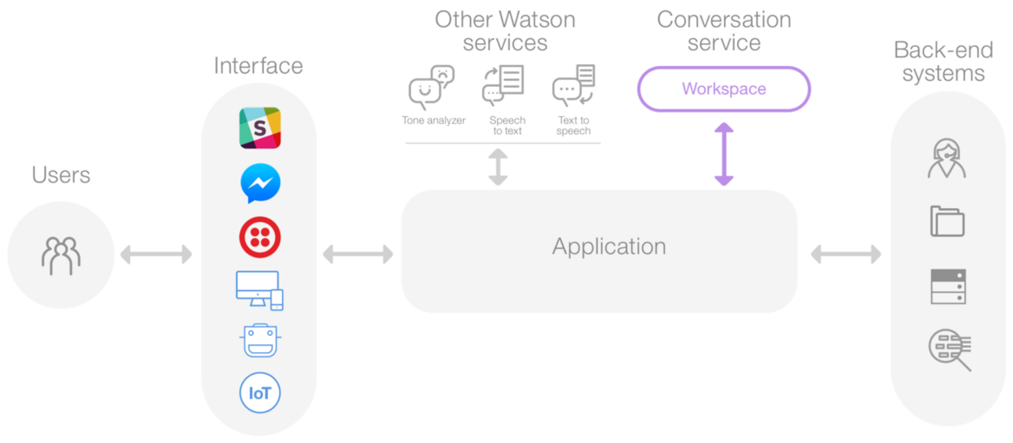
Getting Started with Watson Conversation on IBM Cloud

Note – This is a guideline only. The actual screens may slightly differ.

Topic

Creating a Watson Chatbot or a Virtual Agent using IBM Cloud . The Chatbot will greet our users and answers queries on Mobile broadband and TV , Report a lost phone and other information.



Goal

In this exercise we will use **Telegram** as the interface through which the user can interface with the Conversation Service. We will use **Node-Red** for the Application / Orchestration and we will use **IBM Watson Conversation tool** for modeling the dialog

With the instructions given, you will learn how to create a simple chatbot, powered by Watson Conversation API, which can then connect to virtually any API.

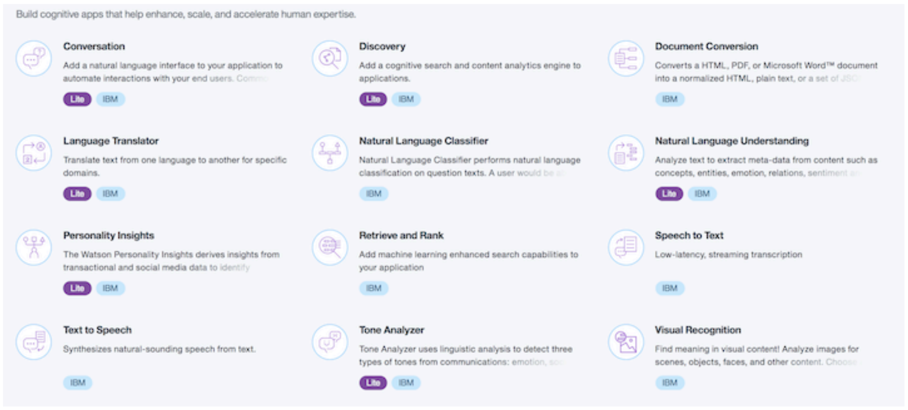
Prerequisite

1. Create an account on IBM Cloud via the following link: https://console.ng.bluemix.net/registration/
2. Download **Telegram** app on your phone, or PC, or run it directly from the browser
3. Android: https://play.google.com/store/apps/details?id=org.telegram.messenger
4. iOS: https://itunes.apple.com/app/telegram-messenger/id686449807

Task 1

The first task is to create an instance of Conversation service instance on IBM Cloud.

1. Make sure that you are logged in to your IBM Cloud account. Click **Catalog** and then click **Services > Watson > Conversation**.



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

Conversation
Add a natural language interface to your application to automate interactions with your end users.
Lite IBM

Discovery
Add a cognitive search and content analytics engine to applications.
Lite IBM

Document Conversion
Converts a HTML, PDF, or Microsoft Word™ document into a normalized HTML, plain text, or a set of
IBM

Language Translator
Translate text from one language to another for specific domains.
Lite IBM

Natural Language Classifier
Natural Language Classifier performs natural language classification on question texts. A user would
IBM

Natural Language Understanding
Analyze text to extract meta-data from content such as concepts, entities, emotion, relations, sentiment
Lite IBM

Personality Insights
The Watson Personality Insights derives insights from transactional and social media data to identify
Lite IBM

Retrieve and Rank
Add machine learning enhanced search capabilities to your application
IBM

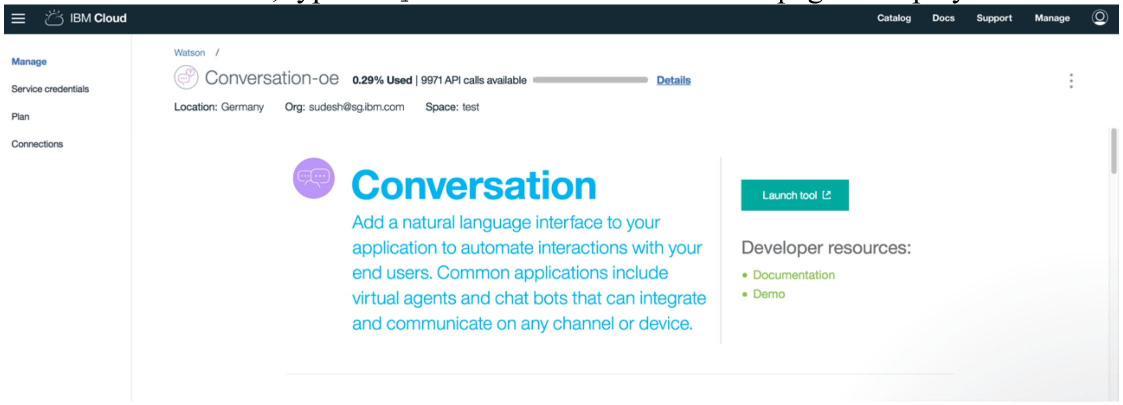
Speech to Text
Low-latency, streaming transcription
IBM

Text to Speech
Synthesizes natural-sounding speech from text.
IBM

Tone Analyzer
Tone Analyzer uses linguistic analysis to detect three types of tones from communications: emotion,
Lite IBM

Visual Recognition
Find meaning in visual content! Analyze images for scenes, objects, faces, and other content.
IBM

For the service name, type `<anyname>`. Click **Create**. This page is displayed:



Conversation-oe **0.29% Used** | 9971 API calls available

Location: Germany Org: sudesh@sg.ibm.com Space: test

Conversation

Add a natural language interface to your application to automate interactions with your end users. Common applications include virtual agents and chat bots that can integrate and communicate on any channel or device.

Developer resources:
- Documentation
- Demo

Step 1: Launch the tool

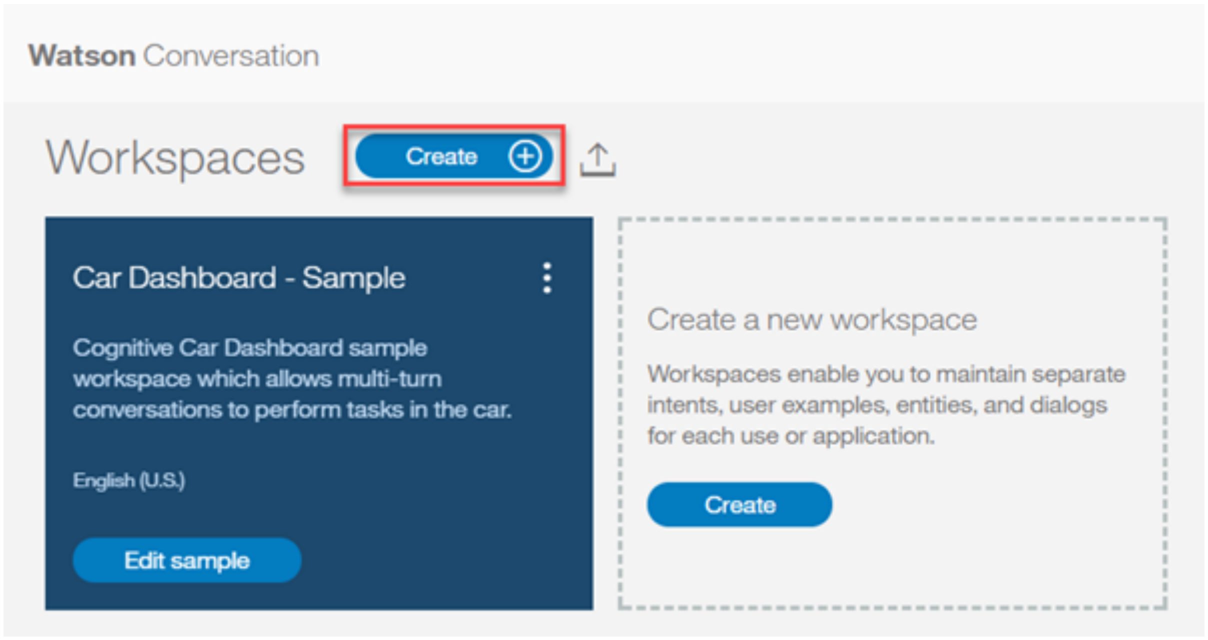
Click **Launch tool** at the top of the page.

Step 2: Create a workspace

Your first step in the Conversation tool is to create a workspace.

A *workspace* is a container for the artifacts that define the conversation flow for an application.

1. In the Conversation tool, click **Create**.
2. Give your workspace the name `Conversation example` and click **Create**. You'll land on the **Intents** tab of your new workspace.



Cognitive Car Dashboard sample workspace which allows multi-turn conversations to perform tasks in the car.

English (U.S.)

Create a new workspace

Workspaces enable you to maintain separate intents, user examples, entities, and dialogs for each use or application.

Step 3: Create intents

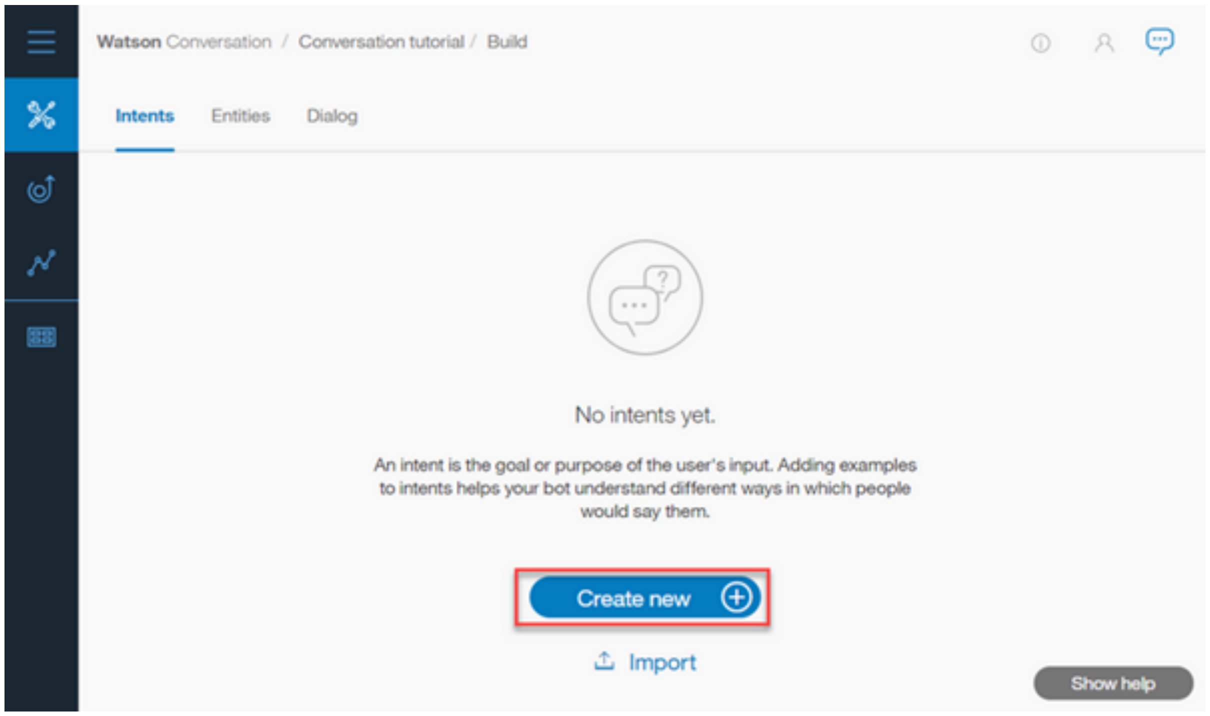
An intent represents the purpose of a user's input. You can think of intents as the actions your users might want to perform with your application.

For this example, we're going to keep things simple and define only two intents: one for saying hello, and one for saying goodbye.

1. Make sure you're on the Intents tab. (You should already be there, if you just created the workspace.)
2. Click **Create new**.
3. Name the intent `hello`.
4. Type `hello` as a **User example** and press Enter.

   Examples tell the Conversation service what kinds of user input you want to match the intent. The more examples you provide, the more accurate the service can be at recognizing user intents.

5. Add four more examples and click **Create** to finish creating the #hello intent:
   - `good morning`
   - `greetings`
   - `hi`
   - `howdy`

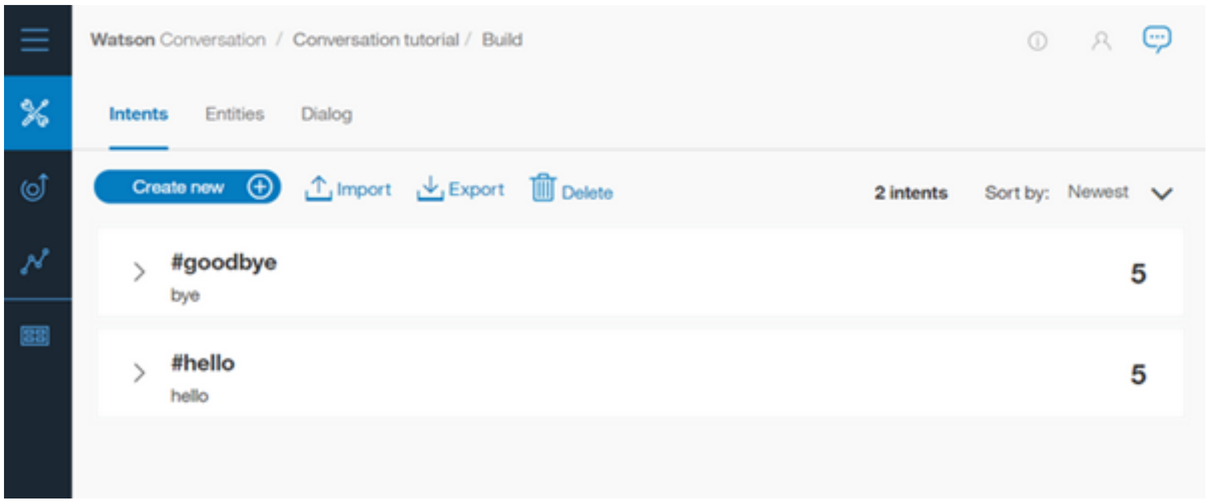


An intent is the goal or purpose of the user's input. Adding examples to intents helps your bot understand different ways in which people would say them.

6. Create another intent named #goodbye with these five examples:
   - `bye`
   - `farewell`
   - `goodbye`
   - `I'm done`
   - `see you later`

Result

You've created two intents, #hello and #goodbye, and provide sample user inputs to train Watson to recognize these intents in your users' input.



#goodbye bye	5
#hello hello	5

Step 4: Build a dialog

A dialog defines the flow of your conversation in the form of a logic tree. Each node of the tree has a condition that triggers it, based on user input.

We'll create a simple dialog that handles our #hello and #goodbye intents, each with a single node.

Adding a start node

1. In the Conversation tool, click the **Dialog** tab.
2. Click **Create**. You'll see two nodes.
    - **Welcome**: Contains a greeting that is displayed to your users when they first engage with the bot.
    - **Anything else**: Contains phrases that are used to reply to users when their input is not recognized.
3. Click the **Welcome** node to open it in the edit view.
4. Replace the default response with the text, `Welcome to the Conversation example!`. Click **Close** to close the edit view.

Result

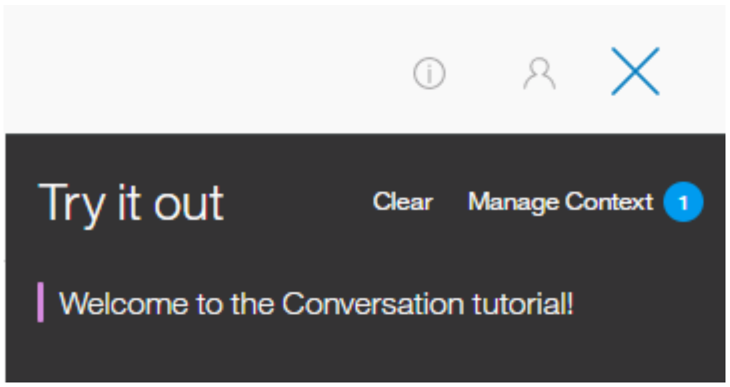
You created a dialog node that is triggered by the `welcome` condition, which is a special condition that indicates that the user has started a new conversation. Your node specifies that when a new conversation starts, the system should respond with the welcome message.

Testing the start node

You can test your dialog at any time to verify the dialog. Let's test it now.

- `

  Click the icon to open the "Try it out" pane. You should see your welcome message.

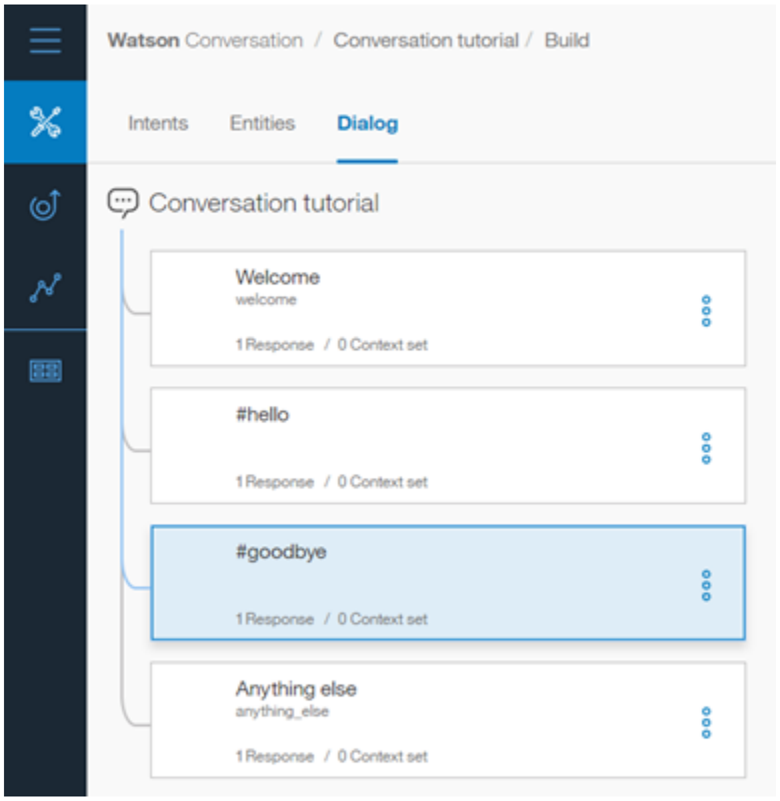


Welcome to the Conversation tutorial!

Adding nodes to handle intents

Now let's add nodes to handle our intents between the `conversation_start` node and the `anything_else` node.

1. Click the **More** icon below the `Welcome` node, and then select **Add node below**.
2. Type `#hello` in the **Enter a condition** field of this node. Then select the **#hello (create new condition)** option.
3. Add the response, `Good day to you.`.
4. Click **X** to close the edit view.
5. Click the **More** on this node, and then select **Add node below** to create a peer node. In the peer node, specify `#goodbye` as the condition, and `OK! See you later.` as the response.

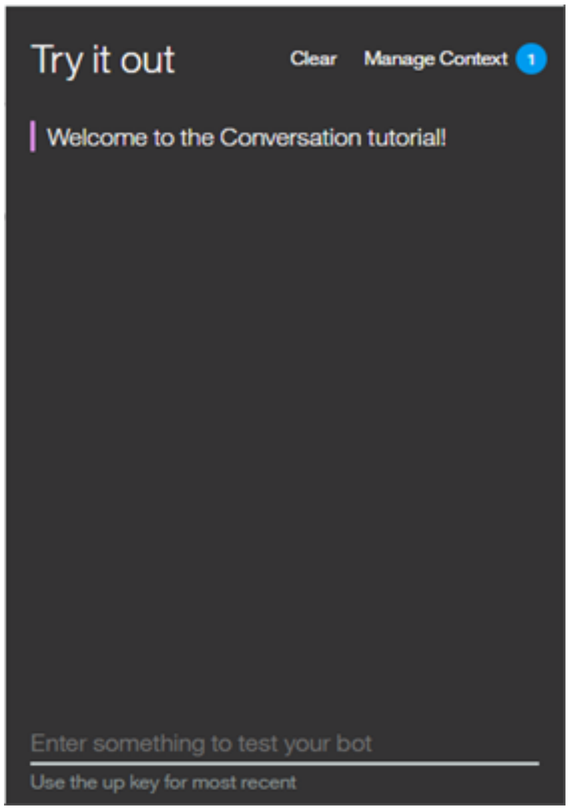

Watson Conversation / Conversation tutorial / Build
Intents
Entities
Dialog
Conversation tutorial
Welcome
welcome
1 Response / 0 Context set
#hello
1 Response / 0 Context set
#goodbye
1 Response / 0 Context set
Anything else
anything_else
1 Response / 0 Context set

Testing intent recognition

You built a simple dialog to recognize and respond to both hello and goodbye inputs. Let's see how well it works.

1. Click the icon to open the "Try it out" pane. There's that reassuring welcome message.
2. At the bottom of the pane, type `Hello` and press Enter. The output indicates that the #hello intent was recognized, and the appropriate response (`Good day to you.`) appears.

3. Try the following input:
   - `goodbye`
   - `howdy`
   - `see ya`
   - `good morning`
   - `sayonara`


Try it out
Clear
Manage Context
1
Welcome to the Conversation tutorial!
Enter something to test your bot
Use the up key for most recent

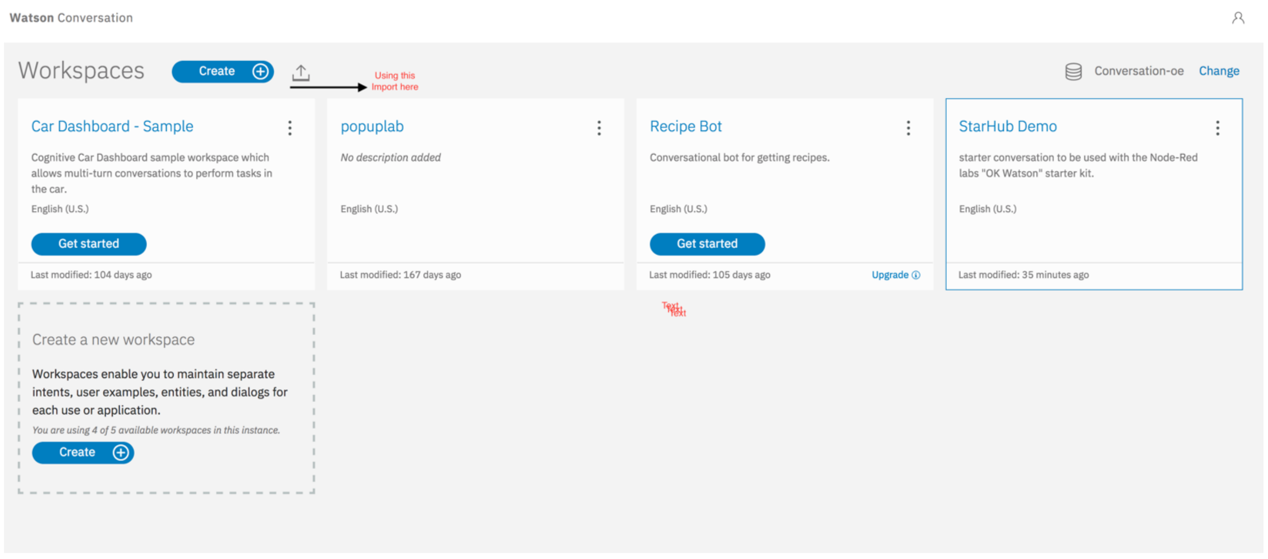
Watson can recognize your intents even when your input doesn't exactly match the examples you included. The dialog uses intents to identify the purpose of the user's input regardless of the precise wording used, and then responds in the way you specify.

Result

That's it. You created a simple conversation with two intents and a dialog to recognize them.

Step 5: Review the sample workspace given.

1. Download the json file from https://github.com/sriramsudesh/workshop
2. Import this into the Conversation Workspace as screen shots shown



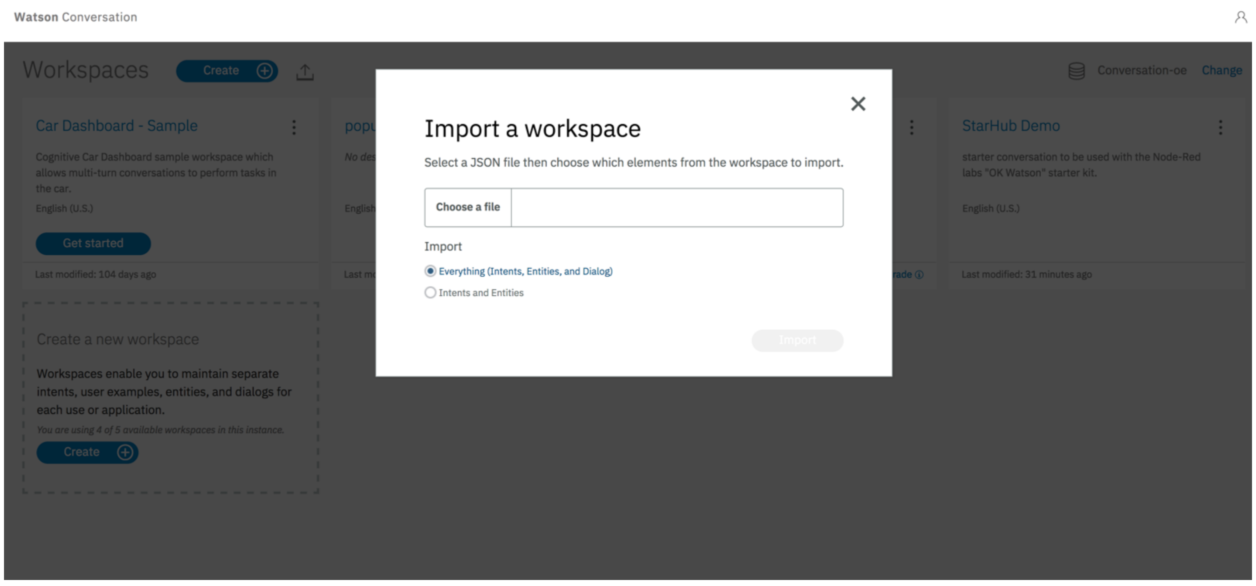
Cognitive Car Dashboard sample workspace which allows multi-turn conversations to perform tasks in the car.
English (U.S.)
Last modified: 104 days ago

No description added
English (U.S.)
Last modified: 167 days ago

Conversational bot for getting recipes.
English (U.S.)
Last modified: 105 days ago

starter conversation to be used with the Node-Red labs "OK Watson" starter kit.
English (U.S.)
Last modified: 35 minutes ago

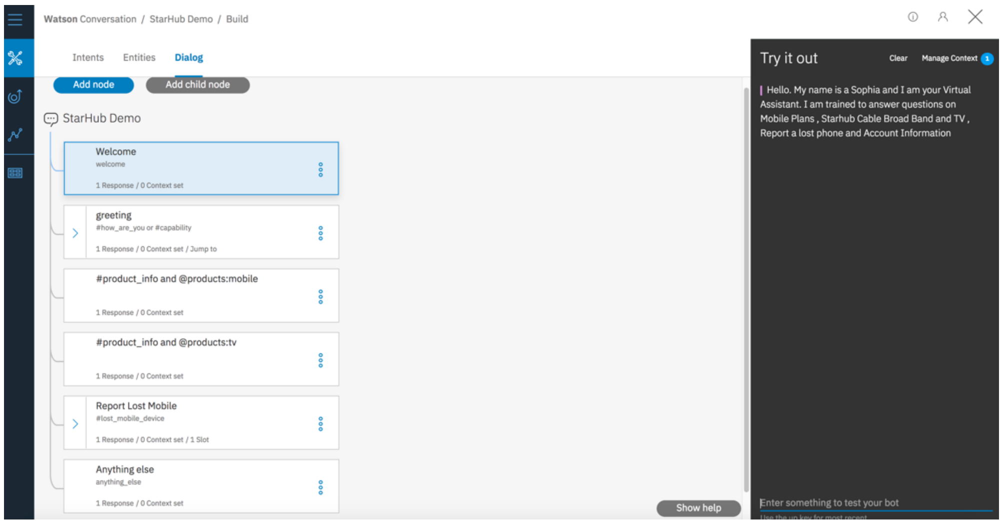
Workspaces enable you to maintain separate intents, user examples, entities, and dialogs for each use or application.
You are using 4 of 5 available workspaces in this instance.



Select a JSON file then choose which elements from the workspace to import.

Import: [✓ Everything (Intents, Entities, and Dialog)] Intents and Entities

Post the Import you will see the full workspace details with various intents , entities and dialog of an existing workspace.



Welcome
welcome
1 Response / 0 Context set

greeting
#how_are_you or #capability
1 Response / 0 Context set / Jump to

#product_info and @products:mobile
1 Response / 0 Context set

#product_info and @products:tv
1 Response / 0 Context set

Report Lost Mobile
#lost_mobile_device
1 Response / 0 Context set / 1 Slot

Anything else
anything_else
1 Response / 0 Context set

Hello. My name is a Sophia and I am your Virtual Assistant. I am trained to answer questions on Mobile Plans , Starhub Cable Broad Band and TV , Report a lost phone and Account Information

Enter something to test your bot
Use the up key for most recent

In the dialog notice the following

1. Use of jumps for Reusability
2. Use of entity identification in case of Product Information Intent
3. Use of Slots in case of Report Lost Mobile intent where series of information is getting collected

You can test this inside the conversation tool by asking a few questions like

a. Hi How are you ?
b. What can you do ?
c. I want to report a lost mobile

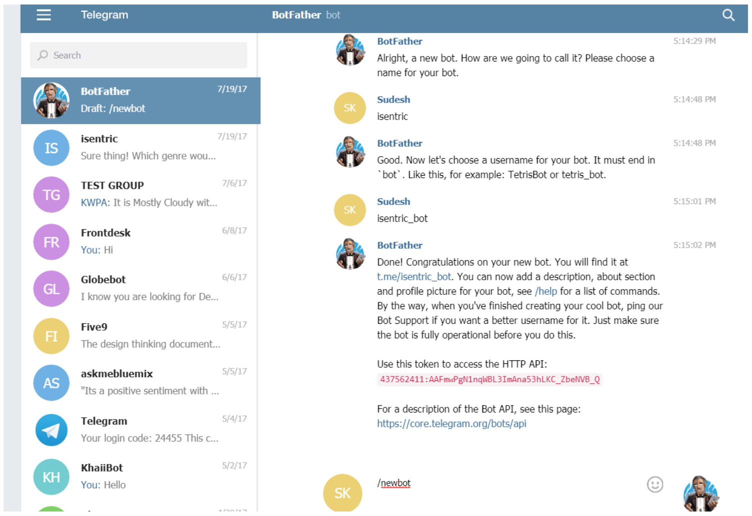
Lets say you would now like this interface to be exposed via an interface like Telegram. You can then follow the rest of the tutorial

TASK 2

You can install Telegram application on your mobile phone. You can go inside the telegram application and then look for BotFather.

Create a new bot on Telegram's BotFather

1. Run the telegram app on your phone or web interface



Telegram | BotFather bot

Search

BotFather 7/19/17
Draft: /newbot

isentric 7/19/17
Sure thing! Which genre wou...

TEST GROUP 7/6/17
KWPA: It is Mostly Cloudy wit...

Frontdesk 6/8/17
You: Hi

Globebot 6/6/17
I know you are looking for De...

Five9 5/5/17
The design thinking document...

askmebluemix 5/5/17
"Its a positive sentiment with ...

Telegram 5/4/17
Your login code: 24455 This c...

KhaiiBot 5/2/17
You: Hello

BotFather 5:14:29 PM
Alright, a new bot. How are we going to call it? Please choose a name for your bot.

Sudesh 5:14:48 PM
isentric

BotFather 5:14:48 PM
Good. Now let's choose a username for your bot. It must end in `bot`. Like this, for example: TetrisBot or tetris_bot.

Sudesh 5:15:01 PM
isentric_bot

BotFather 5:15:02 PM
Done! Congratulations on your new bot. You will find it at t.me/isentric_bot. You can now add a description, about section and profile picture for your bot, see /help for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.

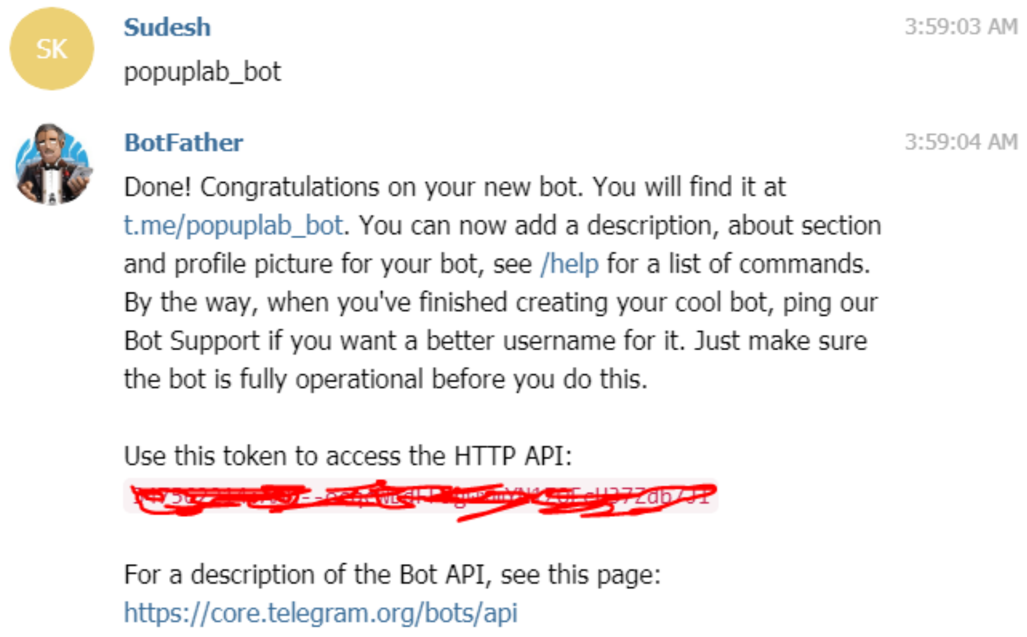
Use this token to access the HTTP API:
`437562411:AAFmwPgN1nqWBL3ImAna53hLKC_ZbeNVB_Q`

For a description of the Bot API, see this page:
https://core.telegram.org/bots/api

/newbot

2. send /newbot command / message to BotFather
3. Enter the name and username of your bot, for example:
4. name: Popuplab (You can give any name here ….)

5. username: Popuplab_bot (This can be any name …)



Sudesh 3:59:03 AM

popuplab_bot

BotFather 3:59:04 AM

Done! Congratulations on your new bot. You will find it at t.me/popuplab_bot. You can now add a description, about section and profile picture for your bot, see /help for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.

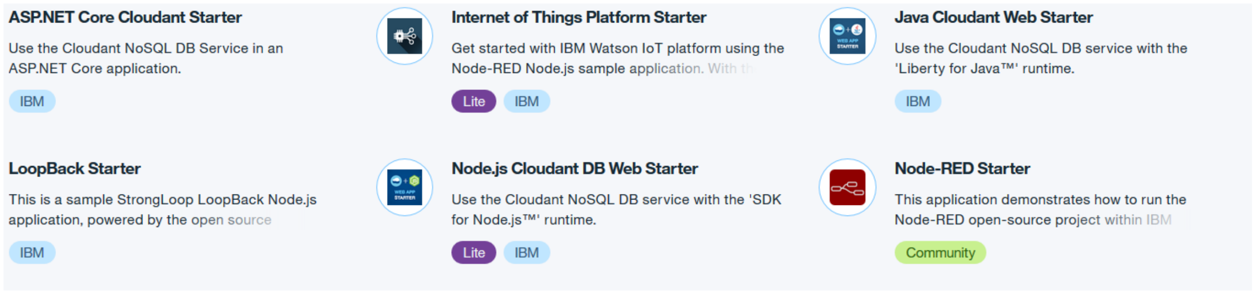
Use this token to access the HTTP API:
[redacted]

For a description of the Bot API, see this page:
https://core.telegram.org/bots/api

6. Once created, you'll be given a token string
7. Save the token on your text editor

Create a Node-RED app using the Node-RED Starter boilerplate

1. Login into the IBM Cloud console via the following link: https://console.ng.bluemix.net/
2. Click **Create App**
3. Search and select **Node-RED Stater** boilerplate



ASP.NET Core Cloudant Starter
Use the Cloudant NoSQL DB Service in an ASP.NET Core application.
IBM

Internet of Things Platform Starter
Get started with IBM Watson IoT platform using the Node-RED Node.js sample application. With th
Lite IBM

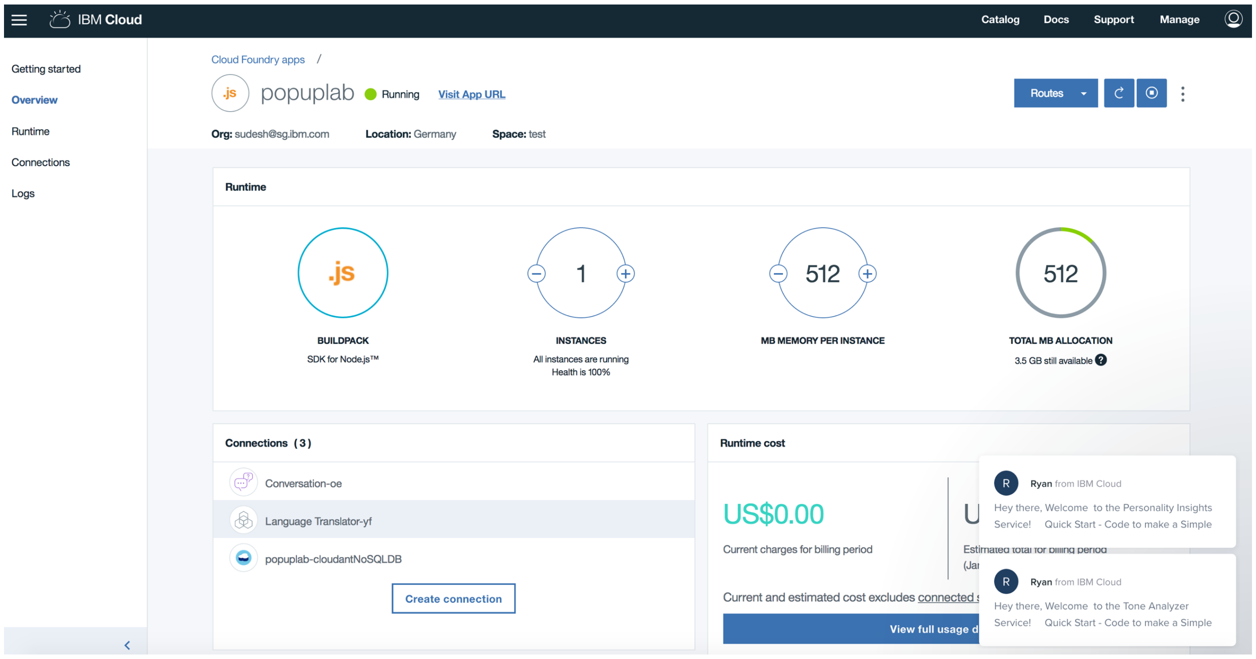
Java Cloudant Web Starter
Use the Cloudant NoSQL DB service with the 'Liberty for Java™' runtime.
IBM

LoopBack Starter
This is a sample StrongLoop LoopBack Node.js application, powered by the open source
IBM

Node.js Cloudant DB Web Starter
Use the Cloudant NoSQL DB service with the 'SDK for Node.js™' runtime.
Lite IBM

Node-RED Starter
This application demonstrates how to run the Node-RED open-source project within IBM
Community

4. Enter the App name and Host name
5. App name: **popuplab (you can give any that is unique) . My url is https://popuplab.eu-de.mybluemix.net/**
6. Host name: **popuplab (you can give any that is unique)**
7. Click **Create**
8. **Click on the Visit App URL**



Org: sudesh@sg.ibm.com Location: Germany Space: test

Runtime

BUILDPACK	INSTANCES	MB MEMORY PER INSTANCE	TOTAL MB ALLOCATION
.js	1	512	512
SDK for Node.js™	All instances are running Health is 100%		3.5 GB still available

Connections (3)

- Conversation-oe
- Language Translator-yf
- popuplab-cloudantNoSQLDB

Runtime cost

US$0.00

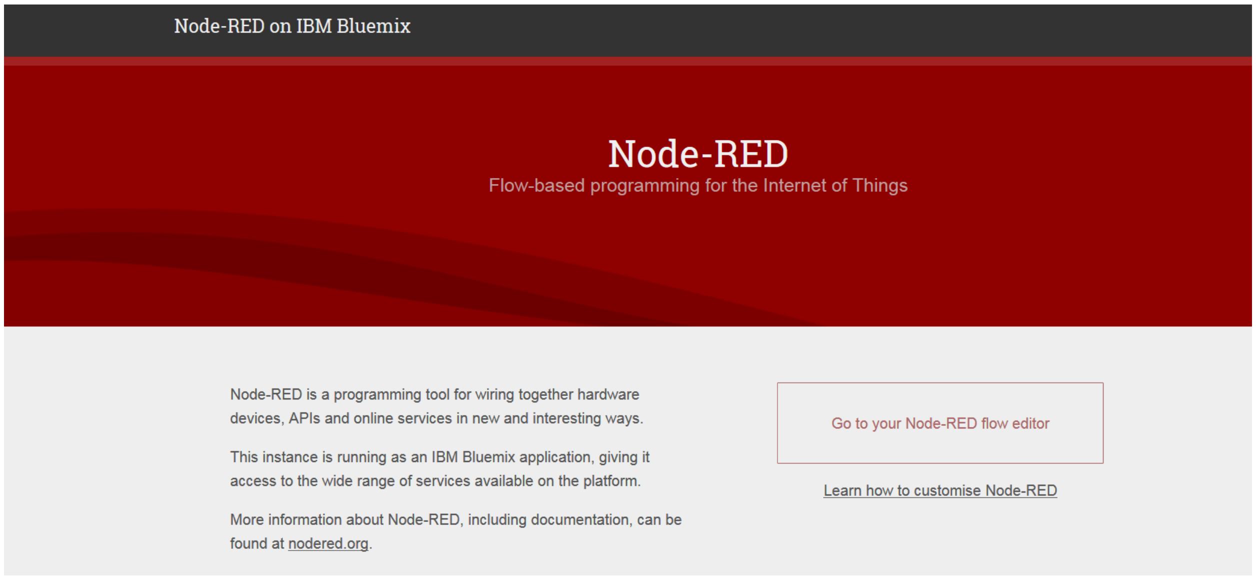
Current charges for billing period

Estimated total for billing period

Current and estimated cost excludes connected s

Ryan from IBM Cloud
Hey there, Welcome to the Personality Insights Service! Quick Start - Code to make a Simple

Ryan from IBM Cloud
Hey there, Welcome to the Tone Analyzer Service! Quick Start - Code to make a Simple


Node-RED on IBM Bluemix

Node-RED

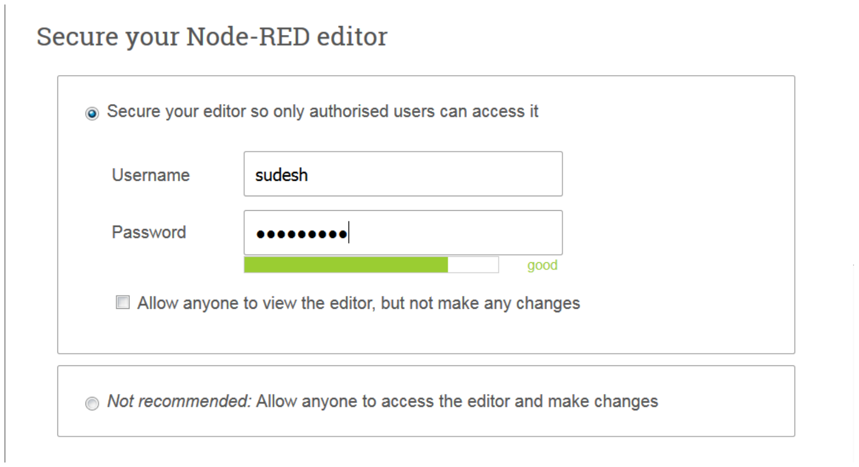
Flow-based programming for the Internet of Things

Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.

This instance is running as an IBM Bluemix application, giving it access to the wide range of services available on the platform.

More information about Node-RED, including documentation, can be found at nodered.org.

Go to your Node-RED flow editor
Learn how to customise Node-RED


Secure your Node-RED editor

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

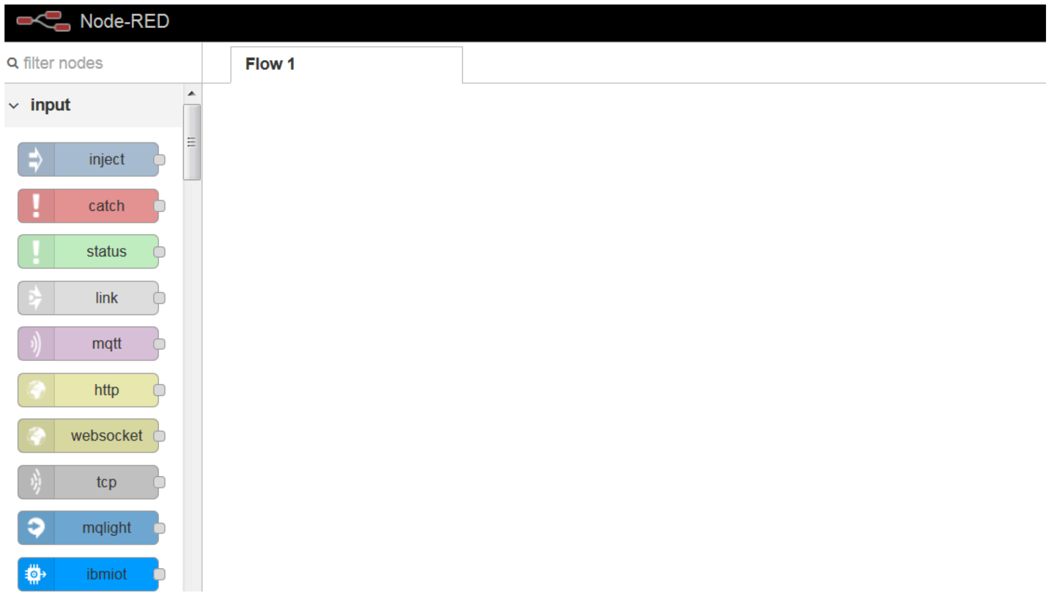
Username: sudesh

Password: ●●●●●●●●● (good)

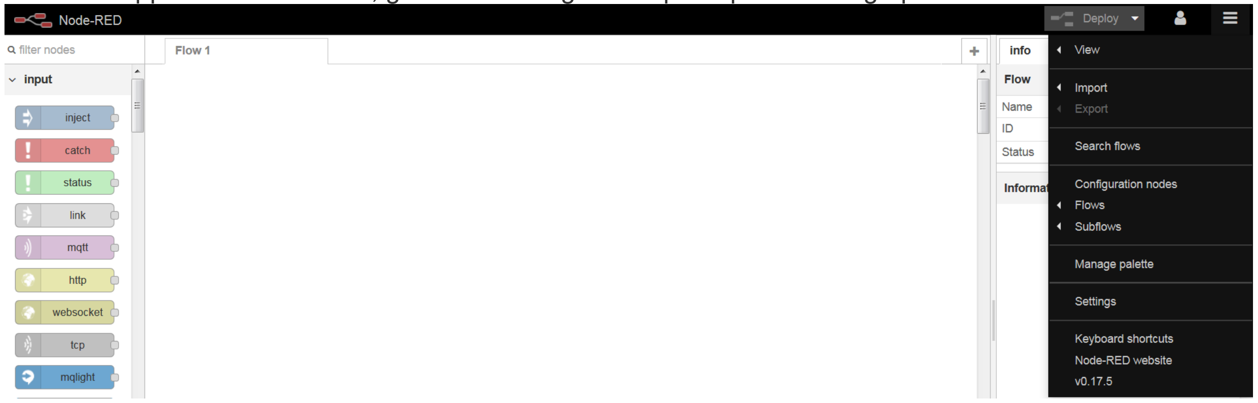
☐ Allow anyone to view the editor, but not make any changes

○ *Not recommended:* Allow anyone to access the editor and make changes

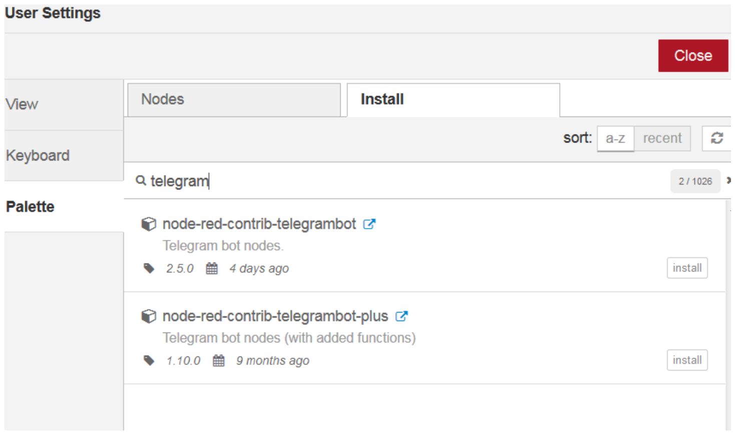
9. Secure your nodered editor by providing username and password
10. You can then login to the editor by entering credentials and you should see the nodered editor



Install Telegram Nodes on Node-RED

1. Once the app has been started, go to the settings and open up the manage palette
2. 

3. Click on the **Install** tab


User Settings

Close

View | Keyboard | Palette

Nodes | Install

sort: a-z recent

telegram 2 / 1026

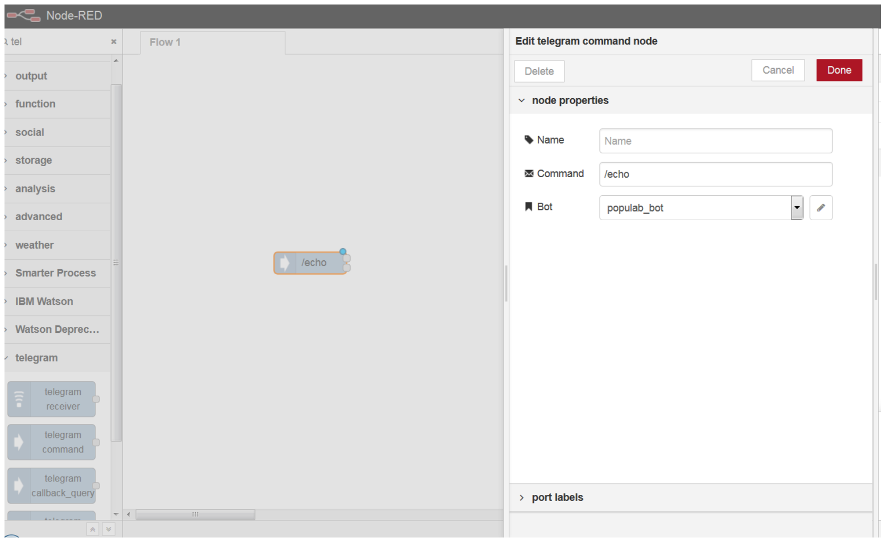
node-red-contrib-telegrambot
Telegram bot nodes.
2.5.0 4 days ago install

node-red-contrib-telegrambot-plus
Telegram bot nodes (with added functions)
1.10.0 9 months ago install

4. Enter **telegram** on the **search module** textbox
5. Click **install** Once the node has been installed, click **Done**
6. Note: There will be 4 telegram nodes installed

Create a Simple Chat Flow

1. For a simple chat flow, we are going to create an echo flow, which will only echo the command sent to the chatbot
2. Drag the **telegram command** node from the pallete onto the workspace
3. Double click the added **Telegram Command** node to edit it
4. Enter **/echo** as the **Command**
5. Click the pencil icon to add the new telegram bot that we have created earlier
6. Enter the following values:
7. Bot-Name: **popuplab_sudesh**
8. Token: **<telegram_bot_token>**


Edit telegram command node

Delete | Cancel | Done

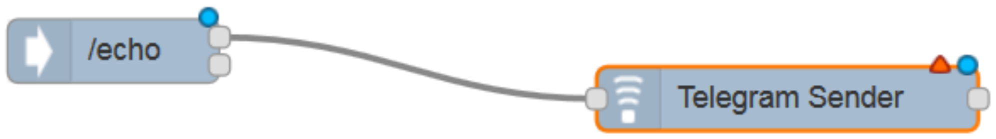
node properties

Name: Name
Command: /echo
Bot: populab_bot

port labels

9. We can ignore the **Users** and **ChatIds** fields for now
10. Click **Add** and then **Done**

11. Drag the **telegram sender** node from the pallete onto the workspace
12. Double click the added **Telegram Sender** node to edit it
13. At the dropdown **Bot** selection, select the bot you have configured earlier, **popuplab_bot**
14. Click **Done**
15. Connect the first output from the **Telegram Command** node to the input of the **Telegram Sender** node



16. Then, click the Deploy button

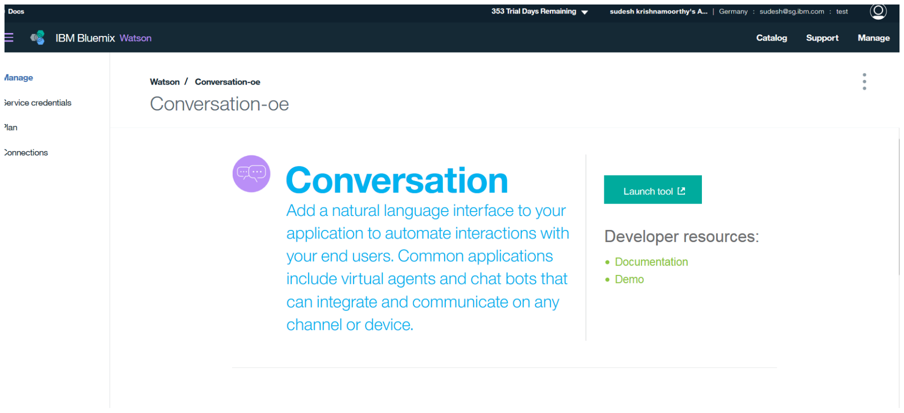
Test the Simple Chat Flow

1. On your **Telegram** app add the bot that you have created as a contact
2. Send the following message:
3.
```
/echo WELCOME TO POPUP LAB
```
4. You should receive the same message back, without the **/echo** command

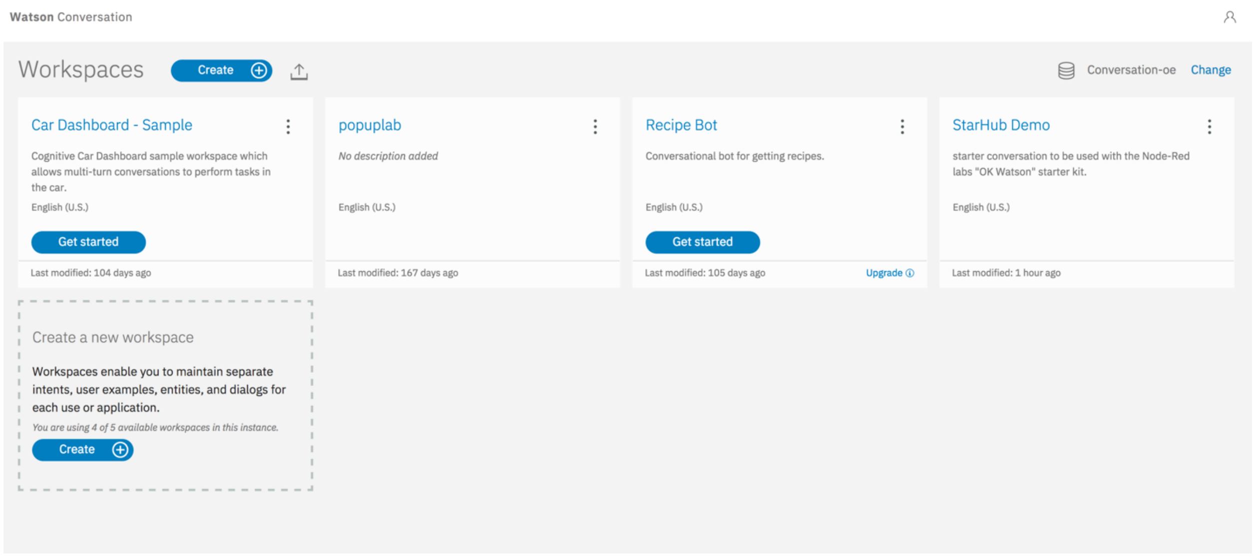
Get Credentials from Watson Conversation Service

From the IBM Cloud , Dashboard , launch the Conversation tool

1. You will see a conversation service details. Click on Launch Tool



Add a natural language interface to your application to automate interactions with your end users. Common applications include virtual agents and chat bots that can integrate and communicate on any channel or device.


Watson Conversation

Workspaces Create

Conversation-oe Change

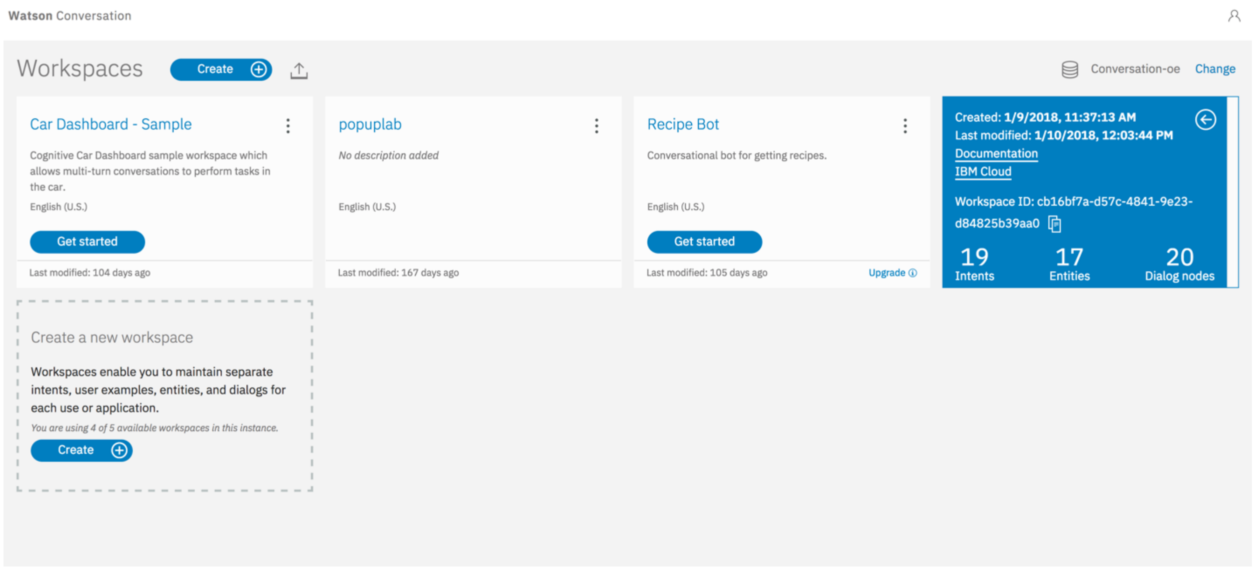
Car Dashboard - Sample
Cognitive Car Dashboard sample workspace which allows multi-turn conversations to perform tasks in the car.
English (U.S.)
Get started
Last modified: 104 days ago

popuplab
No description added
English (U.S.)
Last modified: 167 days ago

Recipe Bot
Conversational bot for getting recipes.
English (U.S.)
Get started
Last modified: 105 days ago Upgrade

StarHub Demo
starter conversation to be used with the Node-Red labs "OK Watson" starter kit.
English (U.S.)
Last modified: 1 hour ago

Create a new workspace
Workspaces enable you to maintain separate intents, user examples, entities, and dialogs for each use or application.
You are using 4 of 5 available workspaces in this instance.
Create


Watson Conversation

Workspaces Create

Conversation-oe Change

Car Dashboard - Sample
Cognitive Car Dashboard sample workspace which allows multi-turn conversations to perform tasks in the car.
English (U.S.)
Get started
Last modified: 104 days ago

popuplab
No description added
English (U.S.)
Last modified: 167 days ago

Recipe Bot
Conversational bot for getting recipes.
English (U.S.)
Get started
Last modified: 105 days ago Upgrade

Created: **1/9/2018, 11:37:13 AM**
Last modified: **1/10/2018, 12:03:44 PM**
Documentation
IBM Cloud
Workspace ID: cb16bf7a-d57c-4841-9e23-d84825b39aa0
19 Intents 17 Entities 20 Dialog nodes

Create a new workspace
Workspaces enable you to maintain separate intents, user examples, entities, and dialogs for each use or application.
You are using 4 of 5 available workspaces in this instance.
Create

2. Click on the top-left menu and select **Credentials**
3. Copy the **Workspace ID** onto the notepad

Integrate Watson Conversation Service into Node-RED

1. Open the Node-RED flow created earlier

2. Add another flow with the following connection sequence of nodes and connect the output of the node to the input of the next node:

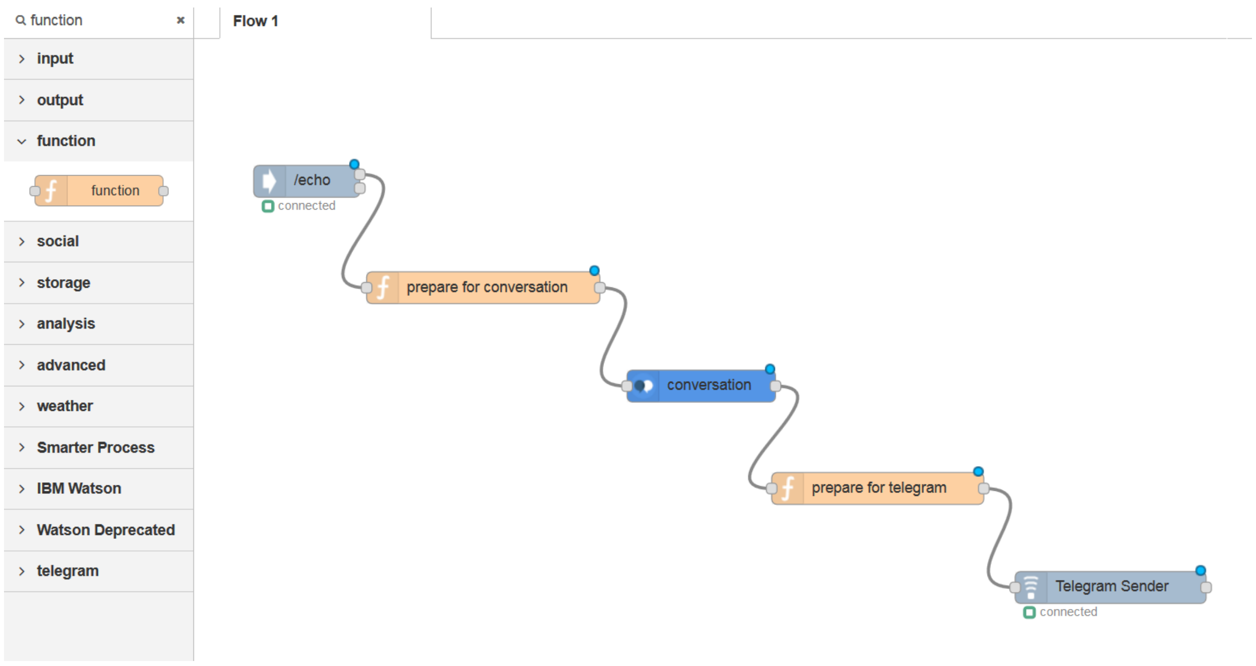
telegram command

function

conversation

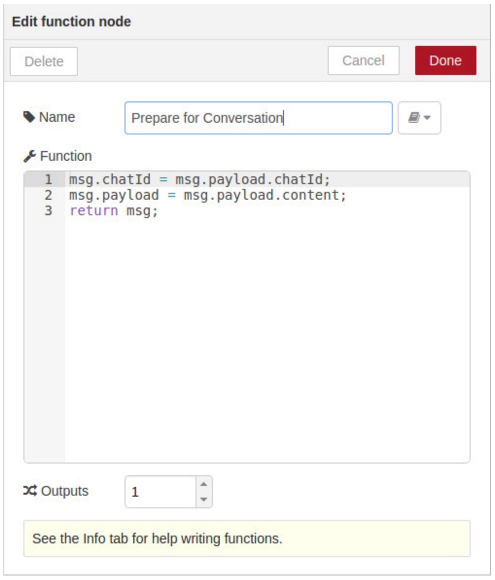
function

telegram sender



3. Configure both telegram nodes to use the **popuplab** and set the Telegram Command to use the **/watson** command
4. Double click the 1st function node to configure it
5. Enter **Prepare for Conversation** as the **Name**
6. Enter the following as the **Function** and click **Done**

```
msg.chatId = msg.payload.chatId;
msg.payload = msg.payload.content;
return msg;
```



Name: Prepare for Conversation

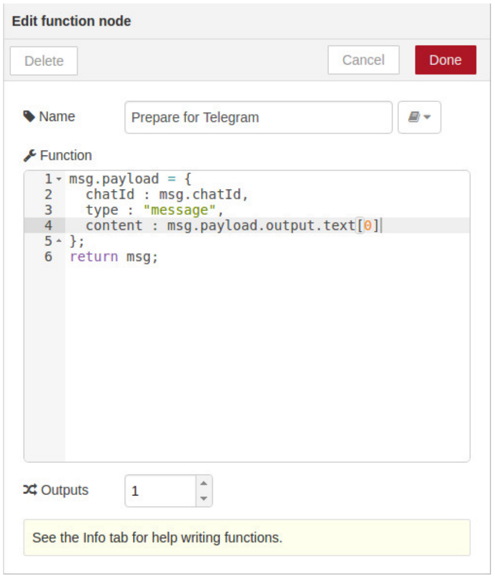
```
msg.chatId = msg.payload.chatId;
msg.payload = msg.payload.content;
return msg;
```

Outputs: 1

See the Info tab for help writing functions.

7. Double click the **Conversation** node to configure it
8. Enter the **Workspace ID** that you have copied earlier, then click **Done**
9. Double click the 2nd **Function** node to configure it
10. Enter **Prepare for Telegram** as the **Name**
11. Enter the following as the **Function** and click **Done**

```
msg.payload = {
  chatId : msg.chatId,
  type : "message",
  content : msg.payload.output.text[1]};
return msg;
```



Name: Prepare for Telegram

```
msg.payload = {
  chatId : msg.chatId,
  type : "message",
  content : msg.payload.output.text[0]
};
return msg;
```

Outputs: 1

See the Info tab for help writing functions.

---

1. Then click the Deploy button

Test Watson Conversation Integration with Node-RED on Telegram

1. On your Telegram app send the following message to the chat bot:

```
2. /watson Hello
```

---

3. You should receive **hello** message back
4. You can then replace the Telegram Receiver instead of using Telegram Command