Developing and Deploying Intelligent Chat Bots

Microsoft

Testing Bots

Objectives

Writing code using Microsoft Bot Framework is fun and exciting. But before rushing to code bots, you need to think about unit testing your code.

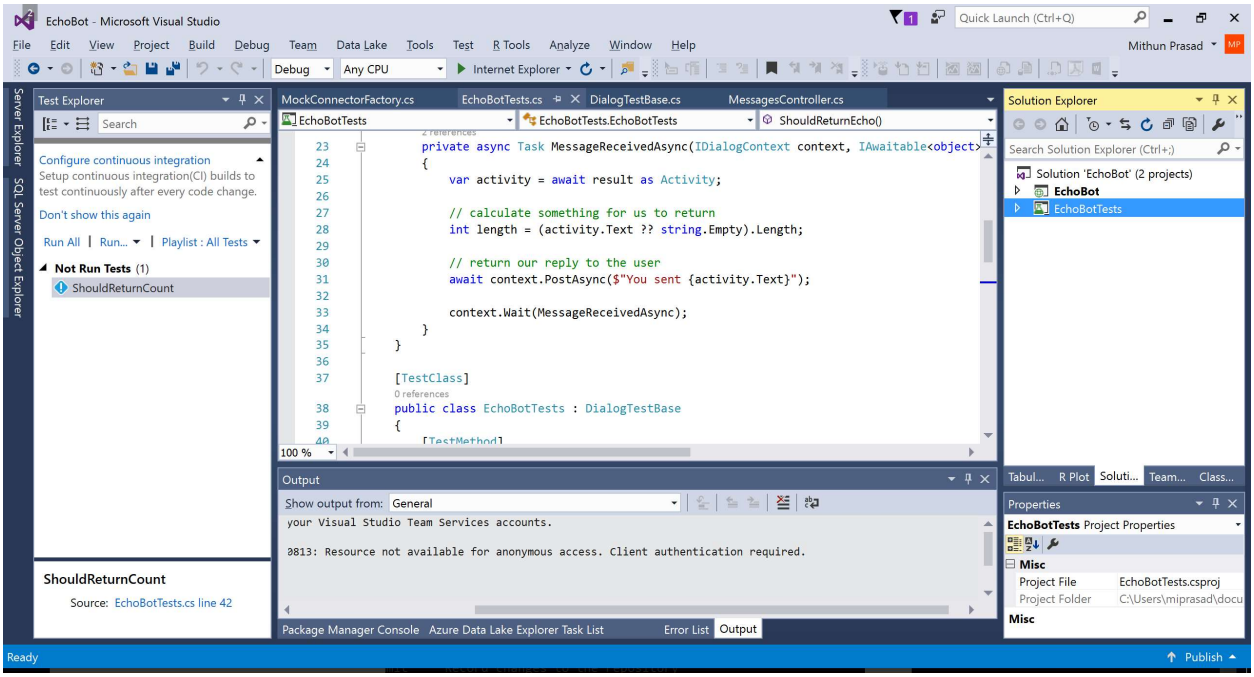
Unit tests can help:

- Verify functionality as you add it
- Validate your components in isolation
- People unfamiliar with your code verify they haven't broken it when they are working with it

The goal of this lab is to introduce unit testing for bots developed using Microsoft Bot Framework.

Setup

Import the EchoBot Solution in VisualStudio from Student-Resources/Labs/CSharp/UnitTests. On successful import, you will see two projects (EchoBot, a Bot Application and EchoBotTests, a Unit Test Project) as shown below.



EchoBot

In this lab, we will use EchoBot to develop unit tests. EchoBot is a very simple bot that echos back to the user with any message typed. For example, if the user types “Hello”, EchoBot responds with the message “You sent: Hello”. The core of EchoBot code that uses Dialogs can be found below. MessageReceivedAsync echos back to the user with “You sent:… ”

```
public class EchoDialog : IDialog<object>
{
    public async Task StartAsync(IDialogContext context)
    {
        context.Wait(MessageReceivedAsync);
```

```
    }

    public async Task MessageReceivedAsync(IDialogContext context,
    IAwaitable<IMessageActivity> argument)
    {
        var message = await argument;
        await context.PostAsync("You said: " + message.Text);
        context.Wait(MessageReceivedAsync);
    }
}
```

EchoBot – Unit Tests

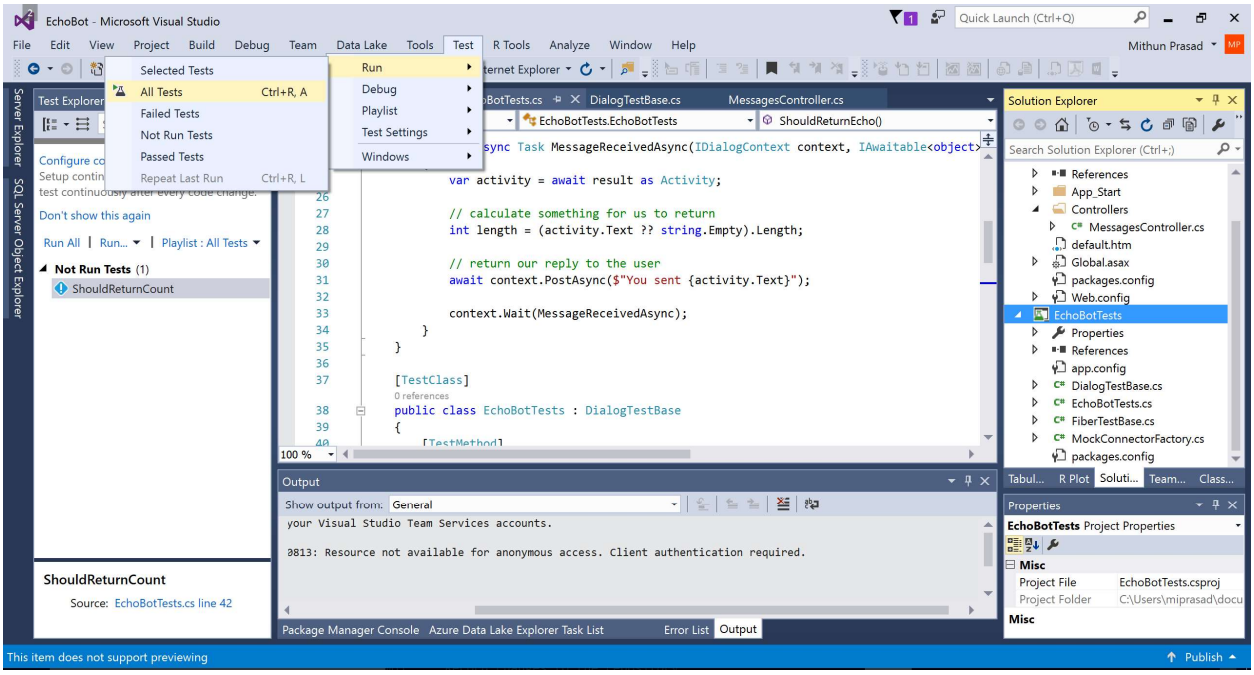
Unit tests can be created using Visual Studio's *Unit Test Project* within the EchoBot Solution. On importing EchoBot.sln, you will see EchoBotTests project which is a Unit Test Project. This project contains a few helper classes (developed by reusing the Bot Builder code) to help develop Unit Tests for Dialogs:

- DialogTestBase.cs
- FiberTestBase.cs
- MockConnectorFactory.cs

Inside EchoBotTests.cs, you will find a TestMethod called *ShouldReturnEcho*. *ShouldReturnEcho* verifies the result from EchoBot. The below line in EchoBotTests.cs mocks the behavior of EchoBot using RootDialog. RootDialog is used to provide the functionality of EchoBot.

```
using (new FiberTestBase.ResolveMoqAssembly(rootDialog))
```

Run all Tests by selecting Test -> Run -> All Tests as shown below and verify the tests run successfully.



Exercises

1. Write another TestMethod called *EchoStartsWith* that verifies the echo prompt begins with "You sent".
2. What sort of unit tests can you develop for an AlarmBot?