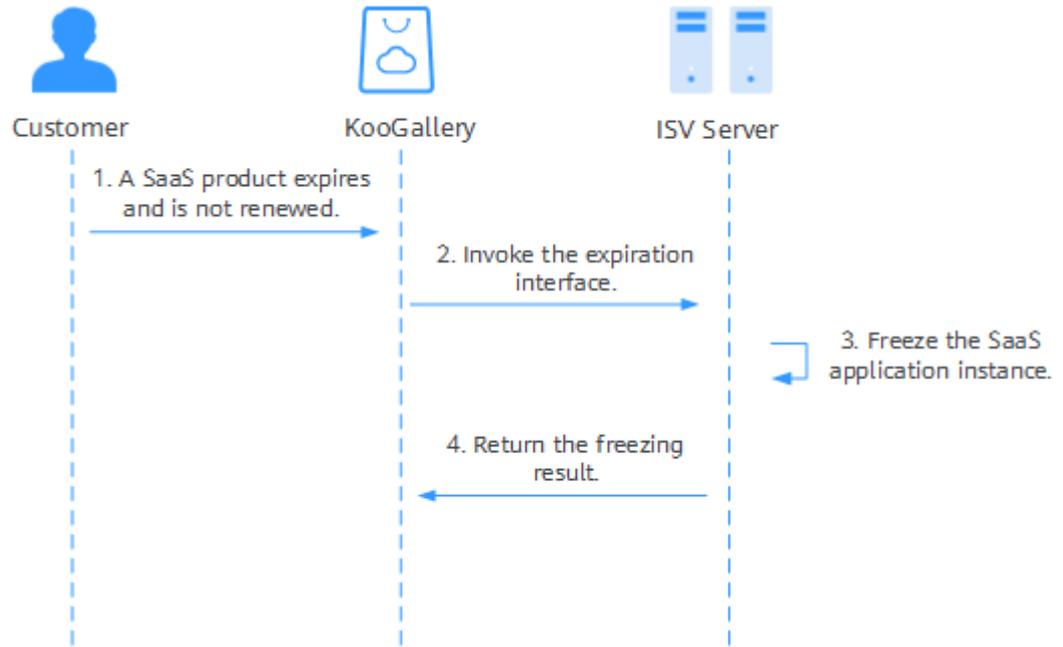
### Description

- 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 to call it 60 times (once every minute). You can view the interface exception information on the [Application Tools > Service Interface Messages](#) page. If the interface exception is resolved, the next call will be successful. Otherwise, KooGallery stops calling the interface. After the exception is solved, go to the Seller Console, locate the order on the [Application Tools > Service Interface Messages](#) page, and click **Restart Debugging** in the **Operation** column 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, resolve 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 of Characters	Description
activity	Yes	String	20	Interface request ID, which is used to distinguish interface request scenarios. For product expiration, the value is <b>expireInstance</b> .
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. <ul style="list-style-type: none"> <li>• <b>1</b>: debugging request.</li> <li>• <b>0</b>: non-debugging request.</li> </ul> The default value is <b>0</b> .

Parameter	Mandatory	Type	Maximum Length of Characters	Description
authToken	Yes	String	50	Security verification token. For the values, see <a href="#">1.3.2 authToken Value</a> .
timeStamp	Yes	String	20	Time (UTC time) when a request is initiated. Format: yyyyMMddHHmmssSSS

Example request:

```
https://isvserver.com/produceAPI?activity=expireInstance&instanceId=03pf80c2bae96vc49b80b917bea776d7
&timeStamp=20170725025113409&testFlag=0&authToken=09lsS5y+KCtxBu
+ON4TXv1SrhH5KVYka9sx2MauHrQU=
```

## 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 <a href="#">1.5 Invocation Result Codes</a> .
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 **instanceId** value, the ISV server needs to return a success response, rather than freeze the instance again.

Example response:

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

## 1.4.4 Resource Release

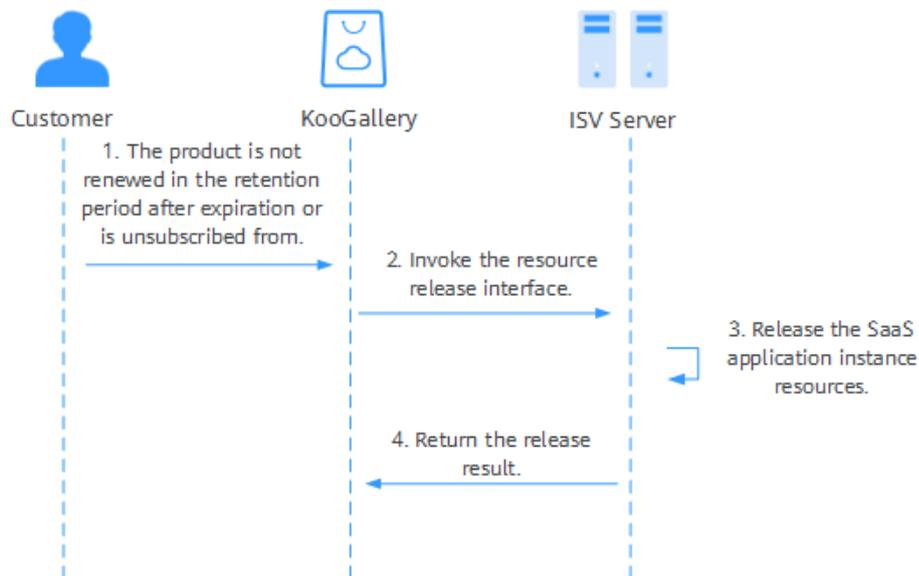
### Description

- 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 to call it 60 times (once every minute). You can view the interface exception information on the [Application Tools > Service Interface Messages](#) page. If the interface exception is resolved, the next call will be successful. Otherwise, KooGallery stops calling the interface. After the exception is solved, go to the Seller Console, locate the order on the [Application Tools > Service Interface Messages](#) page and click **Restart Debugging** in the **Operation** column 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, resolve 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	Mandator y	Type	Maximum Length of Characters	Description
activity	Yes	String	20	Interface request ID, which is used to distinguish interface request scenarios. For resource release, the value is <b>releaseInstance</b> .
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. <ul style="list-style-type: none"> <li>• <b>1</b>: debugging request.</li> <li>• <b>0</b>: non-debugging request.</li> </ul> The default value is <b>0</b> .
authToken	Yes	String	50	Security verification token. For details about the values, see <a href="#">1.3.2 authToken Value</a> .
timeStamp	Yes	String	20	Time (UTC time) when a request is initiated. Format: yyyyMMddHHmmssSSS
orderAmount	No	bigdecimal	20	Order amount. <b>NOTE</b> The amount is the actual payment amount, which you can check during reconciliation. The amount is greater than or equal to <b>0</b> 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=09ls5y+KCtxBu
+ON4TXv1SrjH5KVYka9sx2MauHrQU=
```

## 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 <a href="#">1.5 Invocation Result Codes</a> .
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 **instanceld** value, the ISV server needs to return a success response, rather than release the instance again.

Example response:

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

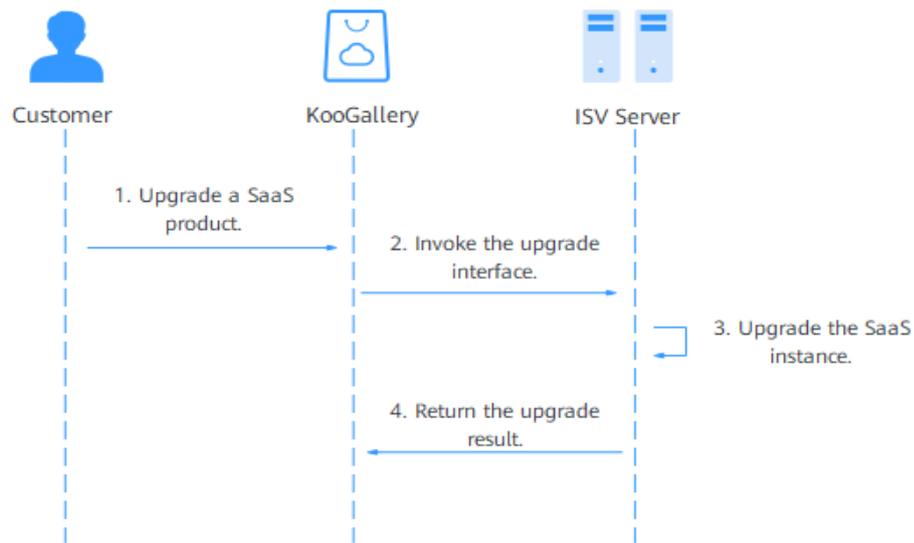
## 1.4.5 Upgrade

### 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 <a href="#">1.3.2 authToken Value</a> .
activity	Yes	String	20	Interface request ID, which is used to distinguish interface request scenarios. For upgrades, the value is <b>upgrade</b> .
instanceId	Yes	String	64	Instance ID. <b>NOTE</b> The upgrade does not change <b>instance ID</b> .
testFlag	No	String	2	Whether a request is submitted for debugging. <ul style="list-style-type: none"> <li>• <b>1</b>: debugging request.</li> <li>• <b>0</b>: non-debugging request.</li> </ul> The default value is <b>0</b> .

Parameter	Mandatory	Type	Maximum Length of Characters	Description
orderId	Yes	String	64	Upgrade order ID. <b>NOTE</b> A new order will be generated during the upgrade and has an ID different from that of a subscription order. Use <b>instanceId</b> to identify the resources.
skuCode	Yes	String	64	Product specification ID after the upgrade. <b>NOTE</b> A specification with custom attributes will change if the customer selects other attribute values during the upgrade. As a result, the <b>skuCode</b> changes.  If the customer only expands the capacity by linearly increasing the attribute value, for example, from 10 users to 20 users, the <b>skuCode</b> does not change.
productId	Yes	String	64	Product ID after the upgrade. The value of <b>productId</b> varies according to the <b>skuCode</b> . If the customer only expands the capacity, the value of <b>productId</b> does not change.
timeStamp	Yes	String	20	Time (UTC time) when a request is initiated. Format: yyyyMMddHHmmssSSS

Parameter	Mandatory	Type	Maximum Length of Characters	Description
amount	No	Integer	4	Product attribute of the quantity type. This parameter is optional. Attribute name: quantity (customizable) Unit: none <b>NOTE</b> 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
diskSize	No	Integer	4	Product attribute of the quantity type. This parameter is optional. Attribute name: disk size (customizable) Unit: GB <b>NOTE</b> 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 <b>NOTE</b> 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=W3sibmFtZSI6ImkTnVtliwidmFsdWUiOiIzNTU1NTU1NTU2NTYifS
x7Im5hbWUiOiI1c2VyTmFtZSI6ImkTnVtliwidmFsdWUiOiIzNTU1NTU1NTU2NTYifS
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 <a href="#">1.5 Invocation Result Codes</a> .
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."
}
```

## 1.4.6 Resource Status Change

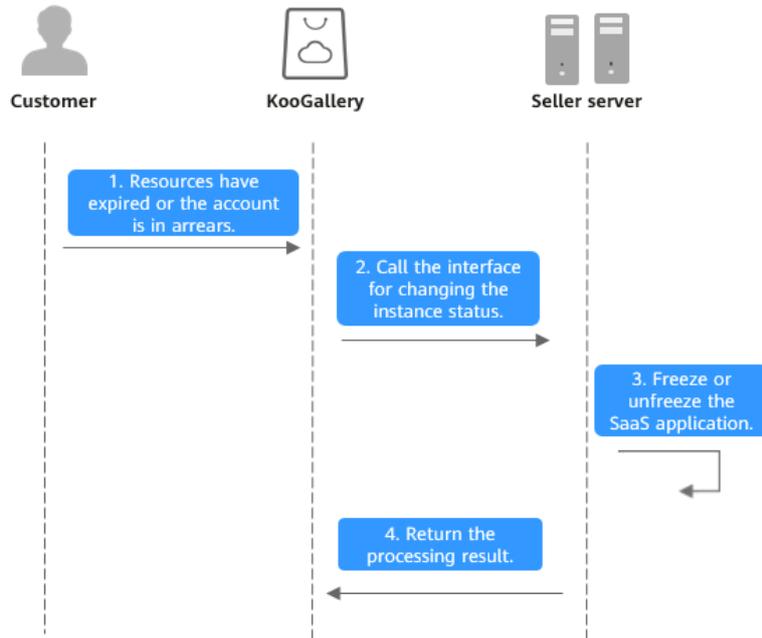
### Description

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 <a href="#">1.3.2 authToken Value</a> .
activity	Yes	String	32	Interface request ID, which is used to distinguish interface request scenarios. For resource status changes, the value is <b>instanceStatus</b> .

Parameter	Mandatory	Type	Maximum Length of Characters	Description
instanceId	Yes	String	64	Instance ID. <b>CAUTION</b> Use the instance ID returned by the pay-per-use billing interface.
instanceStatus	Yes	String	32	New status. <ul style="list-style-type: none"> <li>• <b>FREEZE</b>: frozen.</li> <li>• <b>NORMAL</b>: unfrozen</li> </ul>
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. <ul style="list-style-type: none"> <li>• <b>1</b>: debugging request.</li> <li>• <b>0</b>: non-debugging request.</li> </ul> The default value is <b>0</b> .

Example request:

Freezing an instance: `https://example.isv.com?activity=instanceStatus&instanceId=huaweitest123456&instanceStatus=FREEZE&testFlag=1&timestamp=20230327070251713&authToken=pqlrW7%2BPHC%2F1JE%2BMEjKxC94GGJreoS6PZHd982auw2o%3D`  
 Unfreezing an instance: `https://example.isv.com?activity=instanceStatus&instanceId=huaweitest123456&instanceStatus=NORMAL&testFlag=1&timestamp=20230327070251713&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 <a href="#">1.5 Invocation Result Codes</a> .
resultMsg	No	String	255	Invocation result description.

Example response:

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

## 1.4.7 Usage Push

### Description

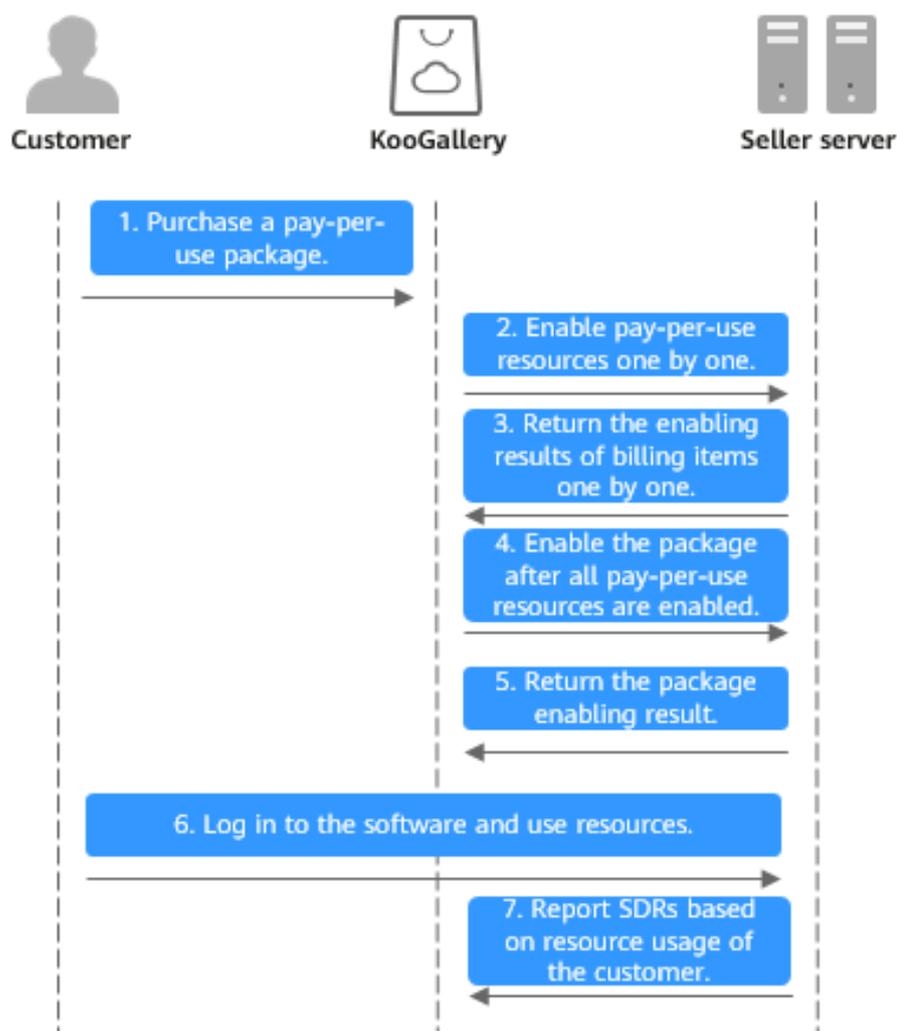
After a customer subscribes to and uses pay-per-use resources in KooGallery, call this interface to upload the service detail records (SDRs) of the customer. After obtaining the SDRs, KooGallery charges the customer for the usage.

#### 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 /rest/marketplace/v1/isv/usage-data](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 of Characters	Description
usage_records	Yes	List<Usage PushData>	1000	Set of service usage records. A set contains up to 1000 <b>UsagePushData</b> records.

**Table 1-1 UsagePushData**

Parameter	Mandatory	Type	Maximum Length of Characters	Description
instance_id	Yes	String	64	Instance ID. Use the instance ID returned by the pay-per-use billing 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.

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 real 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 billing interface instead of that returned by the pay-per-use package billing interface.

## Response Message

The following table describes the response parameters.

Parameter	Mandatory	Type	Maximum Length of Characters	Description
error_code	Yes	String	6	Invocation result code. For details, see the following error codes.
error_msg	No	String	255	Invocation result description.

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 <b>instance_id</b> 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 entity.
500	MKT.9003	Usage records extend size limit.	Too many records. Max. records: 1000.
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"
}
```

## 1.5 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.

## 1.6 Interface Debugging

To ensure that SaaS products can be accessed successfully, an application access debugging page is set up on the Seller Console for you to debug interfaces. On this page, you can verify the correctness of a SaaS interface during 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.

1. Develop interfaces according to the [Product Access Guide](#), and then debug the interfaces on this page.
2. Select the message production type and enter all required parameters. The preset parameter values are for reference only. Change them as required.
3. Click Generate Link Address after entering all required parameters.
4. After you click Debug and Save Case, the system invokes the production system link to debug the interface. If the debugging is successful, the case is saved. If the debugging fails, the error information is displayed in the lower part of the page as a reference for debugging.
5. Cases can be saved and updated only after being debugged successfully. You can manage cases on the [Case Management](#) page.

[More](#)

Subscription
Renewal
Expiration
Release
Upgrade

Parameter Description	Parameter Name	Parameter Value
* Interface address	URL	<input type="text" value="The same production system address must be specified for the subscription, renewal, expiration, release, and upgrade interfaces."/>
* Customer ID	customerid	<input type="text"/>
* Business ID	businessid	<input type="text"/>
IAM username	userName	<input type="text"/>
IAM user ID	userid	<input type="text"/>
* Billing mode	chargingMode	<input checked="" type="radio"/> Yearly/Monthly <input type="radio"/> One-time
* Product ID	productid	<input type="text"/>
* Expiration time	expireTime	<input type="text" value=""/> <span style="float: right; font-size: 0.7em;">X   🗄</span>
* Order ID	orderid	<input type="text"/>
Mobile number	mobilePhone	<input type="text"/>
Email address	email	<input type="text"/>
Customer name	customerName	<input type="text"/>
* Encryption algorithm	encryptType	<input checked="" type="radio"/> AES256_CBC_PKCS5Padding <input type="radio"/> AES128_CBC_PKCS5Padding
Enable trial instance	trialFlag	<input type="radio"/> 0 (No) <input type="radio"/> 1 (Yes) <input type="radio"/> N/A
Specification ID	skuCode	<input type="text"/>
Product Attributes	<span style="border: 1px solid #ccc; padding: 2px 5px;">TTB</span> <span style="font-size: 0.7em;">⊕</span>	amount <input type="text"/>

[Add Extension Parameter](#)

Generate Link Address

Debug and Save Case

**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, the ISV 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 invokes the production system link to debug the interface. If the interface is debugged successfully, go to **Step 6**. If not, the error messages are displayed in the lower part of the page. You can modify the interface parameters based on the error messages.

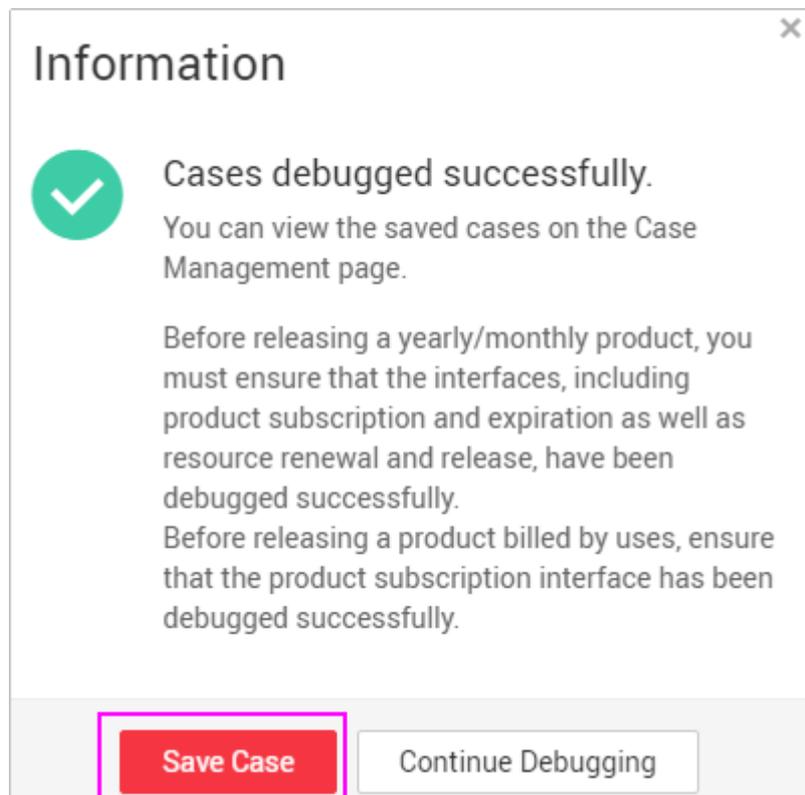
 **NOTE**

- 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.

For details about case management, see [Application Access Debugging and Case Management for SaaS Products](#).

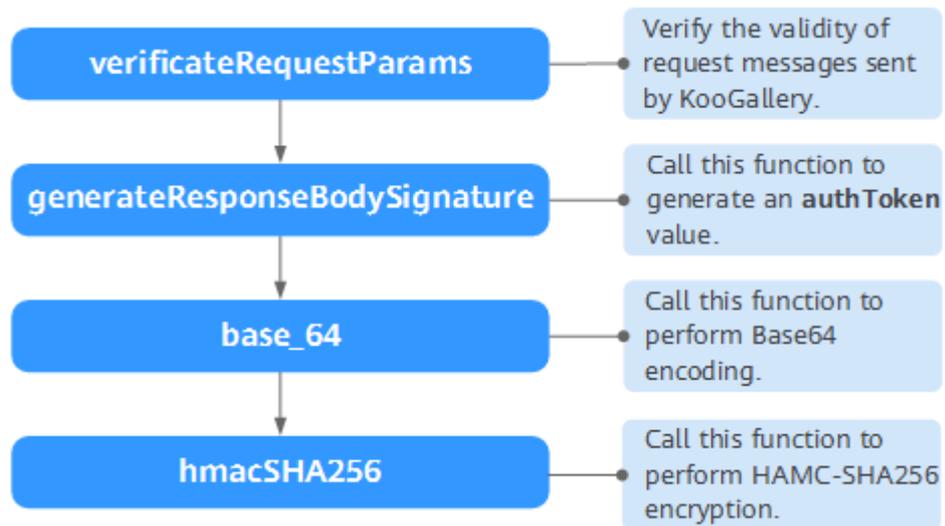


----End

## 1.7 Code Example (Java)

### 1.7.1 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 verificateRequestParams(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));
}
```

```
/**
 *
 * HMAC-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;
}
```

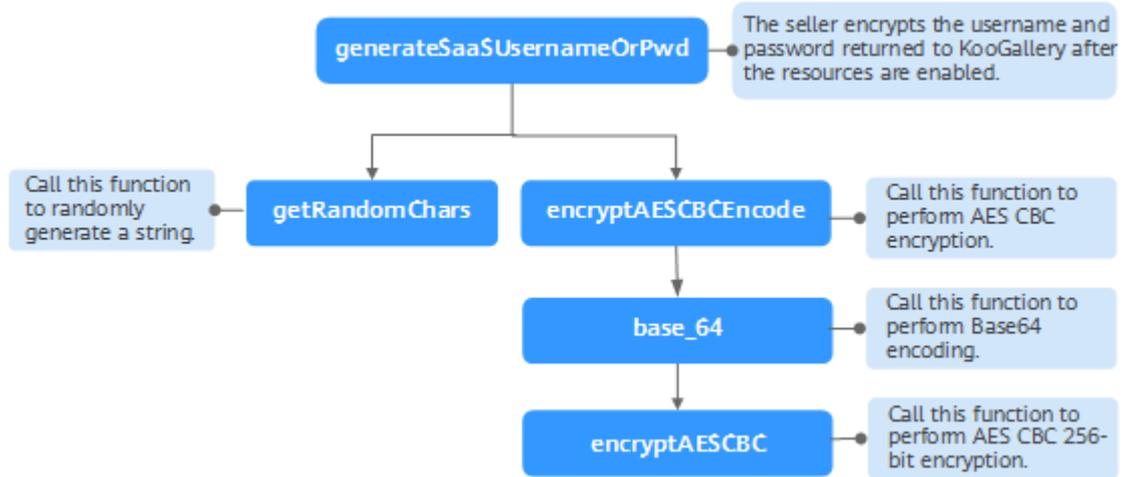
```
/**
 *
 * Convert the byte array into a string.
 * @param bytes --Byte array
 * @return --String
 */
public static String base_64(byte[] bytes)
{
    return new String(Base64.encodeBase64(bytes));
}
```

## 1.7.2 ISV Server Signing a Response Message Body

For the example code, see the `generateResponseBodySignature` method in [1.7.1 ISV Server Verifying Requests](#).

### 1.7.3 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;
}
```

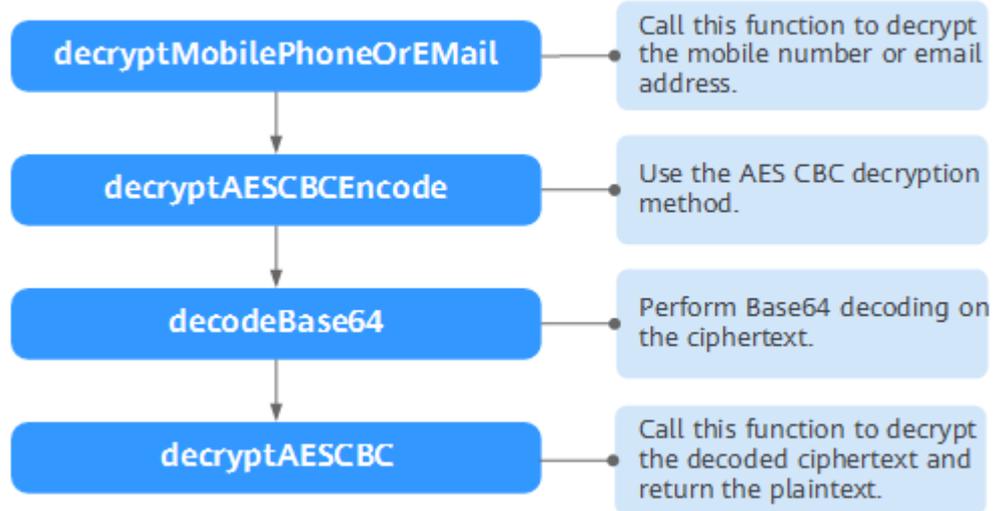
```
/**
 * Randomly generate a string.
 * @param length --Length of the randomly generated string
 * @return --Random string
 */
public static String getRandomChars(int length)
{
    String randomChars = "";
    SecureRandom random = new SecureRandom();
    for (int i = 0; i < length; i++)
    {
        // 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;
}
```

## 1.7.4 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, InvalidAlgorithmParameterEx-
ception, 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;  
}
```

## 1.7.5 Java Code Example

```
package com.huawei.cbc.cbcmarketplacecommentsservice.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=Gzbfjf9LHRBcl3bFVi+
+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[] {"Gzbfjf9LHRBcl3bFVi+
+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;
}
}
```

## 1.8 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 [1.3.3 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 do to 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.

# 2 Automatic Deployment Access Guide

---

[2.1 Introduction](#)

[2.2 Image Access Process](#)

[2.3 Automatic Deployment](#)

[2.4 Associating an Image Asset with an Automatic Deployment Template](#)

[2.5 Releasing and Modifying a Product](#)

[2.6 Automatically Deploying a Product](#)

## 2.1 Introduction

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

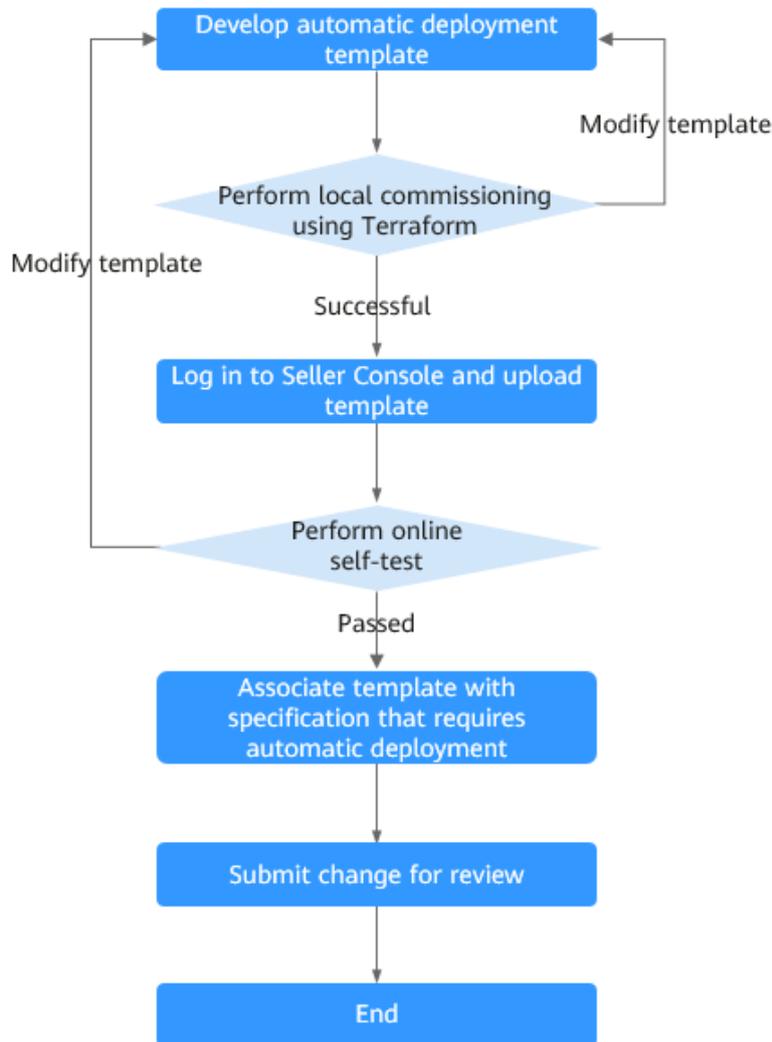
You can develop automatic deployment templates for applications released to KooGallery. After purchasing an application, customers can deploy it on the cloud with one click. Automatic deployment improves the purchase and deployment efficiency of customers and reduces delivery costs of sellers.

### Prerequisites

You have become a seller of Huawei Cloud KooGallery. For details about how to register as a seller, see [Registration Process](#).

## 2.2 Image Access Process

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



## 2.3 Automatic Deployment

### 2.3.1 Developing an Automatic Deployment Template

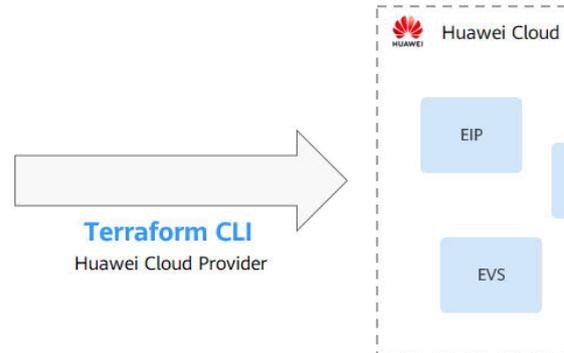
You can develop automatic deployment templates on Terraform. **Terraform** is an open-source automatic resource orchestration tool. The following figure shows the process of using Terraform to manage cloud resources. Before developing a template, install and configure Terraform by referring to [Getting 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 \(RFS\)](#).

```
resource "huaweicloud_ecs_instance_v1" "basic" {
  name      = "server_1"
  image_id  = "ad091b52-742f-469e-8f3c-fd81cadf0743"
  flavor    = "s1.medium"
  vpc_id    = "8eed4fc7-e5e5-44a2-b5f2-23b3e5d46235"

  nics {
    network_id = "55534eea-533a-419d-9b40-ec427ea7195a"
  }

  availability_zone = "cn-north-1a"
  key_name          = "KeyPair-test"
  security_groups  = ["default"]
}
```

Infrastructure as Code



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](#).

## 2.3.2 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.

## 2.3.3 Sample Code

For details about how to develop an automatic deployment template, see [HuaweiCloud Provider Documentation](#).

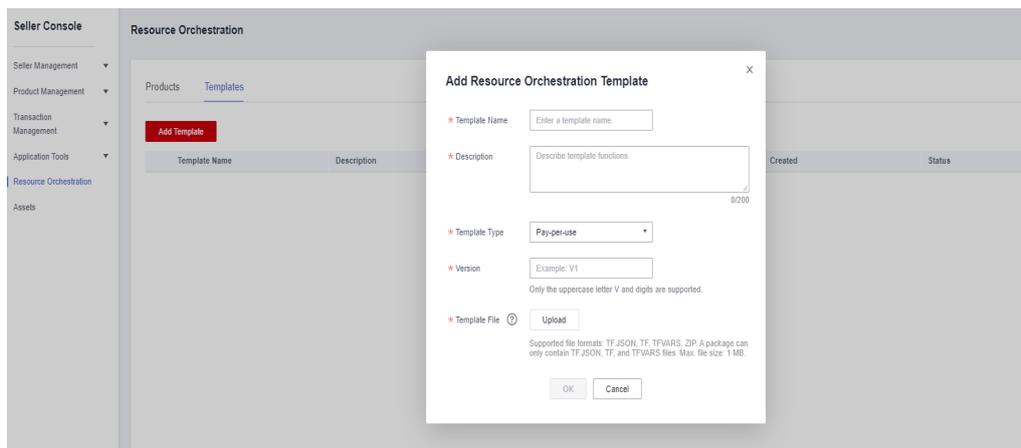
## 2.4 Associating an Image Asset with an Automatic Deployment Template

### Prerequisites

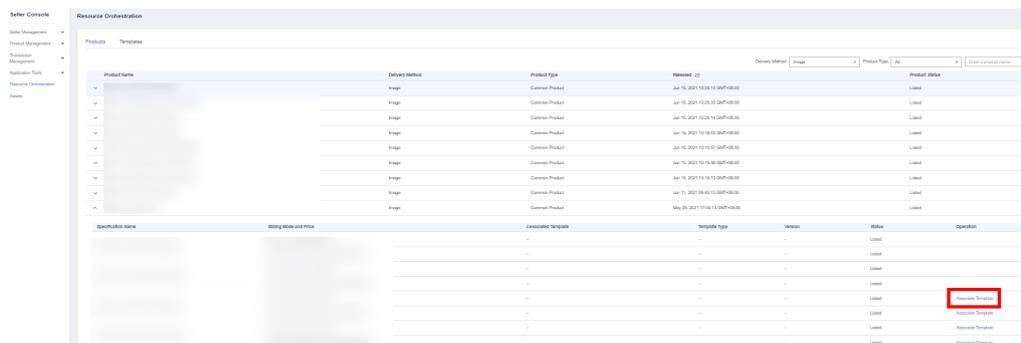
An image product has been released. For details about how to create a private image and release an image product, see [Image Release Guide](#) of Huawei Cloud KooGallery.

### 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

## 2.5 Releasing and Modifying a Product

For details, see [Image Release Guide](#).

## 2.6 Automatically Deploying a Product

For details about how to automatically deploy images, see [Purchasing and Using an Image](#).