

Skebbby e Auth0 per l'autenticazione a più fattori (MFA)

Guida alla configurazione

L'autenticazione a più fattori (MFA) è fondamentale per rafforzare la sicurezza degli account. Utilizzando un servizio come **Auth0** e integrando un **provider di SMS personalizzato** come **Skebbby**, puoi aggiungere un livello di sicurezza robusto e affidabile per i tuoi utenti.

Questa guida ti mostrerà **come configurare Skebbby all'interno di Auth0 per inviare codici di verifica MFA tramite SMS**.

Contenuti:

1. Preparazione.....	1
2. Configura Auth0 per i messaggi SMS personalizzati.....	2
3. Attivare e testare la configurazione.....	10

1. Preparazione

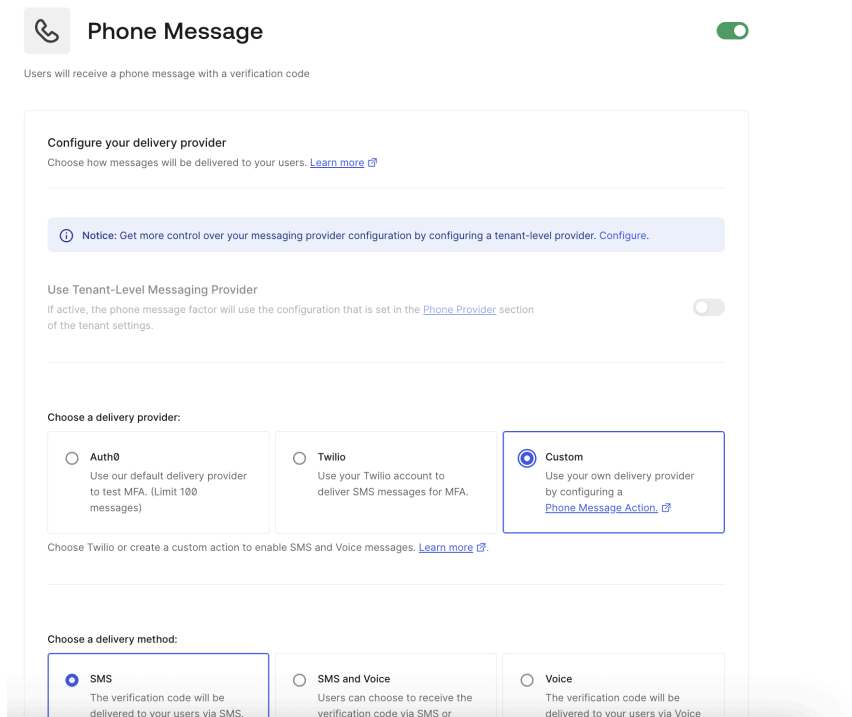
Prima di iniziare, assicurati di avere:

- Un account **Auth0**.
- Un account **Skebbby** con credenziali API attive e del credito disponibile per l'invio degli SMS.

2. Configura Auth0 per i messaggi SMS personalizzati

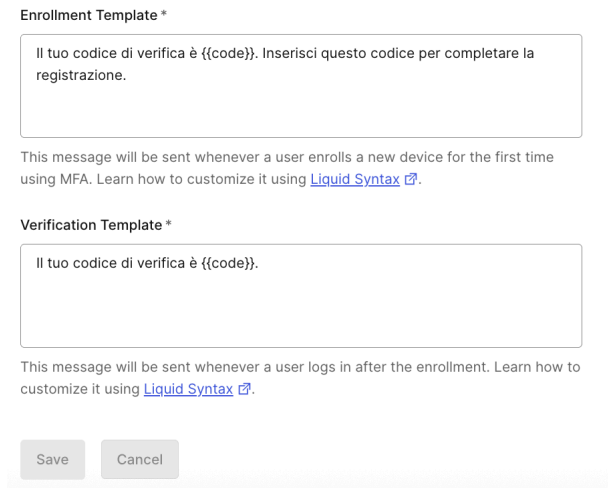
Auth0 ti permette di usare il tuo provider di SMS. Per farlo, devi creare un **Hook** che gestisca l'invio dei messaggi.

1. Accedi al tuo pannello di controllo Auth0.
2. Nel menu laterale, vai su **Security > Multi-Factor Auth**
3. Clicca su Phone message e abilitalo. Nel fare ciò ricordati di selezionare Custom come “delivery provider” e SMS come “delivery method”



The screenshot shows the 'Phone Message' configuration page in Auth0. At the top, there's a toggle switch for 'Phone Message' which is turned on. Below it, a section titled 'Configure your delivery provider' allows choosing how messages will be delivered. A notice indicates that more control can be gained by configuring a tenant-level provider. Under 'Use Tenant-Level Messaging Provider', there's a toggle switch that is currently off. The 'Choose a delivery provider' section has three options: 'Auth0' (selected by default), 'Twilio', and 'Custom' (which is highlighted with a blue border). The 'Custom' option description says to use your own delivery provider by configuring a 'Phone Message Action'. Below this, there's a link to 'Learn more'. The 'Choose a delivery method' section has three options: 'SMS' (selected with a blue dot), 'SMS and Voice', and 'Voice'. The 'SMS' option description says the verification code will be delivered via SMS.

Nella stessa sezione puoi anche customizzare il testo del messaggio inviato



The screenshot shows the 'Enrollment Template' and 'Verification Template' configuration. The 'Enrollment Template' section has a text box containing the message: 'Il tuo codice di verifica è {{code}}. Inserisci questo codice per completare la registrazione.' Below this, a note states: 'This message will be sent whenever a user enrolls a new device for the first time using MFA. Learn how to customize it using Liquid Syntax.' The 'Verification Template' section has a text box containing the message: 'Il tuo codice di verifica è {{code}}.' Below this, a note states: 'This message will be sent whenever a user logs in after the enrollment. Learn how to customize it using Liquid Syntax.' At the bottom, there are 'Save' and 'Cancel' buttons.

4. Una volta fatto ciò vai nel menu laterale, e clicca su **Actions > Triggers**

Seleziona la voce MFA Notifications e `send-phone-message`

MFA Notifications

When using SMS as a factor for Multi-factor Authentication (MFA) or to configure a custom provider.

 **send-phone-message**

Triggers when using a custom provider to send the messages for the enrollment and the challenge process >

Ti si aprirà una pagina come la seguente:

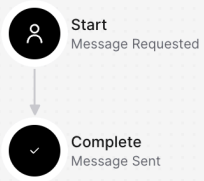
[← Choose trigger](#)

Send Phone Message

Customize the trigger for your SMS provider for multifactor authentication.

[?](#) [Apply](#)

All changes are live



```
graph TD; Start((Start  
Message Requested)) --> Complete((Complete  
Message Sent));
```

Add Action

[+](#)

[Custom](#) [Installed](#)

You do not have any custom actions

[Create Action](#)

5. Nel blocco del flusso, aggiungi una nuova azione e seleziona **Create Custom Action** come tipo di azione.

← Choose trigger

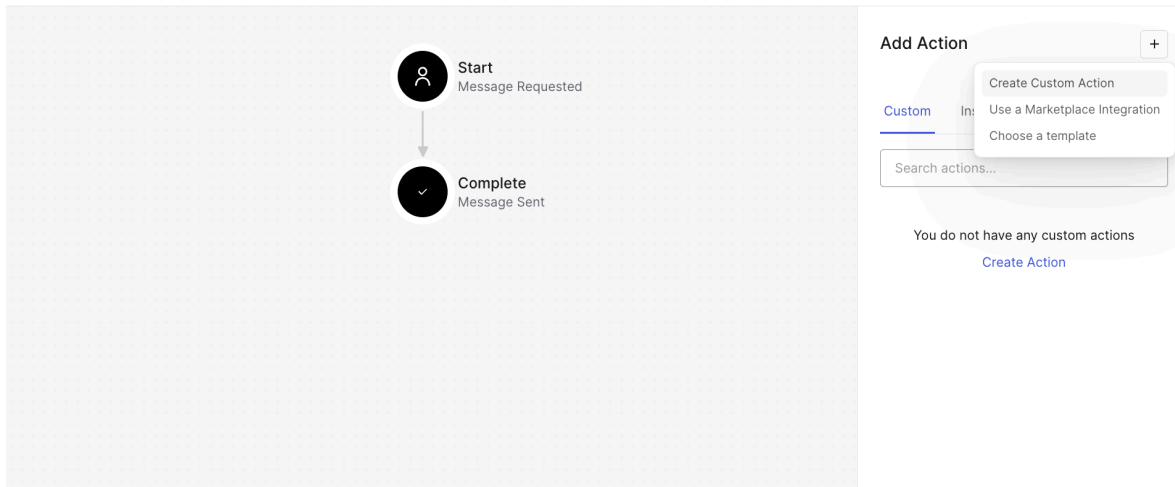
Send Phone Message

Customize the trigger for your SMS provider for multifactor authentication.



Apply

All changes are live



Definisci il nome ad esempio "skebbby" e clicca su crea. Ti si aprirà una scheda come la seguente:

← Back to Triggers

<> **Skebbby** 

Send Phone Message, Runtime: Node 22 (Recommended)



Version History

Save Draft

Deploy

Draft saved

Secrets

Secrets allow you to securely define secret or privileged values that can be accessed in your running code as properties of the event.secrets object.

[Add Secret](#)

```

1 /**
2  * Handler that will be called during the execution of a SendPhoneMessage flow.
3  *
4  * @param {Event} event - Details about the user and the context in which they are logging in.
5  * @param {SendPhoneMessageAPI} api - Methods and utilities to help change the behavior of sending a phone message.
6  */
7 exports.onExecuteSendPhoneMessage = async (event, api) => {
8   //
9 }

```

View Samples

6. Incolla il seguente codice per l'invio degli SMS che userà l'API di Skebbby. Il codice andrà personalizzato con le tue credenziali API di Skebbby.

JavaScript

JavaScript

```
exports.onExecuteSendPhoneMessage = async (event, api) => {

  const axios = require('axios');

  const SKEBBY_USER_KEY = event.secrets.wsbuser;

  const SKEBBY_ACCESS_TOKEN = event.secrets.wsbtokens;

  if (!SKEBBY_USER_KEY || !SKEBBY_ACCESS_TOKEN) {

    console.error("Skebbby API credentials are not configured.");

    // Optionally, you can handle this error gracefully in your
    flow

    return;

  }

  const recipient = event.message_options.recipient;

  const message = event.message_options.text;

  const url = 'https://api.skebbby.it/API/v1.0/REST/sms';

  const body = {

    message: message,

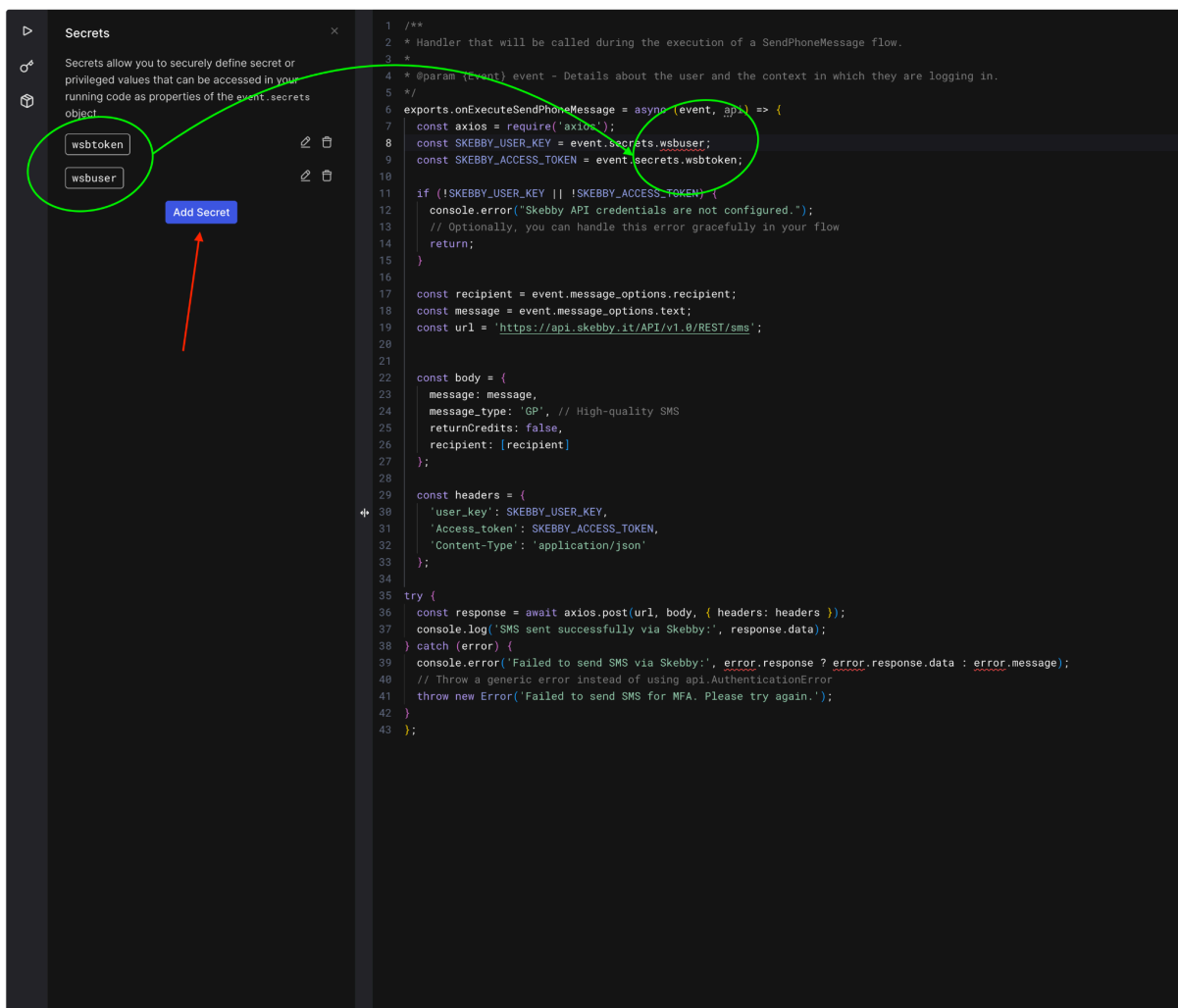
    message_type: 'GP', // High-quality SMS
```

```
    returnCredits: false,  
    recipient: [recipient]  
  };  
  
  const headers = {  
    'user_key': SKEBBY_USER_KEY,  
    'Access_token': SKEBBY_ACCESS_TOKEN,  
    'Content-Type': 'application/json'  
  };  
  
  try {  
    const response = await axios.post(url, body, { headers: headers  
  });  
  
    console.log('SMS sent successfully via Skebbby:',  
response.data);  
  
  } catch (error) {  
  
    console.error('Failed to send SMS via Skebbby:', error.response  
? error.response.data : error.message);  
  
    throw new Error('Failed to send SMS for MFA. Please try  
again.');
```

Nota sulla sicurezza: Per la gestione delle credenziali (come `skebbbyUser` e `skebbbyToken`), è fondamentale usare le **variabili d'ambiente** di Auth0 (denominate "secrets") per non esporre informazioni sensibili direttamente nel codice.

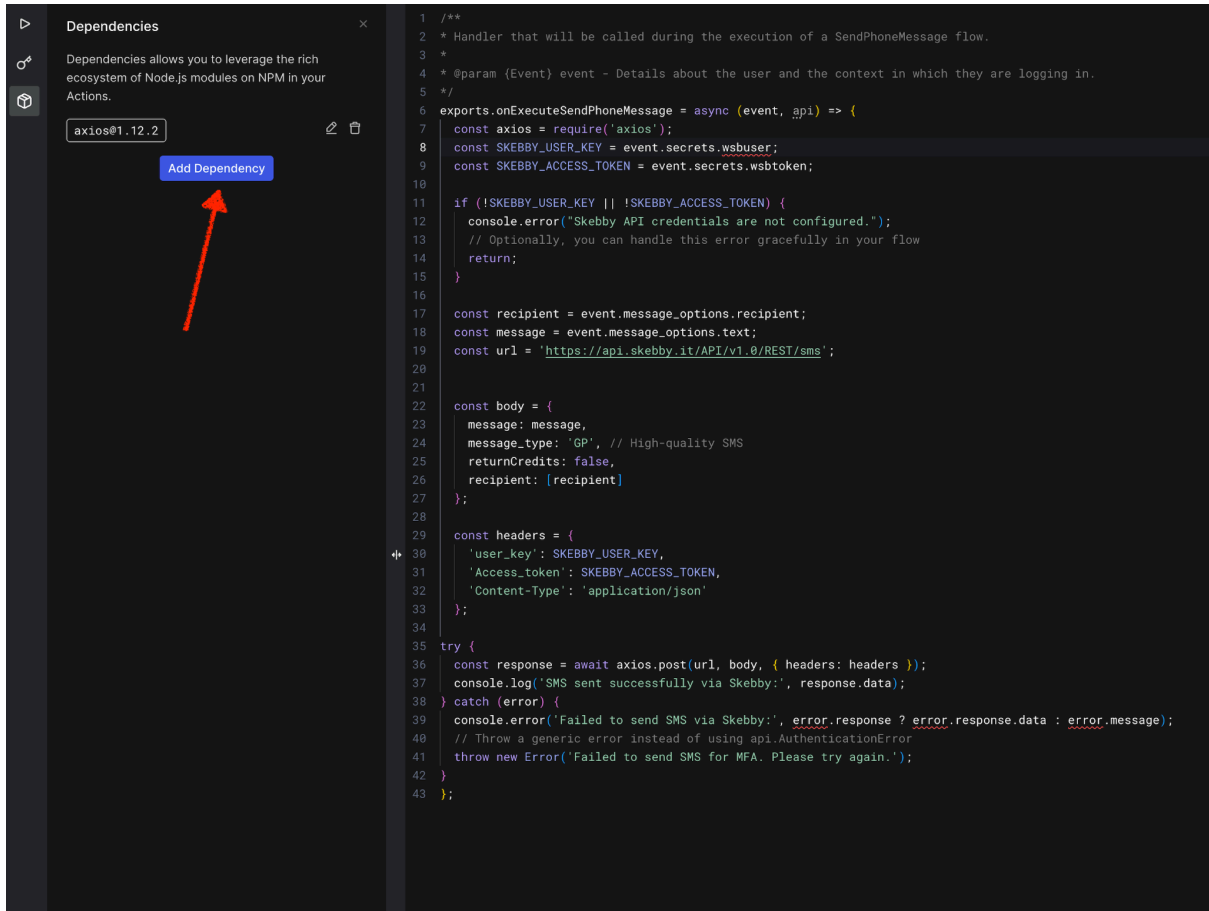
7. Aggiungere i secret:

1. Clicca sul bottone nel menù di sinistra: Add secret
2. Si aprirà un popup in cui inserire il nome del secret
3. Usa **wsbuser** per inserire l'id utente
4. Usa **wsbtoken** per inserire il token segreto



8. Aggiungere la dipendenza a axios

1. Clicca sul bottone a sinistra, si aprirà il form di aggiunta dipendenze
2. Inserisci **axios** e conferma

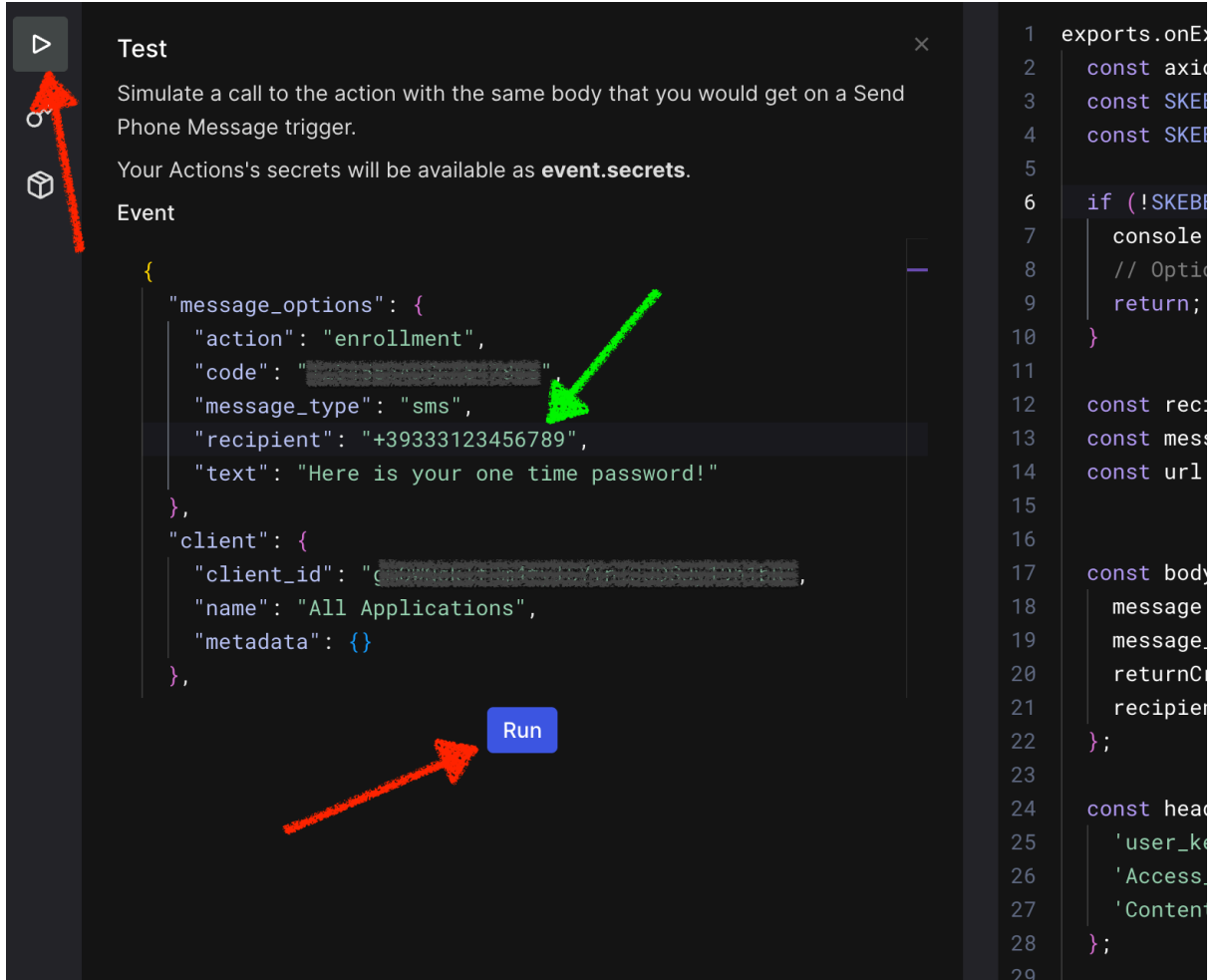


The screenshot shows the Skebbby IDE interface. On the left, the 'Dependencies' panel is open, displaying the text 'Dependencies allows you to leverage the rich ecosystem of Node.js modules on NPM in your Actions.' Below this, there is a search bar containing 'axios@1.12.2' and a blue 'Add Dependency' button. A red arrow points to the 'Add Dependency' button. On the right, the code editor shows a JavaScript script for sending an SMS via the Skebbby API. The script includes comments, imports, and logic for handling the API request and response.

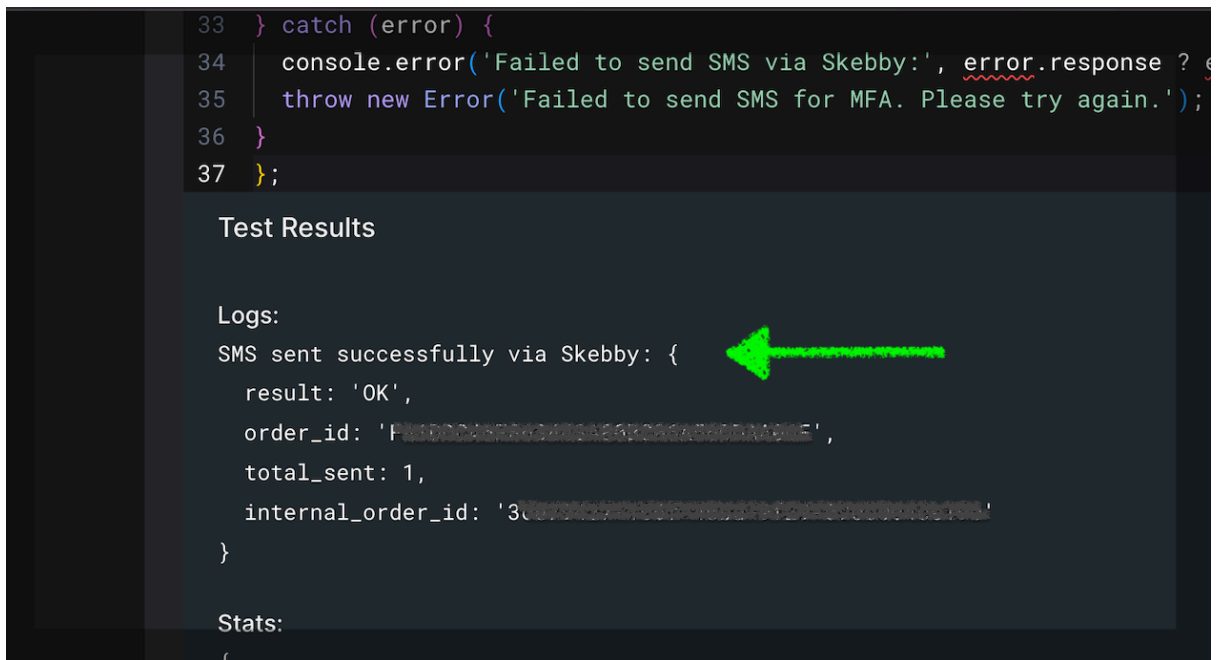
```
1 /**
2  * Handler that will be called during the execution of a SendPhoneMessage flow.
3  *
4  * @param {Event} event - Details about the user and the context in which they are logging in.
5  */
6  exports.onExecuteSendPhoneMessage = async (event, api) => {
7    const axios = require('axios');
8    const SKEBBY_USER_KEY = event.secrets.wsbuser;
9    const SKEBBY_ACCESS_TOKEN = event.secrets.wsbtoken;
10
11    if (!SKEBBY_USER_KEY || !SKEBBY_ACCESS_TOKEN) {
12      console.error("Skebbby API credentials are not configured.");
13      // Optionally, you can handle this error gracefully in your flow
14      return;
15    }
16
17    const recipient = event.message_options.recipient;
18    const message = event.message_options.text;
19    const url = 'https://api.skebbby.it/API/v1.0/REST/sms';
20
21
22    const body = {
23      message: message,
24      message_type: 'GP', // High-quality SMS
25      returnCredits: false,
26      recipient: [recipient]
27    };
28
29    const headers = {
30      'user_key': SKEBBY_USER_KEY,
31      'Access_token': SKEBBY_ACCESS_TOKEN,
32      'Content-Type': 'application/json'
33    };
34
35    try {
36      const response = await axios.post(url, body, { headers: headers });
37      console.log('SMS sent successfully via Skebbby:', response.data);
38    } catch (error) {
39      console.error('Failed to send SMS via Skebbby:', error.response ? error.response.data : error.message);
40      // Throw a generic error instead of using api.AuthenticationError
41      throw new Error('Failed to send SMS for MFA. Please try again.');
```

9. Esegui un test dello script

1. Seleziona il triangolo nel menù a destra
2. Inserisci un recipient valido
3. Premi Run e attendi il messaggio



-
-
-
-
- Verifica l'esito nell'area "Test Results"



10. Ricordati di pubblicare lo script!!

Una volta effettuate le modifiche clicca sul tasto **Deploy in alto a destra**



3. Attivare e testare la configurazione

1. Verifica che nella sezione MFA phone message sia abilitato e che sia selezionato il supplier custom
2. Testa il flusso di login di un utente per verificare che il codice di verifica venga inviato correttamente tramite Skebbby.

Seguendo questi passaggi, potrai integrare Skebbby come provider di SMS personalizzato per l'MFA di Auth0, offrendo un'esperienza di autenticazione sicura e affidabile.