IoT Device Management

User Guide

Issue 02

Date 2019-09-02

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


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

Notice

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

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

Huawei Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: http://www.huawei.com

Email: support@huawei.com

Contents

1 Management Platform

1.1 Logging In to the Management Portal of IoT Device Management

Procedure

1. Log in to the Management Console of the IoT platform.
2. In the navigation tree on the left, choose **IoT Device Management**.
3. Click **Management Portal** in the upper right corner.
   You do not need to enter the account and password to log in to the Management Portal.

Browser Requirements

To ensure good display effect and ease of use, use a browser with good compatibility. **Table 1-1** lists the browser requirements.

Table 1-1 Requirements for the browser

Browser Type	Version Requirements	Recommended Resolution
Internet Explorer	Internet Explorer 11.0 or later	1366 x 768
Firefox	Firefox 51.0-61.0	
Google Chrome	Google Chrome 58.0-67.0	

1.2 Application Management

1.2.1 Create Application

Description

As a cross-industry universal platform designed for enterprise customers, the IoT platform supports access of a variety of applications to meet personal service requirements.

To use the IoT platform, you must first create an application. The application can be treated as the project space for your services on the IoT platform, and you can connect your network applications (NAs) and devices to it. After an application is created, the IoT platform assigns the application and device access addresses and ports to support fast access of the NA and devices.

The IoT platform allows a user to create a maximum of 10 applications. (To create more, contact HUAWEI CLOUD customer service.) Applications are isolated from each other for independent management. The IoT platform also supports **application authorization** to enable cross-application management.

Procedure

NOTE

A default application is provided. If you want to use the default application, **reset its secret**.

Step 1 Choose **Application List**, and click **Create Application**.

Step 2 Set the parameters based on **Table 1-2**.

Table 1-2 Application creation parameters

Parameter	Description
Basic Information	
Application Name	Specify the name of an application. It must be unique under the user and cannot be changed.
Industry	Select a value based on the industry attributes of the application.

Parameter	Description
Message Tracing Authorization	Specify whether the IoT platform operations administrator can trace faulty devices. ● If message tracing authorization is enabled, the IoT platform operations administrator, when helping you locate faults, can trace service data reported by devices. When authorization is enabled, **Authorization Validity** must also be specified. The value of **Authorization Validity** can be set to **Custom** or **Always**. To ensure user data rights, the IoT platform operations administrator can retain the device data for a maximum of three days. ● If message tracing authorization is disabled, the IoT platform operations administrator cannot trace service data reported by devices. This may reduce fault locating efficiency. You are advised to enable authorization.
Message Push	
Protocol Selection	**Push Protocol** The push protocol is determined by the transport protocol set when a network application (NA) subscribes to device information from the IoT platform. If the transmission channel for data push is set to HTTP on the NA, you can use HTTPS or HTTP to transmit data. ● **HTTPS**: Encrypted transmission is used between the IoT platform and NA. A CA certificate must be uploaded to the NA. ● **HTTP**: Non-encrypted transmission is used between the IoT platform and NA. This mode is relatively less secure, and data sent between the IoT platform and NA may be disclosed. **CA Certificate** The CA certificate is provided by the NA and used by the IoT platform to verify the NA. **NOTE** The CA certificate preconfigured on the IoT platform is used only for commissioning. In commercial scenarios, use the CA certificate provided by the NA.
Platform Capability	
Device Data Management	The IoT platform can store historical device data. You can enable or disable the storage function. The default value is **On**. ● If the value is **On**, the IoT platform stores historical data. The storage duration is subject to that displayed. ● If the value is **Off**, the IoT platform does not store historical data.
Push Service	The NA subscribes to device information from the IoT platform, and the IoT platform pushes messages to the NA.
Other	
Description	Describe the application.
Application Icon	Specify the icon of the application.

Step 3 Select **I have read and agree to the Terms of Personal Data Use**, and click **Confirm**. After the application is created, the **Success** dialog box is displayed, showing basic information about the application, including the application ID, application secret, application access address, and device access address.

- Click **Save Secret to Local** to save the application secret. The secret is invisible on the application details page. Keep it secure. If you forget the secret, click and choose **Reset Secret**. Alternatively, you can open the application details page, click the **Information** tab page, and click **Reset** under **Security**.

  NOTE

  The application ID and application secret are used by the NA to connect to the IoT platform. If you reset the secret, the old secret becomes invalid, and the NA server must use the new secret to access the IoT platform. Exercise caution when performing this operation.

- Click **Go to Application Details** to view the application details page.
- Click **Return to Application List** to display the page for creating an application. Click the application icon to view its details.

Step 4 (Optional) Perform this step only if you want to use a specified email server or SMS server, Generally, you do not need to select a server. If multiple servers exist, you can manually specify a server; otherwise, the system selects a server randomly.

1. Click the created application, and open the **Information** tab page.
2. Click **Edit**. In the **Platform Capability** area, select the specified email server and SMS server.

   NOTE

   Contact the IoT platform operations administrator to add an email server and SMS server. These servers can be selected only after they are added.

----End

1.2.2 Information

After creating an application, you can view or modify its information. Click an application on the **My Application** tab page to view its details.

Table 1-3 Basic application information

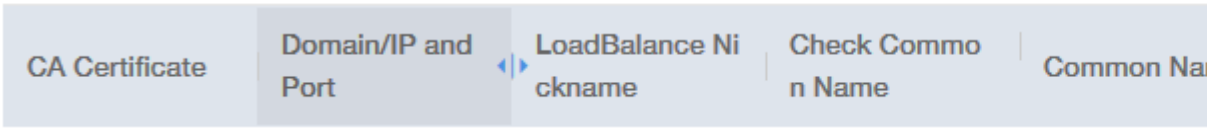
Parameter	Description
Basic	You can query basic information about an application, including its name, ID, creation time, and industry to which it belongs. (The application ID is used by NAs to access the IoT platform.) You can change the industry based on site requirements.

Parameter	Description
Security	● You can reset the application secret. After resetting, the old secret becomes invalid, and the NA server must use the new secret to access the IoT platform. ● You can set **Message Tracing Authorization** to **On** or **Off**. – If message tracing authorization is enabled, the IoT platform operations administrator, when helping you locate faults, can trace service data reported by devices. When authorization is enabled, **Authorization Validity** must also be specified. The value of **Authorization Validity** can be set to **Custom** or **Always**. To ensure user data rights, the IoT platform operations administrator can retain the device data for a maximum of three days. – If message tracing authorization is disabled, the IoT platform operations administrator cannot trace service data reported by devices. This may reduce fault locating efficiency. You are advised to enable authorization.
Access Mode	● You can query the IP address used by devices bound to the application to access the IoT platform, as well as the port numbers corresponding to different protocols. ● You can also query the IP address and port number of NAs bound to the application to access the IoT platform.
Message Push	The push protocol is determined by the transport protocol set when the NA subscribes to device information from the IoT platform. If the transmission channel for data push is set to HTTP on the NA, you can use HTTPS or HTTP to transmit data. If HTTPS is used, a CA certificate must be uploaded. For details on how to upload the CA certificate, see **Step 1**. **NOTE** If HTTPS is used and an NA cancels the subscription, the bound certificate is automatically unbound. If a new subscription begins, you must upload the CA certificate again.
Platform Capability	You can select the email server or SMS server to send emails or SMSs in the rule triggering, user registration, and password retrieval scenarios. Generally, you do not need to select a server. If multiple servers exist, you can manually specify a server; otherwise, the system selects a server randomly. **NOTE** Contact the IoT platform operations administrator to add an email server and SMS server. These servers can be selected only after they are added.
Other	You can modify the application icon and description.

To load a CA certificate, perform the following steps:

Step 1 Click **Manage Certificate**. In the dialog box displayed, click **Add**.

Figure 1-1 CA Certificate dialog box



Step 2 Set the parameters based on **Table 1-4**, and click **Confirm**.

Table 1-4 CA Certificate parameters

Parameter	Description
CA Certificate	You must apply for and purchase a CA certificate file in advance. The CA certificate is provided by the NA.
Domain/IP and Port	Specify the domain name or IP address and port number used by the IoT platform to push messages to the NA server. Set this parameter to the domain name or IP address and port number in callback URL in the subscription interface. Example values are **api.ct10649.com:9001** and **127.0.1.2:8080**.
LoadBalance Nickname	Specify the nickname of the LoadBalance to which the certificate is loaded. If there are multiple VPNs, select the corresponding nickname. In other scenarios, use the default value **default.**
Check Common Name	Specify whether the common name of the CA certificate is verified to ensure that the loaded certificate matches the applied certificate. It is recommended that the common name be verified.
Common Name	Specify the common name of the CA certificate. This parameter is displayed when **Check Common Name** is set to **ON**. Obtain the value from the certificate applicant.

----End

1.2.3 Service Settings

The power-saving configuration and northbound push management are available.

Power-Saving Configuration	Parameter Description

Working Mode	The IoT platform supports three working modes for NB-IoT devices: PSM, DRX, and eDRX. The timeout interval for command delivery varies by working mode. If the IoT platform does not receive a command execution result during the timeout period, the task state changes to **timeout**. ● PSM: The default timeout interval is 300 seconds. A device does not receive downstream data during the non-service period. It can receive downstream data cached by the IoT platform only after sending upstream data (MO data) to the platform. This mode is suitable for services that have no platform-to-device data delay limit. Devices in this mode have low power consumption and are powered by batteries. An example service is meter reading. ● DRX: The default timeout interval is 300 seconds. Downstream service data can reach devices at any time. In each DRX cycle (1.28s, 2.56s, 5.12s, or 10.24s), a device detects whether there is a downstream service. This mode is suitable for services that require low delay. Generally, devices in this mode are powered by a wired current. An example service is street lamps. ● eDRX: The default timeout interval is 2 x eDRX cycle + 120 seconds. In each eDRX cycle, a device can receive downstream data only during the preset paging time window (PTW); otherwise, it remains in the dormant state and does not receive downstream data. This mode balances between service delay and power consumption. An example service is remote gas shut-off. Click **Set** on the right of the page to set the working mode. **NOTE** The IoT platform does not deliver the configuration to the network or device. Before configuration, obtain the working mode and parameters of the device from the carrier.
Downstream Message Direction	When this function is enabled, if the IoT platform has a pending command to deliver to a device using the PSM mode, the platform includes the **hasMore** field in the response to a data reporting message sent from the device. After detecting this field in the response, the device does not enter the dormant mode immediately.
Northbound Push Management	**Parameter Description**
Push Service Control	To prevent an NA from occupying excessive resources due to an exception, the IoT platform uses flow control for push services. After confirming that an exception occurs, you can temporarily disable the push service. After the exception is resolved, you can enable it again. ● **Maximum Concurrent Messages per Second**: specifies the maximum number of messages pushed by the IoT platform to NAs per second over HTTP. You can change the maximum number based on service requirements. ● **Maximum Connections per Second**: specifies the maximum number of connections per second when the IoT platform uses MQTT to push messages to NAs.

Subscription/ Push Service (HTTP/HTTPS)	The IoT platform supports query of northbound push configuration and the callback URL list. If the IoT platform fails to push messages to a callback URL 10 consecutive times within 180 seconds, the callback URL is automatically set to invalid. The IoT platform periodically checks whether the callback URL is restored. If so, the platform sets the status of the callback URL to valid.
Subscription/ Push Service (MQS)	The IoT platform pushes data reported by devices to the big data platform through the message queue service (MQS) for analysis and processing. The MQS is a third-party component provided by Huawei. Interworking data between the IoT platform and MQS must be configured in advance. ● **iPaaS Address**: address of the MQS Namesrv. The format is IP:port, for example, **10.10.11.11:8965**. ● **Username**: user name for accessing the MQS. ● **Password**: password for accessing the MQS. (Obtain **iPaaS Address**, **Username** and **Password** from the MQS administrator.) ● **Encryption**: whether messages sent from the IoT platform to the MQS are encrypted. You are advised to select **Enable**. ● **Topic**: topic name, for example, **To_Bigdata**. Different topics are used to process different message types. Currently, only one topic can be configured, indicating that all information is sent to the same topic for processing. Obtain the value from the MQS administrator.
Subscription/ Push Service (MQTT)	The IoT platform uses MQTT to push messages to an NA. You can set the subscribed-to message type on the SP portal. Currently, data reporting is supported.
Subscription/ Push Service (ROMA)	The ROMA service is an enterprise business integration platform and is to be launched. **NOTE** After the IoT platform is connected with ROMA, you must configure data forwarding rules to forward data to ROMA. For details on how to set a rule, see **Rules**.

1.2.4 Statistics

The **Statistics** tab page provides statistics on communication between an application and the IoT platform. You can click **View** on the right of each row to view details.

Table 1-5 Items on the Statistics tab page

Item	Description	Data Update Period
Data Reporting Statistics	The number of data reporting messages from directly connected devices in the past 72 hours (excluding device status change data, for example, device go-online). The reported data type can be the temperature, battery, and others. ● The metrics include **Total**, **Successful**, **Failed**, and **Success rate**. ● The metrics are measured over time, and the IoT platform supports filtering by metric or time range.	30 minutes
Command Event Statistics	The number of control instructions sent from the IoT platform or NAs to devices. ● The metrics include **Total**, **Successful**, **Failed**, and **Success rate**. ● The metrics are measured over time, and the IoT platform supports filtering by metric or time range.	30 minutes
Application Flow Control	The IoT platform controls the number of messages simultaneously reported by all devices under the same application ID. This prevents system performance from deteriorating or the system from breaking down when the devices experience exceptions or are attacked. The data of the past 24 hours is retained.	Manually refreshed

1.2.5 Grant List and Accept List

The IoT platform supports authorization between applications of the same user and different users. In addition, two applications can be mutually authorized. If application A is authorized to application B, application B can view and manage all devices connected to application A on the Management Portal. For example, application B can create rules and perform batch operations on devices connected to application A.

- Authorizations are not transitive. If application A is authorized to application B and application B is authorized to application C, application A is not necessarily authorized to application C.
- There are two types of authorization: view and edit. The object to be edited is devices connected to the application, not the application itself. For example, if application A is authorized to application B, the user of application B can operate the devices connected to application A but cannot edit the information on the application details page of application A.

Procedure

To authorize application A to application B, perform the following steps:

Step 1 Click application A to view its details.

Step 2 Click the **Grant List** tab.

Step 3 Click **Authorized Applications**. You can delete authorized applications from the list.

Step 4 Click **Grant Authority**. In the dialog box displayed, select application B. If application B belongs to another user, select **Application of another user** for **Target Application** and specify **Application ID** and **Permission**. The user of application A can operate devices connected to application B only if **Permission** is set to **Edit**.

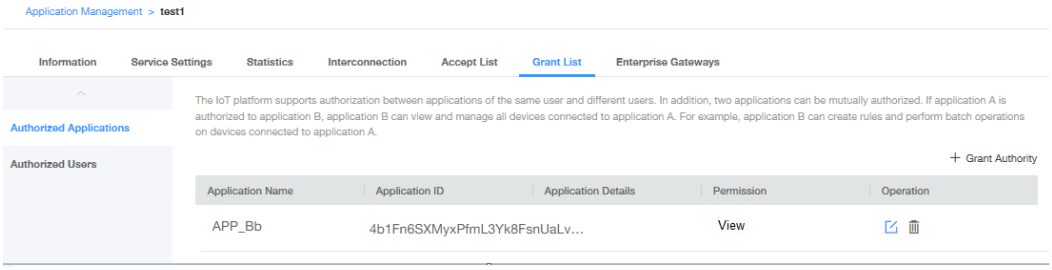
NOTE

To obtain an application ID, use either of the following operations:

- When a new application is created, the IoT platform automatically returns the application ID.
- For an application that has been created, you can view its ID on the **Information** tab page of the application details page.

Step 5 Click **Grant**.

- On the page displayed after you choose **Grant List** > **Authorized Applications** of application A, you can view application B. To cancel the authorization, click .



- On the page displayed after you choose **Accept List** of application B, you can view basic information about application A. If applications A and B belong to different users, you can log in to the Management Portal as the user corresponding to application B, choose **Application Management** > **Application List**, and click the **Accept List** tab to view basic information about application A.
- If applications A and B belong to different users, you can choose **Grant List** > **Authorized Users** of application A to view the name of the user to which application B belongs. To revoke the authorization, click . Doing so removes all authorized applications of the user from the authorized application list.

Step 6 View and operate all devices connected to application A through application B.

----End

1.3 Device Management

1.3.1 Home

The IoT platform provides a variety of dashboards to intuitively display data and supports exporting of statistics by month or day. After you choose **Device Management** > **Home**, the dashboards are displayed. **Table 1-6** provides the names and functions of the dashboards provided by the IoT platform.

Table 1-6 Dashboard parameters and functions

Dashboard	Description	Data Update Frequency
Total Devices	The number of devices, including online devices, offline devices, and abnormal devices, for the current application. You can click anywhere in the **Total Devices** area to display monthly and daily statistics.	Every minute
API Calls	The number of API calls every day. You can click anywhere in the **API Calls** area to display the data for **API Calls by All Applications** and **API Calls by Current Application**.	Every hour
Report Messages	The number of messages reported by devices every day. You can click anywhere in the **Reported Messages** area to display the data reporting trend and average message reporting rate.	Every hour
Delivered Commands	The number of commands delivered by the IoT platform every day. You can click anywhere in the **Delivered Commands** area to view the message delivery trend.	Every hour
General Device Trend	The total number of devices and the number of online devices. You can click **Details** to display monthly and daily statistics.	Every hour
Online Device Rate	The device online rate as a percentage.	Every hour
Device Data Chart	The number of newly added devices, deleted devices, activated devices, offline devices, and abnormal devices.	Every hour
Push Messages	The number of messages pushed to NAs.	At 00:00 every day
Command Status	The number of commands delivered by the IoT platform, including pending, timeout, failed, successful, sent, canceled, delivered, and expired commands.	At 00:00 every day
Alarm Statistics by Application	The number of alarms generated by all devices connected to the application every day.	At 00:00 every day
Active Devices	The number of devices that have reported data in the last one hour.	Every hour
Bootstrap Devices	The number of devices that are registered and activated using the bootstrap service.	5 minutes
Software Upgrade Status	The number of historical software upgrade tasks of all devices in the current application, and the number of devices that are successfully and fail to be upgraded.	In real time

Dashboard	Description	Data Update Frequency
Firmware Upgrade Status	The number of historical firmware upgrade tasks of all devices in the current application, and the number of devices that are successfully and fail to be upgraded.	In real time
Configuration Update Status	The number of historical configuration update tasks of all devices in the current application, and the number of devices that are successfully and fail to be updated.	In real time

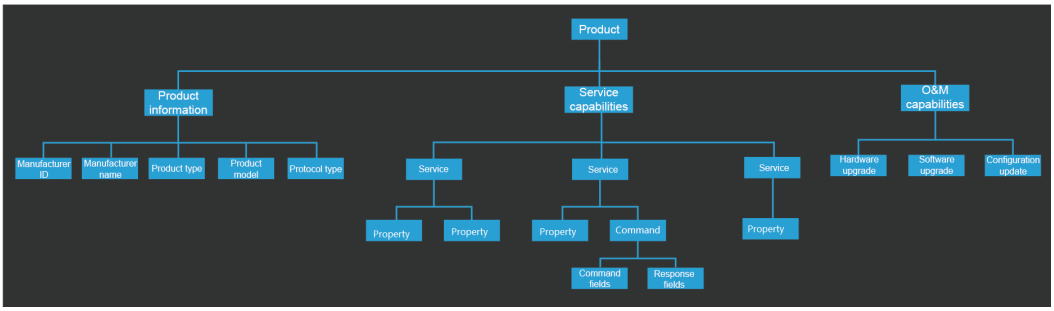
1.3.2 Product Models

Description

A product model (also called profile file) describes the capabilities and features of a device. Developers build an abstract model of a device by defining a profile file on the IoT platform so that the IoT platform can understand the services, properties, and commands supported by the device, such as color and switch.

A profile file consists of product information, service capabilities, and maintenance capabilities. The quintuple of the product information uniquely defines a device type. After defining a product model, you can select the imported product during **device registration**.

Figure 1-2 Profile file structure



You can import a product model in either of the following ways:

- Import from the Product Center. Define a product model in the Developer Center and release it to the Product Center.

  NOTE

  If binary code stream is selected as the data format during the definition of a profile file in the Developer Center, you must also develop a codec plug-in online.

- Import from your local PC. Develop a product model offline and import it by uploading the product package.

  NOTE

  The product model imported from your local PC does not contain the codec plug-in. If devices report binary code streams, contact the IoT platform operations administrator to upload the codec plug-in.

- Manually create a product model. A manually created product does not contain a profile file. This operation applies to the scenario in which LWM2M is used for device access. The IoT platform automatically generates a product model (profile file) based on the ObjectID and ResourceID carried by the device.

Prerequisites

- In the case of import from the Product Center, the product model has been defined and released to the Product Center.
- In the case of import from your local PC, the profile file has been created.

Procedure

Step 1 Choose **Product Models**, and click **Add**.

Step 2 Import the product model from the Product Center or local PC.

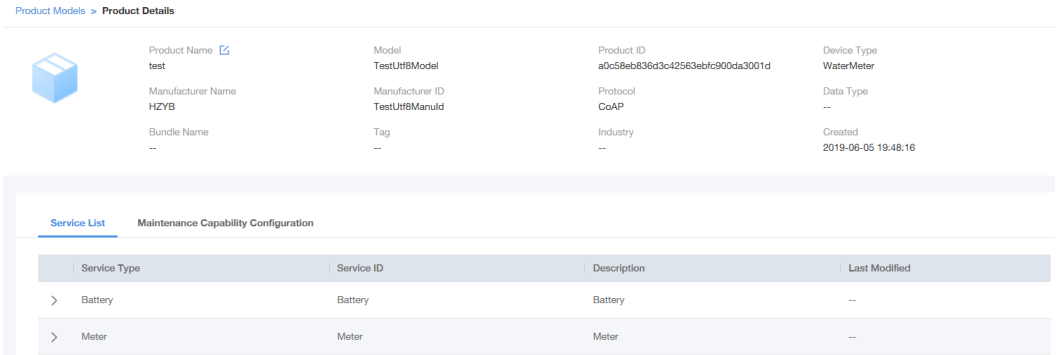
- Import from the Product Center.
  a. Choose **Import from Product Center** to open the **Product Center** page.
  b. Search for a product by product name, device type, or manufacturer name. In the search result, click the name of the product to be imported.
  c. Check whether the product is a public product.
     - For a public product, you can click **Import** to import the product model from the Product Center to the IoT platform.
     - For a private product, you must enter the verification code obtained from the Product Center. If the verification is successful, you can view the product details and import the product model to the IoT platform.
- Import from your local PC.
  a. Choose **Import from Local**.
  b. In the dialog box displayed, enter the product name and upload the resource file.
  c. Click **Confirm** and wait until the import is complete.

  NOTE

  The product ID and product key are used for device registration. Click **Save to Local** to save the product key. The product key is not displayed on the product model details page. Keep it secure.

Step 3 View the import result on the **Product Models** page.

- Import failure: You can view the cause of the import failure in the **Failure Cause** area. This helps with fault locating.
- Import success: You can click **Details** to view product model details.



NOTE

You can delete a disused product from the product list by clicking **Delete**. After deletion, the devices of this product cannot be used. The functions of the devices under the product are restored only after the product is imported to the Product Center again.

----End

1.3.3 Devices

1.3.3.1 Device List

The **Device List** page presents the total number of devices, online devices, offline devices, and inactive devices for the current application. You can configure a device. You can click a device to open the details page and view basic information, historical data, device shadow, and device tracking information. You can also modify some information.

The following table lists the statistics on each device status.

Device Status	Description	Refresh Period
Total	Total number of devices that have been registered with the IoT platform, including online, offline, and inactive devices.	Every minute
Online	A device is connected to the IoT platform. If a short-connection device (such as an NB-IoT device) reports no data for 49 consecutive hours (default) after connecting to the IoT platform, the IoT platform changes the device status from **Online** to **Offline**.	
Offline	If a short-connection device (such as an NB-IoT device) reports no data for 49 consecutive hours (default) after connecting to the IoT platform, the IoT platform sets the device status to **Offline**. If a long-connection device (such as an MQTT device) is disconnected from the IoT platform, the device status is changed to **Offline**.	
Inactive	A device has been registered but not connected to the IoT platform. The device activation procedure is described in **Connecting a Device**.	

Configuration Delivery

Click  at the row of a device to configure the device.

NOTE

Before delivering the configuration, open the **Product Models** page, click **Details** of the product model to which the device belongs, and set **Device Configuration** to **Supported** on the **Maintenance Capability Configuration** tab page.

Table 1-7 Parameters for configuring a device configuration delivery task

Parameter	Description
Task Name	Name of a configuration delivery task.
Execution Type	Task execution type. The value can be **Now**, **Device online**, or **Custom**. If the value is **Custom**, **Executed Time** must be set.
Executed Time	Time at which the task is executed. This parameter is valid only if **Execution Type** is set to **Custom**.
Retry Type	● **No**: No retry is performed. ● **Custom**: **Retry Attempts** must be set.
Retry Attempts	This parameter is mandatory if **Retry Type** is set to **Custom**.
Retry Interval (s)	This parameter is mandatory if **Retry Type** is set to **Custom**.
Configuration File	You must upload the configuration file of the device. You can click **Device Configuration File** to download the configuration file that has been delivered last time, modify the file, and upload it. **NOTE** If no configuration is delivered to the device before this operation, a message is displayed indicating that no resource was found when you click **Device Configuration File**. In this case, manually create a configuration file in user-defined mode. The items to be delivered must be consistent with the capabilities supported by the device (defined in the product model) and the configuration file must be in JSON format. An example configuration is as follows: `{ "sensitivity": "0", "dataReportInterval": "20" }`

1.3.3.2 Device Details

1.3.3.2.1 Information

On the device list, click a device to open the **Device Details** page.

The **Information** tab page provides basic information about device registration and access. You can click **Edit** to modify some information.

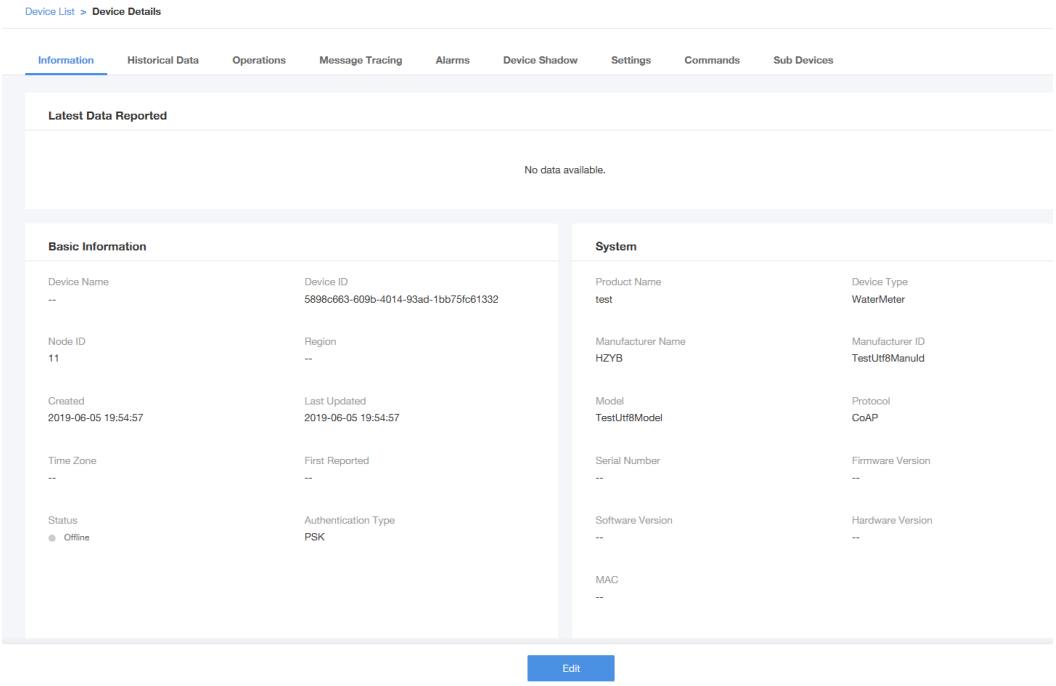


Table 1-8 Information tab page

Item	Description
Latest Data Reported	Latest data reported by the device. By default, the latest eight properties reported are displayed. You can click **View All Properties** to view the latest 20 properties.
Basic Information	Information entered during device registration and the information generated by the IoT platform. ● The registration information includes **Device Name**, **Node ID**, **Region**, **Time Zone**, and **Authentication Type**. ● The automatically generated information includes **Device ID** and **Status**. **NOTE** The device status is as follows: ● **Online**: A device is connected to the IoT platform. If a short-connection device (such as an NB-IoT device) reports no data for 49 consecutive hours (default) after connecting to the IoT platform, the IoT platform changes the device status from **Online** to **Offline**. ● **Offline**: If a short-connection device (such as an NB-IoT device) reports no data for 49 consecutive hours (default) after connecting to the IoT platform, the IoT platform sets the device status to **Offline**. If a long-connection device is disconnected from the IoT platform, the device status is changed to **Offline**. ● **Inactive**: A device has been registered but not connected to the IoT platform. The device activation procedure is described in **Connecting a Device**.

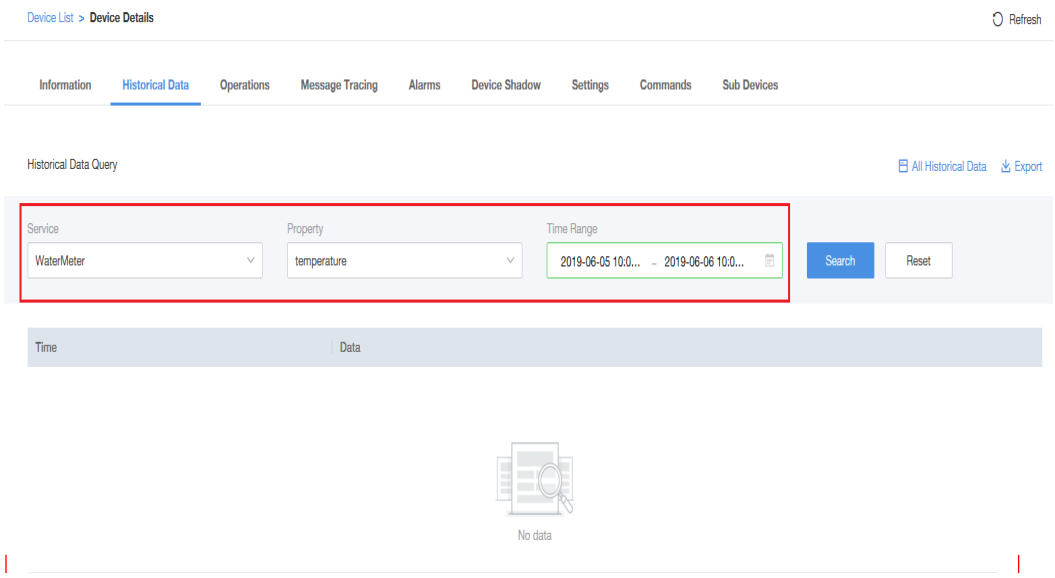
Item	Description
System	Information entered during device registration and basic information reported by the device. ● The registration information includes **Device Type**, **Manufacturer Name**, **Manufacturer ID**, **Model**, and **Protocol**. ● Basic device information reported by the device after being connected to the IoT platform includes **Serial Number**, **Firmware Version**, **Software Version**, and **MAC Address**.
Other	Information as follows: ● **Gateway ID**: If a sub device (for example, a sensor) is connected to the IoT platform through a gateway, the gateway ID is displayed. If the device is directly connected, the device ID is displayed. ● **Group Name**: specifies the name of the group to which the device is bound. A device can be bound to only one group.
Link	Information about the device connected to the IoT platform, including: ● **Access Mode**: specifies the network access mode of the device, such as NB-IoT, LTE, GSM, WLAN, or Bluetooth. It is automatically generated based on the access mode of the device. ● **IMEI**: a type of node ID, which uniquely identifies a device. An administrator can query a device on the Management Portal by IMEI. ● **IP**: specifies the IP address of the device. The device IP address is reported to the IoT platform during device registration. ● **IMSI**: specifies the IMSI of the device, which is used for fault locating on the EPC network. The IMSI is reported to the IoT platform during device registration. ● **Cell ID**: identifies the cell accessed by the device. It is reported during device registration and updated during data reporting. The cell ID can be used to accurately locate the cell of a device.
Location	Longitude and latitude of the device location or area information of the device. If the device can report GPS information and the Location service is included in the product model, the latest longitude and latitude of the device are displayed when the device reports data.
Tag	Tags of the device so that you can quickly identify device properties. For example, if you define the tag **water_Device** with the value set to **A_Region** for a water meter, you know the device is a water meter in region A.

1.3.3.2.2 Historical Data

On the device list, click a device to open the **Device Details** page.

The **Historical Data** tab page presents historical data reported by the device. You can filter this data by **Service**, **Property**, **Time Range**, and their combinations. The tab page also provides the corresponding historical data report and list.

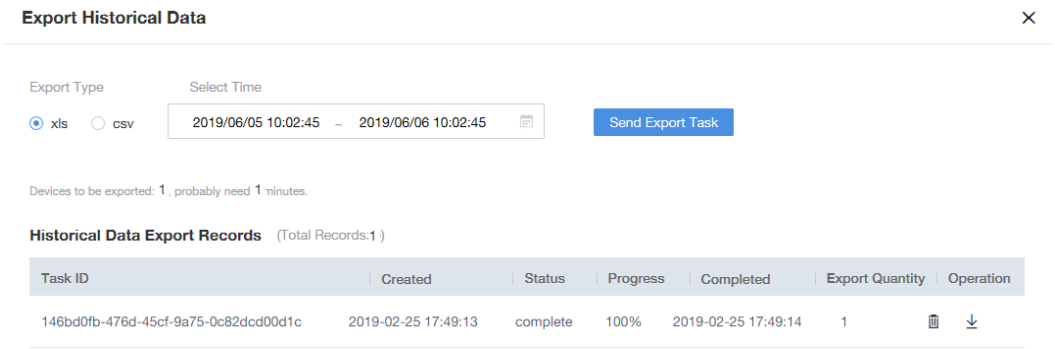
Figure 1-3 Historical Data tab page



You can click **All Historical Data** on the tab page to view the historical data of all service properties.

You can click **Export** on the tab page to export historical data. In the dialog box displayed, set the exported file format, select the export period, and click **Send Export Task**. After export, you can download historical data from the task list to your local PC.

Figure 1-4 Exporting historical data



1.3.3.2.3 Operations

On the device list, click a device to open the **Device Details** page.

The **Operations** tab page presents historical operations performed on the device. The information helps you learn the operation history and the result (success or failure) during fault locating.

When viewing operation records, you can filter the operation type to accurately view the status of specific operation types. The operation type can be **Reboot**, **Configuration**, **Synchronization**, **Sensor upgrade**, **Bundle operation**, **Restore factory settings**, **Software upgrade**, or **Firmware upgrade**.

1.3.3.2.4 Software

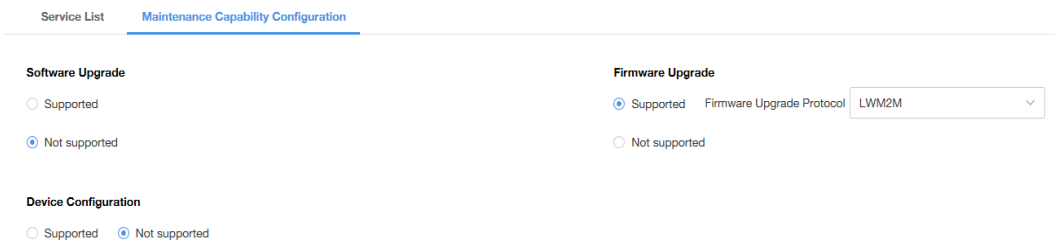
On the device list, click a device to open the **Device Details** page.

The **Software** tab page displays the software and firmware versions of the device. You can manage the software and firmware versions on this page. If new software or firmware is available for the device, you can click the upgrade button to start upgrading.

To use the upgrade function, the following conditions must be met:

- The device must support software or firmware upgrade.
- In the **product model** of the device, **Software Upgrade** or **Firmware Upgrade** on the **Maintenance Capability Configuration** tab page of the product details is set to **Supported**.

Figure 1-5 Maintenance Capability Configuration tab page



- Before upgrading software or firmware, you must upload the **software** or **firmware** package. The uploading function is available only after the FOTA/SOTA upgrade service is purchased.

References

Device Upgrade

1.3.3.2.5 Message Tracing

On the device list, click a device to open the **Device Details** page.

The message tracing function can be used to quickly locate and analyze faults when a fault occurs in device binding, command delivery, data reporting, device information update, and device monitoring. The IoT platform supports message tracing for NB-IoT and MQTT devices. Up to 10 devices bound to an application can be traced simultaneously.

Enabling Message Tracing

Step 1 Log in to the Management Portal, click **Device Management** on the upper navigation bar, and choose **Devices** > **Device List** in the navigation tree.

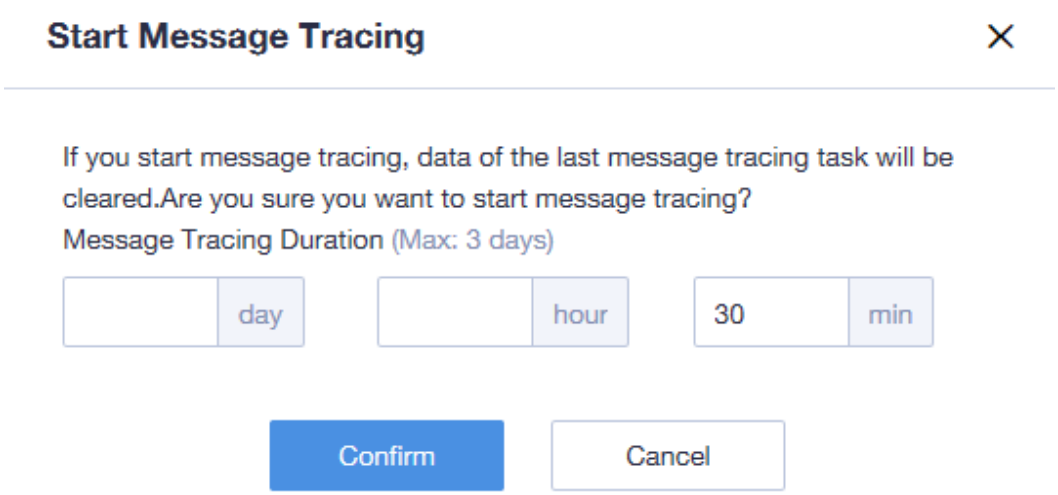
Step 2 Search for the device to be traced and click the device to open the device details page.

Step 3 On the **Message Tracing** tab page, click **Start** or **Restart**. In the dialog box displayed, set the message tracing duration, and click **Confirm**. The message tracing duration starts from the time at which the task starts, and messages generated thereafter are traced. If you click **Restart**, the time restarts.

NOTE

If you start the message tracing again, the historical data of the last message tracing is cleared.

Figure 1-6 Start Message Tracing dialog box

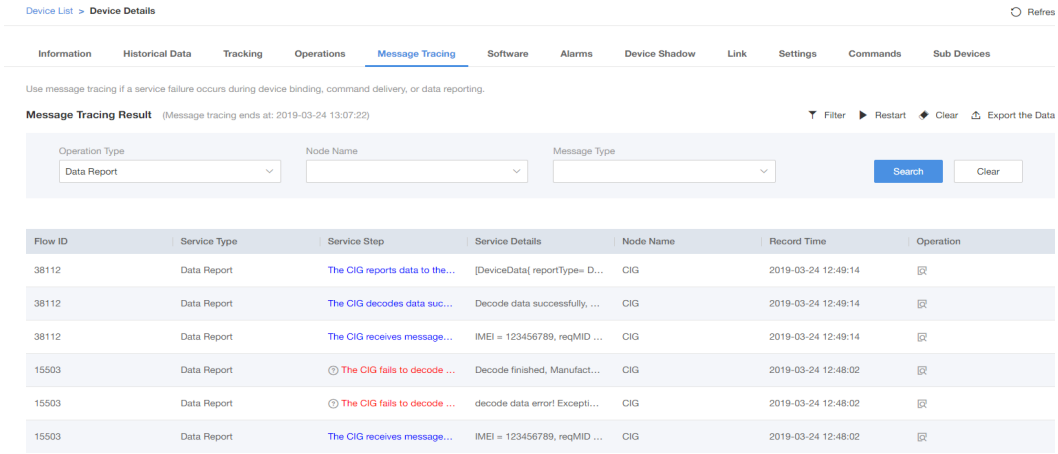


----End

Figure 1-7 shows example tracing results. The tracing records with the same serial number are the result of one service processing. In the following information:

- Blue text: indicates that the node successfully processes the message.
- Red text: indicates that the node fails to process the message. You can click next to a tracing record to view its details and locate and analyze the fault based on the failure handling suggestions.

Figure 1-7 Message tracing results



After starting a message tracing task, the IoT platform traces messages in the following scenarios: device binding, command delivery, data reporting, device information update, and device monitoring. If a large number of tracing records are displayed, you can filter the records by service type, node name, and message status. If you need to further analyze the result data, you can export the data.

1.3.3.2.6 Alarms

On the device list, click a device to open the **Device Details** page.

The **Alarms** tab page displays only alarms defined in **rules**. You can manage the device status based on the defined rules. Pay close attention to device alarms and handle the alarms quickly to ensure that devices run normally. You can also search for historical alarms of devices.

Alarm severity and handling suggestions are as follows:

- Critical: Customer services are interrupted or devices may become unavailable. Measures must be taken immediately to rectify the fault.
- Major: Devices are partially affected or the system performance is affected. Corrective measures must be taken as soon as possible to prevent more serious faults.
- Minor/Warning: There is no current impact on devices. The system has detected potential or imminent faults that may affect services but services are not yet affected. However, you must check for potential faults.

To view all alarms of the current application, follow the instructions provided in **Alarms**.

1.3.3.2.7 Device Shadow

Description

On the device list, click a device to open the **Device Details** page.

Each device has one shadow, which is a JSON file that stores the property value reported by a device and the property value that the IoT platform expects to deliver to the device. Only the latest reported values and expected values are stored in the device shadow.

Application Scenario

- Query device property status
  - If an NA queries the status of a device when the device is offline, the NA cannot obtain the device status in a timely manner. The device shadow stores the latest device status. Once the device status changes, the device synchronizes its status to its shadow. Using the device shadow, the NA can obtain the device status quickly regardless of whether the device is online.
  - Many NAs frequently query the device status. Due to the limited processing capability of the device, frequent queries adversely affect device performance. The device shadow enables the device to actively synchronize its status. The NAs request the device status from the device shadow. In this way, NAs and devices are decoupled.
- Modifying device properties: A device administrator modifies device properties through the SP portal or by calling an API. If the modified configuration cannot be delivered to the device in a timely manner because the device is offline, the IoT platform stores the modified device properties in the device shadow. When the device goes online, the IoT platform synchronizes the new device properties from the device shadow to the device.

  NOTE

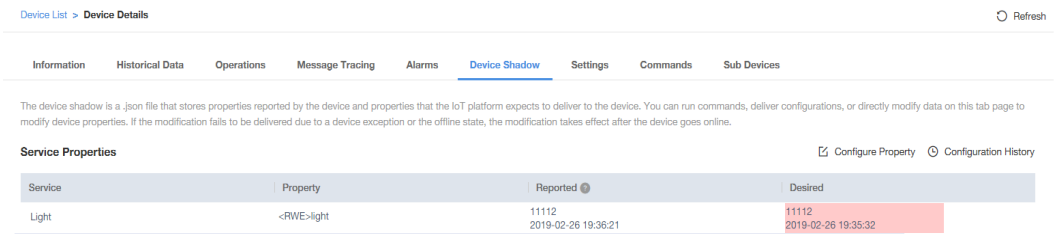
  - This function applies only to devices that support the LWM2M protocol.
  - The property modified in the device shadow can only be one defined in the LWM2M protocol. User-defined device properties cannot be modified.

Query Method

On the **Device Shadow** tab page, you can view current device properties, including the reported value and desired value. The device administrator can modify properties on the Management Portal or by calling an API.

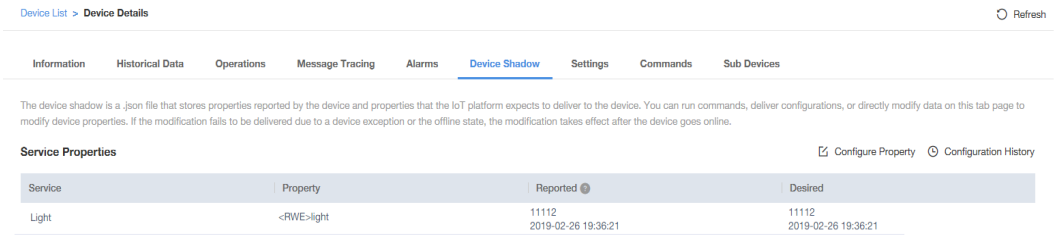
- If the reported value is inconsistent with the desired value, it may be that the device is offline and the value is temporarily stored in the device shadow. The desired value is highlighted.

  Figure 1-8 Inconsistency between the reported value and desired value



- If the reported value matches the desired value, the latest property value reported by the device matches the desired property value. The desired value is not highlighted.

  Figure 1-9 Consistency between the reported value and desired value



1.3.3.2.8 Settings

On the device list, click a device to open the **Device Details** page.

On the **Settings** tab page, you can perform routine operations on a device, including restarting a module, collecting logs, resetting a pre-secret, resetting a secret, moving an application, adding the device mode (NWI), and connecting to the platform.

Figure 1-10 Settings tab page

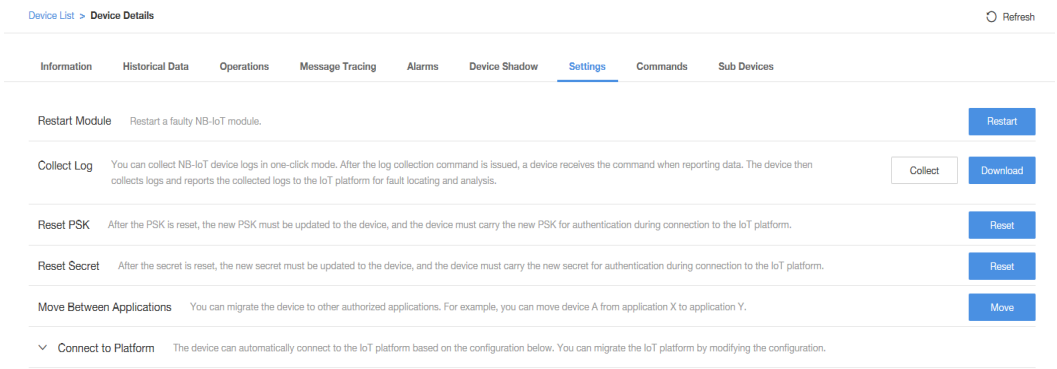


Table 1-9 Functions on the Settings tab page

Function	Description
Restart Module	Note the following about the **Restart Module** operation: ● Only the NB-IoT module can be restarted. ● The device must support remote module restart. **Software Upgrade** or **Firmware Upgrade** on the **Maintenance Capability Configuration** tab page of the product details (**Product Models**) corresponding to the device must be set to **Supported**. ● When you click **Restart**, the IoT platform sends the restart instruction to the device only after the device reports data. After a module restart task is executed, you can view the task status on the **Operations** tab page. The module restart task has the following possible states: ● **Waiting**: The task is in the waiting state. After the task is created, the system sets a 10-second timer. After 10 seconds, the state changes to **Processing**. ● **Processing**: The IoT platform is processing the task. If the device reports no data for 25 consecutive hours, the IoT platform changes the state to **Fail**. Otherwise, the IoT platform delivers the module restart instruction to the device. After restarting the module, the device returns the processing result to the IoT platform, which then changes the state to **Success** or **Fail**. ● **Success**: The module has restarted. If the device returns a result indicating that the module is restarted, the state changes to **Success**. ● **Fail**: The module failed to be restarted. The state changes to **Fail** if the IoT platform does not deliver the module restart instruction to the device within 25 hours, the IoT platform delivers the module restart instruction to the device but does not receive a processing result from the device within 30 minutes, or the device returns the result indicating that the module restart failed.

Function	Description
Collect Log	Note the following about the **Collect Log** operation: ● Log collection applies only to NB-IoT devices. ● The device must support remote log collection. **Software Upgrade** or **Firmware Upgrade** on the **Maintenance Capability Configuration** tab page of the product details (**Product Models**) corresponding to the device must be set to **Supported**. ● When you click **Collect**, the IoT platform sends the log collection instruction to the device only after the device reports data. After a log collection task is delivered, you can click **Download** to download and view the collected log files. The log collection task has the following possible states: ● **Waiting**: The task is in the waiting state. After a task is created, the state changes to **Processing** only after the IoT platform delivers the log collection instruction to the device. If the device reports no data for 25 consecutive hours, the state changes to **Fail**. ● **Processing**: The log collection instruction has been delivered to the device and is waiting for the device to return the log collection result. According to the returned result, the IoT platform changes the state to **Success** or **Fail**. If no result is received within 30 minutes, the state changes to **Fail**. ● **Success**: The log collection is successful. If the device returns a result indicating that the log collection succeeded, the state changes to **Success**. ● **Fail**: The log collection fails. The state changes to **Fail** if the IoT platform does not deliver the log collection instruction to the device within 25 hours, the IoT platform delivers the log collection instruction to the device but does not receive a processing result from the device within 30 minutes, or the device returns the result indicating that the log collection failed.
Reset Pre-secret	The pre-secret is used for authentication when a device connects to the IoT platform. It is used to encrypt the transmission channel between NB-IoT devices and the IoT platform, and to authenticate devices integrated with the AgentLite SDK when they attempt to access the IoT platform. After the pre-secret is reset, the new pre-secret must be updated on the device, and the device must carry the new pre-secret for authentication during IoT platform connection.
Reset Secret	The secret is used for authentication when devices call the MQTT interface to connect to the IoT platform. After the secret is reset, the new secret must be updated on the device, and the device must carry the new secret for authentication during IoT platform connection.

Function	Description
Move Between Applications	You can migrate the device to other authorized applications. For example, you can move device A from application X to application Y. The following conditions must be met to ensure that a device can be migrated and services on the migrated device are normal: ● If the device has been registered with the IoT platform, the device to be migrated is an NB-IoT device. ● The source and target applications belong to the same user. ● The source application has been authorized to the target application. ● The target application has the profile file corresponding to the device. ● The power saving mode of the target application is the same as that of the source application.
Add Device Mode (NWI)	This function is used to connect a device to the IoT platform through a gateway. This function applies only to the Agent access mode. Only one device can be added each time. After this function is enabled, it is automatically disabled if no device is added within 180 seconds. If a device is added within 180 seconds, you can manually disable the function. Then you can enable the function to add another device. You can view the added devices on the **Sub Devices** tab page.
Connect to Platform	This function displays access information about the bootstrap service and can be enabled by the IoT platform operations administrator. The bootstrap service enables the bootstrap server to automatically assign the IoT platform's IP address to an NB-IoT device so that the device can register with the IoT platform for unified management. ● **IoT platform IP**: IPv4 or IPv6 address for the device to connect to the IoT platform. ● **Service Server ID**: index of the bootstrap server. ● Security Connection Mode – **PSK**: Devices connect to the IoT platform in DTLS mode. – **Non-secure**: Devices connect to the IoT platform in non-encrypted mode. – **Optimized PSK**: Devices connect to the IoT platform in DTLS+ mode. ● **Device Registration Lifecycle (s)**: time at which an NB-IoT device needs to re-register with the IoT platform. The bootstrap server delivers the information to the device. When the time expires, the device sends a reregistration request to the IoT platform. The IoT platform does not process this parameter.

1.3.3.2.9 Commands

On the device list, click a device to open the **Device Details** page.

- On the **Commands** page, you can deliver commands to an individual device. Specifically, click **Send Command**, select a command and set command parameters in the dialog box displayed, and click **Confirm**.

  NOTE

  - IoT Device Management can deliver commands only to NB-IoT devices. To deliver commands to MQTT devices, call the IoT platform APIs.
  - The commands can be delivered only after being defined in the product model.
  - The parameters with the asterisk (*) are mandatory. The parameters displayed vary by command.

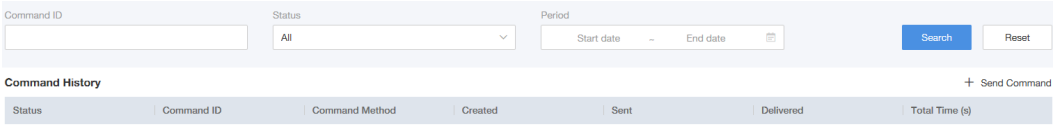
- The **Commands** tab page displays historical commands delivered by the IoT platform or NA to the device. On this page, you can view information such as the creation time of the command delivery task, the time at which the IoT platform sent the command, the time at which the command was delivered, and the command status. This information helps you learn the command execution status. The task status transition is listed in **Table 1-10**. If there are many historical commands, you can filter them by **Command ID**, **Status**, and **Period**.

Table 1-10 Task status transition

Status	Description
Pending	● For an NB-IoT device that uses the pending delivery mode, the IoT platform caches a command if the device does not report data. The task status is **Pending**. ● This status does not exist for an NB-IoT device that uses the immediate delivery mode. ● This status does not exist for an MQTT device.
Expired	● For an NB-IoT device that uses the pending delivery mode, if the IoT platform does not deliver a command to the device within the specified expiration time, the task status is **Expired**. The expiration time is subject to the value of **expireTime** carried by the NA. If **expireTime** is not carried, the default value (48 hours) is used. ● This status does not exist for an NB-IoT device that uses the immediate delivery mode. ● This status does not exist for an MQTT device.
Canceled	If you cancel a pending task, the task status is **Canceled**.
Sent	● For an NB-IoT device that uses the pending delivery mode, the IoT platform sends the cached command when receiving data reported by the device. In this case, the task status changes from **Pending** to **Sent**. ● For an NB-IoT device that uses the immediate delivery mode, if the device is online when an IoT platform issues a command, the task status is **Sent**. ● If an MQTT device is online when the IoT platform issues a command, the task status is **Sent**.
Timed out	If the IoT platform does not receive a response within 180 seconds after issuing a command to an NB-IoT device, the task status is **Timed out**. This status does not exist for an MQTT device.

Status	Description
Delivered	If the IoT platform receives a response from a device, the task status is **Delivered**.
Successful	If the IoT platform receives a result indicating that the command is successfully executed, the task status is **Successful**.
Failed	● If the IoT platform receives a result indicating that the command fails to be executed, the task status is **Failed**. ● For an NB-IoT device that uses the immediate delivery mode, if the device is offline when an IoT platform issues a command, the task status is **Failed**. ● If an MQTT device is offline when the IoT platform issues a command, the task status is **Failed**.

If the NA calls the batch task creation API of the IoT platform to deliver control instructions to devices in batches, you can view the task execution status and result on the **Command Delivery** page.



1.3.3.2.10 Sub Devices

On the device list, click a device to open the **Device Details** page.

The **Sub Devices** tab page presents devices (sensors) connected to the IoT platform through gateways. You can view the status, device ID, and device type of a sub device.

NOTE

The status of a sub device indicates the access status to a gateway, and the gateway reports the status to the IoT platform for updating. If the gateway cannot report the status of a sub device, the sub device status is not updated on the IoT platform. For example, after a sub device connects to the IoT platform through a gateway, the sub device status is displayed as online. If the gateway is disconnected from the IoT platform, the gateway can no longer report the sub device status. Therefore, the sub device status remains online.

On the **Sub Devices** tab page, you can click a sub device to view its details. For details, see **Information**, **Historical Data**, and **Operations**.

1.3.3.3 Registration

Register a device on the IoT platform and define device parameters. Then the device can connect to the IoT platform if authentication succeeds.

The IoT platform supports individual registration and batch registration. NAs can also call the registration API to register an individual device on the IoT platform. Currently, batch device registration by using an API is not supported.

NOTE

After the registration API is called to register a device or a batch device registration task is created on the Management Portal, the IoT platform automatically deletes a device if it does not connect to the IoT platform within a specified duration.

- When the registration API is used, the duration is specified by the **timeout** parameter. If the parameter is set to **0**, the registered device information is permanently valid. If the parameter is not set, the default value is used.
- For individual device registration on the Management Portal, the registered device information is permanently valid.

Registering an Individual Device

Step 1 Choose **Devices** > **Registration**.

Step 2 Click the **Individual Registration** tab, and then click **Register**. In the dialog box displayed, set the parameters based on **Table 1-11**, and click **Confirm**.

Table 1-11 Individual device registration parameters

Parameter	Configuration Rule
Product	Select a product. You can select a product only after it is defined on the **Product Models** page. If the product model has not been uploaded, upload or create it first.
Node ID	Specify the unique physical identifier of a device, such as its IMEI or MAC address. This parameter is carried during device access and used by the IoT platform to authenticate the device. ● For a native MQTT device, the device ID (corresponding to the node ID) and secret generated after the registration are used for IoT platform connection. ● For an NB-IoT device or a device integrated with the AgentLite SDK, the node ID and pre-secret entered during the registration are used for IoT platform connection.
Pre-secret	● For an NB-IoT device, the pre-secret is used to encrypt the transmission channel between it and the IoT platform. ● For a device integrated with the AgentLite SDK, the pre-secret is used by the IoT platform to authenticate its access. ● A native MQTT device does not require a pre-secret.
Confirm Pre-secret	Enter the pre-secret again.

----End

Registering a Batch of Device

Step 1 Choose **Devices** > **Registration**.

Step 2 Click the **Batch Registration** tab page, and then click **Create**. In the dialog box displayed, enter the task name, select a product, upload the batch registration file, and click **Submit**.

Table 1-12 Parameters in the batch registration file template

Parameter	Value
Product	Select a product. You can select a product only after it is imported on the **Product Models** page. If the product model has not been uploaded, upload or create it first.
nodeId	Specify the unique physical identifier of a device, such as its IMEI or MAC address. This parameter is carried during device access and used by the IoT platform to authenticate the device. ● For a native MQTT device, the device ID (corresponding to the node ID) and secret generated after the registration are used for IoT platform connection. ● For an NB-IoT device or a device integrated with the AgentLite SDK, the node ID and pre-secret entered during the registration are used for IoT platform connection.
preSecret	● For an NB-IoT device, the pre-secret is used to encrypt the transmission channel between it and the IoT platform. ● For a device integrated with the AgentLite SDK, the pre-secret is used by the IoT platform to authenticate its access. ● A native MQTT device does not require a pre-secret.

----End

1.3.4 Groups

Description

A group is a set of devices. Groups are used for batch device operations, such as task delivery, software upgrade, and firmware upgrade. To upgrade software of a certain type of devices, first add them to a group and then select the group.

A device belongs to only one device group.

Managing a Group

Step 1 Choose **Groups**.

Step 2 Click buttons to add, unbind, move, or delete a group.

Table 1-13 Buttons on the Groups page

Icon	Description
	Adds a root group. The group name and description need to be specified.
	Adds a child group. The group name and description need to be specified.

Icon	Description
	Unbinds a child group from a parent group.
	Moves a child group to a different parent group.
	Deletes the selected group. The deletion operation cannot be undone. The default group cannot be deleted. **NOTE** Deletion is classified into the following types: ● **Cascade deletion**: When a parent group is deleted, its child groups are automatically deleted. ● **Individual deletion**: After a parent group is deleted, its child groups become root groups. If multiple child groups exist, the corresponding number of root groups is generated. For example, parent group A has child groups B and C. If parent group A is deleted, child groups B and C become independent root groups.
	Exports the topology of all groups. This button is displayed on the right of the page when you click **All Groups**.

----End

Binding/Unbinding a Device

After creating a group, you can bind a device to or unbind a device from the group, or move a device between groups.

Step 1 Choose **Groups**.

Step 2 Select a group, click the **Device** tab, and click buttons to bind, unbind, or move devices.

Table 1-14 Buttons on the Device tab page

Icon	Description
	Binds a device to a group.
	Unbinds a device from a group. **NOTE** After an SP user with the required permission or a global user unbinds a device from a group that has multiple devices bound, the user does not have the permission to view other devices under the group. In other words, the group and devices are no longer available to the user after unbinding. (A common user cannot unbind a device from a group.)
	Moves a device to another group.

----End

1.3.5 Alarms

The **Alarms** page displays only alarms defined in **rules**. You can manage the device status based on the defined rules. Pay close attention to device alarms and handle the alarms quickly to ensure that devices run normally. You can also search for historical alarms of devices.

For example, if a smart water meter does not report data for three consecutive days, the IoT platform generates an alarm to notify maintenance personnel of the water meter fault. Maintenance personnel then locate the faulty water meter based on the alarm information and repair it promptly.

Alarm severity and handling suggestions are as follows:

- Critical: Customer services are interrupted or devices may become unavailable. Measures must be taken immediately to rectify the fault.
- Major: Devices are partially affected or the system performance is affected. Corrective measures must be taken as soon as possible to prevent more serious faults.
- Minor/Warning: There is no current impact on devices. The system has detected potential or imminent faults that may affect services but services are not yet affected. However, you must check for potential faults.

1.3.6 Batch Tasks

1.3.6.1 Command Delivery

The **product model** defines commands that the IoT platform can deliver to devices. NAs can call the batch task creation API of the IoT platform to deliver commands to devices in batches. This allows you to easily configure or modify device service properties and control the devices.

The **Command Delivery** tab page displays the task execution status and result. If the success rate is not 100%, click the task name to open the task details page and view the failure cause.

1.3.6.2 Device Configuration

Description

On the **Device Configuration** tab page, you can modify the properties of devices in batches.

For the batch configuration of LWM2M devices, the IoT platform provides the **device shadow** to store the modified device properties. After a device goes online, the modified device properties are synchronized to it.

NOTE

Before delivering the configuration, open the **Product Models** page, click **Details** of the product model to which the device belongs, and set **Device Configuration** to **Supported** on the **Maintenance Capability Configuration** tab page.

Procedure

Step 1 Choose **Batch Tasks** > **Device Configuration**, and click **Create**.

Step 2 Set the parameters in the dialog box displayed.

Parameter	Description	Configuration Rule
Task Name	Name of a batch device configuration task.	Set the parameter based on site requirements. The value can contain a maximum of 50 characters.
Execution Policy	Whether an execution policy is configured.	Set the parameter based on site requirements. By default, this parameter is selected.
Execution Type	Time at which the task runs.	● **Now**: The task runs immediately after being created. ● **Device online**: The task runs when the IoT platform is interconnected with the device. ● **Custom**: You can customize the start time and end time. **Start date** and **End date** are available only if **Execution Type** is set to **Custom**.
Retry Type	Whether the IoT platform automatically retries the task upon failure. The default value is **No**.	● **No**: The IoT platform does not retry the failed task. ● **Custom**: The IoT platform retries the task a specified number of times.
Retry Attempts	This parameter is available only if **Retry Type** is set to **Custom**.	The value ranges from 1 to 10.
Retry Interval (s)	This parameter is available only if **Retry Type** is set to **Custom**.	The maximum interval is 1200s.

Step 3 Click **Next** to enter the next configuration page. Set the parameters.

Step 4 Select the device group to which the configuration is to be delivered and click **Next** to enter the next page. Click  to upload the device configuration file in JSON format.

NOTE

Prepare a configuration file as follows:

- If a configuration file has been delivered, select **Devices** > **Device List**, click  in the row where the device of the same type resides. In the **Device Configuration** dialog box, download the configuration file that was delivered last time, modify the file, and import the file again.
- If no configuration file has been delivered, manually create a configuration file. The configuration items that can be delivered and modified must be consistent with the capabilities supported by the product model, and the configuration file must be in JSON format. For example:

```
{    "sensitivity": "0",
 "dataReportInterval": "20"
}
```

Step 5 Click **Submit** to complete the task creation.

Step 6 The task execution status and result are displayed. If the success rate is not 100%, click the task name to open the task details page and view the failure cause.

----End

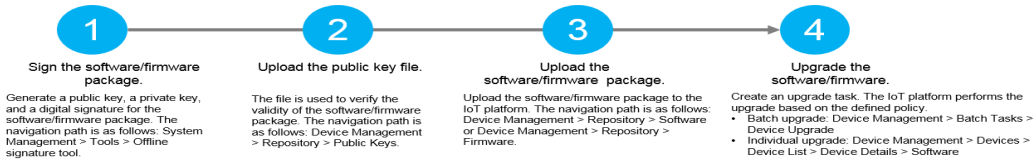
1.3.6.3 Device Upgrade

Description

The IoT platform supports batch software or firmware upgrades. You can upgrade software or firmware of multiple devices simultaneously. You can also query a created upgrade task and its details, such as the status and success rate.

Only software and firmware of NB-IoT devices can be upgraded.

Figure 1-11 Software or firmware upgrade



NOTICE

During a device upgrade, do not deliver other commands to the device. This operation may cause the device upgrade to fail. To avoid delivering any commands during the device upgrade, the NA can call northbound APIs of the IoT platform to subscribe to the device upgrade status.

Prerequisites

- You have uploaded the software and firmware upgrade packages by following the instructions provided in **Firmware** and **Software**.
- Devices have been registered. To upgrade the devices in batches, create a **group** and add the devices to the group.

Software Upgrade

Step 1 Click the **Software Upgrade** tab, and click **Create** in the upper right corner.

Step 2 Set the parameters based on **Table 1-15**.

Table 1-15 Software upgrade parameters

Parameter	Description	Configuration Rule
Task Name	Name of a software upgrade task.	Set the parameter based on site requirements. The value can contain a maximum of 50 characters.
Execution Type	Time at which the task runs.	● **Now**: The task runs immediately after being created. ● **Device online**: The task runs when the IoT platform is interconnected with the device. ● **Custom**: You can customize the start time and end time. **Start date** and **End date** are available only if **Execution Type** is set to **Custom**.
Retry Type	Whether the IoT platform automatically retries the task upon failure.	Set the parameter based on site requirements. The default value is **Custom**. ● **No**: The IoT platform does not retry the failed task. ● **Custom**: The IoT platform retries the task a certain number of times (specified by **Retry Attempts**, ranging from 1 to 10) at a certain interval (specified by **Retry Interval (s)**, with the maximum value of 1200 seconds).
App Confirm	Whether the software upgrade task is sent to the NA.	Set the parameter based on site requirements. The default value is **No**.

Step 3 Click **Next**, and select the software package version to upgrade.

Step 4 Click **Next**, and select a device or group.

Step 5 Click **Submit**.

- You can view the created tasks and their status on the task list.
- You can click the row of a task to view its details, including the basic task information and device execution details. In the device execution details, you can view the number of task execution times, start time, end time, execution result, and failure cause.
- During the upgrade, you can click  in the row of the task to stop the upgrade task. After a task is stopped, you cannot manually start it. Instead, you must create another upgrade task.

NOTE

If the total number of concurrent software upgrade tasks and firmware upgrade tasks reaches the upper limit (300,000 for each user), you cannot create another upgrade task until:

- An existing upgrade task is completed.
- You stop an existing upgrade task.

----End

Firmware Upgrade

Step 1 Click the **Firmware Upgrade** tab, and click **Create** in the upper right corner.

Step 2 Set the parameters based on **Table 1-16**.

Table 1-16 Firmware upgrade parameters

Parameter	Description	Value
Task Name	Name of a firmware upgrade task.	Set the parameter based on site requirements. The value can contain a maximum of 50 characters.
Retry Type	Whether the IoT platform automatically retries the task upon failure.	Set the parameter based on site requirements. The default value is **Custom**. ● **No**: The IoT platform does not retry the failed task. ● **Custom**: The IoT platform retries the task a specified number of times.
Retry Attempts	This parameter is available only if **Retry Type** is set to **Custom**.	The value ranges from 1 to 10.

Step 3 Click **Next**, and select the firmware package version to upgrade.

Step 4 Click **Next**, and select a device or group.

Step 5 Click **Submit**.

- You can view the created tasks and their status on the task list.
- You can click the row of a task to view its details, including the basic task information and device execution details. In the device execution details, you can view the number of task execution times, start time, end time, execution result, and failure cause.
- During the upgrade, you can click  in the row of the task to stop the upgrade task. After a task is stopped, you cannot manually start it. Instead, you must create another upgrade task.

NOTE

If the total number of concurrent software upgrade tasks and firmware upgrade tasks reaches the upper limit (300,000 for each user), you cannot create another upgrade task until:

- An existing upgrade task is completed.
- You stop an existing upgrade task.

----End

References

Document	
Tools	**Public Keys**
Software	**Firmware**

1.3.7 Rules

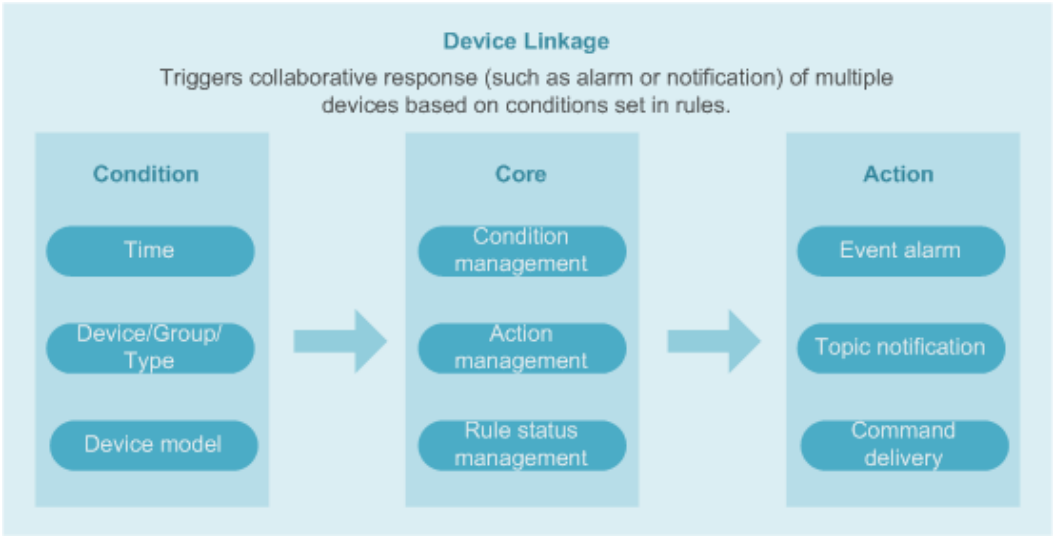
Description

The rule engine function allows you to set rules for devices connected to the IoT platform. If the conditions set in a rule are met, the IoT platform triggers the corresponding action. Device linkage and data forwarding rules can be created.

Device linkage rule

Device linkage is triggered by condition. Based on preset rules, the IoT platform triggers collaborative response of multiple devices to implement device linkage and intelligent control. If **Topic notification** is selected for **Action Type** in a rule, the IoT platform works with the **Simple Message Notification (SMN)** service of HUAWEI CLOUD to set and deliver topic notification messages.

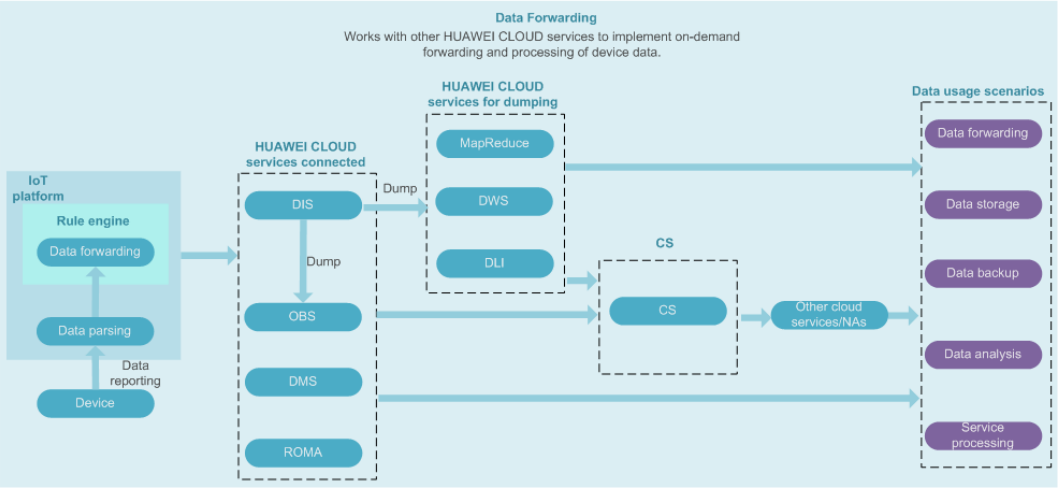
For example, when the battery level of a water meter is less than or equal to 20%, a low-battery alarm is reported. In this way, you can replace the battery in time.



Data forwarding rule

Data forwarding works with other HUAWEI CLOUD services to implement on-demand forwarding and processing of device data. You do not need to purchase servers to store, calculate, and analyze device data.

- Works with Data Ingestion Service (DIS) (available soon) to enable efficient data collection, transmission, and distribution. You can download data by using the SDKs or APIs provided by DIS. You can also use dump tasks to forward data to **Object Storage Service (OBS)**, **MapReduce**, **Data Warehouse Service (DWS)**, and **Data Lake Insight (DLI)** for subsequent data processing, such as data storage and analysis.
- Works with **DMS** to provide message queues for device data. DMS is a message middleware service based on distributed, highly available clusters. The IoT platform functions as a producer to send messages to the DMS message queue. Your applications consume messages from the message queue. In this way, messages can be transmitted between multiple application components.
- Works with **OBS** to persistently store device data. (The IoT platform can store device data for 7 days). OBS is an object-based massive storage service that provides massive, secure, reliable, and low-cost data storage capabilities. It can archive, back up, and store data reported by devices. OBS can work with Cloud Stream (CS, available soon) to analyze data flows in real time. The analysis result is used for data visualization for other cloud services or third-party applications.
- Works with the Message Queue Service (MQS) component of ROMA – Enterprise Business Integration Platform (available soon) to provide a secure, standard message channel between the IoT platform and NAs. MQS is enterprise-level message middleware that uses Kafka and a unified message access mechanism. It provides basic and advanced functions to offer a unified message channel for enterprise data management. The basic functions include message publishing and subscription, topic management, user permission management, resource statistics, monitoring and alarming. The advanced functions include message tracking, network isolation, and integration of cloud and on-premises applications.



Creating a Device Linkage Rule

Step 1 Click **Create** in the upper right corner, and click **Device Linkage** in the dialog box displayed.

Step 2 Set the parameters based on **Table 1-17**.

Table 1-17 Device linkage rule parameters

Parameter	Description	Example Value
Rule Name	Name of a rule to be created.	Create a rule as follows: ● **Rule Name**: **test** ● **Activate Now**: **Yes** ● **Validity Period**: **Always** ● **Device Behavior**: Water meter temperature reaching 35°C ● **Send Notification or Alarm**: major alarm After the rule is created, a major alarm is reported when the temperature of the water meter reaches 35°C.
Activate Now	Whether the rule is triggered immediately if the last data reported before the rule creation meets the condition. ● **Yes**: After a rule is created, the rule takes effect immediately. The IoT platform checks the last reported data to determine whether to trigger the rule. ● **No**: After a rule is created, the IoT platform checks only subsequently reported data to determine whether to trigger the rule.	
Rule Type	**Cloud Rule**: a rule for devices that are directly connected to the cloud IoT platform **Edge Rule**: a rule for devices that access the cloud IoT platform through edge nodes	
Edge Node	This parameter is available when **Rule Type** is set to **Edge rule**. Select the edge node to which the rule applies.	
Validity Period	● **Always**: There is no time limit. The IoT platform always checks whether conditions are met. ● **Custom**: You can select a time segment during which the IoT platform checks whether conditions are met.	

Parameter	Description	Example Value
If	**Device Behavior**: You can click **Add** to add a device, device type, or device group as a condition. ● Condition Type – **Device**: Data that meets the conditions is reported by a device. – **Device type**: Data that meets the conditions is reported by devices of the specified type. – **Device group**: Data that meets the conditions is reported by devices in the specified group. ● **Select Device Model**: Select the model of the device that reports the data meeting the conditions. After selecting a device model, select the corresponding service type and set the data reporting rule. ● **Data Validity Period (s)**: For example, when **Data Validity Period** is set to 30 minutes, a device generates data at 19:00, and the IoT platform receives the data at 20:00, the action is not triggered regardless of whether the conditions are met. ● **Delay (min)**: delay for triggering an action after the condition is met. The default value is **0**. For example, an alarm indicating an unlocked door is triggered when the door status sensor is turned on. Upon detecting that the door status sensor is activated, an alarm is generated immediately if the default value is used. **Time**: Click **Add** to set the time at which the rule is triggered. It is usually used for periodic triggering conditions, such as turning off street lamps at 07:00 every day. ● **Start Time**: start time for triggering a rule. ● **Repeated Triggering Attempts**: number of times that the rule can be triggered. The value ranges from 1 to 1440. ● **Interval (min)**: interval for triggering the rule after the start time. The value ranges from 1 to 1440.	

Parameter	Description	Example Value
Then	**Device Behavior**: You can click **Add** to add the action to be triggered if the conditions are met. ● **Action Type**: type of an action. The default value is **Device**, indicating that a command is issued to a device. ● **Select Device Model**: Select the device model corresponding to the action and the device that executes the action in the model. After selecting a device model, select the corresponding service type. ● **Command Status**: whether the action is valid. The default value is **Enabled**. – **Enabled**: The action is valid. The action is executed when the condition is met. – **Disabled**: The action is invalid. The action is not executed when the condition is met. ● **Command Request ID**: ID of the command to be delivered. ● **Callback URL**: URL to receive a notification if the command status changes, such as failed, successful, timed-out, sent, or delivered. ● **Command Expiration (s)**: validity period of a command, in units of seconds. If **Command Expiration** is set to **0**, the command is delivered immediately. If **Command Expiration** is set to other values, the command is cached before being delivered. If **Command Expiration** is not carried, the default value (48 hours) is used. **Send Message or Alarm**: You can click **Add** to set a theme notification or event alarm. 1. **Theme notification**: You must enable SMN before configuring the theme content on this page. a. Choose **System Management** > **Cloud Service Configuration** to configure connection with HUAWEI CLOUD. For details, see **Cloud Service Configuration**. b. On the current page, click the SMN link and go to the HUAWEI CLOUD website to enable SMN. c. On the current page, select the region where SMN is located, select the theme name, and set the message title and message content. 2. **Event alarm**: Define the alarm type, alarm severity, alarm name, and alarm content. If the conditions are met, an alarm is displayed on the **Alarms** page. **NOTE** If **Event alarm** is selected, **Time** under **If** cannot be specified.	
Description	Description of the rule.	

Step 3 Click **Submit**.

The newly created rule is in the activated state by default. You can disable a rule in the **Status** column on the rule list.

----End

Creating a Data Forwarding Rule

Step 1 Click **Create Rule** in the upper right corner, and click **Data Forwarding** in the dialog box displayed.

Step 2 Set the parameters based on **Table 1-18**.

Table 1-18 Data forwarding rule parameters

Parameter	Description	Example
Rule Name	Name of a rule to be created.	Create a rule as follows: ● **Rule Name**: **test** ● **Object Type**: **All devices** ● **Action Type**: **DIS** ● **Region**: **cn-north-1** ● **Channel**: **dis-DMPLiteTest** ● **Data Type**: **JSON** After the rule is created, the IoT platform forwards data in JSON format to DIS in cn-north-1 through dis-DMPLiteTest.
If	● **Object Type**: Select **All devices** (only data forwarding for all devices is supported). ● **Add Filter**. By default, the function is disabled. If the function is enabled, you must specify **Property Name**, **Operate**, and **Value**. The IoT platform forwards the packets that meet the filter criteria. For details, see the **data forwarding example**.	
Then	● **Action Type**: The value can be **DIS** (available soon), **DMS**, **OBS**, or **ROMA** (available soon). **NOTE** If you have not enabled the service, perform the following steps: 1. Choose **System Management** > **Cloud Service Configuration** to configure connection with HUAWEI CLOUD. For details, see **Cloud Service Configuration**. 2. On the current page, click a service link to go to the HUAWEI CLOUD website to enable the service. ■ **DMS** (available soon): Data can be forwarded to common queues and advanced queues that support logical multi-tenant. Data forwarding to RabbitMQ instances with physical multi-tenant and Kafka instances is not supported. ■ ROMA (available soon): You must subscribe to ROMA and configure the connection with ROMA by following the instructions provided in **Subscription to Push Service (ROMA)**. ● **Forward To**: On the page (shown in **#li1324241203120**) for creating a data forwarding rule, select the region, channel, data type, and other information of the service to be forwarded.	

Parameter	Description	Example
Description	Description of the rule.	

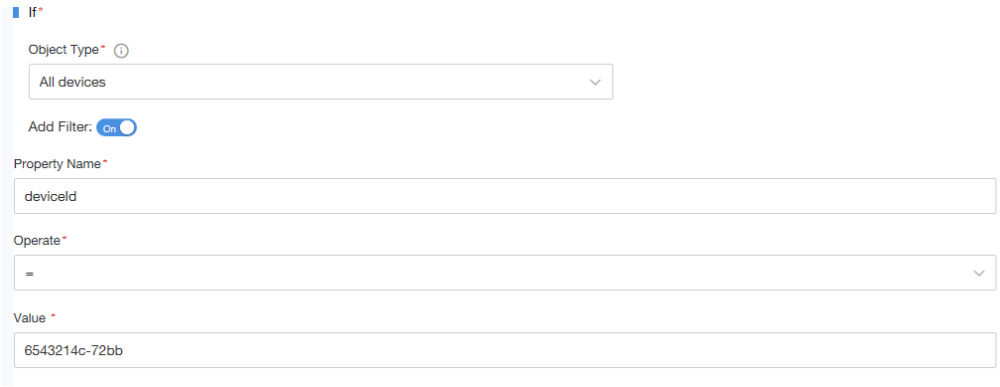
- Data forwarding example:

  The IoT platform parses and matches device data in JSON format after encoding and decoding.

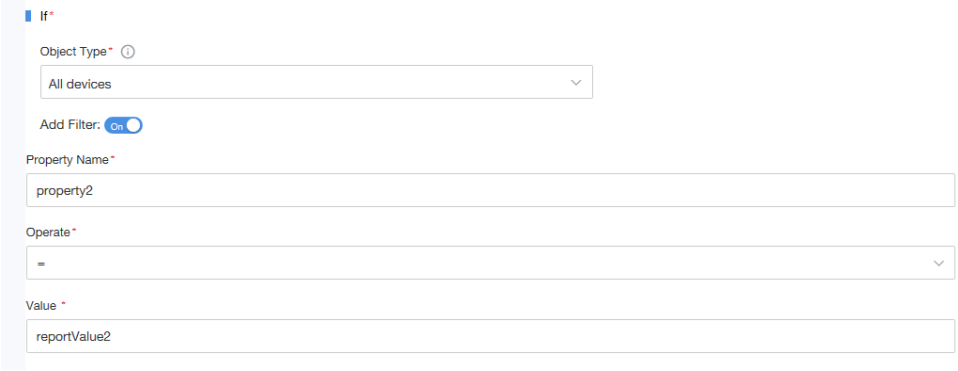
```
{
       "deviceId": "6543214c-72bb-4131-9e14-fad974123456",
       "services": [{
              "serviceId": "serviceId1",
              "data": {
                     "property1": "reportValue1",
                     "property2": "reportValue2",
                     "property3": "reportValue3"
              },
              "eventTime": "20190425T021157Z"
       }]
}
```

  Adding filter criteria for data forwarding

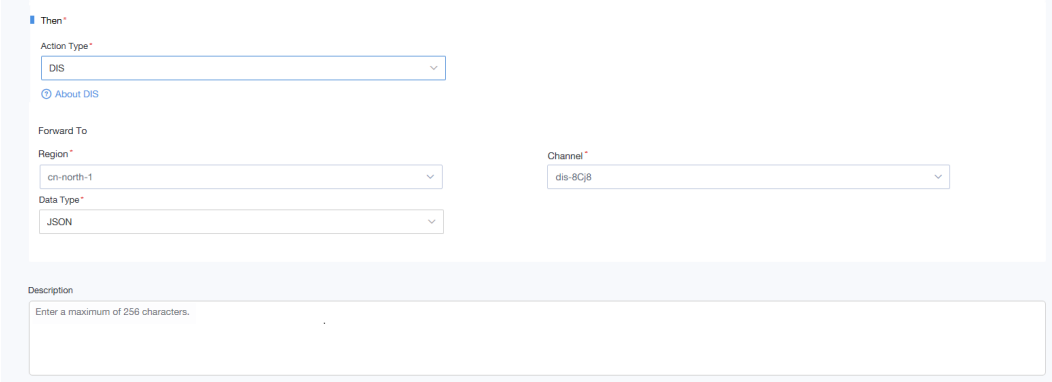
  - Filter data based on deviceId and forwards data of a device



  - Filter data based on the service ID or property1/property2/property3 to forward batch device data.



- **Data forwarding rule**



Step 3 Click **Submit**.

The newly created rule is in the activated state by default. You can disable a rule in the **Status** column on the rule list.

----End

Comparison of Data Forwarding Solutions

In many scenarios, you need to process data reported by devices to the IoT platform or use the data for service applications. The IoT platform provides the subscription push and rule engine data forwarding functions to forward device data.

- The rule engine data forwarding function provides basic data filtering capabilities. It enables the IoT platform to filter device data and forward data to other HUAWEI CLOUD services.
- Subscription push: An NA can subscribe to service data of a device on the IoT platform. When the service data changes (for example, the device is registered, the device reports data, and the device status changes), the IoT platform can push a change notification to the NA. Device messages can be quickly obtained and no data filtering is available. The function is simple but easy to use and efficient.

Solution	Application Scenario	Advantages and Disadvantages
Data forwarding rule	● Reported device data to the cloud ● Complicated scenario	Advantages: ● Forwards data to other HUAWEI CLOUD services. ● Filters data based on conditions. Disadvantages: Only data reported by devices can be forwarded. Data about device registration and device status change cannot be forwarded.

Solution	Application Scenario	Advantages and Disadvantages
Subscription and push	● Device data to NAs ● Device data receipt	Advantages: ● Device registration, device data reporting, and device status change can be pushed to NAs. Disadvantages: ● No filtering. ● NAs need to implement operations such as storage and analysis of pushed data and cannot use other HUAWEI CLOUD services. The IoT platform provides only weak HTTP push capabilities. Data forwarding rules are recommended for push messages higher than 10 TPS.

1.3.8 Message Tracing List

The message tracing function can be used to quickly locate and analyze faults when a fault occurs in device binding, command delivery, data reporting, device information update, and device monitoring. The IoT platform supports message tracing for NB-IoT and MQTT devices. Up to 10 devices bound to an application can be traced simultaneously.

After creating a message tracing task on the **Message Tracing** tab page of the device details page, you can view all the devices that are being traced on the **Message Tracing List** page. (If you click ■ to stop message tracing, the message tracing record disappears from the list.) If you click the device that is being traced, the **Message Tracing** tab page is displayed. You can locate and analyze the fault based on the messages displayed on the tab page.

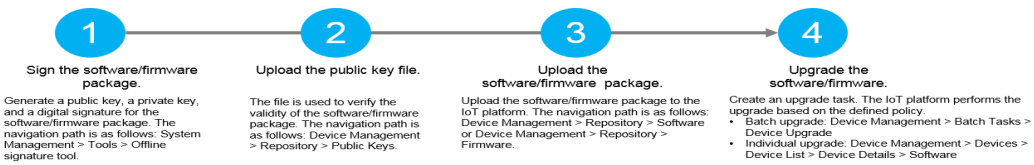
1.3.9 Repository

1.3.9.1 Public Keys

Description

The IoT platform supports loading of device software and firmware packages on the platform and delivering of the packages to devices for upgrade. The IoT platform must **digitally sign** the uploaded software and firmware packages and the corresponding public key file must also be uploaded. The following figure shows the complete software and firmware upgrade process.

Figure 1-12 Software or firmware upgrade



Procedure

Step 1 Choose **Repository** > **Public Keys**, and click **Upload** on the page displayed.

Step 2 Select the public key file generated by using the **offline signature tool** and select the corresponding manufacturer name.

Step 3 Click **Save**.

----End

1.3.9.2 Software

Description

Before upgrading a software package, upload the package to the IoT platform. The following figure shows the complete software and firmware upgrade process.

Figure 1-13 Software or firmware upgrade

The IoT platform automatically loads the existing software package and displays it on the **Software** page. You can click a software package to view its details and download or delete it.

Prerequisites

- You have imported a product model by following the instructions provided in **Product Models**.
- You have digitally signed the software package by following the instructions provided in **Tools**.
- You have uploaded a public key file by following the instructions provided in **Public Keys**.

Procedure

Step 1 Choose **Repository** > **Software**, and click **Upload** on the page displayed.

Step 2 Click and select the target software package on your local PC.

Step 3 Click **OK**.

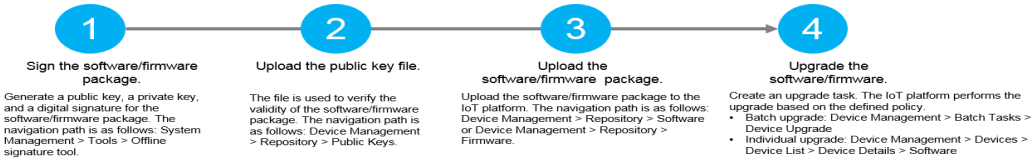
----End

1.3.9.3 Firmware

Description

Before upgrading a firmware package, upload the package to the IoT platform. The following figure shows the complete software and firmware upgrade process.

Figure 1-14 Software or firmware upgrade



The IoT platform automatically loads the existing firmware package and displays it on the **Firmware** page. You can click a firmware package to view its details and download or delete it.

Prerequisites

- You have imported a product model by following the instructions provided in **Product Models**.
- You have digitally signed the firmware package by following the instructions provided in **Firmware**.
- You have uploaded a public key file by following the instructions provided in **Public Keys**.

Procedure

Step 1 Choose **Repository** > **Firmware**, and click **Upload** on the page displayed.

Step 2 Specify the parameters, and click **Save**.

----End

1.4 System Management

1.4.1 Logs

Operation, security, personal data query, and business logs can be used to audit user operations and diagnose faults.

Table 1-19 Log type description

Log Type	Description
Operation log	Records operations on applications, for example, saving a configuration, restoring default settings, uploading a logo, and querying details. Records operations on resources, for example, importing a product model, viewing dashboards, and querying device details.
Security log	Records user login, user logout, and session timeout.

Log Type	Description
Personal data query	Records queries for the user list and enterprise list.
Business log	Records service-related operations such as gateway creation, gateway login, and sensor creation.

1.4.2 Tools

The signature tool is used to set and verify digital signatures of software packages. A digital signature, also called public key digital signature or electronic seal, functions similarly to a physical signature. A digital signature is implemented by public key cryptography to authenticate digital information. A set of digital signatures is typically defined for two supplementary types of operations: one for signing and the other for verification.

NOTE

Public keys are centrally managed on the Huawei platform, whereas private keys are stored and managed using the manufacturer's own system. To enhance the security of private keys, a password for private key encryption must be entered during private key production. If the password or private key is lost or discovered by others, there is no way to upgrade the package signature mechanism, and the manufacturer is liable for resulting security issues.

Downloading the Signature Tool

Step 1 Log in to the portal and choose **System Management** > **Tools**.

Step 2 Click  to download the tool.

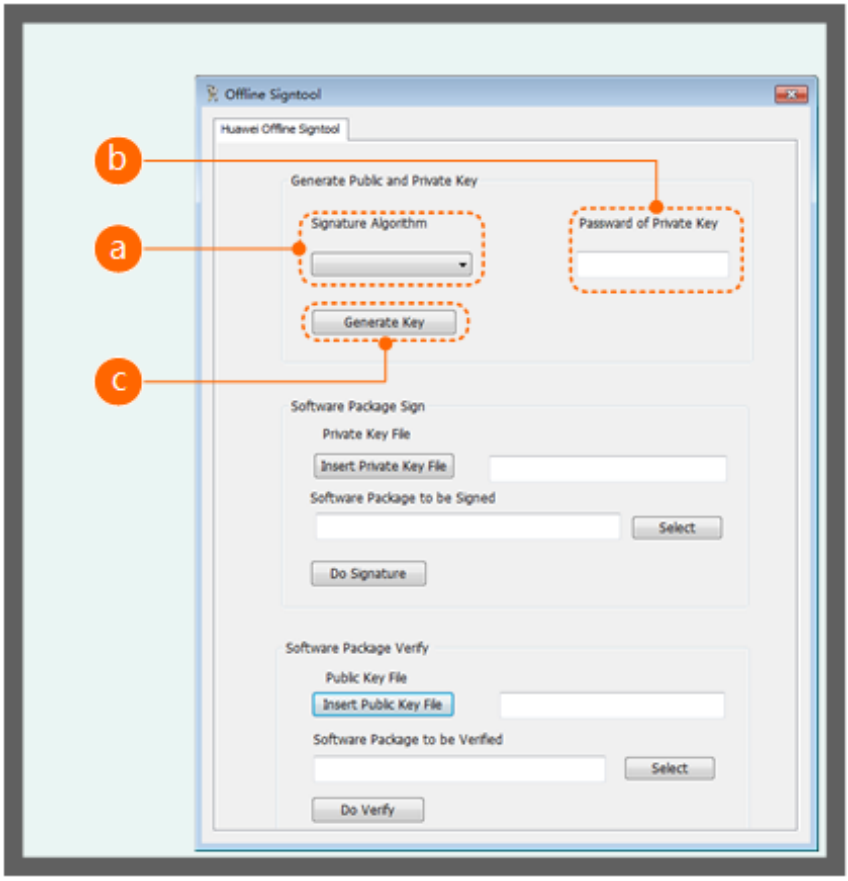
----End

Using the Signature Tool

Decompress the downloaded package and double-click **signtool.exe** to start the signature tool.

Step 1 Generate a public-private key pair, as shown in **Figure 1-15**.

Figure 1-15 Generating a public-private key pair



1. Set **Signature Algorithm**.

   Select **ECDSA_256K1+SHA256** from the drop-down menu.

2. Set **Password of Private Key**.

   The password must contain at least six characters of two or more types chosen from the following: A–Z, a–z, 0 - 9, :~`@#$%^&*()-_=+|?/<>[]{},.;'!"

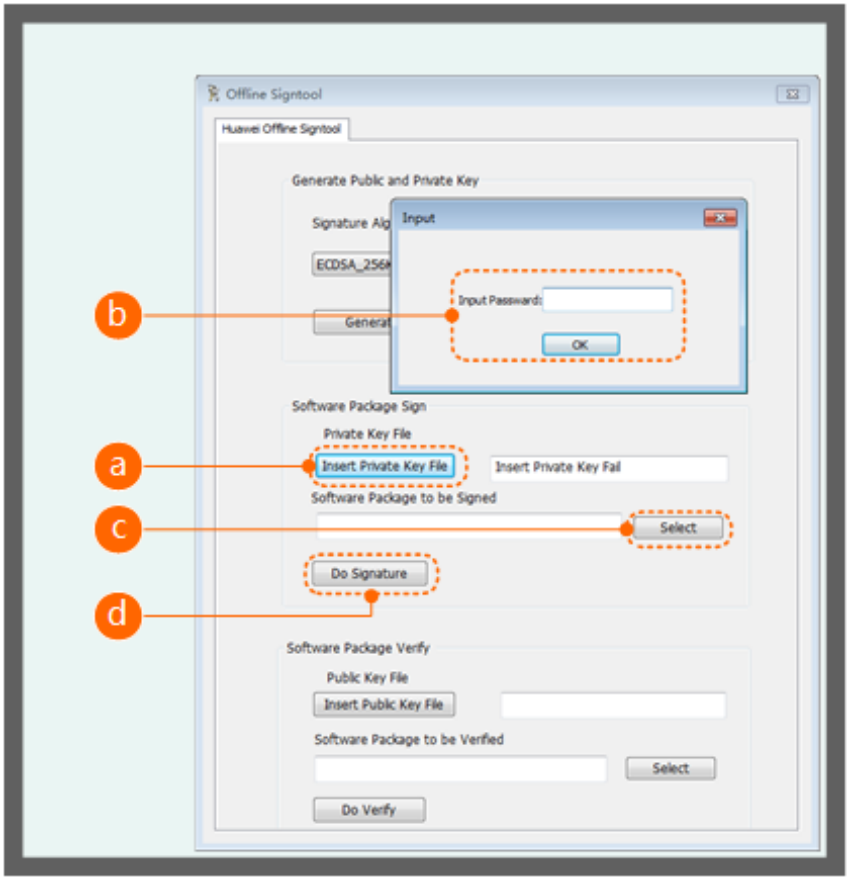
3. Click **Generate Key**.

   In the dialog box displayed, select a path to save the key and click **OK**. The **Success!** dialog box is displayed, indicating that the key is generated.

   The tool generates two key files: **public.pem** and **private.pem**.

Step 2 Set a digital signature for the software package.

Figure 1-16 Setting a digital signature for the software package



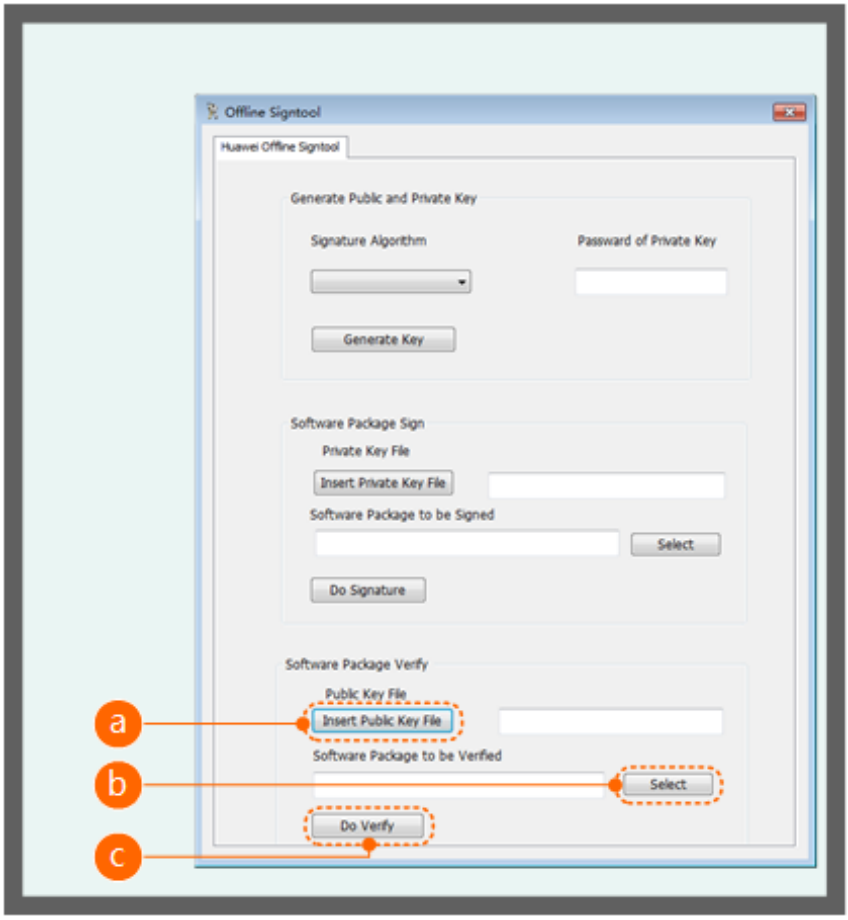
1. Click **Insert Private Key File**, and import the private key file **private.pem** generated in **Step 1.3**.
2. Set **Input Password** to the encryption password entered in **Step 1.2**.

   If the password is correct, the status bar displays the path of the private key file; otherwise, it displays **Insert Private Key File**.
3. Click , and specify the path of the software package to be signed.
4. Click **Do Signature**.

   If the private key has been imported, the software package is signed; otherwise, a dialog box is displayed, asking you to enter the private key.

Step 3 Verify the software package signature.

Figure 1-17 Verifying the software package signature



1. Click **Insert Public Key File**, and import the public key file **public.pem** generated in **Step 1.3**.
2. Click  and specify the path of the software package to be verified.
3. Click **Do Verify**.

   If the public key has not been imported, a dialog box is displayed asking you to enter the public key.

----End

1.4.3 Cloud Service Configuration

The IoT platform can interconnect with HUAWEI CLOUD services. Obtain the access key (AK) and secret access key (SK) from the HUAWEI CLOUD management console to connect the IoT platform with HUAWEI CLOUD. After purchasing other HUAWEI CLOUD services, create **rules** to forward data from the IoT platform to these services.

NOTE

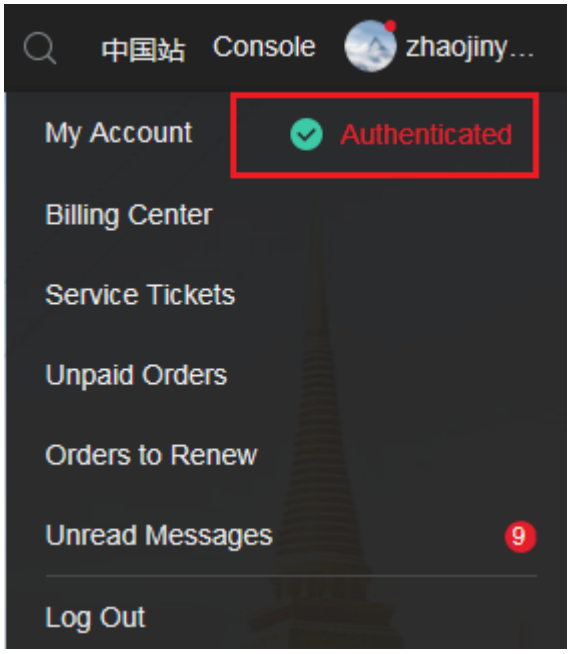
AK: identifies the access key. It is a unique identifier associated with a private access key. The AK and SK are used together to encrypt a request. SK: works with AK to encrypt a request, identify the sender, and prevent the request from being modified.

Obtaining the AK and SK

Step 1 Log in to HUAWEI CLOUD and access the management console.

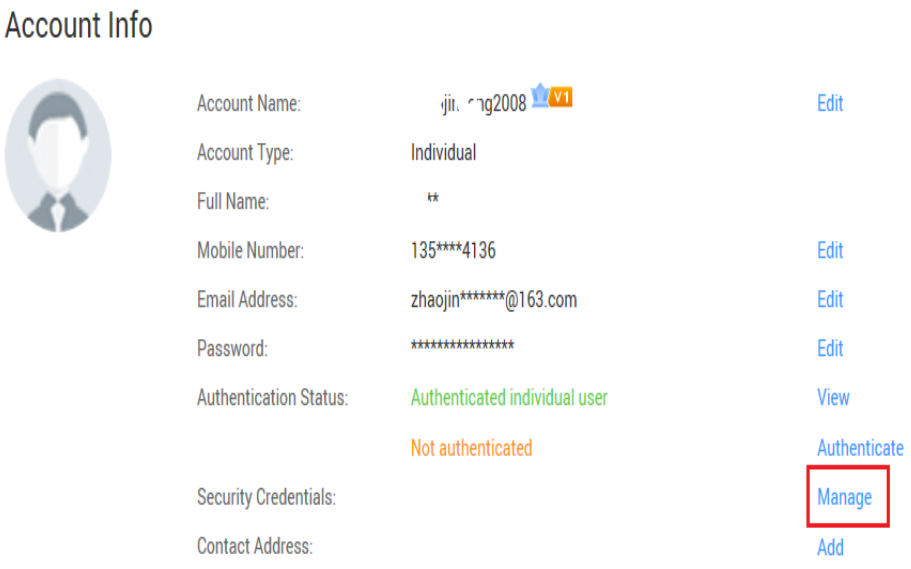
Step 2 Click **Basic Information**.

Figure 1-18 Basic Information



Step 3 In the **Account Info** area, click **Manage**.

Figure 1-19 Manage My Credential



Step 4 On the **My Credentials** page, click the **Access Keys** tab and click **Create Access Key**.

Step 5 Enter the HUAWEI CLOUD login password and the SMS verification code, and click **OK**. The access key information, including the AK and SK, is automatically downloaded.

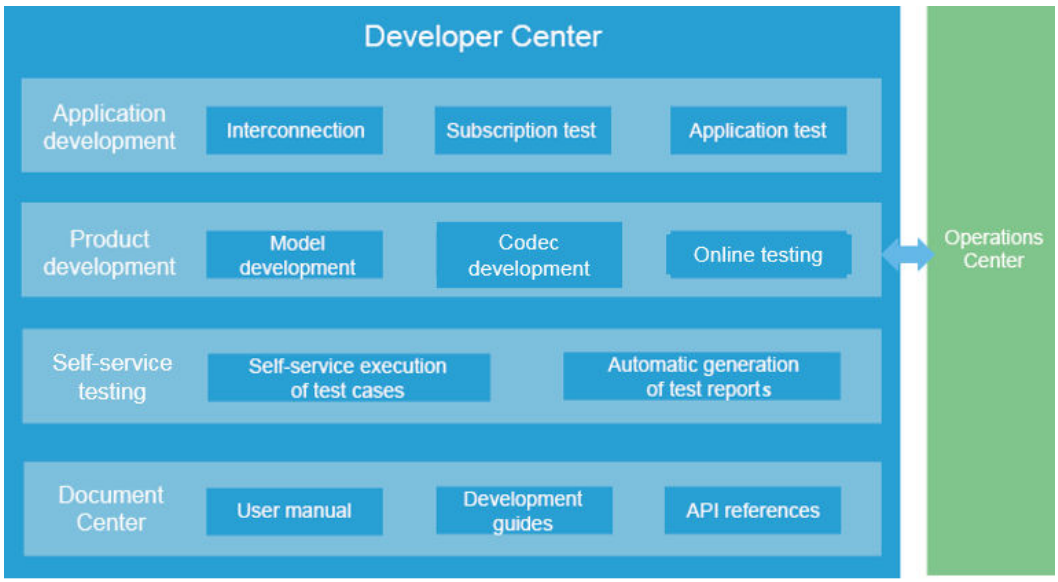
----End

2 Developer Center

2.1 Introduction to the Developer Center

As a one-stop IoT development tool platform, the Developer Center provides a variety of capabilities, such as product development, application development, self-service testing, and product release. It also provides resources such as development guides and API references to help developers improve integration development efficiency and shorten the construction period of IoT solutions.

Architecture of the Developer Center



Functions of the Developer Center

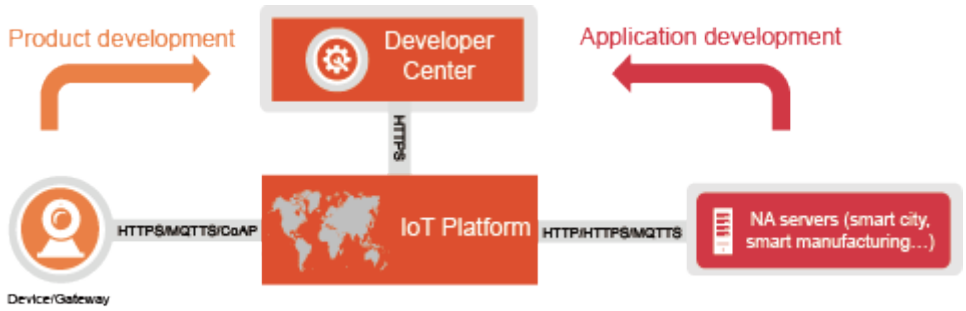
- **Product Development**: provides E2E development guides for profile definition, codec development, and product testing, facilitating quick launch of IoT products.
- **Application Development**: provides interconnection information, subscription tests, and application tests, helping developers develop and test applications and improve the independence of application development.
- **Self-Service Testing**: automatically tests product information, product models, and codecs, generates test reports, and checks whether the products meet the release standards.
- **Document Center**: provides online resources such as user manuals, development guides, and API references to provide real-time help.
- **Product Release**: interconnects with the Operations Center, enabling developers to apply for product release by one click after a product is tested in the Developer Center.

Related Concepts

- **IoT Platform**

  The IoT platform integrates data, device, and operations management to implement unified and secure network access, flexible device adaptation, and data collection and analysis, thereby creating new values. The IoT platform provides open APIs for various industries to help partners quickly develop IoT service applications and meet personalized service requirements of customers. The IoT platform provides access for a variety of devices in wireless and wired access mode.

  As a one-stop development tool platform based on the open capabilities of the IoT platform, the Developer Center helps developers quickly build an IoT platform-based solution.



- **Project**

  A project refers to the resource space of the IoT platform. Developers need to create independent projects based on their own industries before developing IoT products and applications in the project space.

- **Product**

  A collection of devices with the same capabilities or features is called a product. In addition to physical devices, a product includes product information, product models (profiles), codecs, and test reports generated during IoT capability building.

- **Product Model**

  A product model (also called profile) is used to describe the capabilities and features of a device. Developers construct an abstract model of a device by defining a profile file on the IoT platform so that the IoT platform can understand the services, properties, and commands supported by the device.

- **Software Development Kit (SDK)**

  SDK is a set of development tools used by software engineers to create application software for specific software packages, software frameworks, hardware platforms, and operating systems. Generally, the SDK is used in developing applications on the Windows platform. SDK can provide API files for a programming language or complex hardware that communicates with an embedded system.

  The IoT platform provides developers with SDK on the application side and device side to help them quickly integrate applications or devices.

- **Message Queue Telemetry Transport (MQTT)**

  MQTT is an IoT transmission protocol designed for lightweight release/subscription message transmission. It aims to provide reliable network services for IoT devices in low-bandwidth and unstable network environments.

  MQTTS refers to the combination of MQTT and SSL/TLS. The SSL and TLS protocols are used for encrypted transmission.

- **Constrained Application Protocol (CoAP)**

  CoAP is a software protocol designed to enable simple devices to perform interactive communication on the Internet.

  CoAPS refers to CoAP over DTLS. The DTLS protocol is used for encrypted transmission.

- **Lightweight Machine to Machine (LWM2M)**

  LWM2M is an IoT protocol defined by Open Mobile Alliance (OMA). It is mainly applied to NB-IoT devices with limited resources (such as limited storage and power supply).

2.2 Logging In to the Developer Center

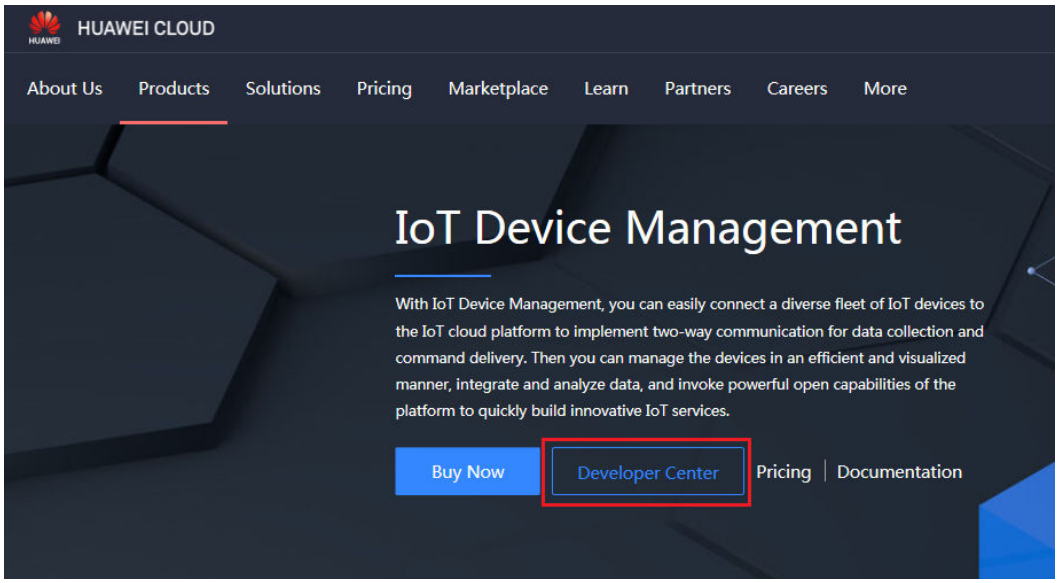
Prerequisites

You have registered a HUAWEI CLOUD account and completed real-name authentication.

Procedure

Step 1 Visit HUAWEI CLOUD, and open the page of **IoT Device Management**.

Step 2 Click **Developer Center**.



Step 3 Click **Access Developer Center**. The Developer Center is automatically displayed.

----End

Browser Requirements

To ensure good display effect and ease of use, use a browser with good compatibility. The table below lists the browser requirements.

Browser Type	Recommended Version	Resolution
Internet Explorer	Internet Explorer 11.0 or later	1366 x 768
Firefox	Firefox 51.0-61.0	
Google Chrome	Google Chrome 58.0-67.0	

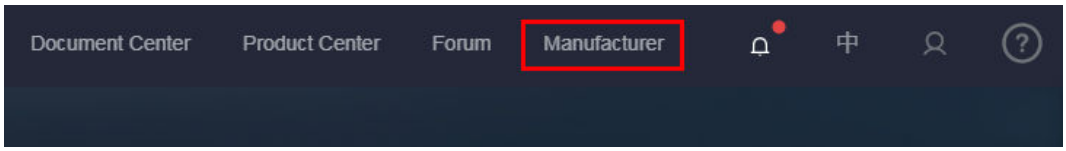
2.3 Manufacturer

Overview

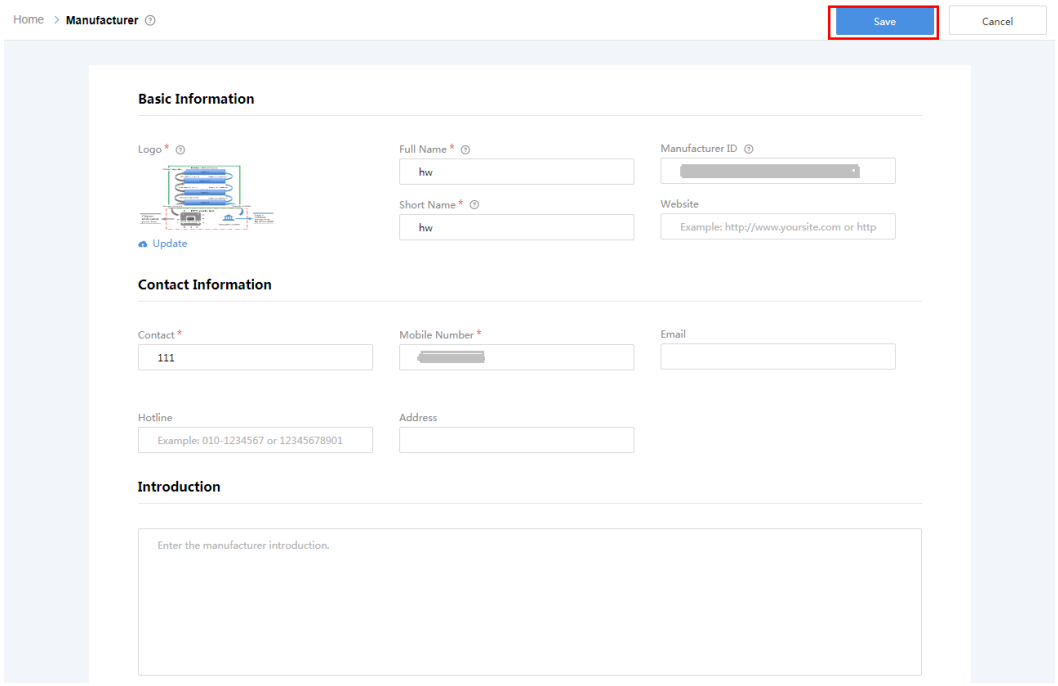
The manufacturer information includes the enterprise logo, name, website, and scale. When a developer accesses the Developer Center for the first time, the developer must supplement the manufacturer information.

Modifying manufacturer information

Step 1 On the home page of the Developer Center, click **Manufacturer** to edit manufacturer information.



Step 2 Supplement the manufacturer information and click **Save**.



----End

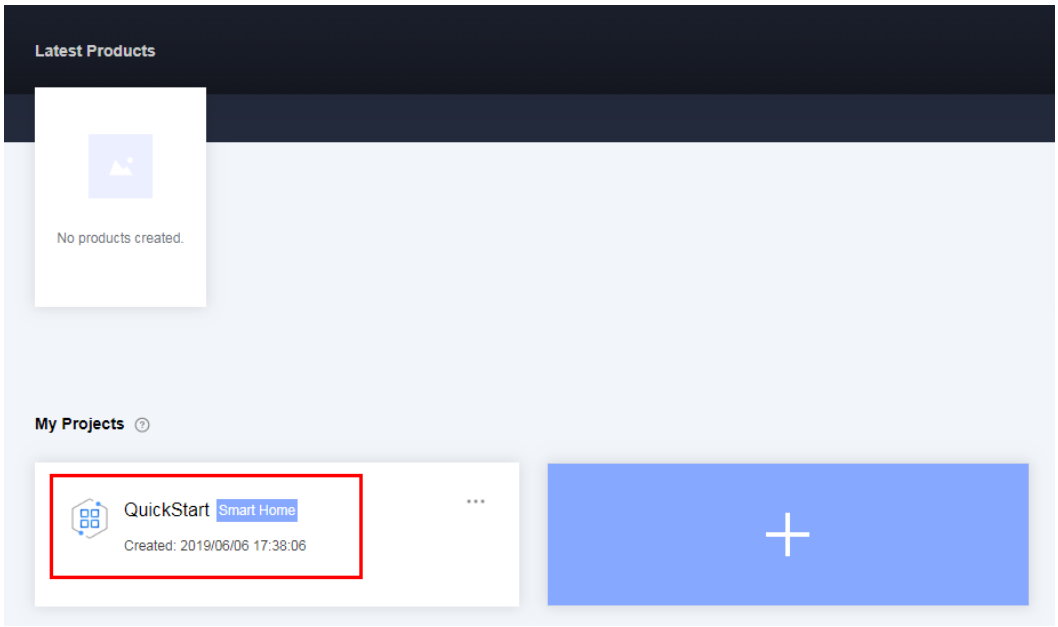
2.4 Project

Overview

Before developing an IoT solution, developers must create an independent project based on their own industry. In the project space, developers can develop IoT products and applications.

Creating a Project

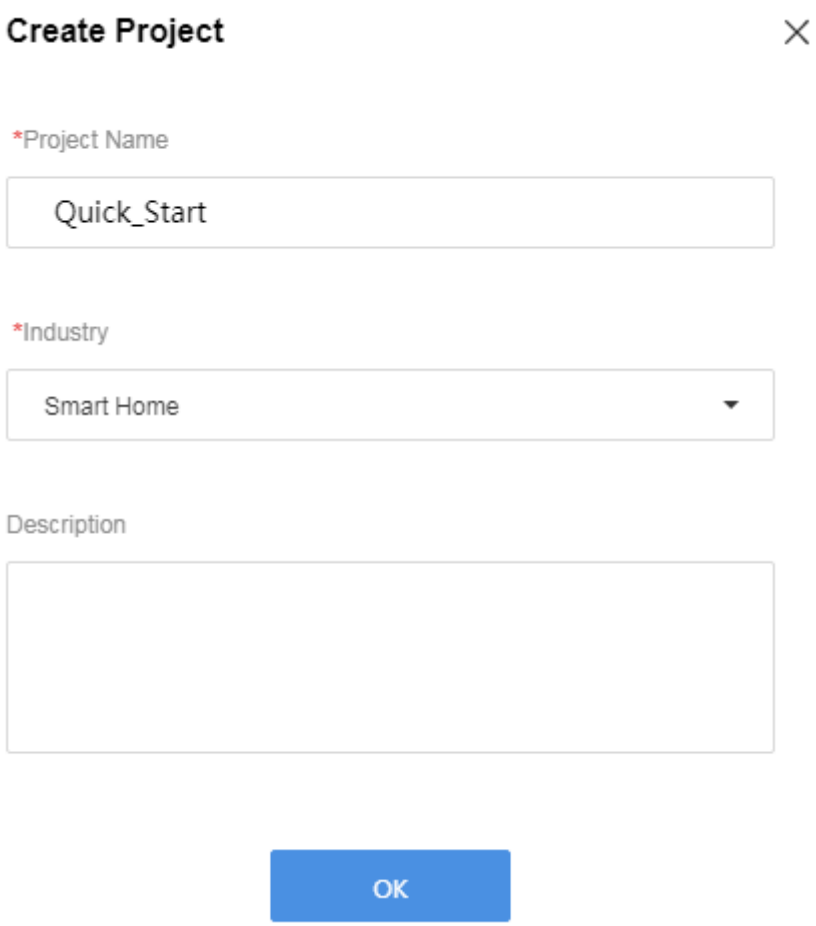
Step 1 On the home page of the Developer Center, click **Create Project**.



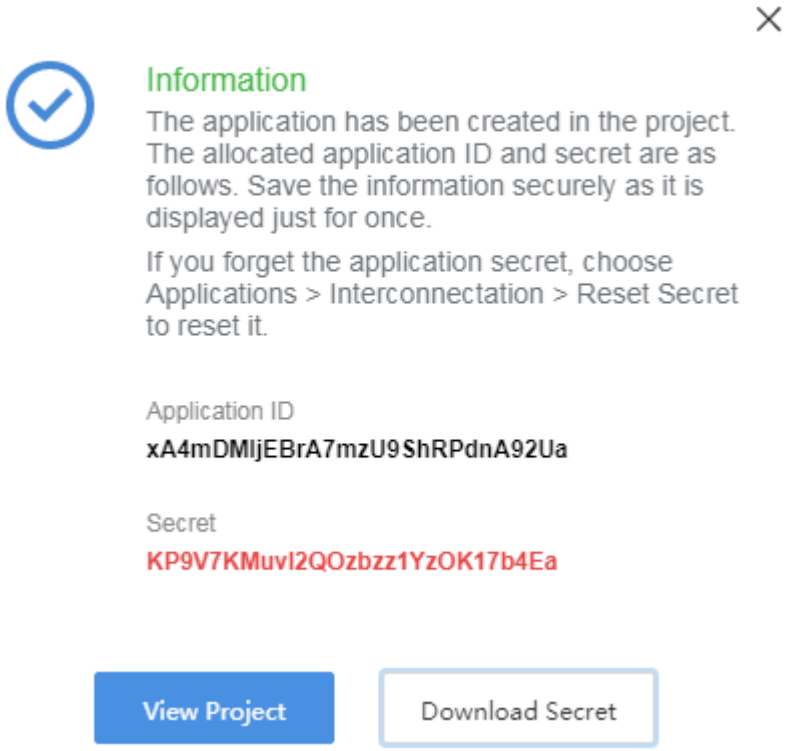
Step 2 Specify **Project Name**, **Industry**, and **Description**, and click **OK**.

NOTE

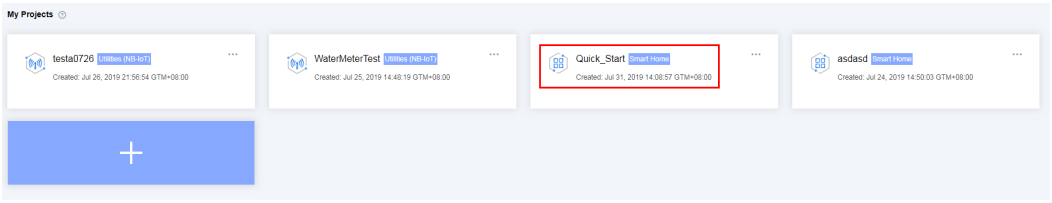
Project Name must be unique and cannot be duplicate with other projects. Otherwise, the creation fails.



After a project is created, the system returns **Application ID** and **Secret**. These two parameters are required for the interconnection between the application and the IoT platform. Therefore, developers are advised to keep them securely. In case that developers forget them, reset them in **Applications** > **Interconnection** > **Application Security**.



Step 3 Click the newly added project to enter the project space.

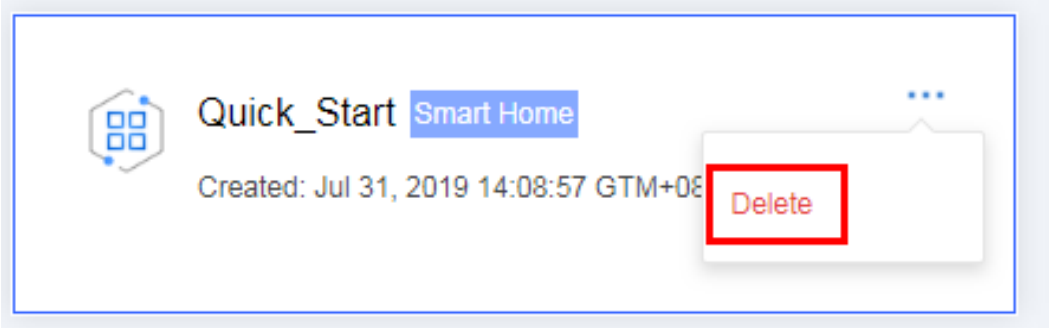


----End

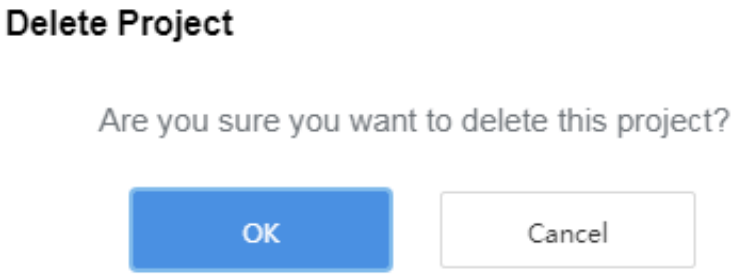
Deleting a Project

After a project is deleted, all resources under the project are cleared. Exercise caution when deleting a project.

Step 1 Under **My Projects** on the home page of the Developer Center, select the project to be deleted, click ⋯ beside the project name, and click **Delete**.



Step 2 In the **Delete Project** dialog box, click **OK** to delete the project.



----End

2.5 Product

2.5.1 Introduction

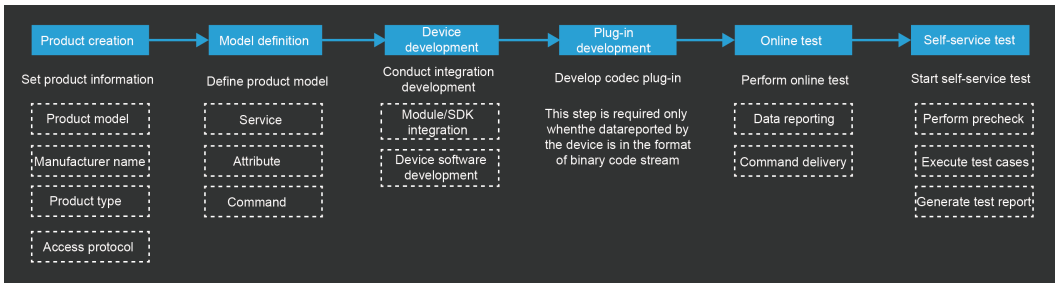
The Developer Center consists of product development, device management, and upgrade debugging modules.

- Product development provides end-to-end guides for developers to develop and release IoT products by performing operations such as model definition, device development, codec development, and online testing in sequence. For details, see **Product Development**.
- Device management displays all physical and virtual devices of a project, and provides functions such as categorized statistics, online tests, and device logs for developers to manage devices and locate faults. For details, see **Device Management**.
- Upgrade debugging enables you to upgrade the firmware and software of devices. For details, see **Upgrade Debugging**.

2.5.2 Product Development

2.5.2.1 Overview

The IoT platform supports various device access modes, such as NB-IoT, 2G/3G/4G, and wired network. To connect different types of devices to the IoT platform and build interconnection solutions, a series of processes are required, such as profile definition, device development, codec development, and certification tests.



2.5.2.2 Product Creation

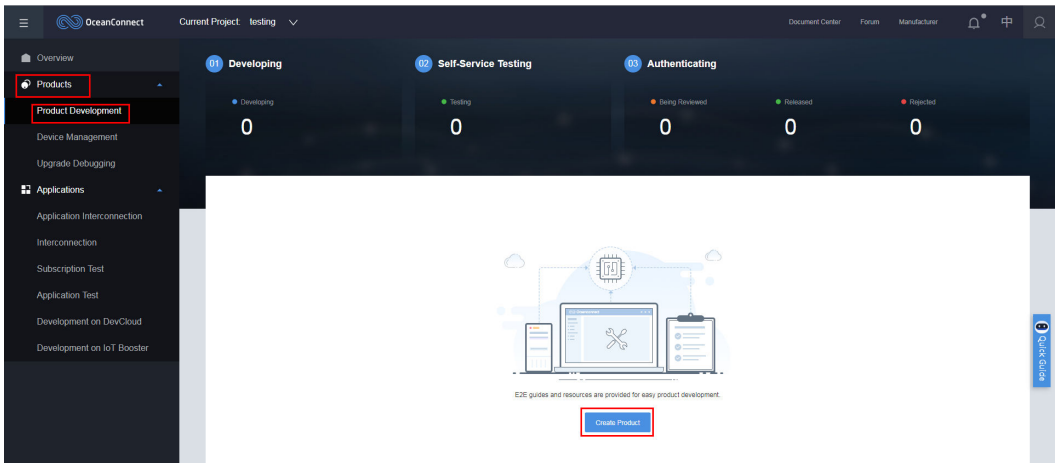
Overview

A collection of devices with the same capabilities or features is called a product. In addition to physical devices, a product includes product information, product models (profiles), codecs, and test reports generated during IoT capability building.

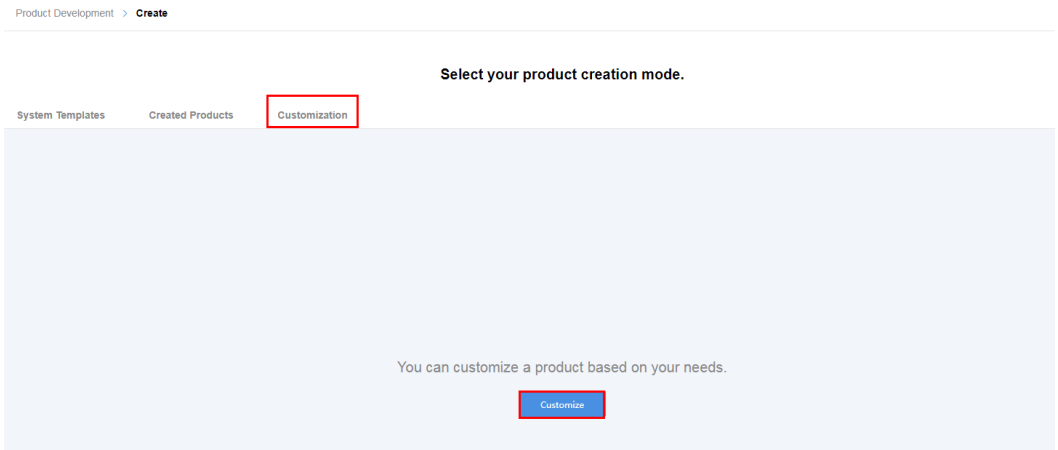
Customizing a Product

Customizing a product refers to defining a new product without using a preset product template.

Step 1 In the project space, choose **Products** > **Product Development**, and click **Create Product**.



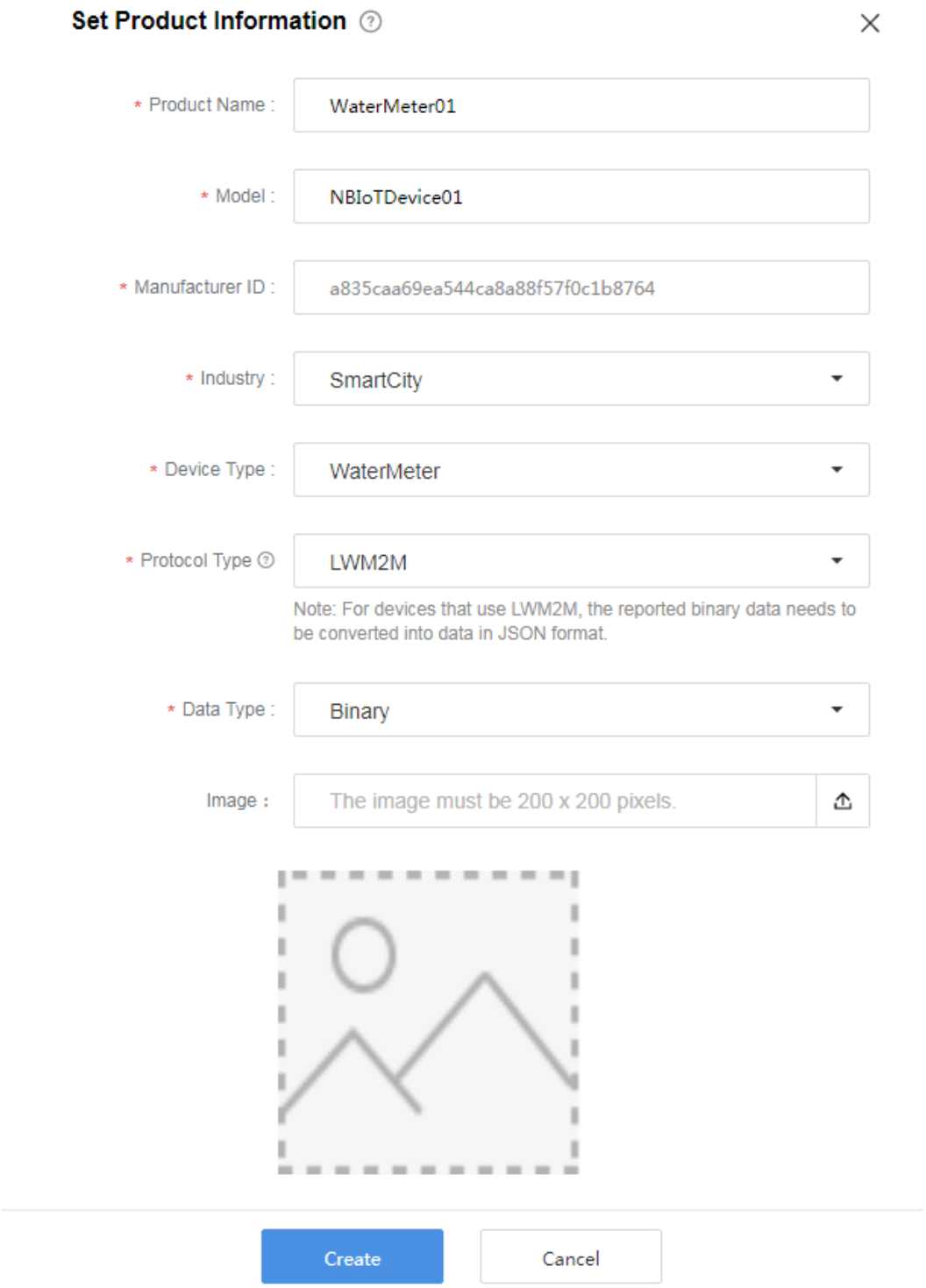
Step 2 Select the **Customization** tab and click **Customization**.



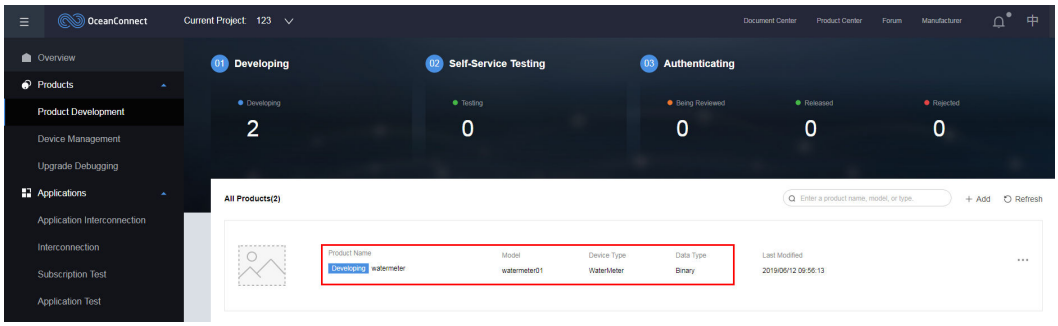
Step 3 In the **Set Product Information** dialog box displayed, specify the parameters such as **Product Name** and **Model**, and click **Create**.

NOTE

- **Product Name** and **Model** must be unique in the project. Otherwise, the creation fails.
- Specify **Industry**, **Device Type**, **Protocol Type**, and **Data Type** based on site requirements.
- If **Data Type** is **Binary**, codec development is required for the product. If **Data Type** is **JSON**, codec development is not required.



Step 4 On the **Product Development** page, select a product to enter its development space.

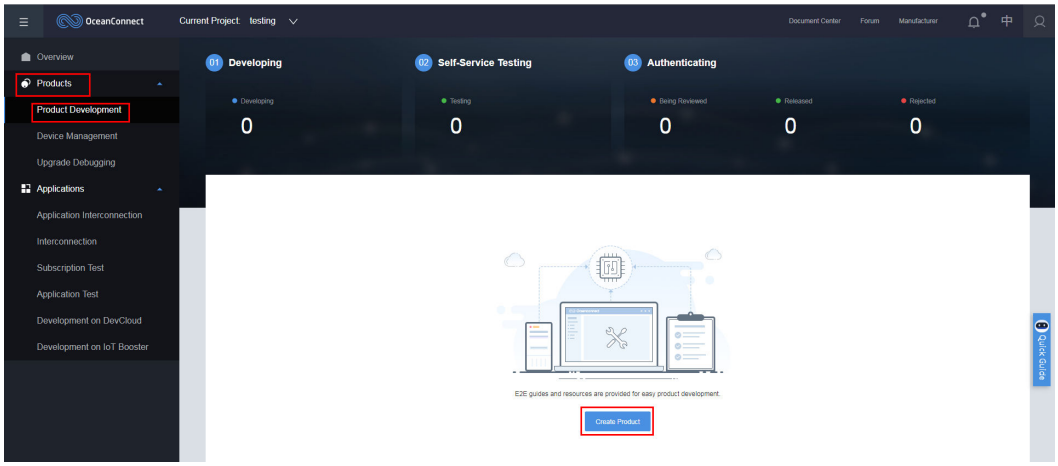


----End

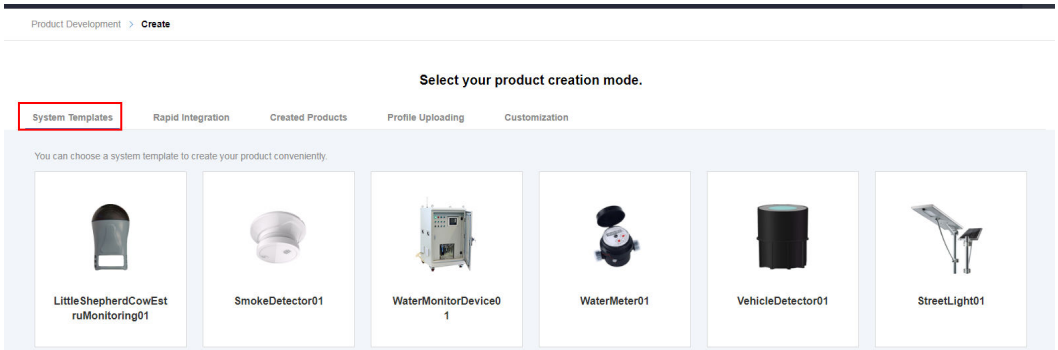
Quickly Creating a Product

Quickly creating a product refers to defining a product by using a preset product template (or a created product template).

Step 1 In the project space, choose **Products** > **Product Development**, and click **Create Product**.



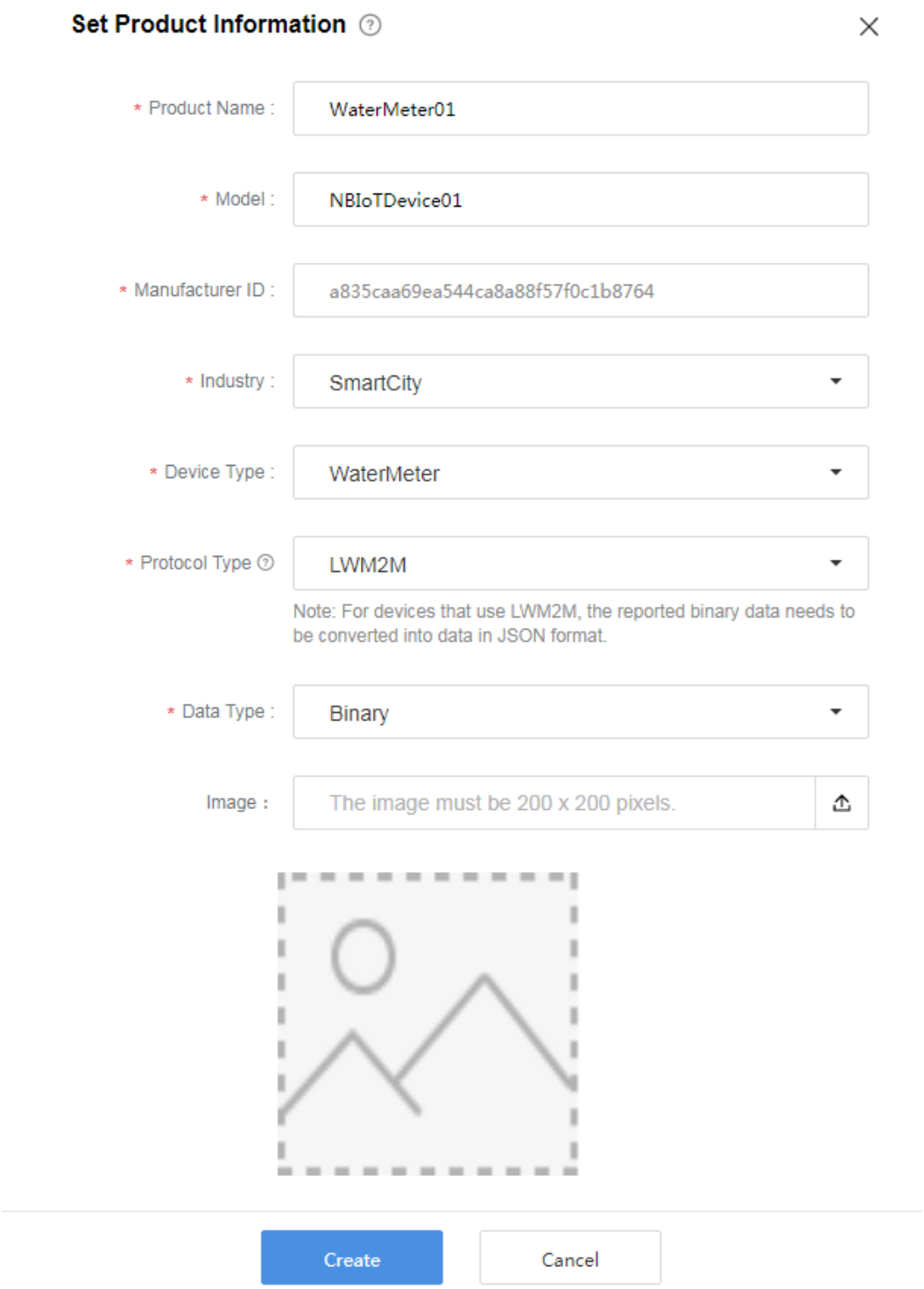
Step 2 On the **System Templates** tab page, select a required system template and click **Select**.



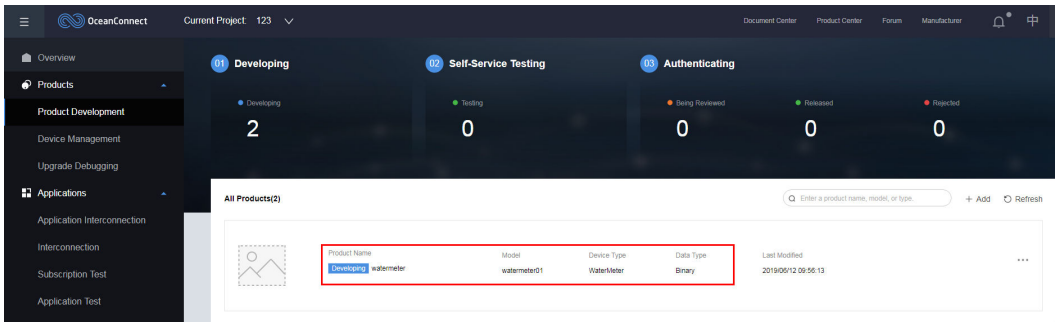
Step 3 In the **Set Product Information** dialog box displayed, specify the parameters such as **Product Name** and **Model**, and click **Create**.

NOTE

- **Product Name** and **Model** must be unique in the project. Otherwise, the creation fails.
- Specify **Industry**, **Device Type**, **Protocol Type**, and **Data Type** based on site requirements.
- If **Data Type** is **Binary**, codec development is required for the product. If **Data Type** is **JSON**, codec development is not required.



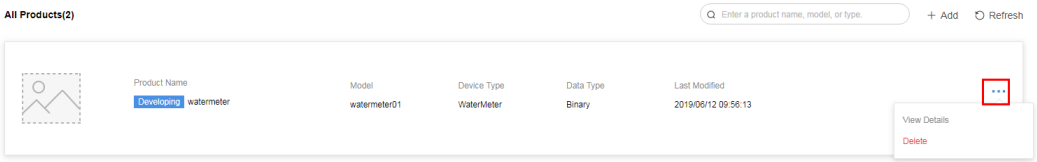
Step 4 On the **Product Development** page, select a product to enter its development space.



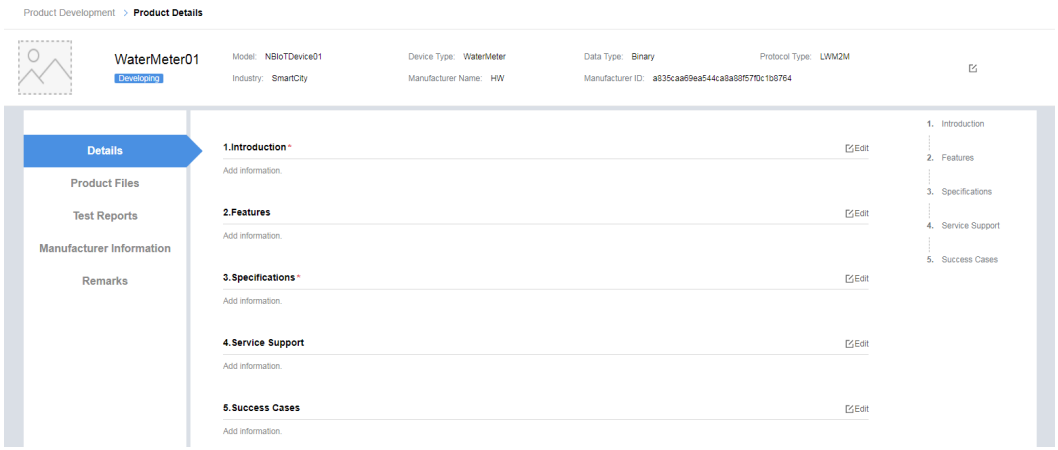
----End

Supplementing Product Information

Step 1 On the **Product Development** page, select the desired product, click ⋯ on the right, and click **View Details**. The **Product Details** page is displayed.



Step 2 On the **Product Details** page, you can view **Details**, **Product Files**, **Test Reports**, and **Manufacturer Information**. On the **Details** page, you can supplement the product information such as **Introduction**, **Features**, and **Specifications**.

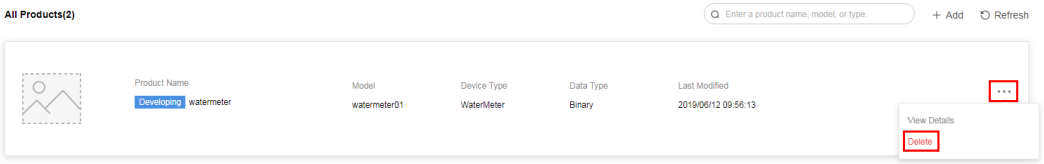


----End

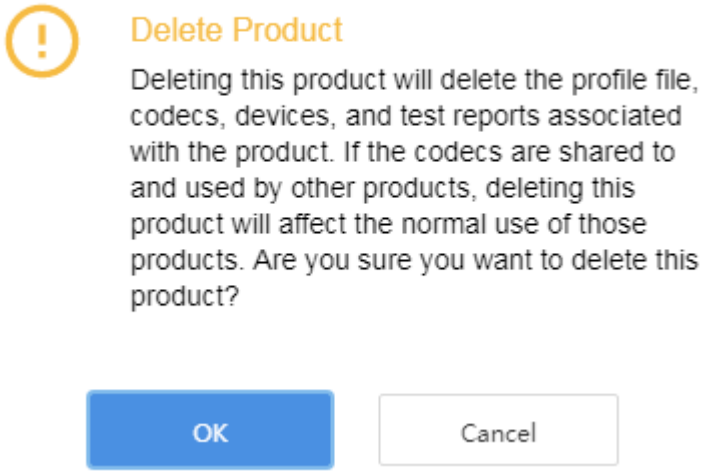
Deleting a Product

After a product is deleted, resources such as the profile files and codecs of the product will be cleared. Exercise caution when deleting a product.

Step 1 On the **Product Development** page, select the desired product, click ⋯ on the right, and click **Delete**.



Step 2 In the **Delete Product** dialog box, click **OK** to delete the product.

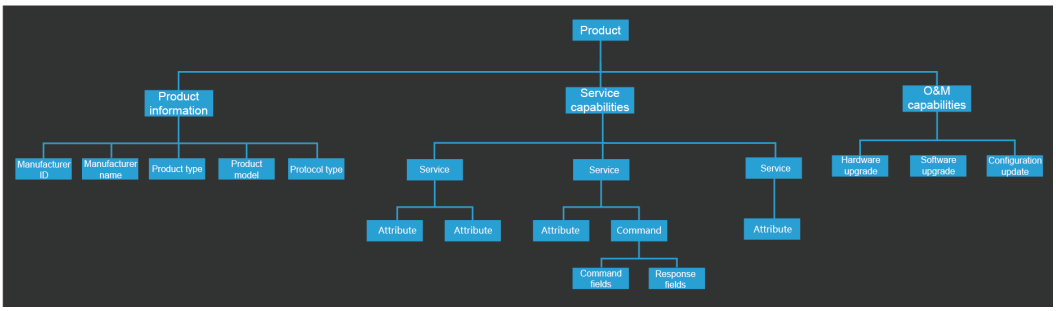


----End

2.5.2.3 Profile Definition

Overview

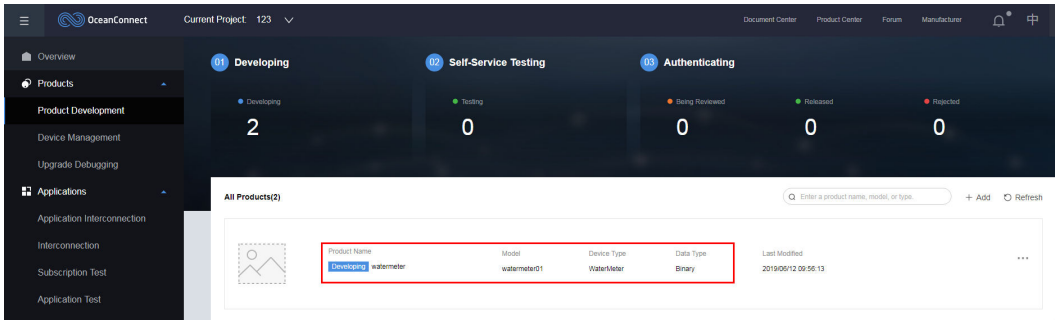
A product model (also called profile) is used to describe the capabilities and features of a device. Developers construct an abstract model of a device by defining a profile file on the IoT platform so that the IoT platform can understand the services, properties, and commands supported by the device.



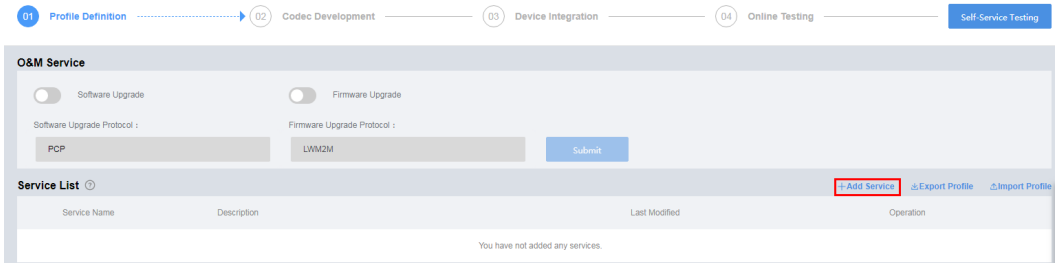
Defining a Profile

If you choose to use a default template during **Product Creation**, the system automatically chooses the corresponding profile template. You can directly use or modify the profile template. If a customized product template is used, perform the following operations to fully define the profile:

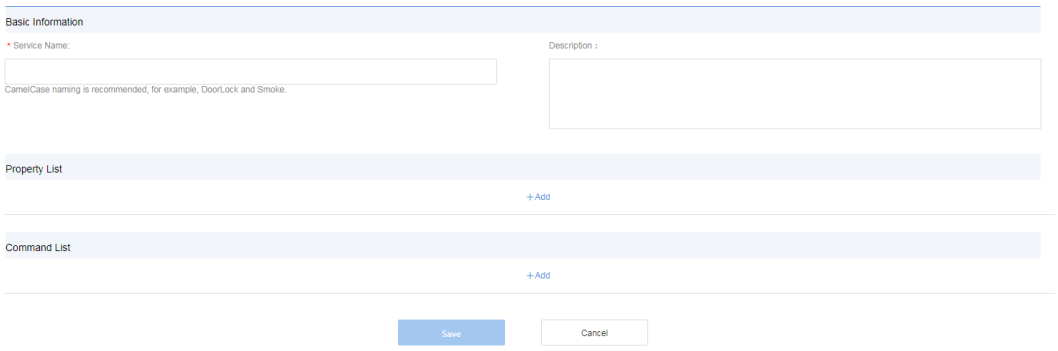
Step 1 On the **Product Development** page, select a product to enter its development space.



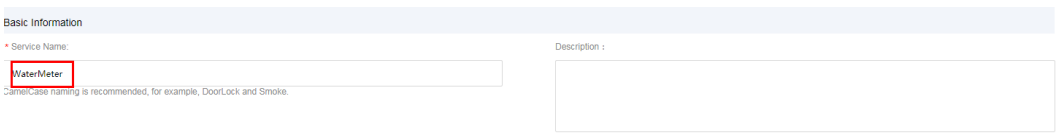
Step 2 In the development space, choose **Profile Definition** and click **Add Service**.



Step 3 In the **Add Service** area, define the service name, properties, and commands. Each service can contain both properties and commands or only one of them. Configure the properties and commands based on the actual situation.



1. Enter **Service Name** using Camel-Case naming method, such as WaterMeter and Battery.

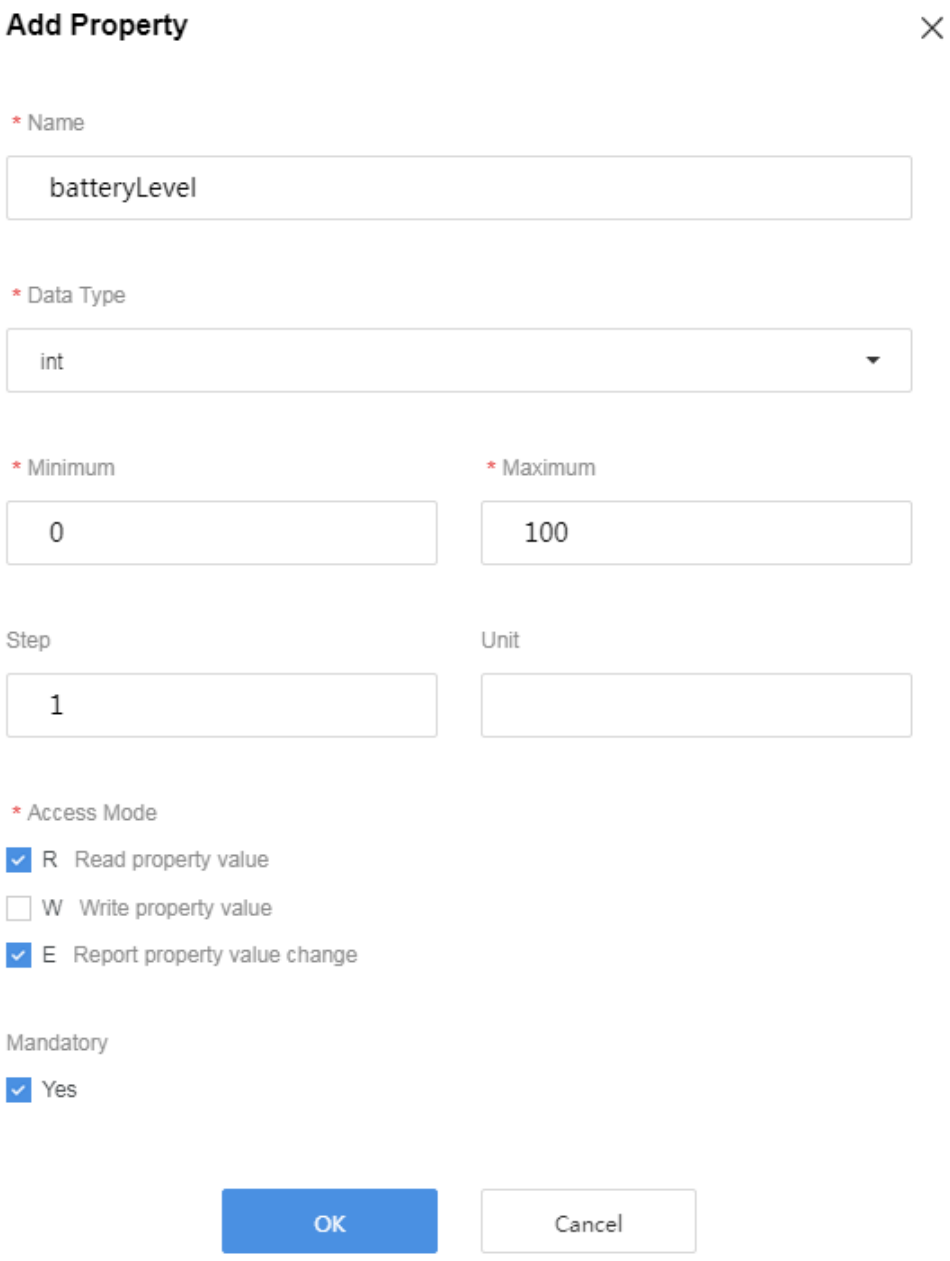


2. Click **Add** under **Property List**, set the parameters in the displayed dialog box, and click **OK**. For **Name**, the first letter of the first word must be lowercase, and the first letters of subsequent words are capitalized, for example, batteryLevel or internalTemperature. For other parameters, set them based on the actual situation.

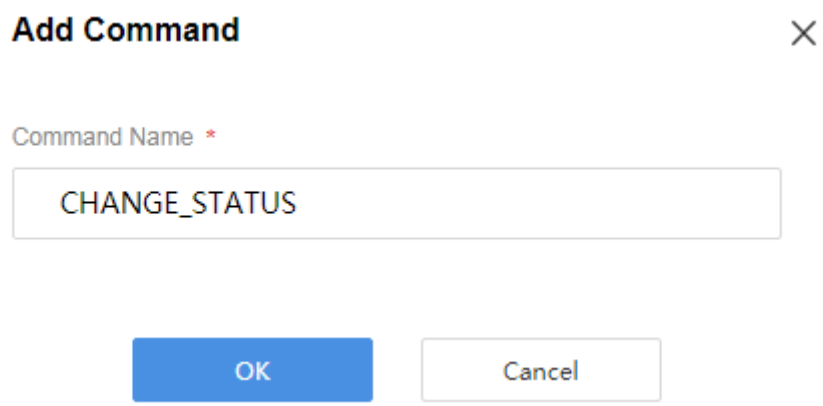
   The rules for configuring **Data Type** are as follows:

   - **int**: If the reported data is an integer or Boolean values, set the data type to this value.

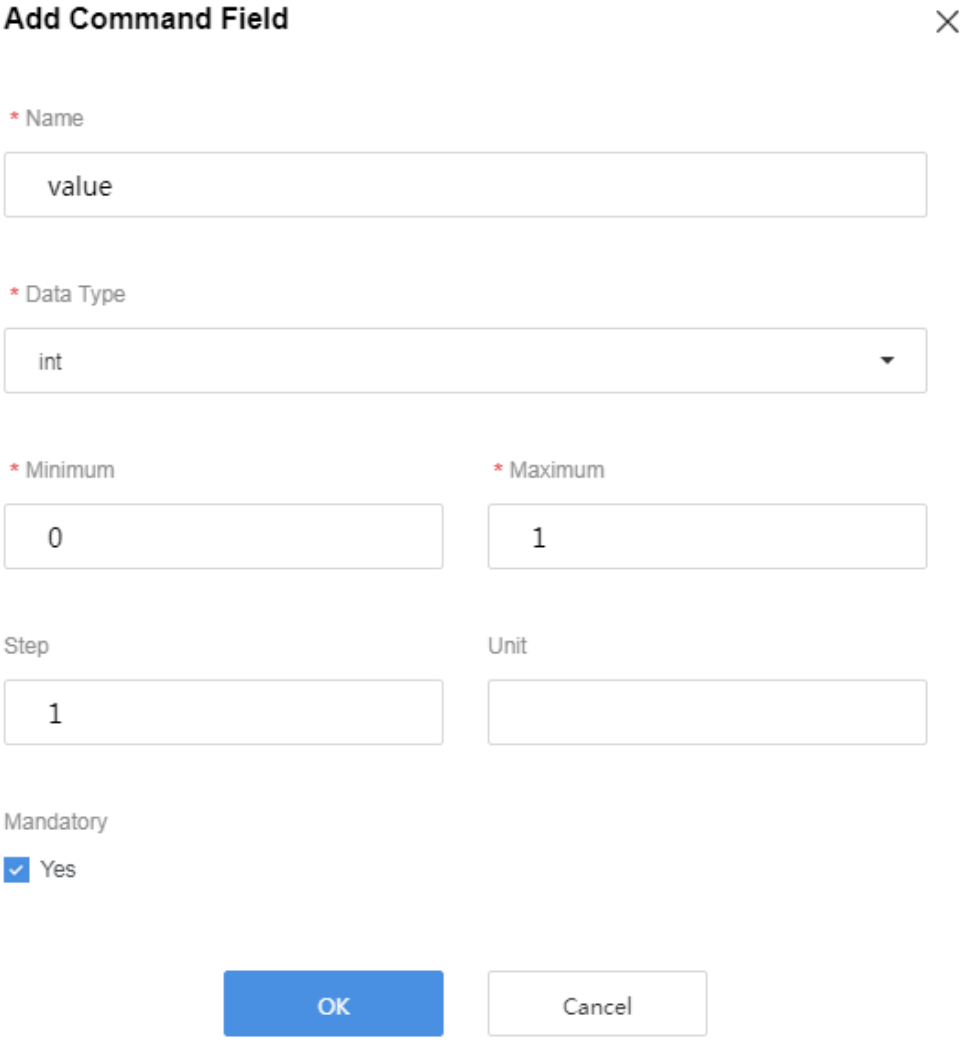
- **decimal**: If the reported data is a decimal, set the data type to this value.
- **string**: If the reported data is a string, enumerated values, or Boolean values, set the data type to this value. If enumerated or Boolean values are reported, use commas (,) to separate the values.
- **DateTime**: If the reported data is a date, set the data type to this value.
- **jsonObject**: If the reported data is in JSON structure, set the data type to this value.



3. Click **Add** under **Command List**. In the displayed dialog box, set **Command Name** and click **OK**. The value of **Command Name** must consist of uppercase letters, and the words are separated by underscores (_), for example, DISCOVERY or CHANGE_STATUS.



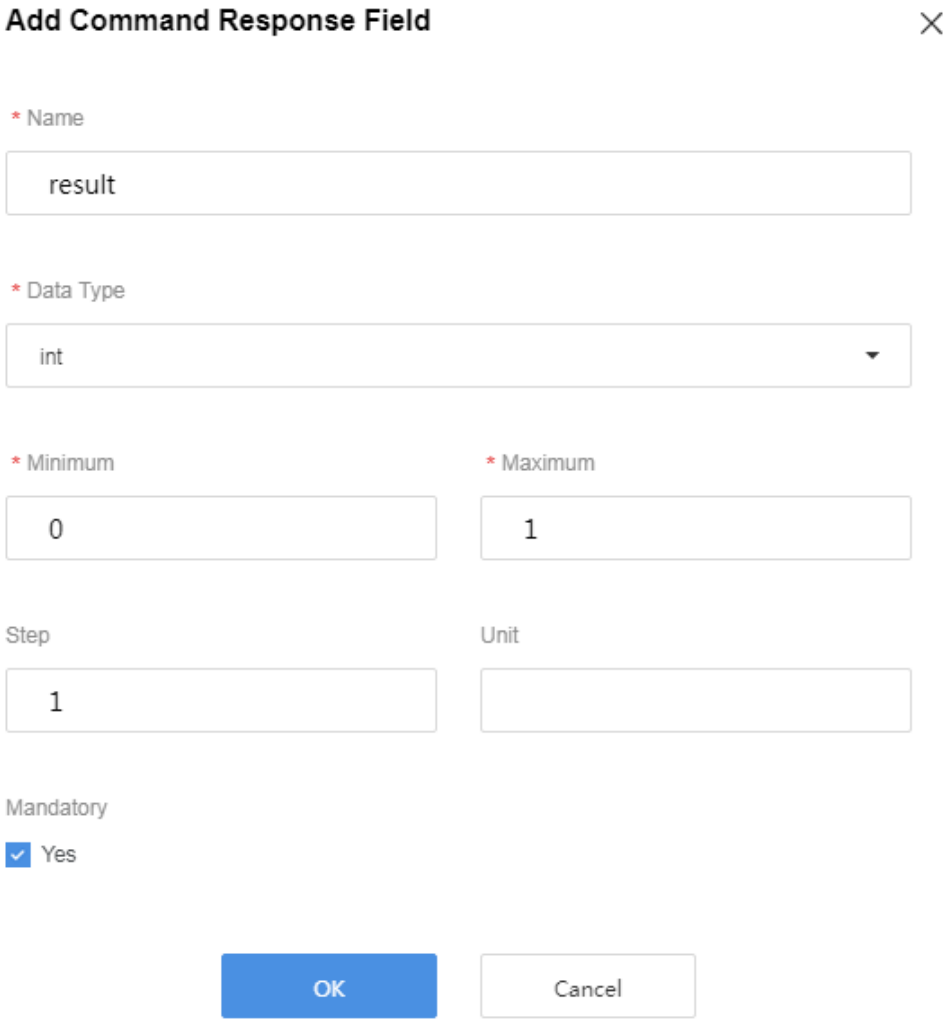
4. Click **Add** under **Command Fields**. In the dialog box displayed, set the parameters and click **OK**. For **Name** of the command field, the first letter of the first word is lowercase, and the first letters of the subsequent words are capitalized, for example, value. For other parameters, set them based on the actual situation.



5. Click **Add** under **Command Response Fields**. In the dialog box displayed, set the parameters and click **OK**. For **Name** of the command response field, the first letter of the

first word is lowercase, and the first letters of the subsequent words are capitalized, for example, result. For other parameters, set them based on the actual situation.

The command response field is optional. This field needs to be defined only when the device needs to return the command execution result.



----End

2.5.2.4 Codec Development

Overview

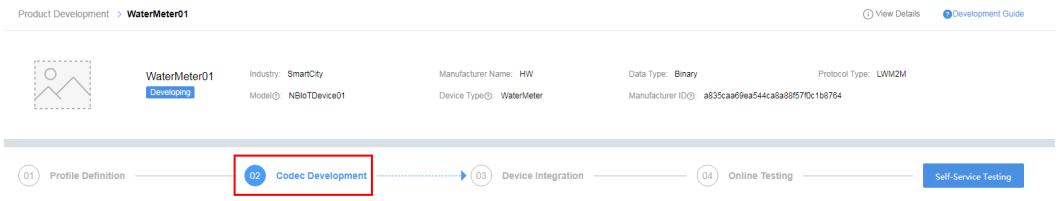
When a device reports data, if **Data Type** is **Binary**, a codec needs to be developed for the product. If **Data Type** is **JSON**, codec development is not required.

For example, in the NB-IoT scenario where devices communicate with the IoT platform using CoAP, the payload of the CoAP message is the data at the application layer and the data type is defined by the device. As NB-IoT devices have demanding requirements on power saving, data at the application layer is in binary format instead of JSON format. However, the IoT platform communicates with NAs by sending data in JSON format. Therefore, codec development is needed for the IoT platform to convert data in binary and JSON formats.

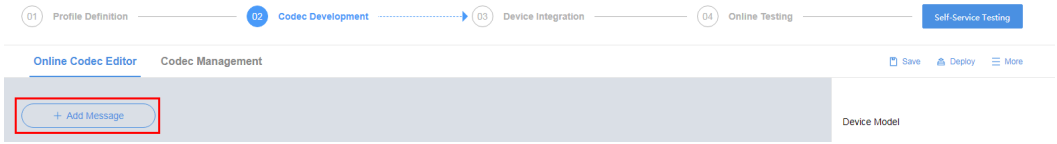
Developing a Codec

When you are **Customizing a Product**, if the system template is used, you can directly use or modify the codecs contained in some of the templates. If you choose to customize a product, you need to develop a complete codec. The development process is as follows:

Step 1 In the product development space, click **Codec Development**.

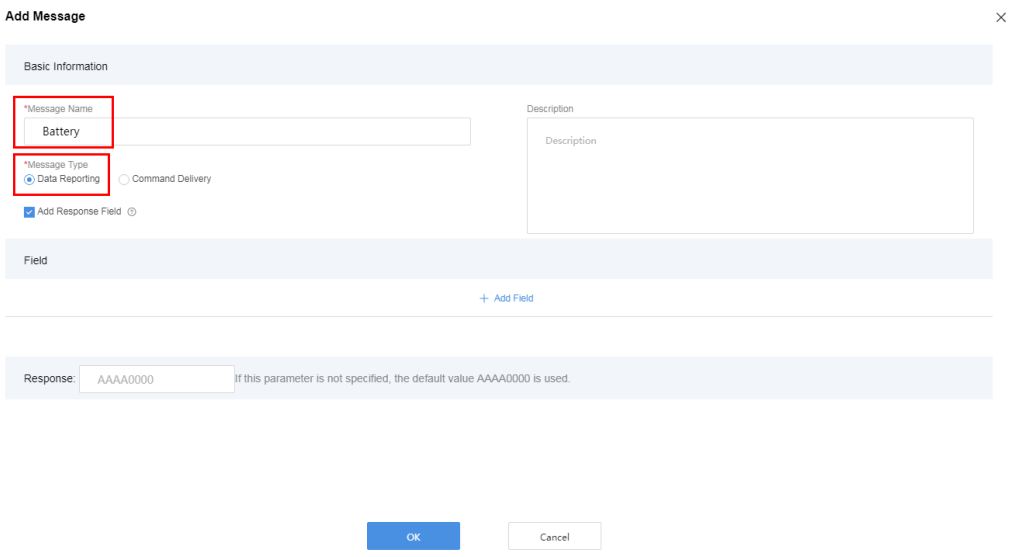


Step 2 In the **Online Codec Editor** area, click **Add Message**.

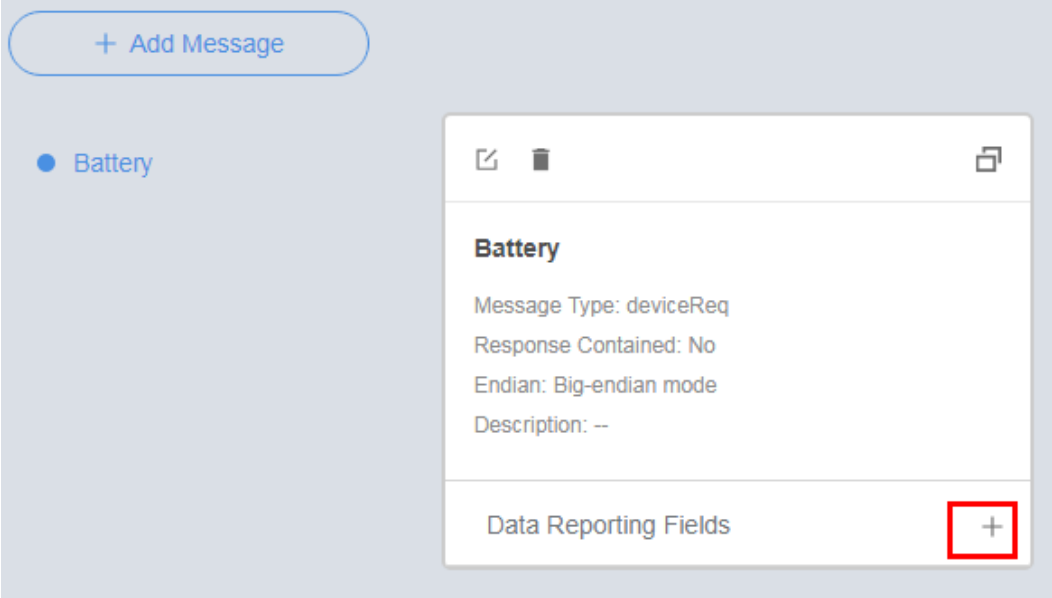


Step 3 In the **Add Message** dialog box displayed, specify **Message Name**, set **Message Type** to **Data Reporting**, and click **OK**.

- If the IoT platform is required to return an ACK message after the device reports data, **Add Response Field** must be selected. The data carried in the ACK message can be configured in **Response**. The default value is **AAAA0000**.
- **Message Name** can contain only letters, digits, underscores (_), and dollar signs ($) and cannot start with a digit.

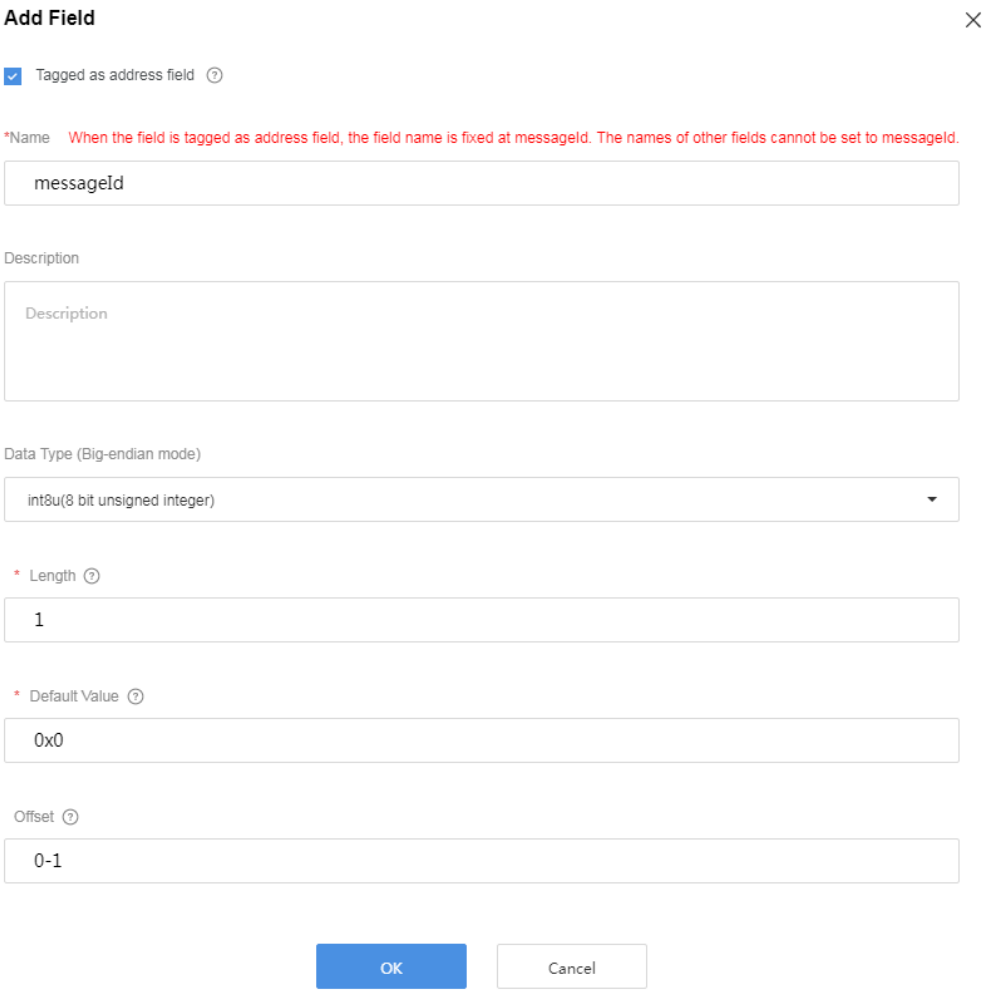


Step 4 Click + next to **Data Reporting Fields**.

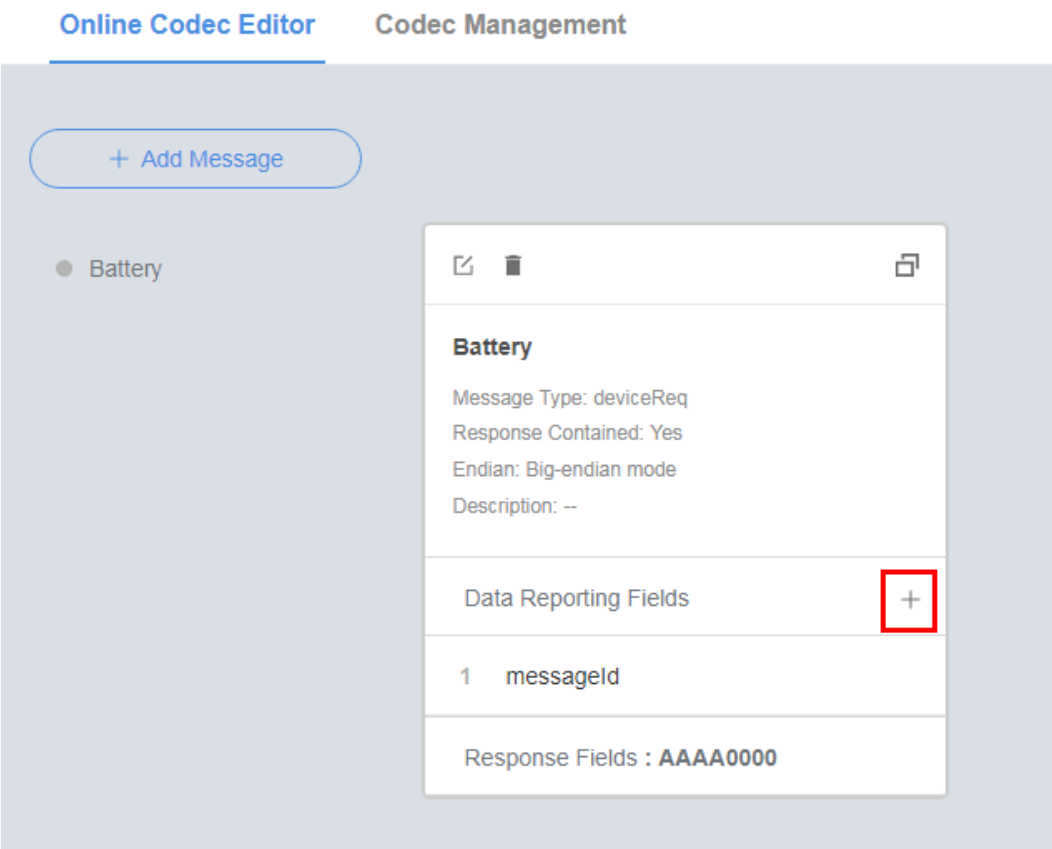


Step 5 In the **Add Field** dialog box displayed, select **Tagged as address field** and other parameters will be set automatically. Then, click **OK**.

When messages of the same type are created, such as two data reporting messages, this option must be selected and this field in every such message must be in the same place on the field list. Command response can be regarded as a type of data reporting message. Therefore, if a command response exists, an address field needs to be added to the data reporting message.

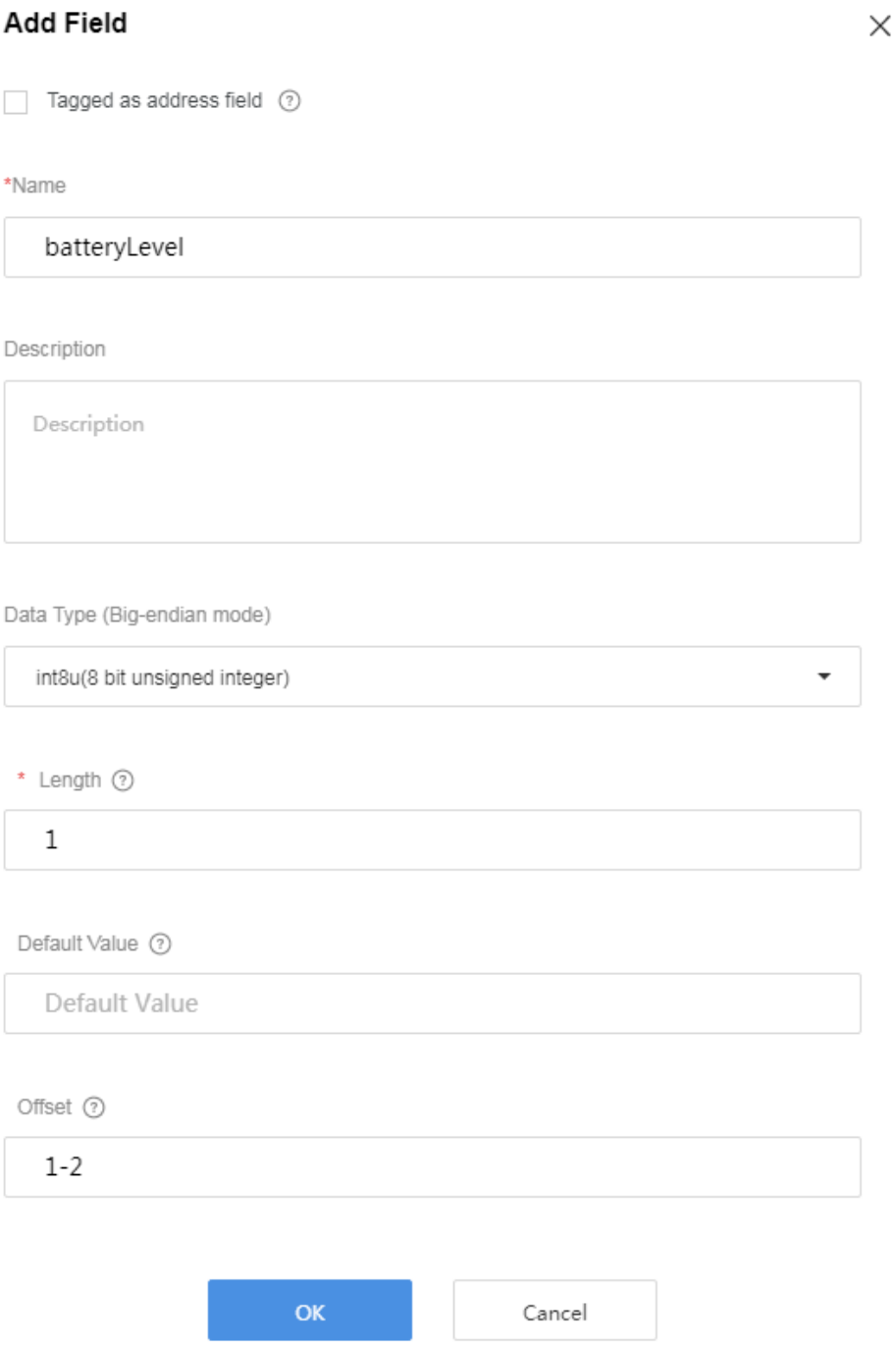


Step 6 Click + next to **Data Reporting Fields**.

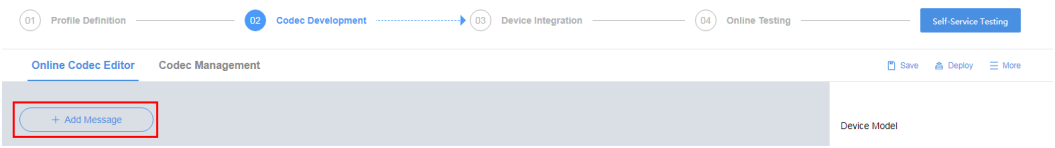


Step 7 In the **Add Field** dialog box displayed, set the parameters and click **OK**.

- **Name** can contain only letters, digits, underscores (_), and dollar signs ($) and cannot start with a digit.
- **Data Type** is configured based on the data reported by the device and must match the type of the corresponding field defined by the profile.

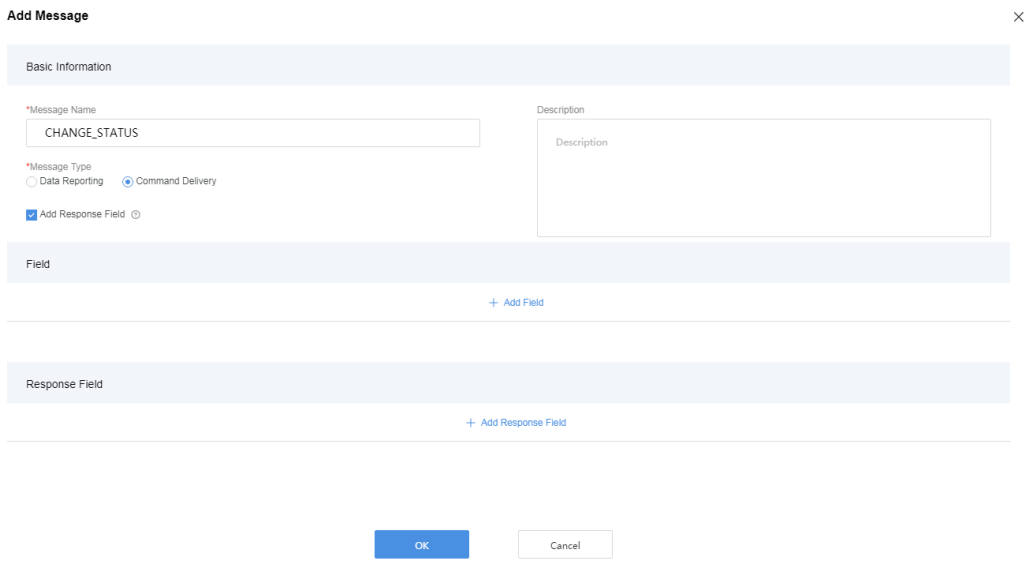


Step 8 In the **Online Codec Editor** area, click **Add Message**.

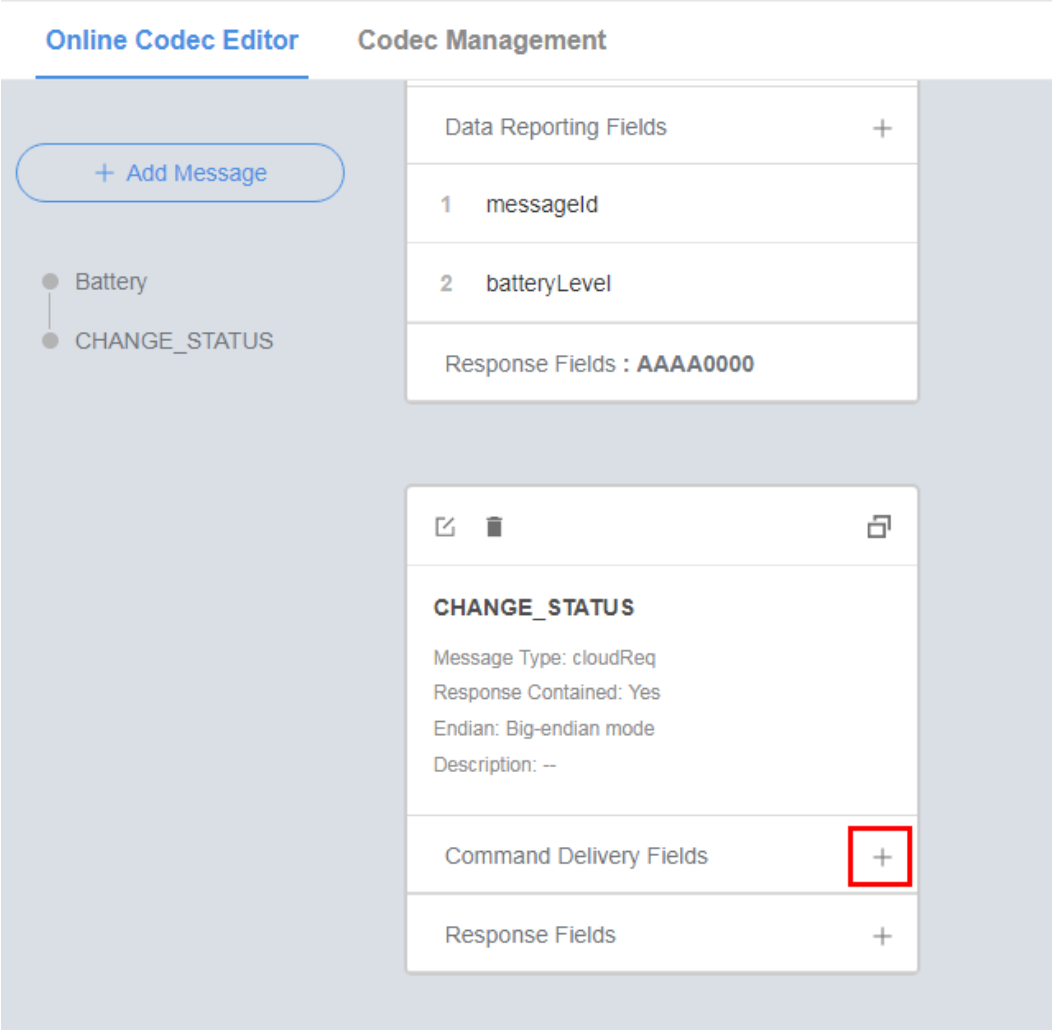


Step 9 In the **Add Message** dialog box displayed, specify **Message Name**, set **Message Type** to **Command Delivery**, and click **OK**.

- If the device is required to return the command execution result, select **Add Response Field**. After the check box is selected:
  - The address field must be defined in both the data reporting message and the command response, and this field in the two messages must be in the same place on the field list, so that the codec can distinguish the data reporting message from the command response.
  - The response ID field must be defined in the command delivery message and the command response, and this field in the two messages must be in the same place on the field list, so that the codec can associate the command delivery message with the corresponding command response.
- **Message Name** can contain only letters, digits, underscores (_), and dollar signs ($) and cannot start with a digit.

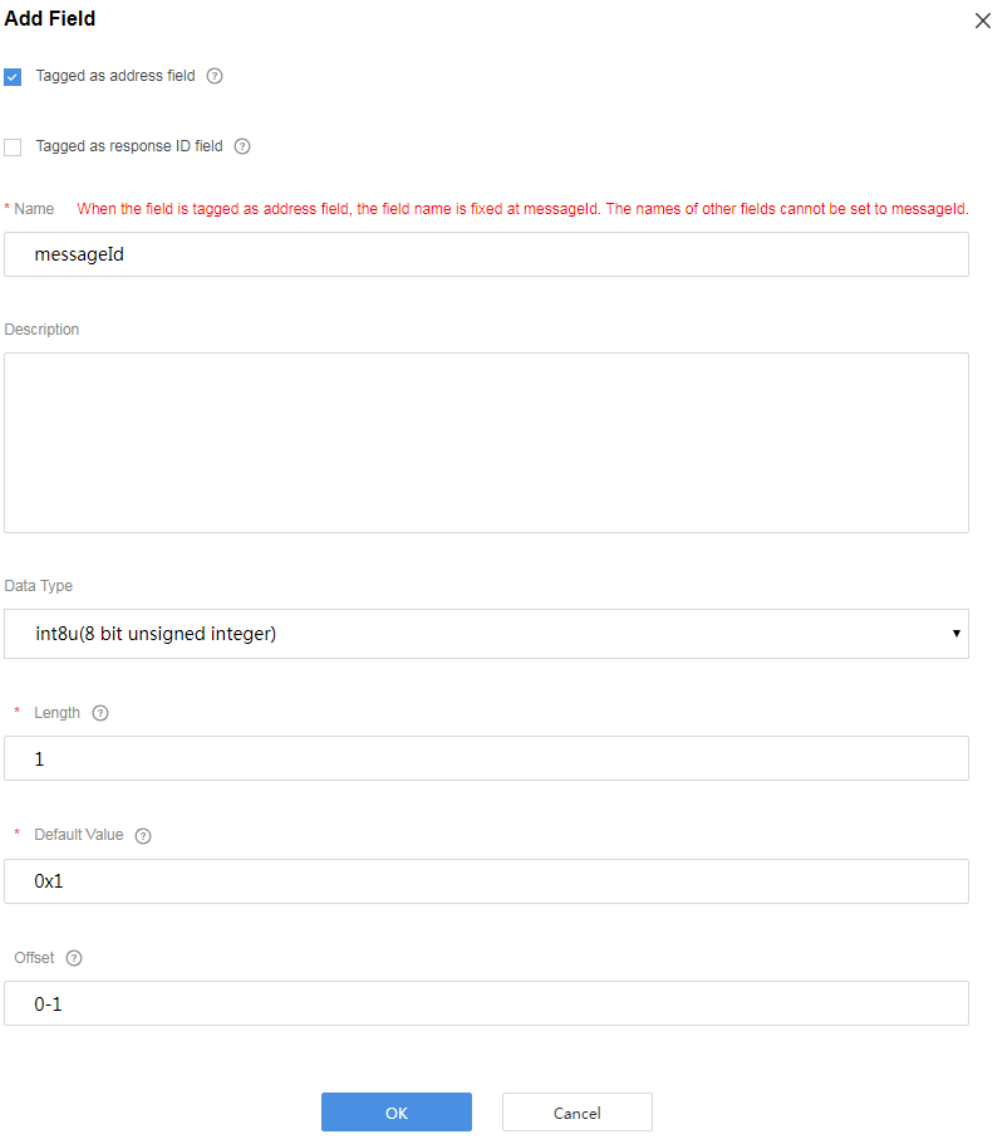


Step 10 Click + next to **Command Delivery Fields**.

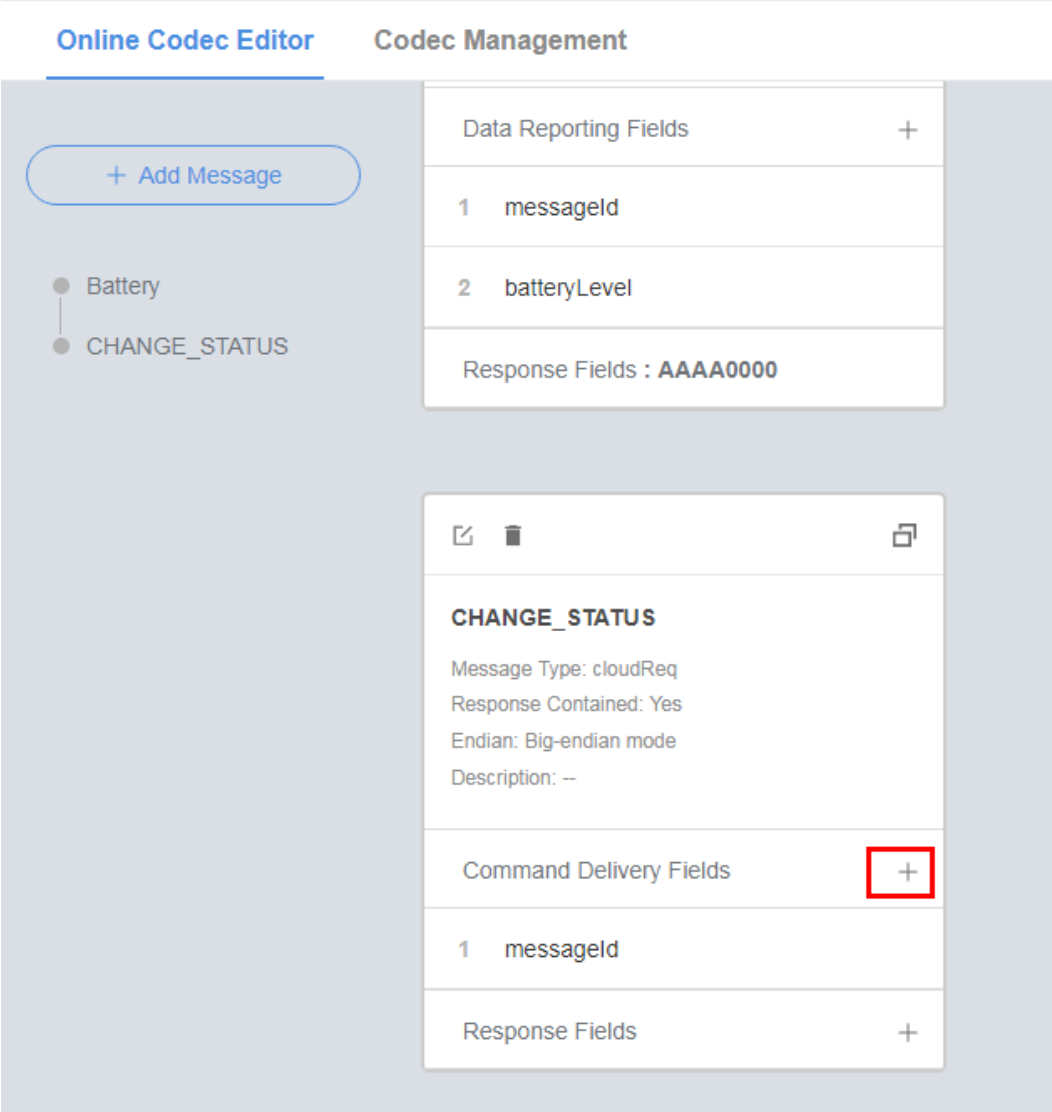


Step 11 In the **Add Field** dialog box displayed, select **Tagged as address field** and other parameters will be set automatically. Then, click **OK**.

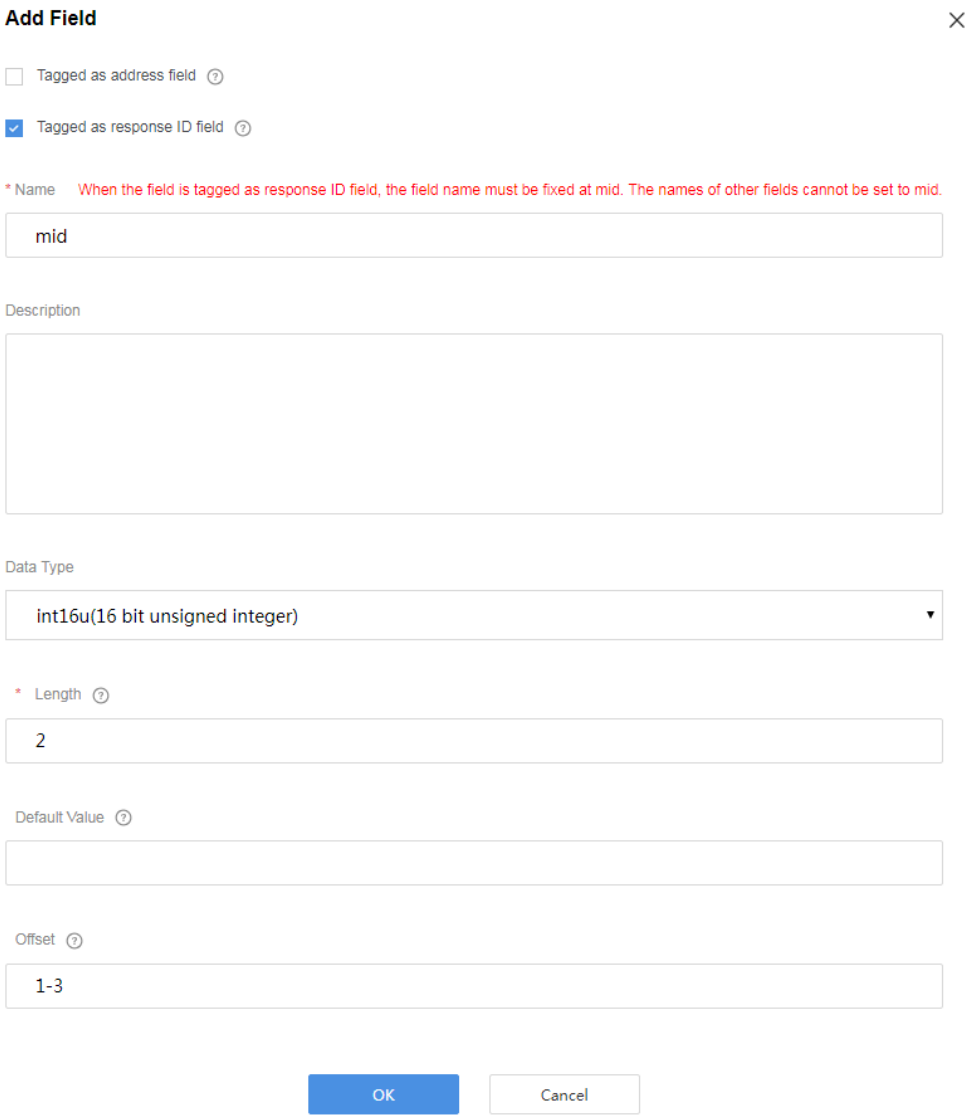
When messages of the same type are created, such as two command delivery messages, this option must be selected and this field in every such message must be in the same place on the field list. Data reporting response can be regarded as a type of command delivery message. Therefore, if a data reporting response exists, an address field needs to be added to the command delivery message.



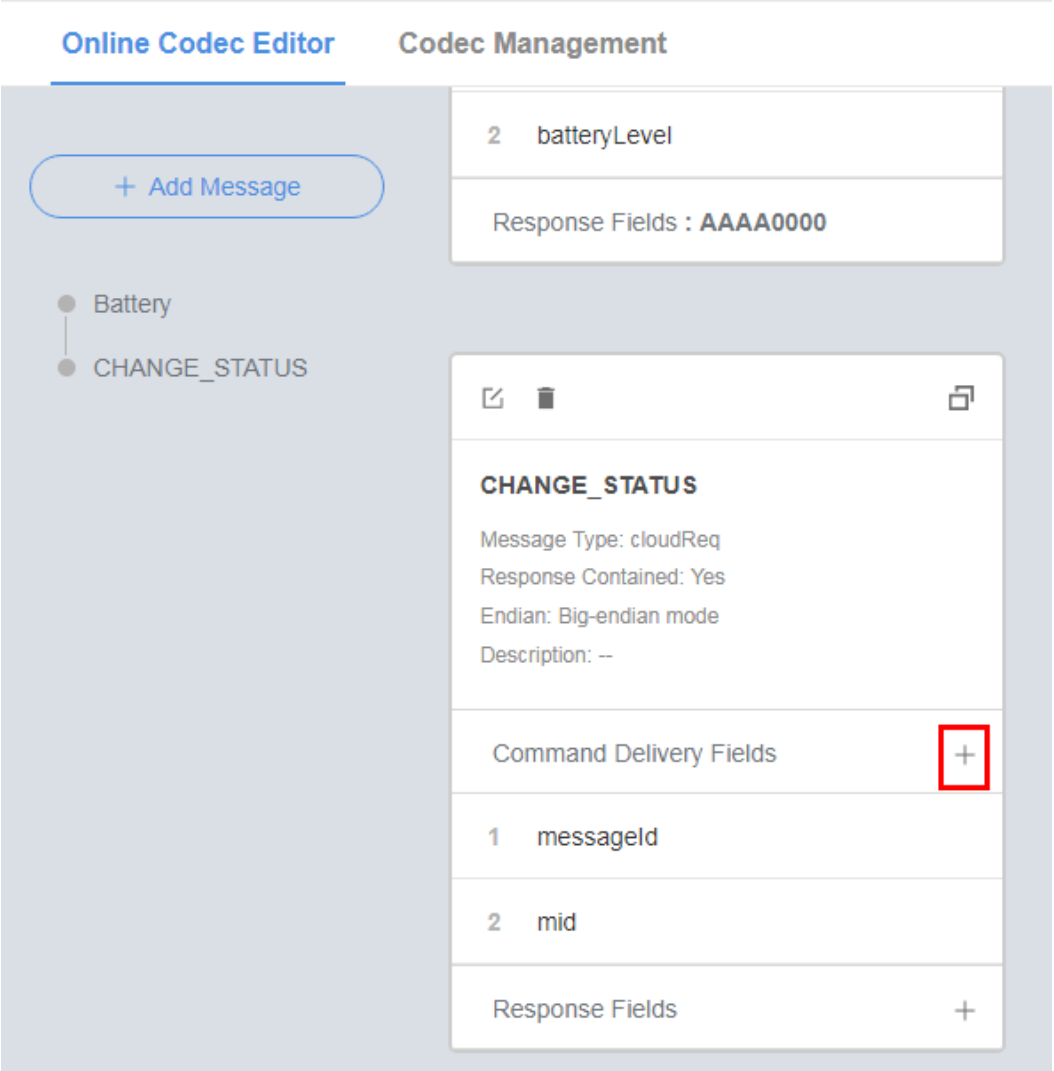
Step 12 Click + next to **Command Delivery Fields**.



Step 13 In the **Add Field** dialog box displayed, select **Tagged as response ID field** and other parameters will be set automatically. Then, click **OK**.

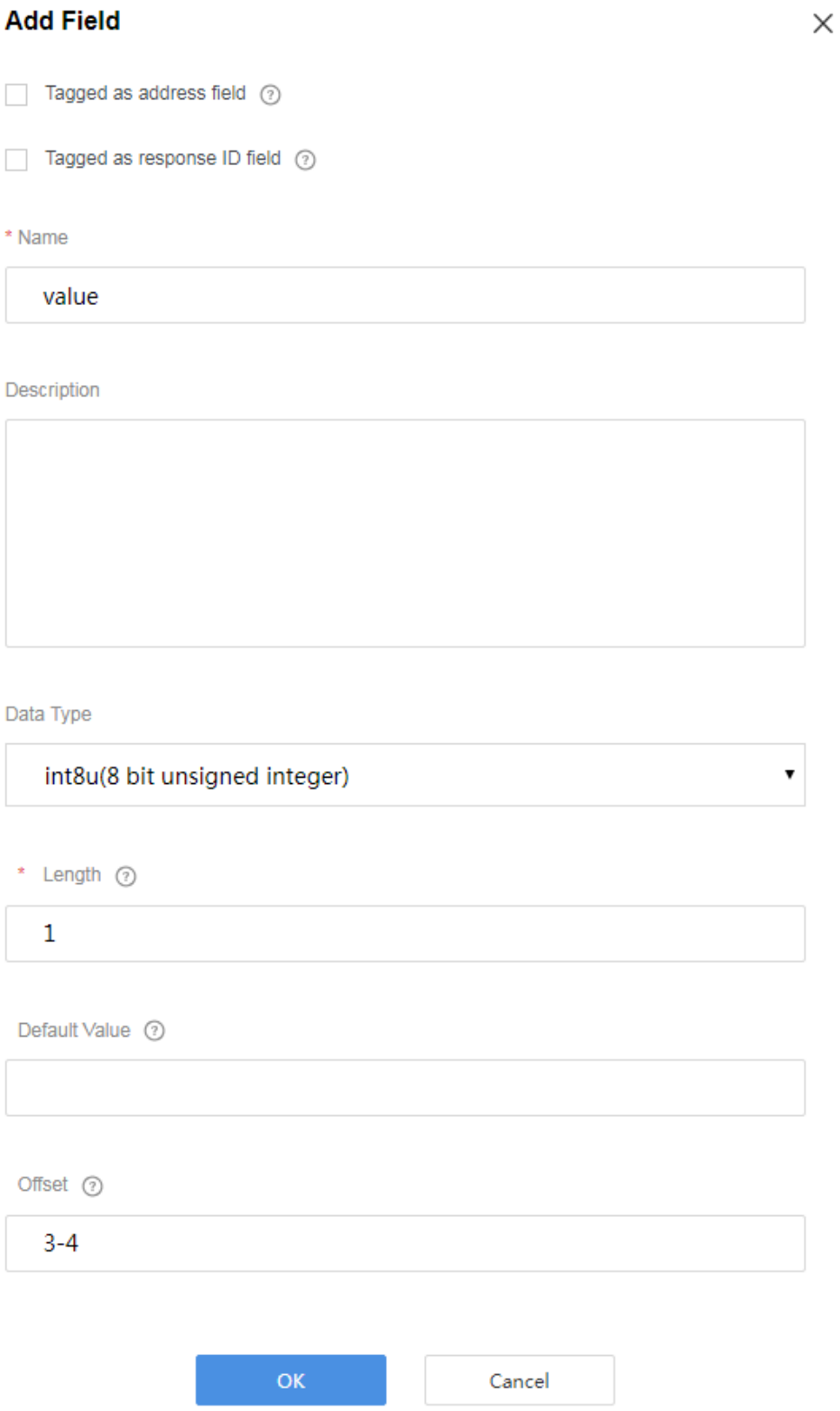


Step 14 Click + next to **Command Delivery Fields**.

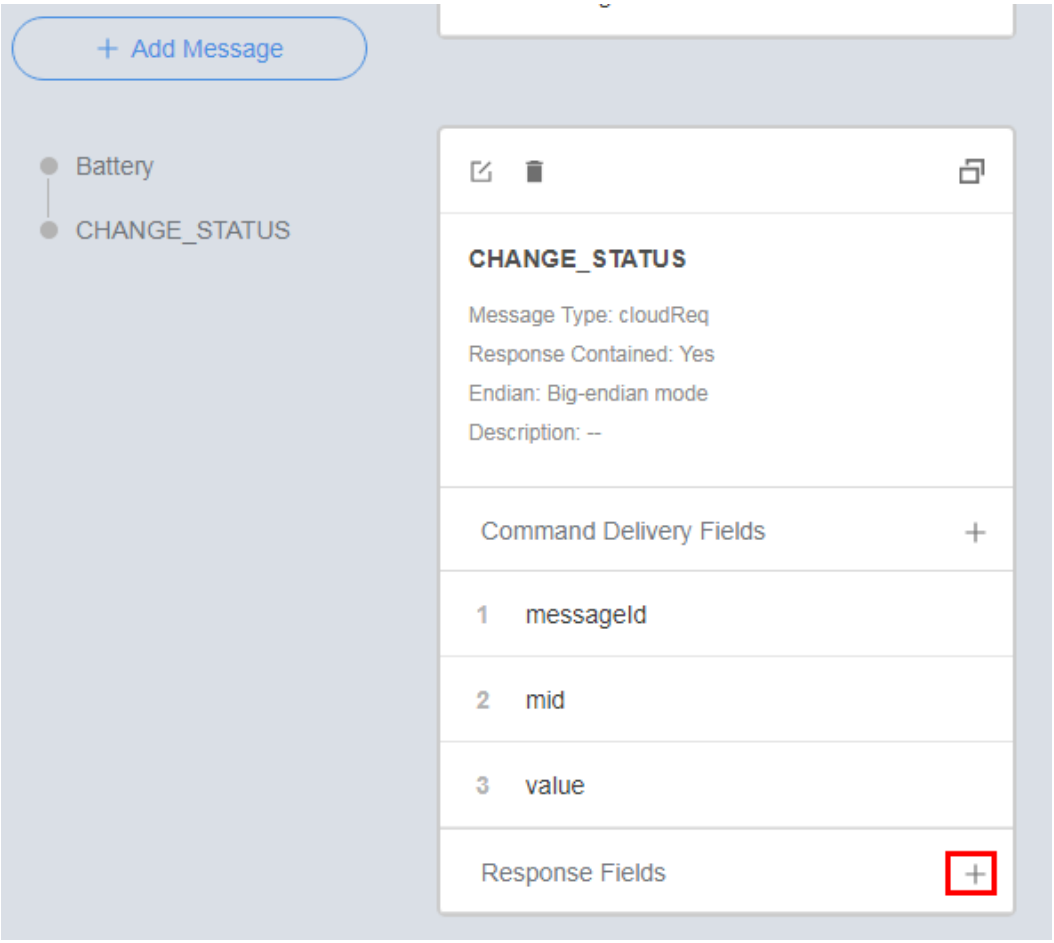


Step 15 In the **Add Field** dialog box displayed, set the parameters and click **OK**.

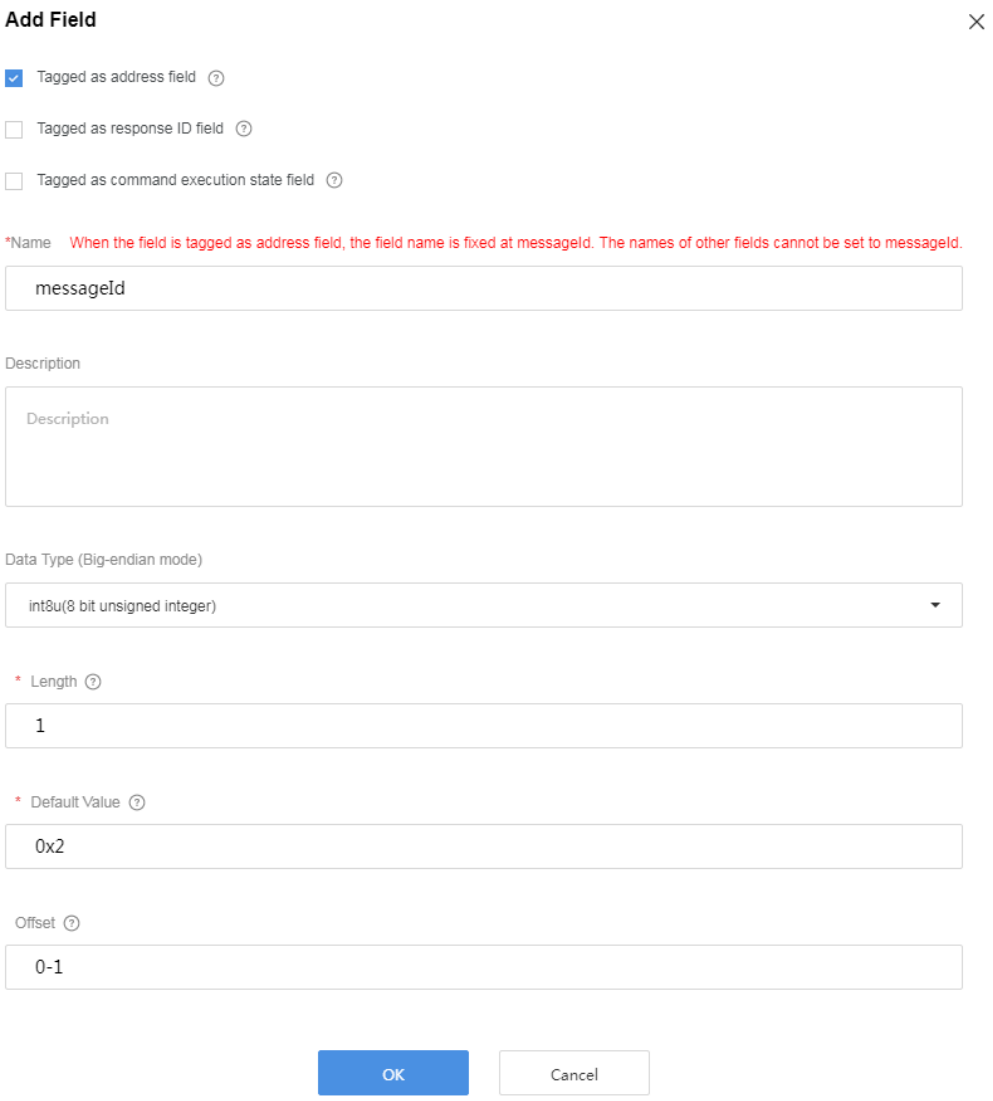
- **Name** can contain only letters, digits, underscores (_), and dollar signs ($) and cannot start with a digit.
- **Data Type** is configured based on the data reported by the device and must match the type of the corresponding field defined by the profile.



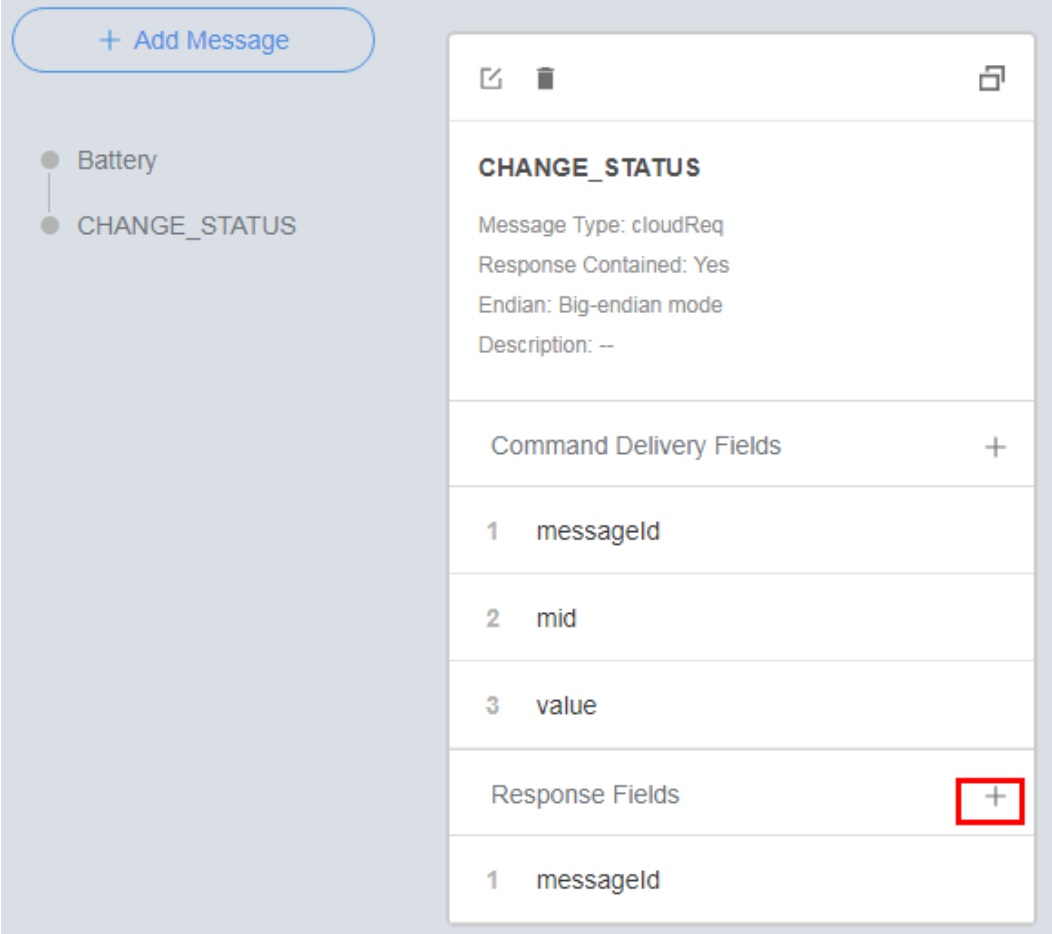
Step 16 Click + next to **Response Fields**.



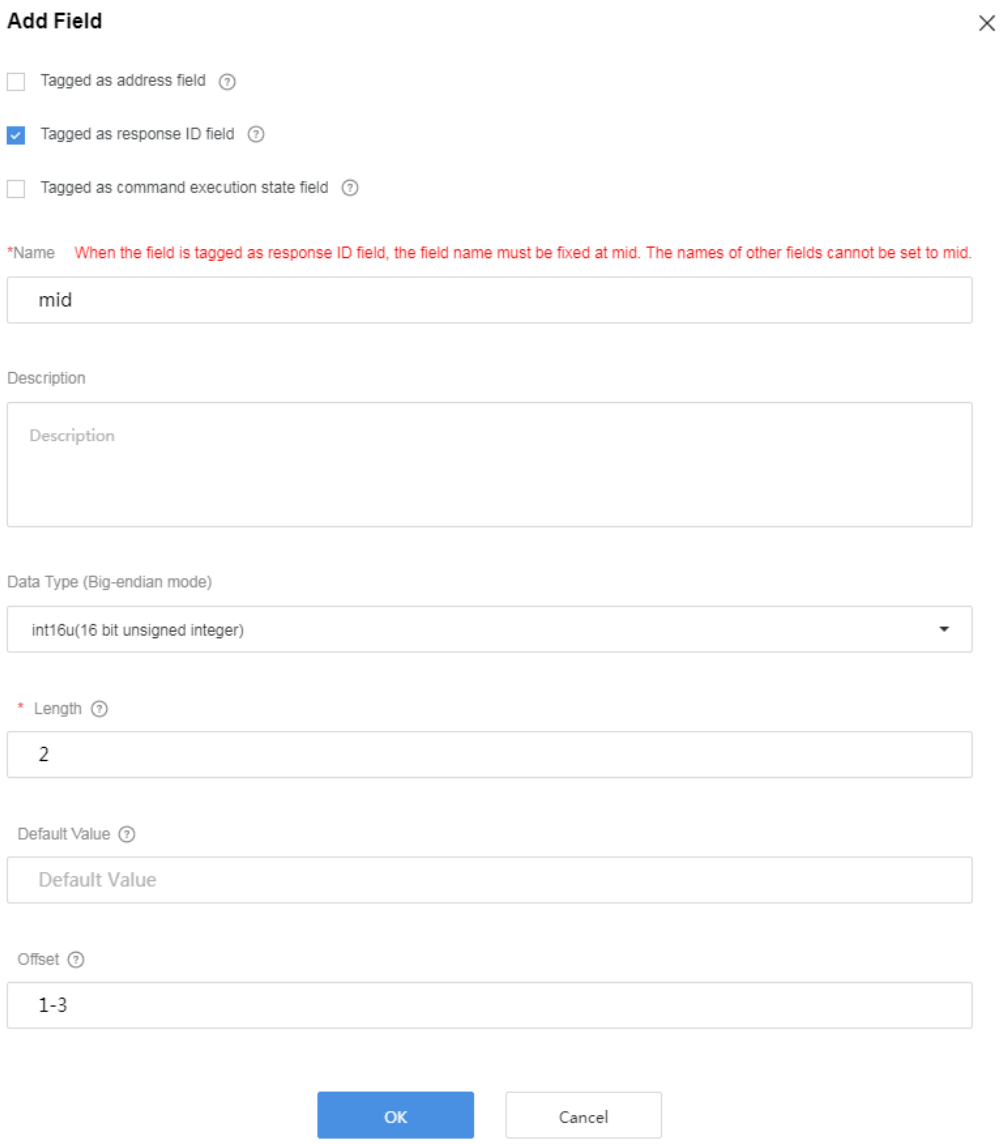
Step 17 In the **Add Field** dialog box displayed, select **Tagged as address field** and other parameters will be set automatically. Then, click **OK**.



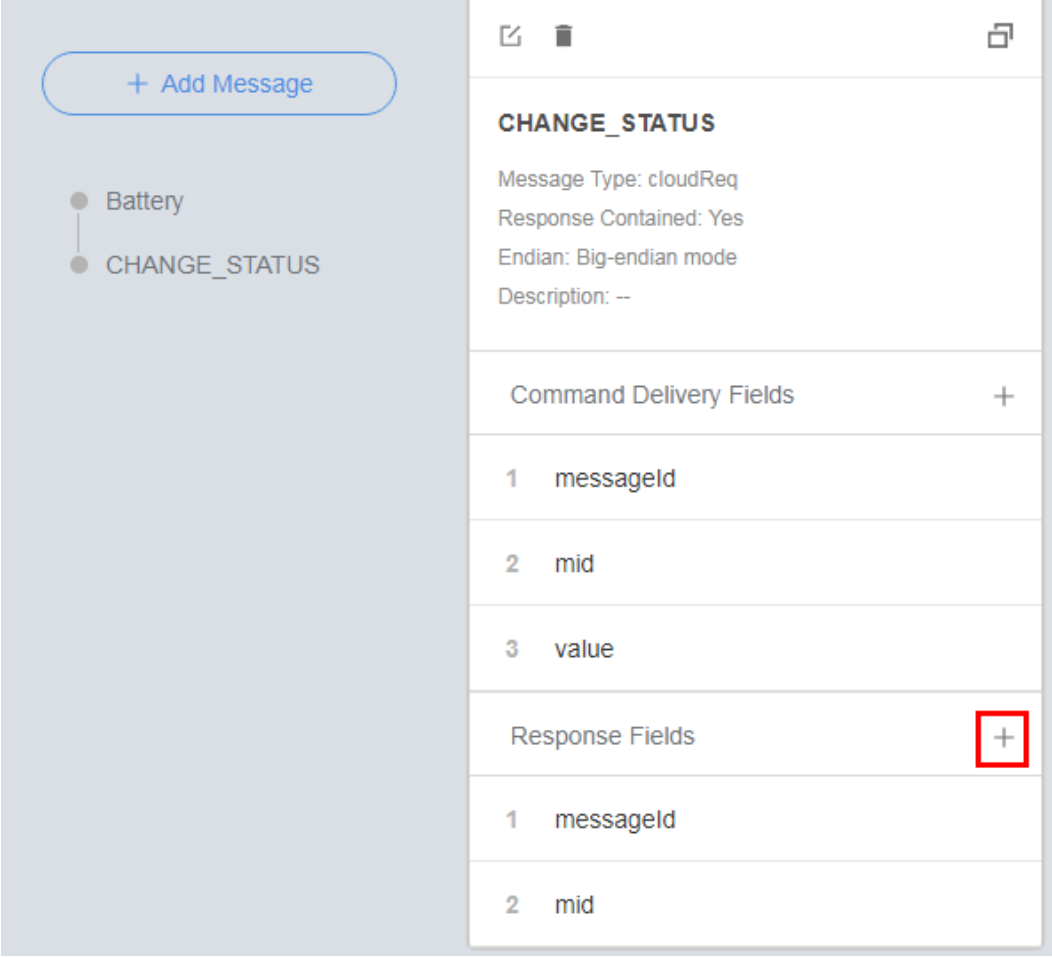
Step 18 Click + next to **Response Fields**.



Step 19 In the **Add Field** dialog box displayed, select **Tagged as response ID field** and other parameters will be set automatically. Then, click **OK**.

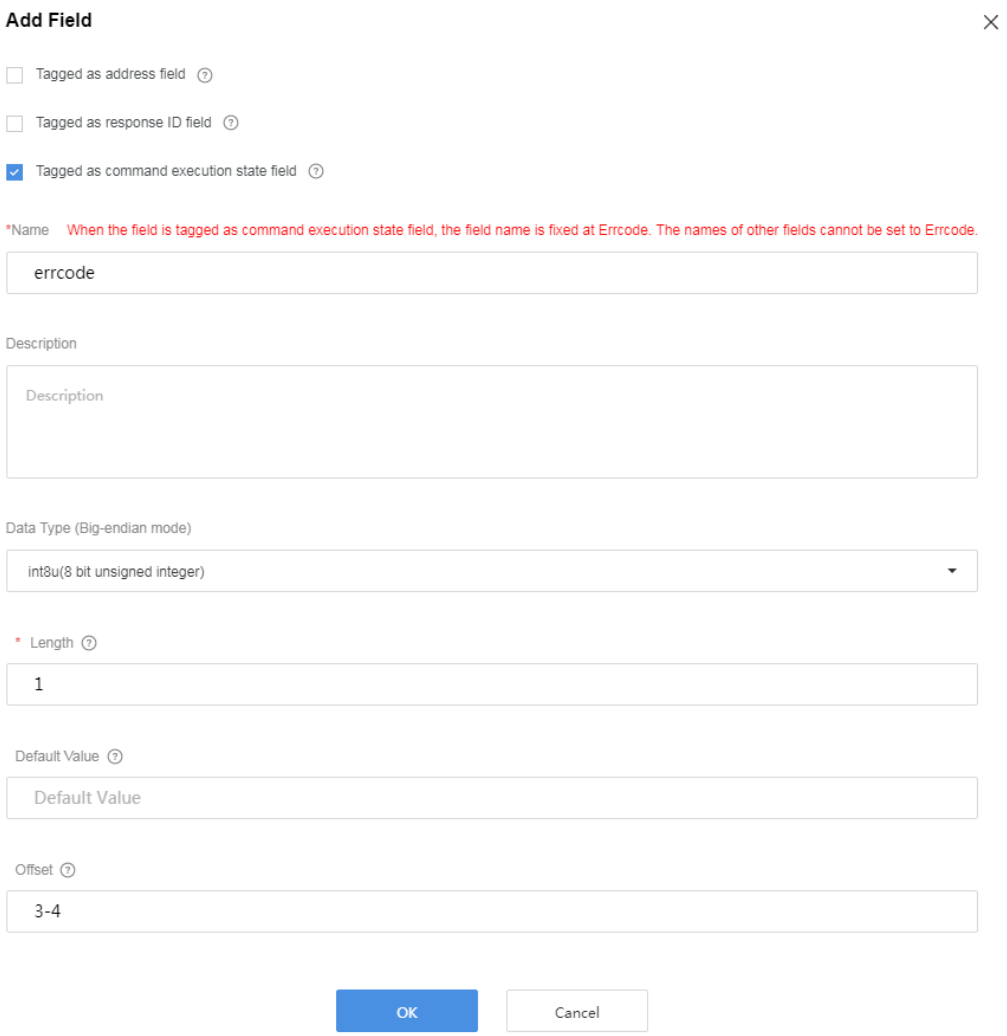


Step 20 Click + next to **Response Fields**.

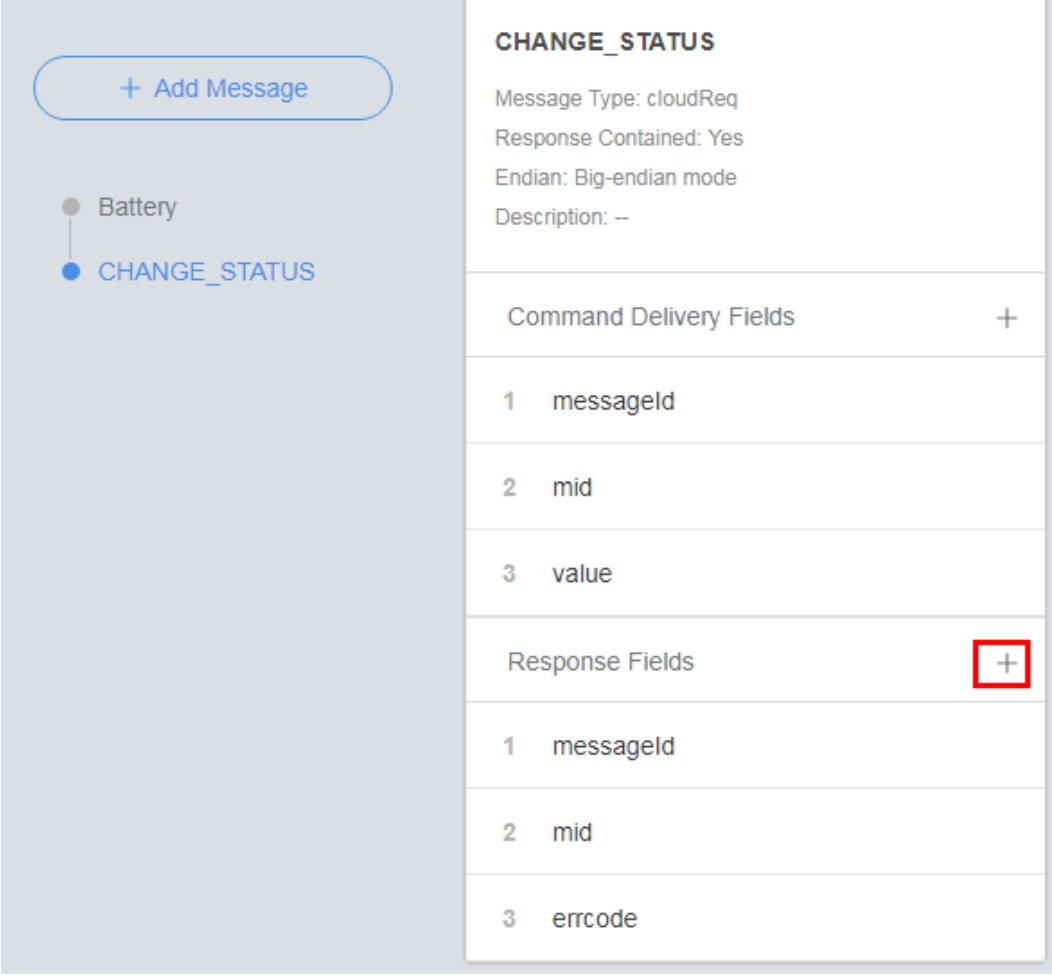


Step 21 In the **Add Field** dialog box displayed, select **Tagged as command execution state field**, set the parameters, and click **OK**.

- The value of **Name** is automatically populated.
- **Data Type** is configured according to the actual command response and must match the type of the corresponding field defined by the profile.

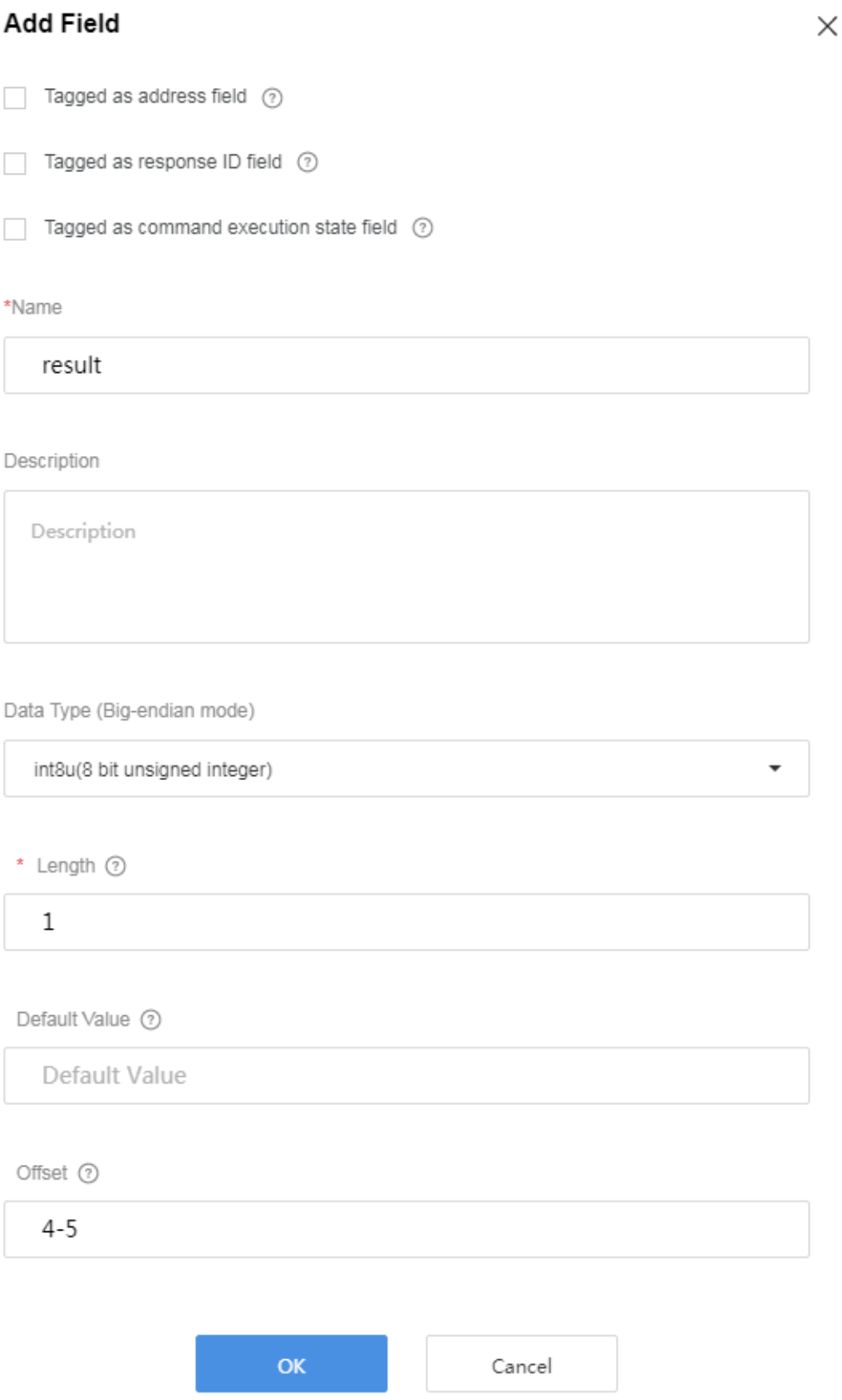


Step 22 Click + next to **Response Fields**.

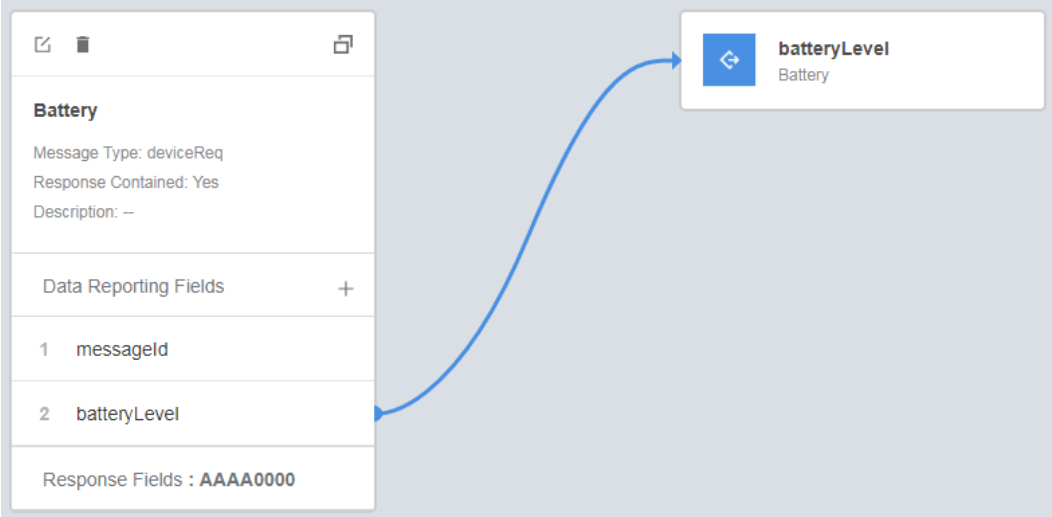


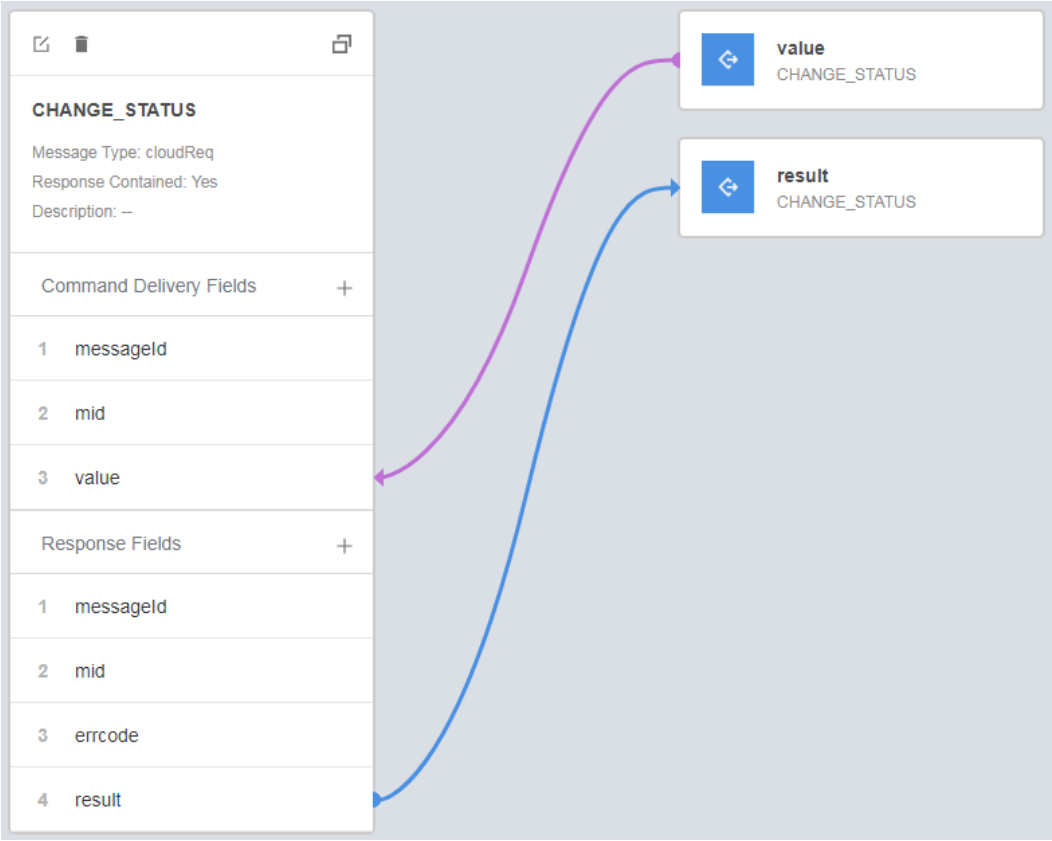
Step 23 In the **Add Field** dialog box displayed, set the parameters and click **OK**.

- **Name** can contain only letters, digits, underscores (_), and dollar signs ($) and cannot start with a digit.
- **Data Type** is configured based on the data reported by the device and must match the type defined by the profile.

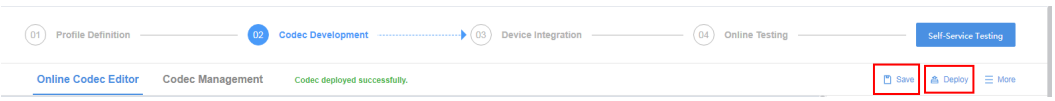


Step 24 Map the property fields, command fields, and response fields in **Device Model** on the right with the corresponding fields in the data reporting message, command delivery message, and command response.





Step 25 Click **Save** and then **Deploy** to deploy the codec on the IoT platform.



----End

2.5.2.5 Device Development

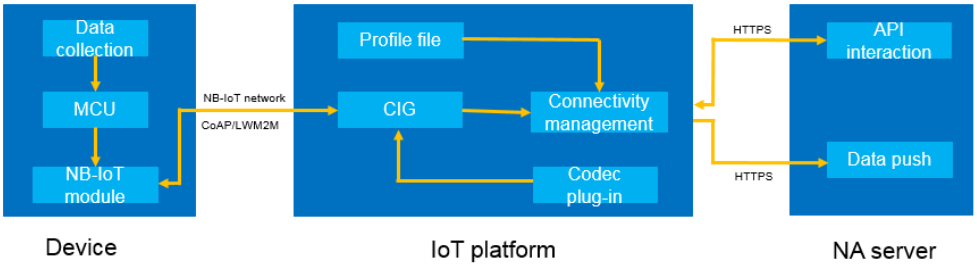
Overview

Based on the protocol used by the device to connect to the IoT platform, there are two access scenarios:

- Access using CoAP or LWM2M. In this scenario, devices can connect to the IoT platform by integrating NB-IoT modules or LiteOS SDK.

  Integrating NB-IoT Modules

  Devices integrated with NB-IoT modules can connect to the IoT platform through NB-IoT networks.



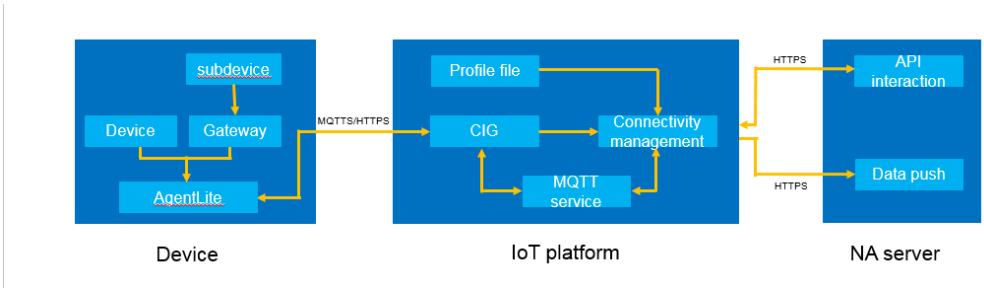
Features	● Wide coverage: The gain is 20 dB higher than that of LTE. ● Low power consumption: The solution focuses on applications with small data volume at a low rate. ● Massive amounts of connections: A single sector supports a maximum of 50,000 connections. ● Low cost: NB-IoT chipsets or modules are cost-effective for its low rate, low power consumption, and low bandwidth.
Scenarios	Low requirements on data timeliness, small data packets, fixed locations, and power supply from batteries. For example, smart metering and smart street lamp.
Applicable Networks	● NB-IoT network: constructed by carriers ● NB-IoT SIM card: purchased from NB-IoT network carriers ● NB-IoT module: purchased from the module manufacturers.
Communication Protocols	CoAP/LWM2M

  Integrating LiteOS SDK

  The LiteOS SDK is a lightweight SDK integrated on the device. Its features are as follows:

Features	● Protocols and security details are shielded. Users can focus on their applications without paying attention to the implementation of protocols and security policies. ● An adaptation layer is provided. Users can migrate LiteOS SDK by adapting only a few interfaces. ● Data reported by devices can be cached and retransmission and acknowledgment mechanisms are provided to ensure data reporting reliability. ● Firmware upgrade, resumable download, and integrity protection for firmware packages are supported. ● Security and non-security connection modes are supported.
Running Environment Requirements	RAM > 32 KB FLASH > 128 KB
Applicable Networks	NB-IoT, 2G/3G/4G, and wired networks
Communication Protocols	CoAP and LWM2M

- Access using MQTT/MQTTS. In this scenario, devices can connect to the IoT platform by integrating AgentLite SDK.



AgentLite SDK is a lightweight SDK integrated on the device. Its features are as follows:

Features	● Multiple network access modes are supported, such as Wi-Fi, 2G/3G/4G, and wired network. ● After devices are integrated with the lightweight SDKs, they can be connected to the IoT platform by calling APIs. ● Data in JSON format is used for frequent communication and communication involving huge amount of data.

Running Environment Requirements	RAM > 4 MB FLASH > 600 KB Currently, the following platforms are supported: ARM Linux (Embedded Linux) MIPS Linux (Embedded Linux) x86 Linux x86_64 Linux x86 Windows x86_64 Windows Android (Java)
Applicable Networks	2G/3G/4G and wired network
Communication Protocols	HTTPS, MQTT, and MQTTS

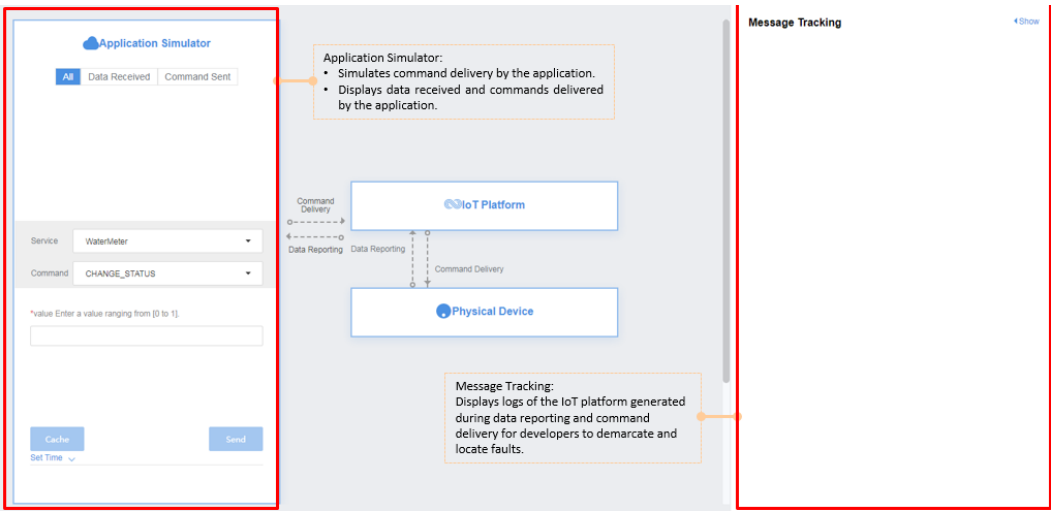
2.5.2.6 Online Testing

Overview

Online testing supports device simulation and application simulation. It offers scenarios such as data reporting and command delivery to test devices, profile files, and codecs.

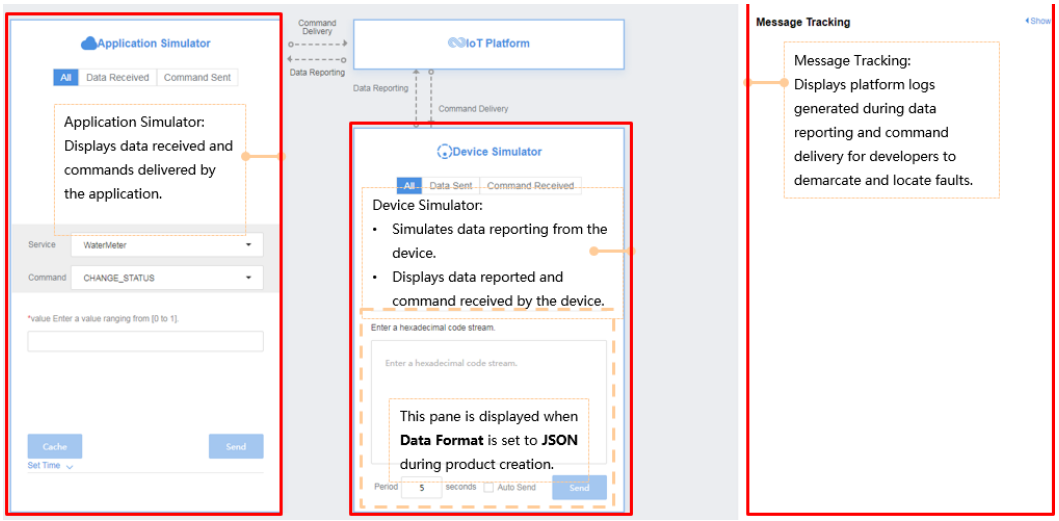
You can use physical or virtual devices for online testing.

- When the device development is complete but the application development is not, you can add physical devices and use the application simulator to test devices, profile files, and codecs. The interface of online testing using a physical device is as follows:



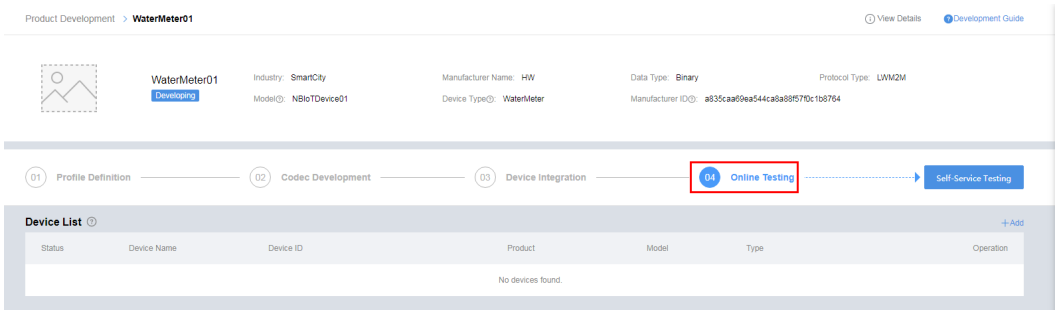
- When both device development and application development are not completed, you can create virtual devices and use the application simulator and device simulator to test

profile files and codecs. The interface of online testing using a virtual device is as follows:

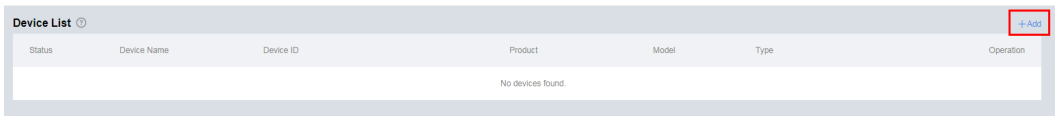


Using a Physical Device for Online Testing

Step 1 In the product development space, click **Online Testing**.

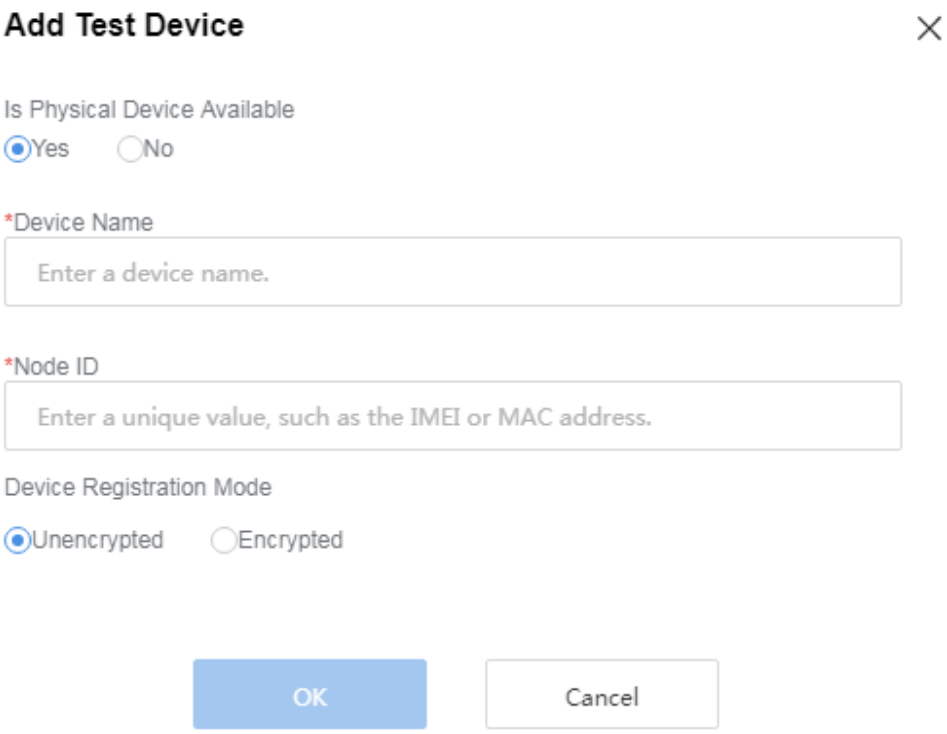


Step 2 Click **Add** at the row where **Device List** resides.

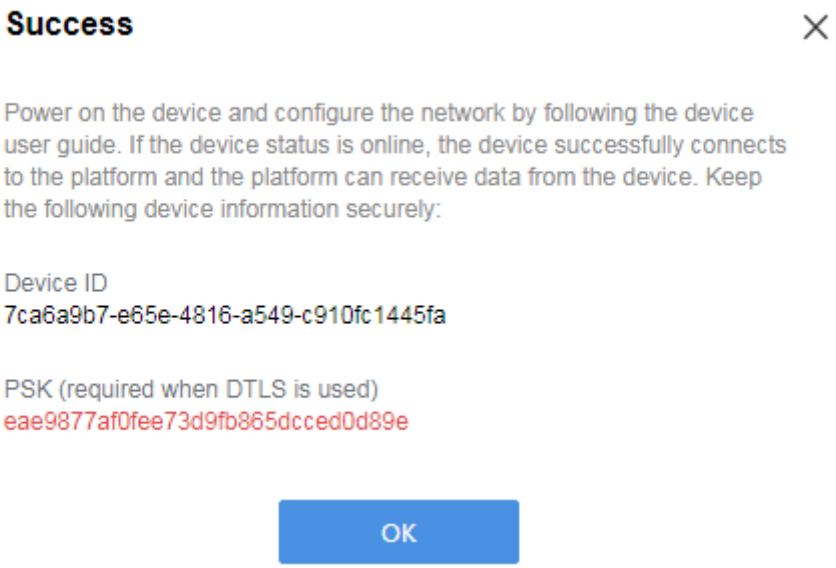


Step 3 In the **Add Test Device** dialog box displayed, select **Yes**, set the parameters, and click **OK**.

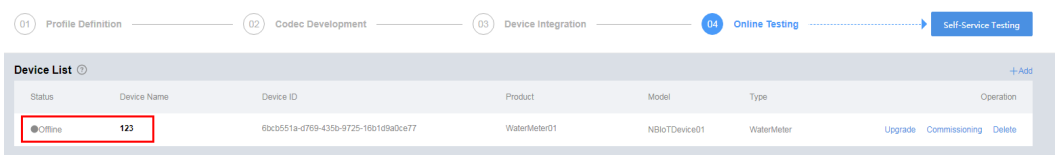
- **Device Name** can contain only letters, digits, and underscores (_) and must be unique in the product.
- **Node ID** must be set to a unique value, such as the IMEI or MAC address of the device.
- Choose **Unencrypted** or **Encrypted** based on site requirements.



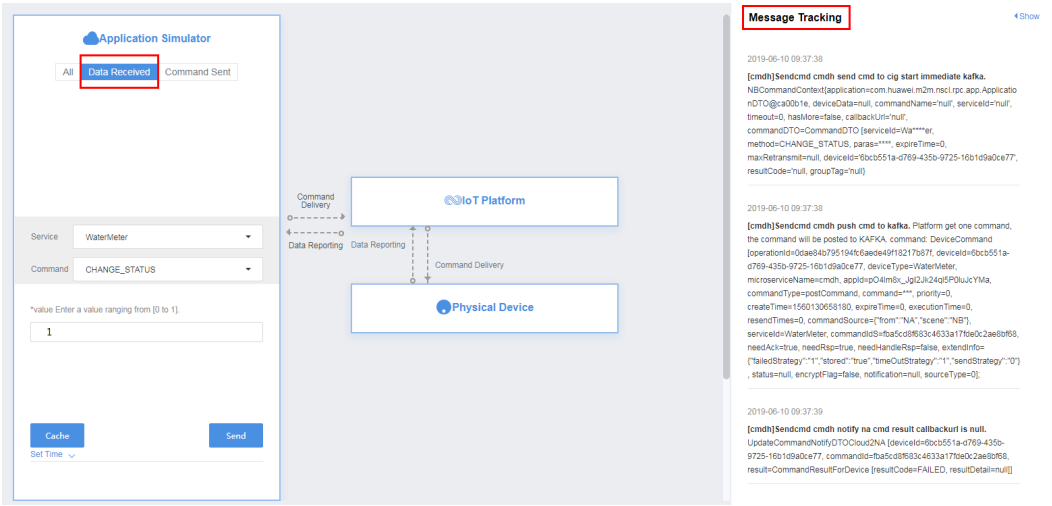
After the device is added, **Device ID** and **PSK** are returned. Keep the PSK securely as it is required when the device uses DTLS to connect to the IoT platform.



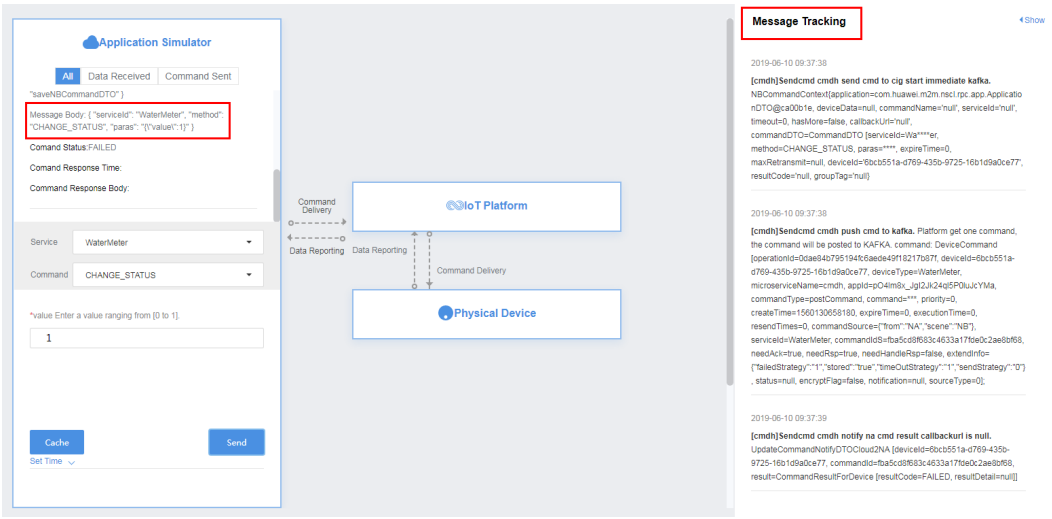
Step 4 In the device list, select the newly added physical device to enter the **Online Testing** page.



Step 5 Connect the device to the IoT platform and report data. View the data reporting result in **Application Simulator** and processing logs of the IoT platform in **Message Tracking**.



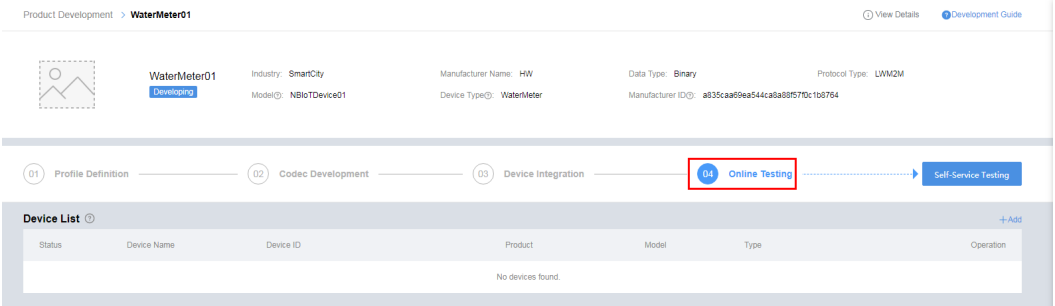
Step 6 Deliver a command in **Application Simulator**. View processing logs of the IoT platform in **Message Tracking** and check the received command on the device.



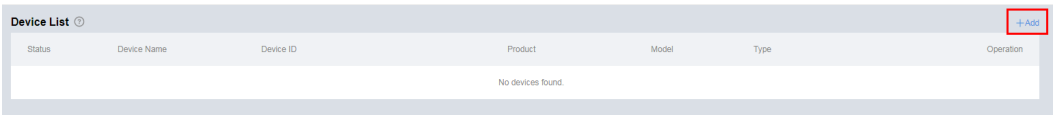
----End

Using a Virtual Device for Online Testing

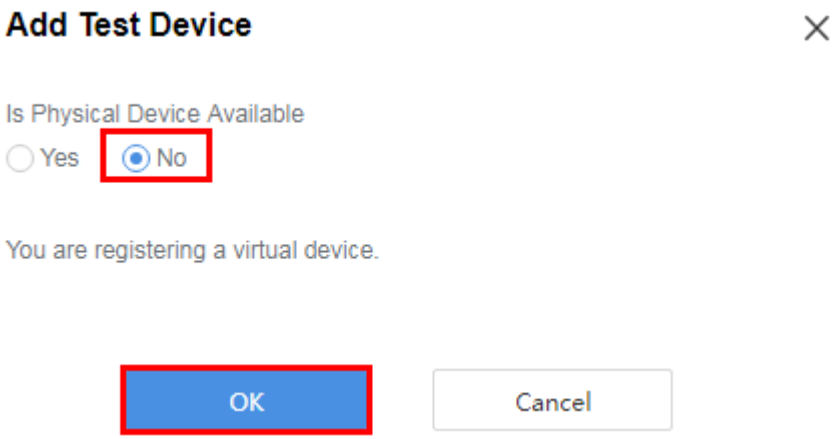
Step 1 In the product development space, click **Online Testing**.



Step 2 Click **Add** at the row where **Device List** resides.

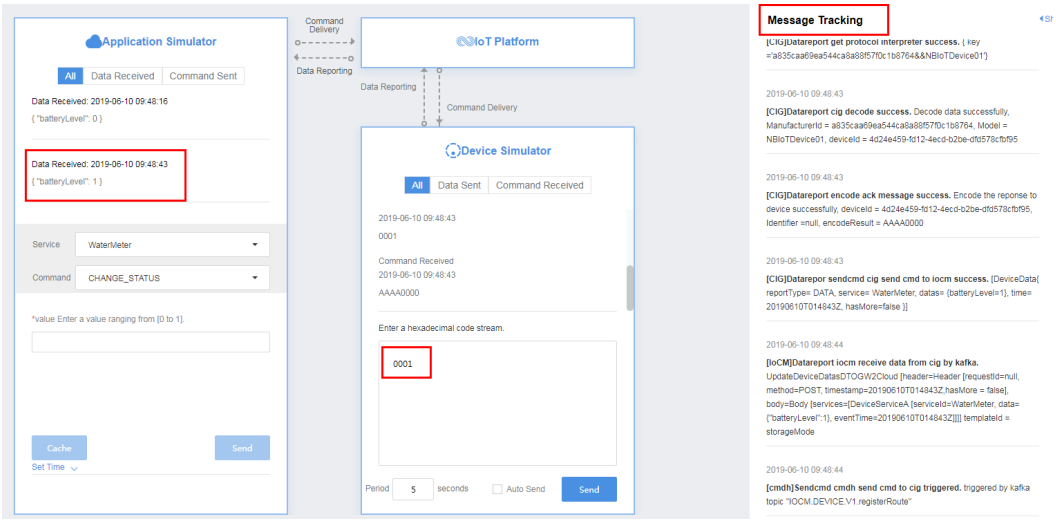


Step 3 In the **Add Test Device** dialog box displayed, select **No** and click **OK**.

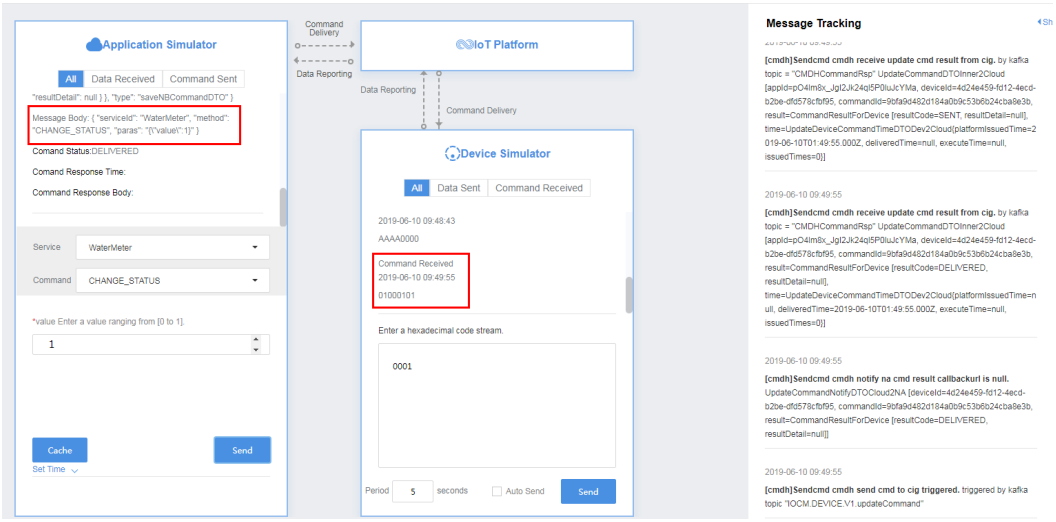


Step 4 In the device list, select the newly added virtual device to enter the **Online Testing** page. The name of the virtual device is in the format of **Product Name**+**Simulator**. Only one virtual device can be added for each product.

Step 5 In **Device Simulator**, enter a hexadecimal code stream and click **Send**. Then, view the data reporting result in **Application Simulator** and processing logs of the IoT platform in **Message Tracking**.



Step 6 Deliver a command in **Application Simulator**. View the received command (for example, a hexadecimal code stream) in **Device Simulator** and processing logs of the IoT platform in **Message Tracking**.



----End

2.5.2.7 Self-Service Testing

Overview

Self-service testing provides end-to-end test cases to help developers test basic device capabilities, such as data reporting and command delivery. It aims to help you find product defects or problems and shorten the time to market (TTM). After the testing is complete, a test report is generated by the Developer Center for product release certification.

Prerequisites

You have defined the product profile, developed the codec, and deployed the codec.

Procedure

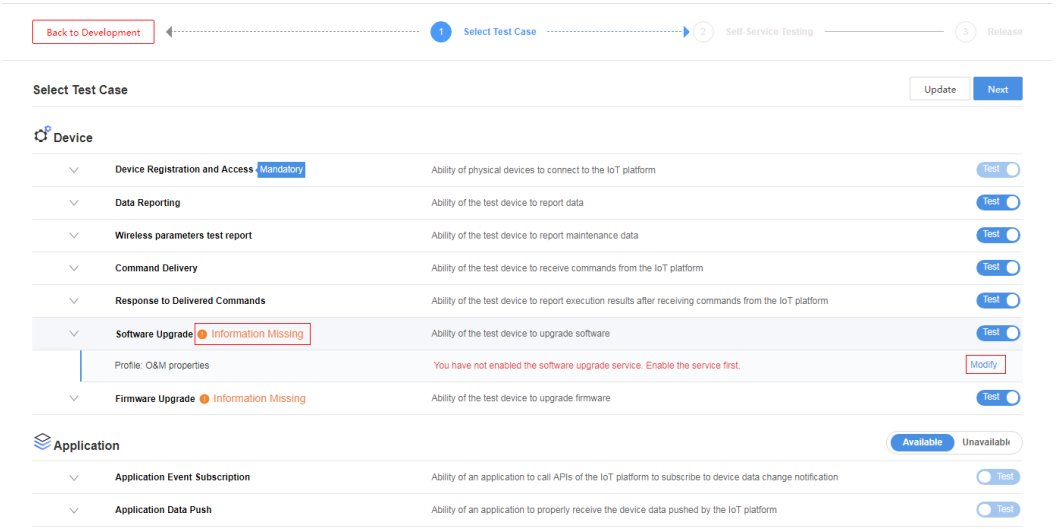
Step 1 In the product development space, click **Self-Service Testing**.

Step 2 The **Select Test Case** page is displayed. You can select test cases as needed. The system automatically checks whether the selected test cases meet the test requirements and returns the check results.

- If all selected test cases pass the check, click **Next** to proceed to the next phase.
- If a test case fails to pass the check, click **Information Missing** on the right of the test case and modify the profile file or codec as prompted.

  NOTE

  - Before starting the self-service testing, either **Data Reporting** or **Command Delivery** must be selected, in addition to the mandatory test case.
  - The more cases of a product pass the test, the higher the pass rate of the product release to the Product Center. It is recommended that either **Software Upgrade** or **Firmware Upgrade** be selected and all other test cases be included.



Step 3 Perform the self-service testing as prompted. After the testing is complete, you can preview the test report or apply for releasing the product.

----End

2.5.2.8 Product Release

Overview

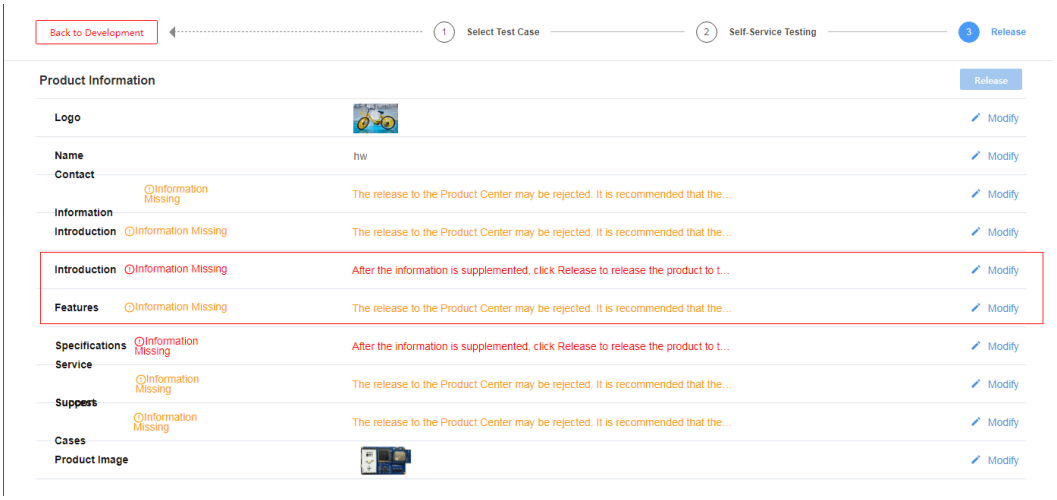
If the Developer Center has interconnected with the Product Center, you can apply to the Product Center for product release. You can release your product and display it in the Product Center or set it visible only to yourself.

Applying for Product Release

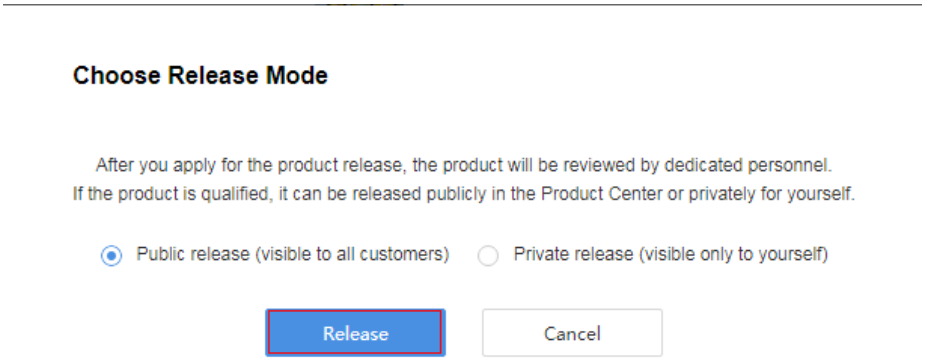
Step 1 Click **Apply for Release** after the product passes the test cases.

Step 2 The system automatically checks the integrity of the manufacturer and product information. If no important information is missing, click **Release**.

- Information missing in yellow: Some information is incomplete, which does not affect the product release. However, the product may fail to be approved for release in the Product Center. It is recommended that the information be supplemented.
- Information missing in red: Important information is missing. The product can be released only after the information is supplemented.



Step 3 Select a release mode and click **Release**.



----End

2.5.3 Device Management

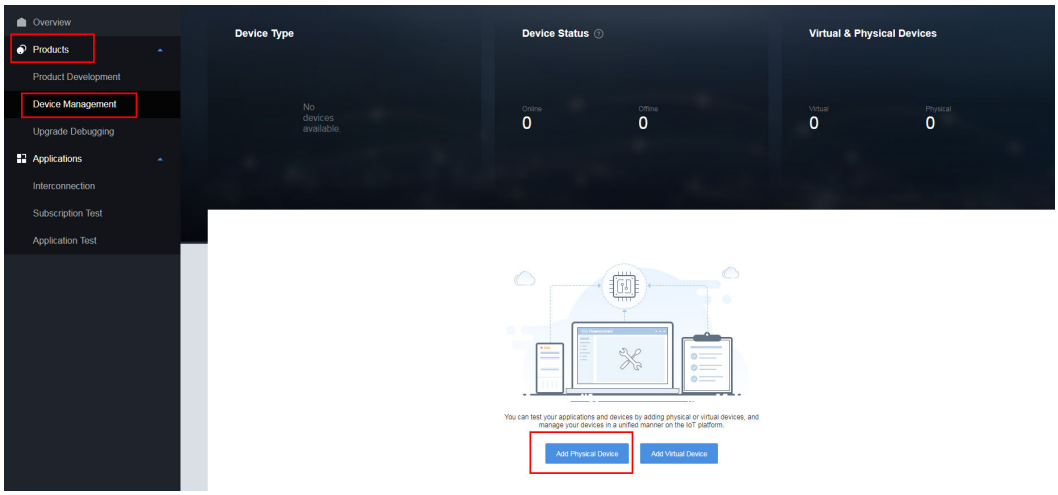
Overview

Device Management displays all physical and virtual devices of a project, and provides functions such as statistics by type, online testing, and device logs for developers to manage devices and locate faults.

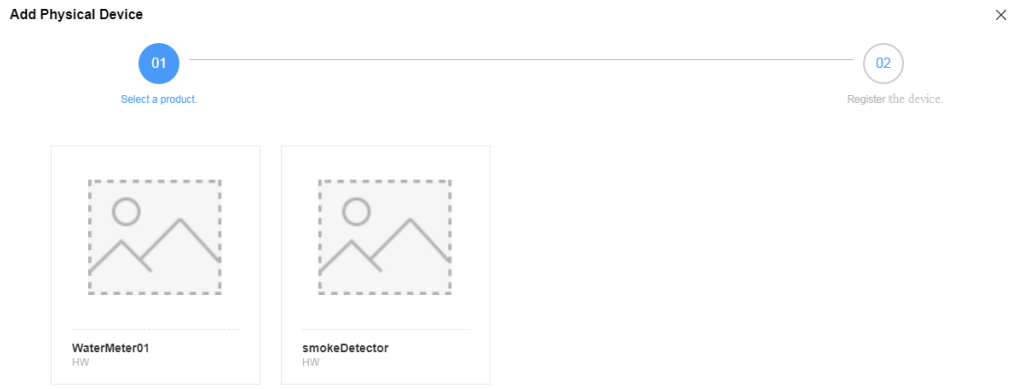
- Add a physical device on the **Device Management** page. The device name is defined by the developer. When the device development is complete, developers can add physical devices on the Developer Center where developers can perform end-to-end testing on the physical devices, codecs, and NAs.
- Add a virtual device on the **Device Management** page. The device name is defined by the IoT platform. The name of the virtual device is in the format of **Product Name** +**Simulator**. Only one virtual device can be added for each product. When the device development is not complete, developers can create a virtual device on the Developer Center to test the codecs and NAs.

Adding a Physical Device

Step 1 Choose **Products** > **Device Management** and click **Add Physical Device**.

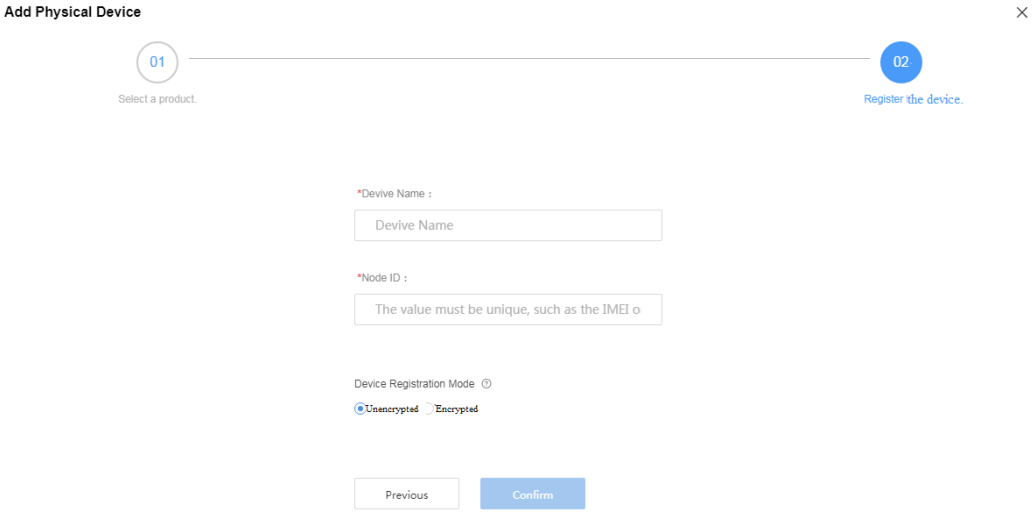


Step 2 In the **Add Physical Device** dialog box displayed, select a device.

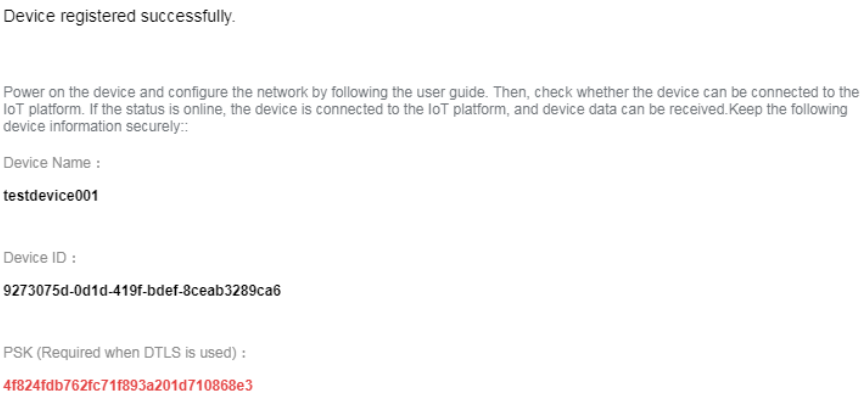


Step 3 Configure device information and click **Confirm**.

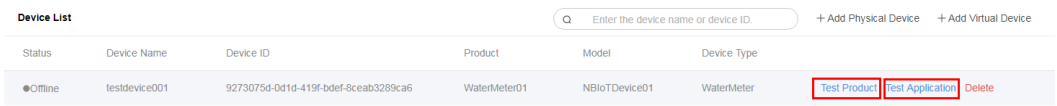
- **Device Name** can contain only letters, digits, and underscores (_) and must be unique in the product.
- **Node ID** must be set to a unique value, such as the IMEI or MAC address of the device.
- Select **Unencrypted** or **Encrypted** based on site requirements. If this parameter is set to **Unencrypted**, the device uses the CoAP/UDP protocol to connect to the IoT platform. If this parameter is set to **Encrypted**, the device uses the CoAPS/DTLS protocol to connect to the IoT platform.



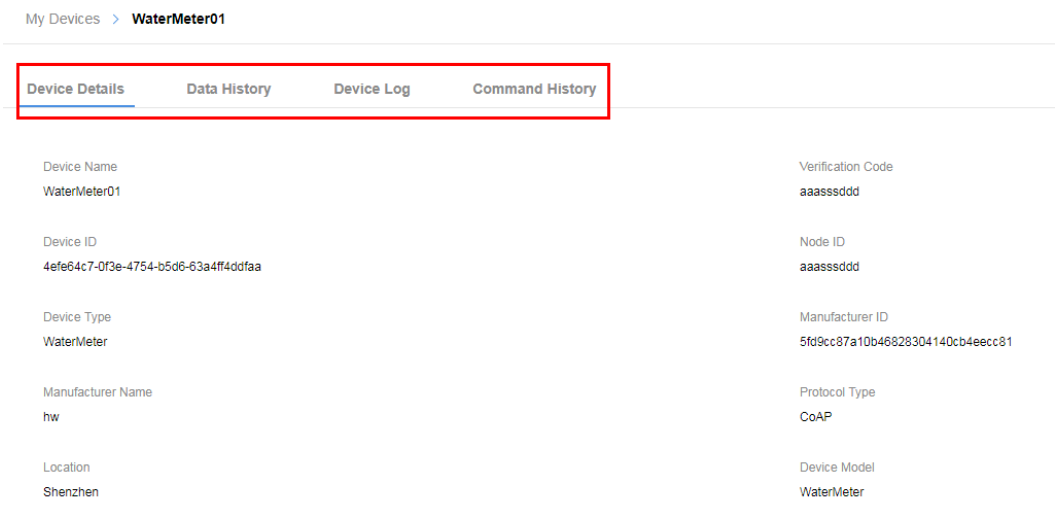
After the device is added, **Device ID** and **PSK** are returned. Keep the PSK securely as it is required when the device uses DTLS to connect to the IoT platform.



Step 4 After a physical device is added, you can view details and perform tests on the device and the application.

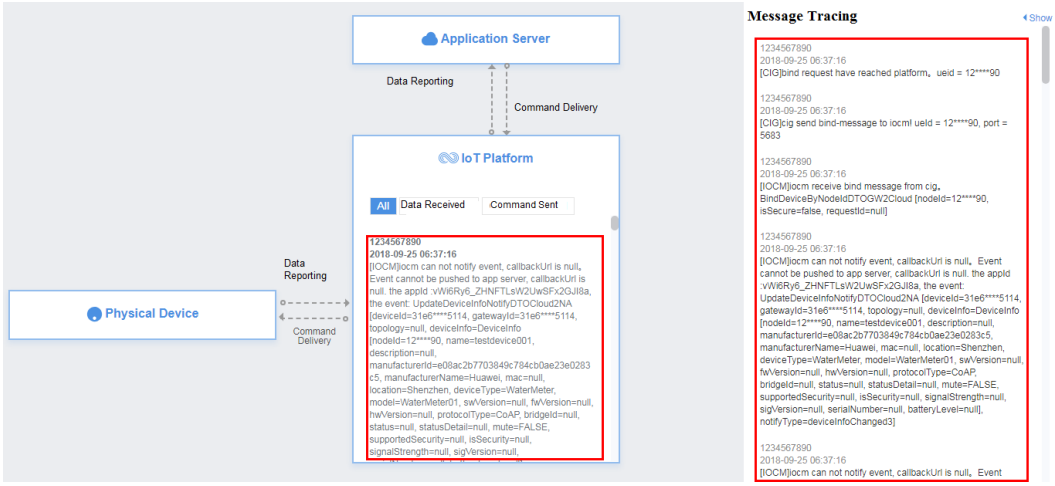


- In the device list, click the newly added device. On the page displayed, you can view device information, historical data, logs, and historical commands.

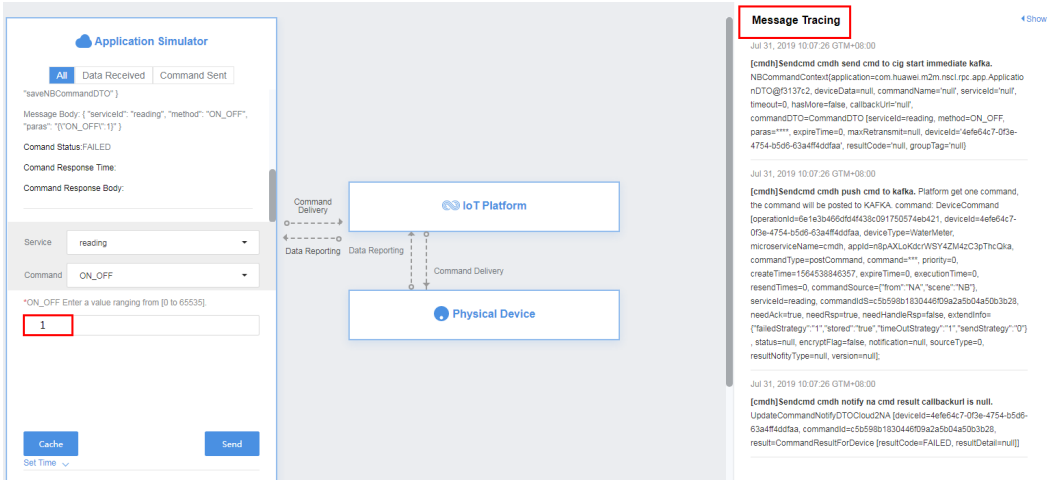


- In the device list, click **Test Product** at the row where the newly added device resides to test the product.

  Connect the device to the IoT platform and report data. View the data reporting result in **Application Simulator** and processing logs of the IoT platform in **Message Tracing**.



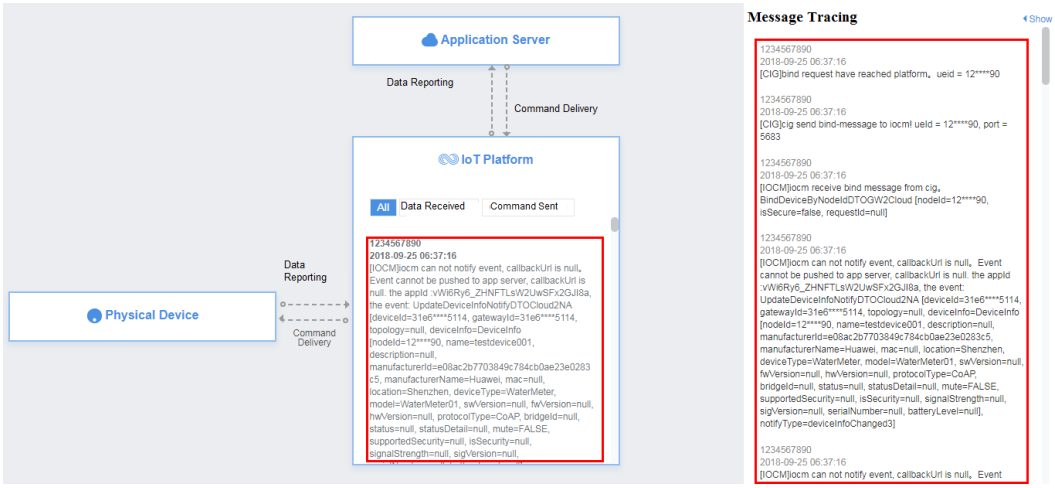
  Deliver a command in **Application Simulator**. View processing logs of the IoT platform in **Message Tracing** and check the received command on the device.



- In the device list, click **Test Application** at the row where the newly added device resides to test the application.

  Connect the device to the IoT platform and report data. View the data reporting result in **IoT Platform** and **Application Simulator** and processing logs of the IoT platform in **Message Tracing**.

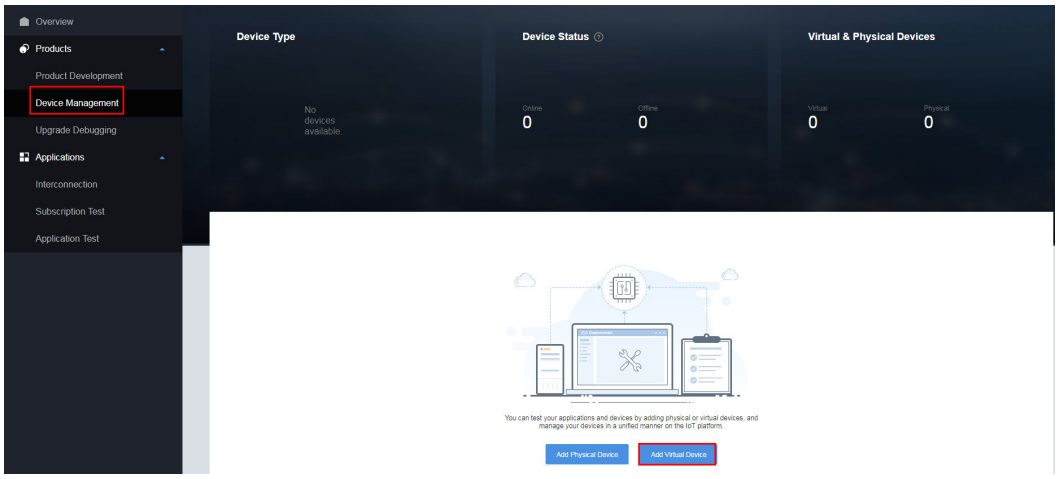
  Connect the NA to the IoT platform and deliver a command. View the command delivery result in **IoT Platform** and on the device and processing logs of the IoT platform in **Message Tracing**.



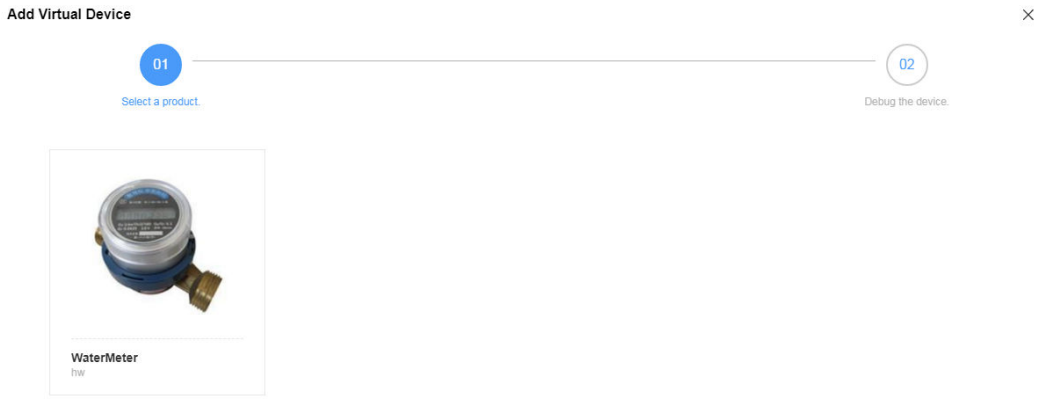
----End

Adding a Virtual Device

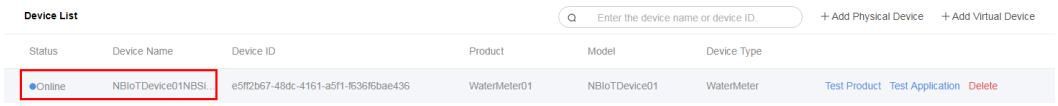
Step 1 Choose **Products** > **Device Management** and click **Add Virtual Device**.



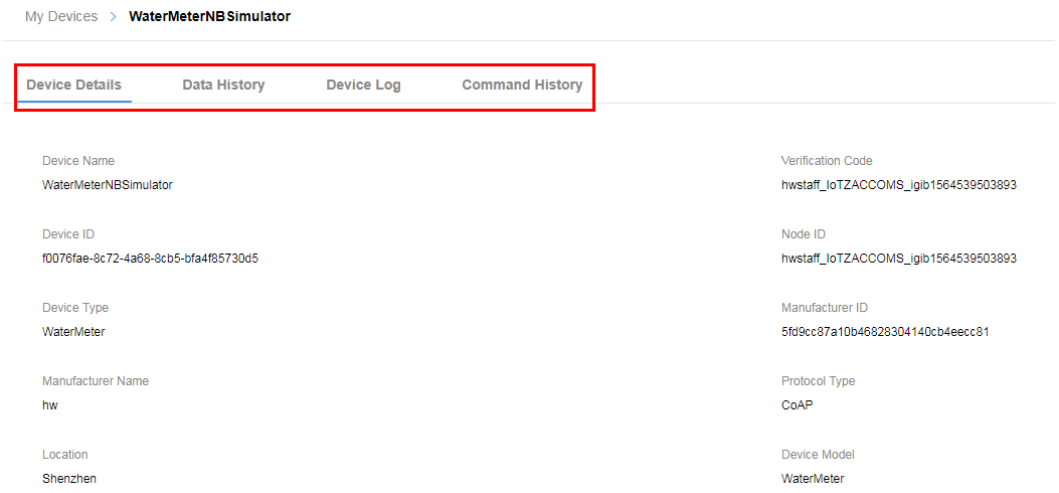
Step 2 In the **Add Virtual Device** dialog box displayed, select a device.



Step 3 After a virtual device is added, you can view details and perform testing on the device and the application.

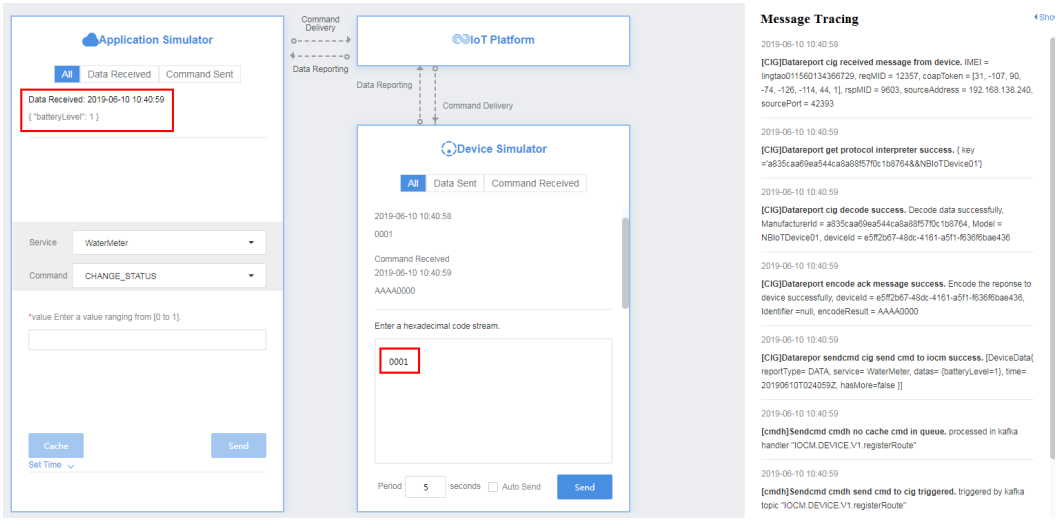


- In the device list, click the newly added device. On the page displayed, you can view device information, historical data, logs, and historical commands.

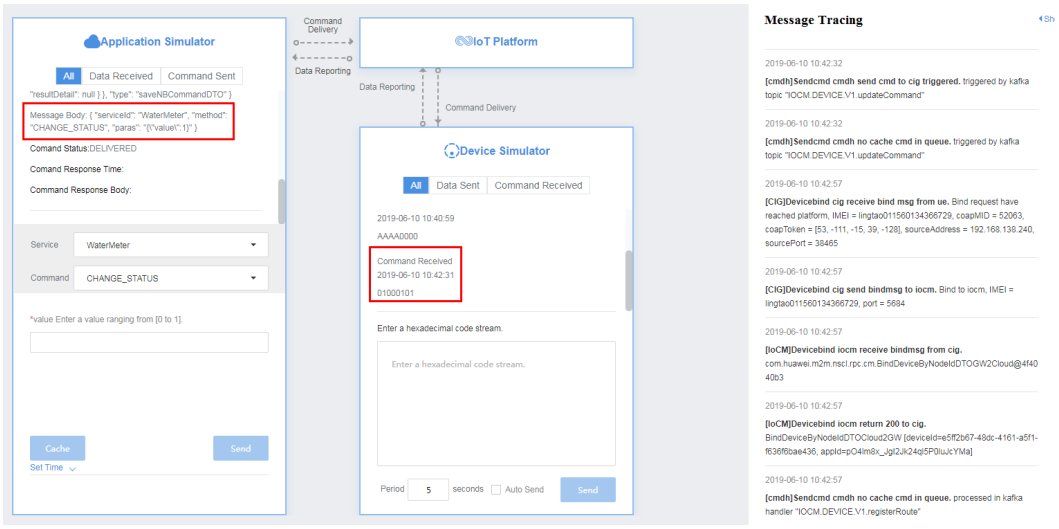


- In the device list, click **Test Product** at the row where the newly added device resides to test the product.

  In **Device Simulator**, enter a hexadecimal code stream or JSON data (for example, enter a hexadecimal code stream) and click **Send**. Then, view the data reporting result in **Application Simulator** and processing logs of the IoT platform in **Message Tracing**.



Deliver a command in **Application Simulator**. View the received command (for example, a hexadecimal code stream) in **Device Simulator** and processing logs of the IoT platform in **Message Tracing**.



- In the device list, click **Test Application** at the row where the newly added device resides to test the application.

  In **Device Simulator**, enter a hexadecimal code stream or JSON data (for example, enter a hexadecimal code stream) and click **Send**. Then, view the data reporting result in **IoT Platform** and **Application Simulator** and processing logs of the IoT platform in **Message Tracing**.

After the NA delivers a command, view the received command (for example, a hexadecimal code stream) in **Device Simulator** and view processing logs of the IoT platform in **Message Tracing**.

----End

2.5.4 Upgrade Debugging

2.5.4.1 Overview

Upgrade debugging enables you to remotely upgrade the firmware and software of devices. When a new software or firmware version is released, developers can remotely upgrade the device and manage the new firmware or software upgrade package in real time.

2.5.4.2 Firmware Upgrade

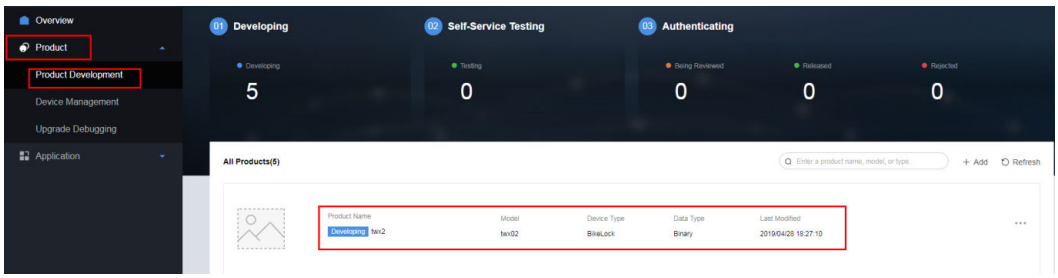
Overview

The performance of NB-IoT chipset is constantly updated and optimized, and the chipset firmware of devices needs to be upgraded accordingly. The upgrade of NB-IoT modules is called firmware upgrade.

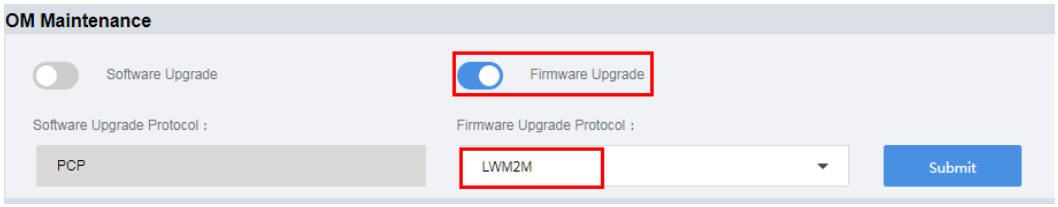
Prerequisites

Before upgrading the firmware, ensure that the device supports the firmware upgrade.

Step 1 Choose **Product** > **Product Development**. Click a product name to enter the product space.



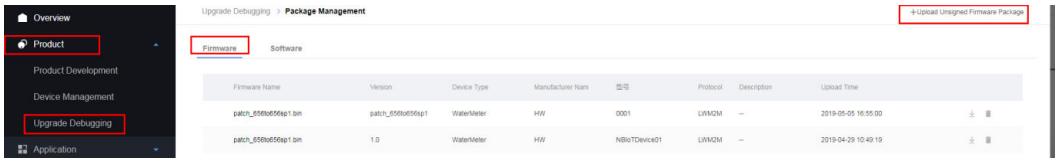
Step 2 In **Define Profile** section to view **OM Maintenance** details, and ensure that **Filmware Upgrade** function is enabled.



----End

Uploading a Firmware Package

Step 1 Choose **Product** > **Upgrade Debugging** > **Upgrade Package Management** > **Firmware**, and click **Upload Unsigned Firmware Package**.

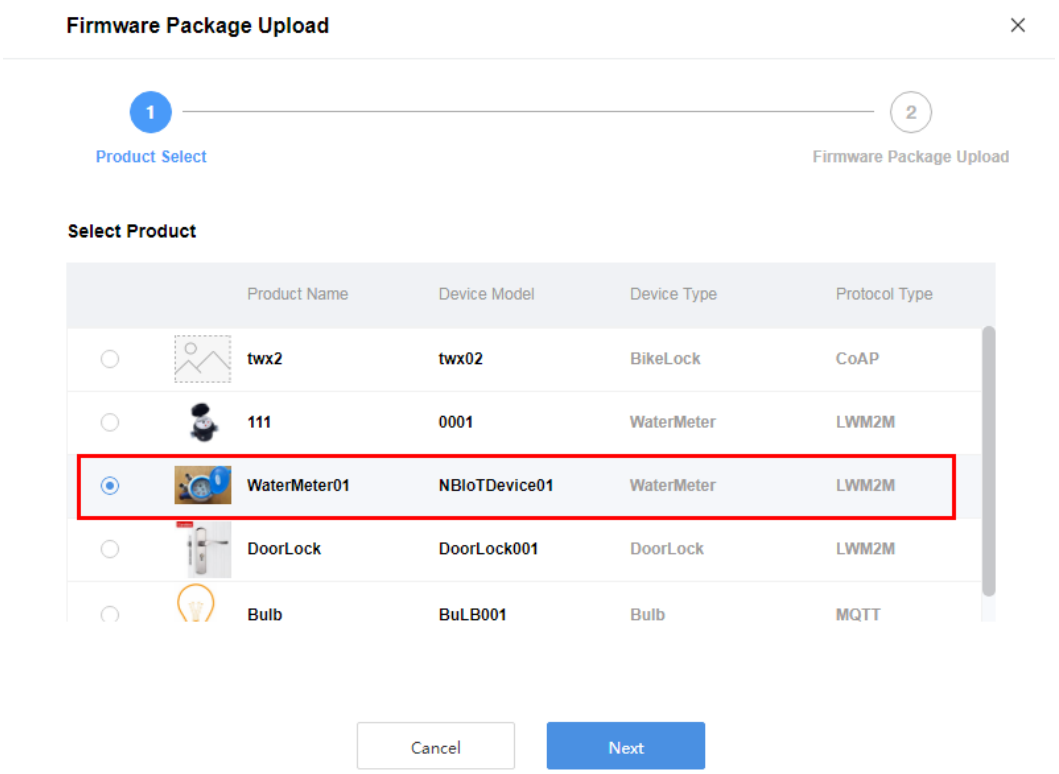


Step 2 Upload the firmware package according to the wizard.

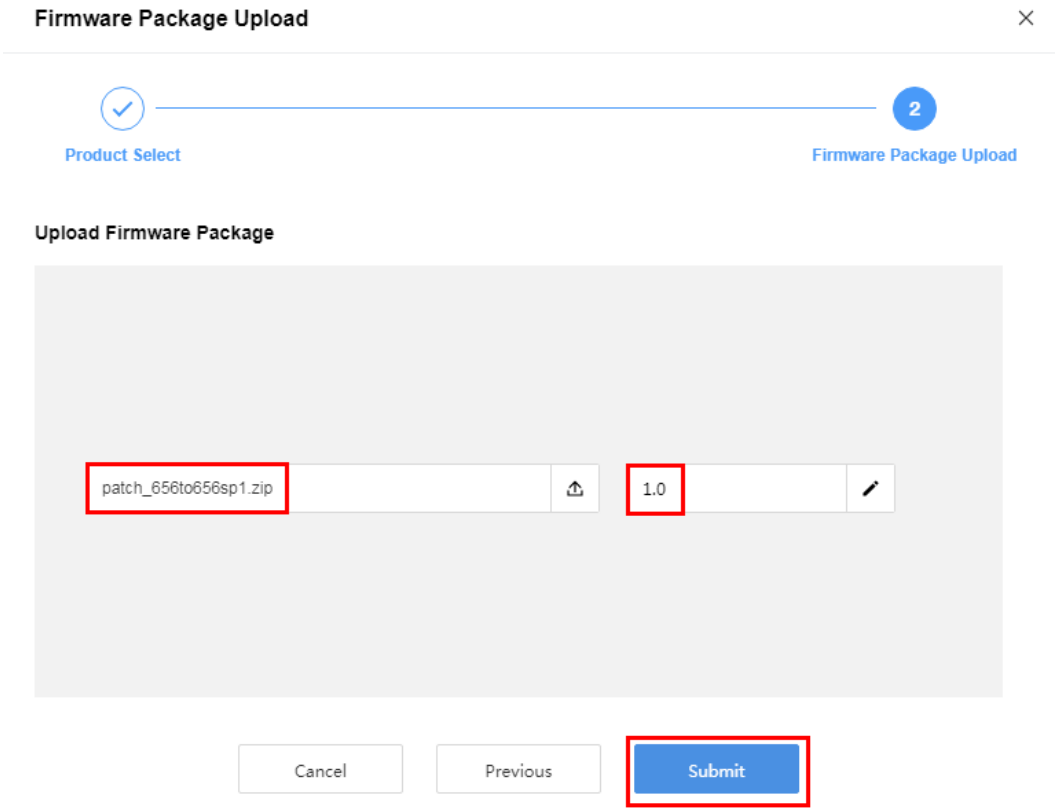
1. Select a product and click **Next**.

   NOTE

   The firmware package contains only the bin file used for firmware upgrade. Therefore, the Developer Center cannot directly obtain the product model information from the firmware package. You need to select a product and associate it with the firmware package.



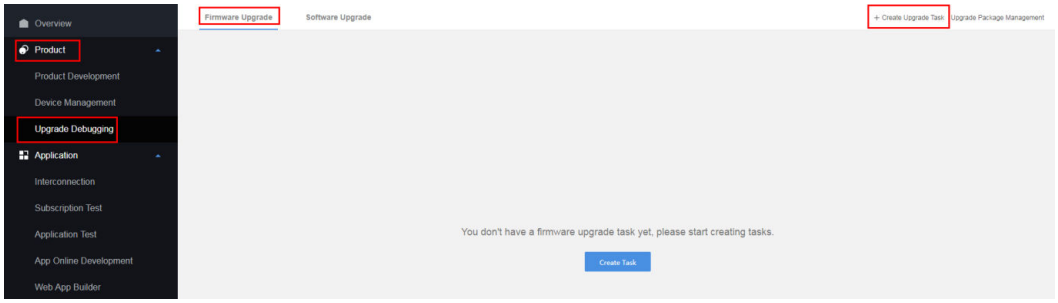
2. Select the unsigned firmware package to be uploaded, enter the version number of the firmware package, and click **Submit**.



----End

Creating a Firmware Upgrade Task

Step 1 Choose **Product** > **Upgrade Debugging** > **Firmware Upgrade**, and click **Create Upgrade Task**.



NOTE

If no upgrade task is available, click **Create Task** to create a firmware upgrade task.

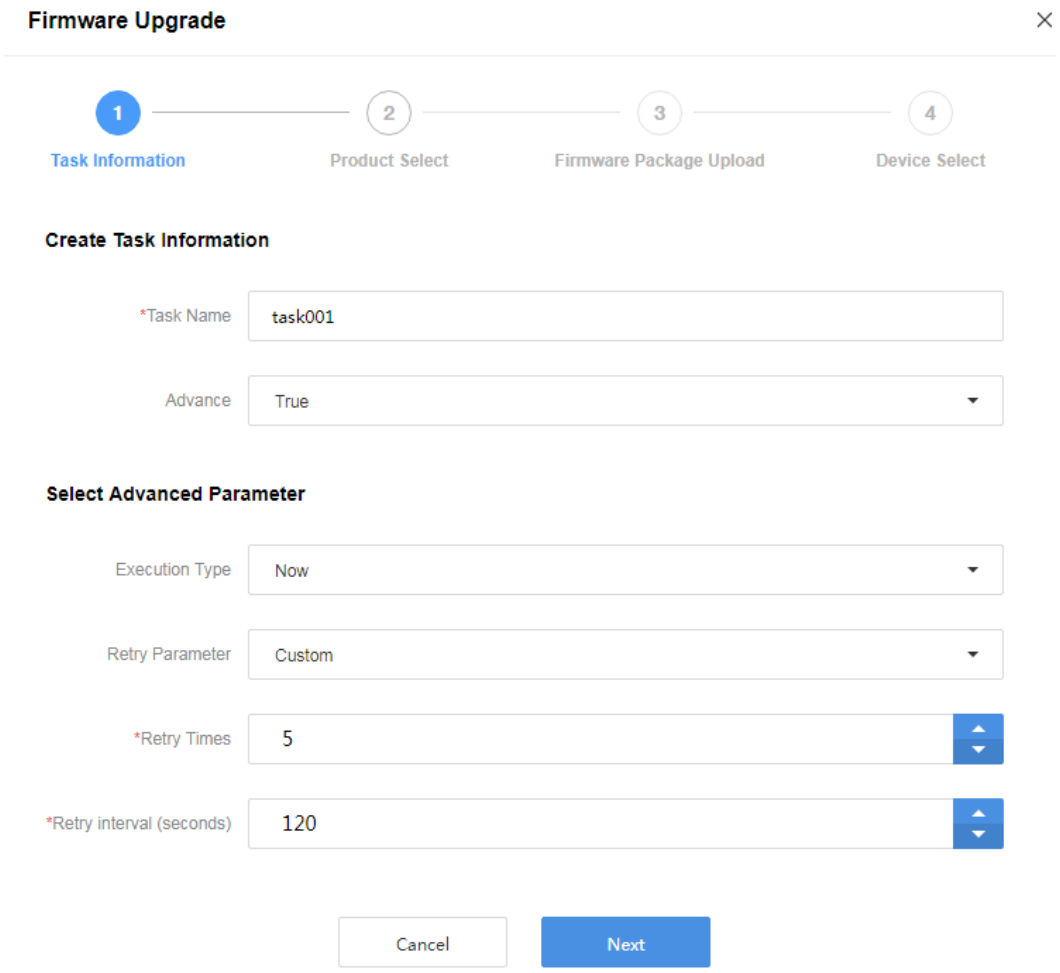
Step 2 Create the upgrade task according to the wizard.

1. Enter basic information and click **Next**.

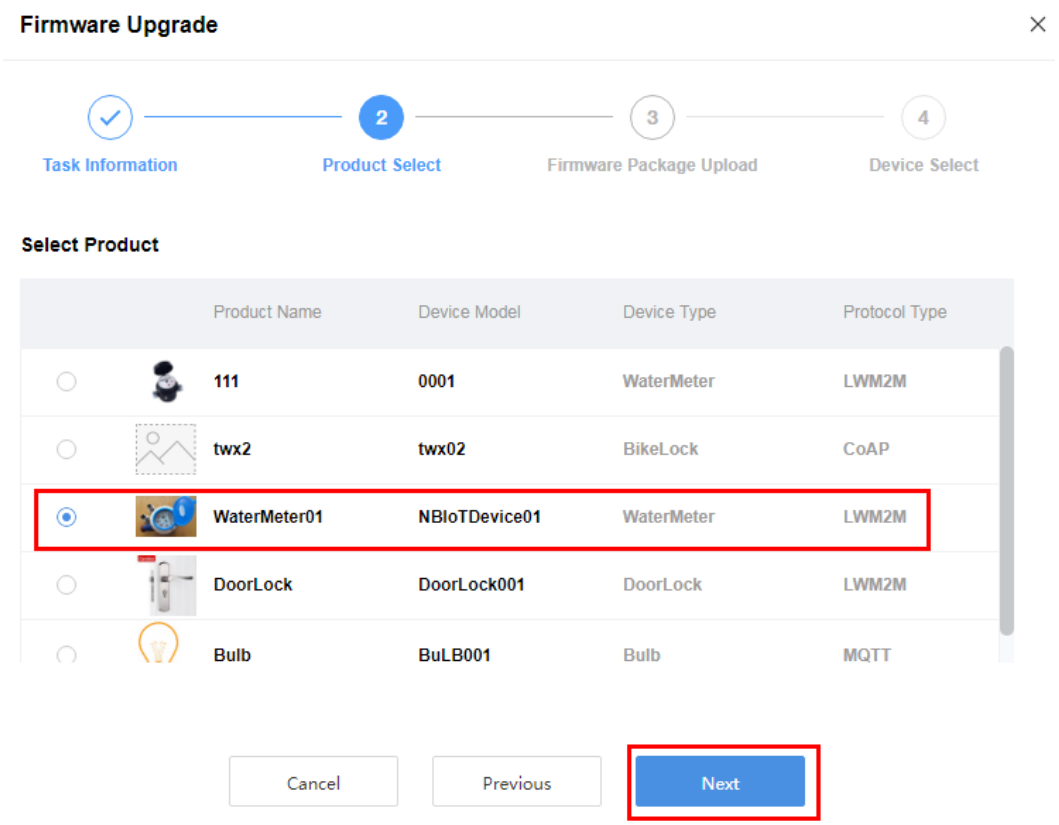
   If you need to configure the parameters **Execution Type** and **Retry Parameter**, set the parameter **Advance** to **True**.

   - **Execution Type** indicates the time when the Developer Center delivers an upgrade task to the device. The value options include **Now**, **Custom**, and **Device Online**.

- **Retry Parameter** indicates whether to execute failed task again.

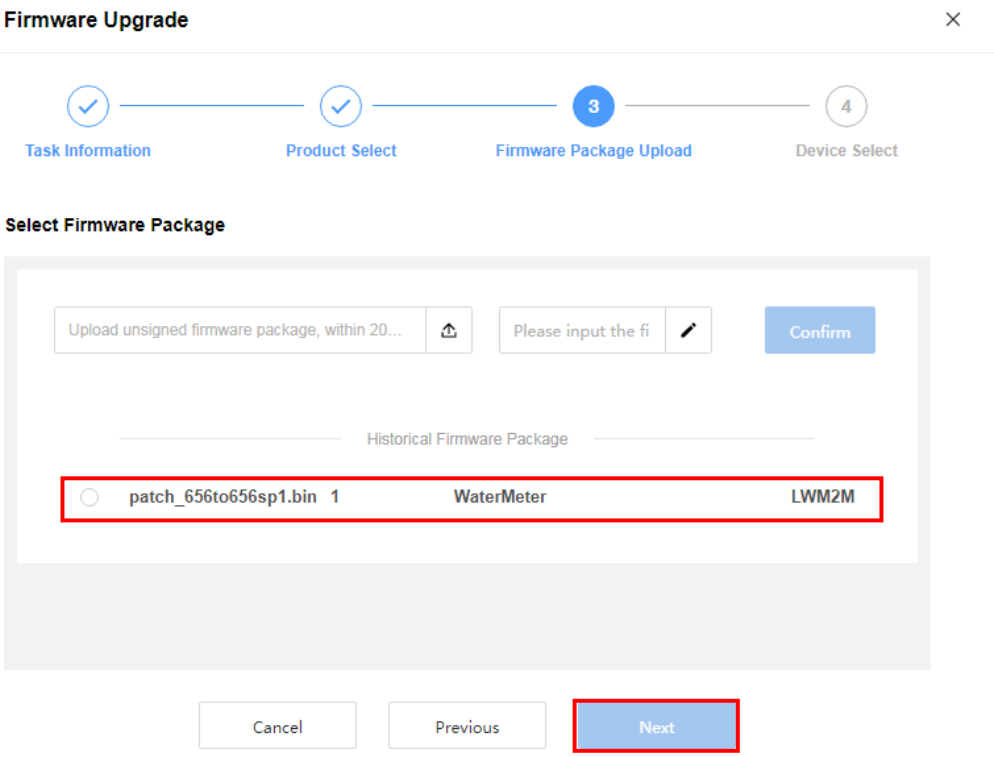


2. Select a product and click **Next**.

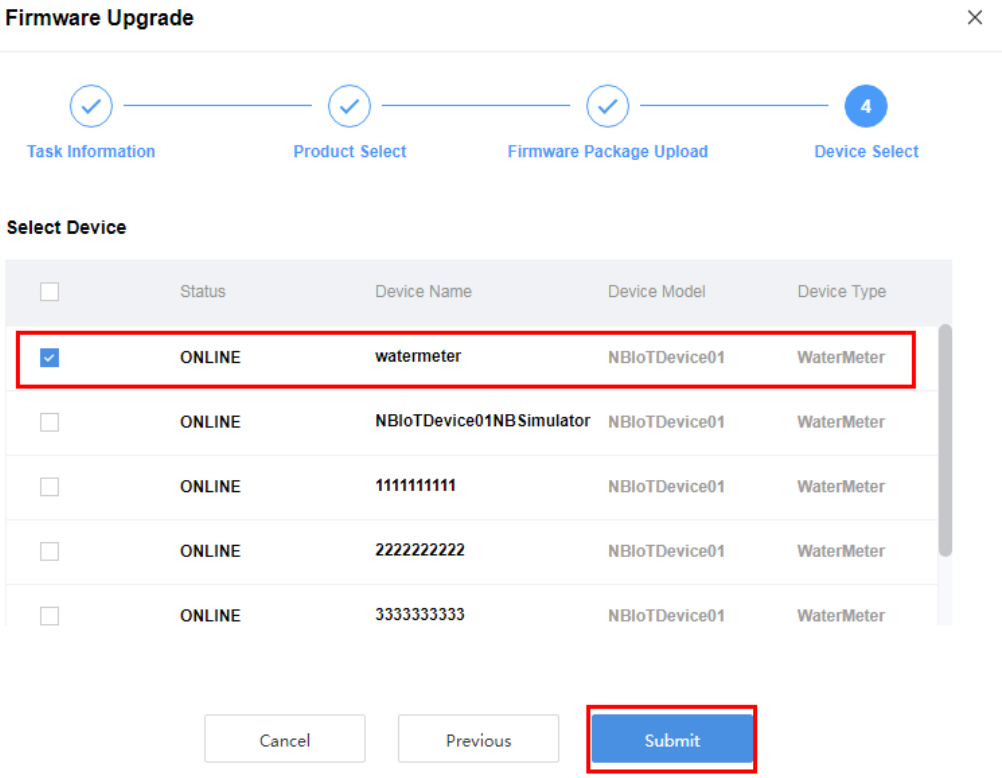


3. Select the firmware package to be upgraded and click **Next**.

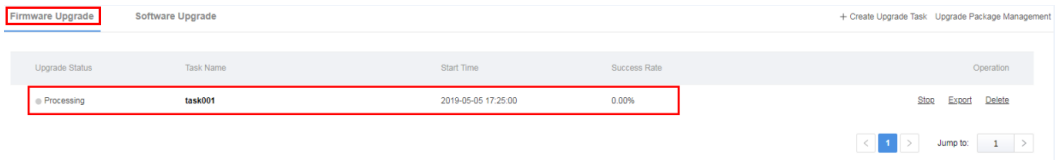
   If no required firmware package exists in the **Historical Firmware Package** list, upload a new unsigned firmware package. Ensure that the firmware package to be uploaded can be used for firmware upgrade.



4. In the **Device Select** tab page, all product devices are displayed. You can select one or more devices for upgrade and click **Submit**.



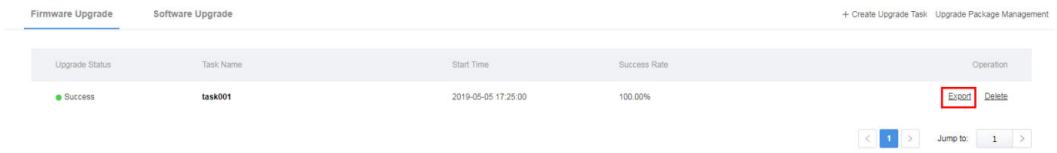
Step 3 After a firmware upgrade task is created, you can manage it on the **Firmware Upgrade** tab page. Click a task to view the **Basic Information** and **Upgrade Detail**.



NOTE

During the upgrade, service interaction is not allowed for the NB-IoT modules.

Step 4 After the task is complete, click **Export** to export the task details file.



----End

2.5.4.3 Software Upgrade

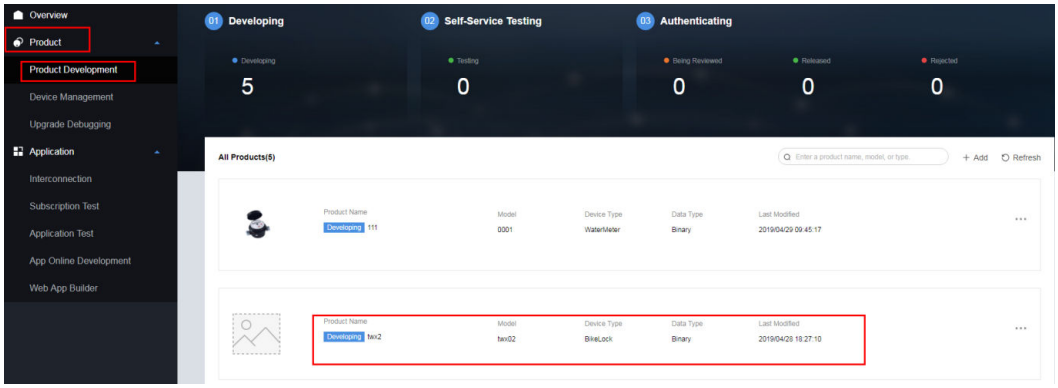
Overview

The IoT platform provides firmware upgrade function using LWM2M for NB-IoT modules. However, most NB-IoT modules do not provide the upgrade interface for the MCU. The MCU can be upgraded only based on the application layer. The MCU upgrade is called software upgrade.

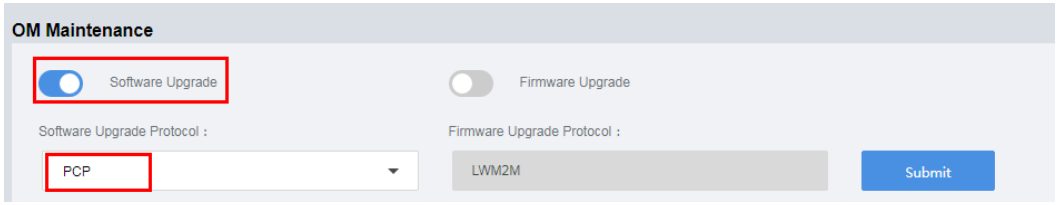
Prerequisites

Before upgrading the software, ensure that the device supports the software upgrade.

Step 1 Choose **Product** > **Product Development**. Click a product name to enter the product space.



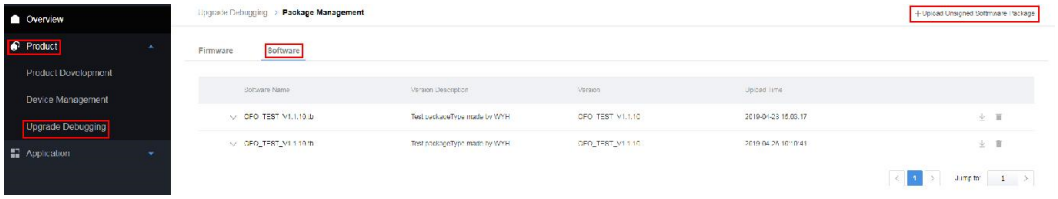
Step 2 In **Define Profile** section to view **OM Maintenance** details, and ensure that **Software Upgrade** function is enabled.



----End

Uploading Software Packages

Step 1 Choose **Product** > **Upgrade Debugging** > **Upgrade Package Management** > **Software**, and click **Upload Unsigned Software Package**.



Step 2 On the **Software Package Upload** tab page, select the unsigned software package to be uploaded and click **Submit**.

Before uploading the software package, ensure that the corresponding product model exists in the Developer Center.

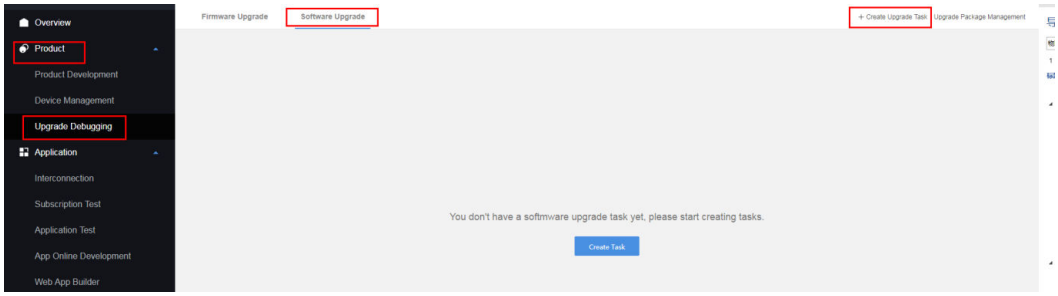
NOTE

Developer Center can obtain the product model information from the JSON file in the software package. Therefore, you do not need to select a product and associate it with the software package.

----End

Creating a Software Upgrade Task

Step 1 Choose **Product** > **Upgrade Debugging** > **Software Upgrade**, and click **Create Upgrade Task**.



NOTE

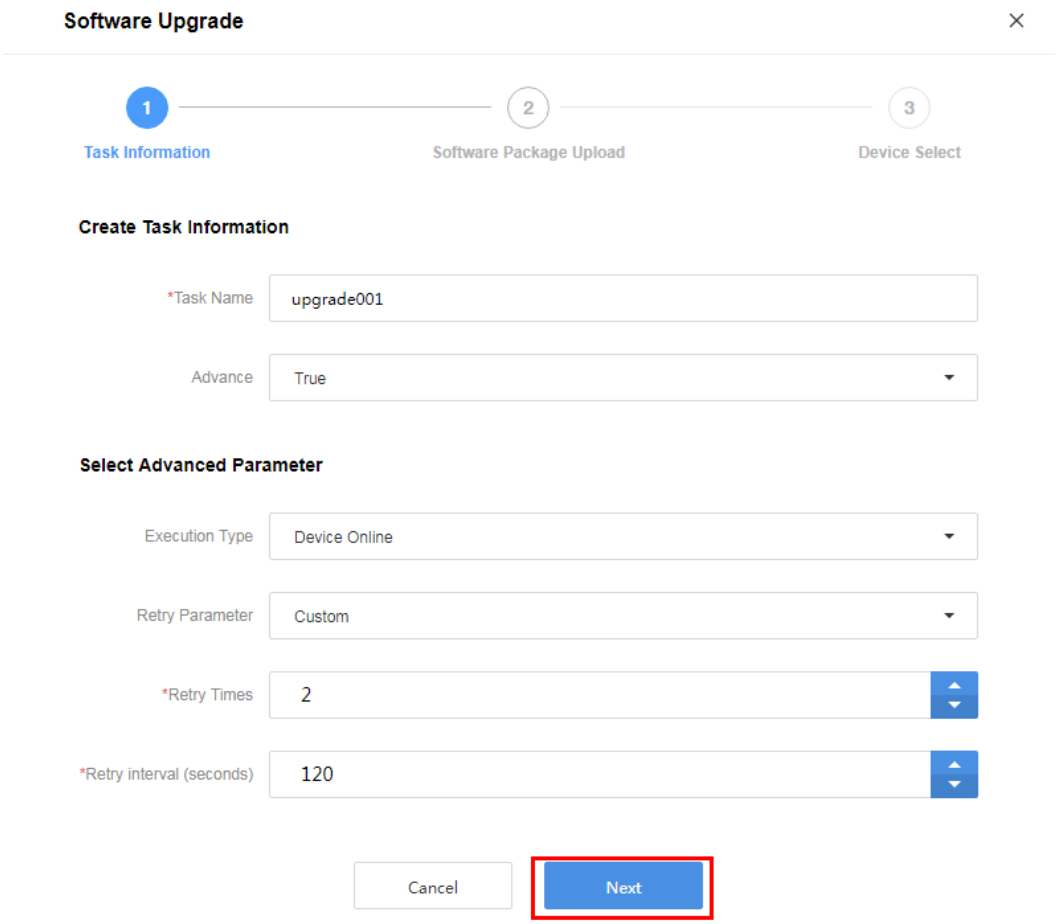
If no upgrade task is available, click **Create Task** to create a software upgrade task.

Step 2 Create the upgrade task according to the wizard.

1. Enter basic information and click **Next**.

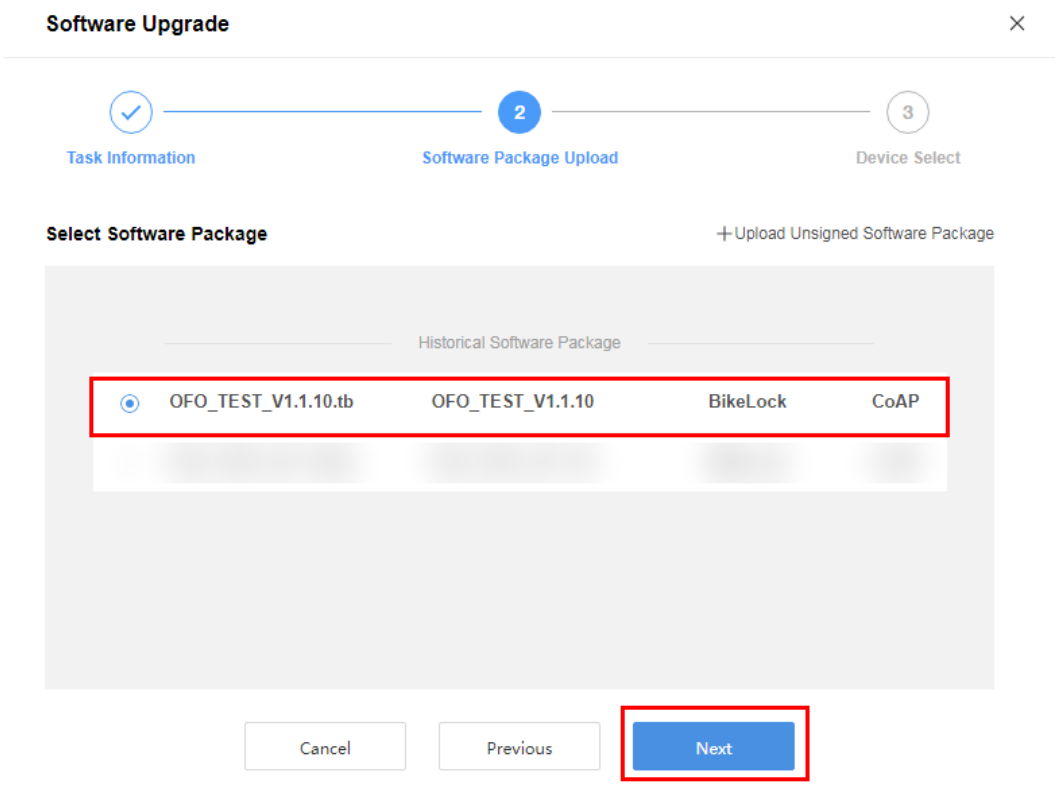
   If you need to configure the parameters **Execution Type** and **Retry Parameter**, set the parameter **Advance** to **True**.

- **Execution Type** indicates the time when the Developer Center delivers an upgrade task to the device. The value options include **Now**, **Custom**, and **Device Online**.
- **Retry Parameter** indicates whether to execute failed task again.

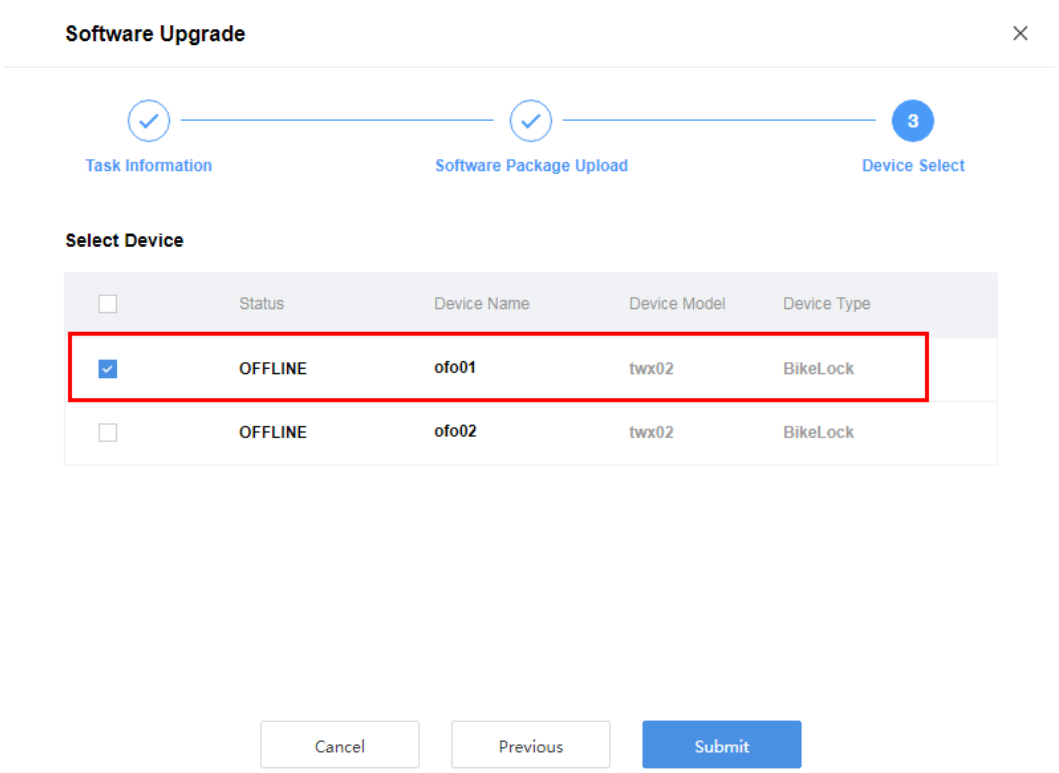


2. Select the software package to be upgraded and click **Next**.

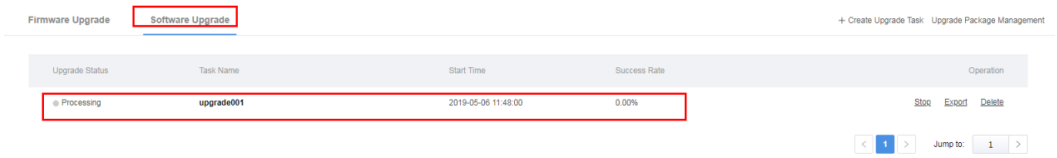
   If no required software package exists in the **Historical Software Package** list, click **Unsigned Software Package** to upload an unsigned software package. Ensure that the software package to be uploaded can be used for software upgrade. Before uploading the software package, ensure that the corresponding product model exists in the Developer Center.



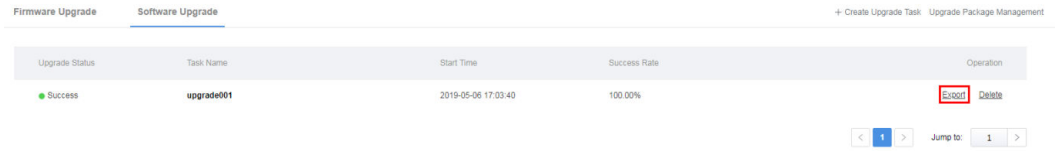
3. In the **Device Select** tab page, you can select one or multiple devices for upgrade and click **Submit**.



Step 3 After a software upgrade task is created, you can manage it on the **Software Upgrade** tab page. Click a task to view the **Basic Information** and **Upgrade Detail**.



Step 4 After the task is complete, click **Export** to export the task details file.



----End

2.6 Application

2.6.1 Introduction

The application module of the Developer Center consists of three functional modules: **Interconnection**, **Subscription Test**, and **Application Test**.

- **Interconnection** provides access information about the IoT platform for developers, such as addresses, ports, and protocols. For details, see **Interconnection**.
- **Subscription Test** simulates the subscription interface calling function to check the validity and connectivity of the push addresses. For details, see **Subscription Test**.
- **Application Test** enables developers to test the NA server by using a physical device or a virtual device. For details, see **Application Test**.

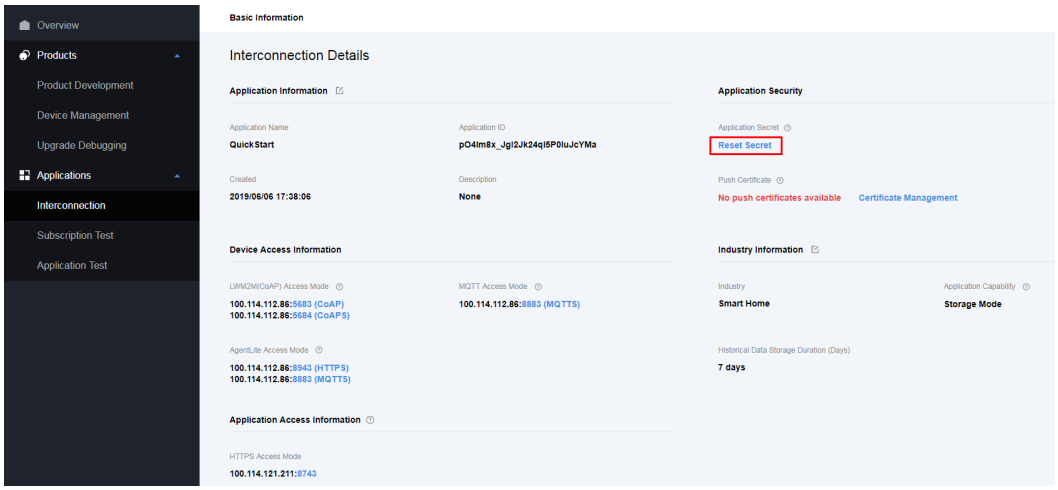
2.6.2 Interconnection

Overview

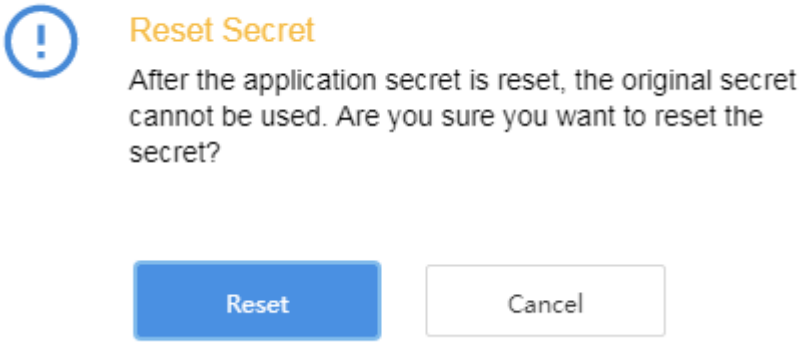
Interconnection provides an entry for editing the application security, access information, and industry information.

Resetting the Application Secret

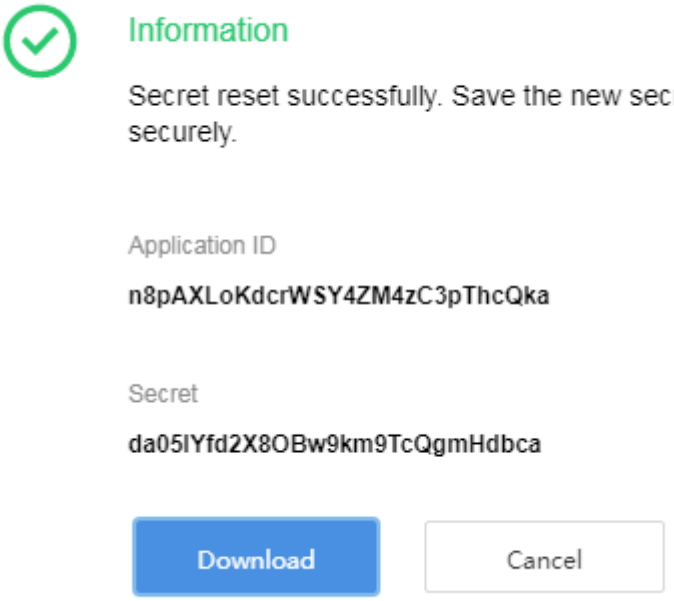
Step 1 Choose **Applications** > **Interconnection**. Click **Reset Secret** under **Application Secret**.



Step 2 In the **Reset Secret** dialog box displayed, click **Reset**.



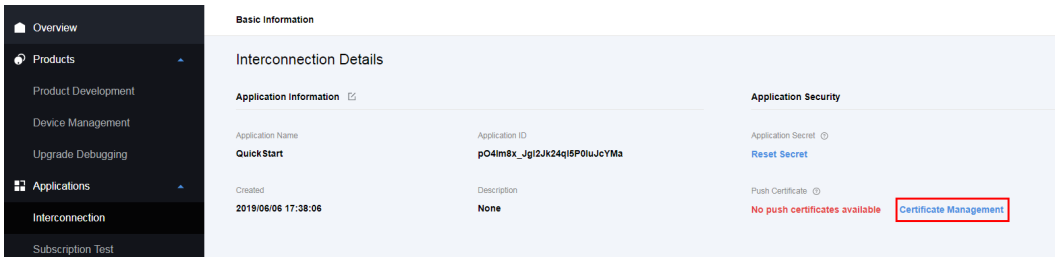
Step 3 The IoT platform returns new **Application ID** and **Secret**. Keep the new application ID and secret securely.



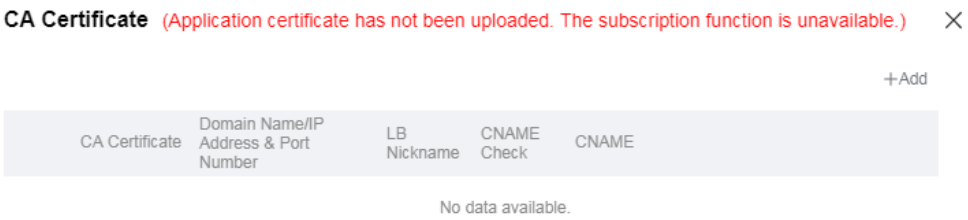
----End

Managing CA Certificates

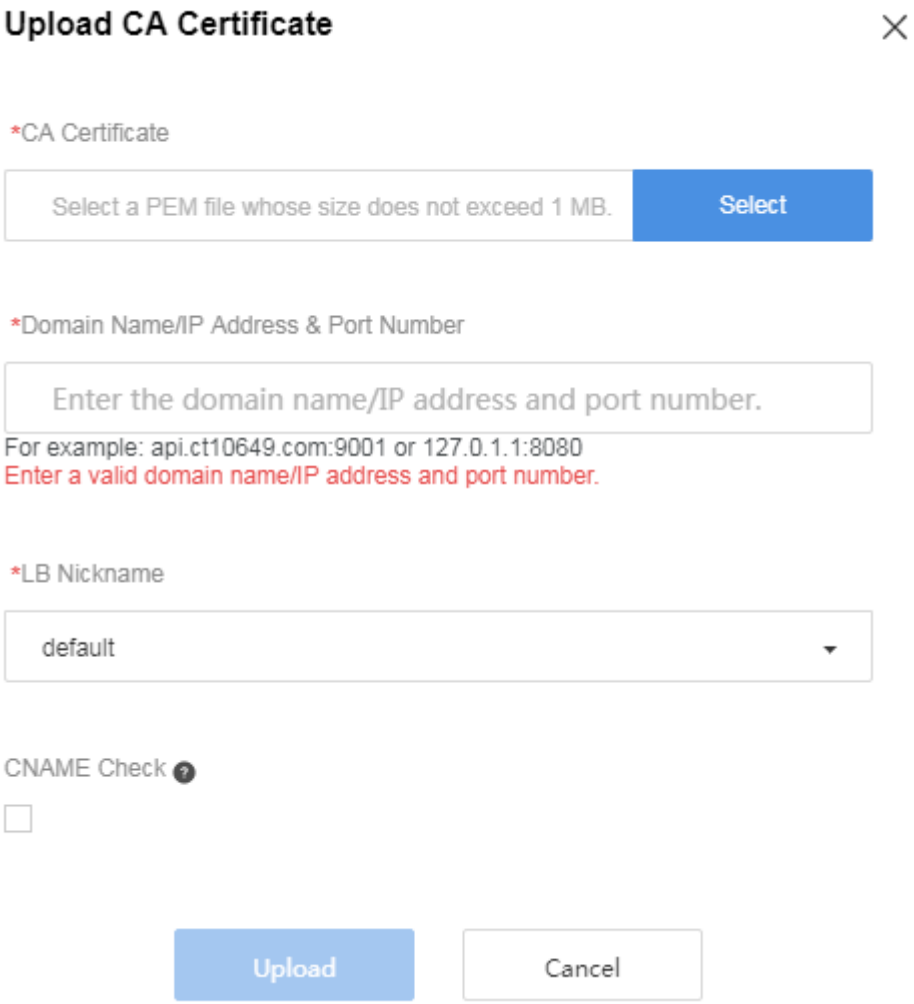
Step 1 Choose **Applications** > **Interconnection**. In the **Push Certificate** area, click **Certificate Management**.



Step 2 The **CA Certificate** dialog box is displayed. Check whether the CA certificate has been uploaded. If not, click **Add**.



Step 3 In the displayed **Upload CA Certificate** dialog box, select the certificate file, set parameters, and click **Upload**.



----End

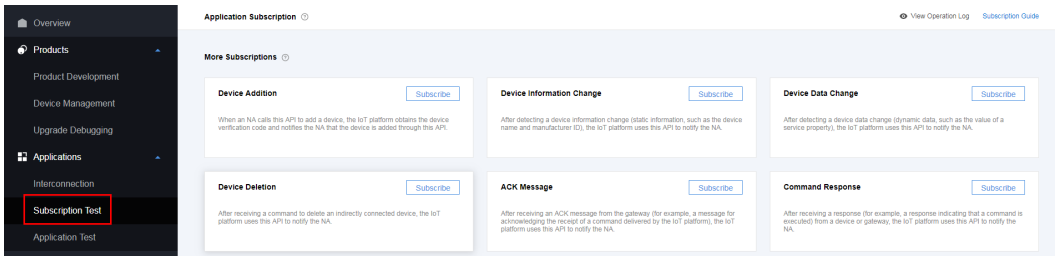
2.6.3 Subscription Test

Overview

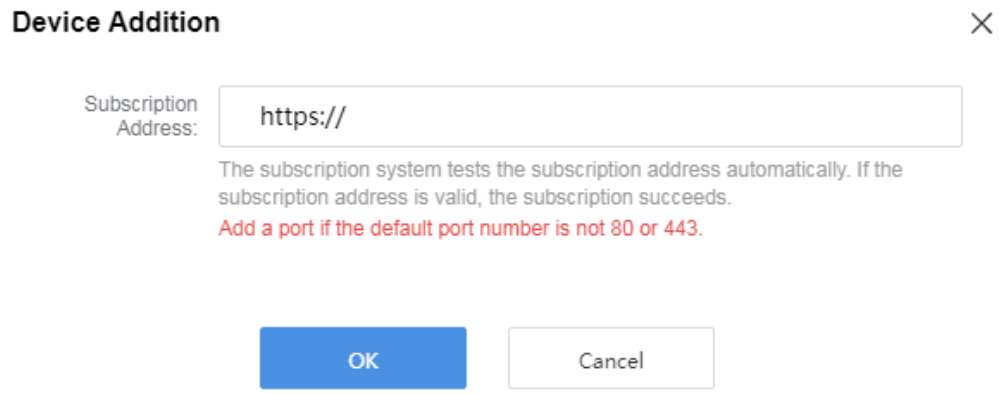
Subscription Test simulates the subscription API calling function to check the validity and connectivity of push addresses.

Creating a Subscription

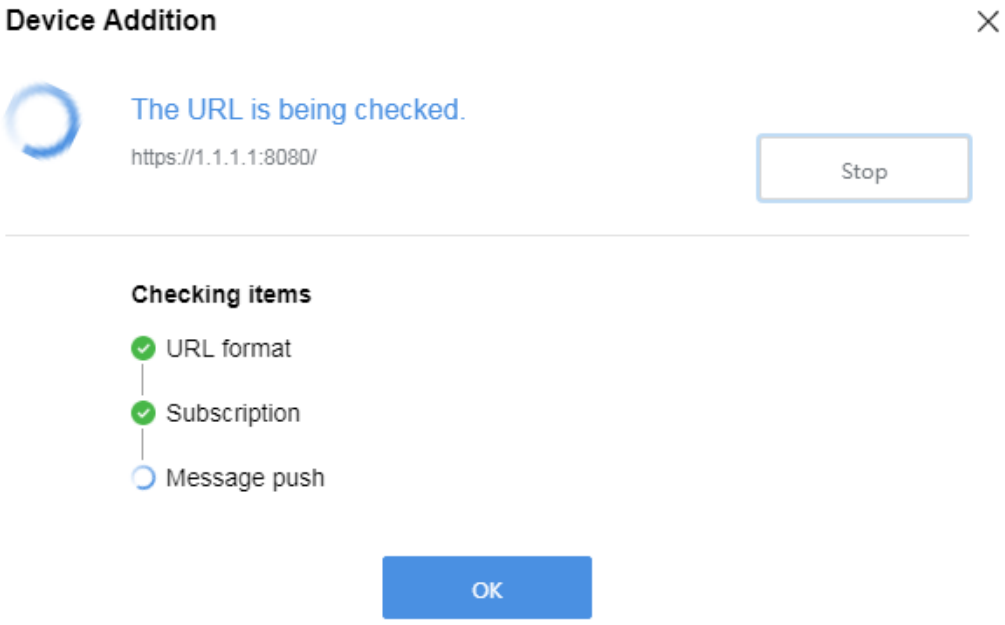
Step 1 Choose **Applications** > **Subscription Test**. Select a notification type as required, and click **Subscribe**. The following uses **Device Addition** notification as an example.



Step 2 In the displayed dialog box, enter the subscription address and click **OK**.



Step 3 The IoT platform checks the validity and connectivity of the subscription address.



----End

2.6.4 Application Test

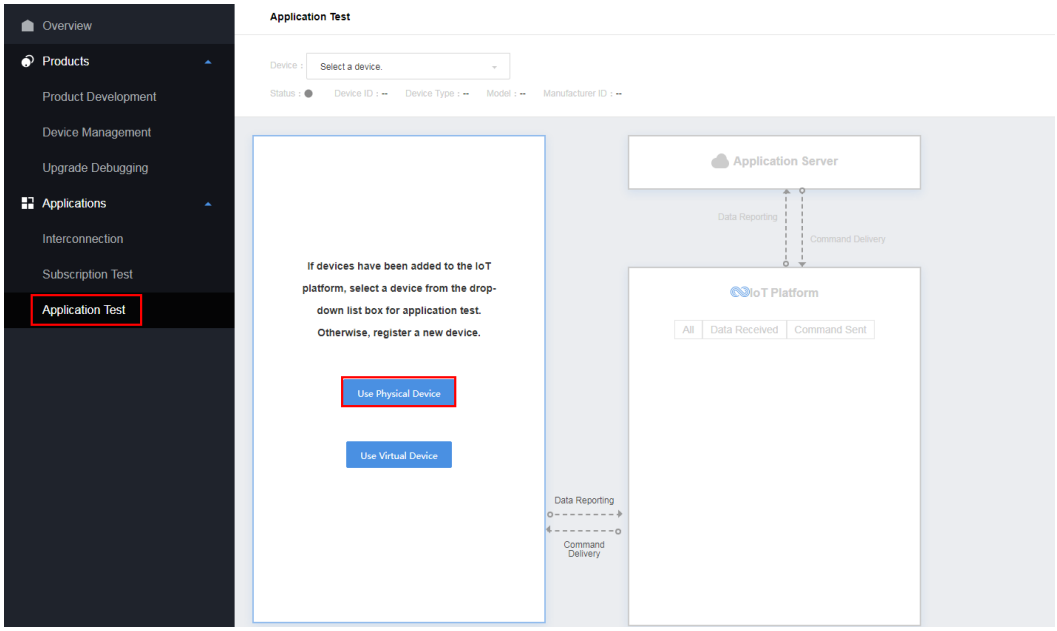
Overview

Application Test enables developers to test the NA by using a physical device or a virtual device.

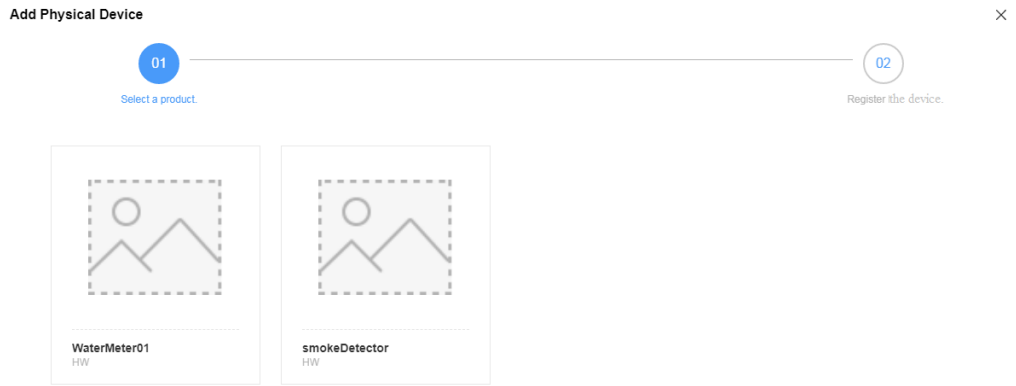
- When the device development is complete, developers can use a physical device to test the NA.
- When the device development is not complete, developers can use a virtual device to test the NA.

Using a Physical Device for Testing

Step 1 Choose **Applications** > **Application Test**. Click **Use Physical Device**.



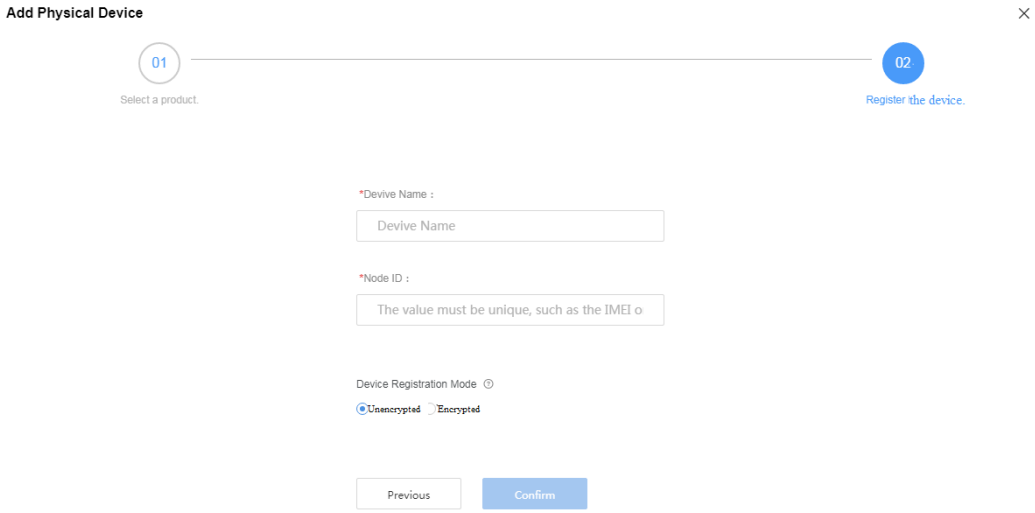
Step 2 In the **Add Physical Device** dialog box displayed, select a device.



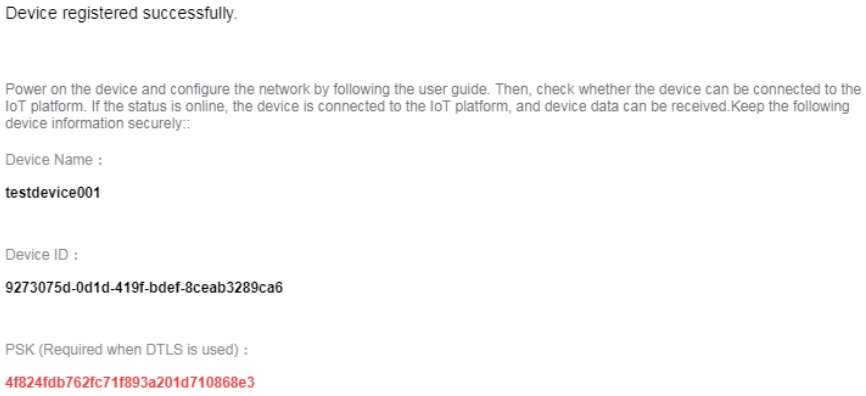
Step 3 Configure device information and click **Confirm**.

- **Device Name** can contain only letters, digits, and underscores (_) and must be unique in the product.

- **Node ID** must be set to a unique value, such as the IMEI or MAC address of the device.
- Select **Unencrypted** or **Encrypted** based on site requirements. If this parameter is set to **Unencrypted**, the device uses the CoAP/UDP protocol to connect to the IoT platform. If this parameter is set to **Encrypted**, the device uses the CoAPS/DTLS protocol to connect to the IoT platform.



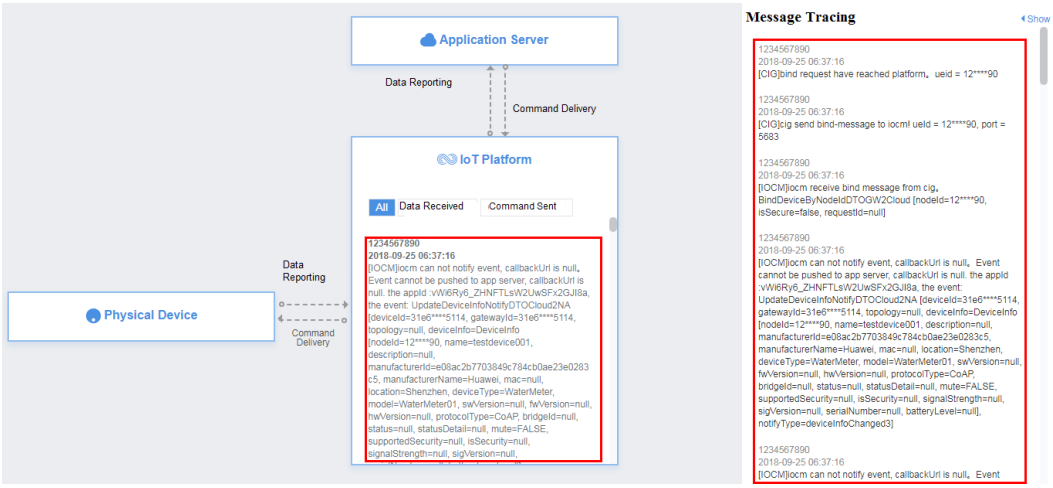
After the device is added, **Device ID** and **PSK** are returned. Keep the PSK securely as it is required when the device uses DTLS to connect to the IoT platform.



Step 4 After a physical device is added, you can test the data reporting and command delivery of the NA.

Connect the device to the IoT platform and report data. View the data reporting result in **IoT Platform** and **Application Simulator** and processing logs of the IoT platform in **Message Tracing**.

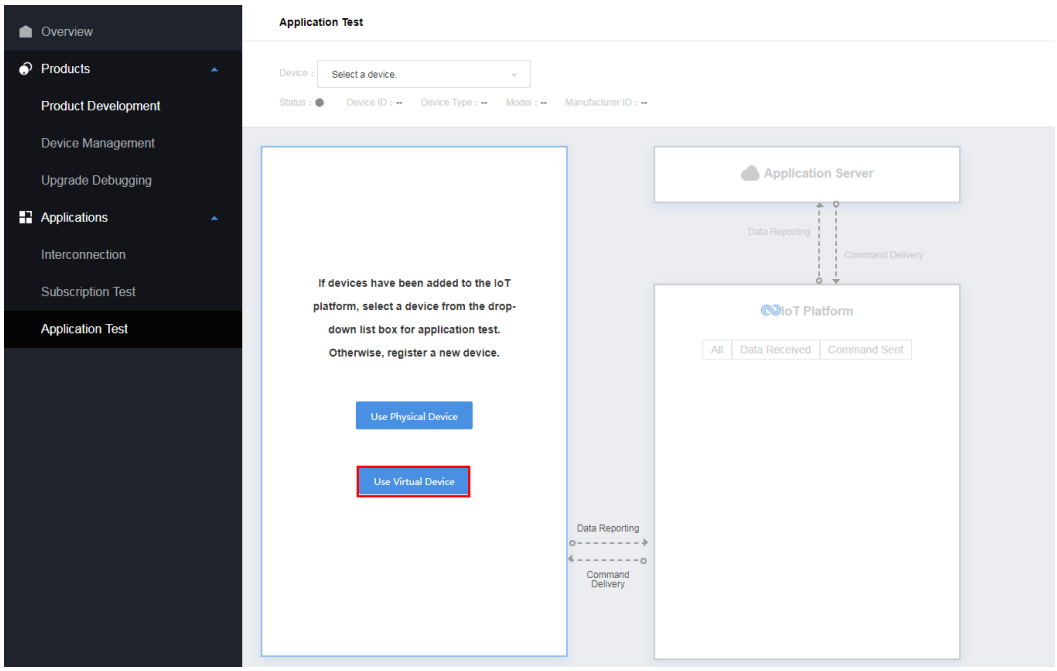
Connect the NA to the IoT platform and deliver a command. View the command delivery result in **IoT Platform** and on the device and processing logs of the IoT platform in **Message Tracing**.



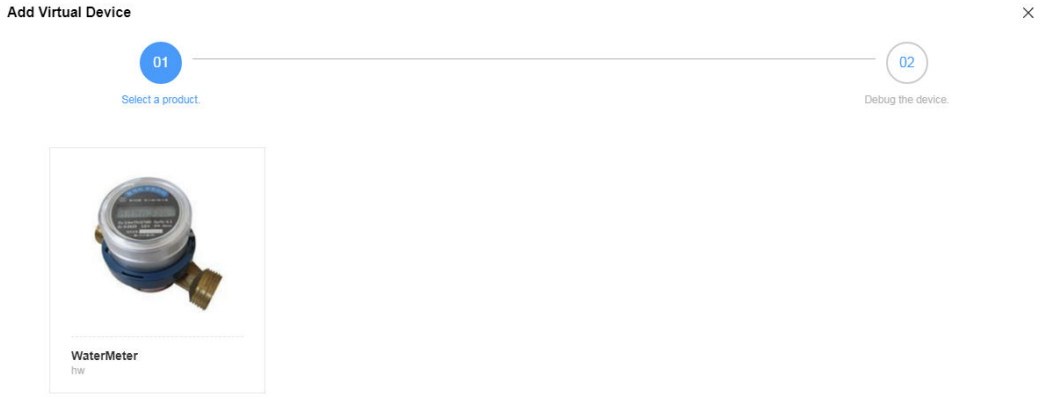
----End

Using a Virtual Device for Testing

Step 1 Choose **Applications** > **Application Test**. Click **Use Virtual Device**.



Step 2 In the **Add Virtual Device** dialog box displayed, select a device.



Step 3 After a virtual device is added, you can test the data reporting and command delivery of the NA.

In **Device Simulator**, enter a hexadecimal code stream or JSON data (for example, enter a hexadecimal code stream) and click **Send**. Then, view the data reporting result in **IoT Platform** and **Application Simulator** and processing logs of the IoT platform in **Message Tracing**.

After the NA delivers a command, view the received command (for example, a hexadecimal code stream) in **Device Simulator** and view processing logs of the IoT platform in **Message Tracing**.

----End

