Introduction to Artificial Intelligence

Lab 4 – Getting Started with Bots

Overview

Bots are digital agents that users can interact with through chat sessions, social media platforms, email, or other channels. At its simplest, a bot receives and responds appropriately to messages from users. Bots can be very simple, with a limited set of well-defined interactions, or they can leverage AI capabilities, such as those provided by the Cognitive Services APIs, to respond intelligently to a comprehensive range of inputs.

In this lab, you will create simple Bots by using the Microsoft Azure Bot Service.

What You'll Need

To complete this lab, you will need the following:

- A Microsoft account (for example, an *outlook.com*, *live.com*, or *hotmail.com* address)
- A Microsoft Azure subscription
- A Windows, Linux, or Mac OS X computer
- The lab files for this course

Note: To set up the required environment for the lab, follow the instructions in the setup guide for this course.

Publishing a QnA Service

Bots are often used to answer frequently asked questions. You can train a QnA service with a knowledge base, and then use a Bot to enable users to ask questions that can be answered from the knowledge base.

Create a QnA Service

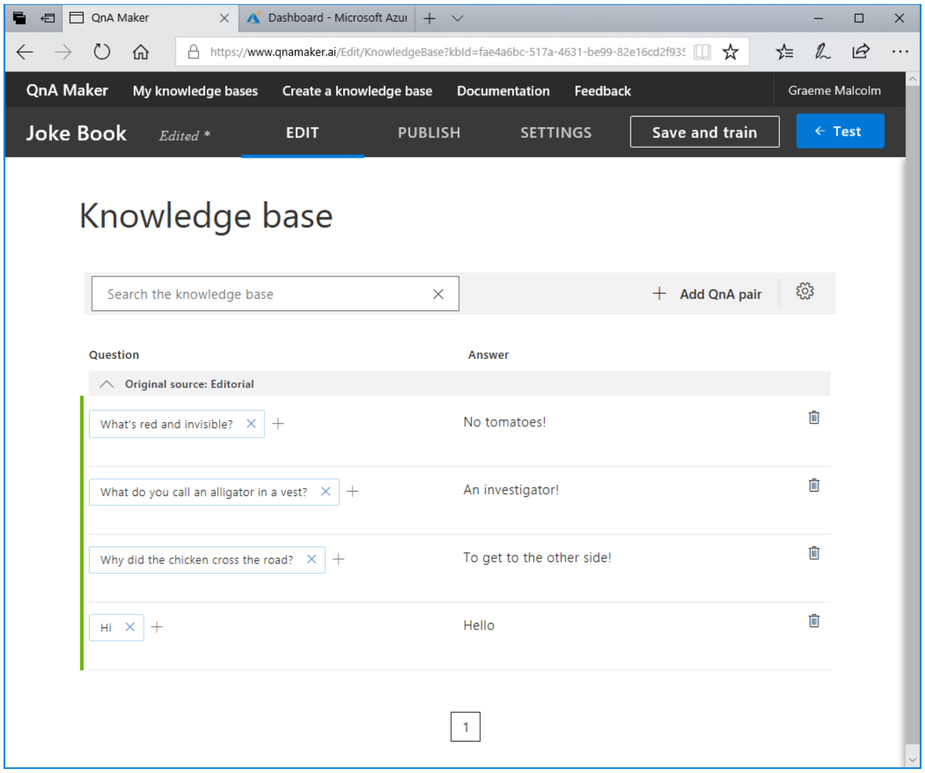
1. Open a new browser tab and navigate to https://qnamaker.ai.
2. Sign in using the Microsoft account associated with your Azure subscription. If this is the first time you have signed into the QnA Maker site, you may need to grant the app permission to access your info and agree to the terms and privacy statement (use the links to view these).
3. After you have signed in, click **Create a knowledge base**.
4. In step 1, click **Create a QnA service**. This will take you to the Azure portal in a new browser tab.
5. In the Azure portal, create a QnA Maker service with the following settings:
    - **Name**: JokeBook
    - **Subscription**: Your Azure subscription

- **Management pricing tier**: F0.
- **Resource Group**: Select the existing resource group you used in the previous labs (or create a new one).
- **Search pricing tier**: F
- **Search location**: Choose a location near you
- **App name**: Create a unique app name in the format *JokeBook12345*.
- **Website location**: Choose a location.
- **App insights**: Enable
- **App insights location**: Choose a location

6. Wait for the service to be created. Then return to the QnA Maker portal and refresh the page.
7. In Step 2, select your **Microsoft Azure Directory ID**, **Azure subscription name**, and **Azure QnA Service**.
8. In Step 3, name your knowledge base **Joke Book**.
9. Skip step 4 (which enables you to populate your knowledge base with items from an existing FAQ).
10. In Step 5, click **Create your KB**.
11. When your Q&A service has been created, it is empty.
12. Click **Add new QnA pair** to add a row to the knowledge base table, and then in the new blank **Question** cell, enter *Hi*, and in the **Answer** cell type *Hello*
13. Repeat the previous step to add the following question and answer pairs:

Question	**Answer**
Why did the chicken cross the road?	To get to the other side!
What do you call an alligator in a vest?	An investigator!
What’s red and invisible?	No tomatoes!

14. After you’ve added these question and answer pairs, ensure that your service looks like this:



https://www.qnamaker.ai/Edit/KnowledgeBase?kbId=fae4a6bc-517a-4631-be99-82e16cd2f93

Joke Book *Edited **

Knowledge base

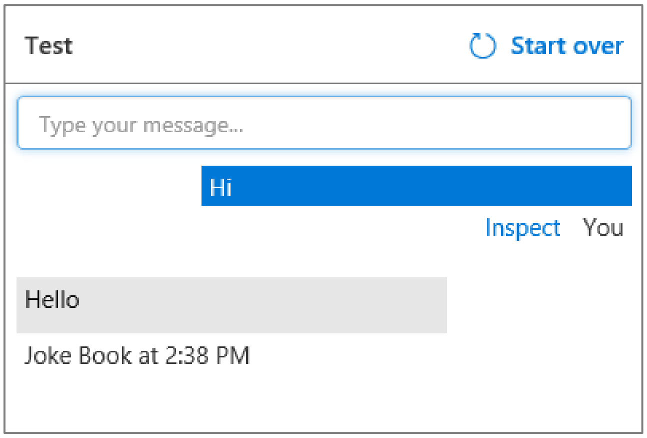
Question	Answer
Original source: Editorial	
What's red and invisible?	No tomatoes!
What do you call an alligator in a vest?	An investigator!
Why did the chicken cross the road?	To get to the other side!
Hi	Hello

15. Click **Save and retrain**.

Test the QnA Service

Now that you’ve defined a knowledge base for your QnA service, you can test it.

1. In the QnA Maker page for your service, click **Test**.
2. Enter the message ***Hi*** and wait for the response ***Hello*** as shown here:
3.
4. This displays a chat interface in which you can test responses to questions. The initial exchange of *hi* and *hello* has been completed as an example:



Hi

You

Hello

Joke Book at 2:38 PM

5. In the chat window, enter the message *hi there!*, and note that the response *hello* is returned. *Hi there!* has been interpreted as *hi*.
6. Enter the message *Why did the chicken cross the road?*, and verify that the answer *To get to the other side!* is returned.

Publish the QnA Service

After you’ve created the knowledge base for your service, you can publish it so that it can be consumed by a bot.

1. In the QnA Maker page for your service, click **Publish**
2. On the **Joke Book** page, click **Publish**.
3. After your service has been deployed, note the sample HTTP request that can be used to call your service from a bot. You will need elements of this later.

Creating a Bot

You can get started creating bots directly in the Azure portal by using the Azure Bot Service.

In this exercise, you’ll create a simple bot that uses the QnA service you created in the previous exercise to answer user questions.

Create a Bot Service

To start the development of a bot, you need to provision an Azure Bot Service in your Azure subscription.

1. Switch to the browser tab in which the Azure portal is open.

2. In the Microsoft Azure portal, click **+ Create a resource** and search for **Web App Bot**. Then in the list of services that are found, click **Web App Bot**, and on the **Web App Bot** blade, click **Create**.
3. In the **Web App Bot** blade, enter the following details and then click **Create**.
    - **Bot name**: *Enter a unique name for your bot.*
    - **Subscription:** *Choose your Azure subscription*.
    - **Resource Group**: *Use the same resource group you used for the QnA Maker service*.
    - **Location**: *Choose region where you want to host your bot*.
    - **Pricing Tier**: *Choose the* **F0** *pricing tier*.
    - **App Name**: The same name as your bot.
    - **Bot Template**: *Select the **SDK v3** category, and then select the* **Question and Answer** *template (you can use either C# or Node.JS – you will not need to write any code*)
    - **App service plan / Location**: *Create a new app service plan in any available location.*
    - **Azure Storage**: *Create a new Azure Storage account with a unique name*.
    - **Application Insights**: On
    - **Application Insights Location**: *Choose any available location*.
    - **Microsoft App ID and password**: Auto create App ID and password.
4. Wait for the bot to be deployed, and then in the **All resources** blade, click your bot.
5. In the blade for your bot, view the **Application Settings** page.
6. In the **Application Settings** page, in the **App Settings** section, set the **QnAKnowledgebaseId**, **QnAAuthKey**, and **QnAEndpointHostName** values to the appropriate values in the HTTP request URL for your published QnA service as shown here:

    ```
    POST /knowledgebases/QnAKnowledgebaseId/generateAnswer
    Host: QnAEndpointHostName
    Authorization: EndpointKey QnAAuthKey
    Content-Type: application/json
    {"question":"<Your question>"}
    ```

7. Click **Save** to save the settings.

Test the bot

The code that has been generated implements a working bot. You can test it to see it in action.

1. In the blade for your bot, click **Test in Web Chat**. This opens a chat window.
2. At the bottom of the **Chat** pane, type *Hello*. After a short interval, the bot will be initialized, and the message *hello* will be displayed.
3. Enter the message *What do you call an alligator in a vest?*, and verify that the answer *An investigator!* is returned.
4. Enter the message *A chicken crossed the road. Why?*, and verify that the answer *To get to the other side!* is returned – despite the question not being exactly the same as the format in the knowledge base.
5. Close the blade for your bot.

Summary

In this lab, you have created a simple bot that uses a QnA service to answer user questions. The Azure Bot Service provides the application “scaffolding” you need to create bots that accept and respond to user input.

To learn more about developing bots – including more sophisticated bots that take advantage of cognitive services, see https://dev.botframework.com.