Developing and Deploying Intelligent Chat Bots

Microsoft

A Simple Tennis Bot

Objectives

The aim of this lab is to build a Simple Tennis bot.

The concept of a tennis bot is to send a tennis score to a bot and have the bot respond with a status of the game. For example, if the score is 1-1, the bot would respond with "We are making progress". If the score is 5-3, the bot would respond with "We are making good progress" and so on and so forth.

We will model the bot using dialogs to model a conversation and manage conversation flow. The dialogs are adapted from:

https://docs.botframework.com/en-us/csharp/builder/sdkreference/dialogs.html

.

Setup

Create a new Bot Project using Bot Application Template in Visual Studio as shown below:
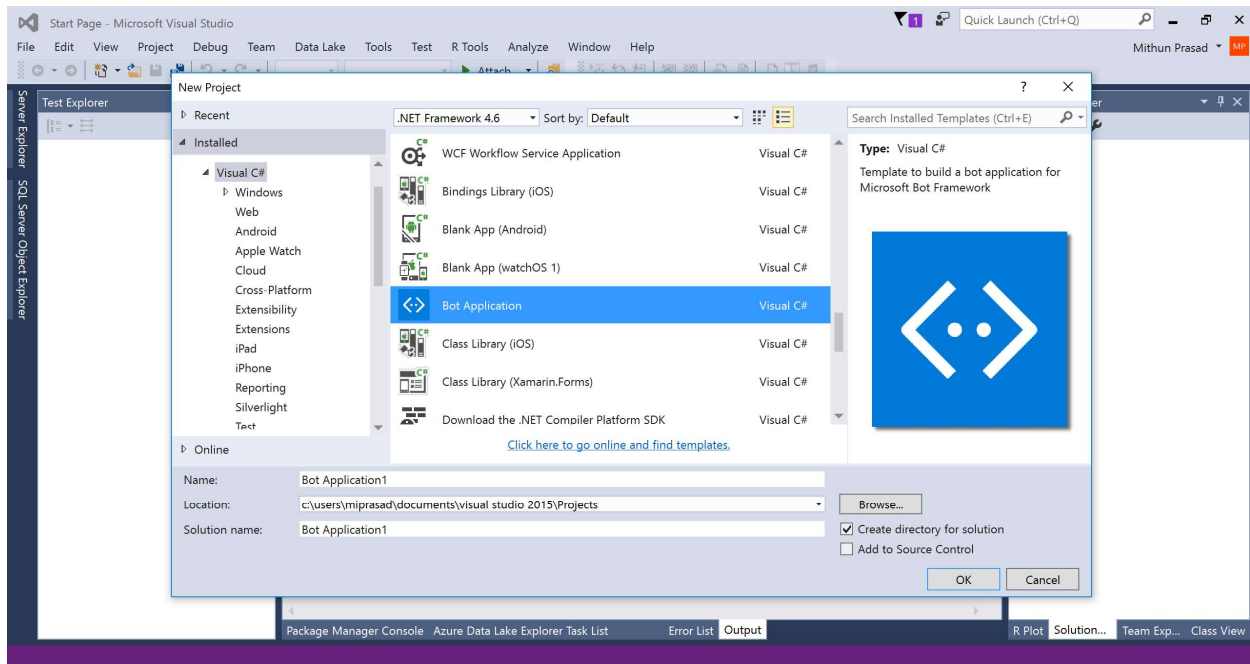
Once your project has been created, just take a moment to view what is included out of the box. Essentially, your bot application is just an ASP.NET Web API with a 'Messages' endpoint defined within the MessagesController.cs

Tennis Bot

From Bot Framework .NET template, perform the following steps to add Dialog functionality:

1. Add this namespace:

    using Microsoft.Bot.Builder.Dialogs;

2. Add this class:

```
[Serializable]
    public class SimpleDialog : IDialog
    {
        public async Task StartAsync(IDialogContext context)
        {
            context.Wait(ActivityReceivedAsync);
        }

        public async Task ActivityReceivedAsync(IDialogContext context,
        IAwaitable<object> result)
        {
            var activity = await result as Activity;

            String text = activity.Text;
            if (text.Contains("-"))
            {
                String[] stringTokens = text.Split('-');
                if(stringTokens.Length==2)
                {
                    int intScore = 0;
                    if(int.TryParse(stringTokens[0], out intScore))
                    {
                        if(intScore >= 6)
                            await context.PostAsync("Yaaay");
                        else if (intScore > 4)
                            await context.PostAsync("We are making good progress");
                        else if (intScore > 2)
                            await context.PostAsync("We are making progress");
                        else if (intScore >= 0)
                            await context.PostAsync("We are playing OK");
                    }
                }
            }

            context.Wait(ActivityReceivedAsync);
        }
    }
```

3. Replace the Post method with the below code. The Post method is marked async because Bot
    Builder uses the C# facilities for handling asynchronous communication. It returns a Task object,
    which represents the task that is responsible for sending replies to the passed-in message.

    Most of the action in our application takes place in the Post method of the MessagesController
    class and it is this method that handles receiving the message from the end user and the action of
    replying to it.

```
public virtual async Task<HttpResponseMessage> Post([FromBody] Activity
activity)
{
    if (activity!=null && activity.GetActivityType()==ActivityTypes.Message)
    {
```
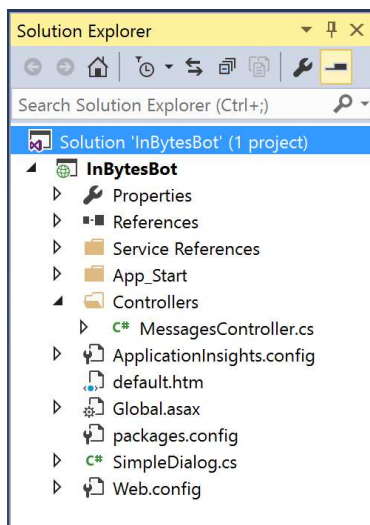
```
            await Conversation.SendAsync(activity, () => new SimpleDialog());
        }
        else
        {
            HandleSystemMessage(activity);
        }
        var response = Request.CreateResponse(HttpStatusCode.OK);
        return response;
    }
```
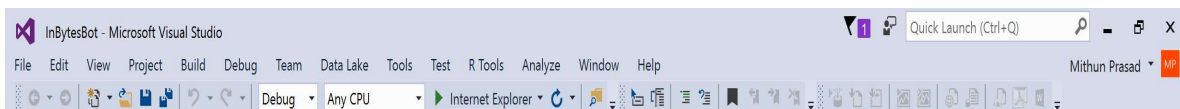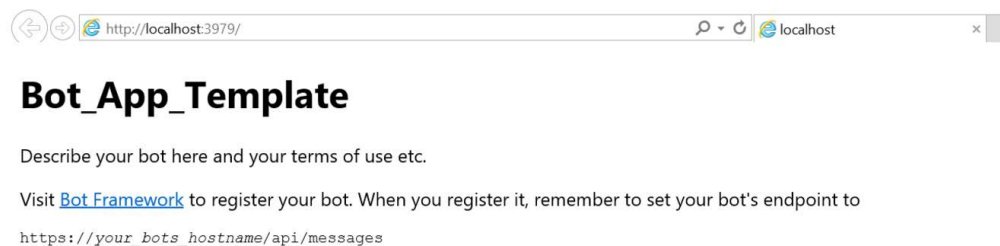
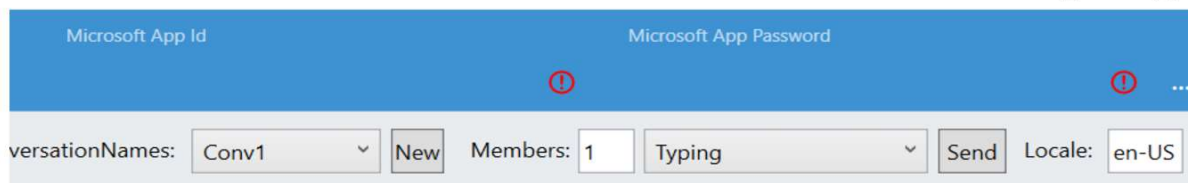The solution explorer would look as follows:

Run from Visual Studio by clicking *Internet Explorer* as shown below.

This should open a browser window with the endpoint (showing the default.htm file from project).

Bot_App_Template

Describe your bot here and your terms of use etc.

Visit Bot Framework to register your bot. When you register it, remember to set your bot's endpoint to

```
https://your_bots_hostname/api/messages
```

4. Open the bot emulator and change the Bot Url to http://localhost:3979/api/messages. You will need to add /api/messages to the Url. Ensure the fields Id and password fields are clear.

5. Test the bot by sending messages such as "hi", "hello", etc. Send a tennis score such as "3-1" and see if the bot responds to the score.

Exercise

1. Notice how the code only looks at the first part of the scores (for example, 6 in 6-1). Can you change the logic to incorporate the second part of the score?
2. Integrate scores as part of sentences. For example, "the score now is 6-3 I think"