This is a guide to setting up a sample chatbot which integrates Microsoft Bot Framework with UiPath Orchestrator. The chatbot maps user intents to Orchestrator processes, triggers jobs for those processes, and adapts the conversation based on the job results. This integration of chatbot technology with UiPath eliminates the need to have human workers interact with customers and manually trigger UiPath jobs.

1. Download the reference code

Run command: Git clone https://github.com/UiPath/Chatbot-Samples.git

The code for the sample is in Chatbot-Samples-master/BotFramework/SampleBot/

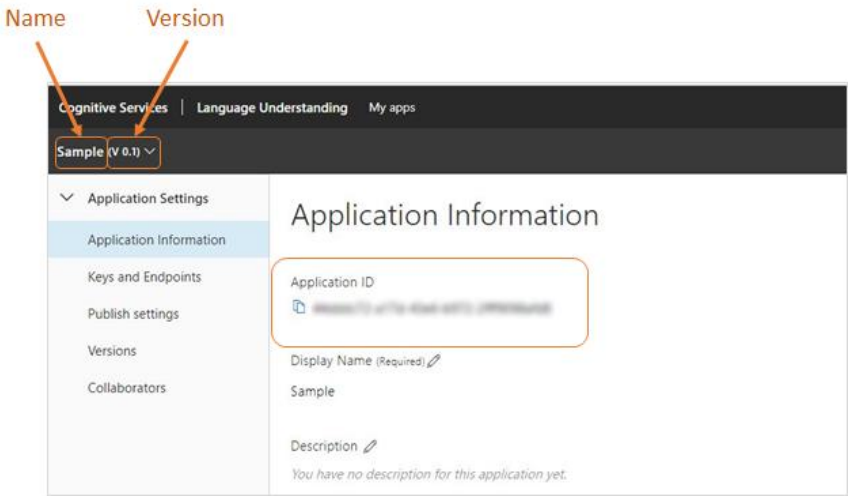
This reference code includes:

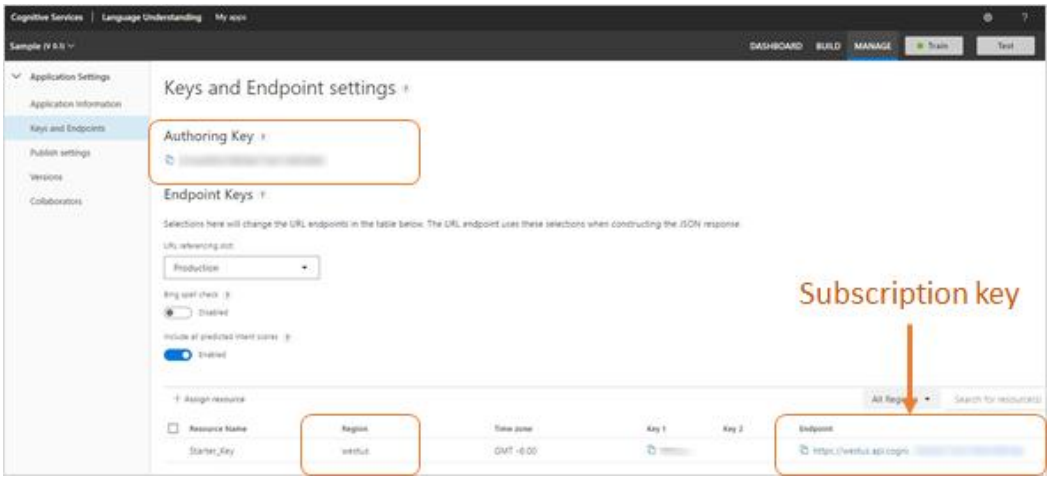
a. A sample LUIS application to get you started. This sample chatbot represents a virtual agent that helps with online retail purchases, and can be found in Chatbot-Samples-master/BotFramework/Data/Luis
b. Processes to be run by UiPath robots, based on intents that the chatbot identifies.

2. [Optional] Create a LUIS application

You may skip this step if you choose to use the sample chatbot provided with the reference code.

a. Navigate to https://www.luis.ai/applications and create a LUIS application. Alternatively, you can also use the BotBuilder suite to create the LUIS app Helpful command line documentation can be found here.
b. Update the Chatbot-Samples-master/BotFramework/SampleBot/samplebot.bot file with information about your LUIS app, taken from the luis.ai site





3. Create UiPath processes

a. Create a machine that will host your robots(s).
b. Create an environment by navigating to Robots -> Environments and clicking the '+' button
c. To use the processes provided in this sample:
    i. Navigate to Processes -> Packages and click the upload button
    ii. Upload the sample robots in Chatbot-Samples-master\BotFramework\Data\Packages
d. Next, create a process for each package by navigating to Processes and clicking on the '+' button

4. Create UiPath robots

a. Navigate to Robots and click the '+' button to create one or more robots
b. Robot types must be **development** or **unattended**.
c. Finally, make sure that your robot is in the same environment as the processes. To add a robot to an environment, navigate to Environments, hover over the environment that contains your processes and from the menu button on the right, select Manage. From here select the robot that you'd like to add to it and click Update

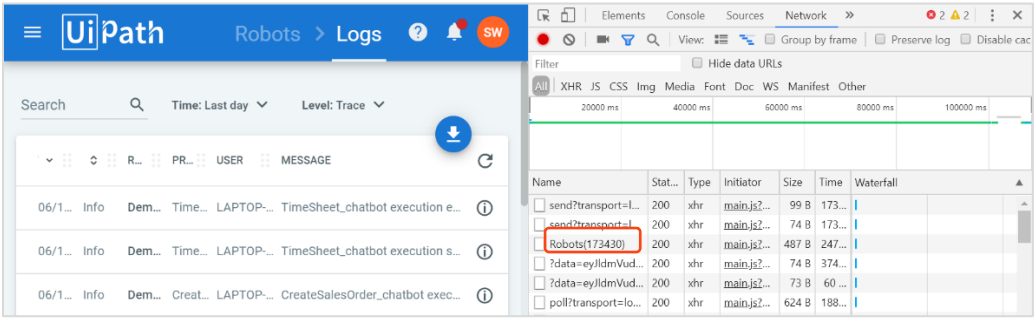
5. Update the chatbot appsettings file

If you're using the sample LUIS chatbot, the appsettings file can be found at:
Chatbot-Samples-master\BotFramework\SampleBot\appsetting.json

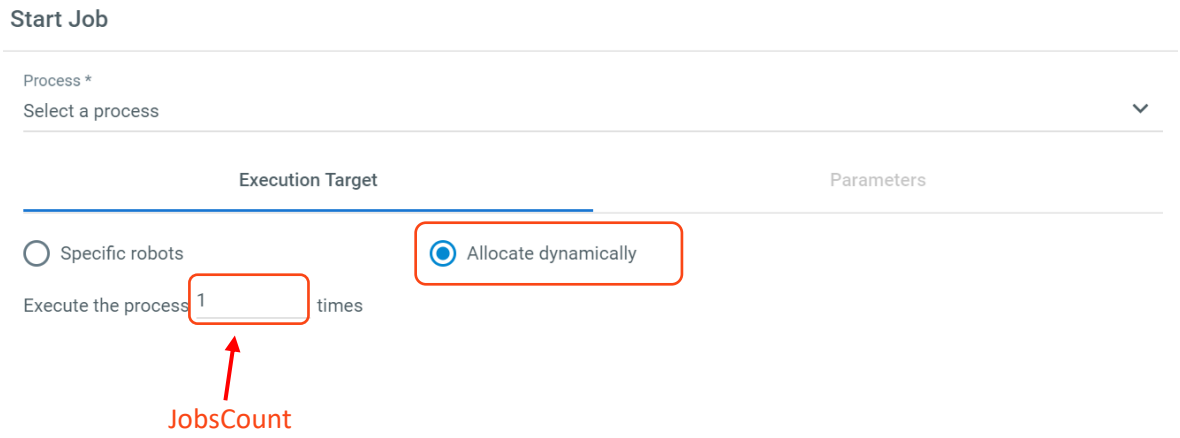
Note that this file displays information in plain text format. If you're using this sample for production purposes, you should ensure that sensitive information in this file is secured.

In this file:

a. *Botfilepath* field points to a .bot file. If you're using the sample LUIS app, this is the samplebot.bot
b. In production, it also has *botFileSecret*.
c. If using Orchestrator **on premise:**
    i. update login info- *TenancyName* (if you have one), *UsernameOrEmailAddress*, *Password* and *Baseurl*.
    ii. *Authmode*: "Basic".
    iii. *BaseUrl* points to the orchestrator URL that has basic auth enabled.
d. If using Orchestrator on **Cloud**:
    i. *Authmode*: *"Cloud"*.
    ii. Add *RefreshToken, ServiceInstanceLogicalName, AccountLogicalName*. Complete all steps **before** "Making Orchestrator API Calls" section.
    iii. Set *baseUrl* to https://platform.uipath.com.
    iv. Remove basic auth related setting (Optional)
e. *StatusCheck* is the timeout setting (in milliseconds) on an orchestrator call
f. *Strategy* refers to how the process will run. The choices are:
    i. "Specific", meaning the process will run on one specified robot
    ii. "JobsCount", meaning available robots will be automatically selected
g. If "Specific", then *RobotIds* must be specified in appsettings.json.
    i. *RobotIds* must be an **array** of Id's, such as [1230, 2987, 3430].
    ii. Two methods of finding RobotId's:
        1. DB query: `SELECT Id FROM [UiPath].[dbo].[Robots] WHERE Name = '<Robot Name>'`
        2. Go to Robots -> View Log, inspect the webpage, navigate to 'Network' tab, and click on 'Robots' element (see image below).



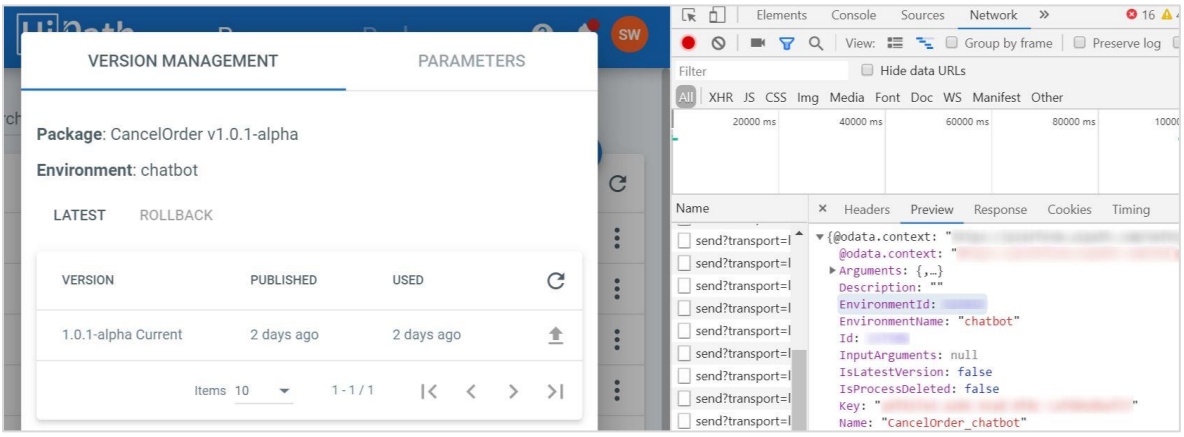
If "JobsCount", then 'Jobscount' must be set to at least '1' (See image for context).



h. Fill out *ProcessKeys* information as follows:
   i. *Process* maps to the name of the process that the chatbot will execute. This should not change without a code update.
   ii. *Key* is the release key for each process.
   There are two ways to find the release key:
      1. DB query

```
SELECT [Key], ProcessKey
FROM [UiPath].[dbo].[Releases]
WHERE ProcessKey in ('CreatePurchaseOrder',
                     'CreateSalesOrder',
                     'CancelOrder',
                     'GetItems')
```

      2. Open your web inspector (right click -> inspect) on the Orchestrator webpage, navigate to 'Network' tab in the inspector, view a process, and click on 'Releases' element (see image below).



6. Run the chatbot

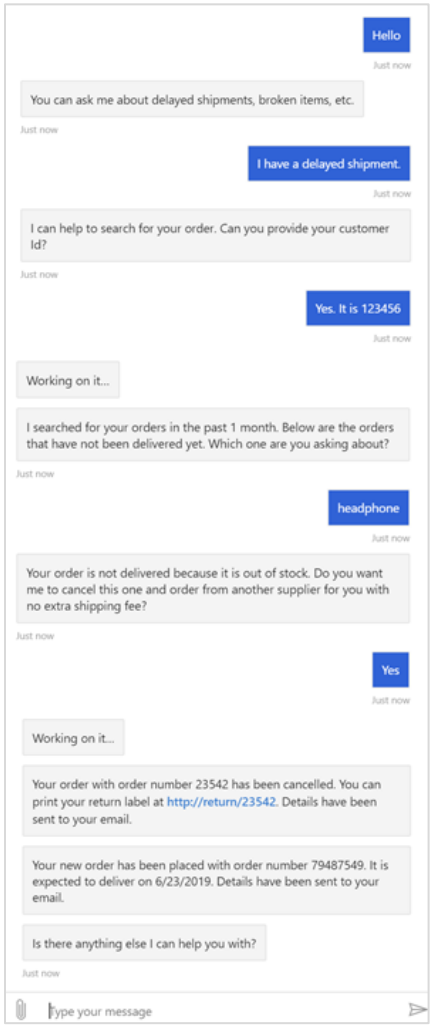
a. Build and run the solution (.sln file) in Chatbot-Samples-maste/BotFramework
b. If On Premise, make sure the version of Orchestrator referenced in appsetting.json is running.
c. Open the Bot Emulator
    i. In Bot Emulator, open Chatbot-Samples-master/BotFramework/SampleBot/samplebot.bot
    ii. Set your endpoint in the emulator to the web address specified in the terminal window that is displayed when you run your .sln file. Append "/api/messages" to the endpoint web address. It should look something like: "http://localhost:53597/api/messages").

7. Finishing up

At this point, your bot should be up and running. If you are using our sample LUIS files, your bot represents a contact center for online retail purchases. You can try talking with the bot about delayed orders, broken items, etc.

Note that the purpose of this sample is not to demonstrate a highly intelligent chatbot, but rather to demonstrate our integration of chatbot technology with UiPath's technology. The intelligence of the bot can be improved by adding detail in LUIS for Natural-Language-Processing capabilities and by adding to the code for conversation strategy (how the bot responds to user messages).

Below is a sample conversation with our bot:



8. Security

The sample code is to show case the key technology detail for the integration, and it is not meant to be used in production. Your use of this code sample in production is at your own risk and UiPath is not responsible for any inaccuracies or security issues.

Below are a few things we highly recommend:

a. Don't put secrets in the appsettings.config as plain text, consider using key management services, such as KeyVault for sensitive data
b. Grant bot just enough permission to execute the list of jobs and always audit bot activities in orchestrator
c. Keep the data sensitivity level as low as possible
d. Apply data protection in transit and in storage
e. Follow the guidelines below:
    i. Secure your bot: https://docs.microsoft.com/en-us/azure/bot-service/dotnet/bot-builder-dotnet-security?view=azure-bot-service-3.0
    ii. LUIS security https://docs.microsoft.com/en-us/azure/cognitive-services/luis/luis-concept-security
    iii. .net core security https://docs.microsoft.com/en-us/aspnet/core/security/?view=aspnetcore-2.2
    iv. Azure security https://docs.microsoft.com/en-us/azure/security/