FREE eBook

LEARNING botframework

Free unaffiliated eBook created from
Stack Overflow contributors.

#botframework

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: botframework

It is an unofficial and free botframework ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official botframework.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

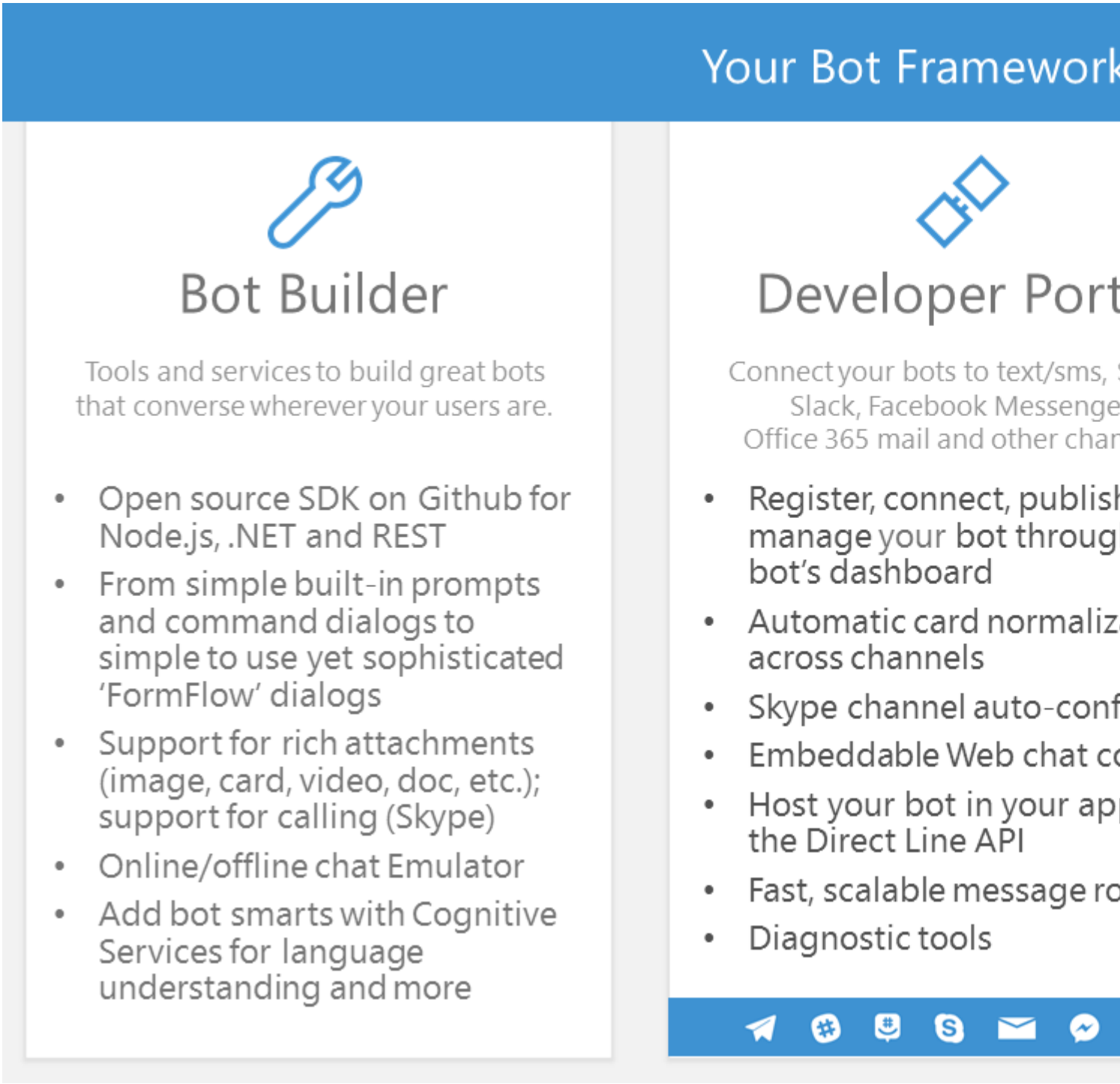
Chapter 1: Getting started with botframework

Remarks

Microsoft Bot Framework is a comprehensive offering to build and deploy high quality bots for your users to enjoy in their favorite conversation experiences. Developers writing bots all face the same problems: bots require basic I/O; they must have language and dialog skills; they must be performant, responsive and scalable; and they must connect to users – ideally in any conversation experience and language the user chooses. Bot Framework provides just what you need to build, connect, manage and publish intelligent bots that interact naturally wherever your users are talking – from text/sms to Skype, Slack, Facebook Messenger, Kik, Office 365 mail and other popular services.

Bots (or conversation agents) are rapidly becoming an integral part of one's digital experience – they are as vital a way for users to interact with a service or application as is a web site or a mobile experience. Developers writing bots all face the same problems: bots require basic I/O; they must have language and dialog skills; and they must connect to users – preferably in any conversation experience and language the user chooses. The Bot Framework provides tools to easily solve these problems and more for developers e.g., automatic translation to more than 30 languages, user and conversation state management, debugging tools, an embeddable web chat control and a way for users to discover, try, and add bots to the conversation experiences they love.

The Bot Framework consists of a number of components including the Bot Builder SDK, Developer Portal and the Bot Directory.



Versions

Bot Builder Latest Releases

Language	Version	Release Date
Node.js	3.7.0	2017-02-23
C#	3.5.5	2017-03-07

Previous releases can be found here.

Examples

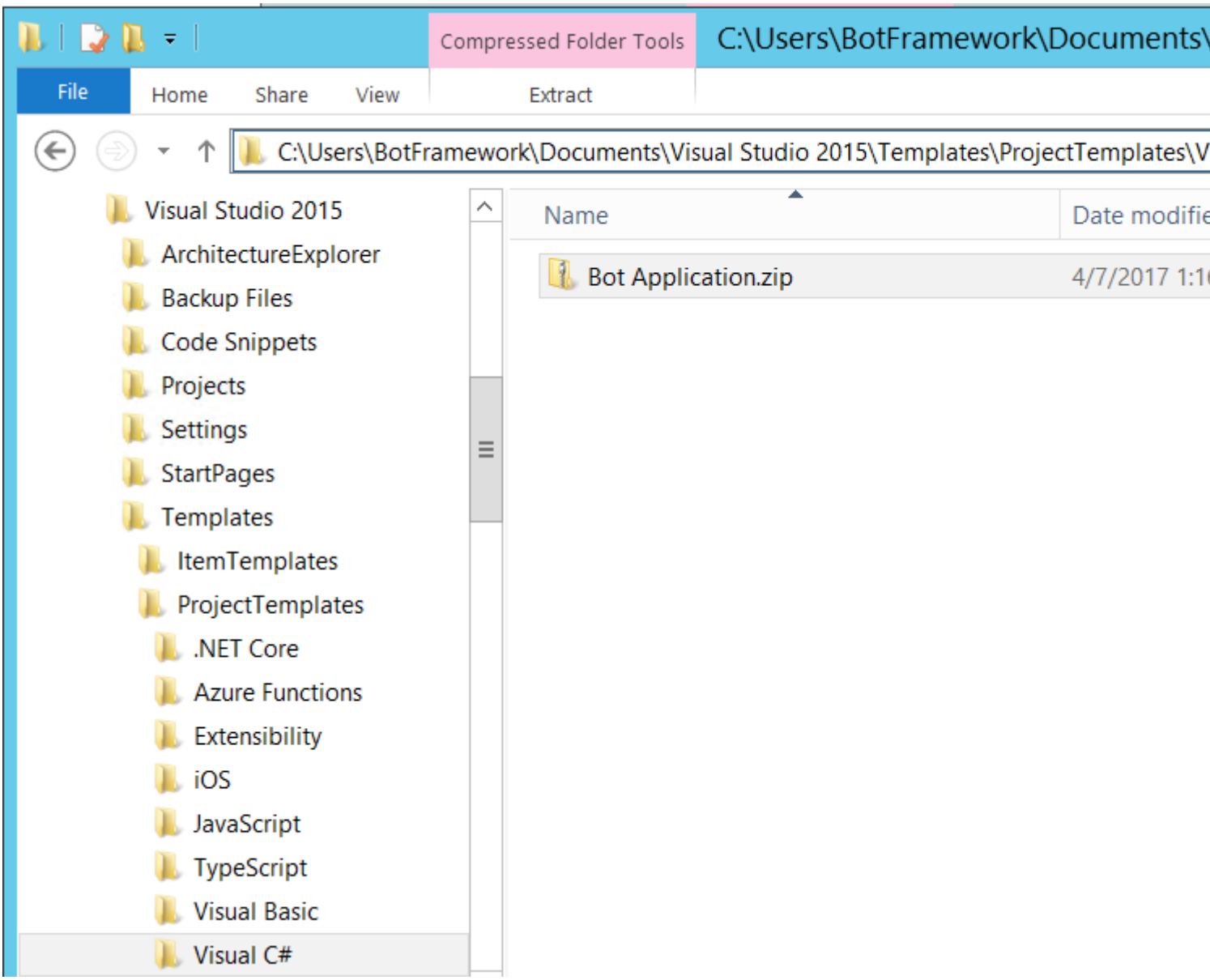
Installation or Setup

C#

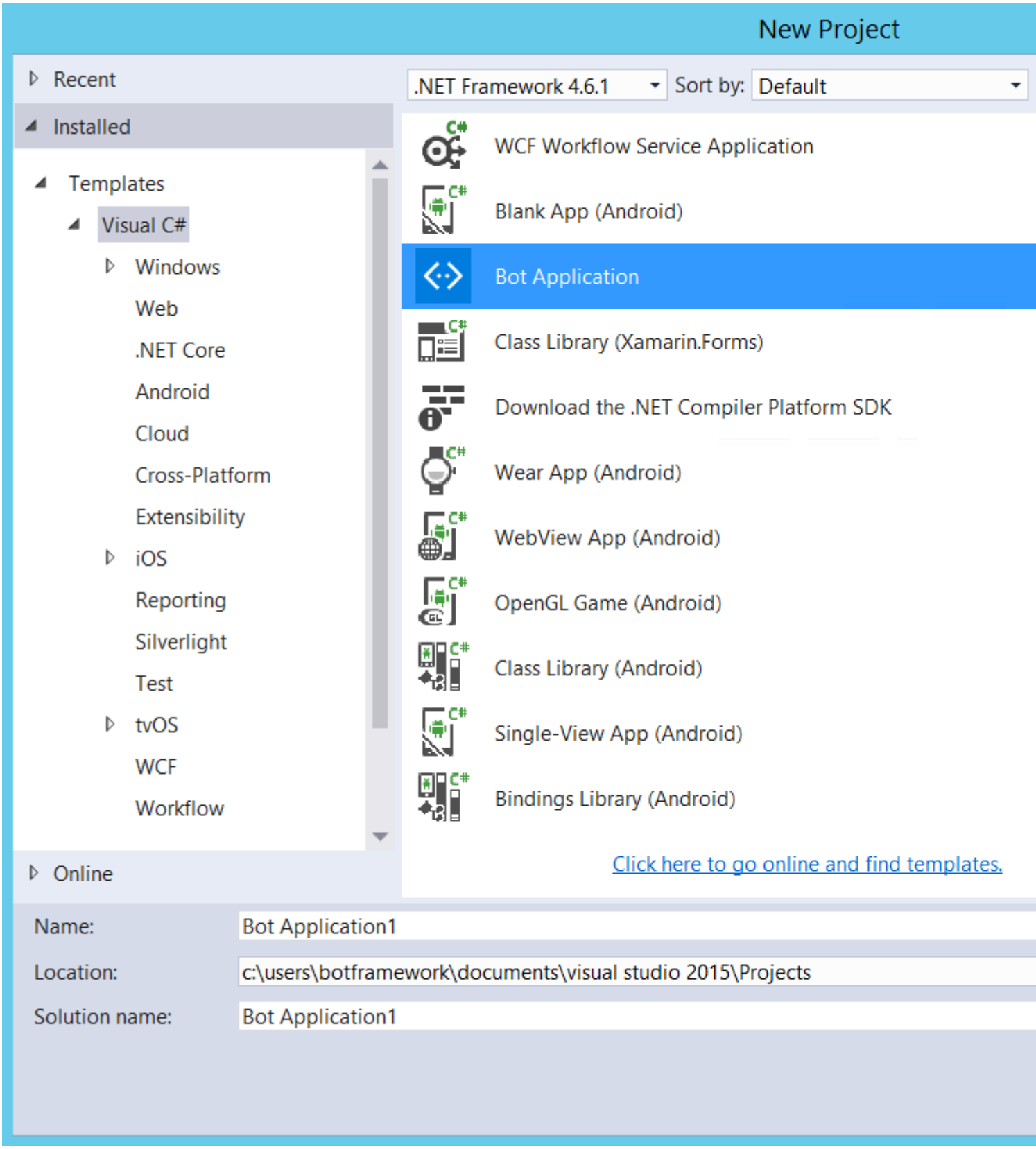
1. **Visual Studio 2015** (latest update) - you can download the community version here for free: www.VisualStudio.com

2. Important: **update all VS extensions** to their latest versions Tools->Extensions and Updates->Updates

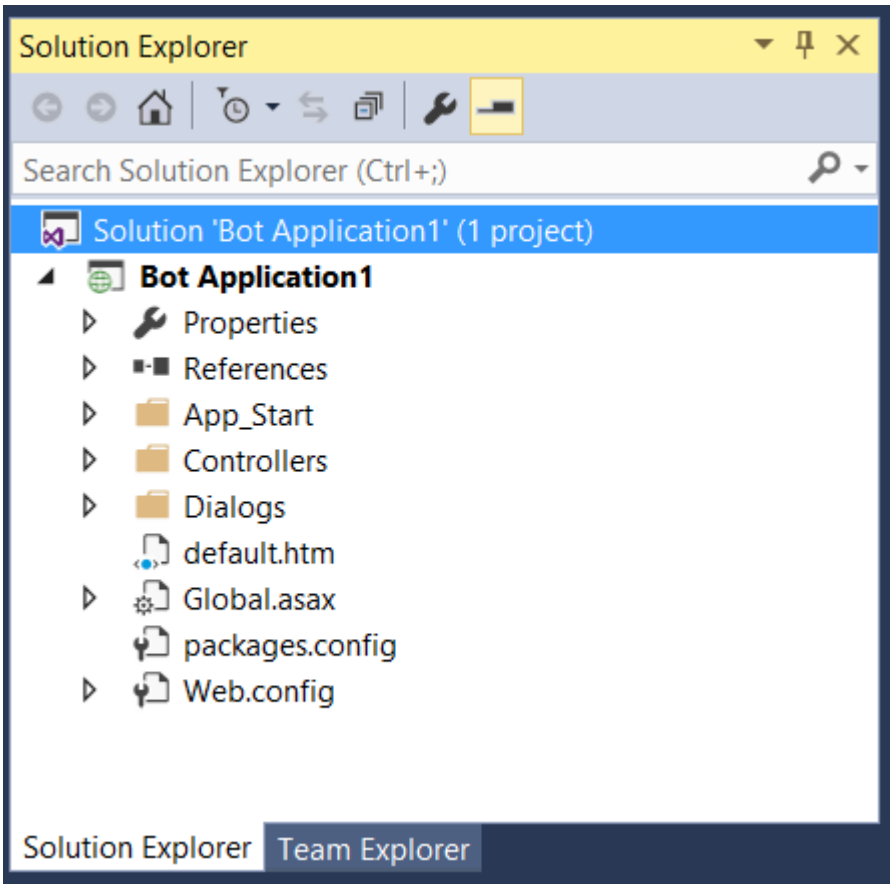
3. Download the **Bot Application template** from here: Template Download Save the zip file to your Visual Studio 2015 templates directory which is traditionally in "%USERPROFILE%\Documents\Visual Studio 2015\Templates\ProjectTemplates\Visual C#" Note: you will need to restart visual studio after this step, in order to use the template.



4. Create a **new C# project** using the new Bot Application template



Once your bot is finished being created, you should have a solution similar to this:



5. **Run the application** by hitting F5, or by clicking the green Run button in the tool bar. Since our new bot is actually a WebAPI project, a browser window will be opened to the default.htm page. The bot is now running, and exposed locally. Note the url ... it will be needed to setup the Bot Framework Emulator in the next step.

Node.js

1. Create a new node.js project by using `npm init`.
2. Install the botbuilder sdk and restify using the following npm commands:

```
npm install --save botbuilder
npm install --save restify
```

3. To create your bot, create a new file called index.js, and copy the following code to initialize the bot.

```
var restify = require('restify');
var builder = require('botbuilder');

// Setup Restify Server
var server = restify.createServer();
server.listen(process.env.port || process.env.PORT || 3978, function () {
   console.log('%s listening to %s', server.name, server.url);
});

// Create chat connector for communicating with the Bot Framework Service
var connector = new builder.ChatConnector({
    appId: process.env.MICROSOFT_APP_ID,
    appPassword: process.env.MICROSOFT_APP_PASSWORD
```

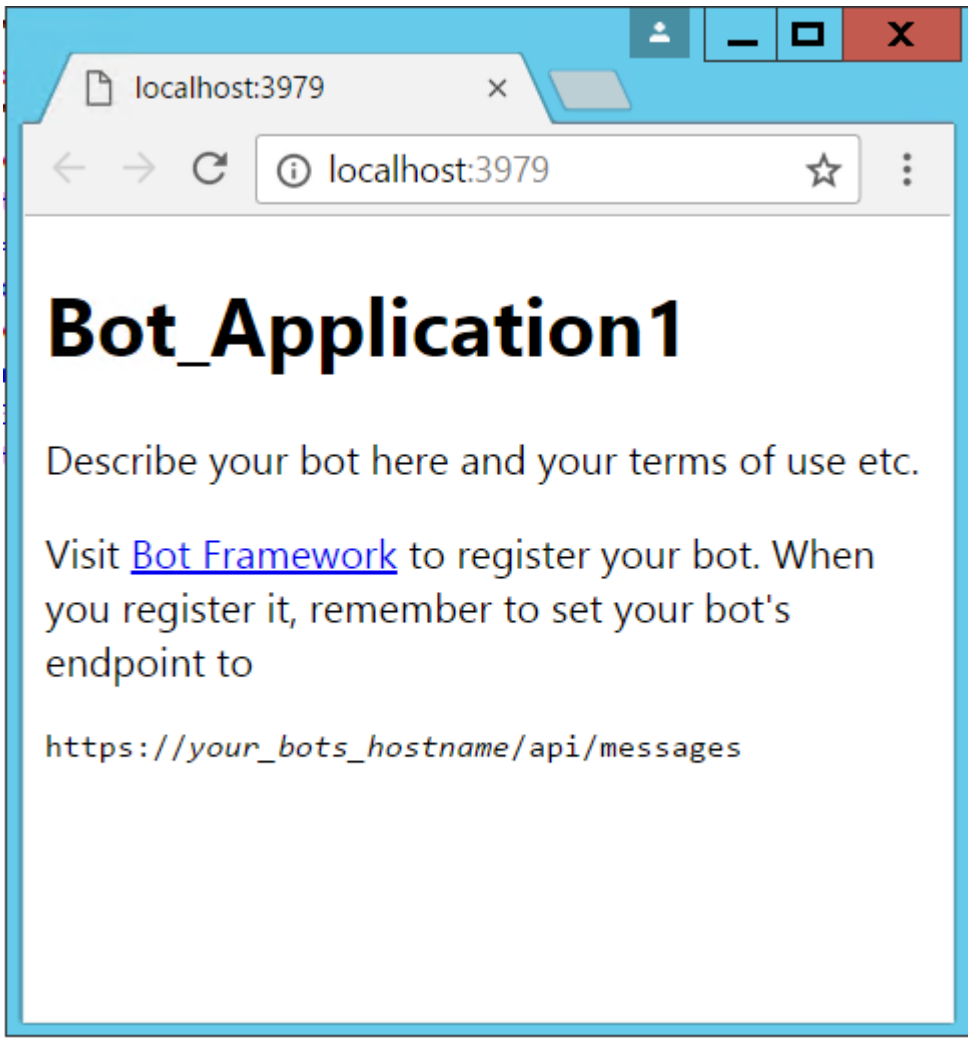
```
});

var bot = new builder.UniversalBot(connector);
```

4. You should now be able to run this file using `node index.js`.

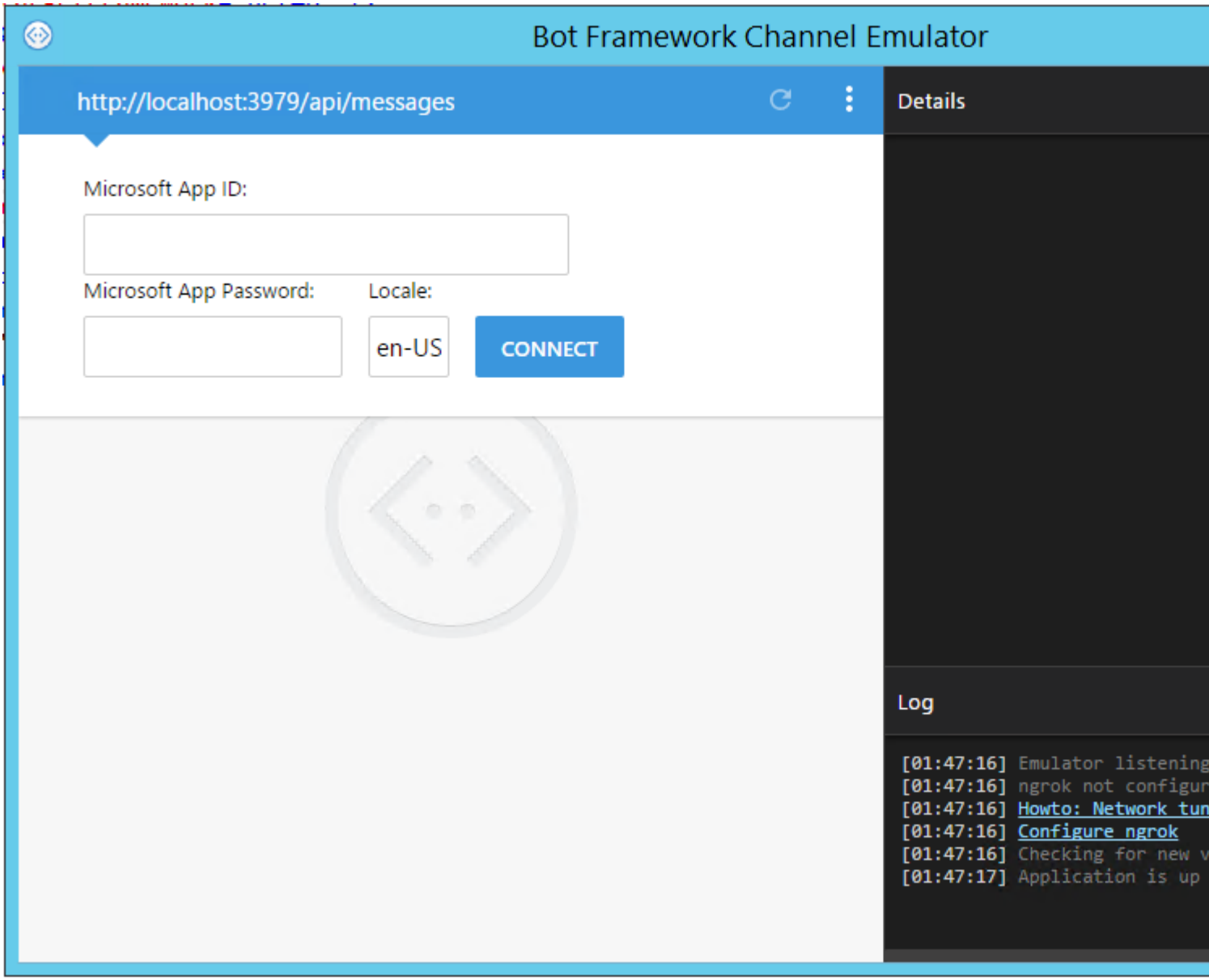
This is a basic setup that will be required for all bots created with bot framework. You can treat this as a blank template project to start with. It initializes a restify server for your bot and creates a connector to connect local machines with your server.

Downloading Emulator for Debugging (Both for node and C#)

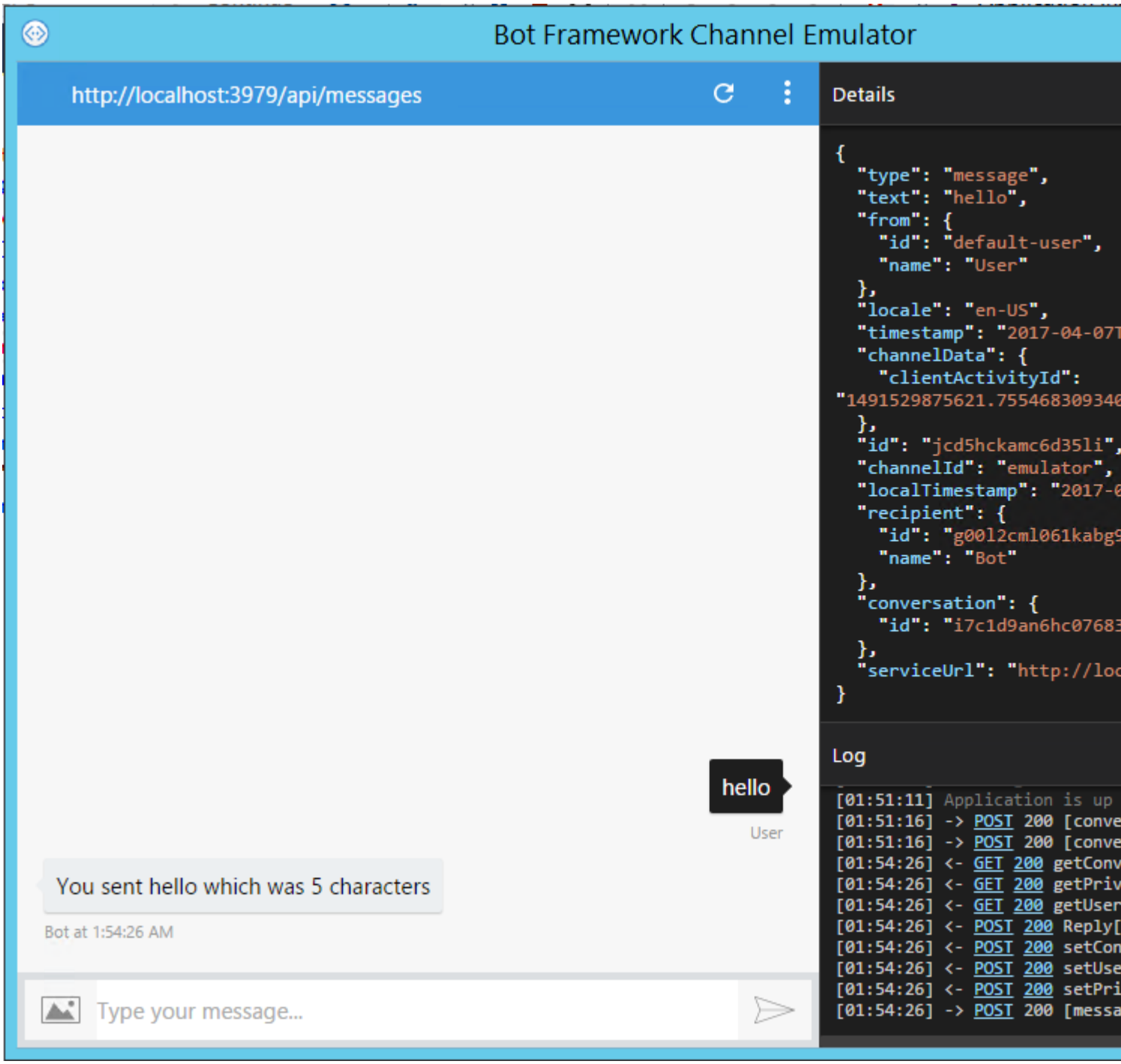


1. Download and install the **Bot Framework Emulator** Emulator Download

2. Run the emulator, and enter the url from step 5 (C#) into the **Endpoint URL** text box. Then, click "Connect".



3. You should now be able to communicate with your bot using the chat window in the emulator. You will see the conversation details logged in the bottom right, and you can click on the Post and Get line items to see the json that has been passed back and forth.



Congratulations on creating a Bot using the Microsoft Bot Framework!

Read Getting started with botframework online:
https://riptutorial.com/botframework/topic/7509/getting-started-with-botframework

Chapter 2: Adding Natural Language Processing

Introduction

Bot Framework supports `Recognizers`. A recognizer is used to recognize what to do whenever a user sends the bot any message. Therefore you can design your bot to recognize intents based on the user input. The recognizer can be used with LUIS API in order to add natural language understanding for the bot.

Syntax

- var recognizer = new builder.LUISRecognizer('Your model's URL');
- var intents = new builder.IntentDialog({recognizers: [recognizer]});

Examples

Initializing and Adding LUISRecognizer

Once you're up with a new project with the basic template provided in the Introduction, you should be able to add a LUISRecognizer like so -

```
var model = ''    // Your LUIS Endpoint link comes here
var recognizer = new builder.LuisRecognizer(model);
```

Now, `recognizer` is a LUISRecognizer and can pass intents based on your defined LUIS Model. You can add the `recognizer` to your intents by

```
var intents = new builder.IntentDialog({recognizers: [recognizer]});
```

Your bot is now capable of handling intents from LUIS. Any named intents on LUIS can be detected by using the `matches` property of `IntentDialog` class. So say, an intent named `hi` is defined in the LUIS model, to recognize the intent on the bot,

```
intents.matches('hi', function(session) {
    session.send("Hey :-)");
});
```

Defining a LUIS Model with Intents

Creating a LUIS Model requires little to no programming experience. However, you need to be familiar with 2 important terms that will be used extensively.

1. **Intents** - These are how you identify functions that need to be executed when the user types in something. Eg - An intent named `Hi` will identify a function that needs to be executed whenever the user sends "Hi". Intents are uniquely named in your program/model.
2. **Entities** - These identify the nouns in a statement. Eg - "Set an alarm for 1:00 pm", here `1:00 pm` is an entity that needs to be recognized by the chat-bot to set an alarm.

Note: Images of the website are not provided as the front-end my change, but the core concept remains the same.

To create a new model, go to LUIS.ai and sign-in with your Microsoft Account to be taken to the app creation page. Where a blank project can be created.

Defining Intents:

Intents can be defined on the `Intents` tab. They identify what function you need to perform when the user enters anything.

All applications have a default `None` intent, which is activated whenever the user input matches no other intent.

To define an intent,

1. Give it a unique name relevant to the function you want to perform.
2. Once the naming is complete, you should add `utterances` to the intent. Utterances are what you want the user to send in order to activate the intent that you are defining. Try feeding as many different `utterances` as possible in order for LUIS to associate `intents` and `utterances` properly.
3. Train your LUIS Model, by clicking the `Train` button on `Train and Test` Tab. After training the app can be tested in the panel below.
4. Finally publish your app in the `Publish App` Tab. You should now get an endpoint URL that should be put in while defining `LUISRecognizer` in your bot code.

Adding Entities to LUIS Model

An entity is the information that your bot extracts from a particular utterance conforming to an intent.

Eg- Let `My name is John Doe` belong to an intent called `introduction`. For your bot to understand and extract the name `John Doe` from the sentence, you need to define an entity which does so. You can name the entity whatever you wish, but it is best to name it as something pertaining to what it extracts. In our example, we can call our entity `name`.

Entities can be re-used between different intents, to extract different things. So the best principle would be to make an entity that extracts only type of data and use it across different intents. Therefore, in our above example, say `Book a flight on Emirates` belongs to the intent `booking`, then the same entity, `name`, can be used to extract the flight name `emirates`.

You need to keep in mind two things before you go on defining entities -

1. Entities should be unique per utterance in an intent. An entity cannot be used twice in the same utterance.
2. LUIS is case insensitive. This implies that everything extracted and received through entity extraction will be in lower-case. So extracting case-sensitive data through entities is probably a bad idea.

Adding pre-built entities

Pre-built entities are, as the name suggests, pre-built i.e. they are already configured to extract a particular type of data across the intent they are added to. An example can be the entity `number` that extracts numbers from the intent it is assigned to. The numbers can be either in numeric or alphabetical like `10` or `ten`.

For a full list of all pre-built entities, you can visit [Pre-built Entities List][1].

To add pre-built entities,

1. Go to the `entities` tab.
2. Click `Add pre-built entities` and select the entity you want to add to the model and hit save.

Adding Custom Entities Custom Entities are of 4 types,

1. **Simple**: A simple entity extracts a particular data, `name` in the examples above is a simple entity.
2. **Hierarchical**: A parent entity with children entities (sub-types) which are dependent on the parent.
3. **Composite**: A group of 2 or more entities independent together.
4. **List**: An entity that recognizes words only from a given list.

Defining Simple Entities

1. Go the the `entities` tab.
2. Click on `Add Custom Entities`
3. Name your entity, check the required entity type and hit `Save`.

All other type of entities can be added in the same way by just changing the `Entity Type` to one of the above types. In hierarchical and composite entity types, you'll also need to give the children names along with the parent entity name. Defining List entities is a little different than the rest.

Defining List Entities

After you follow the above steps to create a `List Entity` by putting th `Entity Type` as List, you'll be directed to the details page of the entity you just defined.

1. Define a canonical value. This is a standard value that the bot will receive when the user types in any of the synonyms.
2. Define synonyms to the canonical value. They will be converted to the canonical value upon being encountered by the entity.

You can also import entire lists by using an array of JSON Objects, of the form:

```
[
    {
        "canonicalForm": "Hey",
        "list": [
            "Howdy",
            "Hi"
         ]
    },
    .
    .
    .
]
```

Associating an entity with an intent

`Pre-built` and `list` entities already have a set of values defined which can be extracted from all utterences, however, `Simple`, `Hierarchical` and `Composite` utterances need to be trained to pick up values.

This can be done by

1. Go to the `intents` tab and choose the intent you'd like to add the entity to.
2. Add an utterance with a dummy value that you would like to be extracted. Say, you can add `My name is John Doe` as an utterance.
3. Click and drag the mouse over the words you want the entity to extract. You will need to highlight `john doe` in the above example.
4. A drop-down will open with a list of all entities available in your project. Select the corresponding one as you see fit. `Name` will be the entity selected in the above example.
5. Add more utterances with different dummy values each time and all possible structures you can think of.
6. Train and publish your LUIS Model.

Read Adding Natural Language Processing online:
https://riptutorial.com/botframework/topic/10004/adding-natural-language-processing

Chapter 3: Getting started with Azure Bot Service

Introduction

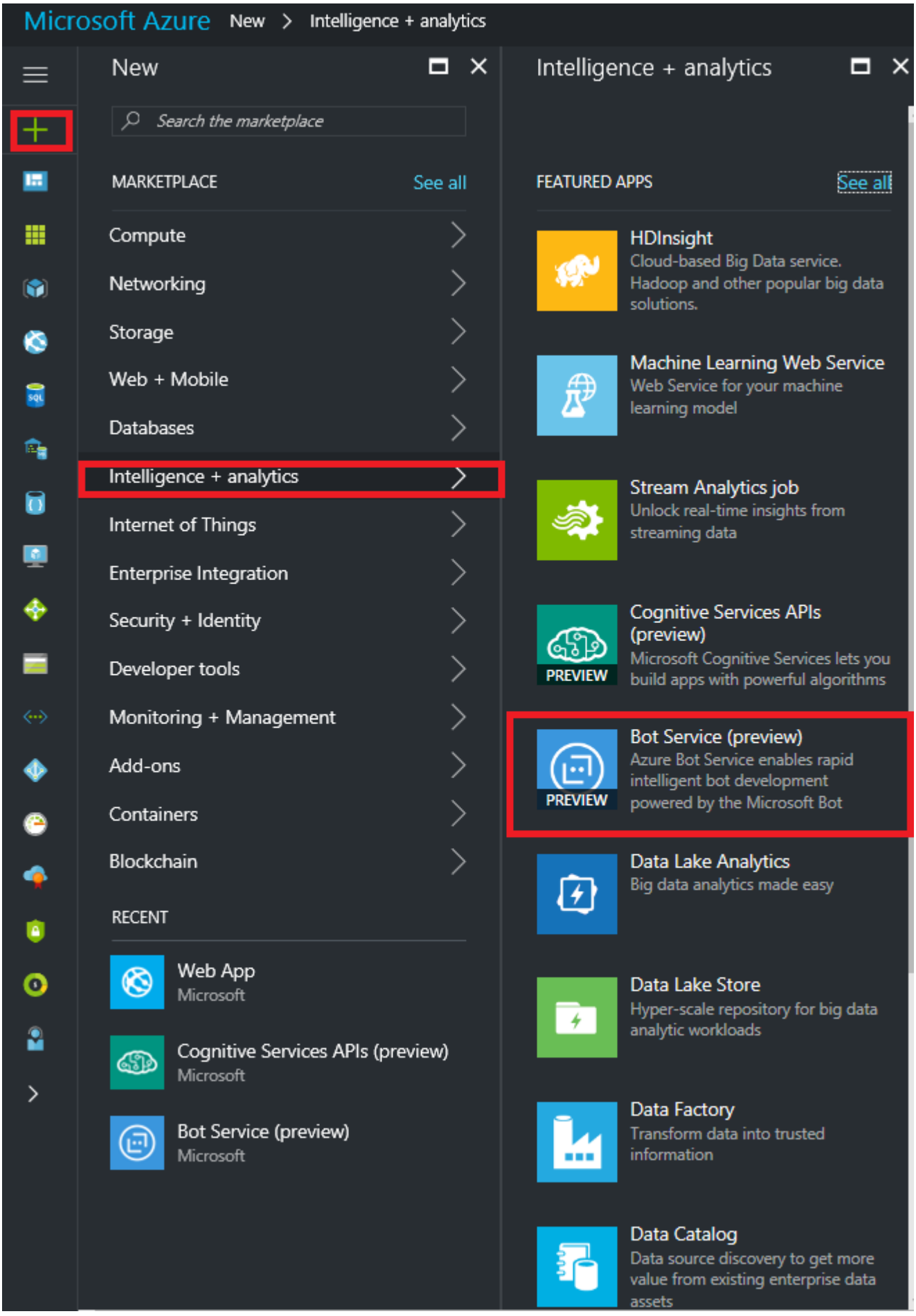
The **Azure Bot Service** provides an integrated environment that is purpose-built for bot development, enabling you to build, connect, test, deploy and manage intelligent bots, all from one place. You can write your bot in C# or Node.js directly in the browser using the Azure editor, without any need for a tool chain. You can also increase the value of your bots with a few lines of code by plugging into Cognitive Services to enable your bots to see, hear, interpret & interact in more human ways

Examples

Getting started with Azure Bot Service

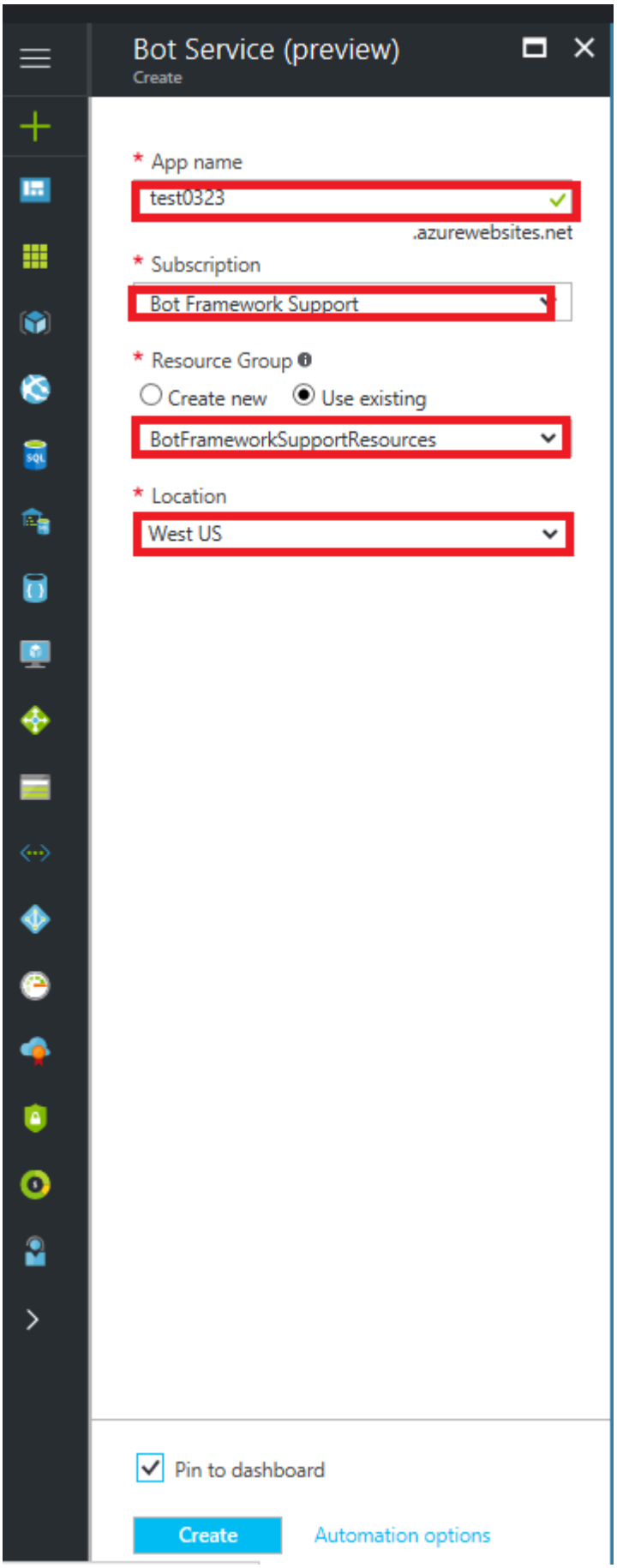
Create a new bot in Azure following this documentation

Login into Azure and from Intelligence + Analytics category, select Bot Service and provide required information.



Enter the required details for the bot, they are identical to the required details of an App Service,for

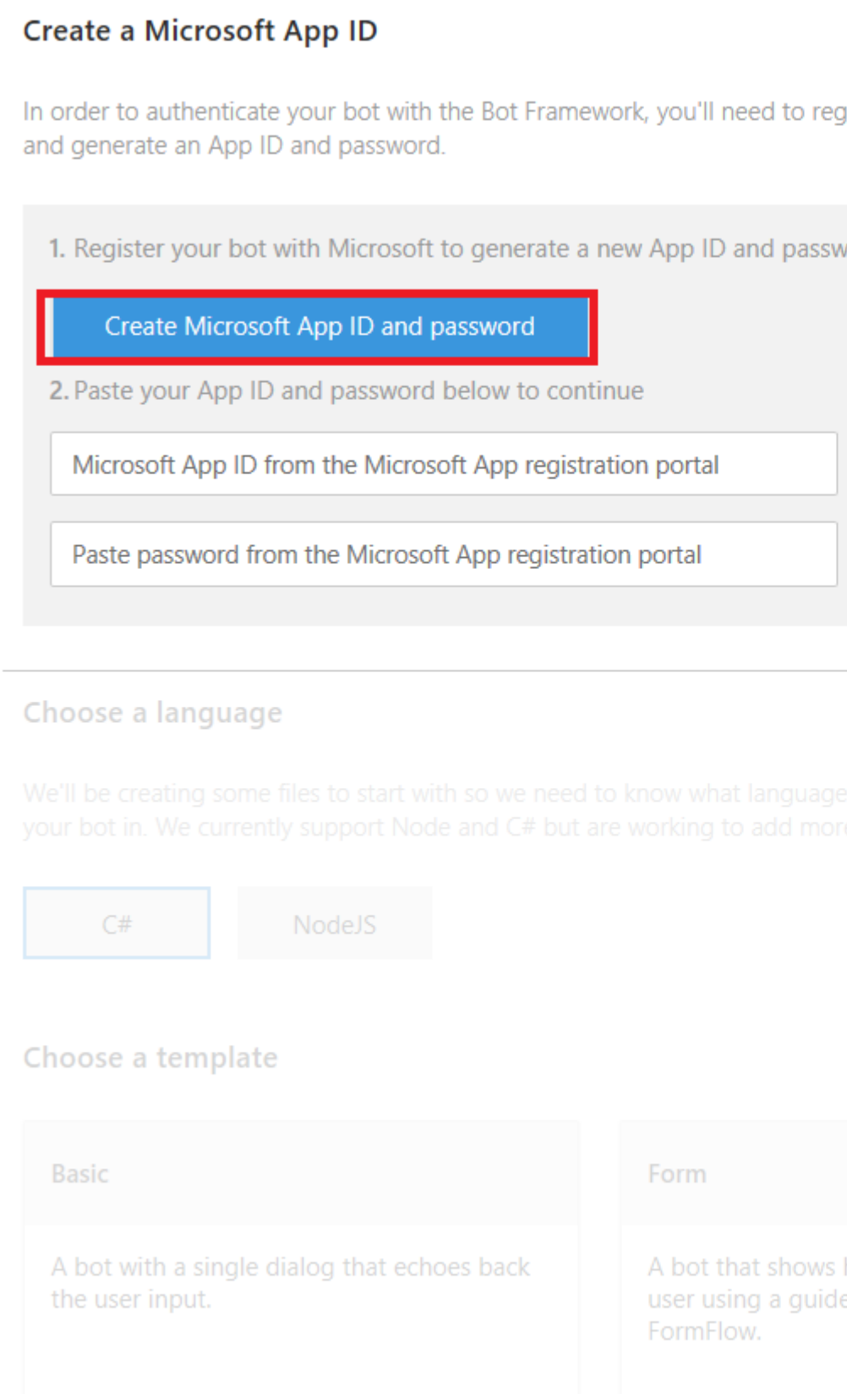
example App Name, Subscription, Resource Group and Location. Once entered, click the Create button.



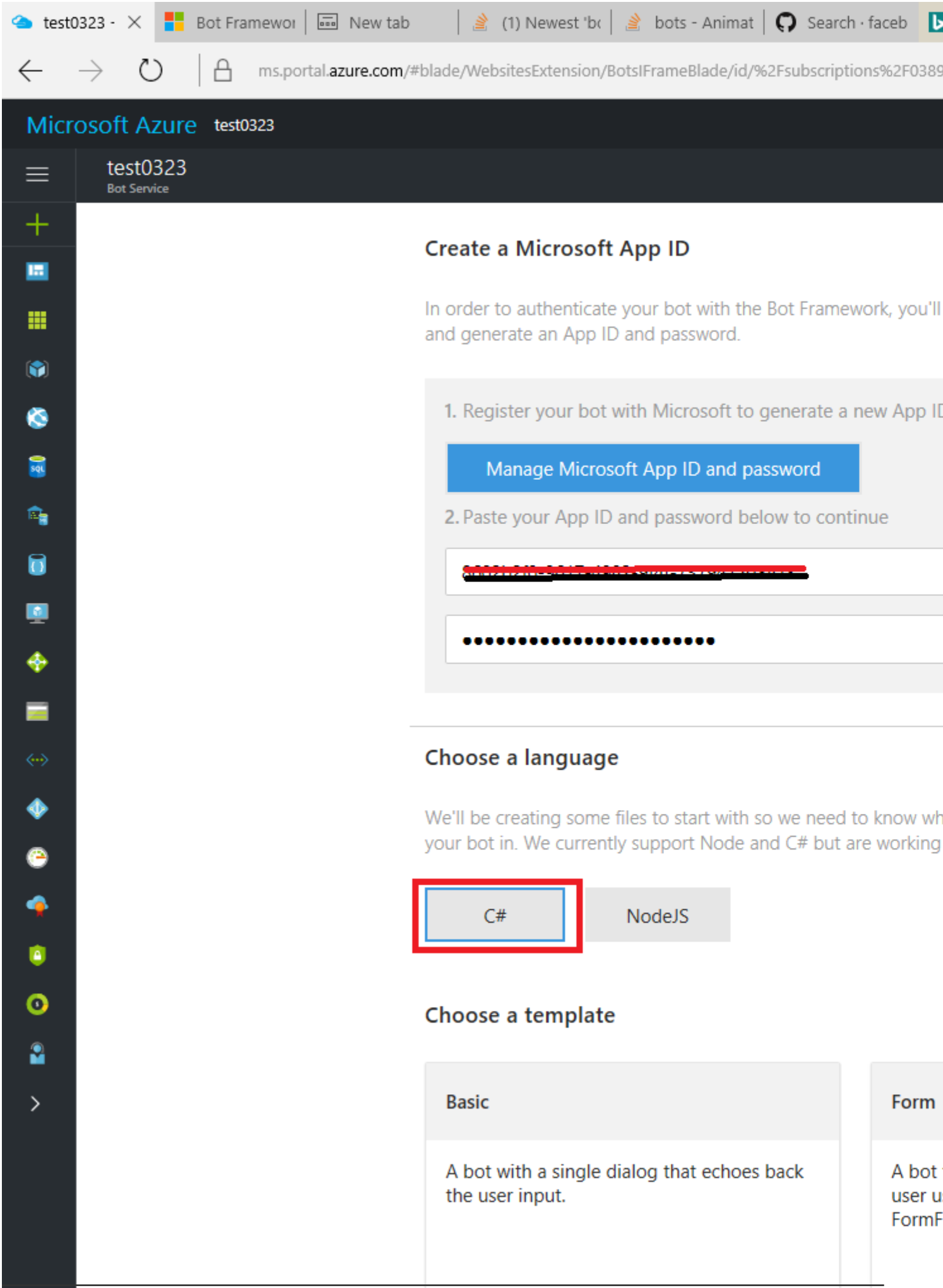
Once created/deployed, navigate to the Bot by clicking on the link either from the main page, if you pinned it to the dashboard or open the resource group and click the link.

Remember that there may be a slight delay before the splash screen displays indicating that the Bot Service is generating your bot; don’t click Create bot again.

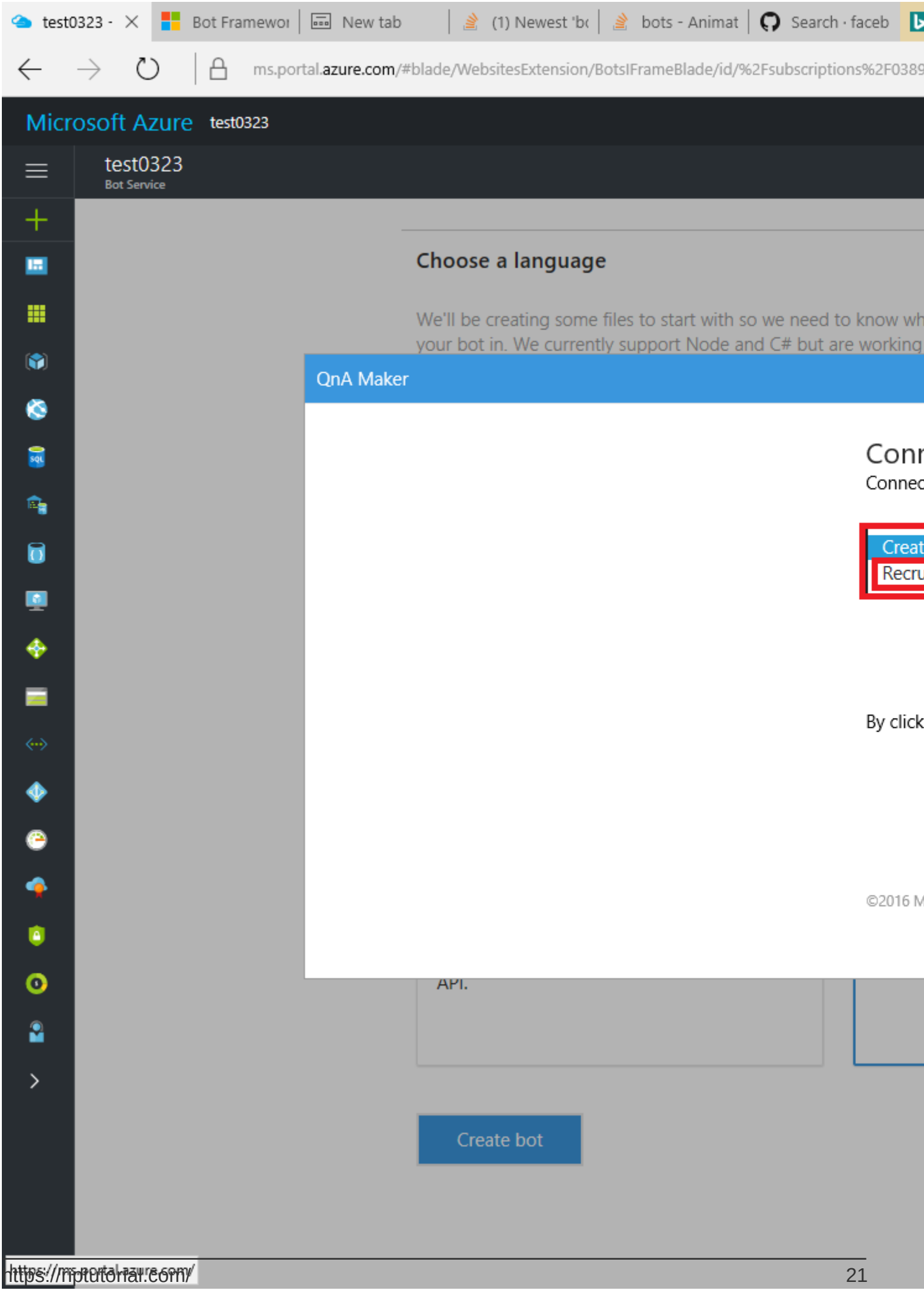
After confirming the deployment generate and configure microsoft app ID and app password.



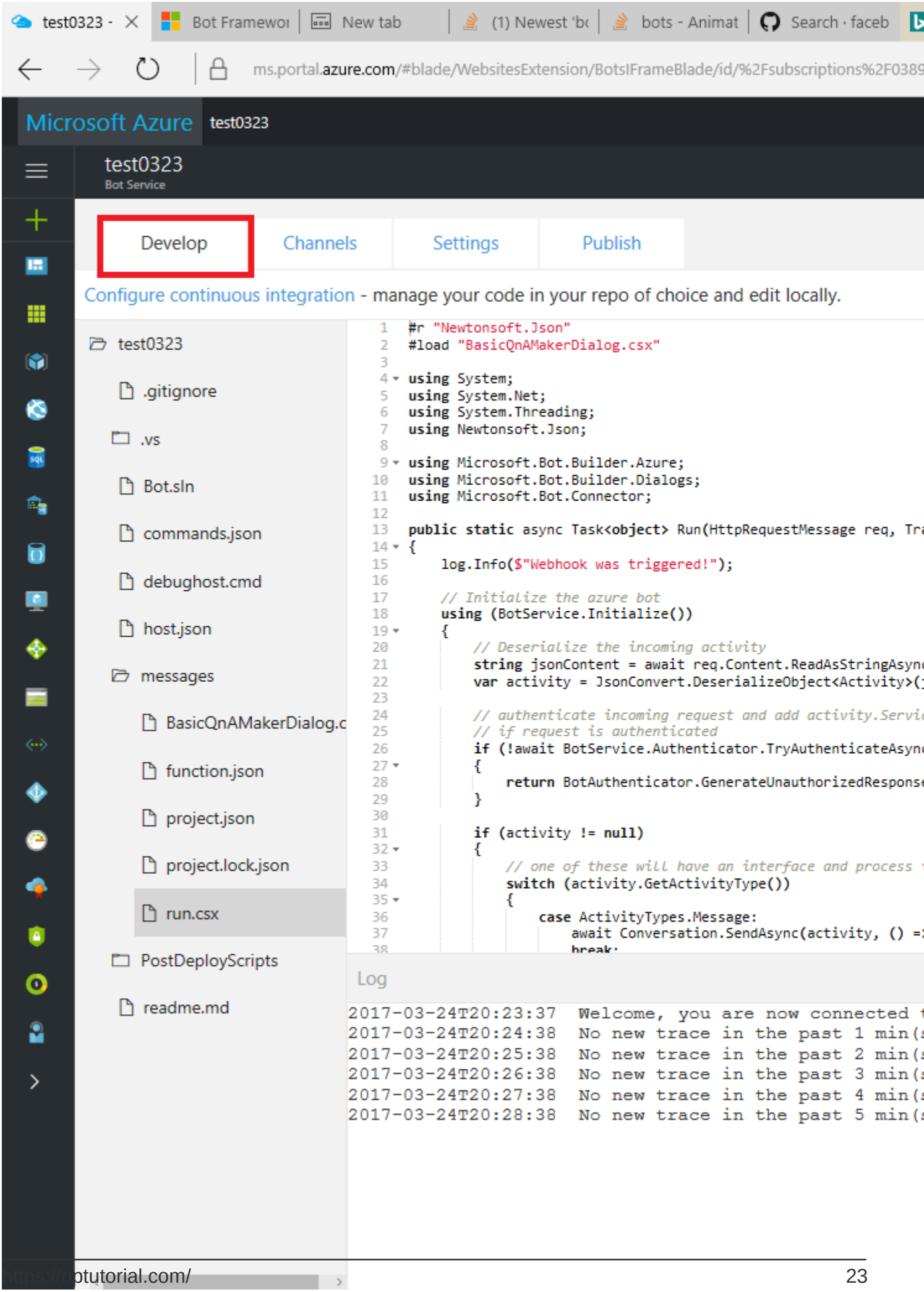
Select programming language of your choice (*I selected C#*) and select **Question and Answer template**.


test0323 -
Bot Framewo
New tab
(1) Newest 'bo
bots - Animat
Search · faceb
ms.portal.azure.com/#blade/WebsitesExtension/BotsIFrameBlade/id/%2Fsubscriptions%2F0389
Microsoft Azure
test0323
test0323
Bot Service
Create a Microsoft App ID
In order to authenticate your bot with the Bot Framework, you'll
and generate an App ID and password.
1. Register your bot with Microsoft to generate a new App ID
Manage Microsoft App ID and password
2. Paste your App ID and password below to continue
Choose a language
We'll be creating some files to start with so we need to know wh
your bot in. We currently support Node and C# but are working
C#
NodeJS
Choose a template
Basic
A bot with a single dialog that echoes back
the user input.
Form
A bot
user u
FormF

generate a new one. As I had already created a knowledge base with my subscription, I selected it. This made my work much easier, reducing the time required to include all the keys in the Azure bot code related to the Knowledge base.


test0323 -
Bot Framewo
New tab
(1) Newest 'bo
bots - Animat
Search · faceb
ms.portal.azure.com/#blade/WebsitesExtension/BotsIFrameBlade/id/%2Fsubscriptions%2F0389
Microsoft Azure
test0323
test0323
Bot Service
Choose a language
We'll be creating some files to start with so we need to know wh
your bot in. We currently support Node and C# but are working
QnA Maker
Conn
Connec
Creat
Recru
By click
©2016 M
API.
Create bot
https://ms.portal.azure.com/

the functional bot in the **chat control**. The **default code** is generated when you create Bot Azure Service. You can change the logic of the code based on your requirements.



Create a new repository in the github to configure continuous deployment with Azure and copy the

following this documentation.

You can track the build updates and errors using *Azure Analytics*.

Looking forward to update the Bot and move to next level.

Read Getting started with Azure Bot Service online:
https://riptutorial.com/botframework/topic/9557/getting-started-with-azure-bot-service

Chapter 4: Getting Started with QnA Services

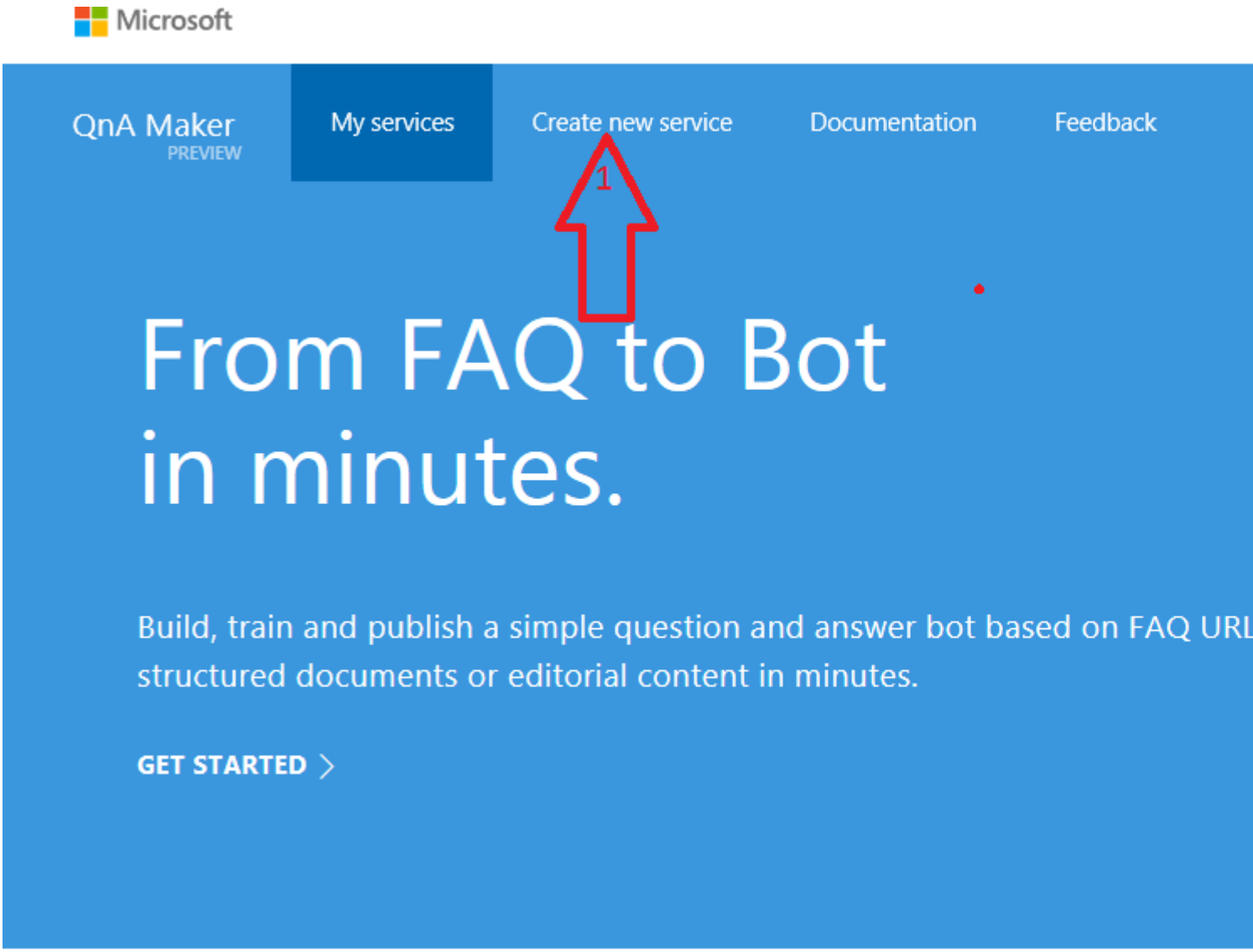
Introduction

The **QnA Maker** is a free, easy-to-use, REST API- and web-based service that trains AI to respond to users' questions in a more natural, conversational way. With optimized machine learning logic and the ability to integrate industry-leading language processing, QnA Maker distills semi-structured data like question and answer pairs into distinct, helpful answers.

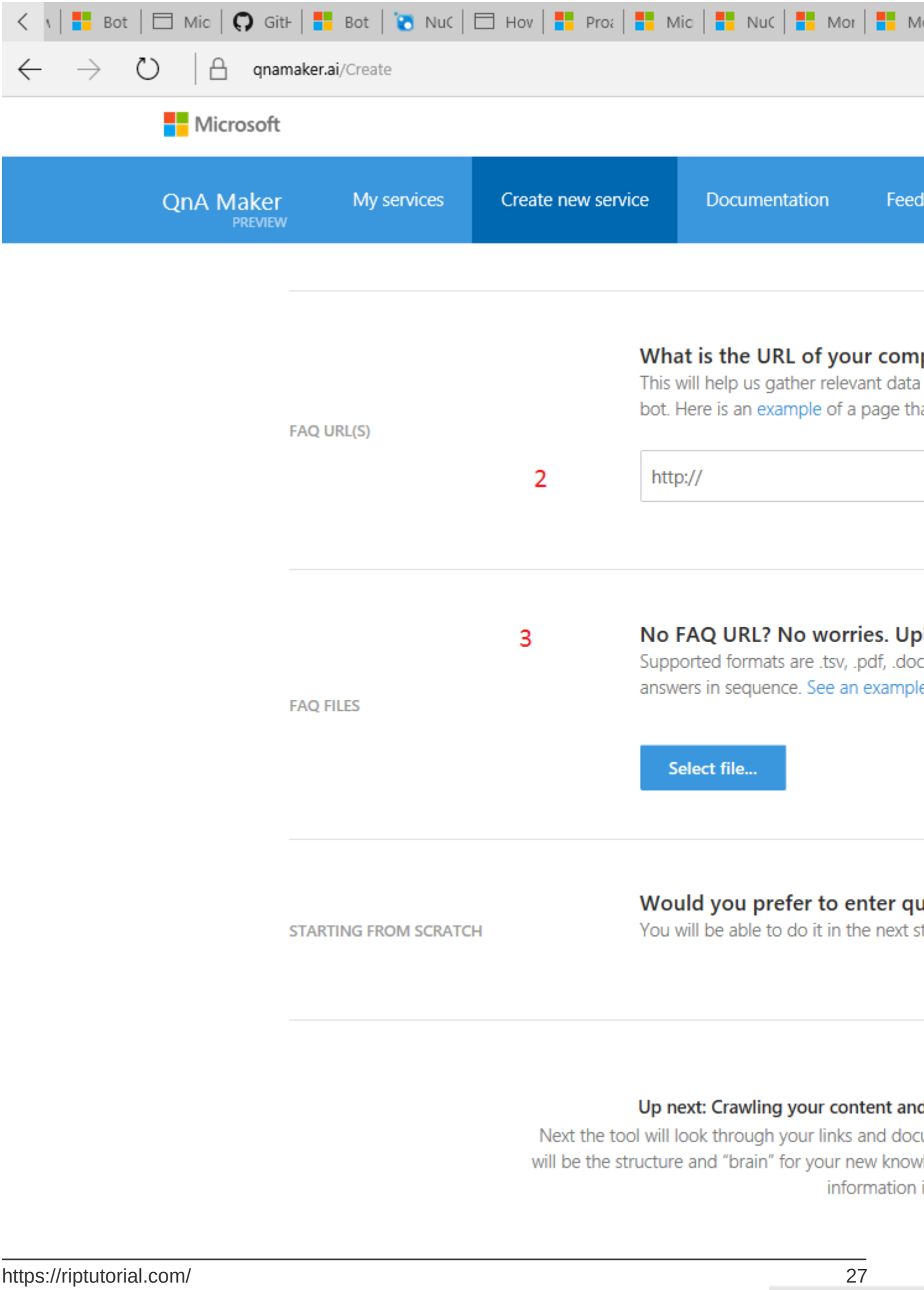
Examples

Creating our own QnA Service manually

Providing your microsoft account credentials you can authenticate and receive subscription keys to start with the services. This document describes the various flows in the tool to create your own knowledge base.



QnA Maker works in three steps: extraction, training and publishing. To start, feed it anything from existing FAQ URLs to documents and editorial content. I created my own question and answers manually.


qnamaker.ai/Create
Microsoft
QnA Maker
PREVIEW
My services
Create new service
Documentation
What is the URL of your comp
This will help us gather relevant data
bot. Here is an example of a page tha
FAQ URL(S)
2
http://
3
No FAQ URL? No worries. Up
Supported formats are .tsv, .pdf, .doc
answers in sequence. See an example
FAQ FILES
Select file...
Would you prefer to enter qu
STARTING FROM SCRATCH
You will be able to do it in the next st
Up next: Crawling your content and
Next the tool will look through your links and docu
will be the structure and "brain" for your new know
information i

and clicking on “Feedback” in the top navigation.

Read Getting Started with QnA Services online:
https://riptutorial.com/botframework/topic/9520/getting-started-with-qna-services

Credits

S. No	Chapters	Contributors
1	Getting started with botframework	Community, Eric Dahlvang, Ezequiel Jadib, Mr. Kaffe Kup, Rajat Jain
2	Adding Natural Language Processing	Rajat Jain
3	Getting started with Azure Bot Service	Eric Dahlvang, Jyo Fanidam
4	Getting Started with QnA Services	Jyo Fanidam