Facebook Integration

Index

Introduction

The purpose of this documentation is to define the integration of Inbenta's Chatbot and Hyperchat solution with Facebook Messenger.

Features

These are the supported answer types and features:

- Text answers (2,000 characters limit)
- Multiple Options buttons (3 buttons max., with titles limited to 20 characters)
- Polar Questions
- Chained answers
- Forms
- Display a button to open a URL (button title limited to 20 chars., and content text limited to 640 characters)
- Custom FAQ title in buttons when displaying multiple options

Other important features:

- Content ratings (yes/no + comment)
- HyperChat escalation after X no-results answers (triesBeforeEscalation)
- Escalate to HyperChat when the FAQ has 'ESCALATE' setting set to 'TRUE'
- Retrieve Facebook tokens from ExtraInfo
- Retrieve translations from ExtraInfo

Prepare your Inbenta instances

Create tokens object in ExtraInfo

1. Open your Inbenta App instance and go to *Knowledge > Extra Info.* Click on '*Manage groups and types'*, then click on *'Add a new group'* and name it **'facebook**'.
2. Click *'Add type'* in your group. Name it **'verify_token**'. Set the property name type to '**value**' and select 'text' from the dropdown. Click on *'OK'*.
3. Click *'Add type'* in your group again. Name it '**page_tokens**'. This object has three properties: '**development**', '**preproduction**' and '**production**', all with *'text'* type. Add all three properties and click on *'OK*'.
4. Go to *Knowledge > Extra Info*. Click on '*New Entry'*. Set the name to '**verify_token**'. Select the group '**facebook**' and select the type '**verify_token**'. Set an alphanumeric string as value. You need this string when you set up your UI on Facebook.
5. In *Knowledge > Extra Info,* click on *'New Entry'*. Set the name to '**page_tokens**'. Select the group '**facebook**' and select the type '**page_tokens**'. Later, once you have your page tokens from your Facebook pages, you can fill them in this object. You can have three different page tokens, one for development, another for preproduction and another production.
6. Publish your changes to Extra Info: Go to *Knowledge > Contents*, click on the "Post" button and select the relevant option.

> **Information**
> Remember: Always publish your changes to *Extra Info* using the 'Post' button in *Knowledge > Contents*.

Create translations object in ExtraInfo (optional)

You can manage the translation labels from Extra Info. Here are the steps to create the translations object:

1. In your Inbenta App instance, go to *Knowledge > Extra Info.* Click on *'Manage groups and types' > facebook > Add type*. Name it '**translations**', add a new property with 'Multiple' named with your chatbot's language label (en, es, it...).
2. Inside the language object, add all the labels that you want to override. Each label should be a 'text' type entry (you can find the labels list below).
3. Save your translations object.

You can now create the ExtraInfo object by clicking the **New entry** button, selecting the 'translations' type and naming it as 'translations'. Then, fill each label with your desired translation and remember to publish ExtraInfo by clicking the **Post** button in *Knowledge > Contents*.

Here is a list of all the current labels with their English value:

- **agent_joined** => 'Agent $agentName has joined the conversation.'
- **api_timeout** => 'Please, reformulate your question.'
- **ask_rating_comment** => 'Please tell us why'
- **ask_to_escalate** => 'Do you want to start a chat with a human agent?'
- **chat_closed** => 'Chat closed'
- **creating_chat** => 'I will try to connect you with an agent. Please wait.'
- **error_creating_chat** => 'There was an error joining the chat'
- **escalation_rejected** => 'What else can I do for you?'
- **no** => 'No'
- **no_agents** => 'No agents available'
- **queue_estimation_first** => 'There is one person ahead of you.'
- **queue_estimation** => 'There are $queuePosition people ahead of you.'
- **rate_content_intro** => 'Was this answer helpful?'
- **thanks** => 'Thanks!'
- **yes** => 'Yes'

Information

Remember: Always publish your changes to *Extra Info* using the 'Post' button

```
in Knowledge > Contents.
```

HyperChat integration (optional)

If you use HyperChat, you must subscribe your UI to the HyperChat events. Open your Case Management instance and go to *Case Management > Settings > Chat > Webhooks*. Find the 'Events' column and type "queues:update,invitations:new,invitations:accept,forever:alone,chats:close, messages:new,users:activity". In the 'Target' column paste the URL of your UI, then click on the '+' button to the right.

Building the Facebook template

Setup Chatbot Facebook Template

Download Facebook Template

You can find the Chatbot Facebook template in GitHub. You can download or clone the template from https://github.com/inbenta-integrations/facebook_chatbot_template.

Required Configuration

In your UI directory, go to **conf**. There is a readme file with examples of structure and explanations of use. If you only want to build a chatbot, fill the **key** and **secret** values inside the **conf/custom/api.php** file with your Inbenta Chatbot API credentials. If you want to modify other configuration parameters, copy the desired file(s) from **conf/default** into **conf/custom** and modify the values.

Optional Configuration

You can enable several optional features from the configuration files. You must copy every optional configuration file from **/conf/default** and store the custom version in **/conf/custom**. The bot detects and loads the customized files automatically.

Here is a list of the optional configuration files, with a description of the configuration fields.

HYPERCHAT (chat.php)

- **chat**
    - **enabled**: Enable or disable HyperChat (“**true**” or “**false**”).
    - **version**: HyperChat version. The default and latest one is 1.
    - **appId**: The ID of the HyperChat app. This defines the instance in which the chat opens. You can find it in your instance under *Case Management > Settings > Chat*.
    - **secret**: Your HyperChat instance application secret. You can find it in your instance under *Case Management > Settings > Chat*.
    - **roomId**: The room where the chat opens. This is mapped directly to a queue ID in the Inbenta Case Management App. This is a numeric value, not a string. You can find the list of your rooms in your instance under *Case Management > Settings > Queues*.
    - **lang**: Language code (in ISO 639-1 format) for the current chat. This is used when the engine checks if there are agents available for this language to assign the chat to one of them.
    - **source**: Source id from the sources in your instance. This is a numeric value, not a string. The default value is **3 - Chat**. You can find your sources list it in your instance under *Case Management > Settings > Sources*.
    - **server**: The HyperChat server URL assigned to your instance. Ask your Inbenta representative for this configuration parameter.
    - **server_port**: The port from which you communicate with the HyperChat server. This is defined in your instance under *Case Management > Settings > Chat > Port*
    - **queue**:
        - **active**: Enable or disable the queue system (“**true**” or “**false**”). You **MUST** enable it in your instance too. You do this from *Case Management > Settings > Chat > Queue mode*.
- **triesBeforeEscalation**: Maximum number of no-result answers in a row before the bot should escalate to an agent (if available). This is a numeric value, not a string. Set to zero to leave it disabled.
- **negativeRatingsBeforeEscalation**: Maximum number of negative content ratings in a row before the bot should escalate to an agent (if available). This is a numeric value, not a string. Set to zero to leave it disabled.

CONVERSATION (conversation.php)

- **default:** Contains the API conversation configuration. The values are described below:
    - **answers:**
        - **sideBubbleAttributes:** Dynamic settings to show side-bubble content. Because there is no side-bubble in Facebook the value is an empty array ("**array()**").
        - **answerAttributes:** Dynamic settings to show as bot answer. The default is ["ANSWER_TEXT"]. Setting multiple dynamic settings generates a bot answer with concatenated values with a newline character (\n). Limited to **2,000 characters**.
        - **maxOptions:** Maximum number of options returned in a multiple-choice answer. The maximum allowed value is 3 because Facebook limits the buttons to 3.
    - **forms**
        - **allowUserToAbandonForm:** Whether or not a user is allowed to abandon the form after a number of consecutive failed answers. The default value is **true**.
        - **errorRetries:** The number of times a user can fail a form field before being asked if they want to leave the form. The default value is 3.
    - **lang:** Language of the bot, represented by its ISO 639-1 code. Accepted values: ca, de, en, es, fr, it, ja, ko, nl, pt, zh, ru, ar, hu, eu, ro, gl, da, sv, no, tr, cs, fi, pl, el, th, id, uk
- **user_type**: Profile identifier from the Chatbot App Knowledge Base. Minimum:0. Default:0. You can find your profile list in your Chatbot Instance under *Settings > User Types*.
- **source**: Source identifier (e.g. “facebook”). You use it to filter the logs in the dashboards.
- **content_ratings**
    - **enabled**: Enable or disable the rating feature (“**true**” or “**false**”).
    - **ratings**: Array of options to display in order to rate the content. Every option has the following parameters:
        - **id:** Id of your content rating. You can find your content ratings in your Chatbot instance under *Settings > Ratings*. Remember that your rating type must be "**content**".
        - **label:** Key of the label translation to display within the rating option button. You can configure available labels from **/lang/**. You can also

modify them from your Inbenta App, as described in section **Create translations object in ExtraInfo (optional)**

- **comment:** If **true**, asks for a comment for the rating. This is useful when a user rates a content negatively to ask why the negative rating.
- **isNegative:** If **true**, the bot increments the negative-comments counter in order to escalate with an agent (if HyperChat **negativeRatingsBeforeEscalation** is configured).

- **digester**
  - **button_title**: The value of this dynamic setting shows as the content title in multiple-options buttons. This is useful to set an alternative title that has less than 20 characters because Facebook limits buttons text to **20 chars**.
  - **url_buttons:** Displays a button that opens the configured URL. When a content includes a URL button, the message character limit is reduced to **640 characters**.
    - **attribute_name**: This must be a grouped dynamic setting. Its child values contain the button URL and title.
    - **button_title_var**: Child value in the grouped dynamic setting. Its value contains the url-button title.
    - **button_url_var**: Child value in the grouped dynamic setting. Its value contains the url-button URL.

ENVIRONMENTS (environments.php)

This file allows you to configure a rule to detect the current environment for the connector. It can check the current **http_host** or the **script_name** in order to detect the environment.

- **development:**
  - **type**: Detection type: check the **http_host** (e.g. *www.example.com*) or the **script_name** (e.g. */path/to/the/connector/server.php*).
  - **regex**: Regex to match with the detection type (e.g. "*/^dev.mydomain.com$/m*" will set the "development" environment when the detection type is *dev.example.com*).
- **preproduction**:
  - **type**: Detection type: check the **http_host** (e.g. *www.example.com*) or the **script_name** (e.g. */path/to/the/connector/server.php*).
- **regex**: Regex to match with the detection type (e.g. "*/^.*/staging/.*$/m*" will set the "preproduction" environment when the detection type contains "*/staging/*").

Deployment

You must use a public webserver to serve the Facebook template. This makes it possible for Facebook to send events to it. The environment where the template was developed and tested has the following specifications

- Apache 2.4
- PHP 7.3
- PHP Curl extension
- Non-CPU-bound
- The latest version of **Composer** (Dependency Manager for PHP), to install all dependencies that Inbenta requires for the integration.
- If the client has a **distributed infrastructure**, this means that multiple servers can manage the user session. They must adapt their SessionHandler so that the entire session is shared among all its servers.

Prepare the Facebook environment

Create a Facebook Page

You need a Facebook Account in order to create a page. Log in to your account and go to https://www.facebook.com/pages/creation and fill the required information (Company/Brand or Community, page name and page category) to create your Facebook page. Inbenta suggests that you turn off page notifications in *Page > Settings > Notifications*.

Create a Facebook APP

You need a Developer Facebook Account (This is a standard account with additional dev policies to accept). If you have not set your account as a developer, Facebook will ask you to do it when you login into the developers portal. Go to https://developers.facebook.com and click in *My Apps > Create app* at the top-right corner. Fill the required information to get your Facebook App.

Link Facebook page with a Facebook app

Linking your Facebook page to your Facebook app

Go to your Facebook App under *Products + > Messenger* and click 'Set Up'. Scroll down to *'Access Tokens'* and click on *'Add and remove pages*'. Then, continue with your account and select your page in the check-list. Accept the permissions, click on *'Done'*. Your page is now linked.

Token generation

Go to your Facebook App under *Messenger > Settings > Access tokens* and click on *'Generate token'* in the desired page on the table. Accept the security advice. The token appears.. Copy and save this token in your Inbenta instance under *Knowledge > ExtraInfo > facebook > page_tokens*. Remember to **publish** the "*Extra Info*" after the change and wait until publication is complete.

Webhook Subscription

In your Facebook app, go to *Messenger> Settings > Webhooks* and click on *'Add Callback URL'*. Fill the form with the URL of your UI and your verify token, and click *'Verify and save*'. A table with

your page appears. In the table, click on *'Add subscriptions'* and select '**messages**' and '**messaging_postbacks**' and click on '*Save*'.

```
Information
To verify your webhook, Facebook sends a request to your UI. You must specify the same
verify_token in the form and in your Inbenta instance ExtraInfo.
```

Persistent Menu integration (optional)

What is the persistent menu and how is it used?

You can set the persistent menu for your bot to help people discover and more easily access certain functionalities throughout the conversation. This is useful for users since they can perform direct actions that you may want them to have "easy access" to. In our example app, we created a persistent menu for our messenger application that allows the users to "Contact Agents" directly by clicking on a button.

This button has an action that we set in our chatbot to make a **directCall** to the **escalateToAgents** content. This starts the contact process.

See below how you can integrate this functionality and how it works.

How to add the Get Started button?

Before you can get the Persistent Menu, you must have a Get Started button. Send a simple request to the Facebook API to set the "Get started" button on the messenger application.

To do this easily, make the following request using the tool Postman.

```
// url
https://graph.facebook.com/v3.1/me/messenger_profile?access_token=<YOUR_ACCE
SS_TOKEN>

// json body request
{
  "get_started":{
    "payload":"{\"message\":\"\", \"directCall\":\"sys-welcome\"}"
  }
}
```

This triggers the **directCall** "**sys-welcome**" when a user clicks on the "Get started" button. After you do this, proceed to send the request for the persistent menu. That's it!

The result of this request and the persistent menu should be:

```
{
    "result": "success"
}
```

How to create a persistent menu?

Suppose that you already have a chatbot integration connected to your messenger app.

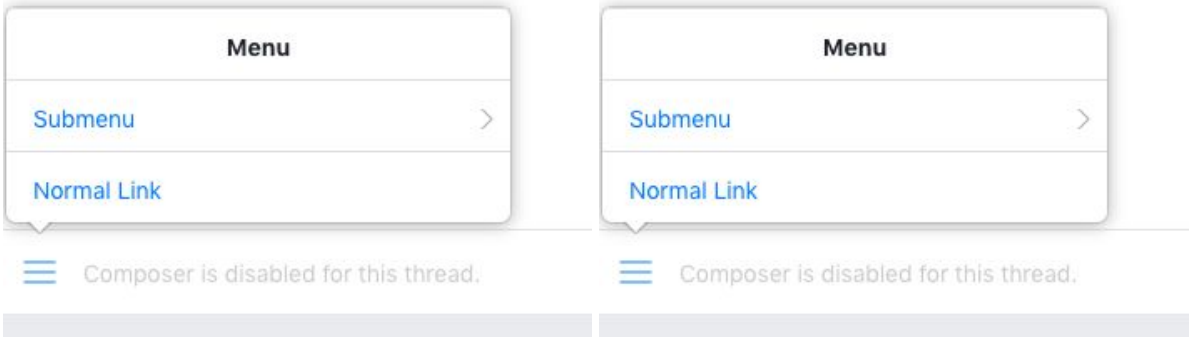
- You simply have to send some POST requests to the Facebook API.
- To create simple buttons like in the example above, send a request like this:

```
// We can do this request with Curl or PostMan - we are going to show the
example in postman as it has an easy user interface

//url we have to send request
https://graph.facebook.com/v2.6/me/messenger_profile?access_token=XXXXXXXXX
//access token have to be taken from (*)(explained below)

// POST REQUEST
// In the body params of the request we have to send a JSON like:
{
  "recipient": <YOURD_PAGE_ID>, (this is the ID of your page) (*)(shown
below)
  "persistent_menu":[
    {
      "locale":"default",
      "composer_input_disabled": true,
      "call_to_actions":[
        {
          "title":"Contact Agents",
          "type":"postback",
          "payload":
"{\"message\":\"\",\"directCall\":\"escalateToAgents\"}"
        }
      ]
    }
  ]
}
```

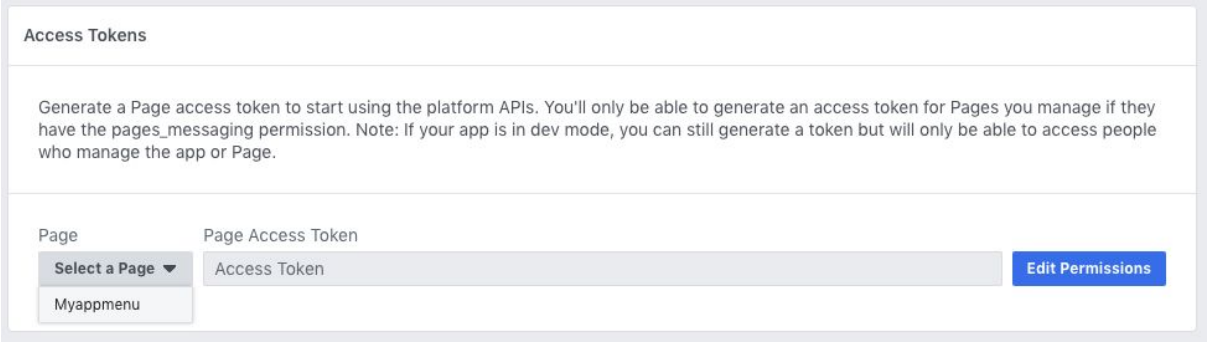
- The example above would look like the previous example. But you can also do nested links, as shown in the example below:

```
{
  "recipient": <YOURD_PAGE_ID>,
  "persistent_menu":[
    {
      "locale":"default",
      "composer_input_disabled": true,
      "call_to_actions":[
        {
          "title":"Submenu",
          "type":"nested",
          "call_to_actions": [
                {
                    "type": "postback",
                    "title": "Contact Agents",
                    "payload":
"{\"message\":\"\",\"directCall\":\"testContent\"}"
                }
            ]
        },
        {
          "title":"Normal Link",
          "type":"postback",
          "payload": "{\"message\":\"\",\"directCall\":\"testContent\"}"
        }
      ]
    }
  ]
}
```

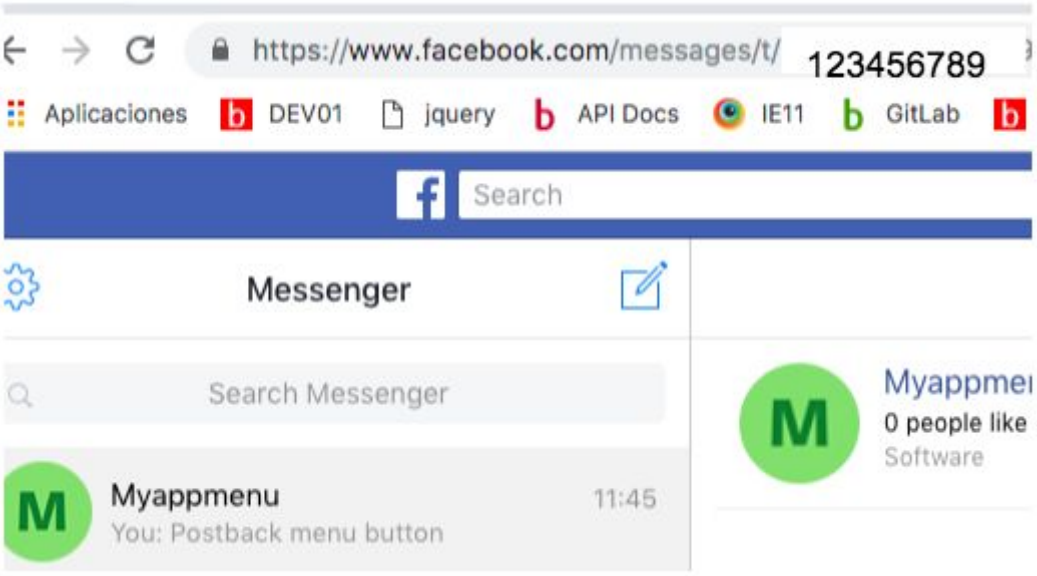
This looks like this:



- How to get an access token: If you already have the Page Access Token from your bot configuration, use it. If you do not, go to your messenger application and select the page you are using. Copy the access token given by Facebook. You must be set as an "Administrator" to see the page:



- How to get the recipient: you can find it easily in the Messenger app page in the URL



Troubleshooting persistent menu

When you send a request to show the persistent menu and you did not set the "Get Started" button, the response from this request is the following error:

```
(#100) You must set a Get Started button if you also wish to use persistent menu.
```

Check the “Get started” section to fix this issue.

Important things to know about the Persistent menu and Get started button

- If your application is already set with a persistent menu, be careful with this information.
- If you do a POST with a new persistent menu, this *DELETES* the current one.
- How do you make changes in the current Get Started and Persistent menu?
- Send a GET request like this: (via postman also)

```
// url
https://graph.facebook.com/v3.1/me/messenger_profile?access_token=<YOUR_ACCESS_TOKEN>&fields=p
ersistent_menu,get_started
```

The answer looks like this:


```
{
    "data": [
        {
            "persistent_menu": [
                {
                    "locale": "default",
                    "composer_input_disabled": true,
                    "call_to_actions": [
                        {
                            "type": "nested",
                            "title": "Submenu",
                            "call_to_actions": [
                                {
                                    "type": "postback",
                                    "title": "Contact Agents",
                                    "payload":
"{\"message\":\"\",\"directCall\":\"testContent\"}"
                                }
                            ]
                        },
                        {
                            "type": "postback",
                            "title": "Normal Link",
                            "payload": "{\"message\":\"\",\"directCall\":\"testContent\"}"
                        }
                    ]
                }
            ],
            "get_started": {
                "payload": "{\"message\":\"\", \"directCall\":\"sys-welcome\"}"
            }
        }
    ]
}
```


Now that you have this information, you can easily modify only the parts (e.g. links or submenus) that you want to change, without losing anything.

Relevant URL's

- Facebook Developers Portal <https://developers.facebook.com>
- Your Facebook Pages <https://www.facebook.com/bookmarks/pages>
- Messenger Documentation <https://developers.facebook.com/docs/messenger-platform/send-messages>
- Messenger Portal <https://www.messenger.com>
- Facebook Token debugger <https://developers.facebook.com/tools/debug/accesstoken>

Troubleshooting

Missing HyperChat messages

Check that your UI is subscribed to HyperChat webhooks, as described **HyperChat integration (optional)**. You must also check if the HyperChat settings in the UI are valid in **conf/custom/chat.php**.

The bot is not answering (webhook configuration)

Check that you configured the URL of your UI correctly in your Facebook App. You can check the current webhook URL in your FB App under *Webhooks > Page > Edit Subscription*. You must also check if the tokens in your instance ExtraInfo are valid.

The bot is not answering (page configuration)

Check that you configured the access token correctly and that it corresponds to the appropriate page. You can use the Facebook Token debugger to check this.

The bot is not answering (application cache)

If the two previous tips do not bring the expected results, maybe your application is caching older token values from "*Extra Info*". If this is the case, delete the cached session files in your server. These files are stored in the configured system temporary path returned by the PHP function **sys_get_temp_dir()**. Usually, it's "/tmp" or "/var/tmp" but may vary depending on your server system and configuration. When you locate the directory, remove all the files named like "*cached-accesstoken-XXXX*" and "*cached-appdata-XXX*".

The bot is not answering (subscription configuration)

Sometimes, Facebook disables your webhook (multiple timeouts, 5XX errors). A quick solution is to resubscribe the webhooks, to try and force Facebook to start sending the requests again.

The bot is not answering (full reboot):

If your bot needs a full reboot, you must unsubscribe your webhook from your App. Then, go to your ***Page > Configuration > Messenger Platform > Applications subscriptions*** and uninstall your application from your page. Finally, resubscribe your webhooks from scratch in your Facebook app.

How to test it?

Test your UI from your Facebook page. Go to https://www.messenger.com and search for your page name in the search bar (top left). Your bot should be answering your messages.