Callback Configuration

Last Updated: Friday, September 30, 2022

Overview

A callback entity is a registration to receive callback events to an external URL. It is identified by a *CallbackID*, which uniquely identifies the record. Additionally, each callback is unique for a given callback type, destination URL, and active date range.

Topics in this Section

> **Multiple Destination URLs**
>
> You can have a same callback go to two destination URLs by registering two callbacks with different destination URLs.

Callback Configuration

Callback Entity

The following is the JSON representation of a callback entity within the APIs.

```
{
   "callback_id":"Hzh39LP3",
   "callback_type":"person_added",
   "event_type":"person_added", //deprecated
   "destination":{
      "url":"http://destination.url",
      "method":"POST",
      "content_type":"application/json"
   },
   "start_date":"2017-01-01T00:00Z",
   "end_date":"2017-02-01T00:00Z",
   "url":"/companies/:company_key/config/callbacks/:callback_id",
   "created_at":"2017-04-15T14:30Z",
   "updated_at":"2017-05-18T19:30Z"
}
```

Elements

Data Element	Type	Description	Required	Default
callback_id	String	Vibes unique identifier for each callback record.	*N/A*	
callback_type	String	Callback type to trigger this callback. Replacement for parameter event_type, which has multiple meaning throughout the platform.	event_type OR callback_type is required	
event_type (Deprecated)	*String*	*Callback/Event type to trigger this callback. (Deprecated)*	*event_type OR callback_type is required*	
destination. url	String	External URL to submit the callback data. You may register multiple unique callback entities to send to different url endpoints (Limit 50). Length is limited to 1024 characters.	Required	
destination. method	String	Method to submit the data to the destination URL. POST and PUT are valid methods.	Optional	POST

destination.content_type	String	Content type to set when submitting the data to the destination URL. Valid content types: • `application/json` • `text/json`	Optional	`application/json`
start_date	Timestamp	Date this callback starts. It should be in the ISO 8601 format - for example: 2017-04-05T14:30Z.	Optional	Now
end_date	Timestamp	Date this callback ends. It should be in the ISO 8601 format - for example: 2017-04-05T14:30Z.	Optional	No end date
url	String	Unique resource URL for the callback.	*N/A*	
created_at	Timestamp	Date this callback was created. It will be in the ISO 8601 format - for example: 2017-04-05T14:30Z.	*N/A*	
updated_at	Timestamp	Date this callback was last updated. It will be in the ISO 8601 format - for example: 2017-04-05T14:30Z.	*N/A*	
list_id	String	Vibes unique identifier for a Subscription List.	Required for Subscription Callback Types	
campaign_id	String	Vibes unique identifier for an Acquisition Campaign.	Required for Acquisition Callback Types	
source_code	String	The short code, long code or alpha code you want to receive unmatched message events for.	Required for Unmatched Message Callback Types	
country_code	String	The country code for the source code you want to receive unmatched message events for.	Required for Unmatched Message Callback Types	
campaign_token	String	The campaign token for the wallet campaign.	Required for Wallet Item Callback Types	

Examples

Person Callback Types

The *person_added* callback event is triggered when a new Person record is added into the Mobile Database.

To register an endpoint to receive the *person_added* callback, follow this example

```
{
  "callback_type":"person_added",
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

The *person_updated* callback event is triggered when an existing Person record has been modified in the Mobile Database.

To register an endpoint to receive the *person_updated* callback, follow this example

```
{
  "callback_type":"person_updated",
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

More information on Person Callbacks is available here.

Subscription Callback Types

The *subscription_added* callback event is triggered when a Person has been added to a Subscription List.

The *subscription_removed* callback event is triggered when a Person has been removed from a Subscription List.

To register an endpoint to receive the *subscription_added* callback, follow this example:

```
{
   "callback_type":"subscription_added",
   "subscription_added":{
         "list_id":"YOUR_SUBSCRIPTION_LIST_ID"
  },
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

To register an endpoint to receive the *subscription_removed* callback, follow this example:

```
{
   "callback_type":"subscription_removed",
   "subscription_removed":{
         "list_id":"YOUR_SUBSCRIPTION_LIST_ID"
  },
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

More information on Subscription Callbacks is available here.

Acquisition Campaign Callback Types

The *ack_participant_added* callback event is triggered when a new Participant has been added to the Acquisition Campaign.

To register an endpoint to receive the *ack_participant_added* callback, follow this example:

```
{
  "callback_type": "ack_participant_added",
  "ack_participant_changed": {
      "campaign_id": "YOUR_ACQUISITION_CAMPAIGN_KEY"
    },
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

The *ack_participant_changed* callback event is triggered when a Participant has confirmed or declined the opt-in.

To register an endpoint to receive the *ack_participant_changed* callback, follow this example:

```
{
  "callback_type": "ack_participant_changed",
  "ack_participant_changed": {
      "campaign_id": "YOUR_ACQUISITION_CAMPAIGN_KEY"
    },
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

More information on Acquisition Callbacks is available here.

Subscription List Callback Types

The *subscription_list_added* callback event is triggered when a new Subscription List has been created in the Mobile Database.

To register an endpoint to receive the *subscription_list_added* callback, follow this example:

```
{
  "callback_type":"subscription_list_added",
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

The *subscription_list_updated* callback event is triggered whenever a Subscription List has been modified in the Mobile Database.

To register an endpoint to receive the *subscription_list_updated* callback, follow this example:

```
{
  "callback_type":"subscription_list_updated",
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

More information on Subscription List Callbacks is available here.

Unmatched Message Callback Types

The *unmatched_message_received* callback event is triggered when a message originating from a consumer does not match any campaign or system keyword.

To register an endpoint to receive the *unmatched_message_received* callback, follow this example:

```
{
   "callback_type": "unmatched_message_received",
   "unmatched_message_received": {
        "source_code": "YOUR_SHORT_CODE",
        "country_code": "YOUR_COUNTRY_CODE"
   },
   "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

More information on Unmatched Message Callbacks is available here.

Wallet Callback Types

The wallet_item_install callback event is triggered when a new wallet item is installed by an end user.

To register an endpoint to receive the *wallet_item_install* callback, follow this example:

```
{
  "callback_type": "wallet_item_install",
  "wallet_item_install": {
    "campaign_token": "YOUR_WALLET_CAMPAIGN_TOKEN"
  },
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

The wallet_item_remove callback event is triggered when a wallet item is removed by an end user.

To register an endpoint to receive the *wallet_item_remove* callback, follow this example:

```
{
  "callback_type": "wallet_item_remove",
  "wallet_item_install": {
    "campaign_token": "YOUR_WALLET_CAMPAIGN_TOKEN"
  },
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

More information on Wallet Callbacks is available here.

Push Device Callback Types

The device_added callback event is triggered when a new app has been installed on a device or when a device is moved from another person to this person.

To register an endpoint to receive the *device_added* callback, follow this example:

```
{
  "callback_type": "device_added",
  "device_added": {
    "app_id": "YOUR_APPLICATION_ID"
  },
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

The device_removed callback event is triggered when an app has been uninstalled from a device, otherwise no longer available, or when a device is moved from this person to another person.

To register an endpoint to receive the *device_removed* callback, follow this example:

```
{
  "callback_type": "device_removed",
  "device_added": {
    "app_id": "YOUR_APPLICATION_ID"
  },
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

The device_updated callback event is triggered when an Apple or Google push_token is added or removed. It will not be triggered for other types of updates.

To register an endpoint to receive the *device_updated* callback, follow this example:

```
{
  "callback_type": "device_updated",
  "device_added": {
    "app_id": "YOUR_APPLICATION_ID"
  },
  "destination":{
     "url":"https://YOUR.DESTINATION.URL",
     "method":"POST",
     "content_type":"application/json"
  }
}
```

More information on Push Device Callbacks is available here.

Event-Triggered Callback Types

The Event-Triggered Callback, also known as the event_processed callback, is triggered when the processing status of an Event has changed. You can configure the callback to be triggered only for a specific *event_type* or choose to receive callbacks for all processed events, regardless of *event_type*.

To register an endpoint to receive the event_processed callback, follow this example:

```
{
  "callback_type":"event_processed",
  "event_processed": {
    "event_type": "order_shipped"
  },
  "destination":{
    "url":"http://some.url",
    "method":"POST",
    "content_type":"application/json"
  },
  "start_date":"2017-01-01T00:00Z",
  "end_date":"2017-02-01T00:00Z",
}
```

To receive callbacks for all processed events, simply remove the event_type. See example below:

```
{
  "callback_type":"event_processed",
  "destination":{
    "url":"http://some.url",
    "method":"POST",
    "content_type":"application/json"
  },
  "start_date":"2017-01-01T00:00Z",
  "end_date":"2017-02-01T00:00Z",
}
```

Find more information on Event-Triggered Callbacks here.