Microsoft Technology Center – Transformation Academy

Azure Bot Workshop

November 15, 2018

We'll start with a quick overview presentation of Azure Bot Service. Then we'll get into the hands-on workshop.
This workshop is a subset of Microsoft's Learn AI Bootcamp, which normally takes two days. Below are the key bot pieces.

Resources:

- https://portal.azure.com
- https://docs.microsoft.com/azure/bot-service/
- https://azure.github.io/LearnAI-Bootcamp/
- https://azure.github.io/LearnAI-Bootcamp/lab02.2-building_bots/0_README

Pre-Requisites:

- Visual Studio 2017 (any edition), preferably v15.9 - https://visualstudio.microsoft.com/vs/
- .NET Core SDK 2.1 SDK - https://www.microsoft.com/net/download
- Clone or download/unzip the repo https://github.com/Azure/LearnAI-Bootcamp (you will need this for some of the labs, and it's a handy reference)

- Lab 1.5 - Create a LUIS App
  a. https://azure.github.io/LearnAI-Bootcamp/lab01.5-luis/1_LUIS.html
  b. If you did not do this Lab - you can use this info to complete the Bot labs below (from a completed Lab 1.5)
     i. LUIS app name: **pzlearnailuis**
     ii. LUIS app ID: **0e2f7f65-997b-488d-9101-00fec18eece4**
     iii. LUIS endpoint: **https://eastus.api.cognitive.microsoft.com/luis/v2.0/apps/0e2f7f65-997b-488d-9101-00fec18eece4?subscription-key=e7be3087d1bb4cb1a5cae4c837886089&timezoneOffset=-360&q=**
     iv. LUIS API Key: **e7be3087d1bb4cb1a5cae4c837886089**
  c. Try a sample query like this to make sure your LUIS app is published correctly: https://eastus.api.cognitive.microsoft.com/luis/v2.0/apps/0e2f7f65-997b-488d-9101-00fec18eece4?subscription-key=e7be3087d1bb4cb1a5cae4c837886089&timezoneOffset=-360&q=find+pictures+of+water
- Lab 2.1 - Create an Azure Search Index
  a. https://azure.github.io/LearnAI-Bootcamp/lab02.1-azure_search/1_AzureSearch.html
  b. Completing this lab requires having done Lab 1.1
  c. **If you did not do this Lab or Lab 1.1 - you can use this info to complete the Bot labs below (from a completed Lab 1.1 and 2.1):**
     i. Search Service Name: **pzlearnai**
     ii. Search Service API Key: **37DB9C8114730839B262D1683D19AF6B**
     iii. Search Index Name: **documentdb-index**
- Create a first simple bot with Bot Builder SDK for .NET

   a. https://docs.microsoft.com/en-us/azure/bot-service/dotnet/bot-builder-dotnet-sdk-quickstart?view=azure-bot-service-4.0
   b. Install VSIX and Bot Emulator
      i. NOTE if Bot does not show in File->New Project, uninstall it and use latest 4.0 VSIX from above download link.
   c. Get ngrok - https://ngrok.com/ - create a free account, download ngrok.exe, and note the path where you saved it. You will need ngrok for the final Lab below.
      i. In Bot Emulator settings, configure it to use ngrok where you saved it in the previous step.
      ii. See https://github.com/Microsoft/BotFramework-Emulator/wiki/Tunneling-(ngrok)
   d. Create new Echo Bot v4 project. Ensure build type is .NET Core 2.1. Update Nuget packages.
   e. Choose to host in IIS Express or self-host. Run.
   f. Open bot emulator by double-clicking the .bot file.
   g. Test some chat with the bot. If self-hosting, observe the debug console.
   h. You do not need this bot project for later steps. You can close the solution and delete the folder.

- Lab 2.2 - 1.1: Deploy a Bot in Azure
   a. https://azure.github.io/LearnAI-Bootcamp/lab02.2-building_bots/1_Dialogs_and_Regex.html
   b. Follow directions under Lab 1.1, "Create an Azure Web App Bot"
   c. Download the source, open in Visual Studio. Ensure build type is .NET Core 2.1. Update Nuget packages ***per the directions in this lab (follow carefully!)***
- Lab 2.2 - 1.2: Edit the Bot from Azure
   a. Follow directions under Lab 1.2, "Creating a simple bot and running it"
   b. Note that the lab is slightly out of date. For example, it says to replace the entire "TurnAsync" method in PictureBot.cs - do not do this, instead examine the code and, minimally (do more if desired) edit the lines that create the responseMessage variable and send it back to the user.
   c. Ignore wording that we're using the preview v4 bot emulator. The emulator went to production release after this lab was created.
- Lab 2.2 - 1.3: Manage state and services
   a. Follow the directions to customize the bot code. Work with the Microsoft people and each other if you get stuck.
- Lab 2.2 - 1.4: Organizing Code
   a. At the end of the lab, your solution will not build. That's OK; Lab 2.2 - 1.5 fixes that.
- Lab 2.2 - 1.5: Regex and Middleware
   a. When you are told to add the code starting with "middleware.Add…" to Startup.cs, if you cannot find the regex comment referenced in the lab manual, then paste this code right below where you earlier added "var middleware = options.Middleware;"
   b. At the end of this lab, your bot code should build and run, and respond in the Bot Emulator. Don't proceed until it does.
- Lab 2.2 - 2: Azure Search
   a. https://azure.github.io/LearnAI-Bootcamp/lab02.2-building_bots/2_Azure_Search.html
   b. See note above (item 2) about Azure Search - if you did not do that lab, use the info in 2c above to complete this lab
   c. When you are done, you should be able to enter "search pics" in the bot emulator chat, then enter a term like "water" and see some image results.
- Lab 2.2 - 3: LUIS
   a. https://azure.github.io/LearnAI-Bootcamp/lab02.2-building_bots/3_LUIS.html
   b. See note above (item 1) about LUIS - if you did not do that lab, use the info in 1b above to complete this lab
   c. When you are done, you should be able to enter something like "search for pictures of water" into the bot emulator chat and see some image results.
- Lab 2.2 - 4: Publish and Register

    a. After publish, are you getting an HTTP 502.5 error? If so, proceed as follows…
        i. Edit your web.config and change the value of "stdoutLogEnabled" to true.
        ii. Change your publish profile to delete existing files, and to publish in Debug, not Release.
        iii. Clean your Project. (Right-click project node, Clean)
        iv. Republish, then go to your published bot app's diagnostic console at https://{YOURBOTNAME).scm.azurewebsites.net/DebugConsole. Check any stdout log files for detailed error info, and troubleshoot accordingly.
        v. Hint: maybe… you need to do a Ctrl-F through your whole Solution and search for "EchoBotWithChat"…
    b. You should be able to start the Bot Emulator, open your .bot file, connect to the production endpoint, and chat with your bot.
- You're done with the workshop. If you are able to chat with your bot after it's published - great work!
- Further Reading and Closing: https://azure.github.io/LearnAI-Bootcamp/lab02.2-building_bots/5_Closing.html

Production enterprise bots need many additional features, such as introduction messages, nicer UI layouts, the ability to involve a human agent in the conversation if needed, and much more. Microsoft provides an Enterprise Bot Template that beings together many good practices. You can start working with it here: https://docs.microsoft.com/en-us/azure/bot-service/bot-builder-enterprise-template-overview?view=azure-bot-service-4.0