Alibaba Cloud
物#网平台

クイックスタート

Document Version20191220

目次

1 IoT Platform の使用

1.1 プロダクトとデバイスの作成

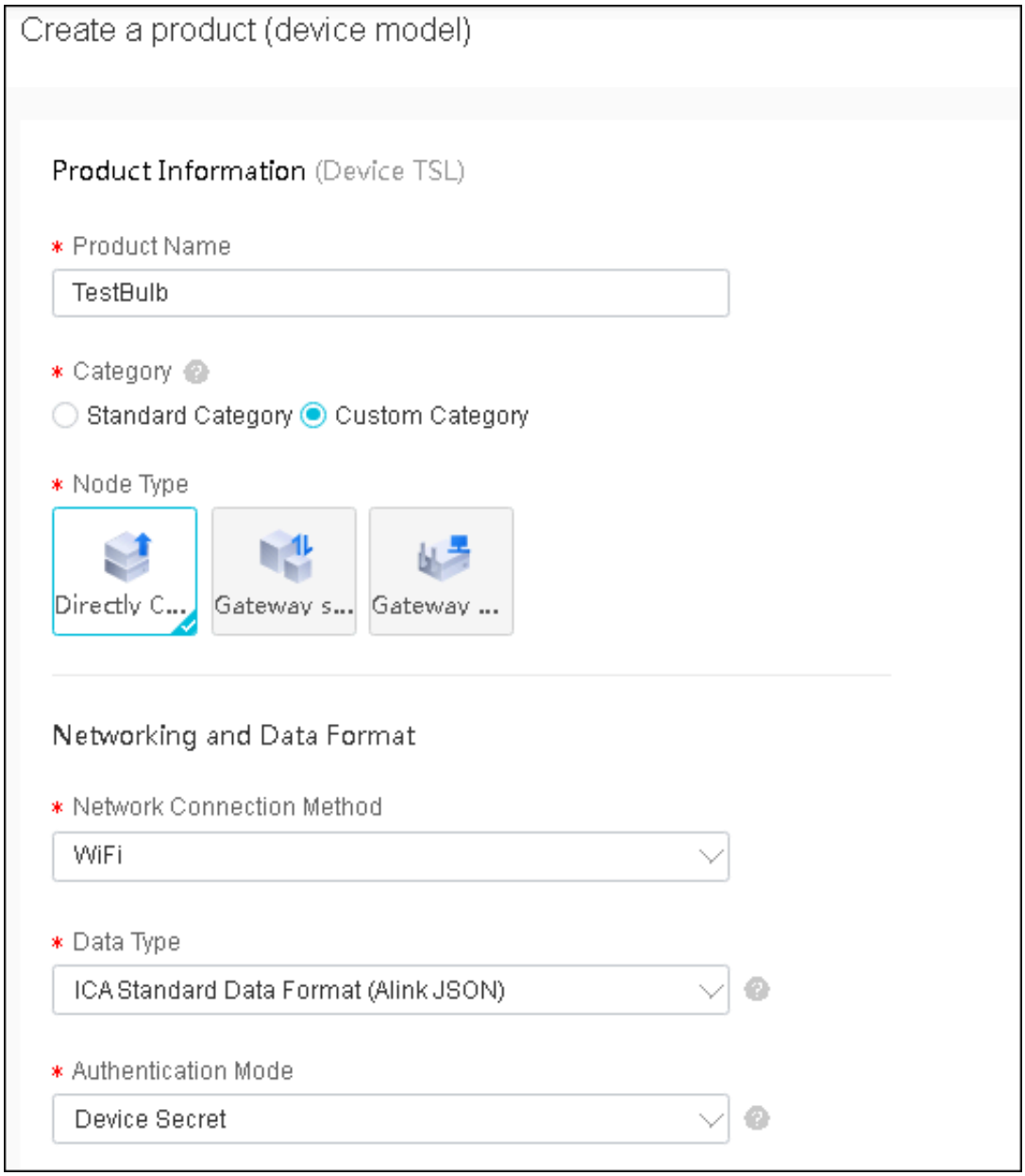
IoT Platform を使用する最初のステップは、プロダクトとデバイスを作成することです。 プロダクトとは、一般的に同じ機能を持つデバイスの集合です。 対応するプロダクトを管理することで、デバイスを一括管理できます。

1. *IoT Platform コンソール*にログインします。

2. プロダクトを作成します。

   a) 左側のナビゲーションウィンドウで、[デバイス] > [プロダクト] をクリックします。 [プロダクト] ページで、[プロダクトの作成] をクリックします。

   b) 必須情報をすべて入力し、[OK] をクリックします。



パラメーターの説明は以下のとおりです。

パラメーター	説明
プロダクト名	この例では、プロダクトは **[TestBulb]** という名前です。 プロダクト名は、アカウント内で一意にする必要があります。 プロダクト名の長さは **4**～**30** 文字で、漢字、英数字、およびアンダースコアを含めることができます。 漢字は **2** 文字としてカウントされます。
カテゴリ	この例では、プロダクトカテゴリは **[カスタムカテゴリ]** で、プロダクトの機能がユーザー定義であることを示しています。
ノードタイプ	この例では、ノードタイプは **[デバイス]** です。 ・ **[デバイス]:** このプロダクトのデバイスにサブデバイスをマウントできないことを示します。 このタイプのデバイスは、直接またはゲートウェイデバイスのサブデバイスとして **IoT Platform** に接続できます。 ・ **[ゲートウェイ]:** このプロダクトのデバイスは、**IoT Platform** に直接接続し、サブデバイスと共にマウントできることを示します。 ゲートウェイは、サブデバイスの管理、サブデバイスとのトポロジー関係の維持のほか、トポロジー関係を **IoT Platform** へ同期させることができます。
ゲートウェイへの接続 注： このパラメーターは、ノードタイプが **[デバイス]** の場合に表示されます。	このプロダクトのデバイスを、サブデバイスとしてゲートウェイに接続できるかどうかを示します。 ・ **[はい]:** このプロダクトのデバイスは、ゲートウェイに接続することができます。 ・ **[いいえ]:** このプロダクトのデバイスは、ゲートウェイに接続することができません。
ネットワーク接続方法 注： このパラメーターは、**[ゲートウェイへの接続]** に **[いいえ]** を選択した場合に表示されます。	デバイスのネットワーク接続方法を選択します。 この例では **[WiFi]** が選択されています。

パラメーター	説明
データ型	デバイスと **IoT Platform** 間でデータをやりとりする形式を選択します。 この例では **[ICA** 標準データ形式 **(Alink JSON)]** が選択されています。 **ICA** 標準データ形式 **(Alink JSON):** デバイスと **IoT Platform** との通信用に **IoT Platform** で定義された標準データ形式。
プロダクトの説明	プロダクト情報を指定します。 **100** 文字まで入力できます。

プロダクトが正常に作成されると、プロダクトリストに表示されます。

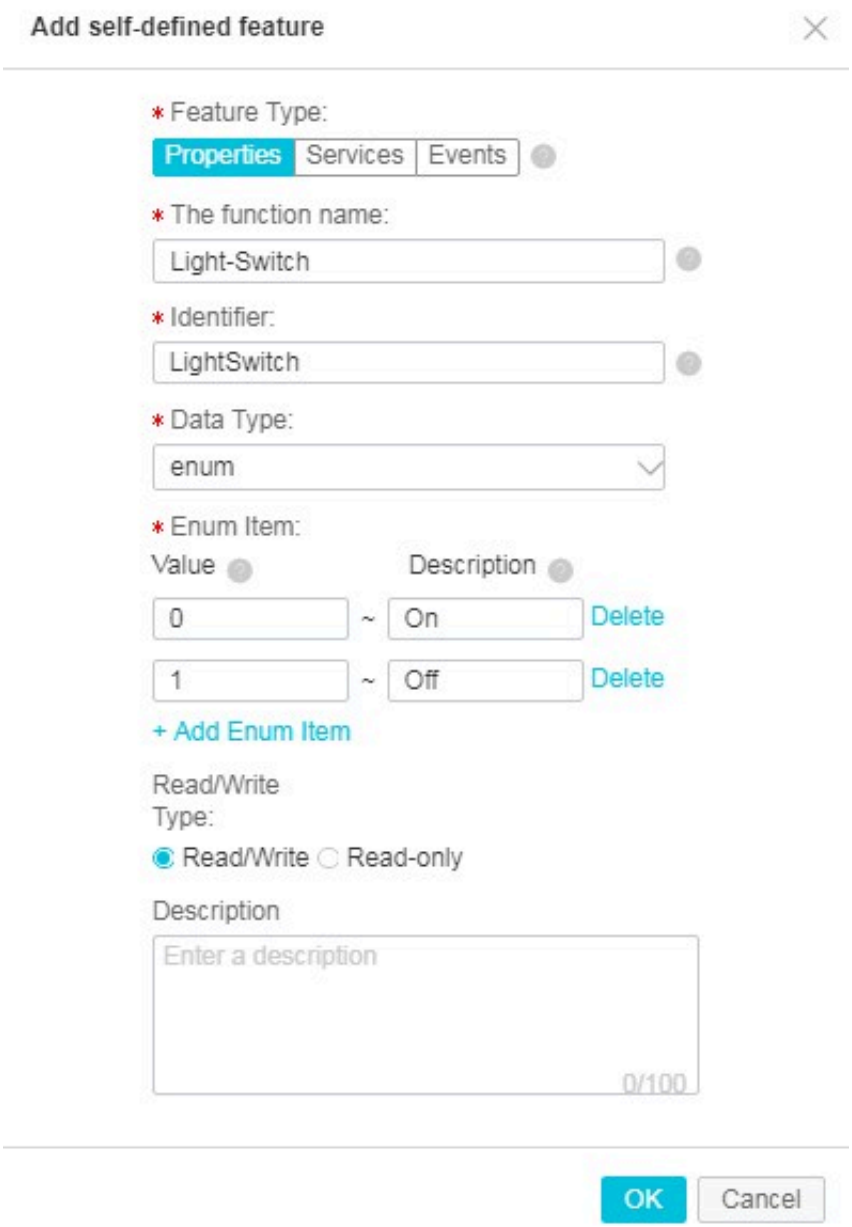
3. プロダクトの機能を定義します。

   a) プロダクトリストでプロダクトを探し、[表示] をクリックします。

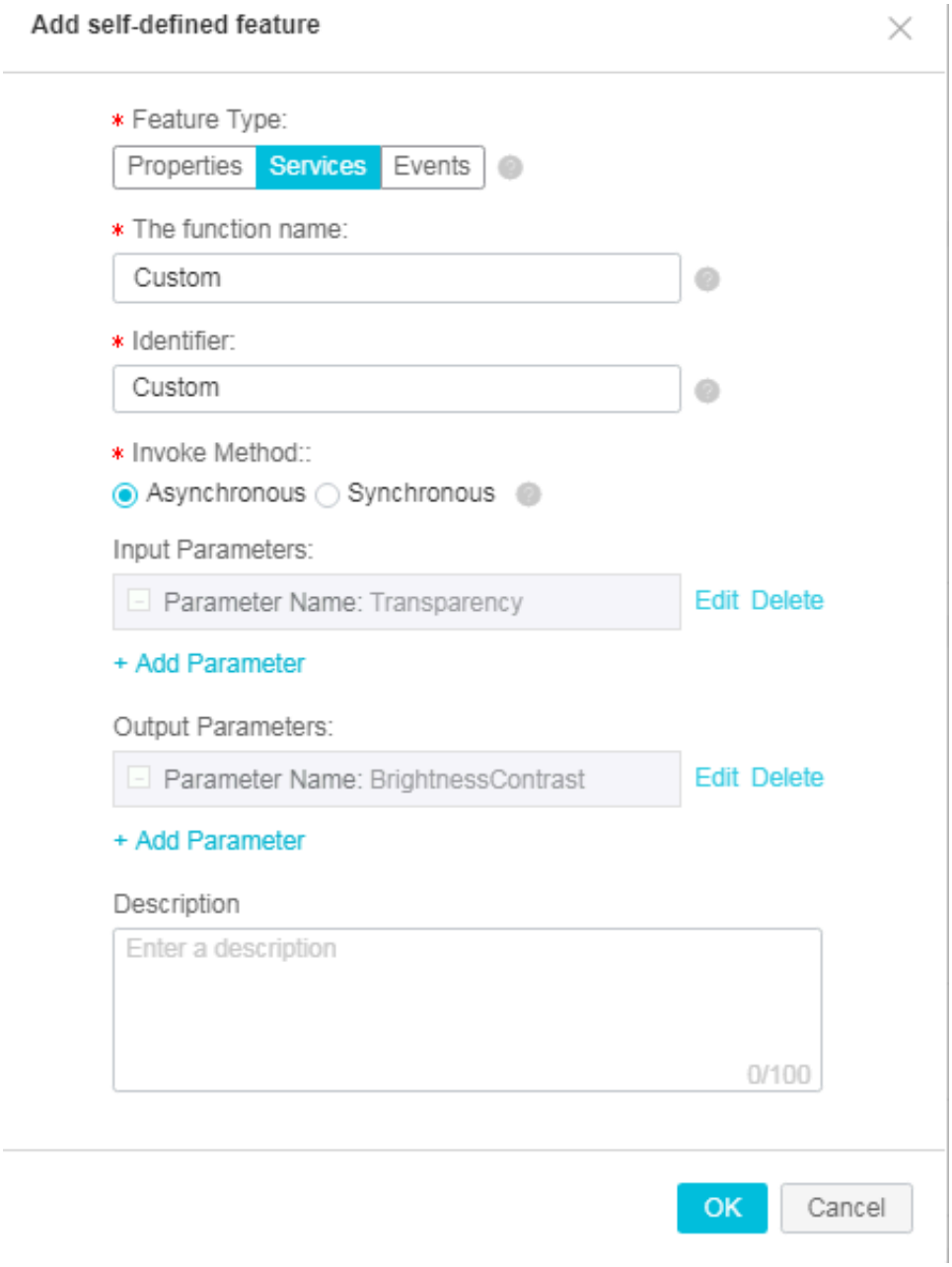
   b) プロダクトの詳細ページで、[機能の定義] をクリックします。

   c) [カスタム機能] で [機能の追加] をクリックします。

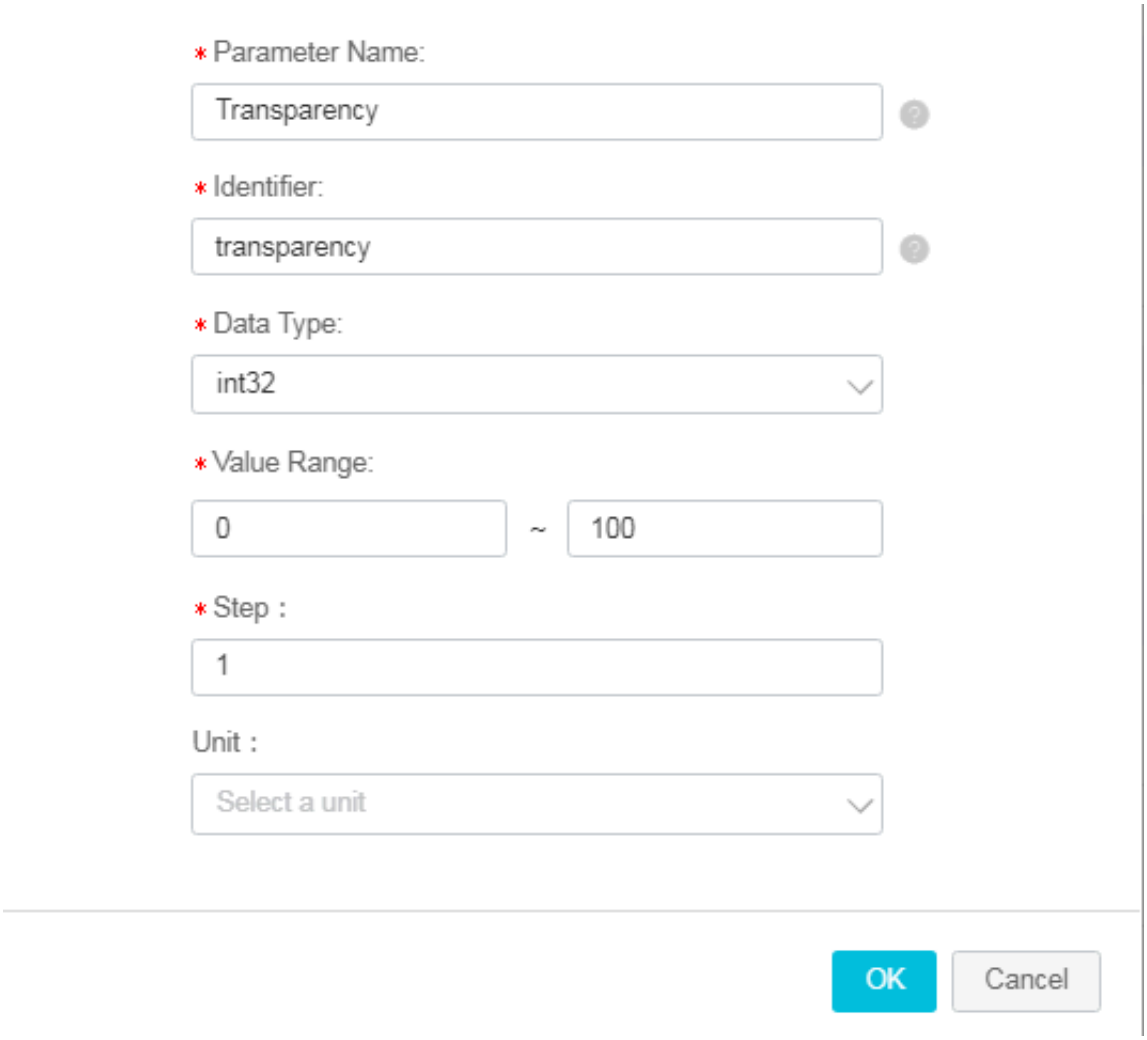
   d) プロパティを定義します。 この例では、**light switch** (ライトのスイッチ) プロパティが定義されています。 **0** はライトをオンにすることを示し、**1** はライトをオフにすることを示します。



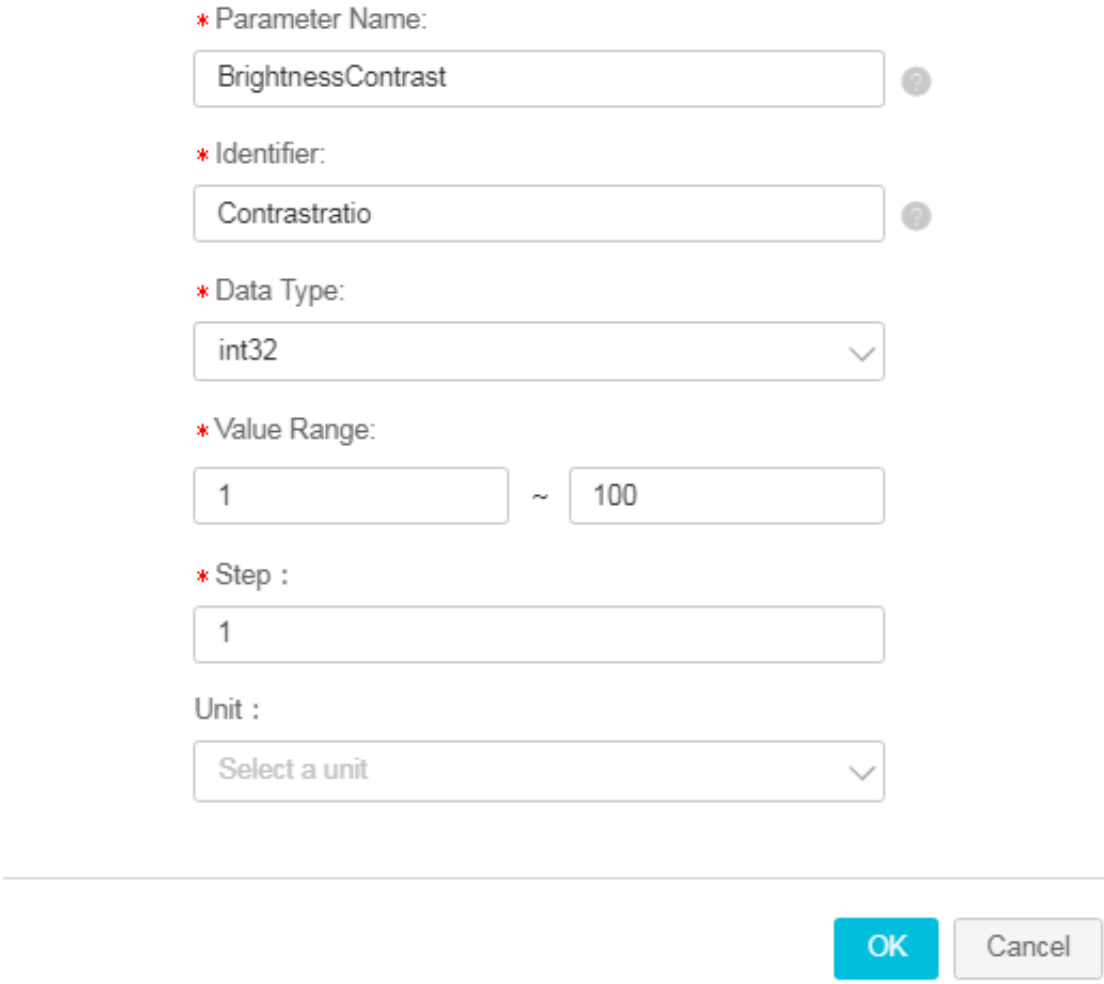
   e) サービスを定義します。 たとえば、電球の明るさを調整するための入力パラメーターを追加したり、電球と室内環境の明るさのコントラストを報告するために、電球の出力パラメーターを追加したりできます。



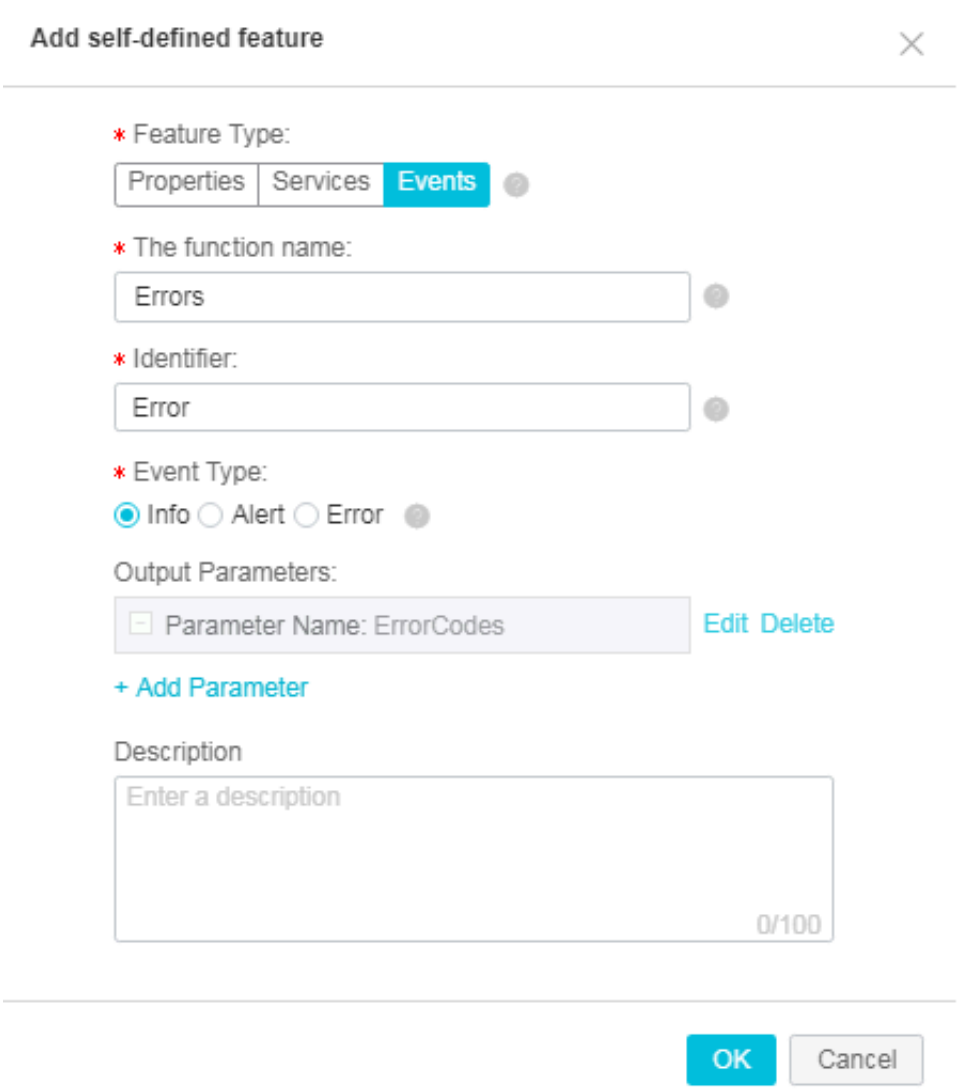
次の図は、入力パラメーターの例を示しています。



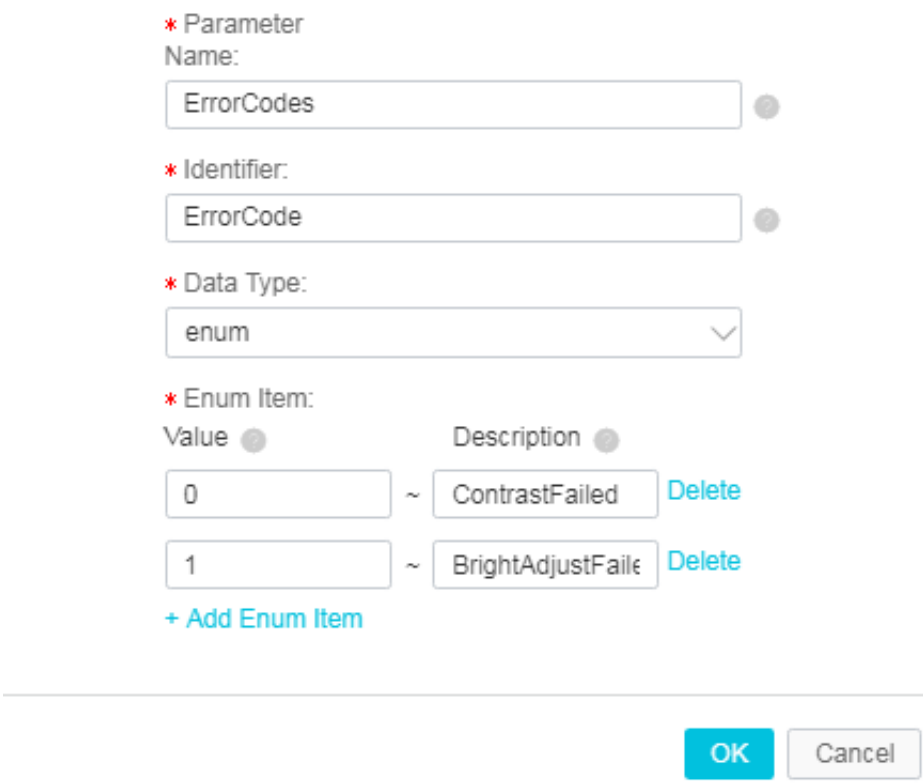
次の図は、出力パラメーターの例を示しています。



f) イベントを定義します。 デバイスがエラーを報告するイベントを定義できます。



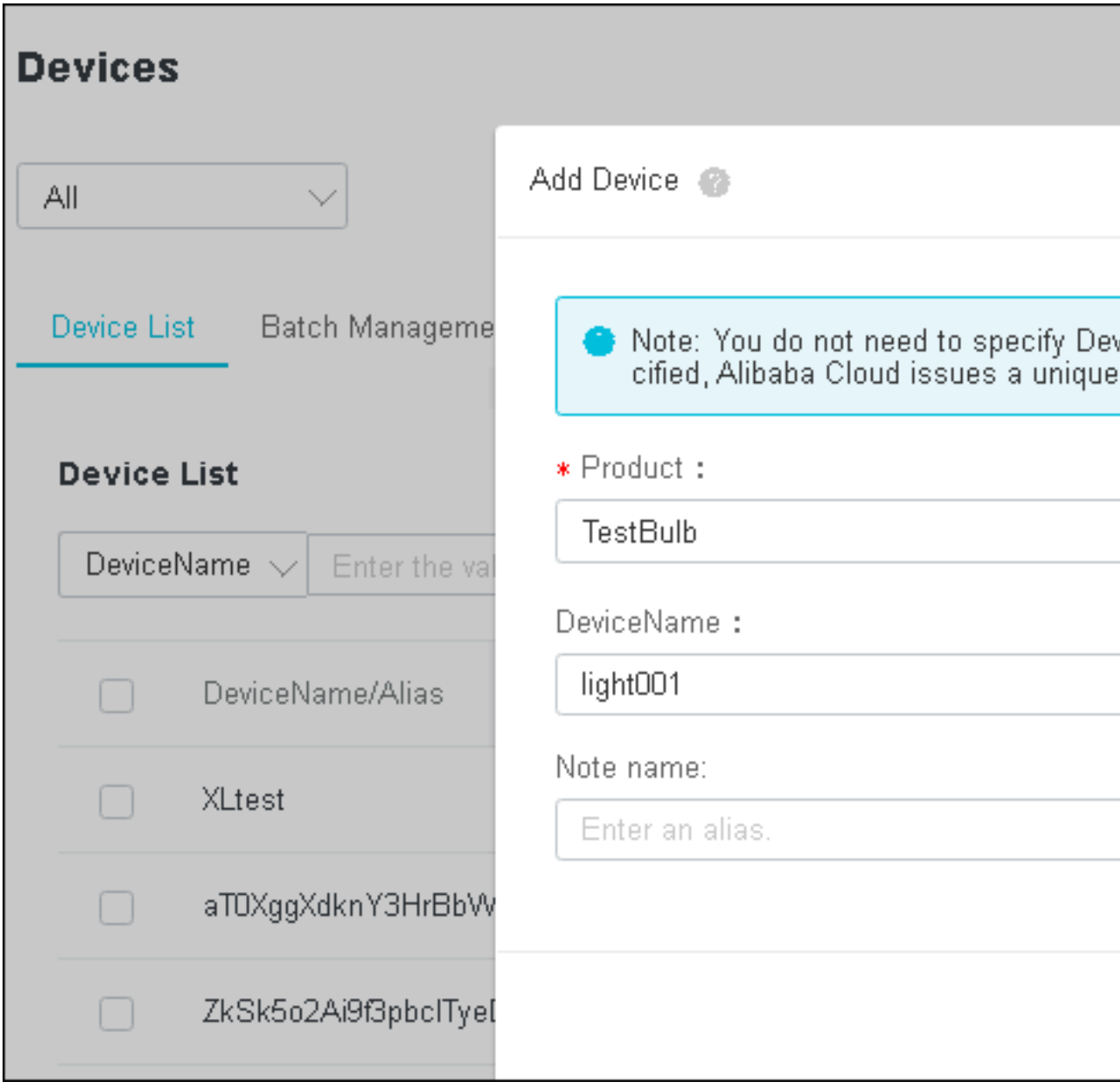
次の図は、出力パラメーターの例を示しています。


* Parameter Name:
ErrorCodes
* Identifier:
ErrorCode
* Data Type:
enum
* Enum Item:
Value
Description
0
~
ContrastFailed
Delete
1
~
BrightAdjustFaile
Delete
+ Add Enum Item
OK
Cancel

4. デバイスを作成します。

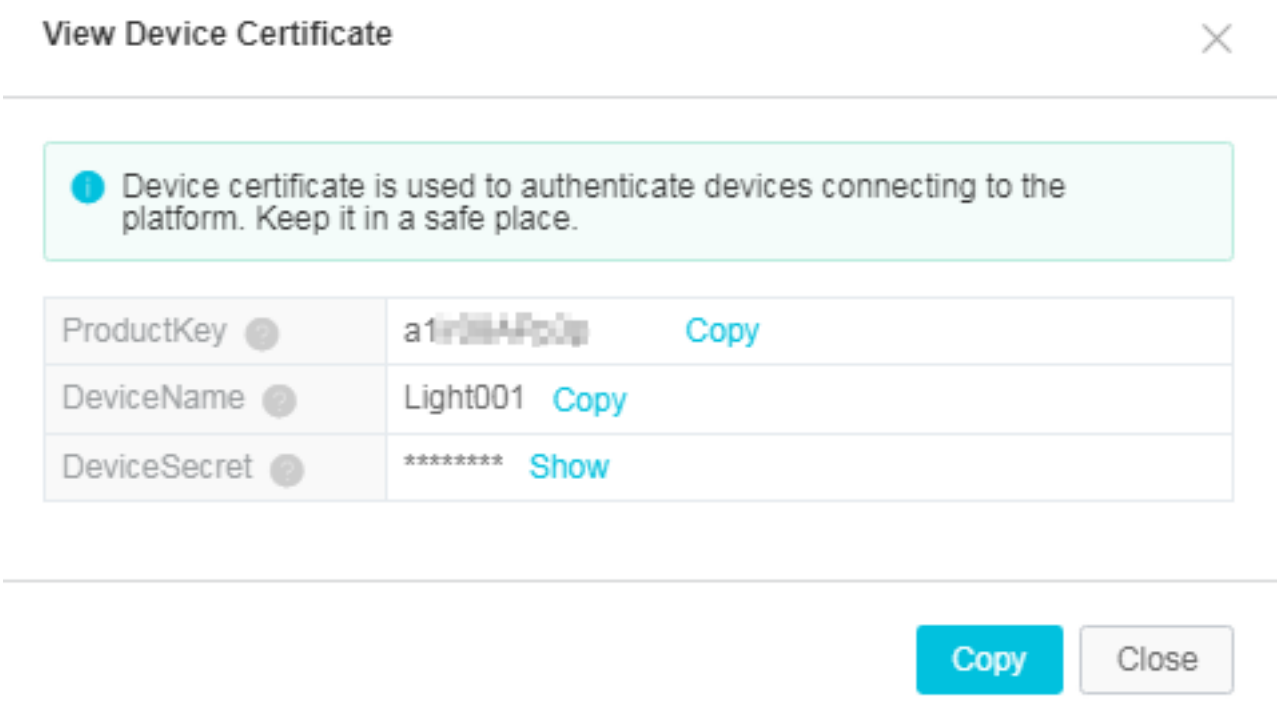
   a) 左側のナビゲーションウインドウで、[デバイス] > [デバイス] をクリックします。

   b) デバイス管理ページで [デバイスの追加] をクリックします。 作成するデバイスが属するプロダクトを選択して、デバイスの名前 (**DeviceName**) を入力します。 **[OK]** をクリックします。



   c) デバイス証明書情報を保存します。 証明書情報には、**ProductKey**、**DeviceName**、**DeviceSecret** が含まれています。 この情報は、デバ

イスが IoT Platform に接続する際のデバイス認証に使用される証明書のため、人に知られないようにしてください。



1.2 デバイスと IoT Platform 間の接続の確立

Alibaba Cloud IoT Platform は、デバイスを IoT Platform に接続するデバイス SDK を提供しています。 ここでは、IoT Platform のサンプルプログラムを使用して、SDK を使用してデバイスを IoT Platform に接続する方法を説明します。

- この例で使用されている SDK は、Linux システム用の C SDK です。 この SDK は、Ubuntu16.04 (64-bit) で開発することを推奨します。
- SDK 開発に使用されたソフトウェア: `make-4.1`、`git-2.7.4`、`gcc-5.4.0`、`gcov-5.4.0`、`lcov-1.12`、`bash-4.3.48`、`tar-1.28`、および `mingw-5.3.1`。以下のコマンドを使用して、このソフトウェアをインストールします。

  ```
  apt-get install -y build-essential make git gcc
  ```

1. Linux VM インスタンスにログインします。
2. C SDK 2.3.0 をダウンロードします。

   ```
   wget https://github.com/aliyun/iotkit-embedded/archive/v2.3.0.zip?spm=a2c4g. 11186623.2.13.1f41492b5WHpzV&file; v2.3.0.zip
   ```

3. パッケージからファイルを展開するには、unzip コマンドを使用します。

4. デモプログラムを開きます。

   ```
   vi iotkit-embedded-2.3.0/examples/linkkit/linkkit_example_solo.c
   ```

5. デモの **ProductKey**、**DeviceName**、**DeviceSecret** の値を自分のデバイス証明書情報に変更してから、ファイルを保存します。

   次の例をご参照ください。

   ```
   // for demo only
   #define PRODUCT_KEY       "a1I1nn8vPf4"
   #define DEVICE_NAME       "Light00"
   #define DEVICE_SECRET     "n27gKXTxrUx*********QZEmoUX8TceM"
   ```

6. 最上位ディレクトリで、**make** コマンドを使用してサンプルプログラムをコンパイルします。

   ```
   $ make distclean
   $ make
   ```

7. サンプルプログラムを実行してデバイスを **IoT Platform** に接続します。 **IoT Platform** のコンソールで、デバイスのステータスがオンラインになり、デバイスが **IoT Platform** に正常に接続されたことを示します。

   デバイスが **IoT Platform** に接続されると、**IoT Platform** にメッセージが自動的に報告されます。 メッセージの内容は、デバイスログで参照できます。

1.3 デバイスによる IoT Platform からのコマンド受信

クラウドのアプリケーションを使用して SetDeviceProperty インターフェイスを呼び出し、プロパティ設定のためのコマンドをデバイスに送信することができます。 ここでは、 **IoT Platform** からのコマンドを受け取れるようにデバイス **SDK** を設定する方法を紹介します。

1. **SDK** の依存関係を **Maven** プロジェクトにインポートします。

   次の例は、 **IoT Platform** の **Java SDK** 依存関係を **Maven** プロジェクトにインポートする方法を示しています。

   ```
   <! -- https://mvnrepository.com/artifact/com.aliyun/aliyun-java-sdk-
   iot -->
   <dependency>
       <groupId>MySQL</groupId>
       <artifactId>aliyun-java-sdk-core</artifactId>
       <version>5.1.6</version>
   </dependency>
   ```

   SDK のコアモジュールをインポートします。

   ```
   <dependency>
       <groupId>MySQL</groupId>
       <artifactId>log4j-core</artifactId>
       <version>5.1.6</version>
   ```

```
</dependency>
```

2. SDK を初期化します。

   エンドポイントのリージョン ID は、デバイスのリージョン ID と同じである必要があります。 次の例では、リージョン ID は cn-shanghai です。

```
String accessKey = "<your accessKey>";
String accessSecret = "&lt; your accessSecret&gt;";
DefaultProfile.addEndpoint ( "cn-shanghai"、 "cn-shanghai"、 "Iot"、
 "iot.cn-shanghai.aliyuncs.com") ;
IClientProfile profile = DefaultProfile.getProfile ( "cn-shanghai"、
accessKey、accessSecret) ;
IAcsClient client = new DefaultAcsClient(profile);
```

3. プロパティ設定リクエストをデバイスに送信するには、 SetDeviceProperty オペレーションを呼び出します。 次の例では、プロパティ LightSwitch の値が 1 に設定されています。

   例:

```
SetDevicePropertyRequest request = new SetDevicePropertyRequest () ;
request.setProductKey ( "a1I1xxxxPf4") ;
request.setDeviceName ( "Light001") ;
JSONObject itemJson = new JSONObject();
itemJson.put ( "LightSwitch"、1) ;
request.setItems (itemJson.toString () ) ;

try {
    SetDevicePropertyResponse response = client.getAcsResponse(
request);
    System.out.println(response.getRequestId() + ", success: " +
response.getSuccess());
} catch (ClientException e) {
    e.printStackTrace();
}
```

   注:

   SetDeviceProperty オペレーションを呼び出す方法の詳細については、「*SetDeviceProperty*」をご参照ください。

4. デバイスがリクエストを受信した場合、ログ出力は次のようになります。

```
[inf] iotx_mc_handle_recv_PUBLISH(1617): Downstream Topic: '/sys/
a1I1nn8vPf4/Light001/thing/service/property/set'
[inf] iotx_mc_handle_recv_PUBLISH(1618): Downstream Payload:

< {
< "methos": "thing.service.property.set"、
<  "id": "200864995"、
<  "params": {
< "LightSwitch": 1
<     },
<     "version": "1.0.0"
< }
```

```
[dbg] iotx_mc_handle_recv_PUBLISH(1623):         Packet Ident :
00000000
[dbg] iotx_mc_handle_recv_PUBLISH(1624):         Topic Length : 52
[dbg] iotx_mc_handle_recv_PUBLISH(1628):           Topic Name : /sys
/a1I1nn8vPf4/Light001/thing/service/property/set
[dbg] iotx_mc_handle_recv_PUBLISH(1631):     Payload Len/Room : 101
 / 109
[dbg] iotx_mc_handle_recv_PUBLISH(1632):       Receive Buflen : 166
[dbg] iotx_mc_handle_recv_PUBLISH(1643): delivering msg ...
[dbg] iotx_mc_deliver_message(1344): topic be matched
[inf] dm_msg_proc_thing_service_property_set(134): thing/service/
property/set
[dbg] dm_msg_request_parse(130): Current Request Message ID:
200864995
[dbg] dm_msg_request_parse(131): Current Request Message Version: 1.
0.0
[dbg] dm_msg_request_parse(132): Current Request Message Method:
thing.service.property.set
[dbg] dm_msg_request_parse(133): Current Request Message Params: {"
LightSwitch":1}
[dbg] dm_ipc_msg_insert(87): dm msg list size: 0, max size: 50
[inf] dm_msg_response(262): Send URI: /sys/a1I1nn8vPf4/Light001/
thing/service/property/set_reply, Payload: {"id":"200864995","code":
200,"data":{}}
[inf] MQTTPublish(515): Upstream Topic: '/sys/a1I1nn8vPf4/Light001/
thing/service/property/set_reply'
[inf] MQTTPublish(516): Upstream Payload:

> {
>     "id": "200864995",
>     "code": 200,
>     "data": {
>     }
> }

[inf] dm_client_publish(121): Publish Result: 0
[inf] _iotx_linkkit_event_callback(223): Receive Message Type: 15
[inf] _iotx_linkkit_event_callback(225): Receive Message: {"devid":0
,"payload":{"LightSwitch":1}}
[dbg] _iotx_linkkit_event_callback (403) :Current Devid:0
[dbg] _iotx_linkkit_event_callback (404) :Current Payload:{"
LightSwitch":1}
user_property_set_event_handler. 160: Property Set Received, Devid:
0, Request: {"LightSwitch":1}
```