Create a Simple Language Translator using Watson Language Translator, Slack and the IBM Cloud

James Belton
Watson and Cloud Adoption Leader
February 2018

Introduction

This guide provides the steps required to build a simple language translator using Watson Language Translator and Slack. Slack is an application used by teams to communicate; how about adding a translator to it, allowing multi-language teams to chat?

This is a good introduction to Watson Language Translator and also a good introduction to Node-Red, the IBM Cloud in general and how it can be used in conjunction with other common business applications, such as Slack.

Requirements

To build this, you will need and IBM Cloud account. Sign up for one, free of charge at https://console.bluemix.com if you don't already have one. You can complete this with a Lite account. Once you have your account, make sure that you have created a space in your organization (Click Manage -> Account -> Cloud Foundry Orgs. Then click on the name of your Organisation and click Add a Cloud Foundry Space).

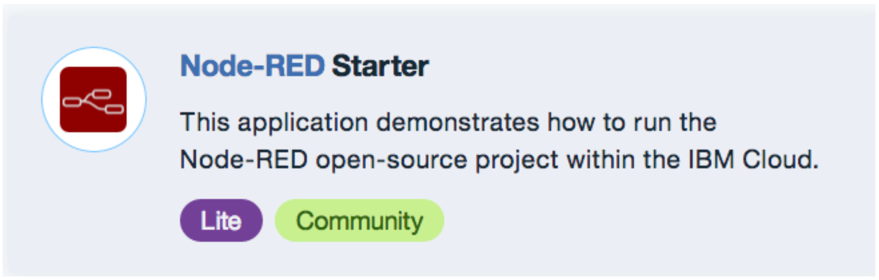
You will also need a Slack account, sign up for one at www.slack.com.

Part One – Building with Node-Red

This application is built in two parts. In this part, you will create several Node-Red flows, one for each language you'd like to translate. The second part will concentrate on configuring Slack.

Provision the Service

Once you have your IBM Cloud account up and running, you can start building the Node-Red elements or Flows. The first task is to provision a Node-Red Starter instance (Note, if you already have a Node-Red starter provisioned, you can use that instance and skip these steps, just access your existing Node-Red editor and use a new Flow Window).
From the IBM Cloud, Click Catalog and then start to type 'Node-Red Starter' into the search bar. Click the service icon:

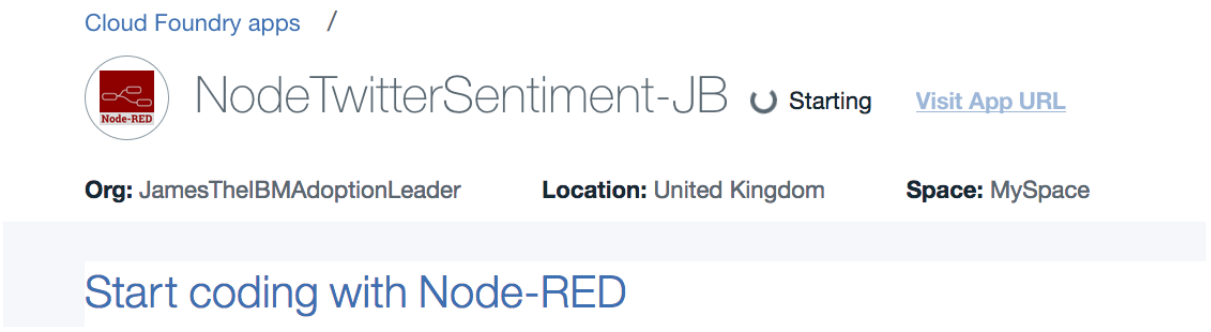


This application demonstrates how to run the Node-RED open-source project within the IBM Cloud.

On the next page, enter a Name for your application. This will need to be unique across the whole of the IBM Cloud (because this forms part of the URL from which your application will be available), so choose something like 'NodeTranslate-XX', where XX are your initials.

Check that the region and space drop-downs are populated (this is where the instance will be created) and then click the 'Create' button. The service will now provision. This may take a few moments but shouldn't take too long to complete.

You should then see a page which looks similar this, note the 'spinning' Starting icon means that the service is still provisioning.


Cloud Foundry apps /

NodeTwitterSentiment-JB Starting Visit App URL

Org: JamesTheIBMAdoptionLeader **Location:** United Kingdom **Space:** MySpace

Start coding with Node-RED

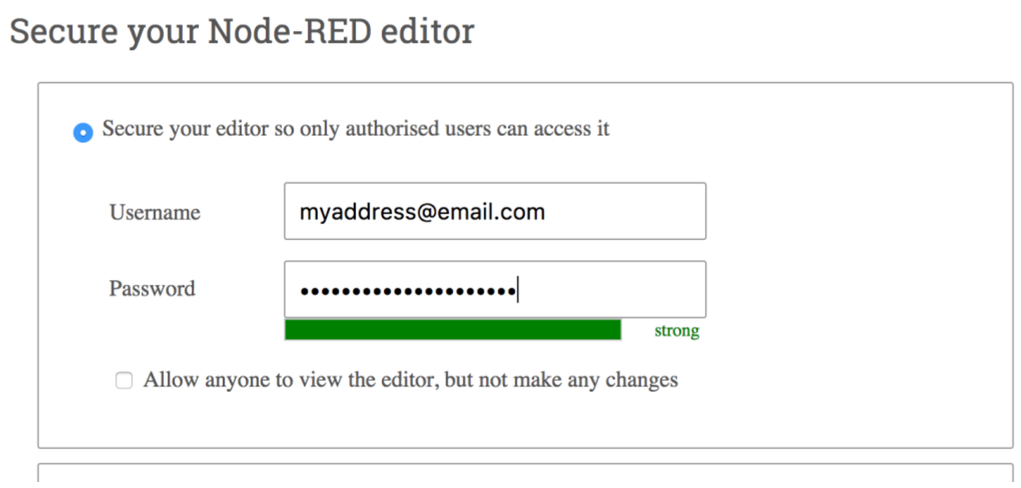
At this point, head to your Dashboard – click the hamburger icon towards the top left of the screen (the three horizontal lines next to the IBM Cloud Logo) and then click Dashboard.

Here, you will see that you have two services provisioned. One is a Cloud Foundry App (this is the Node-Red instance) and the other is a Cloud Foundry Service – a Cloudant NoSQL Database.

Click the URL under 'Route' for your Cloud Foundry app and this will take you to your Node-Red editor.

You should then see a Welcome page, with a couple of bullets about securing your editor and browsing available IBM Cloud Nodes. Click Next at the bottom of the page.

The first step is to secure your editor, so that unauthorized persons cannot access it. Here, you need to provide a username (try your email address) and a password. Make this a strong password (i.e. over 8 characters long and a mix of letter case, numbers and symbols) – I'd also recommend making it unique to this site. Leave the check boxes unchecked.


Secure your Node-RED editor

[✓ Secure your editor so only authorised users can access it]

Username: myaddress@email.com

Password: ••••••••••••••••••••••

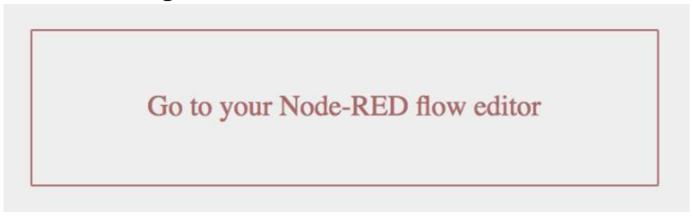
strong

☐ Allow anyone to view the editor, but not make any changes

Click Next.

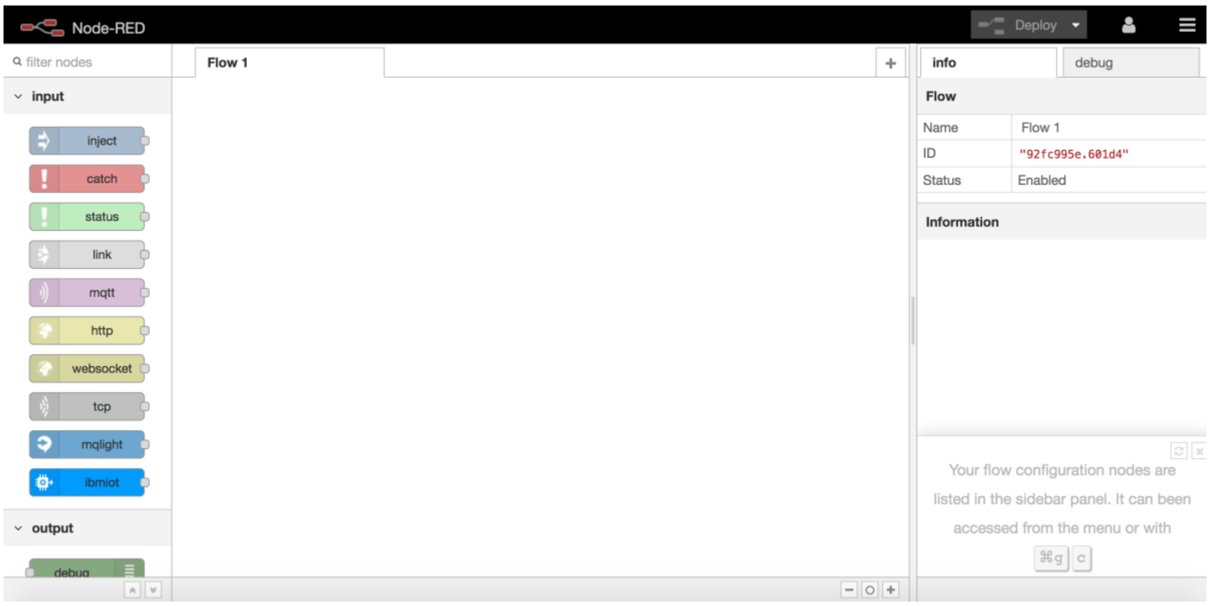
The following screen shows a list of other nodes that you can attach to your Node-Red instance from IBM Cloud but we don't need any of these for this example, so just click next. You'll then get a 'Finish the Install' screen, at which point you can just click 'Finish'.

On the next screen, click the 'Go to your Node-Red Flow Editor' button and we can then start building some flows.



You'll be asked to log into your flow editor, using the credentials that you just supplied.

Once you are connected to the editor, you will see a screen like the following:



Flow	
Name	Flow 1
ID	"92fc995e.601d4"
Status	Enabled

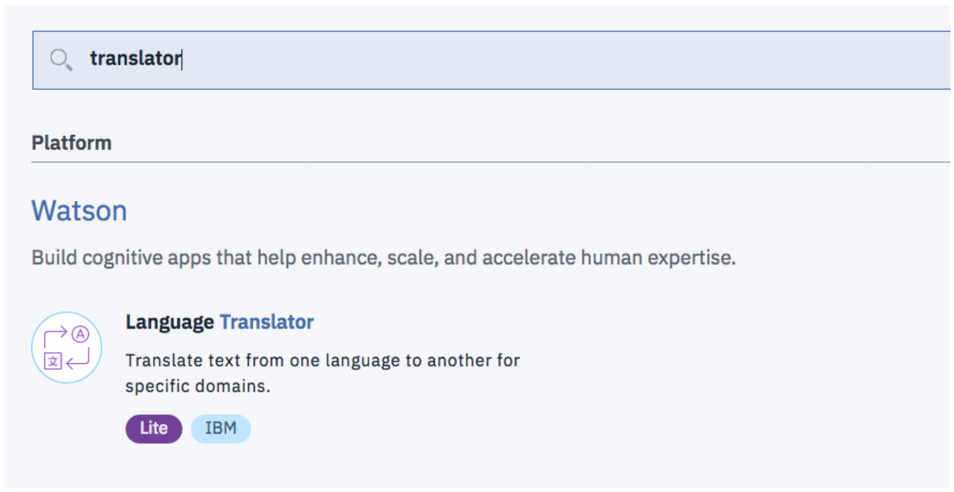
Information

Your flow configuration nodes are listed in the sidebar panel. It can been accessed from the menu or with ⌘g c

In the centre, you have your Flow window. This is where you build your flows. Down the left-hand side, you have nodes which can be dragged into the Flow window. On the right-hand side is the Info window (which provides some documentation) or depending on which tab is active, the Debug window, which will show you the output from your flows.

Create and attach an instance of Watson Translator

From the IBM Cloud console, click catalog. In the search bar, type 'Translator'.


translator

Platform

Watson

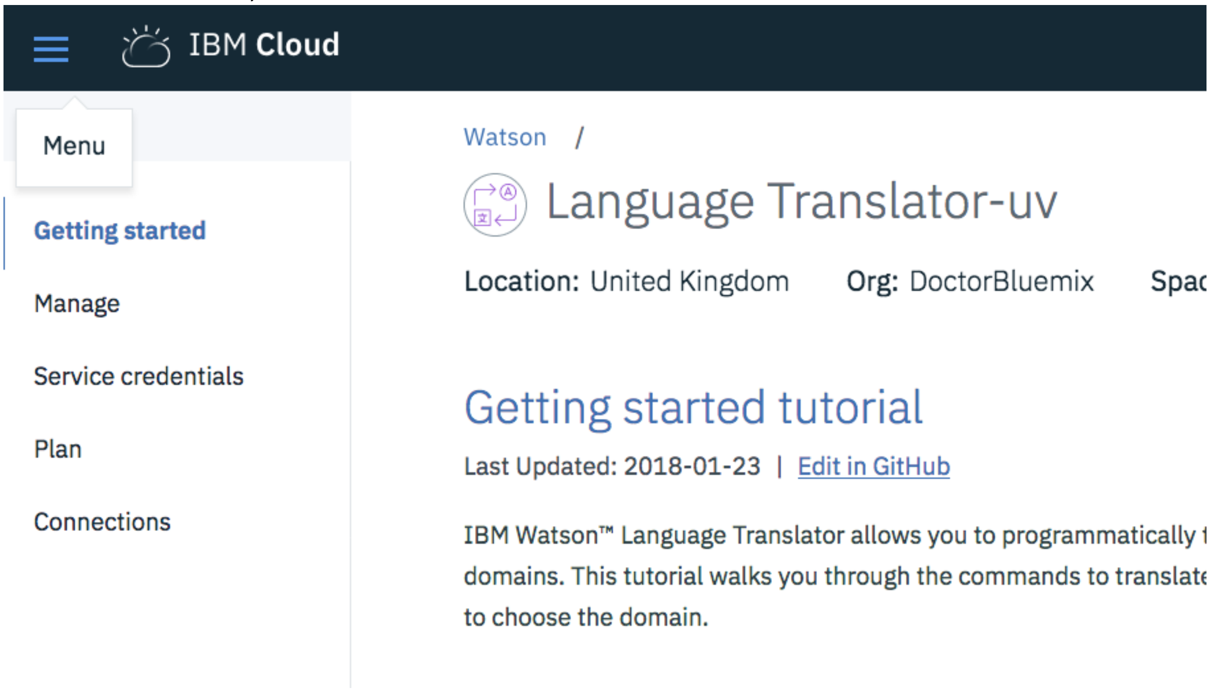
Build cognitive apps that help enhance, scale, and accelerate human expertise.

Language Translator

Translate text from one language to another for specific domains.

Click on Language Translator. On the next page, set the Service name and select your plan (the Lite plan is free of charge and is fine for this demo) and then click 'Create'.

On the next screen, click Connections from the left-hand menu.



Language Translator-uv

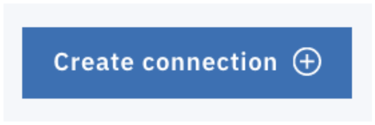
Location: United Kingdom Org: DoctorBluemix Spac

Getting started tutorial

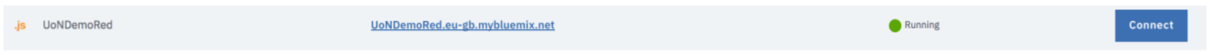
Last Updated: 2018-01-23 | Edit in GitHub

IBM Watson™ Language Translator allows you to programmatically t
domains. This tutorial walks you through the commands to translate
to choose the domain.

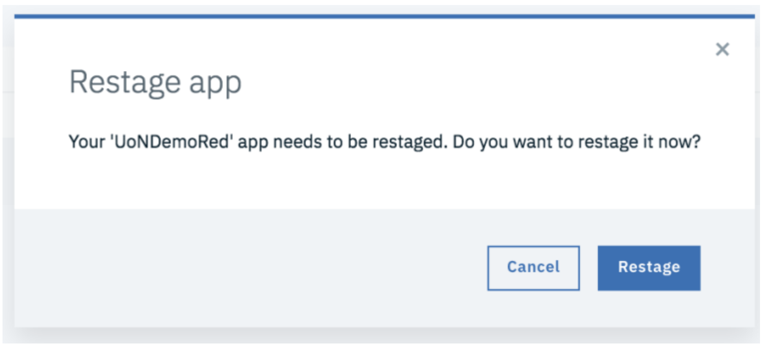
Then click Create Connection.



Look down the list of connectable applications for your Node Red instance. Click the Connect button next to it. For example:



You may then see a panel asking you to restage the application. Click Restage.


Restage app

Your 'UoNDemoRed' app needs to be restaged. Do you want to restage it now?

Your Node-Red application will restage and your Language Translation service will be configured so that you do not need to provide credentials with your Node-Red application to use it.

We're now ready to create the app.

Creating and configuring the First Flow

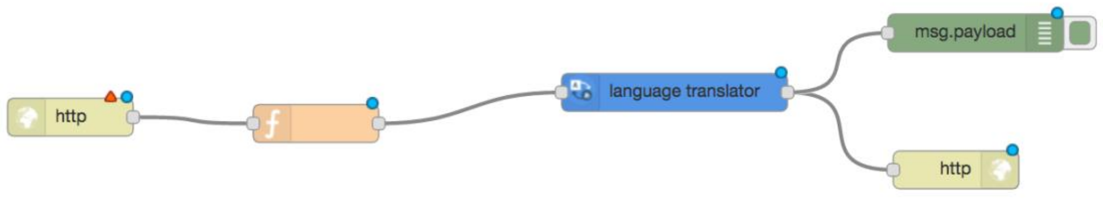
If your Node-Red editor isn't currently open, access it from your IBM Cloud dashboard by clicking the 'hamburger' icon in the top left corner (three horizontal lines next to the IBM Cloud logo) and select Dashboard. Then click the URL link next to your Node-Red Service. You may need to provide the Node-Red editor credentials that you created in earlier steps.

So, let's create our first Node-Red flow. This will be a flow that takes input from Slack, translates that input to the given language and provides the translation as a response.

From the left-hand window, drag one by one into the Flow window:

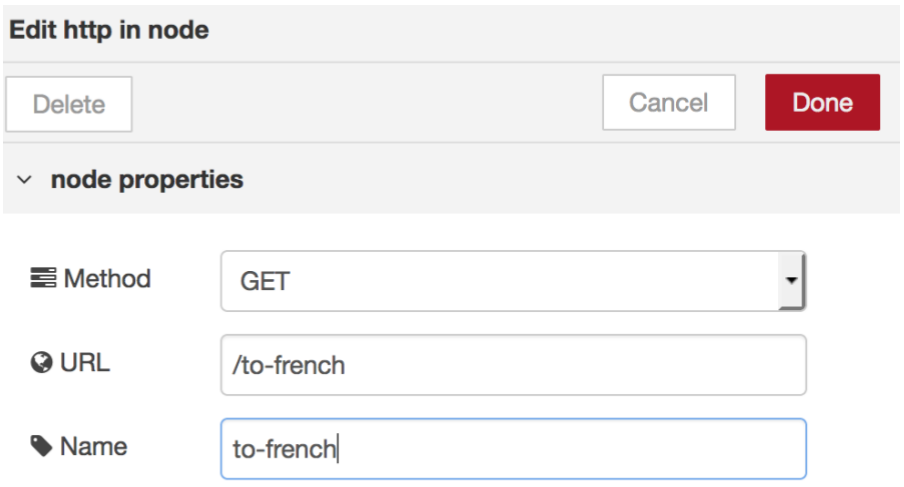
- An Http In node (found under input)
- A function node (found under function)
- A Language Translator node (found under IBM Watson)
- An Http Response node (found under Output)
- A debug node (found under Output)

Next, join the Http In node to the Function node by clicking the grey box on the Http In node's right side and dragging it to the grey box on the left side of the Function node. Join the Function node to the Language Translator node in the same way and then join the Language Translator node to the Http Response node and the Output node. Your Flow window should now look something similar to this:



We're going to configure this flow to translate English to French.

First, configure the Http In node. Double click on it to bring up its Edit pane. Set the URL to /to-french and make sure the method is set to GET.



Method: GET
URL: /to-french
Name: to-french

Click the red Done button when ready.

Now, double click the Function node and type 'MyFunc' into the name box.

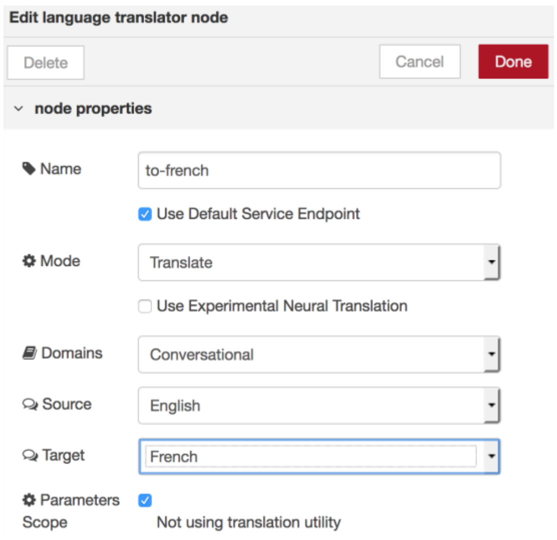
At this point, you'll need to enter some code. Type in the following (make sure that the `return msg;` line only appears once):

```
msg.payload = msg.payload.text
return msg;
```

When completed, click 'Done'

Next, double click the Language Translator node.

Configure this node as follows:

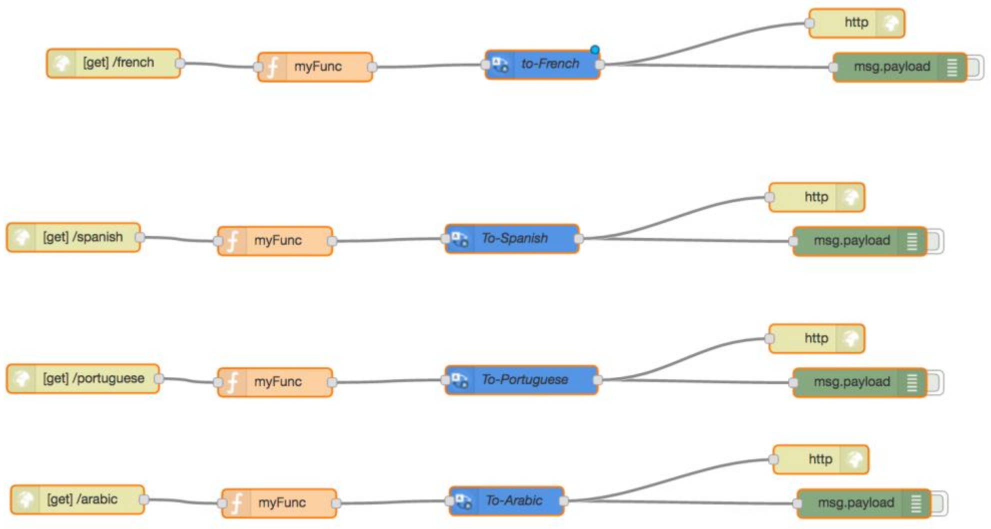


Name: to-french
☑ Use Default Service Endpoint
Mode: Translate
☐ Use Experimental Neural Translation
Domains: Conversational
Source: English
Target: French
Parameters Scope: ☑ Not using translation utility

You don't need to do anything to configure the debug node or the Http Response node. While the debug node isn't essential, it's a good habit to include them, in case you do need to debug a non-working flow.

If you want to set up translators for the other Conversation languages (Spanish, Portuguese and Arabic), simply repeat the steps above but change the URL for the GET Http In node (e.g. so that It's /to-arabic for the Arabic flow and so on) and change the configuration of the Language Translator node so that the Target drop down is set to the appropriate language too.

If you set up flows for each of the available languages, you'll see something like this in your flow editor:



When complete, click the Deploy button (which should now be red) at the top of the screen. This will save and deploy your flow.

Note the 'Get' URL – you'll need it later

Now, make a note of the URL that your Slack installation is going to call to get the data. This is simply the main part the URL that you see in your Node-Red browser when you are editing your nodes, appended with the /get node's URL setting.

So, for example, the URL shown by the browser may be
https://myNodeRed.eu-gb.mybluemix.net/red/#flow/dbdd0e5f.35b5a
and the /get node URL is /french

Using these examples, the URL that you need to access that /get request for a French translation will be:

https://myNodeRed.eu-gb.mybluemix.net/french

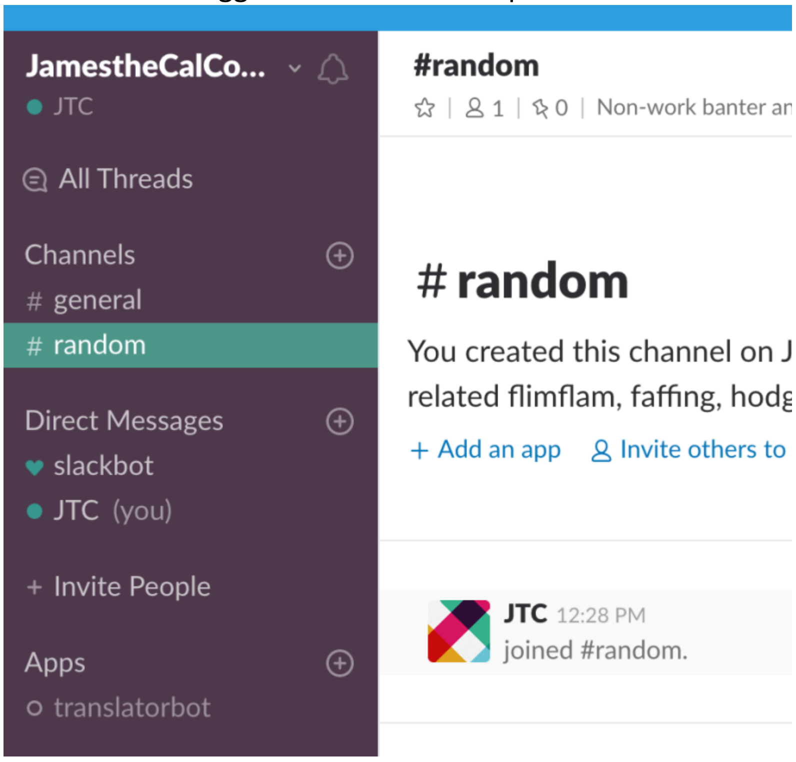
Part Two – Connect Slack to your Translator

In the second part of this walk-through, we'll be working in Slack. Log into the Slack instance that you have created. Note that if you are using a Corporate Slack instance, you may not have the rights to alter that instance to add your translator.
The way this will work is that if you have a phrase that you want to translate, you will first enter a 'slash command' and then the phrase to translate. So for example, entering '/French hello' will send a request to your Node-red flow to translate 'Hello' into French and return 'Bonjour' to the Slack pane.

Node-Red to do the bulk of the work.

Configuring Slack

You should be logged into Slack at this point and see a screen similar to this:

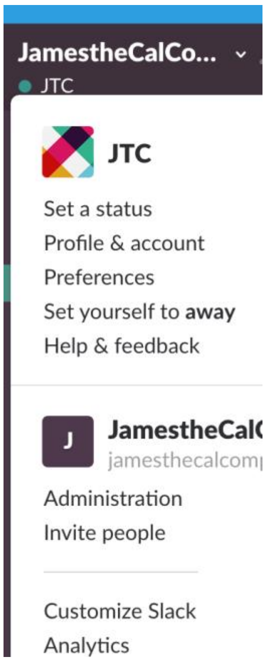


random

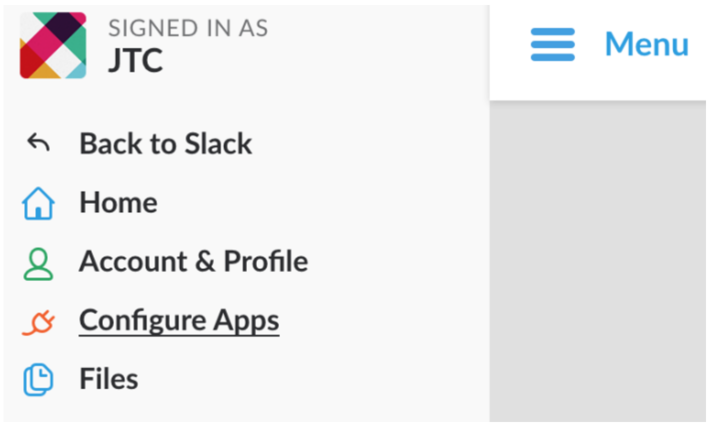
You created this channel on J
related flimflam, faffing, hodg

JTC 12:28 PM
joined #random.

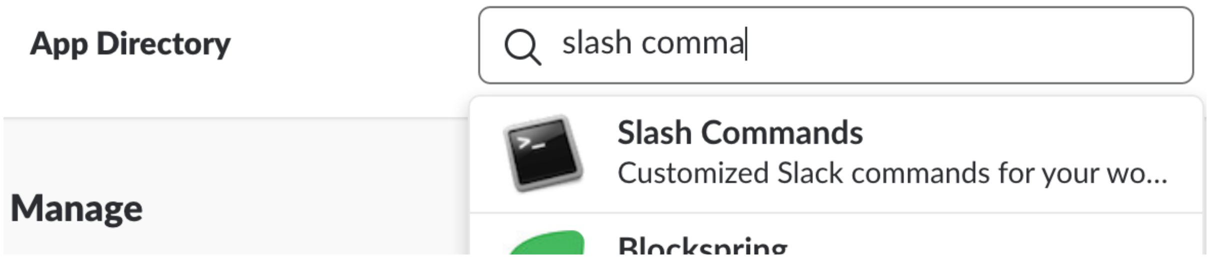
Click the drop-down tab next to the organization name and then click 'Customize Slack'



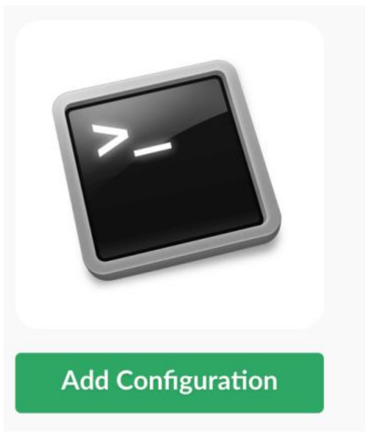
A new screen will display. Click the hamburger in the top left corner (the three horizontal lines) to open a menu to the left and then click 'Configure Apps'



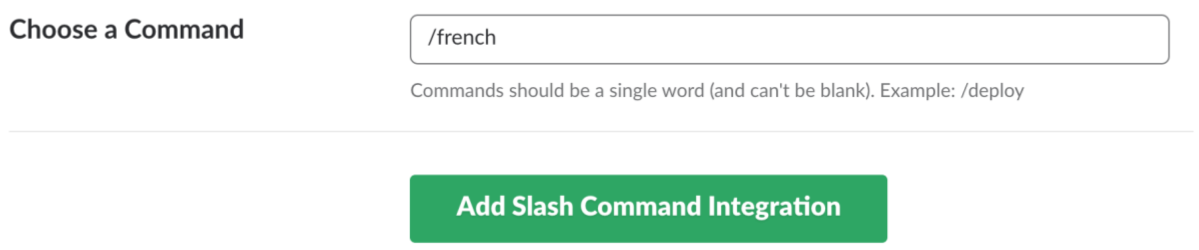
On the next screen, type 'slash commands' into the search bar and select it from the menu:



Click the Add Configuration button:



For this example, we'll set up an English to French translator. In the Choose a Command box, enter '/french' and click the green button:



Choose a Command

/french

Commands should be a single word (and can't be blank). Example: /deploy

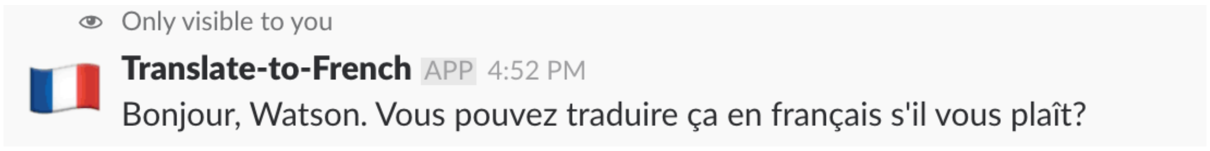
On the next screen, scroll down and set the URL to the URL of your /get-french http in node – see the section above called '*Note the 'Get' URL – You'll need it Later*'. Also, set the Method to GET from the drop down.
Change the Customize Name field to 'French Translator'. You can also click the 'Choose the Emoji' button and find a French flag icon.
Then scroll to the bottom and click Save Integration.

And that's all you need to do.

Go back to your Slack conversation pane and test it out by typing in:

/French Hello, Watson. Can you translate this to French please?

You should see the following output:



Only visible to you

Translate-to-French APP 4:52 PM
Bonjour, Watson. Vous pouvez traduire ça en français s'il vous plaît?

Repeat the steps above to set up the other languages, changing the /command and the URL accordingly.

And there you have it, a multi-language Slack-enabled translation service!