InterConnect
2017

March 19–23
MGM Grand &
Mandalay Bay
Las Vegas, NV

IBM

Hands-on Lab Session 2765 Getting Started with IBM Bluemix and Watson APIs

Clemence Lebrun, Developer Advocate,
IBM Digital Business Group

Clemence.lebrun@fr.ibm.com

© Copyright IBM Corporation 2017

IBM, the IBM logo and ibm.com are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

This document is current as of the initial date of publication and may be changed by IBM at any time.

The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. This information is based on current IBM product plans and strategy, which are subject to change by IBM without notice. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way.

Table of Contents

Part 1: Getting Started with Bluemix 5

Create your Bluemix account 5
Create Watson services instances 5
Initialize the Conversation service 7

Part 2: Watson APIs & Node-RED 10

The Text to Speech service 10
The Speech to Text service 13
The Tone Analyzer service 14
The Conversation service 17

Part 3: Watson APIs and your Node.js web application 20

Create Your Hello World Node.js app 20
Bind your Text to Speech service to your app 21
Improve your code to use the service 22

Overview:

This Lab will give you a jump start on each of the following Watson APIs:

- Conversation
- Tone Analyzer
- Text to Speech
- Speech to Text

In the part 1, you will create these services on Bluemix.
In part 2 you will use the services with a local Node-RED application.
In part 3 you will create a Node.js app on Bluemix and integrate the Text to Speech services.

Peace of advice

Use **Chrome** as a browser. You will need to access Bluemix. Try to keep your tabs open to avoid losing time:

- 1 tab for your Bluemix Dashboard (parts 1, 2 3)
- 1 tab for the Conversation tool (parts 1 and 2)
- 1 tab for your local Node-RED app (part 2)
- 1 tab for your deployed app (part 3)

Part 1: Getting Started with Bluemix

In this section, you will learn how to create service instances with the IBM Bluemix platform in a few minutes.

IBM Bluemix is a cloud Platform as a service (PaaS) which supports several programming languages (Java, Node.js, Go, PHP, Python, Ruby) to build, run, deploy and manage applications easily on the cloud. You can create and deploy your app in the cloud using the runtime of your choice and choosing from 130+ services across IBM Watson, IoT, mobile and more.

Create your Bluemix account

1. **Connect** to https://console.ng.bluemix.net/registration/ , fill the form and click on "Create account". If you already have a Bluemix account, you just need to Log In. Please, chose "United States" as country.

   You can try Bluemix free for 30 days, it does not require any credit card, you have access to 2GB of runtime and container memory free for 30 days, plus access to provision up to 10 services. Your trial comes with help desk support.

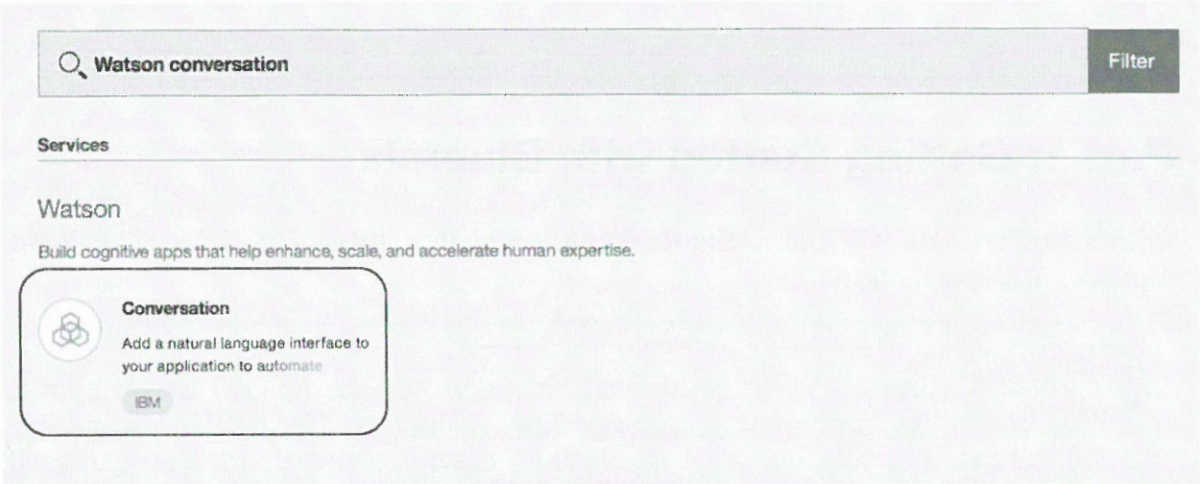
2. To confirm your Bluemix account, please check your emails, you should have received one from "IBM Bluemix". Click on the link in the email to confirm your account and **login** to Bluemix.

Create Watson services instances

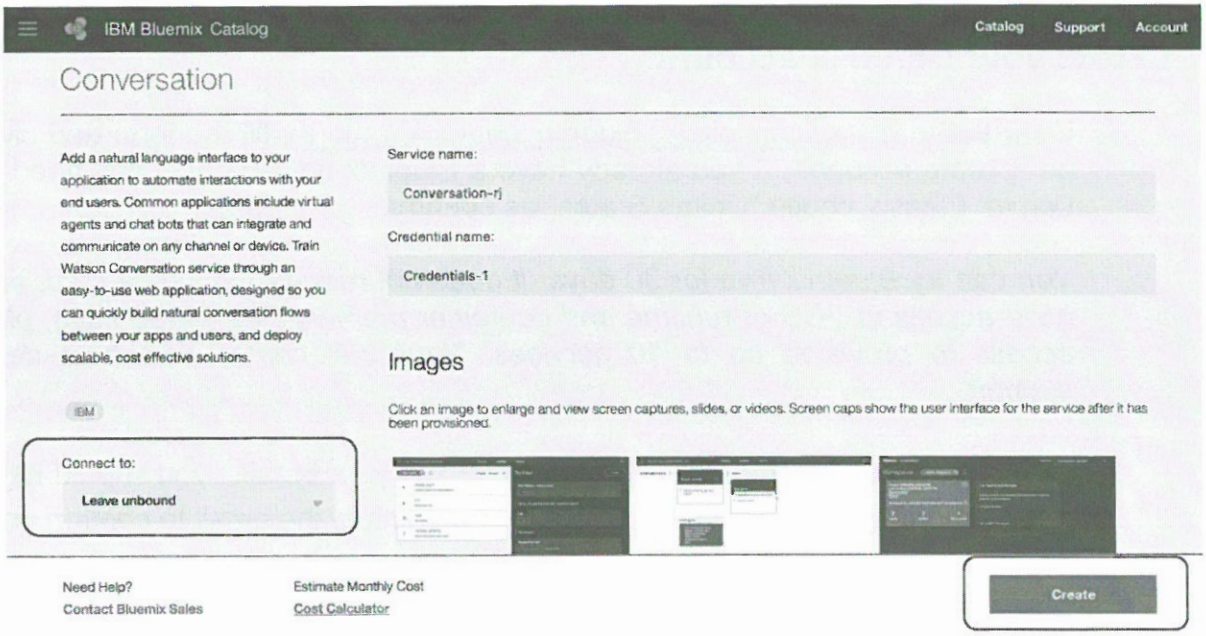
1. Once you are logged in, select "**Catalog**" on the menu bar.

In this part, you are going to create instances of several Watson services available on Bluemix. Credentials will be generated and this way, you will be able to use the Watson APIs from any application.

2. In the search bar, type "Watson conversation" and select the **conversation service**:

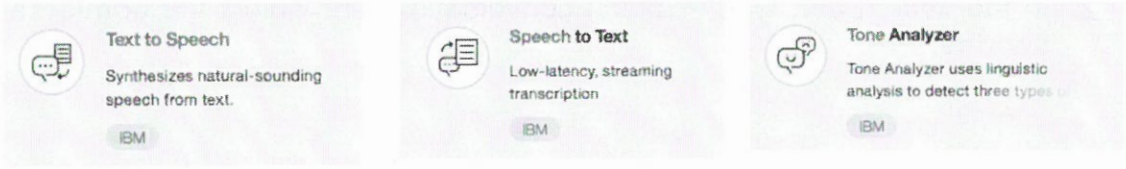


3. In the « Connect » dropdown, chose « **Leave unbound** » and click on « **Create** » to create the service.



4. Go back to the Catalog tab and repeat the same creation process for the following services:

- **Text to Speech** service
- **Speech to Text** service
- **Tone Analyzer** service



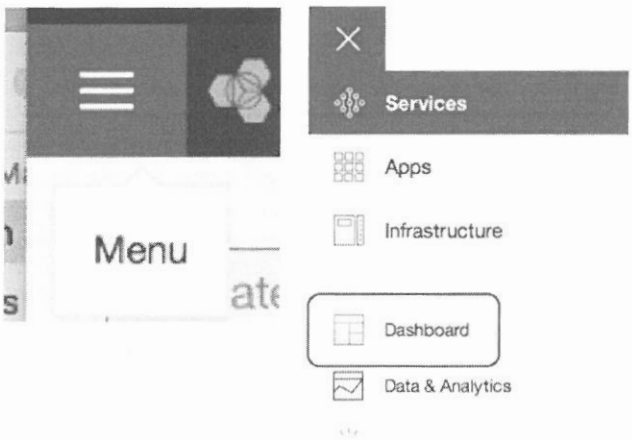
Initialize the Conversation service

The **Conversation service** is used to add a natural language interface to your application and automate interactions with your end users. Common applications include virtual agents and chat bots that can integrate and communicate on any channel or device.
With the **Conversation tool**, you can quickly build natural conversation flows between your apps and users, and train bots with the Conversation service through an easy-to-use web application.

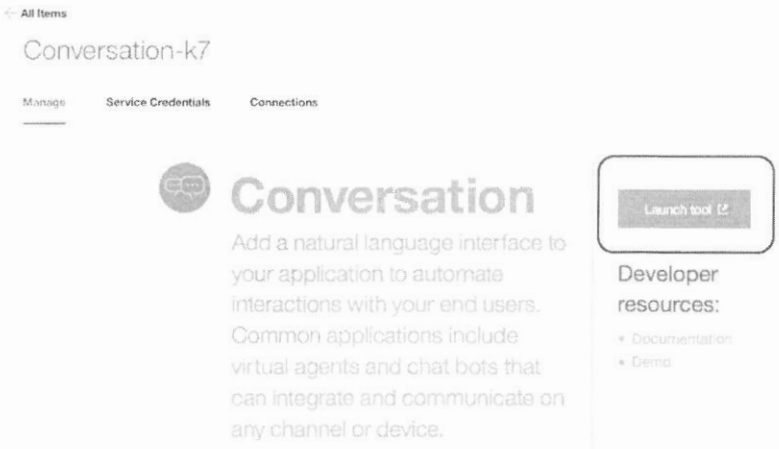
1. In our case, you are not going to create your own conversation flow but to use an existing one. **Download** the folder https://github.com/cllebrun/Interco17

2. In Bluemix, you can access the different instances you created (apps, services ...) from your Dashboard.

The Dashboard is accessed by the menu (up and left):

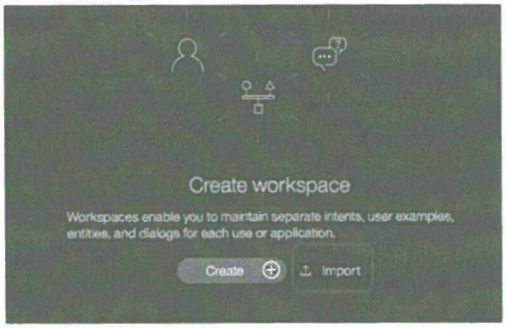


3. In the "services" list, **click** on the Conversation service instance you have created and **launch** the Conversation tool:



Login if needed.

4. **Import** the "**conversation.json**" file from the folder you just downloaded as an existing workspace:



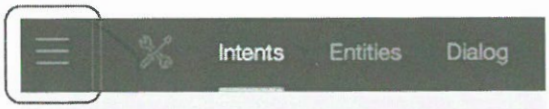
One **workspace** corresponds to one bot — it contains all the configuration information that gives your bot its unique personality. A workspace is a container for all the artifacts that define the behaviour of your service.

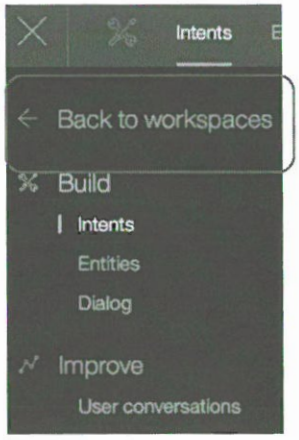
Intents and associated User examples are created to help Watson recognize what the chat user is looking to do.

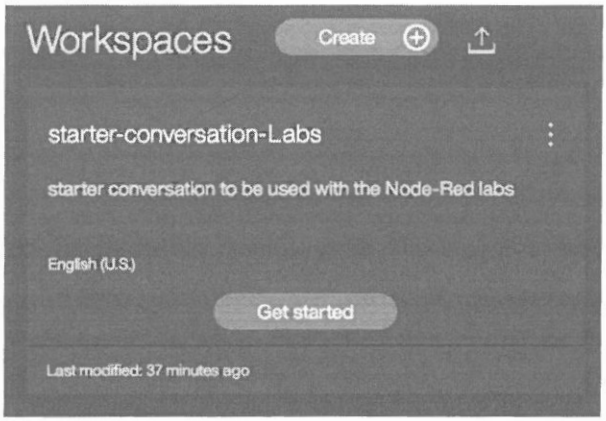
Entities are the specific pieces of information we want to extract from the user response. You'll want to group entities that might trigger a similar response in your dialog. Each value can have multiple synonyms, which define different ways that the same value might be specified in user input.

A **dialog** is a set of conversational nodes that are contained in a workspace. Together the set of nodes makes a dialog tree, on which every branch is a conversation that can be had with a user.

5. Your dialog is already created, go back to the precedent page with your dialog info:







Please, keep this Conversation tool browser tab open for the next steps.

Part 2: Watson APIs & Node-RED

In this section, you will start and develop a local node.js project to use the Watson services created in part 1. You will use Node.RED in top of this node.js app.

Node-RED provides a browser-based flow editor that makes it easy to wire together flows using the wide range nodes in the palette. Flows can be then deployed to the runtime in a single-click. Some of the Bluemix services catalog like the Watson services have their own nodes available.
Node-RED is built on Node.js, you can run it locally like any Node.js application and then push it to Bluemix.

On your VM, under the folder Documents/part1, there is a Node-RED project already installed locally.

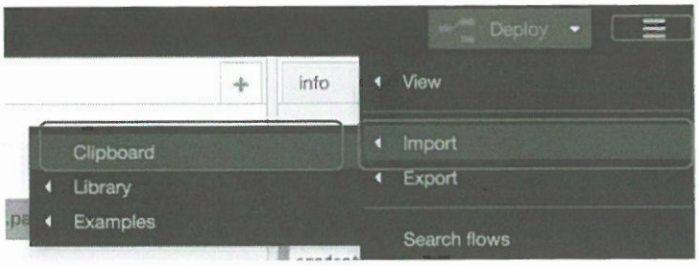
1. Open a terminal window
2. Run the following command to locate in the project folder:
   - cd Documents/part1/node-red
3. Run the following command to start the Node-RED project:
   - node red flow.js
4. Open a **Chrome** tab and browse http://127.0.0.1:1880/

You are now ready to start to develop your application using Watson APIs.

To create flows with Node-RED, you just need to drag and drop nodes from the toolbox on the left and to wire them together. The input and output messages are usually in the "msg.payload" json object. In this part, you are going to import existing flows, configure nodes and deploy your flows.

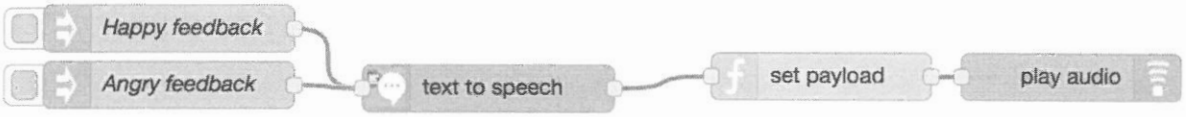
The Text to Speech service

1. **Copy** the content of the "2TexToSpeechtFlow.json" file from: https://github.com/cllebrun/Interco17
2. In Node-RED, click on the menu (up and right) and select **Import** -> **Clipboard**:



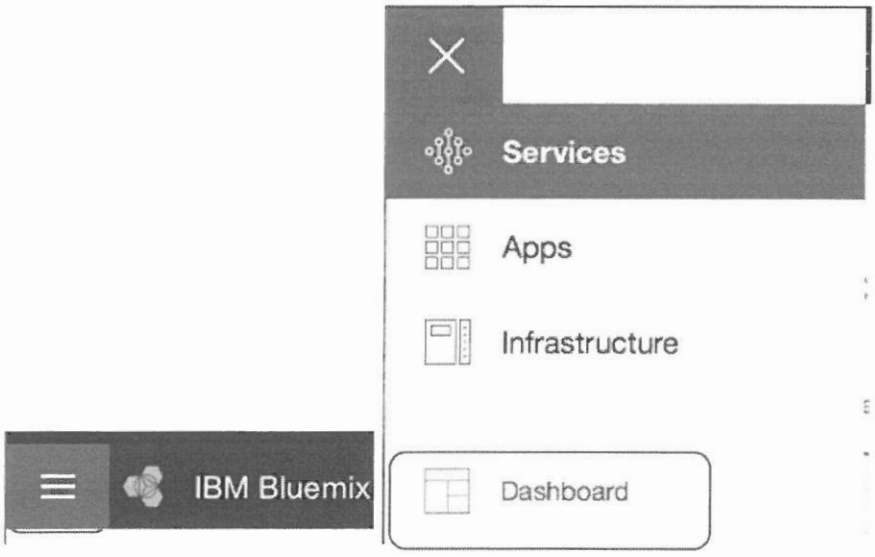
3. **Paste** the content of the "2TexToSpeechtFlow.json" and click on **Import**:

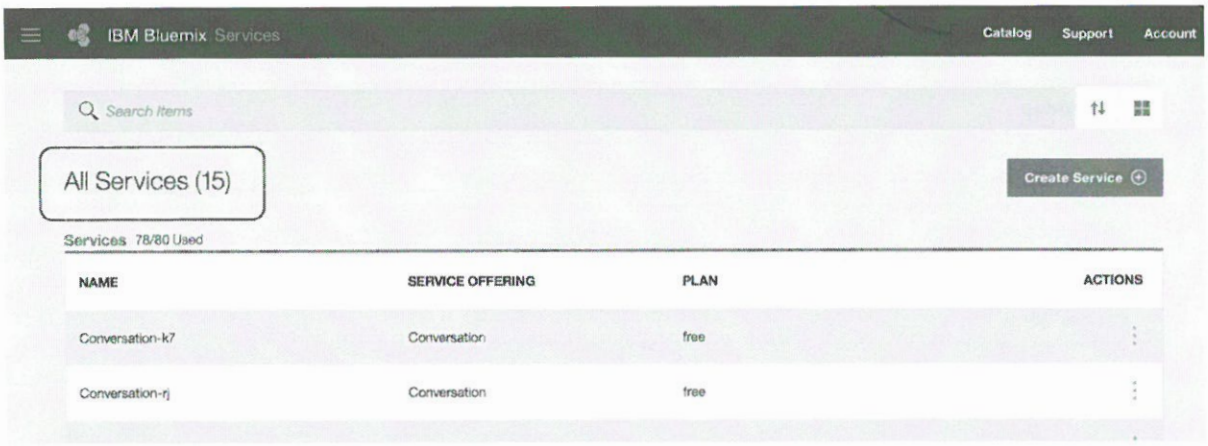
The following flow appears in your workspace:



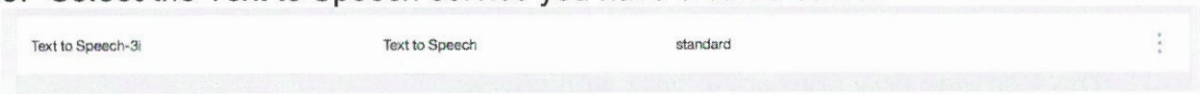
The Text to Speech service understands text and natural language to generate synthesized audio output complete with appropriate cadence and intonation.

4. You need to configure the Text to Speech node with the service credentials you have created in part 1 on Bluemix. Keep your Node-RED tab open and open your Bluemix tab, in your services list available on your dashboard:

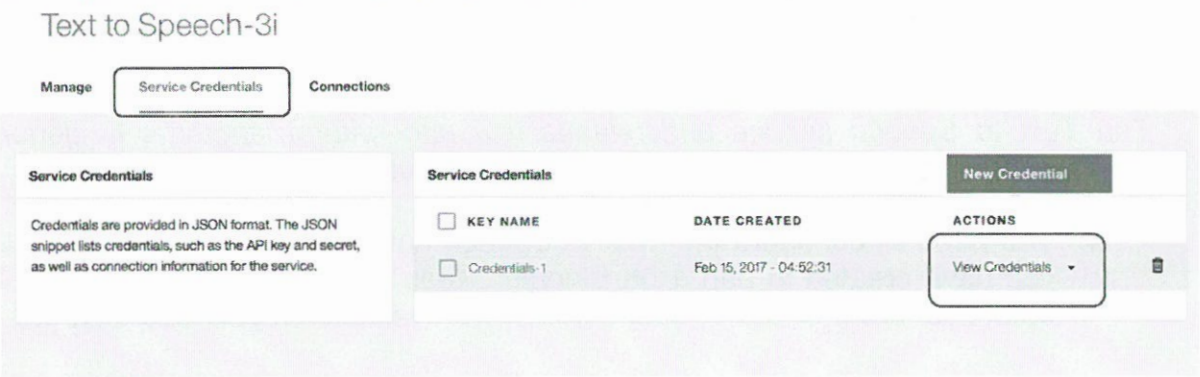




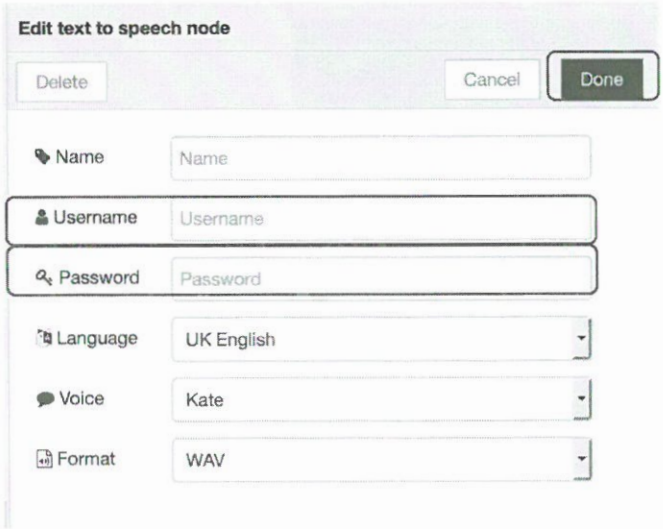
5. **Select** the Text to Speech service you have created earlier:



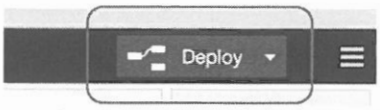
6. **Select** the "Service Credentials" tab and click on "View Credentials"



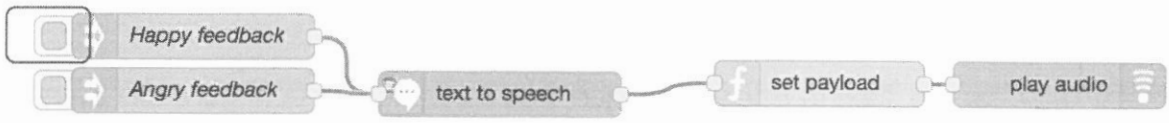
7. **Copy/paste** the username and password in your Node-RED text to speech node (you can open the node with a double click) and click Done:



8. **Deploy** your flow:



9. Make sure your laptop speakers are turned on and **start** your flow with one of the two possible text inputs:



You can listen to the text spoken in your browser.

The Speech to Text service

1. **Copy** the content of the "3SpeechToTextFlow.json" file in: https://github.com/cllebrun/Interco17
2. In Node-RED, click on the menu (up and right) and select **Import** -> **Clipboard**:

3. **Paste** the content of the "3SpeechToTextFlow.json" and click on **Import**:

The following flow appears in your workspace:

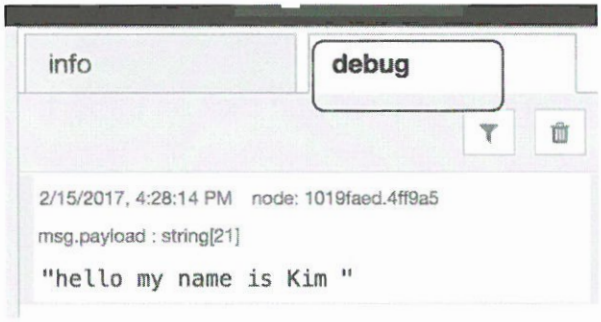
The microphone node records audio directly from the web browser.
The recorded audio is stored in the nodes msg.payload object. The Speech to Text converts the human voice into the written word. This service uses machine intelligence to combine information about grammar and language structure with knowledge of the composition of the audio signal to generate a more accurate transcription.

4. Like for the Text to Speech service, you need now to configure the Speech to Text node. In Bluemix, **get** your Speech to Text service **credentials** (you have created in part 1) like you did for the text to speech service and **copy/paste** it to your speech to text node.

5. **Deploy** your flow:

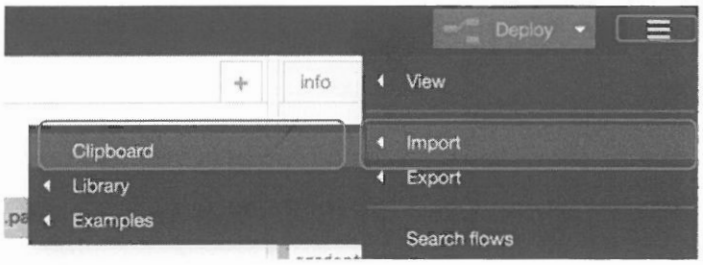
6. **Click** the microphone button to start recording. Say something and click the button again to stop recording:

7. Watch the debug tab to see your record:
   For example, I said "hello, my name is Kim":



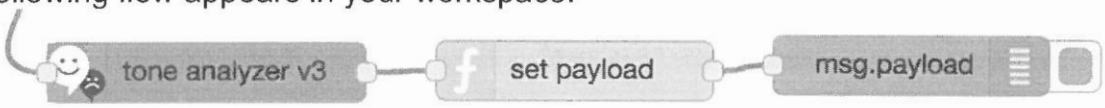
The Tone Analyzer service

1. **Copy** the content of the "4ToneAnalyzerFlow.json" file in:
   https://github.com/cllebrun/Interco17
2. In Node-RED, click on the menu (up and right) and select **Import** -> **Clipboard**:

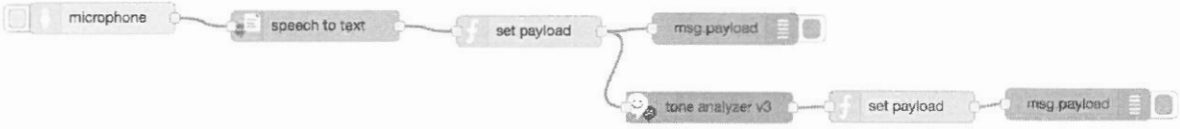


3. **Paste** the content of the "4ToneAnalyzerFlow.json" and click on **Import**:

The following flow appears in your workspace:



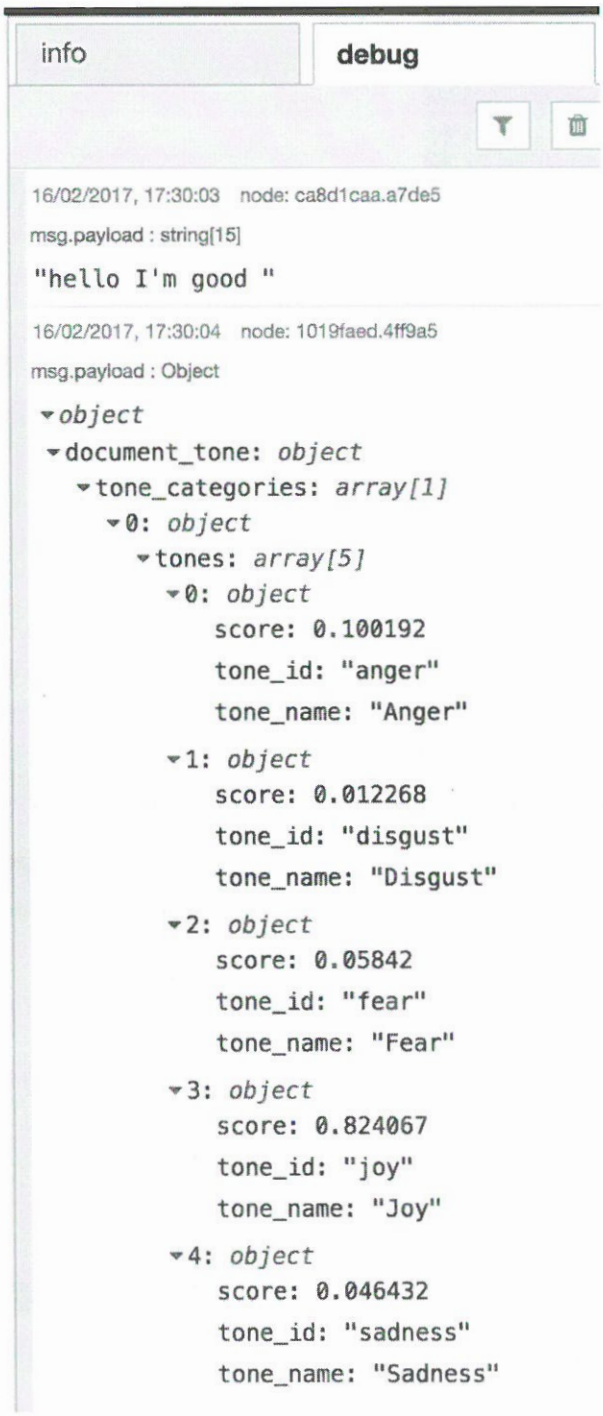
4. Add the Bluemix credentials, corresponding to the Tone Analyzer service you have created in part 1.
5. Link the Tone Analyzer node to the output of the last flow created with the Speech to Text node and deploy your flow:



The Tone Analyzer service uses linguistic analysis to detect emotional tones, social propensities, and writing styles in written communication.

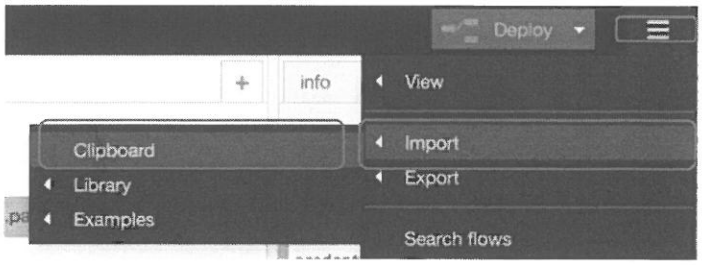
6. **Deploy**. Activate the microphone and say something with an emotion. Watch the result in the debug tab, you will get a score on 4 emotions: **anger**, **disgust**, **joy** and **sadness**.

For example, I said "hello I'm good" and the result was:

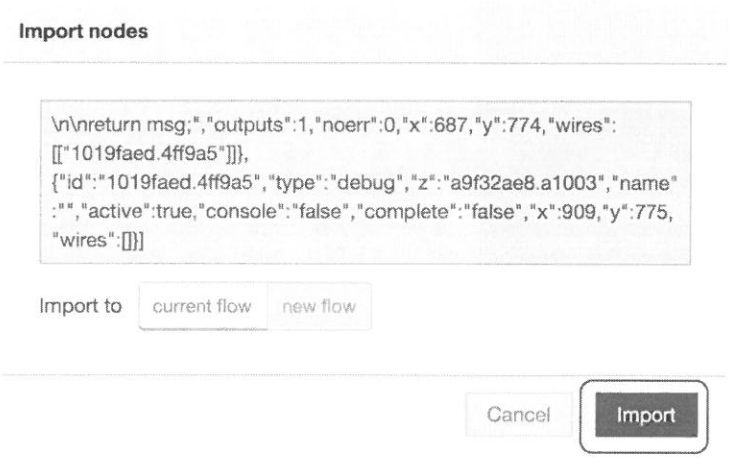


The Conversation service

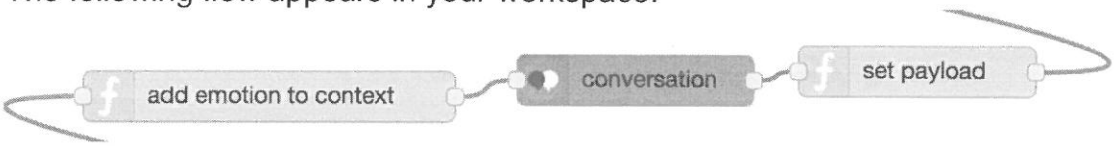
1. **Copy** the content of the "5ConversationFlow.json" file in: https://github.com/cllebrun/Interco17
2. In Node-RED, click on the menu (up and right) and select **Import** -> **Clipboard**:



3. **Paste** the content of the "5ConversationFlow.json" and click on **Import**:



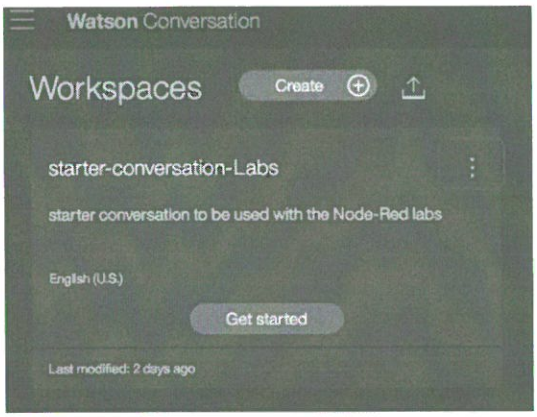
The following flow appears in your workspace:

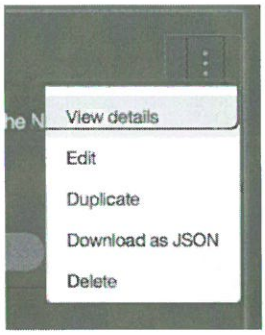


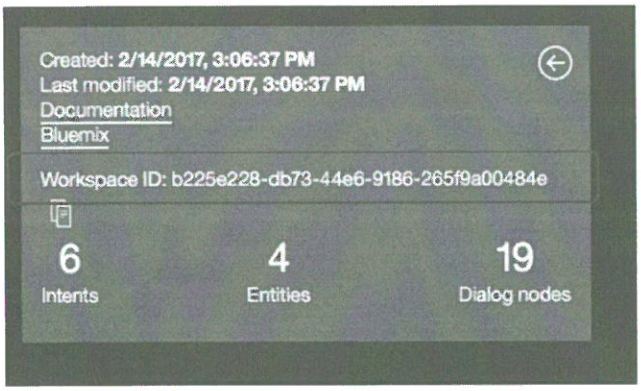
With the IBM Watson Conversation service, you can create cognitive agents – virtual agents that combine machine learning, natural language understanding, and integrated dialog scripting tools to provide outstanding customer engagements.

4. **Add the Bluemix credentials**, corresponding to the Conversation service you have created in part 1.

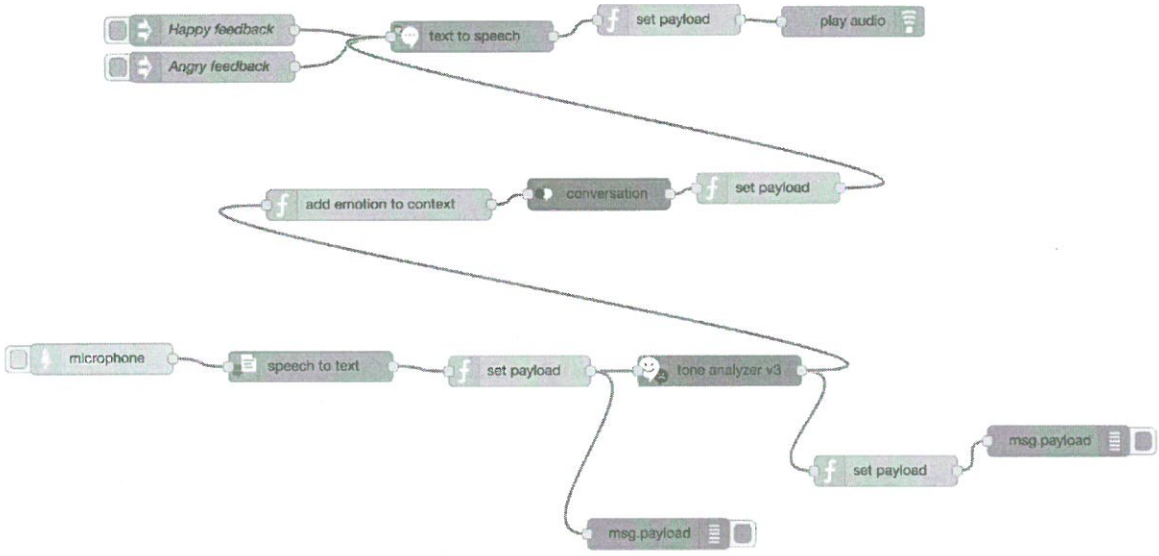
5. **Add your Workspace id**. You find this id when launching your Conversation tool (you should have your browser tab open from part 1 - 5.):







6. **Wire** the nodes: the Tone Analyzer output and the Text to Speech input:



7. **Deploy** your flow.

8. **Click** the microphone button to start recording. Say something about the node-RED project and click the button again to stop recording.

   For example:
   - *"These new nodes are awesome! I love the way they are used in new flows!"*
   - *"I don't like these nodes – they're not documented enough"*
   - *"I'm having trouble following the instructions that are given"*

   Listen to the conversation answer and continue the conversation. You can ask about available demos for example.

   The analyse and answers are detailed in the debug tab.

In part 1 and 2, you have tested several Watson APIs using a local Node-RED project.

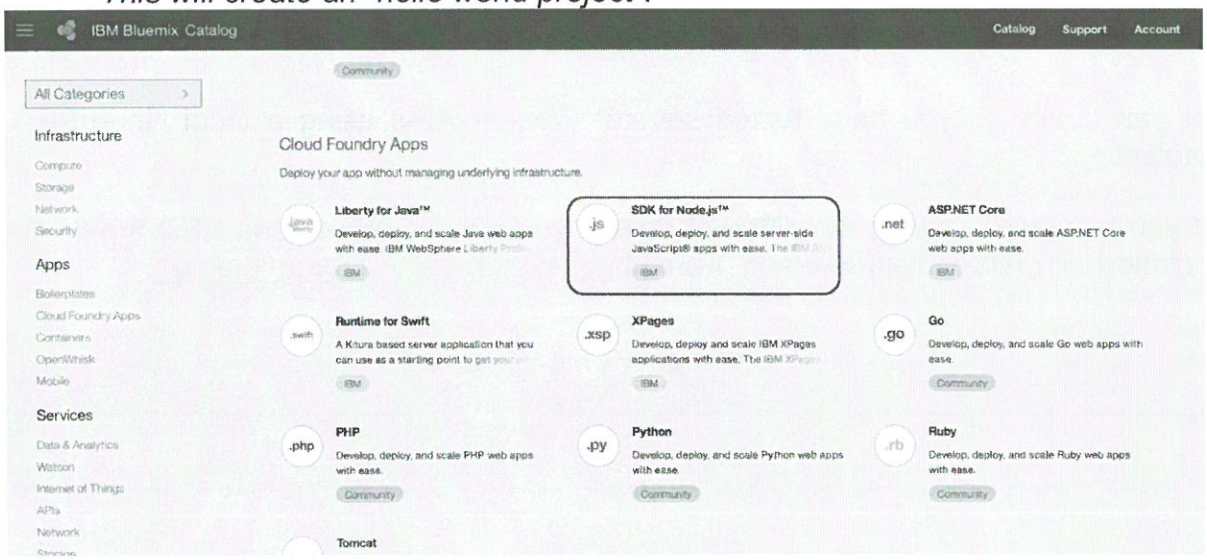
If you had issues wiring the different nodes, you can import in node-RED the finale solution "6TotalSolutionFlow.json" from https://github.com/cllebrun/Interco17

Part 3: Watson APIs and your Node.js web application

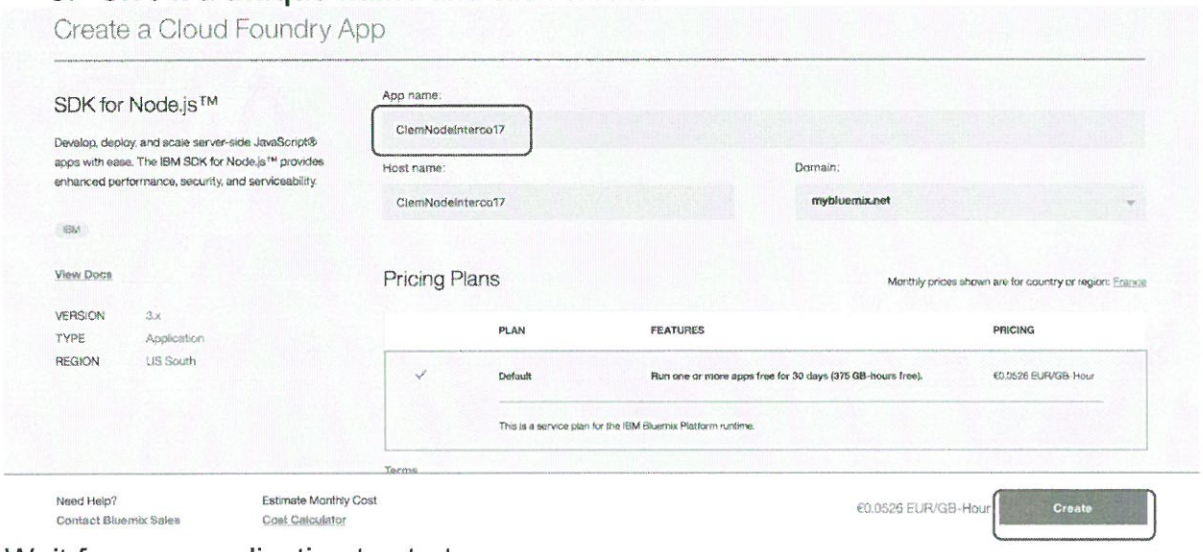
In this part, you will call a Watson API on a basic Node.js application with a User Interface running on Bluemix.

Create Your Hello World Node.js app

1. **Open** Bluemix and go to the Catalog tab
2. From the "Cloud Foundry Apps" section **create** a "**SDK for Node.js**" app. *This will create an "hello world project".*

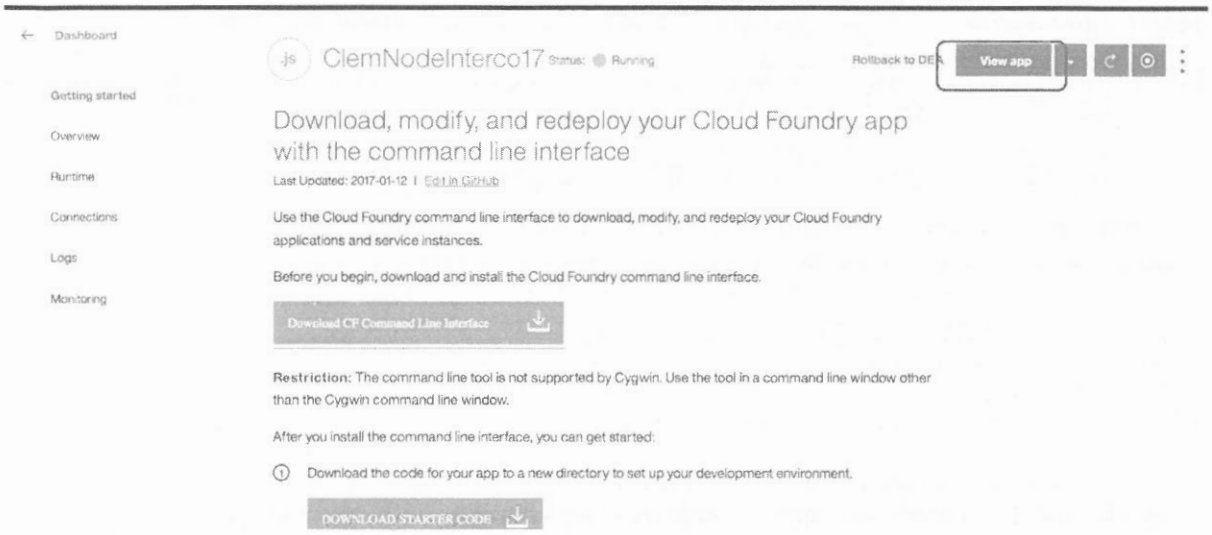


3. Give it a **unique name** and click create:



Wait for your application to start.

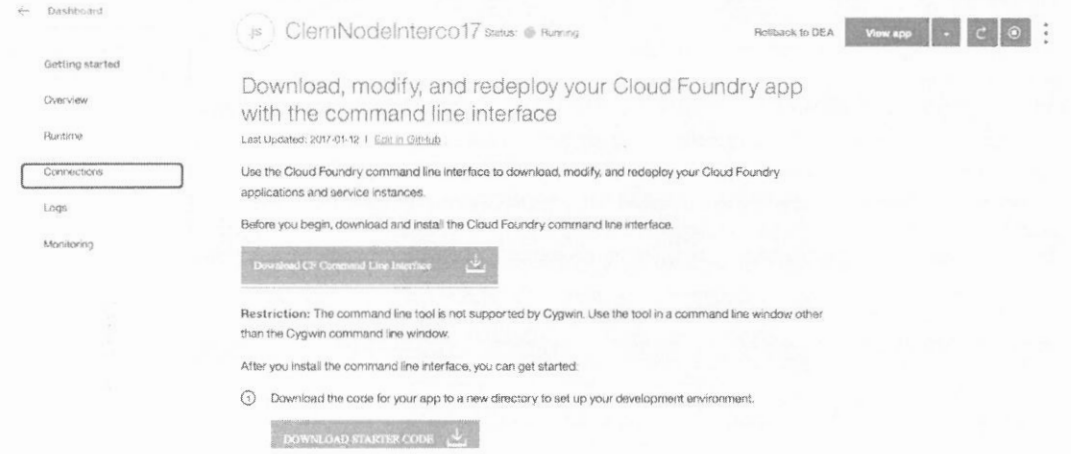
4. When your application is running, you can click on "**View app**" to visualize your Hello World app. This will open a new tab with a public url:



Bind your Text to Speech service to your app

You are now going to bind the Text to Speech service to your Node.js app.

1. In the left menu, select the "**Connections**" tab:



2. Chose "**Connect existing**":

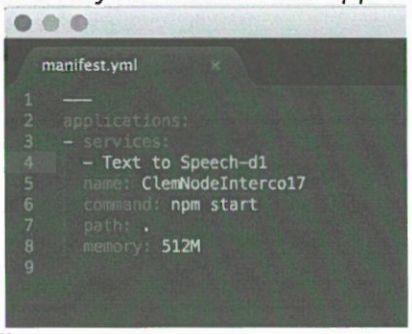
3. Select the **Text to Speech** service you already have created in part 1.
4. **Restage**.

Improve your code to use the service

In this section, you will clone an existing GitHub Node.js project, will configure your manifest file and will push the app to overwrite your Bluemix app.

1. Open your **terminal window** and stop the Node-RED app.
   - Crt + c
2. Locate in the part2 folder
   - cd ..
   - cd ..
   - cd part2
3. Clone the User interface Text to Speech project with git:
   - git clone https://github.com/cllebrun/text-to-speech-nodejs-Interconnect
4. Locate in the project folder
   - cd text-to-speech-nodejs-Interconnect
5. Open the "**manifest.yml**" file with a text editor and edit the name of your app and the name of the Text to Speech service, **corresponding to your Bluemix app:**

In my case, it will be "ClemNodeInterco17" for the app name and "Text to Speech-d1" for the service name (found in your dashboard app overview on Bluemix):



6. **Save** your manifest file.
7. In your **terminal window**, login to Bluemix with the following command:
   - cf https://api.ng.bluemix.net
   - cf login

Enter your login and password when asked.
If asked, chose the org and the space where you created your application node.js.

8. **Push** your app to Bluemix:
   - cf push
9. When your app is started, go back to the Bluemix interface and click on "**View app**"

10. Your node.js web app using your text to speech service is now ready and deployed on the cloud ! Play with it, you can switch languages and change the text input as you like.

Well done ! This is the end of this hands-on lab session. We hope that you have enjoyed it and that it will be useful to you.