# Lab 1 overview

This lab shows you how to launch your own instance of Node-RED by using the IBM Cloud boilerplate. You can deploy Node-RED as a stand-alone application, or flow, but using the Cloud platform as a service means that you don't need to worry about installing Node-RED prerequisites.

**Prerequisites**
You need an IBM Cloud account to complete these lab exercises. You can register for an account @ IBM Cloud registration

**Important**: There are limitations on the use of some services with the free Lite Plans and you may see warning messages as you progress through the exercises. You can use only 12 free services in IBM Cloud and, in this course, you will need 6 services. So, if you already have an IBM Cloud account, be sure you are not using more than 6 services. Please check the service within IBM Cloud for usage on Lite Plans. This exercise has only been tested using Chrome or Firefox as the browser. Some people have reported problems in Lab 3 using **Internet Explorer**

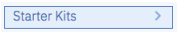
In these labs, you use the following services:

- Language Translator
- Tone Analyzer
- Watson Assistant
- Speech to Text
- Text to Speech
- Cloudant database

## Create a Node-RED instance

In this step, you will create an instance of Node-RED running on IBM Cloud.

1. Log in to IBM Cloud.
2. Click **Catalog**.
3. Click on **Starter Kits**, to list.



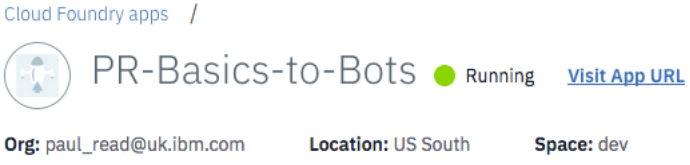

4. From the Starter Kits list, click **Node-RED Starter**.



If you are using an IBM Cloud Lite account, you need to choose the **Internet of Things Platform Starter**, instead.

5. Give your starter application a unique name and host name. The host name does not need to be the same as the app name, but it can be the same.


6. Select **Create**.




7. Wait for your new application to start.

## Connect Watson services

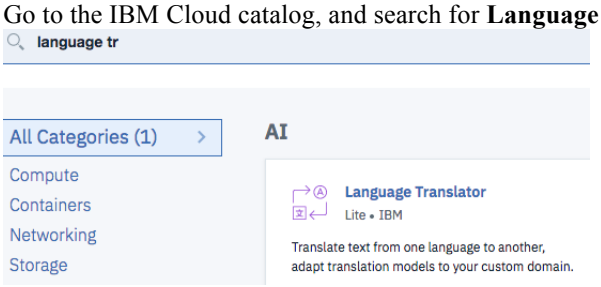
In this step, you'll connect the following Watson services to your Node-RED instance:

- Language Translator
- Watson Assistant (formerly Conversation)

You'll use the Language Translator service in this lab and the Watson Assistant services in Lab 3.

If you already have instances of these Watson services, you can connect them to your Node-RED application. (Skip to step 7.)

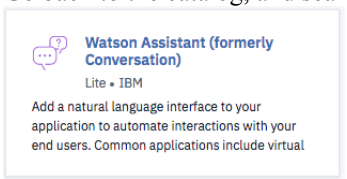
1. Go to the IBM Cloud catalog, and search for **Language Translator**.




2. Select the service and ensure that the region, organization, and space are the same as your Node-RED instance. You will only need the Lite plan for the labs in this course.


Click **Create** to create the service.

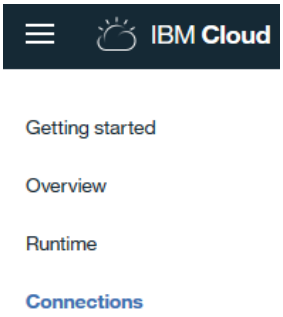
3. Go back to the catalog, and search for **Watson Assistant (formerly Conversation)**.




4. Select the service and ensure that the region, organization, and space are the same as your Node-RED instance and Language Translator service. Select the Lite plan and click **Create**.
The next step is to connect the Language Translator and Watson Assistant services to your Node-RED application. For each service that you create, credentials are automatically generated that allow you to use the service. By connecting the services to your Node-RED instance, the credentials are available for the application to use without you having to manually enter them.

5. From the IBM Cloud dashboard, click your Node-RED application under the name column.

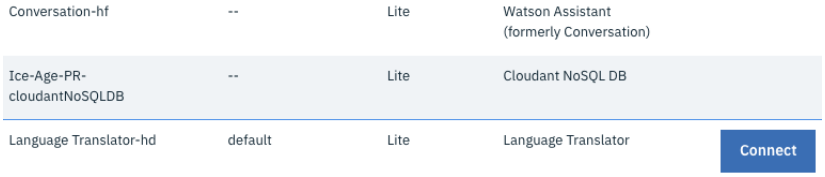
6. Click **Connections**.




7. Click **Create connection**.

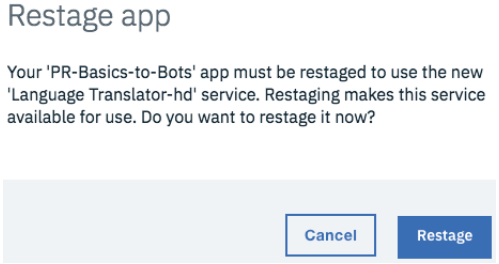

You should see the two services that you just created.

8. Hover over the Language Translator and click **Connect**.



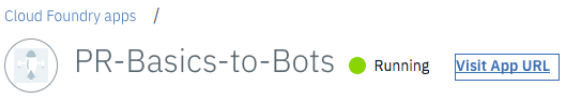

When prompted click **connect** without changing the Service Roles.

9. When you're prompted to restage, click **Cancel** because you want to connect to one more service before you restage.

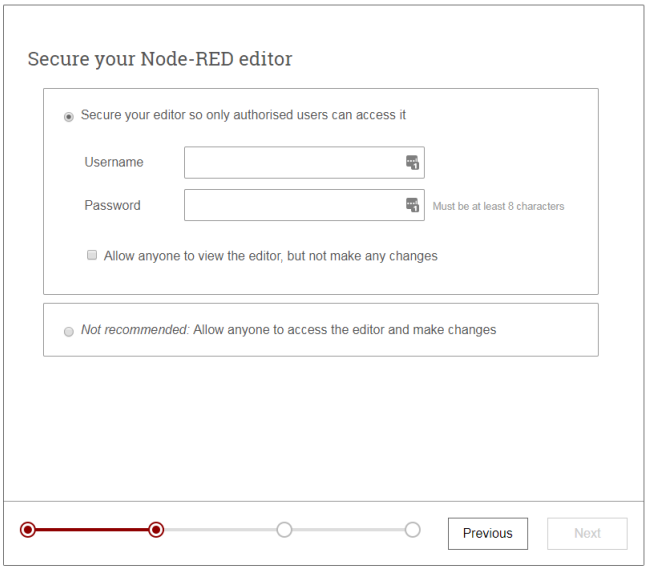



10. Click **Create connection** again and repeat the previous steps to add the Watson Assistant (Conversation) service. This time click **Restage**.

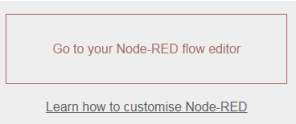
11. Wait for the application to restart. When your application has restarted, you will see the status reported as "Running." Click **Visit App URL** to open your running Node-RED application to see the landing page of your Node-RED instance.



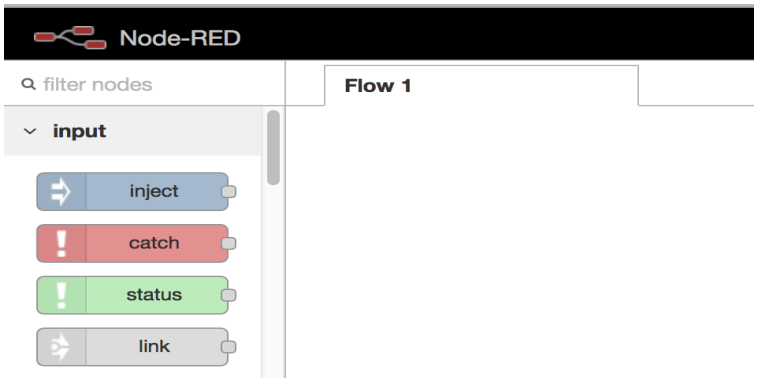

12. Follow the steps to secure your instance (which is advisable) by creating a user name and password.



13. Click **Go to your Node-RED flow editor** to open the editor.

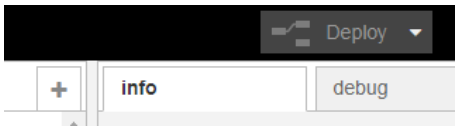



14. You should then be asked to login to your Node-RED instance using the credentials you used to secure your app.



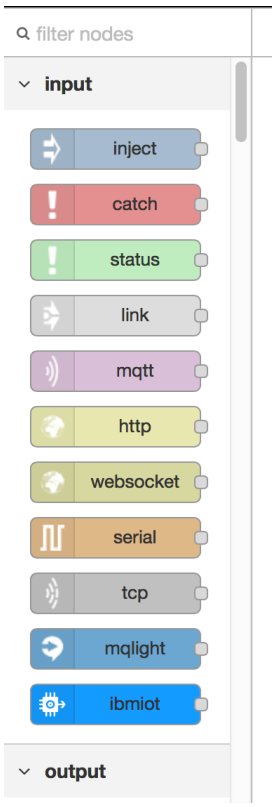

If you created an IoT Start application, a sample app is automatically created in your Node-RED instance., but you won't be using this.

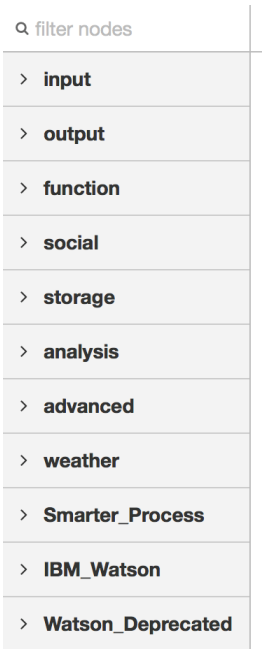
15. Click the **Add** icon (+) to create a new flow.



## Create your first flows **

In this section, you'll create your first flows in the Node-RED flow editor. An application in Node-RED is called a *flow*. The palette in the left column shows you all the available nodes, the sections can be collapsed or expanded by clicking on the arrow next to section title.

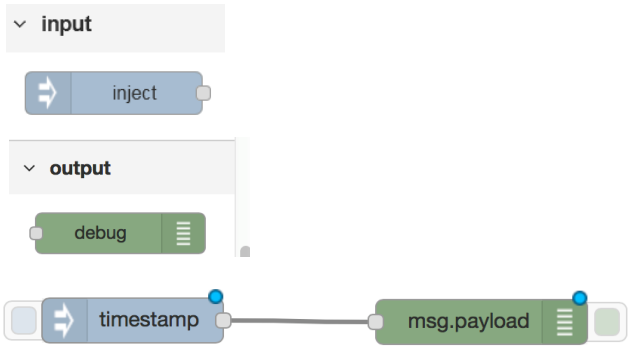



The nodes are grouped by category.

1. Select an input **inject** node and drag it onto the canvas.

2. Select an output **debug** node and drag it onto the canvas.

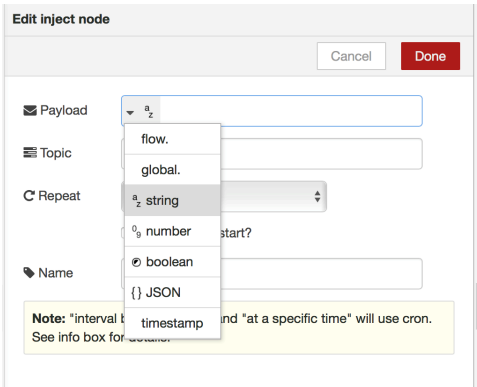


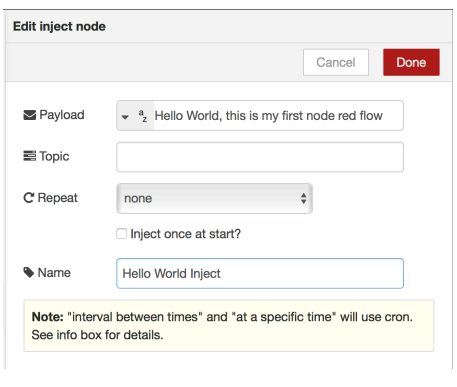

3. Link, or wire, the two nodes together by clicking and dragging your cursor from one node to the other. Note that the debug and inject nodes change their display names when you drag them onto the canvas. This name change is expected and shows additional context for the node.

4. Double-click the **timestamp** node. For the **Payload** field, select **string**.

   Enter a string, such as `Hello World, this is my first Node-RED application`. Then in the **Name** field, enter a name for this node, such as `Hello World inject`.

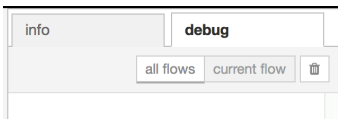
   Click **Done**.



5. The blue circles indicate that your flow has unsaved changes, which means that the application needs to be deployed.

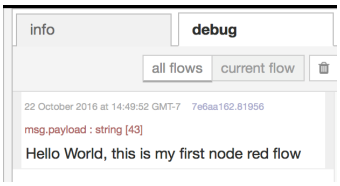

6. Click **Deploy** to deploy and save your changes.

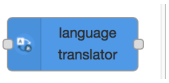

The **debug** node writes to the **debug** tab, which helps you monitor the flow through your application.

7. To initiate the flow, click the tab linked to the **inject** node.

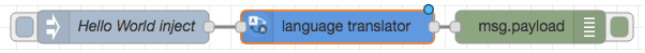
You now see the output on the debug tab.




8. In the **filter nodes** search field, enter `translator` to find the **language translator** node.

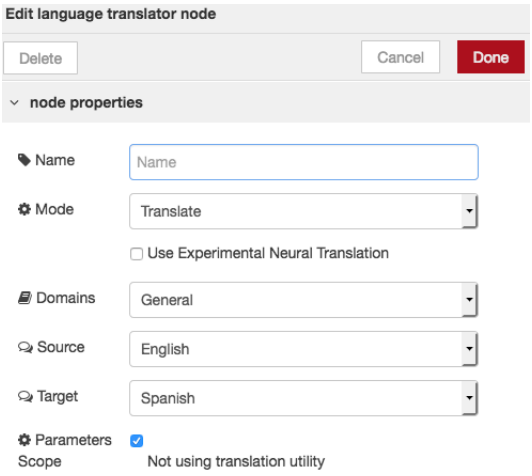



9. Drag the node onto the canvas so that it lies in between the **inject** and **debug** nodes. You can move the nodes to make more space. To remove an unwanted line, select the line and press Delete on your keyboard.

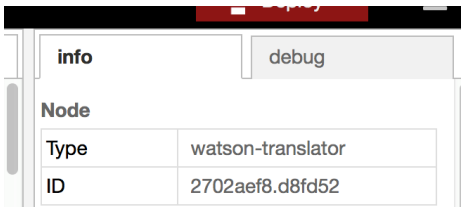



10. Double-click the **language translator** node.
Set Mode to **Translate**
Leave Domains set to **General**
Leave the Experimental Neural Translation blank.
By default, the source will be **English** and the target **Dutch,** however I have used Spanish as the target

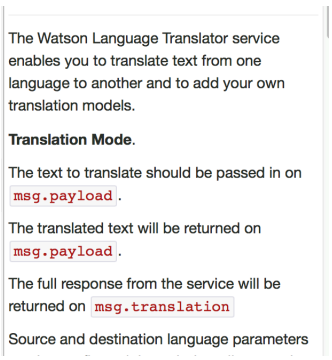
Click **Done** to save your changes.



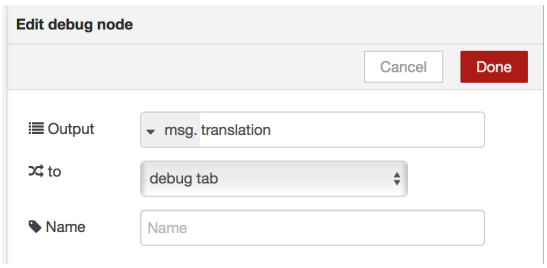

11. Select the **language translator** node and click the **info** tab.




Notice that the node puts its translated output in msg.translation. **msg** is a reserved object that Node-RED uses to allow individual nodes to communicate with each other. Think of **msg** as an envelope into which one node places information that allows another node to read it. The **language translator** node is expecting to find a payload that is already in the msg envelope, and it will insert a translation into the msg envelope.



12. Open the **debug** node and change the output to `msg.translation`. Enter the word `translation` after `msg`.

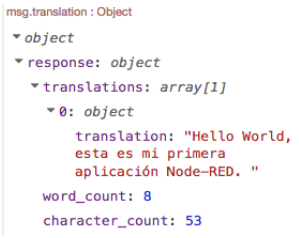



Click **Done** to save your changes. Then, **deploy** your flow.

13. Initiate the flow by clicking the tab on the **inject** node.

    View the translated text in the **debug** tab. The application is translating the text that you entered in the **Payload** field of the **inject** node.

    In most languages, "Hello World" will be translated however Spanish does not translate unless the "w" is lower case.




14. You should now have a running instance of a Node-RED application on IBM Cloud with the Watson Language Translator and Watson Assistant services, and you should be able to create basic flows.

# Lab 2 overview

This lab will expand on your first Node-RED flow. You'll create Node-RED flows that use:

- HTTP and HTML web pages
- JavaScript
- AJAX to consume a REST API
- The Watson Language Identification service

**Prerequisites**

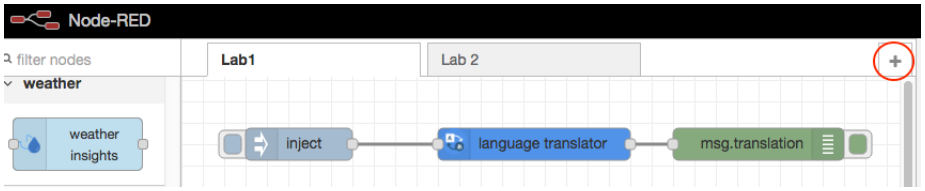
Complete Lab 1 and you should already have a running instance of Node-RED with the Watson Language Translator service connected.

## Create a simple web page

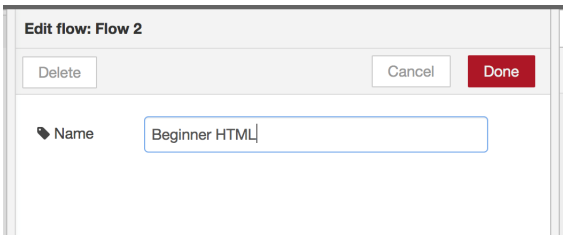
In this section, you will create a basic "Hello World" web page by using Node-RED.

1. Open the flow editor in your instance of Node-RED.

   Create a new flow tab by clicking +.



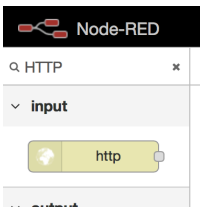

2. Double-click the new tab and enter a name for the new flow tab, then click **Done**.




3. Drag and drop an input **http** node onto the canvas. Use the **filter nodes** search field to find the nodes.

   Drag and drop an output **http response** node onto the canvas.

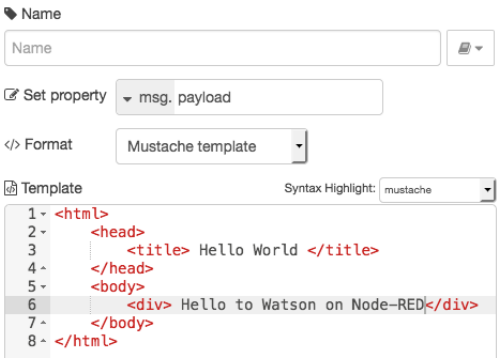
   Drag and drop a **template** node onto the canvas between the **http** and **http response** nodes.



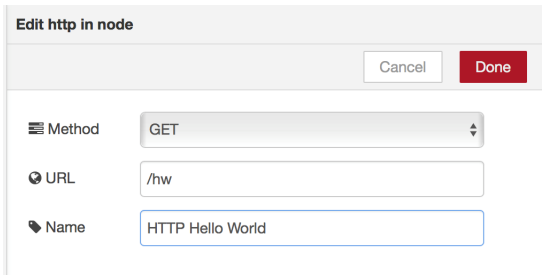

4. Wire the three nodes together.


5. Double-click the **template** node to edit it.
   Enter the simple HTML code:

   Click **Done** to save your changes.

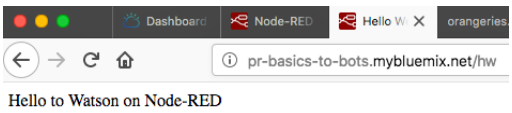


6. Double-click the input **http** node. Edit it to create an HTTP route to your web page by entering `/<some string>` in the **URL** field. Enter a name such as `HTTP Hello World`.




Click **Done** and deploy your changes.

7. Open a new browser tab and navigate to your new web page. The web address will be based on your Node-RED web address that is appended with the URL of your web page.

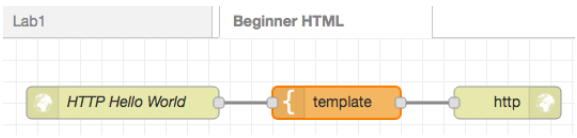



You should see your “Hello World” web page.

## Add JavaScript to your web application

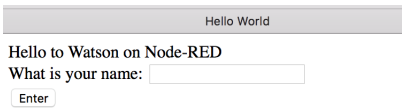
In this section, you'll modify your "Hello World" web page to include a text entry field and JavaScript.

1. Open the flow editor to your instance of Node-RED.

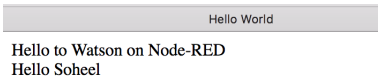



2. Double-click the **template** node and replace the code with the HTML in the file GSEBW-HelloWorld01. Click **Done** and deploy your changes.

3. Try out your JavaScript-enabled web page.




Enter your name and click Enter.

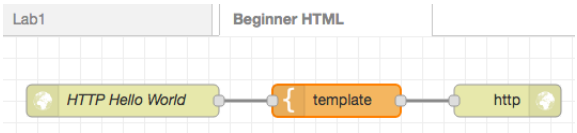


## Create a REST API

In this section, you'll modify your "Hello World" web page to invoke a REST API that you will create. This will show you how to do two important tasks:

- How to consume a REST API. This could be any REST API from any source, which allows you to add extra capability into your application.
- How to create a REST API in Node-RED. This will allow you to create reusable chunks of functionality that can be consumed by other flows that you create in Node-RED, and also be consumed by other applications, even if they are not Node-RED applications and even if they are not running in IBM Cloud.

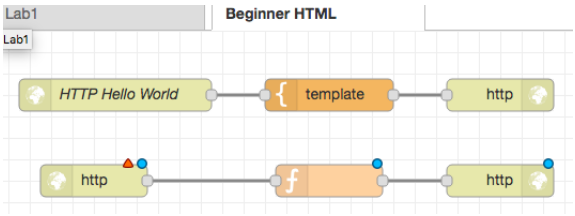
1. Open the flow editor to your instance of Node-RED.



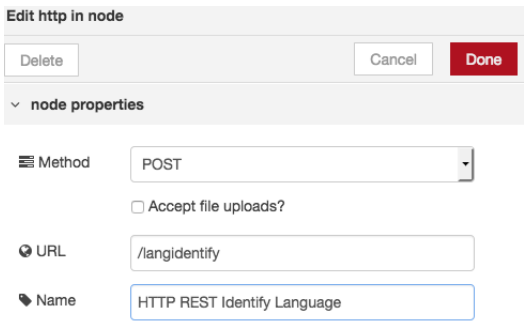

2. Double-click the **template** node and replace text with the HTML from the file GSEBW-HelloWorld02.

3. Drag and drop an **http** input node, another output **http response** and a **function** node onto the canvas.

4. Wire the nodes together.

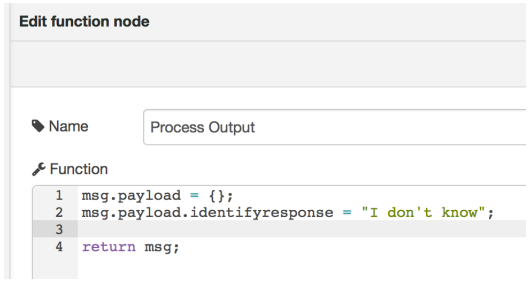



5. Double-click the input **http** node and specify the following information for each field:
   - Method: `POST`
   - URL: `/langidentify`
   - Name: `HTTP REST Identify Language`



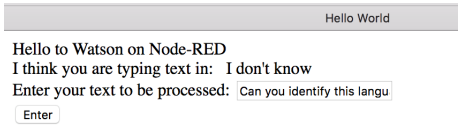

   Click **Done**.

6. Double-click the **function node**. Add the following code under **Function** and add **Process Output** to the name field:




   The phrase "I don't know" will be your default answer when the service is unable to process the request. You haven't yet added the service, so this response is appropriate because your application doesn't know how to process the request. Click **Done** and deploy your changes.

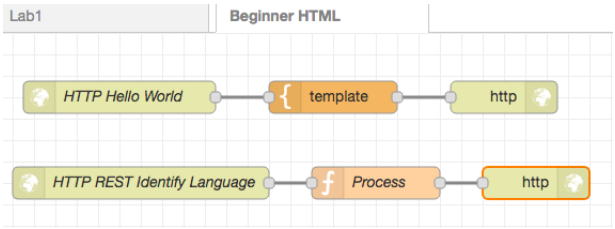
7. Test your application.



## Consume the Watson Translator service

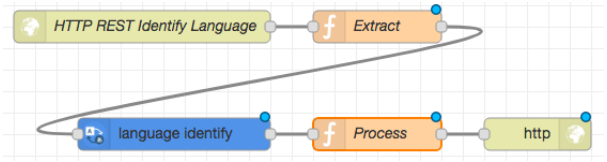
In this section, you will modify the REST API to invoke the Watson Language Translator service's Language Identification method.

1. Open the flow editor in your instance of Node-RED.



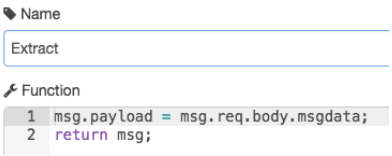

2. Delete the connections between the last three nodes by clicking the connectors and pressing Delete on your keyboard.


3. Drag and drop a **function** node and a **language identify** node onto the canvas. Wire the nodes together.



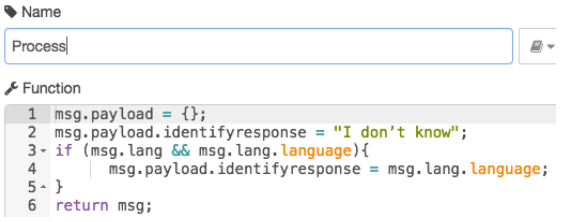

4. Double-click the new **function** node and enter code that takes the input from the REST call and makes it available for the Language Identify service to use. Enter this information and then click **Done**:
   - Name: `Extract Input`
   - Function:

```
msg.payload = msg.req.body.msgdata;
return msg;
```



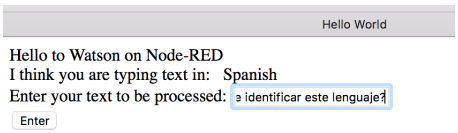

5. Double-click the second **function** node and add code that will send the response from the language identification service back to the client that is invoking the service. Enter this information and then click **Done**:
   - Name: `Process Output`
   - Function:

```
msg.payload = {};
msg.payload.identifyresponse = "I
don't know";
if (msg.lang && msg.lang.language){
   msg.payload.identifyresponse =
msg.lang.language;
}
return msg;
```




6. Deploy your changes.

7. Test your application by entering text other than English, such as Spanish text. Try other languages.




You now have a REST API that invokes the Watson Translator Language Identification method and an HTTP web application that invokes this API.

# Lab 3 overview

This lab shows you how to add Watson services and community nodes to your Node-RED applications.

Node-RED comes with a core set of useful nodes, but you can use a growing number of additional nodes from both the Node-RED project and the wider community.

You'll add the following Watson services:

- Speech to Text
- Language Translator
- Tone Analyzer
- Watson Assistant
- Text to Speech

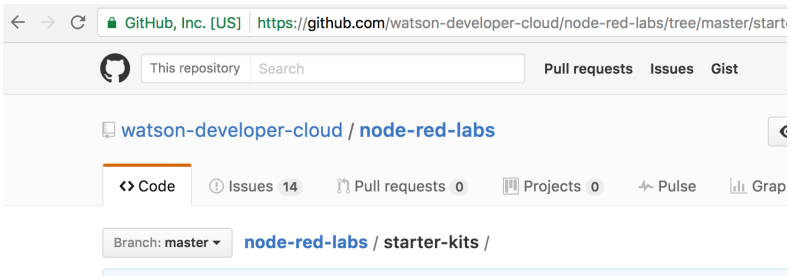
You'll also add code that allows you to send tweets to your Twitter account.

## Find community nodes

The Node-RED instance that you are running in IBM Cloud comes with a sample set of nodes, but you can also use community nodes, which are created and published by a community of developers. You will import node libraries from the community to create nodes for a microphone, audio, and a dashboard. In this section, you will create the Interpreter and a conversation bot (OK Watson) flows from the [Starter kits on the Watson Developer Cloud Github repository.]

1. Open the web page that contains the two starter applications that you will be re-creating:

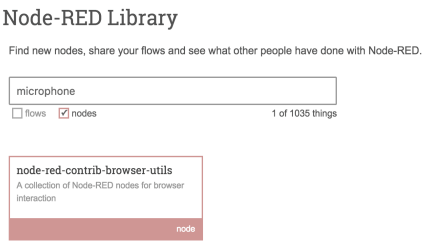
   https://github.com/watson-developer-cloud/node-red-labs/tree/master/starter-kits

   You will now learn how to find nodes like these for yourself.

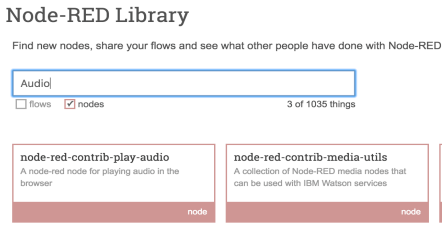
2. Open a new browser page and navigate to the Node-RED libraries at https://flows.nodered.org/.

3. Search for a **microphone** node that you will add to your application. Select the **nodes** check box. Do not select **flows**.

   The **node-red-contrib-browser-utils** library provides nodes for a microphone, camera, file inject, and unzip.

4. Search for **audio**. Among the results you should see two libraries: the **node-red-contrib-play-audio** and the **node-red-contrib-media-utils** library.

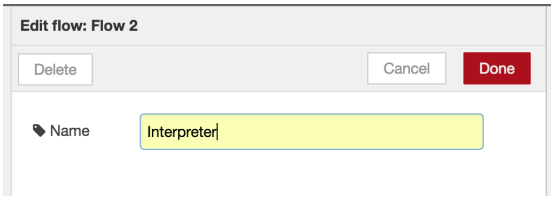
   The **node-red-contrib-play-audio** library provides a **speaker** node. **The node-red-contrib-media-utils** library provides **media** nodes for video and audio streams. We only require the speaker for this lab.

## Add the community nodes to the Node-RED palette

To make these nodes available to your instance of Node-RED, you need to add them to your Node-RED palette.

1. Click the **Add** (+) icon to create a new flow. 

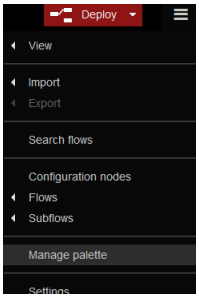
2. Create a new tab and name it `Interpreter`, then click **Done**.



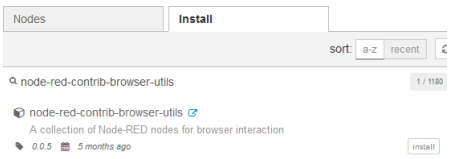

3. In the top right corner next to "Deploy," click the menu to open the options page.

   Select **Manage palette**.

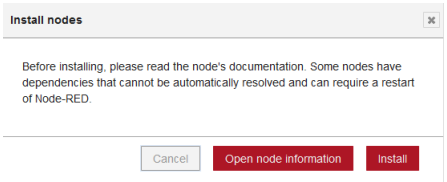
   The Nodes page tells you the nodes and versions that are already installed in your Node-RED instance.




4. Click the **Install** tab and search for node-red-contrib-browser-utils, then click **install**.

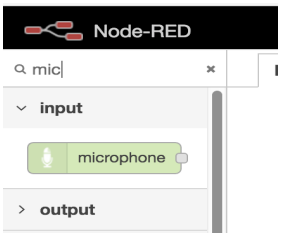



5. Click **Install** again in the "Install nodes" window.




6. Repeat the previous steps to install node-red-contrib-play-audio. After both nodes are installed, click **Close** and you will be returned to the flow.

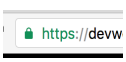
7. Search for the microphone node.




   The microphone node uses the browser capabilities that are available only on Chrome and new versions of Firefox.

8. If you are not using a supported browser for the microphone node, switch to Chrome or Firefox.

   The microphone node will need access to your computer microphone, which is possible only on HTTPS. If your instance of Node-RED is running on a platform such as IBM Cloud, make sure you are using HTTPS.

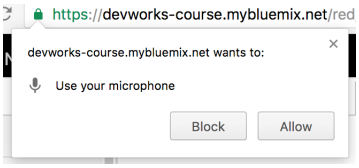



9. Drag and drop an input **microphone** node, a function **delay** node and an output **play audio** node onto the canvas.

10. Wire the nodes together and deploy your changes.


11. Click the tab on the microphone node to start your recording. 

12. If prompted, allow the node to access the microphone or share the selected device depending on your browser.




13. Speak to the microphone, then click the tab to stop the recording.

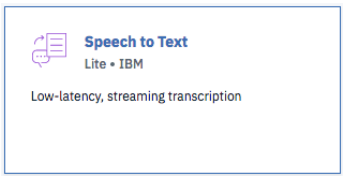
After about five seconds, you should hear your recording.

## Add the Speech to Text, Tone Analyzer, and Text to Speech services

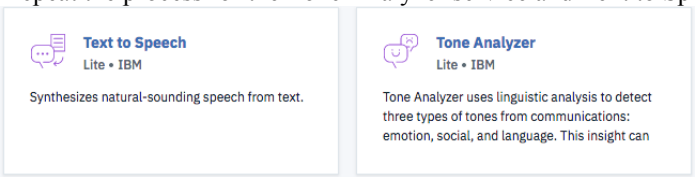
The applications that you'll be creating use the Speech to Text, Tone Analyzer, and Text to Speech Watson services. In this section, you'll connect those services to your Node-RED Instance.

1. Go to the IBM Cloud catalog and search for the Speech to Text service.

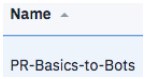
   Click **Create** and then go back to the catalog.



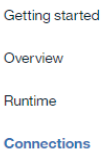

2. Repeat the process for the Tone Analyzer service and Text to Speech service.



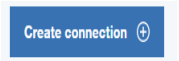

3. Go to your IBM Cloud dashboard and select your Node-RED application.




4. In the navigation pane, select **Connections.**

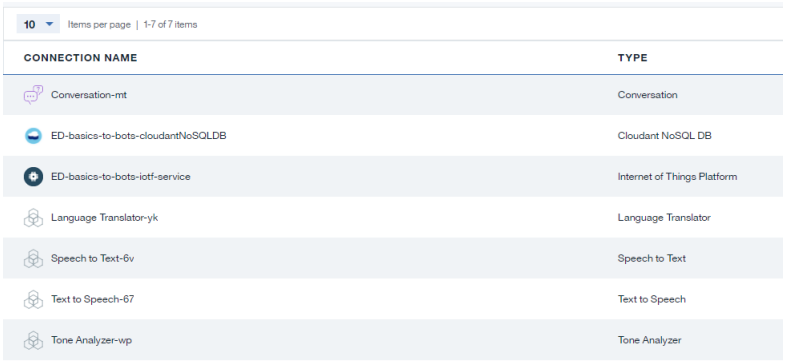



5. Click **Create Connection.**

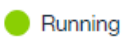



6. Connect the three services you just created to your Node-RED application. For the first two connections, do not restage your application. On the last connection, restage.

   Your connections page should look like this:




7. Wait for your application to restart and you will see green running light.

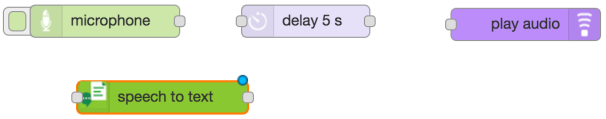


## Re-create the Interpreter application

In this section, you will re-create the [Interpreter application on GitHub](), which is a Node-RED flow that translates audio recorded by a microphone into a variety of languages.

1. Go to your Node-RED flow editor. Select and delete the wires.


2. Drag and drop the speech to text node onto the canvas. Select the **speech to text** node and click the info tab.




3. Scroll down and note that the input is expected to be an audio buffer on msg.payload.

4. Scroll further down and note that the output will be placed on msg.transcription.

The audio file to be analysed should be passed in on `msg.payload`.

Supported `msg.payload` types:.

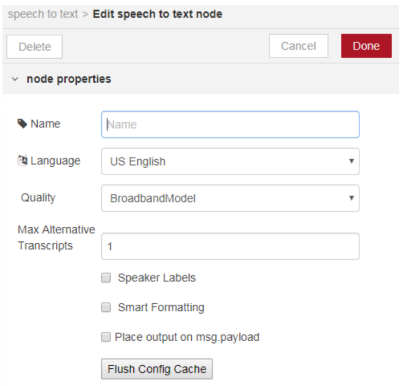
- **String** URL to audio
- **Buffer** Raw Audio Bytes

Audio must be a **WAV, FLAC or OGG encoded file**.

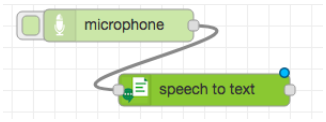
The returned audio transcription will be returned on `msg.transcription`.

The full response, including alternative transcriptions can be found on `msg.fullresult`.

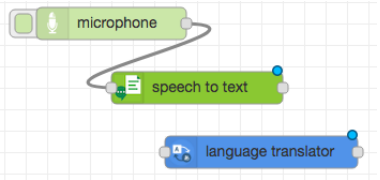
5. Double-click the **speech to text** node to edit the configuration. Set the language to US English and clear the **Speaker Labels** check box. Then, click **Done**.




6. Wire the **microphone** node to the **speech to text** node.




7. Drag and drop a **language translator** node onto the canvas.




8. Select the **language translator** node. Go to the **info tab** and note that the input is expected on msg.payload and the output is on msg.translation.
The **speech to text** node sends output to msg.transcription, but the **language translator** node expects input on msg.payload. This means that the nodes cannot be connected directly.

The text to translate should be passed in on `msg.payload`.

The translated text will be returned on `msg.payload`.

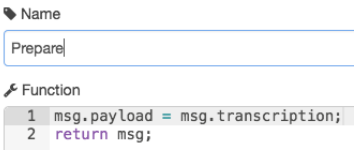
The full response from the service will be returned on `msg.translation`

9. Drag and drop a **function** node onto the canvas. Then, double-click the **function** node to edit it. Add the following name and code that allows the output from the **speech to text** node to be passed to the **language translation** node:

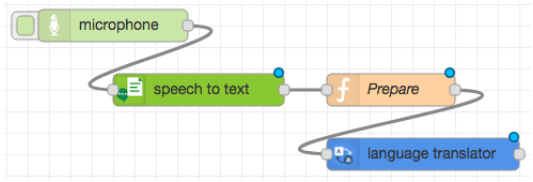
   Name: `Prepare for Translation`

   Function:
   ```
   msg.payload = msg.transcription;
   return msg;
   ```

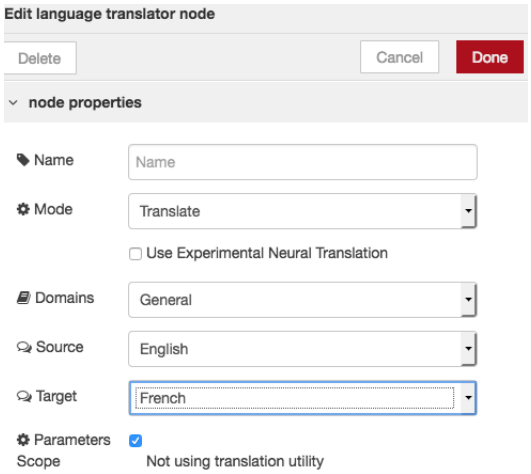
   Click **Done**.




10. Wire the **speech to text**, **function**, and **language translator** nodes.

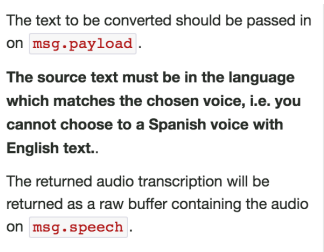



11. Double-click the **language translator** node to configure it. Select your source and target language. For example, select **English** and **French**. Then, click **Done**.




12. Drag and drop the **text to speech** node onto the canvas. Then, select the node and look at the info tab. Scroll down and note that the input is expected on msg.payload and that the output will be placed on msg.speech.

    This means that the **language translator** node, which outputs the simple translation on msg.payload, can be connected directly to the **text to speech** node. However, the **text to speech** node cannot be connected directly to the **play audio** node.

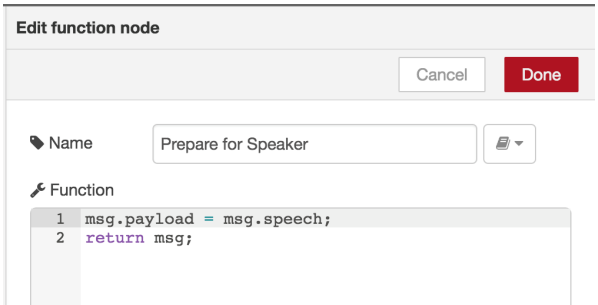



13. Double-click the **text to speech** node to configure it. Specify a language (it should be the same as the language in the translator node), voice, and format. Then, click **Done**.

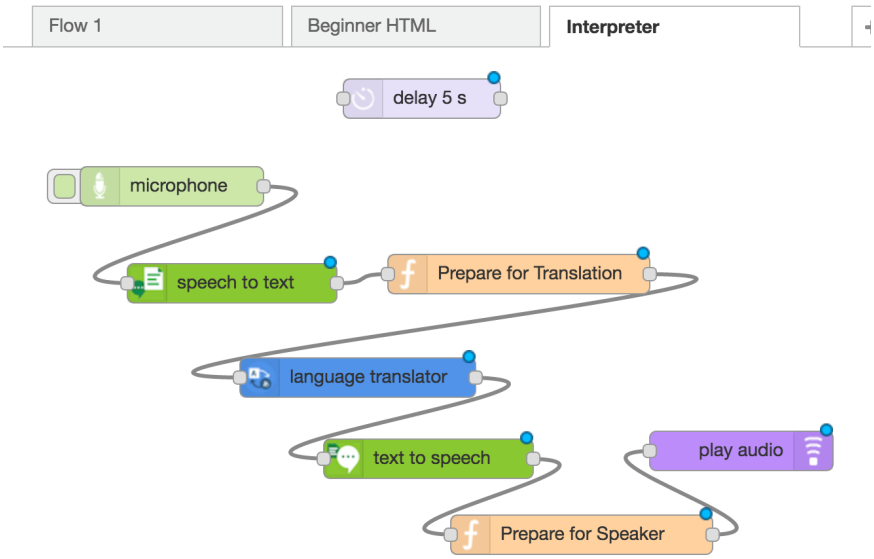
14. Drag and drop a **function** node onto the canvas. Then, double-click the **function** node to edit it. Under Function, set msg.payload = msg.speech so that it points to the audio from the text to speech service and it can be passed to the **play audio** node.

    Name: `Prepare for Speaker`
    Function:
    ```
        msg.payload = msg.speech;
    return msg;
    ```

    Click **Done**.



15. Wire the nodes together.




16. Deploy your flow. You can now try recording audio with your application.

17. To record yourself speaking, click the **microphone** tab, speak to your computer mic, and then click the tab again to stop the recording.

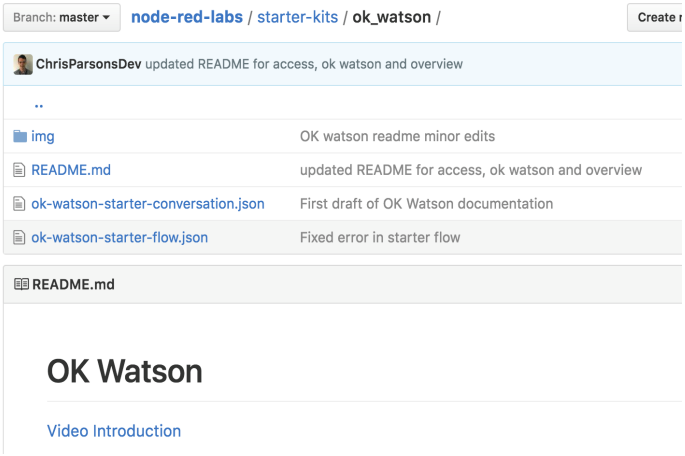
18. Wait for your application to process your input and listen to the translated output.

## Prepare for the OK Watson application

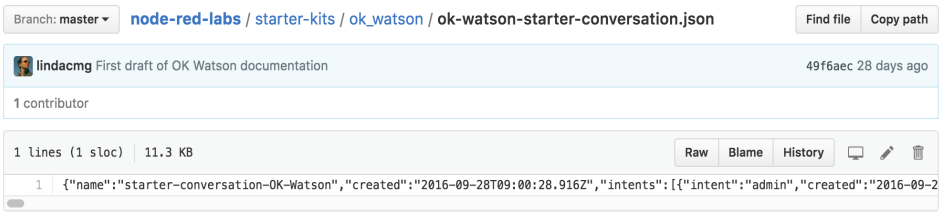
In this section, you will be re-creating a bot application named "OK Watson" by using the Watson Assistant service. For more information on the Watson chat bot, see the Conversation Service Tutorial.

You will import a prebuilt conversation from the OK Watson starter kit.

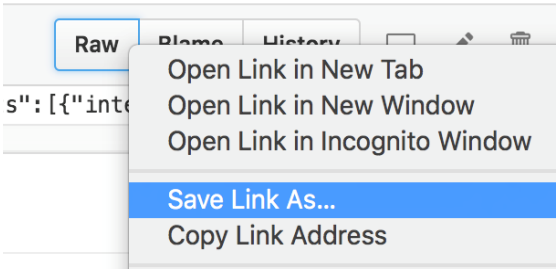
1. Open the OK Watson starter kit page on GitHub.



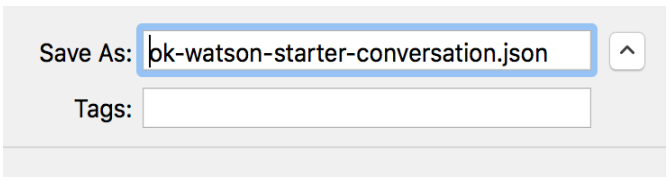

2. Click the **ok-watson-starter-conversation.json** link on the site.



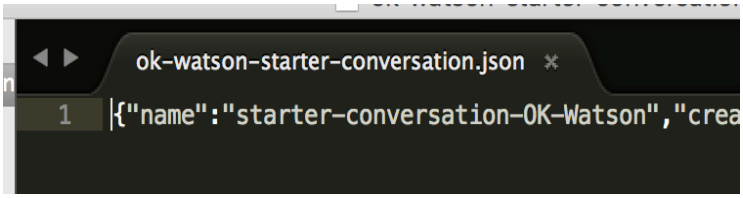

3. Right-click **Raw** and select **Save Link As**.




4. Save the file with a .json file extension.

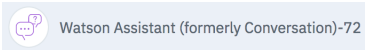



5. Find the file on your machine and open it to verify that the file has been downloaded properly and that is has JSON content.

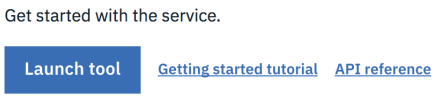



You are now ready to import the file into the Watson Assistant service.

6. Open your IBM Cloud page and select the Watson Assistant (formerly Conversation) service.

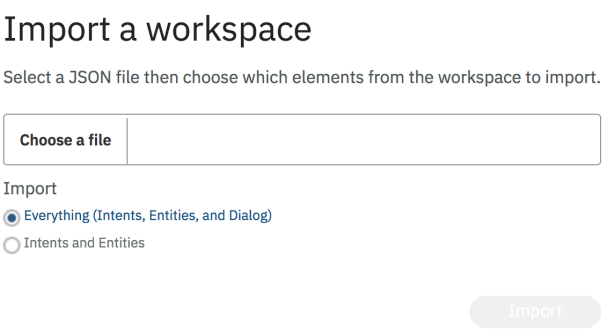



7. Find the section for the Conversation tooling and click **Launch tool**. Log in with your IBM ID, if requested.



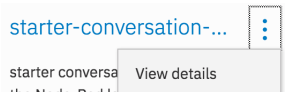

8. Select Workspaces from the home page  then click the **Import** icon. 


9. Select your file. Under **Import,** select **Everything (Intents, Entities, and Dialog)**, and then click **Import**. This action will create your workspace.

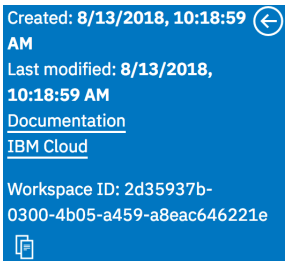



10. Once the import has been successful, you be shown your Conversation. For now, click on the window icon to go back to the list of workspaces.

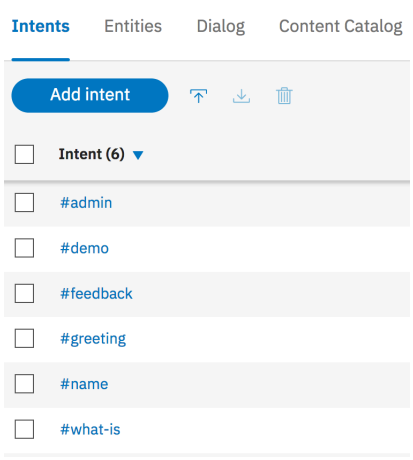
11. Click the **Menu** icon (three dots), and then click **View details**.




This view shows a summary of the workspace and the workspace ID. You need this ID for your Node-RED flow to be able to use the Watson Assistant service.




12. To see the intents, entities, and dialog nodes that were loaded, click the **Return** button. Then click **Get Started** to see your workspace.

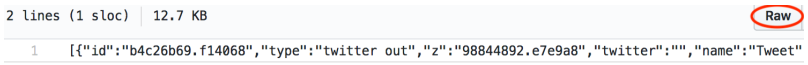


## Add link nodes to aid the view the logic of the application flow

1. Return to the home page for the [OK Watson Starter Kit].

2. Click the link to the JSON flow. OK Watson Flow JSON

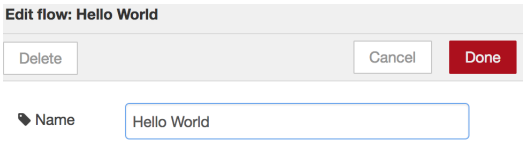
3. Click **Raw** to view all the JSON code.




4. Select all of the code and copy it to the clipboard. You will use this in a later step. The flow that you will import uses link nodes to make it easier to view the logic of the application flow.

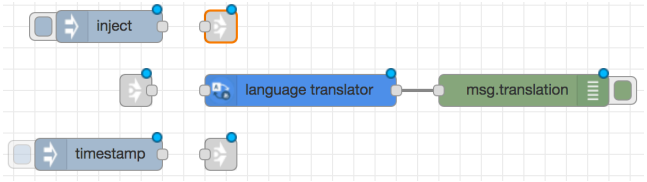
5. To see how these link nodes work, open your Node-RED instance in the flow editor, and open the first flow that you created in this course. (Use **Flow 1**, if you don't have a sample flow; use **Flow 2**, if a sample flow was automatically imported when you created the Node-RED instance.)

6. Provide a name for this tab (Flow 1 or Flow 2), such as `Hello World`, then click **Done**.

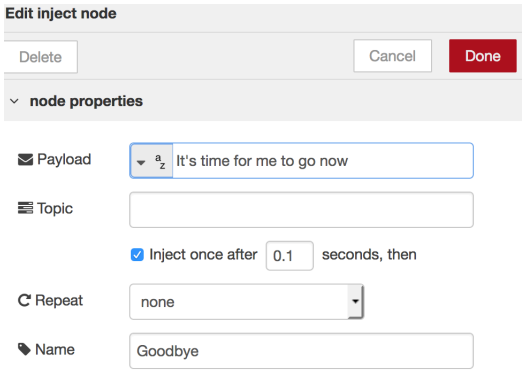



7. Drag and drop an inject node, two output link nodes and an input link node onto the canvas.

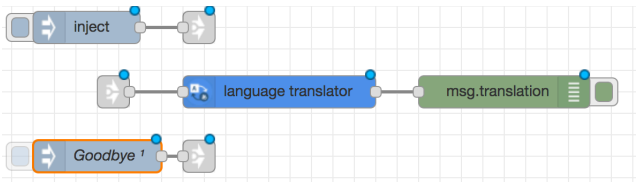
8. Select and delete the link between the original **Inject** node and the **language translator** node.



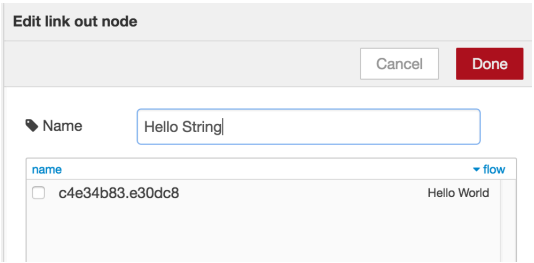

9. Double-click the second **inject** node and name it something like `Goodbye Inject`. Set the **Payload** field to a suitable string, such as `It's time for me to go now`. Then, click **Done**.



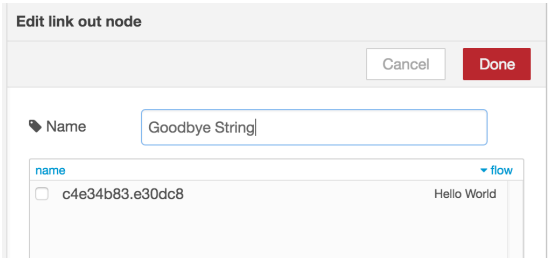

10. Wire the nodes together so that the original **Inject** node, the **language translator** node, and the **Goodbye Inject** node are wired to separate link nodes.



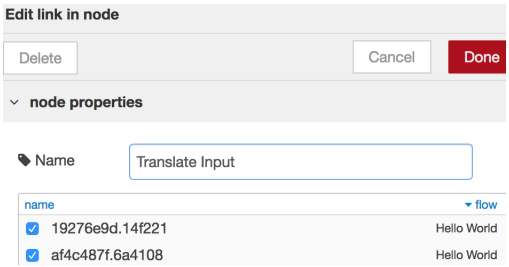

11. Double-click the link that is connected to the original **Inject** node. Name the **link out** node to something like `Hello String`, then click **Done**.



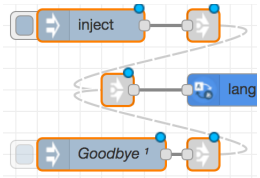
12. Repeat the previous step for the **link** node that's connected from the **Goodbye Inject** node. Name the **link** node to something like `Goodbye String`, then click **Done**.




13. Double-click the **link** node that's wired to the **language translator** node. Name the link something like `Translate Input`, select the check boxes to link to both output links, and then click **Done**.



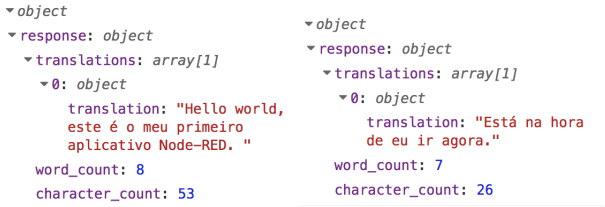

14. To see what a link node is connected to, select that node. For example, select the input **link** node connected to the **language translator** node.




15. Click **Deploy** to save your changes. Click the **Debug** tab. To clear the list, click the **Trash** icon.

16. Click each **inject** node in turn. 

You should see the output in the **Debug** tab.

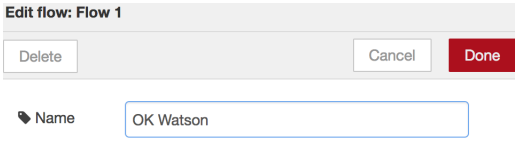



You are now ready to import the OK Watson flow.

## Import the OK Watson flow

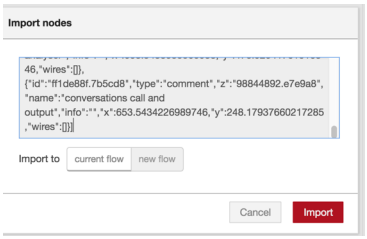
In this section, you will re-create the OK Watson application. This is a more complex flow, and you will be taking a short-cut to create it.

1. Create a new tab and set the name of the tab to something like `OK Watson`. Click **Done**.
   The flow should still be in your clipboard. If not, repeat steps 1-4 in the previous section, "Add link nodes to more easily view the logic of the application flow."

   You are now ready to import the flow into the flow editor.

2. Click the flow editor **Menu, c**lick **Import** > **Clipboard**, then paste your flow into the text box and click **Import**.

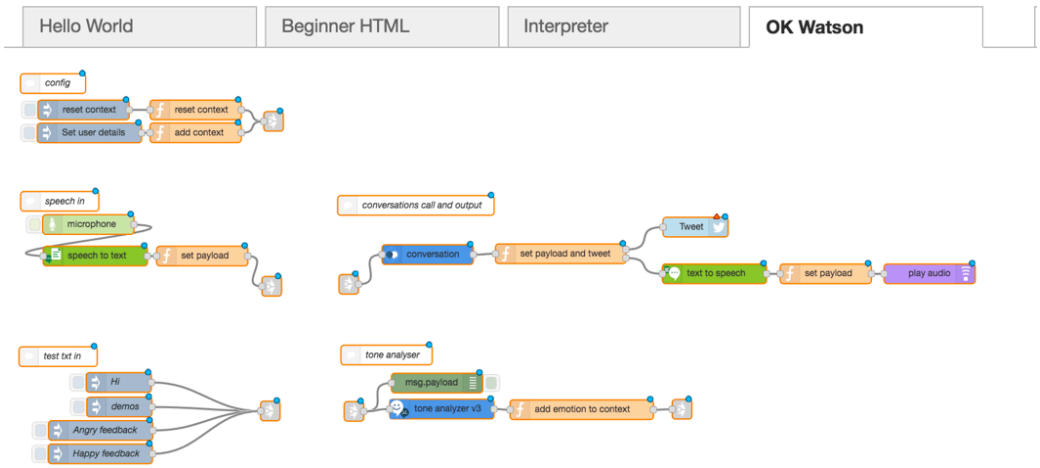
   
   

   You now have the OK Watson flow in your flow editor.

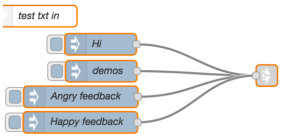
   

3. You can use the Zoom buttons to show the whole flow.

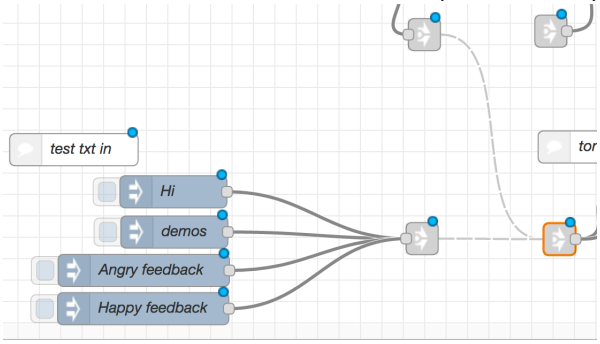
   You have the OK Watson flow in your editor

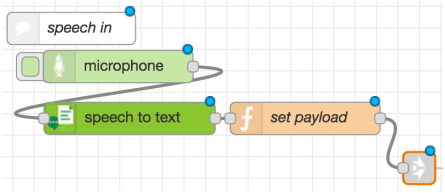
4. Focus on the nodes in the "test txt in" grouping. These nodes provide sample input strings.

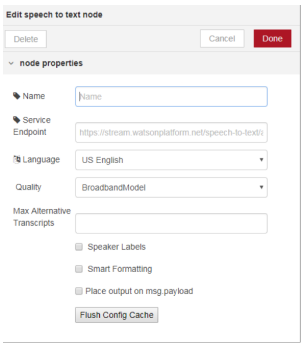
5. Click the **link** node. Note that the text is passed directly to the **tone analyzer** node.

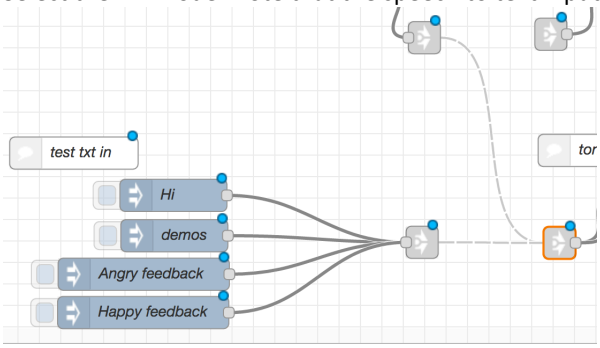
6. Focus on the “speech in” the set of nodes. This group uses the **microphone** and **speech to text** nodes.



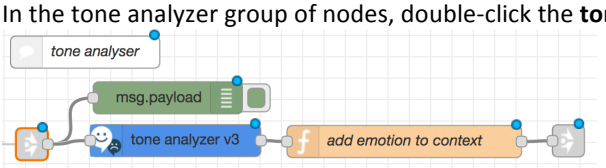

7. Double-click the **speech to text** node to set a spoken language and ensure you use BroadbandModel for Quality. Then, click Done.



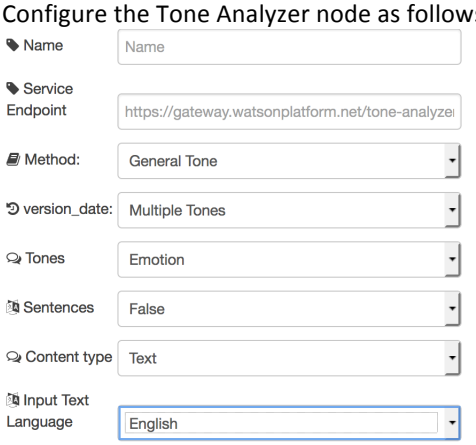

8. Select the **link** node. Note that the speech to text input node also goes to the tone analyzer node.




9. In the tone analyzer group of nodes, double-click the **tone analyzer** node.



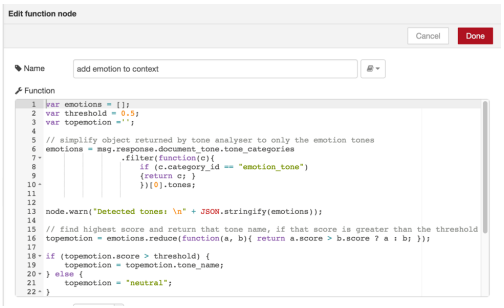

10. Configure the Tone Analyzer node as follows:




Then click **Done**.

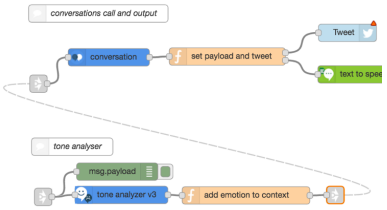
11. Double-click the **add emotion to context** function node to view the code.


The script in this function node determines the highest tone score and sets it to msg fields that the conversation node will look for.

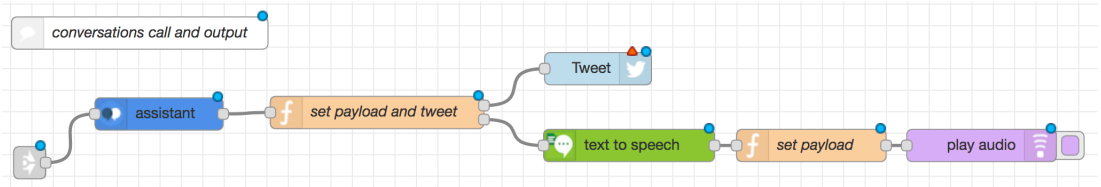



Click **Cancel** to close.

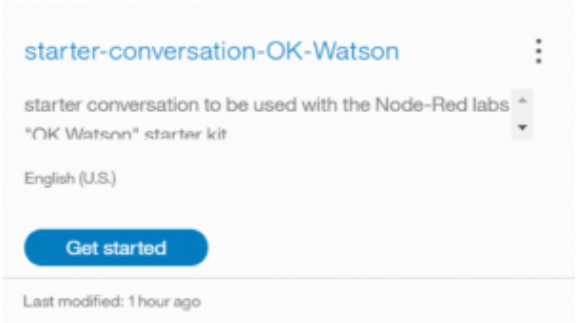
12. Click the **link** node. Note that the output from the tone analyzer links to the conversation node.



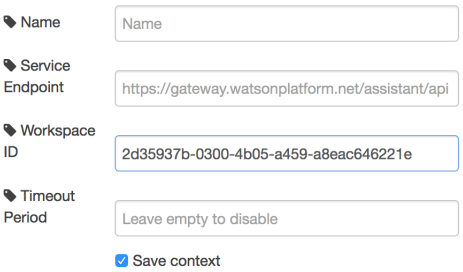

13. Focus on the conversation group of nodes.




14. To configure the conversation node, find your conversation workspace ID that you created in "5. Prepare for the OK Watson application" > step 10. Then, go to IBM Cloud.
    1. Find your instance of the Watson Assistant service and launch the tool.
    2. Click the service and launch the tooling.
    3. Find your workspace.
    4. Click **Options** and **View details** to see your workspace ID.
    5. Click the **Copy** button to copy the ID to the clipboard.



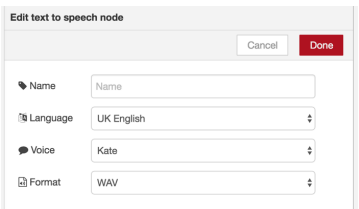

15. Go back to your Node-RED flow editor. Double-click the **assistant** node. Paste in your workspace ID. Select the **Save context** check box to ensure that the node remembers the conversation context.




If you do not select **Save context**, then it is the responsibility of your application flow to pass in the context object that is associated with the context. Setting this option to save the context enables the code in the node to do this control for you.

Click **Done**.

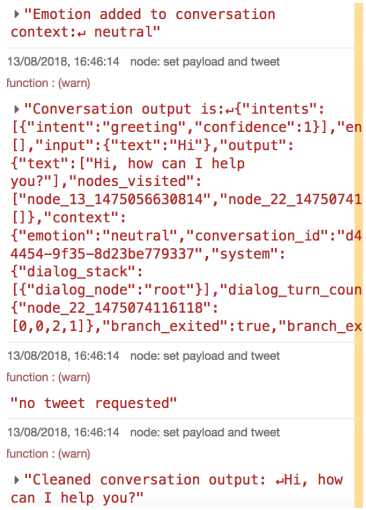
16. Edit the **text to speech** node. Set the output language.




Click **Done**. For now, do not configure the twitter node.

Click **Deploy** to deploy your flow.

17. Try out your application by using either the microphone or any of the sample test texts.




You should now have two running applications that use multiple Watson services:

- Speech to Text
- Translation
- Tone Analyzer
- Watson Assistant
- Text to Speech