

POLYGLOTBOT

Develop in one language
deliver to many



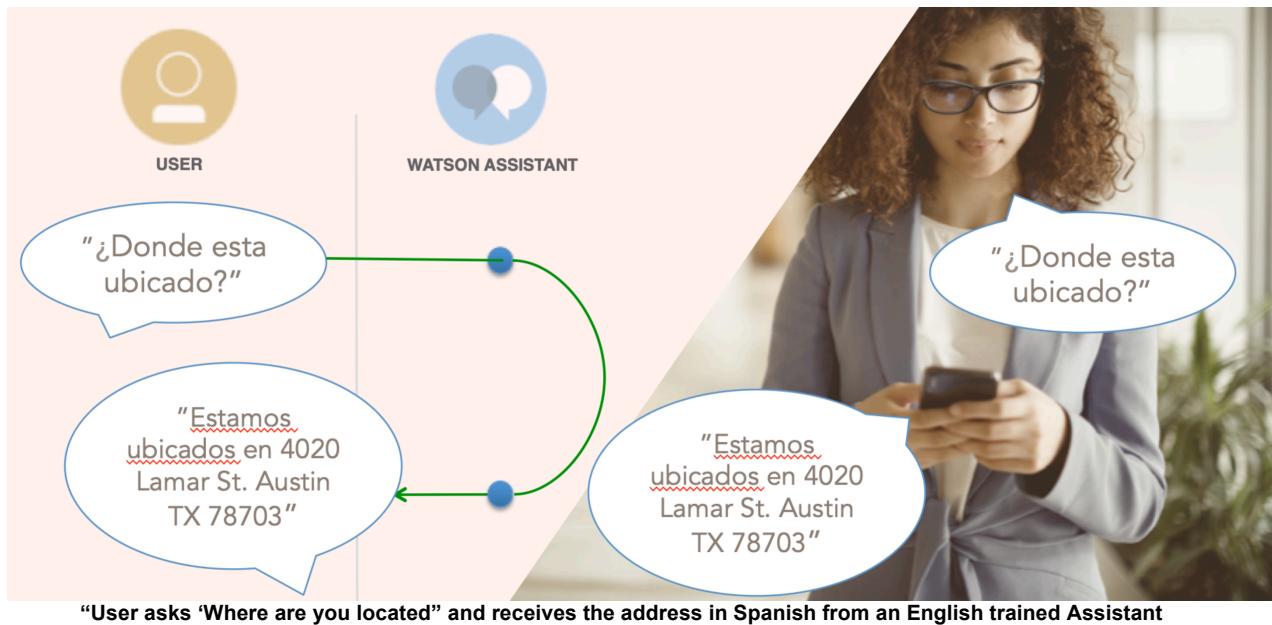
Christopher Crane
Morgan Langlais

Introduction and Overview

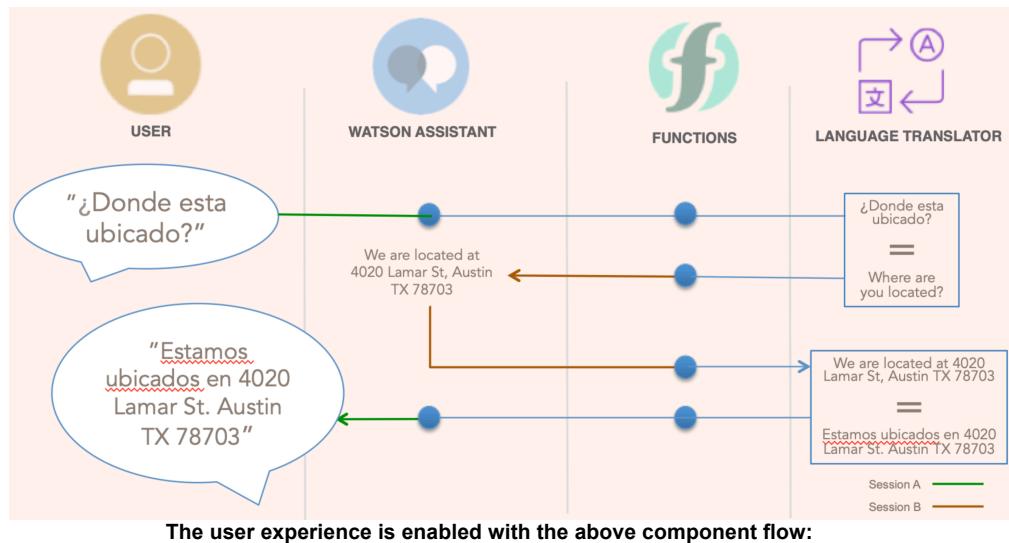
Polyglot Bot (Poly) enables skills to be seamlessly consumed in a wide variety of languages by sending user utterances and responses to the Watson Language Translator service through a Cloud Function. This runbook will walk through:

1. Gathering Requirements
2. Create Cloud Function
3. Create a New Watson Assistant Skill
4. Get the Cloud Functions endpoint
5. Add the webhook to your skill
6. Entity Configuration
7. Dialog Configuration
8. Welcome Configuration

The example below shows a user asking for and receiving an address to an establishment in Spanish despite the fact that the Assistant was configured and trained only in English.



This flow is made possible by linking Watson Assistant to code that sits in [Cloud Functions](#). The Cloud Function is invoked by a webhook from Watson Assistant. Once invoked, the Cloud Function leverages both Language Translator and Assistant to correctly respond to the user.



1. Gathering Requirements

In your IBM Cloud account you should have the following

- Watson Assistant [instance](#).
- Language Translator [instance](#).
- Cloud Functions [instance](#)

On your local computer you should have the following

- [IBM Cloud CLI](#)
- Download the files in the repository [here](#) which includes the following:
 - polyglot.zip (Cloud Function code and dependencies)
 - skill-polyglot-bot.json (the Watson assistant skill)

2. Create Cloud Function

Cloud Functions is a serverless programming platform for developing code that scalably executes on demand. We will be leveraging this tool because it integrates with Watson Assistant via a Webhook and carries low costs.

- 2.1. Create a Cloud Function and note the name
- 2.2. Login to your IBM Cloud from the CLI using the following command:
 - ***ibmcloud login –sso***
- Choose the target space:
 - ***ibmcloud target --cf***
- Place your polyglot.zip file in its own folder on your computer where you can find it.
- Using IBM Cloud CLI Locate the folder containing your zip file (polyglot.zip).
- From within the folder, create your cloud function with the following command:
 - ***ibmcloud fn action create [your cloud function name here] polyglot.zip --kind nodejs:10***
- You can always update this file later with the following command:
 - ***ibmcloud fn action update [your cloud function name here] polyglot.zip --kind nodejs:10***

Action	CLI Command
Login to IBM Cloud via CLI	<code>ibmcloud login –sso</code>
Choose the target space	<code>ibmcloud target --cf</code>
Create CF Action	<code>ibmcloud fn action create [your cloud function name here] polyglot.zip --kind nodejs:10</code>
Update CF Action	<code>ibmcloud fn action update [your cloud function name here] polyglot.zip --kind nodejs:10</code>

CLI command summary

3. Create a new Watson Assistant skill

This Watson assistant skill will be responsible for serving initial language and or target skill choices to the user as well as setting and passing pertinent variables to the cloud function.

- Create a Watson Assistant instance if you do not have one
- If you haven't, download the dialog skill "polyglot-bot.json" from [here](#).
- Launch your Watson Assistant Instance
- Navigate to the Skills page (vertical menu on left, bottom icon) & Click "Create Skill"

- Select skill type, "Dialog Skill" and Click "Next"
- Choose "Import Skill" and select: "Choose JSON File"
- Navigate to the "polyglot-bot.json" file that you downloaded and click "Import"
- Attach the skill to an Assistant. To create an assistant go to the Assistants page (Vertical menu on left, top icon). Once created, link it to the skill.

4. Get the Cloud Functions endpoint for your webhook

- 4.1. Open your Cloud Functions dashboard here: <https://cloud.ibm.com/functions/>
- 4.2. Select the namespace you wish to deploy the function in from the drop-down, then click "Actions" on the left nav bar

The screenshot shows the IBM Cloud Functions dashboard. On the left, there's a sidebar with a 'Functions' icon and several options: Getting Started, Actions (which is highlighted with a red box and circled with a red number 2), Triggers, APIs, Monitor, Logs, and Namespace Settings. The main area has a heading 'IBM Cloud Functions-as-a-Service'. Below it, there's a section titled 'Run your app automatically' with a 'Download CLI' button. To the right, there's a 'What's New' section. At the top, it says 'Current Namespace: morganlanglais@ibm.com_my-test-spa...'. A dropdown menu is open under 'Current Namespace', showing 'ALL NAMESPACES (6)' and a list of namespaces. One namespace, 'morganlanglais@ibm.com_my-test-space Dallas (CF-Based)', is highlighted with a red box and circled with a red number 1.

- 4.3. Click my-action or the name that you previously gave your action in step 2]
- 4.4. Select Endpoints then check "Enable as web action" and click "Save"
- 4.5. Copy the URL in the "Web Action" section and save for later

The screenshot shows the 'Actions' settings for a function named 'main-zip'. On the left, there's a sidebar with 'Code', 'Parameters', 'Runtime', 'Endpoints' (which is selected and highlighted with a grey box), 'Connected Triggers', 'Enclosing Sequences', and 'Logs'. The main area has a 'Web Action' section. It includes a checkbox for 'Enable as Web Action' which is checked, and a note explaining it allows handling HTTP events. There's also an unchecked checkbox for 'Raw HTTP handling'. Below that, there's a table for 'HTTP Method' with rows for 'ANY' (Auth: Public) and a URL field containing 'https://us-south.functions.cloud.ibm.com/api/v1/web/morganlanglais%40ibm.com_my-test-space/default/main-zip'. The URL field is highlighted with a red box.

5. Add the webhook to your skill

5.1. Open the skill you imported in Watson Assistant earlier

5.2. Navigate to **Options -> Webhooks**

The screenshot shows the 'WA Translator' skill configuration. On the left, a sidebar lists 'Intents', 'Entities', 'Dialog', 'Options', and 'Webhooks'. The 'Webhooks' section is selected and expanded, showing 'Webhook setup'. Under 'Webhook setup', there is a 'URL' input field containing 'https://us-south.functions.cloud.ibm.com/api/v1/web/morganlanglais%40ib'. This URL is highlighted with a red box. Below the URL is a 'Headers' section with a note about adding HTTP headers for authorization.

5.3. Replace this URL with your webhook URL that you copied previously

Important: Make sure you append ".json" to the end of this URL in order for Watson Assistant to process the data correctly.

Done! You are now ready to configure your entities and dialog nodes.

6. Entity configuration

All the configuration values are loaded as entities enabling flexibility and ease of implementation.

The screenshot shows the 'Polyglot Bot' skill configuration. On the left, a sidebar lists 'Intents', 'Entities', 'My entities', 'System entities', 'Dialog', 'Options', 'Analytics', 'Versions', and 'Content Catalog'. The 'My entities' tab is selected and expanded, showing a table of configuration entities. The table has columns for 'Entity', 'Values', and 'Modified'. The entities listed are: '@assistantId' (modified 7 days ago), '@assistanturl' (modified 7 days ago), '@language' (modified a month ago), '@lt_apikey' (modified 7 days ago), '@lt_url' (modified 7 days ago), '@lt_version' (modified 7 days ago), '@wa_apikey' (modified 7 days ago), and '@wa_version' (modified 7 days ago).

Entity	Values	Modified
@assistantId	90bc7c04-6cd6-439b-b1ed-c849855d88f8, 3e3ebc0c-69ea-4d45-bc5f...	7 days ago
@assistanturl	https://api.us-east.assistant.watson.cloud.ibm.com, https://gateway.wats...	7 days ago
@language	hu, da, de, el, en, es, et, fi, fr, ga, gu, he, hi, hr, cs, id, it, ja, ko, li, lv, ml, ms...	a month ago
@lt_apikey	eFOaOyIRXRggUTG0JPwzMICuO4JLiy5guQo_ip24ZFv9	7 days ago
@lt_url	https://gateway.watsonplatform.net/language-translator/api	7 days ago
@lt_version	2018-05-01	7 days ago
@wa_apikey	75viwWsIyGMF7epo7r9TQXtSV_RTQSXo8O1eYvjKsWE, 9G1SSSXrRtV-...	7 days ago
@wa_version	2020-04-01	7 days ago

Configuration is done in Assistant via entities

Assistant Id

- 6.1. The assistant Id is the id(s) associated with the target Assistant that the Polyglot Bot will be translating. Find it here in the settings for your Assistant. Also note the Assistant Url and API Key

The screenshot shows the Watson Assistant interface. On the left, there's a sidebar with icons for Assistants, Create assistant, Rename assistant, API details, Search skill, and Inactivity timeout. The main area is titled "CCFA_Assistant". It has sections for "Skills (1)" (CCFA General Skill), "Integrations (1)" (with a "Settings" button highlighted by a blue box), and "Delete". Below these are sections for "API details" and "Assistant details". Under "Assistant details", fields include "Assistant name: Polyglot Bot" and "Assistant ID: d1e0ccce-8295-4323-9899-38044de". The "Assistant URL" field contains a placeholder URL. Under "Service credentials", fields include "Credentials name: Auto-generated service credentials" and "API key: Y9hdrgwBweTrcTbqEDQ18aL_ft0g1V2tDSqogGW".

- 6.2. Copy the Assistant ID and then open the associated entity within the Polyglot Bot skill. From within the @assistantId entity add your Assistant ID in the Value field as shown below. Give the target assistant a name in the “Synonyms” field. Make the name something that uniquely identifies the Assistant such as “assistantId_covidbot” or “assistantId_departmentoftransportation” etc.

- 6.3. If you will be routing to multiple assistants then be sure to uniquely identify each target skill. Use the same naming convention for the API Key

The screenshot shows the Polyglot Bot skill configuration. A search bar at the top has the text "@assistantId". Below it is an "Entity name" field with the placeholder "Name your entity to match the category of values that it will detect." An input field contains the value "@assistantId". At the bottom, there are "Value" and "Synonyms" fields. The "Value" field contains the ID "d5f364-e2c7-4840jlkjlkj-b78d-bb0dc2f7". The "Synonyms" field contains "assistantId_target_skill". Buttons at the bottom include "Add value" and "Recommend synonyms".

Assistant URL

6.4. Similarly, find the url also in the settings for your target assistant(s) and paste it in the associated entity if it does not already exist. Note that you only need the core domain including everything up to but ending with .com or .net. Similar to what is shown here. Be sure to name it with a uniquely identifying name in the “Synonym” field.

The screenshot shows the Entity Settings page for an entity named '@assistanturl'. The 'Entity name' field contains '@assistanturl'. Below it, there's a 'Value' input field with placeholder text 'Type a value, e.g. Checking' and a 'Synonyms' dropdown. A 'Type a synonym, e.g. Deposit' input field and a '+' button are also present. At the bottom, there are 'Add value' and 'Recommend synonyms' buttons, and tabs for 'Dictionary (2)', 'Annotation (0)', and 'Beta'. The 'Dictionary' tab is selected, showing two entries:

Values (2) ↑	Type
<input type="checkbox"/> https://api.us-eastassistant.watson.cloud.ibm.com	Synonyms
<input type="checkbox"/> https://gateway.watsonplatform.net/assistant/api/	Synonyms

Languages

6.5. All the languages as of 2020 are already loaded, there is nothing to do here but it is important to open the language entity up to understand the naming convention as you will reference it later. The format is [language] + “_language” as shown in the right column below. To see a full listing of available languages: <https://cloud.ibm.com/docs/language-translator?topic=language-translator-translation-models>.

The screenshot shows the Entity Settings page for an entity named '@language'. The 'Dictionary' tab is selected, showing 46 entries. The table structure is identical to the one above:

Values (46) ↑	Type
<input type="checkbox"/> ar	Synonyms
<input type="checkbox"/> bg	Synonyms
<input type="checkbox"/> bn	Synonyms
<input type="checkbox"/> cs	Synonyms

Language Translator

Go to your language translator instance to find the following values, see image below. Place them into the corresponding entity values.

- 6.6. **lt_apikey** entity is where you will load the apikey obtained from your language translator instance. Go to the instance and find the info on the intro page.
- 6.7. **lt_url**: Also load the lt_url entity with the url info below (as with Assistant only use the base domain, http to .net)
- 6.8. **lt_version**: As of this writing the most recent version is 2018-05-01, you can check here if it has been updated: <https://cloud.ibm.com/apidocs/language-translator#versioning>. If it hasn't just leave it alone.

The screenshot shows the IBM Watson Language Translator instance. At the top, there's a navigation bar with 'Resource list / Language_Translator' and status indicators ('Active' with a green dot, 'Add tags'). Below this is a 'Manage' sidebar with links: 'Getting started', 'Service credentials', 'Plan', and 'Connections'. The main content area has a 'Start by viewing the tutorial' section with 'Getting started tutorial' and 'API reference' buttons. Below this is a 'Credentials' section. It shows an 'API key:' field containing a redacted value, with 'Download' and 'Show credentials' buttons. An 'URL:' field contains 'https://gateway.watsonplatform.net/language-translator/api'. A link 'View all credentials in the Service credentials tab.' is also present.

7. Dialog configuration

Now that the entity values are loaded it's time to set up the dialog node.

7.1. Navigate to the dialog page and open the “Choose Language and or Target” node

The screenshot shows the Dialog configuration page for a bot named 'Polyglot Bot'. On the left, a sidebar lists sections: 'Intents', 'Entities', 'Dialog' (which is selected and highlighted in blue), 'Options', 'Analytics', 'Versions', and 'Content Catalog'. At the top right are three buttons: 'Add node' (blue), 'Add child node' (grey), and 'Add folder' (black). The main area displays a tree structure under the 'Dialog' section. The root node is 'Choose Language and or Target' with the ID 'welcome'. It has two children: 'Welcome' with the ID '#welcome_welcome' and 'Send to Translation Webhook' with the ID '\$language'. Each node has a description, a count of responses and context sets, and a 'More' button (three dots) to its right. To the left of the tree, there are two vertical arrows indicating nesting levels.

7.2. Scroll down to the Options list. Here is where your user will get a list of available languages for selection. The node is pre-loaded with a number of sample languages. You can delete/add the ones needed as follows...

List label	Value
1 English	English_language; wa_apikey_beta1
2 Espanol	Spanish_language; wa_apikey_beta
3 Hindi	Hindi_language; wa_apikey_betains
4 Japanese	Japanese_language; wa_apikey_be

Assistant responds

Option		
Title	POLYGLOT BOT please choose Language/Idioma/I	Description (optional) Add description
List label	Value	
1 English	English_language; wa_apikey_beta1	
2 Espanol	Spanish_language; wa_apikey_beta	
3 Hindi	Hindi_language; wa_apikey_betains	

7.3. Click in the “List label” box with the label you would like to change. What you enter here is exactly what the user will see. If, for example you wanted the user to be able to choose Hindi as their language choice you would enter that here. Next move to the “Value” field. Here is what gets returned when the user selects that list item. The corresponding Value field is what gets sent to Watson Assistant to trigger the configurations set in the entities. As example, here is a detailed list of each semicolon separated value:

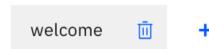
- Hindi_language; wa_apikey_betainstance; assistantId_multiskill; assistanturl_south; wa_ver_202004; lt_version; lt_url, lt_apikey

- These trigger values must match **exactly** what you used as values in your entity synonyms. None of them can be omitted or the cloud function will not work. In this example above, when the user presses “Hindi” it triggers the following:

Value	Significance
Hindi_language	Selects the Hindi value in the @language entity
wa_apikey_betainstance	Triggers the correct api key for the target Assistant
assistantId_multiskill	Triggers the correct Assistant ID for the @assistantId entity
assistanturl_south	Triggers the url associated with the target Assistant
wa_ver_202004	Triggers the version for the target Assistant
lt_version	Triggers the version for the language translation instance
lt_url	Triggers the version for the language translation instance
lt_apikey	Triggers the api key for the @lt_apikey intent.

- In this example, this item in the drop down list allows the user to select Hindi, it then configures the webhook request to target an Assistant called “assistantId_multiskill” that was deployed in Dallas (assistanturl_south) and serve that Assistant to the user in the Hindi language.
- You will want to review each value. If all you are doing is offering the same target instance in multiple languages, you can copy and paste all the subsequent fields just change the language values for each option and delete all the options that aren’t necessary.
- You will not be able to see all the values in the window so you can either work in the constrained box, build the configuration in a separate document and paste them in or if you feel comfortable you can open the json editor and work with them directly there.

If assistant recognizes



Then set context

Variable	Value	⋮
session_id	0	Open JSON editor Close context edit...

8. Welcome

If you are targeting another Assistant (not the one connected the Polyglot Bot skill) there will need to be some minor additions to that target skill.

8.1. Add a #welcome_welcome intent by going to the intents tab and pressing “Create Intent”. Set it up exactly as below:

← | #welcome_welcome

Intent name
Name your intent to match a customer's question or goal

Description (optional)
Add a description to this intent

User example
Add unique examples of what the user might say. (Pro tip: Add at least 5 unique examples to help Watson understand)
Type a user example here, e.g. I want to pay my credit card bill

Add example

User examples (1) ↑

welcome_welcome

- 8.2. Then add a corresponding welcome_welcome dialog node that triggers off of the intent. Navigate to the Dialog page and click “Add node” and configure the trigger as below with “If assistant recognizes: #welcome_welcome

Polyglot Bot

Intents Entities Dialog Options Analytics Versions Content Catalog

Add node Add child node Add folder

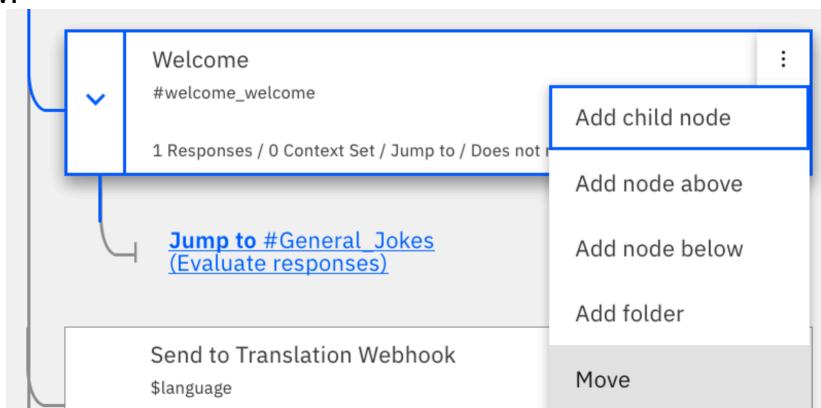
Welcome

If assistant recognizes

#welcome_welcome

Assistant responds

- 8.3. This node will need to replace the current welcome node so alternatively you can just change the existing welcome node to also trigger off of #welcome_welcome by clicking the “+” next to the current trigger and adding the condition. If you add a new #welcome_welcome node be sure to move it so that it is above the current welcome node. To do this use the “Move” option as shown below.



Congratulations, you are done!