

SlackでWebhookを使用する方法

概要

Webhook（別名コールバック）を使用すると、サードパーティーのAPIをトリガーしたりWalkMeイベントデータを分析プラットフォームに追加したりできます。Webhookは、例えばShoutOutが表示されるなど、指定されたWalkMeイベントによってトリガーされます。WebhookはWalkMeと選択したサードパーティーのプラットフォームを統合します。

この記事ではSlack向けのWebhookを定義する方法について説明します。

ユースケース

Webhook for Slackのユースケースには以下のようなものがあります。

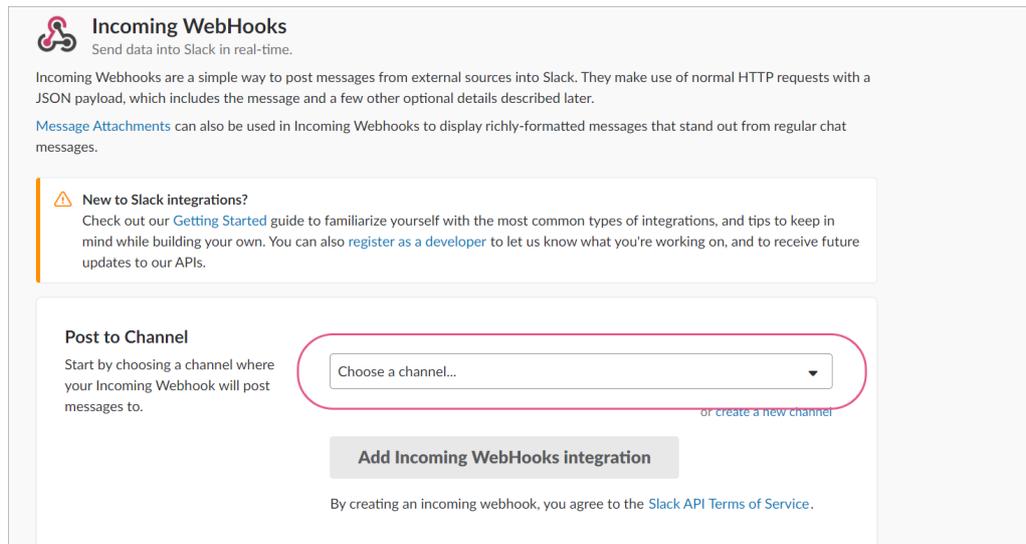
- 追跡するWalkMeイベント（特定のWalkMeエレメントのクリック）を送信して、選択するSlackチャンネルに送信します。
- 指定されたイベントが発生した時点でリアルタイム通知を受けとります。
- エンドユーザーのニーズを満たして、そのジャーニーを追うために、どのようにWalkMeをリアルタイムで使用するかを深く理解します。
- 膨大な[Slack APIライブラリ](#)を使用して、着信するWebhookを操作して活用します。

Webhook for Slackを定義するステップ

ステップ1 Slackチャンネルを作成します

ステップ2 WebhookをSlackチャンネルに追加します

1. Slack検索バーに「Apps」と入力して、[Apps (アプリ)] をクリックします。
2. 左上で [Manage apps... (アプリの管理)] をクリックします。
3. 検索バーで、[Incoming Webhooks] と入力して、[Incoming Webhooks (受信Webhooks)] をクリックします。
4. 緑のインストールボタンをクリックします。
5. [Post to Channel (投稿先チャンネル)] でドロップダウンメニューをクリックして、ステップ1で作成したチャンネルを選択します。



6. [Add Incoming Webhooks integration (受信Webhookの統合を追加)] をクリックします。
7. [Save Setting (設定を保存)] をクリックします。

ステップ3：作成したSlackチャンネルに入ります

ステップ4 [Incoming-webhook (受信Webhook)] をクリックします

ステップ5 [Settings (設定)] をクリックします

ステップ6 [Copy URL (URLをコピー)] をクリックします

ステップ7：「[Webhooksを使用して、サードパーティーシステムにWalkMeイベントデータを送信する方法](#)」を確認して [Insights] で新しいWebhookを定義する方法を学習します

ステップ8 [Webhookウィザード] を使用して進みます

ウィザード — ステップ1：イベントを定義します

- イベントは、 [text] プロパティでSlackに送信されます。
- [+ ADD PROPERTY (プロパティの追加)] をクリックして [text] プロパティを追加します。

Edit Webhook

1 Define Event 2 Set Destination

- env.os.name ? [Rename property](#)
- wm.euld ? [Rename property](#)
- wm.language ? [Rename property](#)
- wm.env ? [Rename property](#)

text : Slack Test 

[+ ADD PROPERTY](#)

[CONTINUE >](#)

ヒント

Slackチャンネルに送信したい文を作成し、「@」を付けて表示されるリストからプロパティ名を選択することで、定義済みのプロパティからデータを組み込むことができます。

ウィザード — ステップ2：送信先を設定します

- **送信先プラットフォームの名前**：送信先のシステム（この場合はSlackを識別するために使用します）。

Edit Webhook ✕ Esc

1 Select Event  2 Select Destination

Destination Platform Name

Request Type **Destination URL**

Authorization Type

Headers
 :

[< BACK](#)

- リクエストタイプを「POST」に設定します。

Edit Webhook ✕ Esc

1 Select Event ✓ 2 Select Destination

Destination Platform Name
Slack

Request Type
POST

Destination URL
https://hooks.slack.com/services/

Authorization Type
No Auth

Headers
Content-type : application/json

< BACK TEST SAVE

- ステップ6でコピーした送信先のURLを貼り付けます。

Edit Webhook ✕ Esc

1 Select Event  2 Select Destination

Destination Platform Name

Request Type

Destination URL

Authorization Type

Headers
 :

[< BACK](#)

- ヘッダ - Content-type application/jsonを追加します

Edit Webhook ✕ Esc

1 Select Event  2 Select Destination

Destination Platform Name

Request Type **Destination URL**

Authorization Type

Headers
 :

[< BACK](#)

- [Test (テスト)] をクリックします。「テストが正常に完了しました!」という緑色のメッセージが表示されます。

Edit Webhook ✕ Esc

1
Select Event

➤

2
Select Destination

Destination Platform Name

Request Type **Destination URL**

Authorization Type

Headers

:

< BACK

✔
Tested Successfully!

SAVE

- 定義したSlackチャンネルを開き、Webhookがランダムな単語を送信したことを確認します。



- [SAVE (保存)] をクリックします。
- 最後に、定義したイベントを実行してSlackで結果を確認してください！

ヒント

[このSlack記事](#)を読んで、高度なフォーマットを使用してWebhookをスタイリッシュにしてください。

実装の例：Slackにアンケートの回答を送信する

ステップ1：イベントからアンケート質問に回答したイベントを選択します

Edit Webhook ✕ Esc

1 Select Event

2 Select Destination

Event

Survey Question Answered

Conditions

Survey Name equal to Rate our product And

[+ ADD CONDITION](#)

Advanced >

[CONTINUE >](#)

ステップ2：イベントのすべてのプロパティのチェックを外します

ステップ3：プロパティ「text」を追加し、「ユーザ「@wm.euIdは私たちが@valueと評価しています」という値を設定します

New Webhook

✕
Esc

1
Define Event

2
Set Destination

- owld [?] [Rename property](#)
- wm.env [?] [Rename property](#)
- oName [?] [Rename property](#)

text

:

The user: @wm.euld has rated us: @value



CONTINUE >