Simple Message Notification

Best Practices

Issue	01
Date	2024-12-17

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


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

Notice

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

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

Security Declaration

Vulnerability

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

Contents

1 Using FunctionGraph to Forward SMN Messages to Slack..... 1

1 Using FunctionGraph to Forward SMN Messages to Slack

Scenarios

This section described how to use FunctionGraph to forward SMN messages to Slack. Slack, developed by Slack Technologies, is cloud-based instant messaging software. Slack does not have an API to receive HTTPS messages sent from SMN. So, you cannot directly add a Slack subscription to receive SMN notifications. Instead, you need to use FunctionGraph to forward messages to a specified URL.

Prerequisites

- You have created an SMN message topic, for example, **smn-test**. For details, see **Creating a Topic**.
- You have downloaded Slack.

Procedure

To use FunctionGraph to send SMN messages to Slack, perform the following operations:

1. **Configuring Slack**
2. **Configuring FunctionGraph**
3. **Configuring SMN**

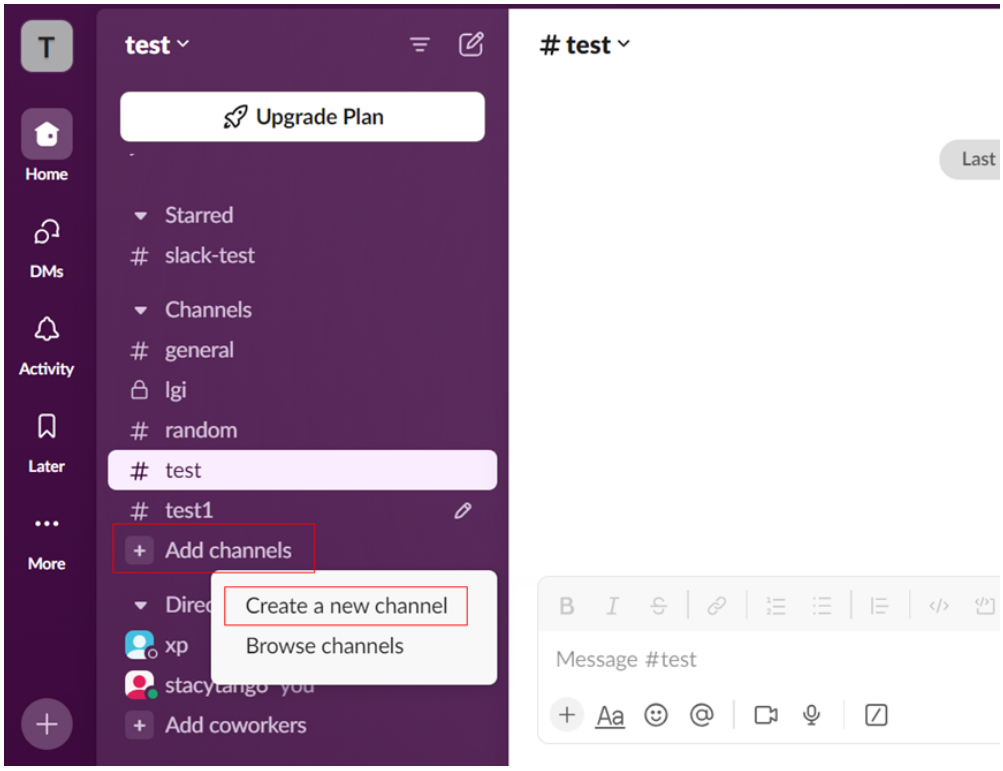
Configuring Slack

NOTE

The following operations are for reference only. For details, visit the **Slack website**.

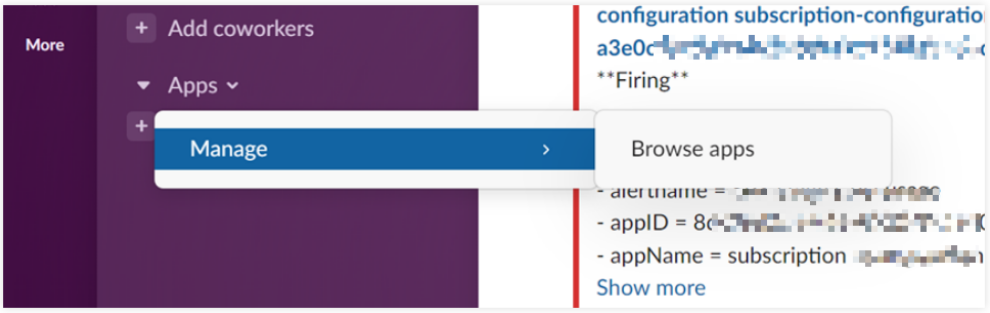
1. Create or select a channel on Slack.

Figure 1-1 Creating a channel



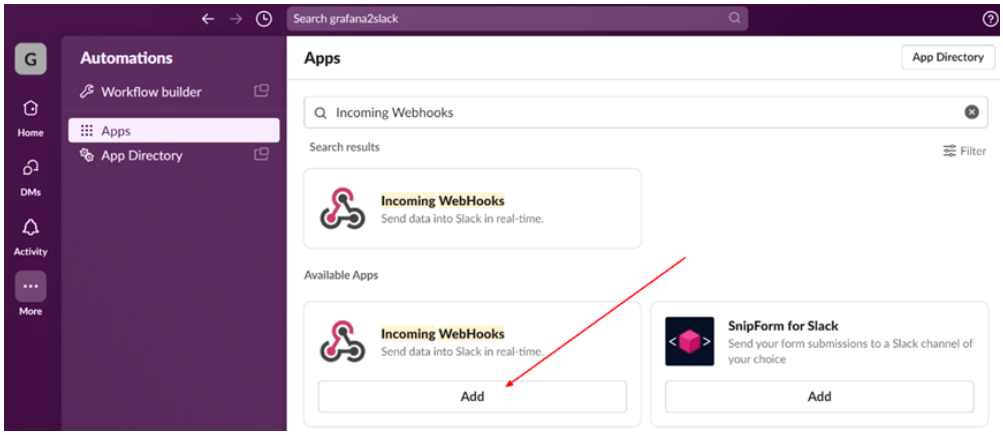
2. Choose **Apps** > **Manage** > **Browse apps**.

Figure 1-2 Choosing **Apps**



3. Search for **Incoming Webhooks** and click **Add**.

Figure 1-3 Adding **Incoming Webhooks**



4. Click **Configuration** to go to the configuration page and obtain the Webhook URL.

Figure 1-4 Clicking **Configuration**

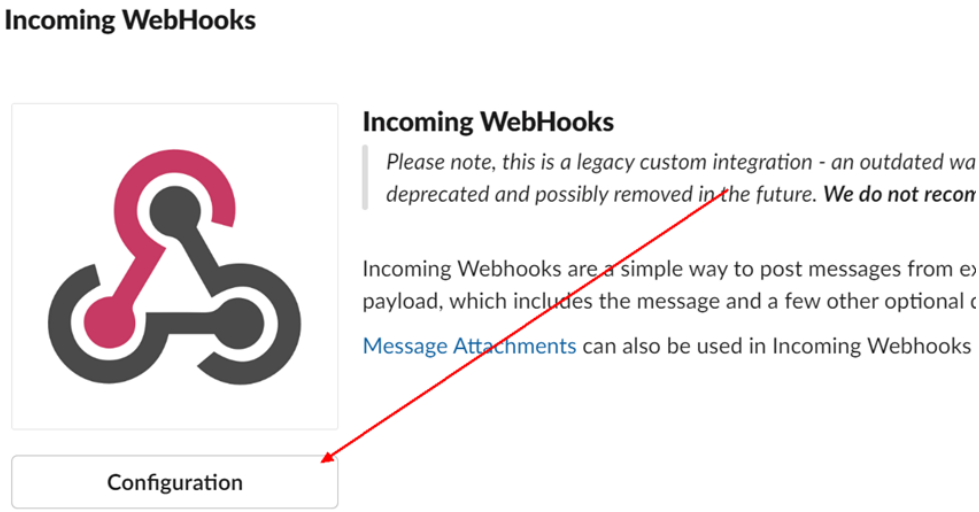
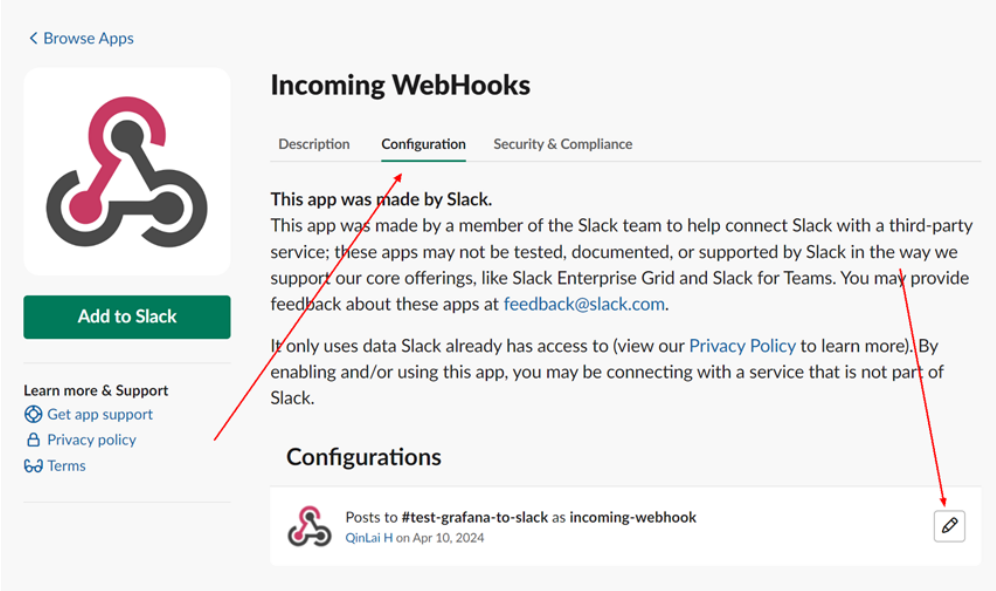
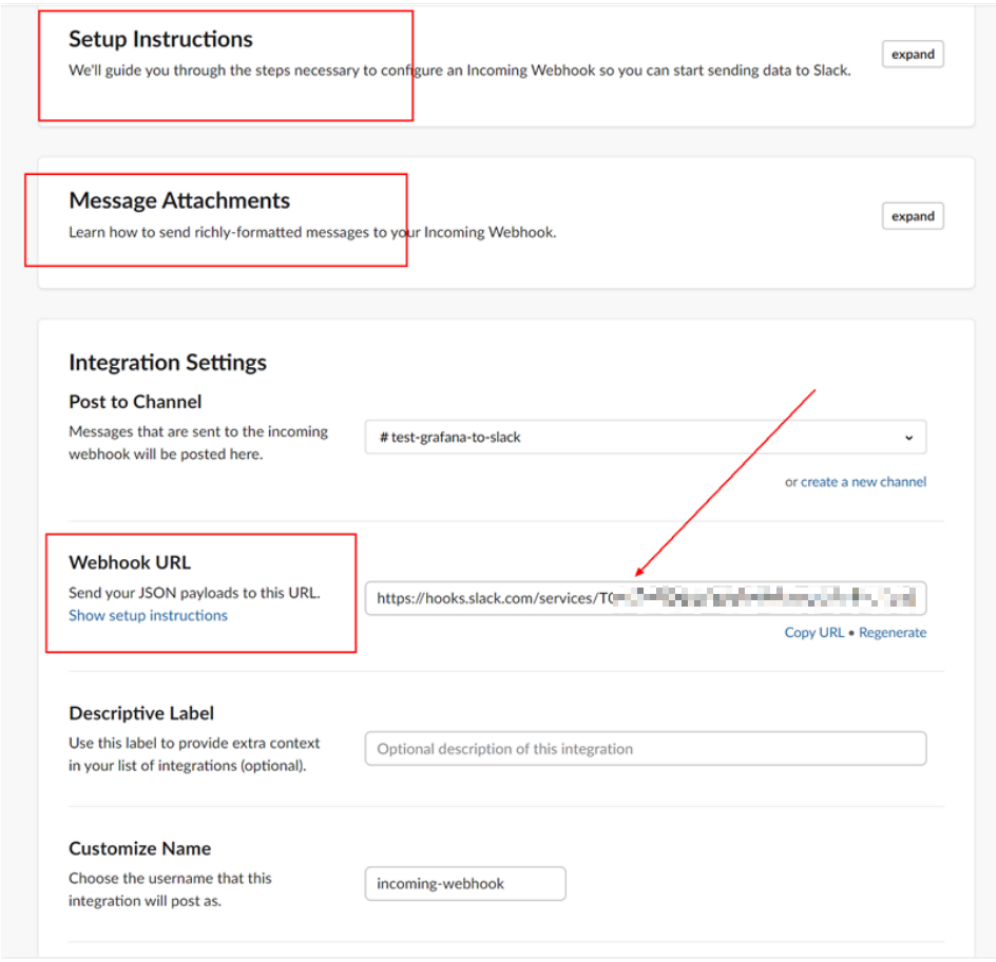


Figure 1-5 Obtaining the Webhook URL



5. Save the obtained Webhook URL.

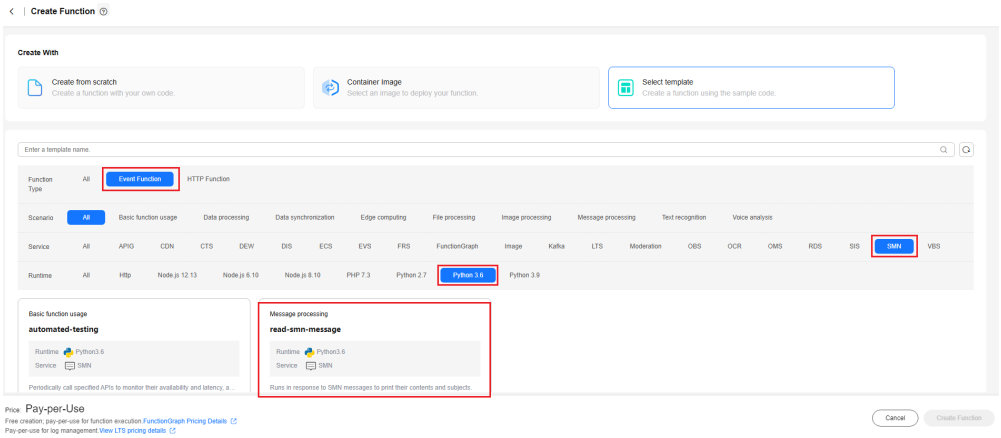
Figure 1-6 Saving the Webhook URL



Configuring FunctionGraph

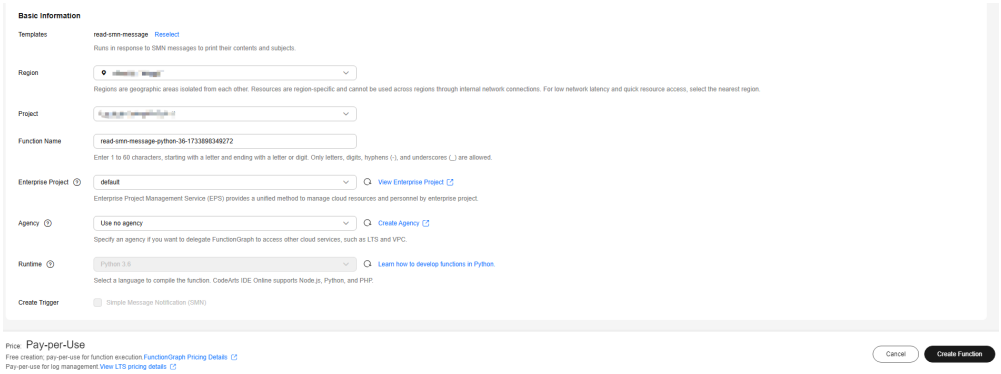
1. Log in to the **FunctionGraph console**. In the navigation pane, choose **Functions** > **Function List**.
2. Click **Create Function** in the upper right corner and choose **Select template**.
3. Select the template shown in **Figure 1-7** and click **Configure**.

Figure 1-7 Selecting a template



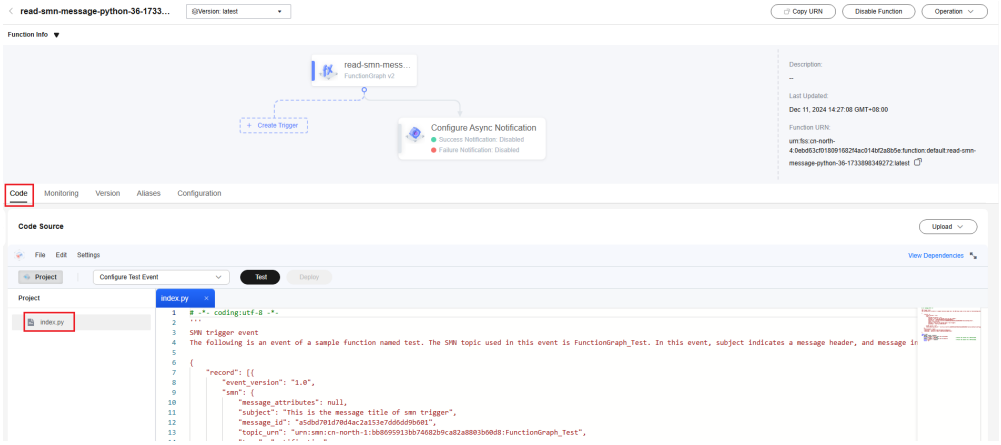
4. Set the function name, select an agency, retain default values for other parameters, and click **Create Function**.

Figure 1-8 Configuring basic information



5. On the configuration details page, click the **Code** tab and edit the code in **index.py**.

Figure 1-9 Editing the code



The following is an example of the code sent to Slack: (In actual use, set **webhook_url** to the Webhook URL obtained from Slack or perform other operations as required.)

```
# -*- coding:utf-8 -*-
import json
import requests

def handler (event, context):
    # Common part: Construct the body fields required by the destination address. event contains all fields sent by SMN.
    payload = {'text': event["record"][0]["smn"]["message"]}

    # Forwarding part: Call a Slack API to forward messages to the Slack platform.
    try:
        webhook_url = 'Webhook URL obtained from Slack'
        response = requests.post(webhook_url, json=payload)
        response.raise_for_status()
        print('Message sent successfully')
    except requests.exceptions.RequestException as e:
        print(f"Error sending message to Slack: {e}")

    return event
```

6. After the editing is complete, press **Ctrl+S** to save the modification and update the code.
7. Click **Create Trigger**.
8. Set the following parameters:
   - **Trigger Type**: Select **Simple Message Notification (SMN)**.
   - **Topic Name**: Select a topic, for example, **smn-test**.

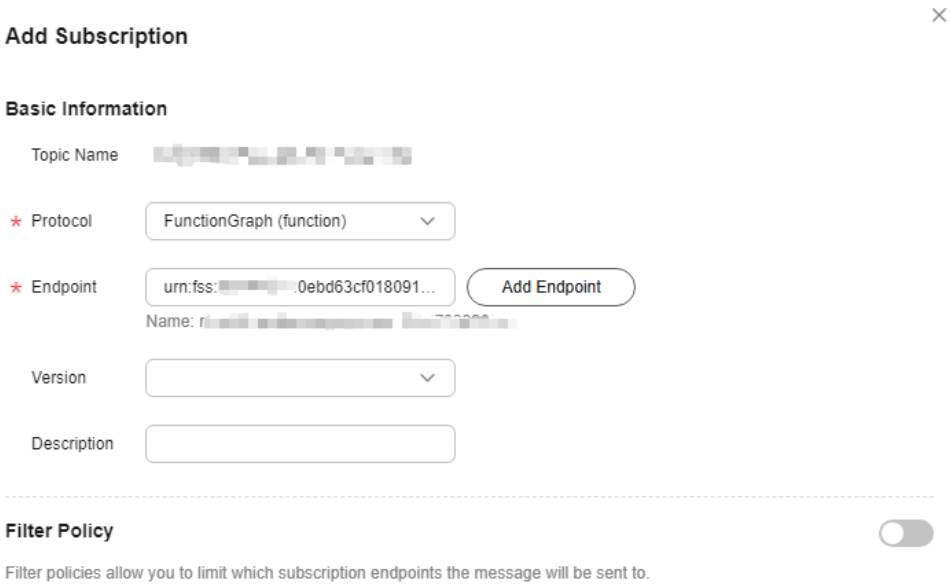
Figure 1-10 Creating a trigger

9. Click **OK**. After the topic is created, all messages sent to the topic are forwarded to the destination address through FunctionGraph.

Configuring SMN

1. Log in to the **SMN console**. In the navigation pane, choose **Topic Management** > **Topics**.
2. Locate the topic **smn-test** and click **Add Subscription** in the **Operation** column.
3. Set the following parameters:
   - **Protocol**: Select **FunctionGraph (function)**.
   - **Endpoint**: Select a created function, for example, **smn-test**.

   Figure 1-11 Adding a subscription



4. Click **OK**.
5. Locate the topic **smn-test** and click **Publish Message** in the **Operation** column.
6. Set the following parameters:
   - **Message Format**: **Text**
   - **Message Content**: Enter the message content.
7. Click **OK**.

Viewing the Received Message

After a message is published, you can receive the message on Slack. **Figure 1-12** shows an example.

Figure 1-12 Message received on Slack