Lab 03: Code Sharing Existing Projects

Prerequisites

You will need a development environment, either a Mac or Windows PC with the Android SDK, iOS SDK and Xamarin tools installed. We will be using the Android and iOS emulators to test the code we are building, so make sure to have a virtual device already configured and ready to run. See the Cross-Platform Installation Guild if you need help getting your environment setup.

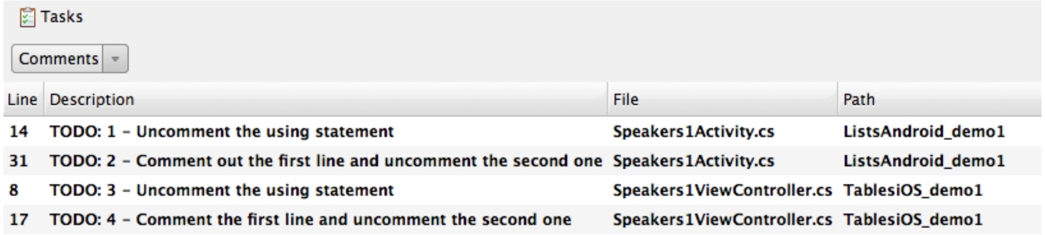
Downloads

Included with this lab document is a folder with resources that you will need in order to complete the lab. The folder name is **Fundamentals Introduction to Cross Platform Resources**. Make sure you have this folder before you begin.

Lab Goals

The goal of this lab is to take an existing Android and iOS mobile project, combine them into a single solution and move the shared code into a library project and use file linking to maximize code re-use while still delivering high-quality native experience on all of the platforms.

The lab has been provided as a starter solution with most of the code already filled in for you – as you following along with the instructor you will make small changes for each step, either writing a little code or uncommenting a block of code. Most of these steps are clearly marked in the supplied solution with `// TODO:` comments. These comments are picked up by Xamarin Studio and shown in the Task Pad, which you can make visible either by clicking the Tasks button in the status bar of the application, or through the **View > Pads > Tasks** menu item. When the Tasks Pad is open, it will look like this:



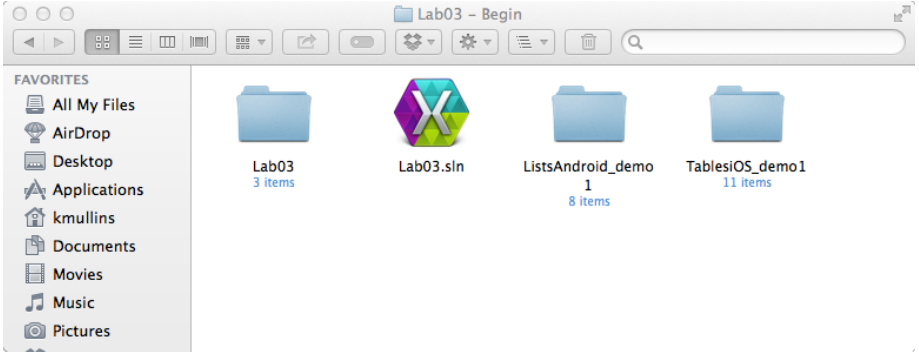
Line	Description	File	Path
14	TODO: 1 – Uncomment the using statement	Speakers1Activity.cs	ListsAndroid_demo1
31	TODO: 2 – Comment out the first line and uncomment the second one	Speakers1Activity.cs	ListsAndroid_demo1
8	TODO: 3 – Uncomment the using statement	Speakers1ViewController.cs	TablesiOS_demo1
17	TODO: 4 – Comment the first line and uncomment the second one	Speakers1ViewController.cs	TablesiOS_demo1

Steps

Open the Begin Solution

1. Launch **Xamarin Studio** using Spotlight or the application icon

2. Click **Open…** on the Xamarin Studio Welcome Screen and navigate to the **Fundamentals Introduction to Cross Platform Resources** folder included with this document
3. Locate the **Lab 03 - Begin** folder – make sure it's the starter and not the completed folder.
4. Inside the **Lab 03 - Begin** folder you will find a `Lab03.sln` file – double click on this file to open the starter solution:

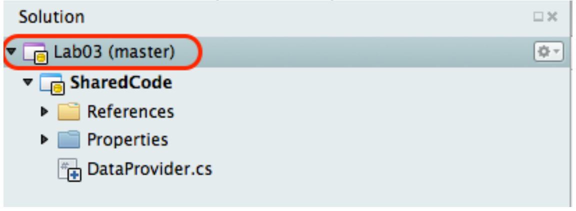


5. Go ahead and build and run the solution to make sure it compiles and your environment is ready.

Adding the Android Project

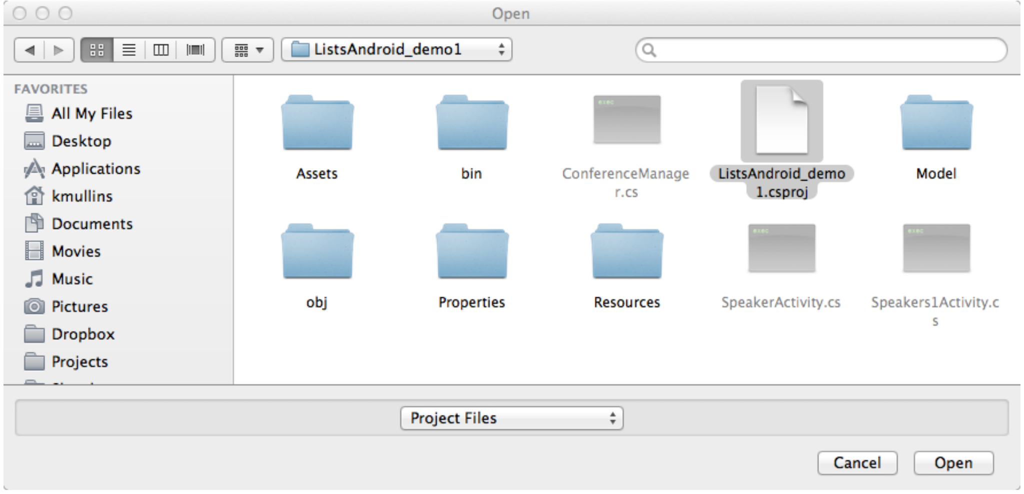
We are going to add an existing Android Project to our solution so we can move the shared code using linked files.

1. Select **Lab03 (master)** from the **Source Tree**:



2. Right click on **Lab03 (master)** and select **Add** > **An Existing Project…**

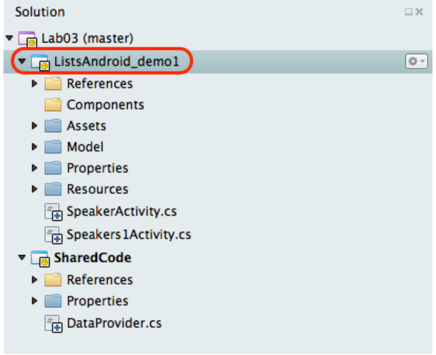
3. Navigate to the **ListsAndroid_demo1** folder, select **ListsAndroid_demo1.csproj** and click **Open**:



Linking to the Common Code

Next we are going to share the common code with our Android project using file linking to the Data Provider in the library project provided with this solution.

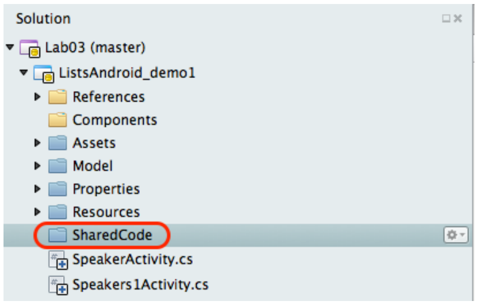
1. Select the **ListsAndroid_demo1** project in the **Source Tree**:



2. Right click on **ListsAndroid_demo1** project and select **Add** > **New Folder**.

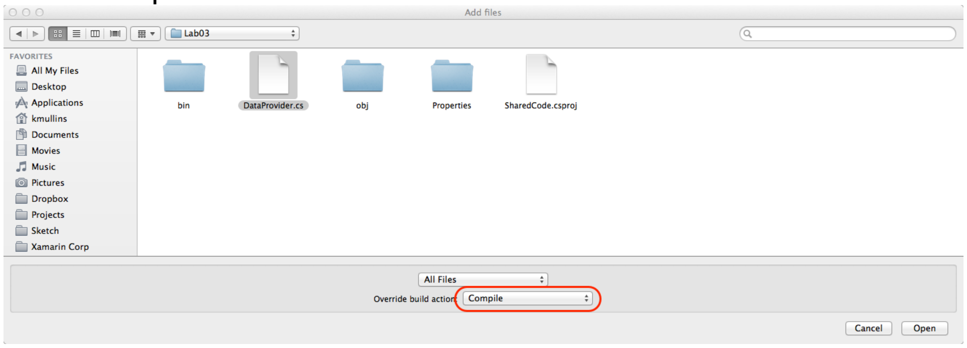
3. Enter `SharedCode` for the name and press **Return**.

4. Select the **SharedCode** folder in the **Source Tree**:



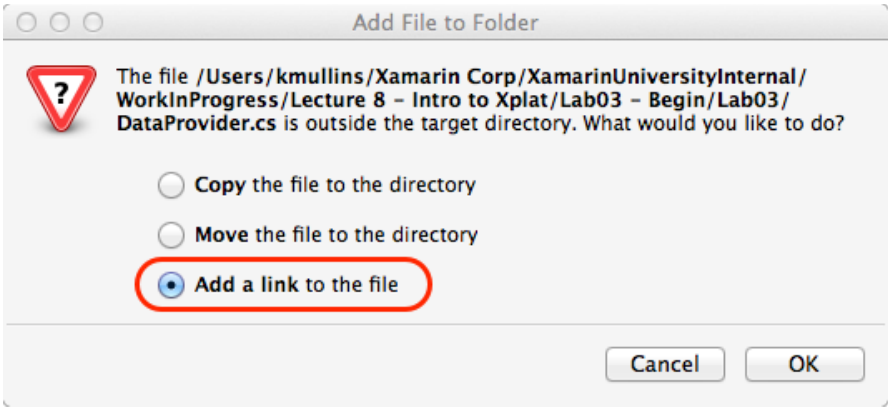
5. Right click on **SharedCode** and select **Add** > **Add Files…**

6. Navigate to the DataProvider.cs file in the SharedCode library project, select the file, set Override Build Action to Compile and click the Open button:



Override build action: Compile

7. From the Add File to Folder dialog box, select Add a link to the file and click the OK button:



The file /Users/kmullins/Xamarin Corp/XamarinUniversityInternal/WorkInProgress/Lecture 8 – Intro to Xplat/Lab03 - Begin/Lab03/DataProvider.cs is outside the target directory. What would you like to do?

○ Copy the file to the directory

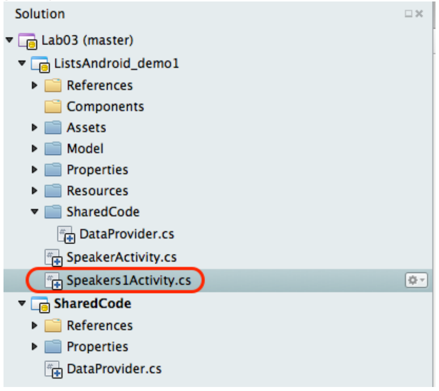
○ Move the file to the directory

● Add a link to the file

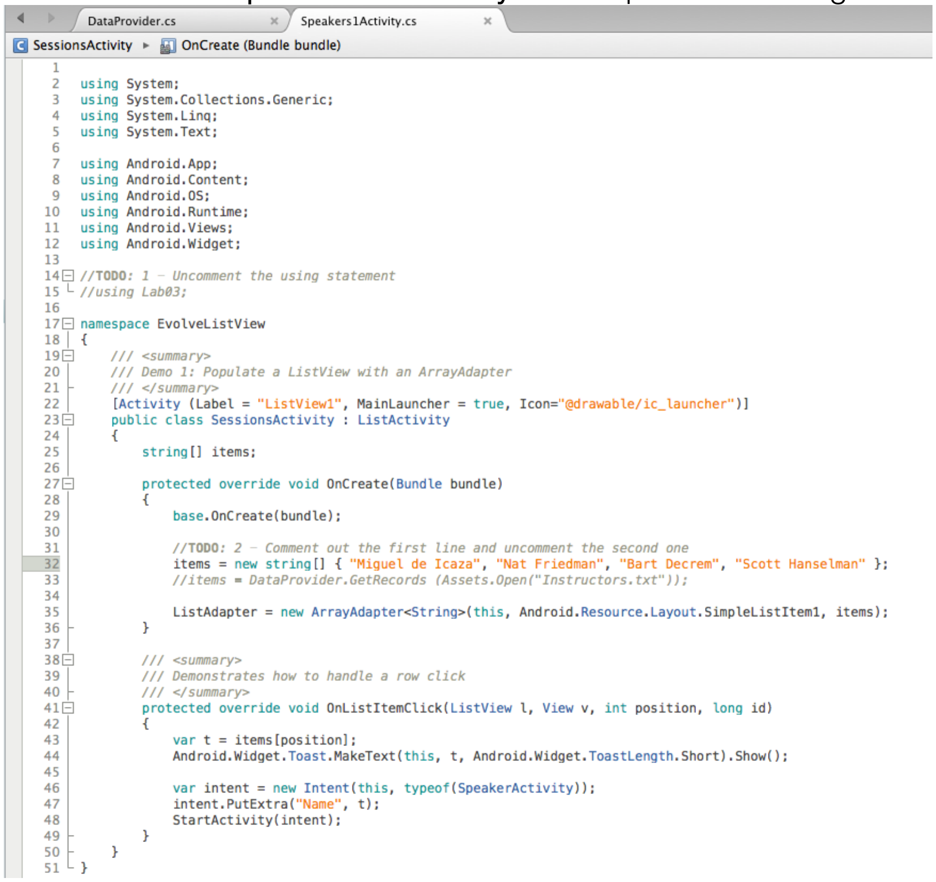
Consuming the Common Code

Next we are going to remove the common code from our Android project and replace it with the data provide from the linked data provider.

1. Select Speakers1Activity.cs file under the ListsAndroid_demo1 project:



2. Double click on `Speakers1Activity.cs` to open it for editing:



```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

using Android.App;
using Android.Content;
using Android.OS;
using Android.Runtime;
using Android.Views;
using Android.Widget;

//TODO: 1 - Uncomment the using statement
//using Lab03;

namespace EvolveListView
{
    /// <summary>
    /// Demo 1: Populate a ListView with an ArrayAdapter
    /// </summary>
    [Activity (Label = "ListView1", MainLauncher = true, Icon="@drawable/ic_launcher")]
    public class SessionsActivity : ListActivity
    {
        string[] items;

        protected override void OnCreate(Bundle bundle)
        {
            base.OnCreate(bundle);

            //TODO: 2 - Comment out the first line and uncomment the second one
            items = new string[] { "Miguel de Icaza", "Nat Friedman", "Bart Decrem", "Scott Hanselman" };
            //items = DataProvider.GetRecords (Assets.Open("Instructors.txt"));

            ListAdapter = new ArrayAdapter<String>(this, Android.Resource.Layout.SimpleListItem1, items);
        }

        /// <summary>
        /// Demonstrates how to handle a row click
        /// </summary>
        protected override void OnListItemClick(ListView l, View v, int position, long id)
        {
            var t = items[position];
            Android.Widget.Toast.MakeText(this, t, Android.Widget.ToastLength.Short).Show();

            var intent = new Intent(this, typeof(SpeakerActivity));
            intent.PutExtra("Name", t);
            StartActivity(intent);
        }
    }
}
```

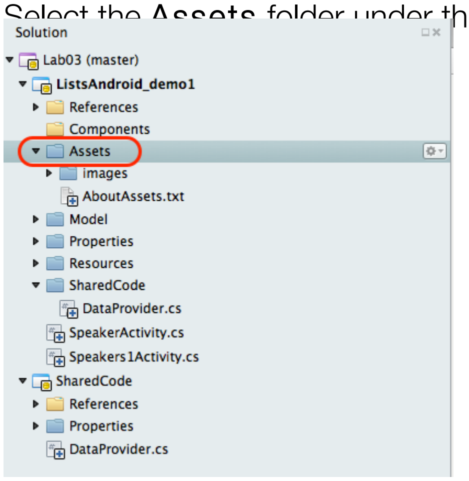
3. Locate the *//TODO: 1 - Uncomment the using statement* and uncomment the using statement:

```
using Lab02;
```

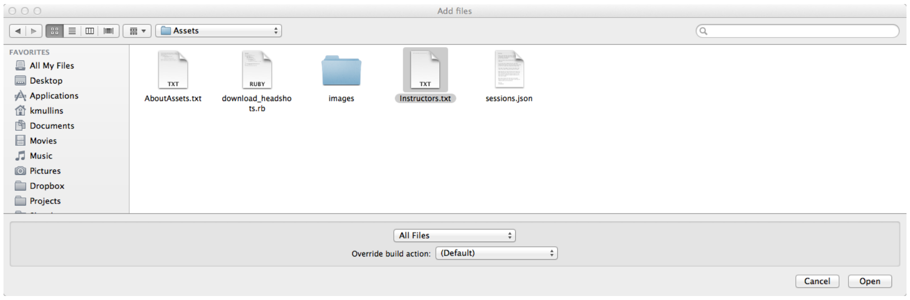
4. Locate the *//TODO: 2 - Comment out the first line and uncomment the second one* and comment the first line and uncomment the second one:

```
//items = new string[] { "Miguel de Icaza", "Nat Friedman", "Bart Dec
rem", "Scott Hanselman" };
items = DataProvider.GetRecords (Assets.Open("Instructors.txt"));
```

5. Select the **Assets** folder under the **ListsAndroid_demo1** project:



6. Right click on **Assets** and select **Add** > **Add Files…**
7. Select the **Instructors.txt** file and click the **OK** button:



All Files

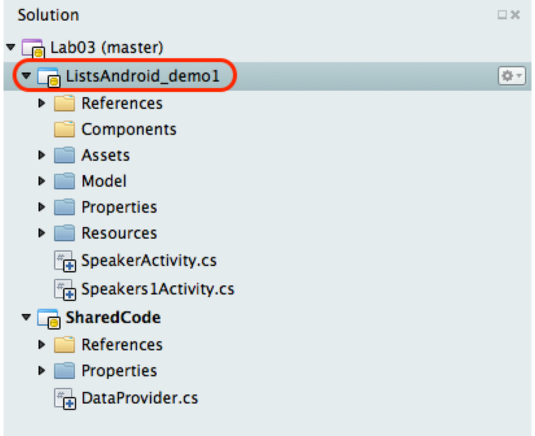
Override build action: (Default)

8. **Save** the project.
9. **Build** the project and ensure that there are no errors.
10. Let the instructor know if you have any trouble.

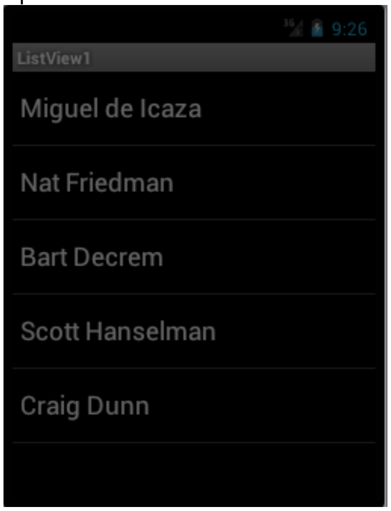
Testing the Android Application

Next we will test our Android application and assure that it is consuming the shared data provided by the linked data provider.

1. Select the **ListsAndroid_demo1** project from the **Source Tree**:



2. Right click on ListsAndroid_demo1 and select Set As Startup Project.
3. Click on the Run menu and select Start Debugging.
4. The application will run in the Android emulator and display a list of speakers:



Miguel de Icaza

Nat Friedman

Bart Decrem

Scott Hanselman

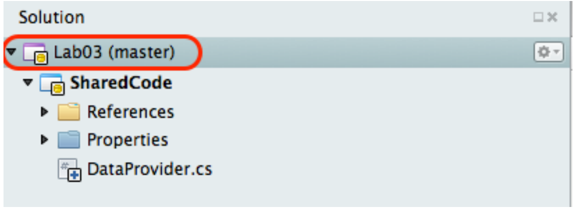
Craig Dunn

5. Stop debugging.

Adding the iOS Project

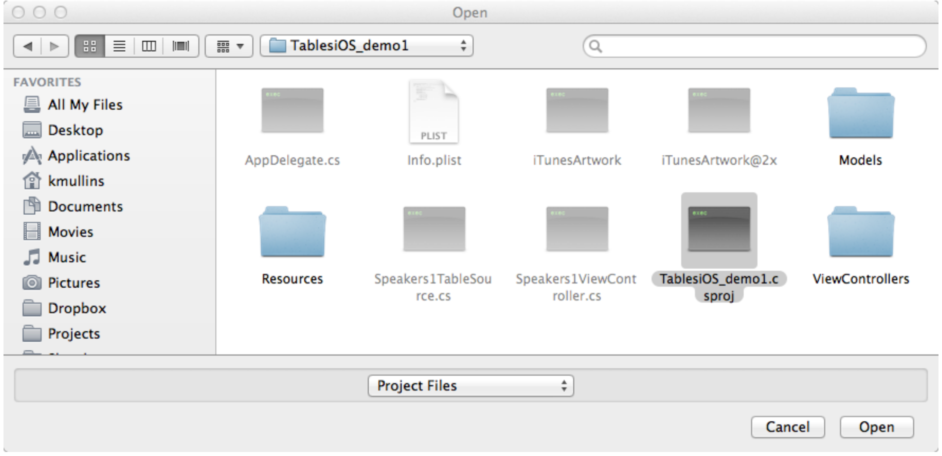
Like the Android project above, we are going to add an existing iOS project to our solution and link in the common shared code.

1. Select Lab03 (master) from the Source Tree:



2. Right click on Lab03 (master) and select Add > An Existing Project…

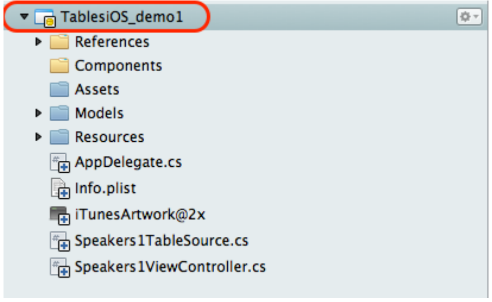
3. Navigate to the **TablesiOS_demo1** folder, select **TablesiOS_demo1.csproj** and click **Open**:



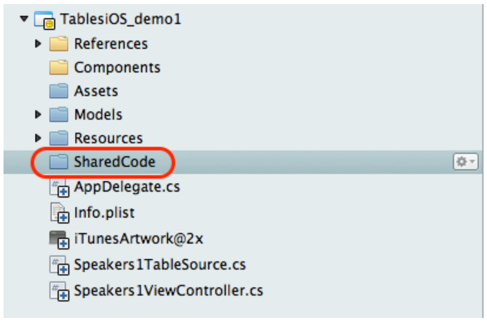
Linking to the Common Code

Next we need to add a reference to the PCL solution so that the iOS project can access the shared data provider.

1. Select the **TablesiOS_demo1** project in the **Source Tree**:

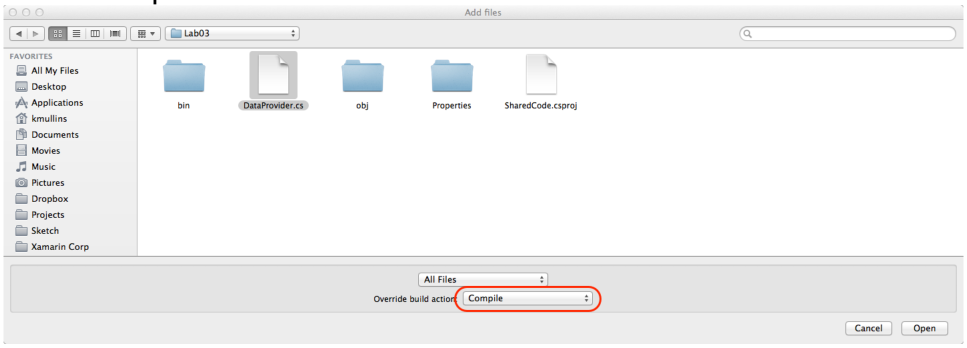


2. Right click on **TablesiOS_demo1** and select **Add > New Folder**
3. Enter **SharedCode** for the name and press **Return**.
4. Select the **SharedCode** folder in the **Source Tree**:



5. Right click on **SharedCode** and select **Add** > **Add Files…**
6. Navigate to the **DataProvider.cs** file in the **SharedCode** library project, select the file, set **Override Build Action** to **Compile** and

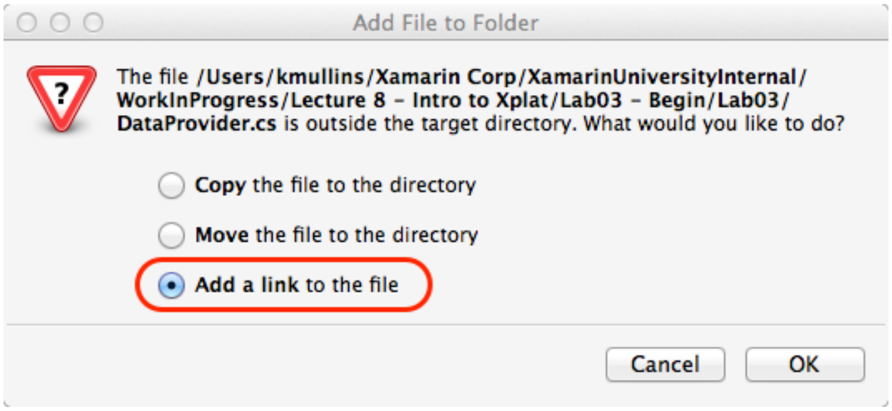
click the **Open** button:



All Files

Override build action: Compile

7. From the **Add File to Folder** dialog box, select **Add a link to the file** and click the **OK** button:



The file **/Users/kmullins/Xamarin Corp/XamarinUniversityInternal/WorkInProgress/Lecture 8 – Intro to Xplat/Lab03 – Begin/Lab03/DataProvider.cs** is outside the target directory. What would you like to do?

○ **Copy** the file to the directory

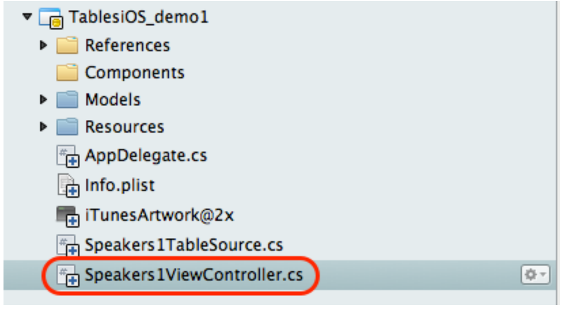
○ **Move** the file to the directory

● **Add a link** to the file

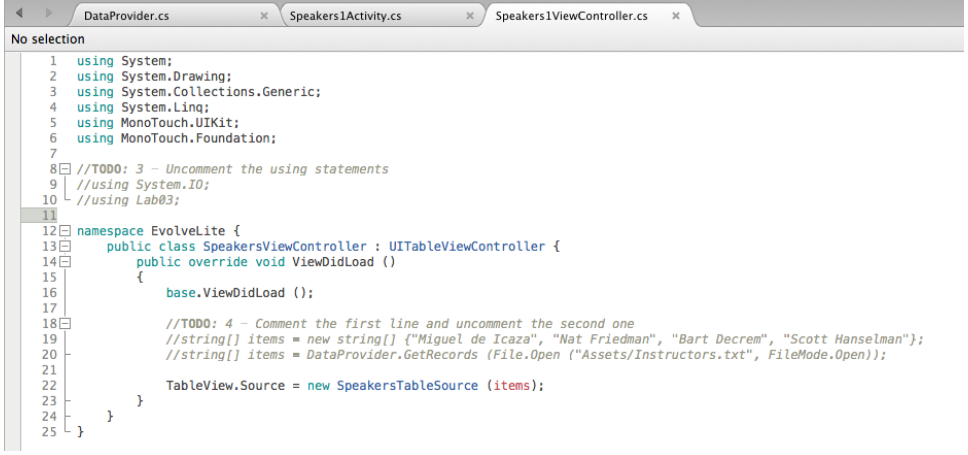
Consuming the Common Code

Next we are going to remove the common code from our iOS project and replace it with the data provide from the PCL included with the solution.

1. Select **Speakers1ViewController.cs** file under the **TablesiOS_demo1** project:



2. Double click on **Speakers1ViewController.cs** to open it for editing:



```
using System;
using System.Drawing;
using System.Collections.Generic;
using System.Linq;
using MonoTouch.UIKit;
using MonoTouch.Foundation;

//TODO: 3 - Uncomment the using statements
//using System.IO;
//using Lab03;

namespace EvolveLite {
    public class SpeakersViewController : UITableViewController {
        public override void ViewDidLoad ()
        {
            base.ViewDidLoad ();

            //TODO: 4 - Comment the first line and uncomment the second one
            //string[] items = new string[] {"Miguel de Icaza", "Nat Friedman", "Bart Decrem", "Scott Hanselman"};
            //string[] items = DataProvider.GetRecords (File.Open ("Assets/Instructors.txt", FileMode.Open));

            TableView.Source = new SpeakersTableSource (items);
        }
    }
}
```

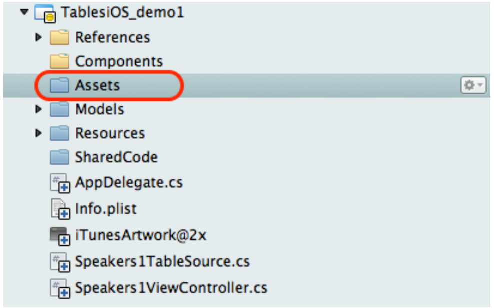
3. Locate the //TODO*: 3 - Uncomment the using statements* and uncomment the using statements:

```
using System.IO;
using Lab02;
```

4. Locate the //TODO*: 4 - Comment out the first line and uncomment the second one* and comment the first line and uncomment the second one:

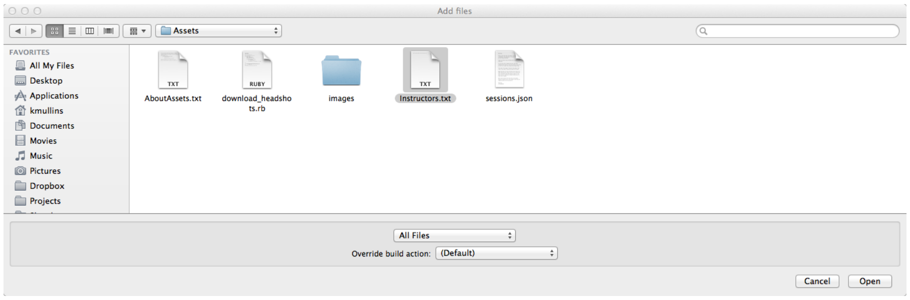
```
//string[] items = new string[] {"Miguel de Icaza", "Nat Friedman", "
Bart Decrem", "Scott Hanselman"};
string[] items = DataProvider.GetRecords (File.Open ("Assets/Instruct
ors.txt", FileMode.Open));
```

5. Select the Assets folder under the TablesiOS_demo1 project in the Source Tree:



6. Right click on the **Assets** folder and select **Add** > **Add Files…**

7. Select the **Instructors.txt** file and click the **OK** button:



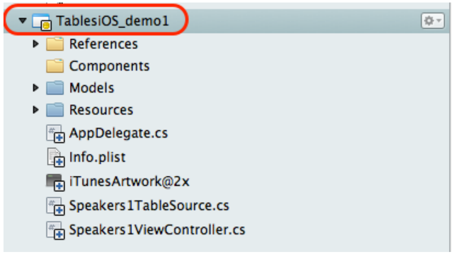
8. **Save** the project.

9. **Build** the project and ensure that there are no errors.

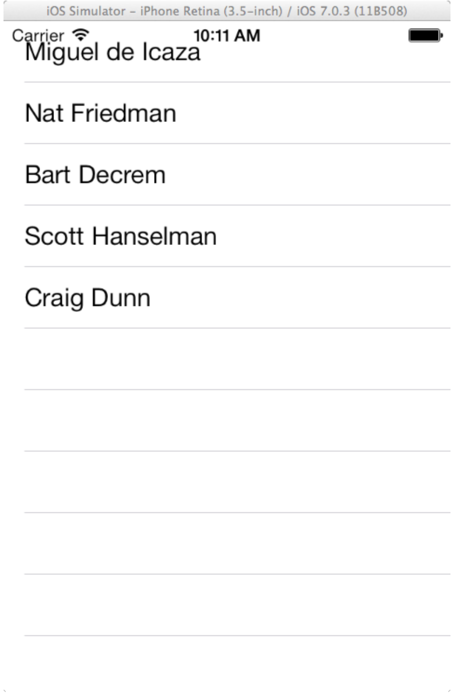
Testing the iOS Application

Next we will test our iOS mobile application and assure that it is consuming the shared data provided by the data provider in our PCL.

1. Select the **TablesiOS_demo1** project from the **Source Tree**:



2. Right click on **TablesiOS_demo1** and select **Set As Startup Project.**
3. Click on the **Run** menu and select **Start Debugging.**
4. The application will run in the iOS Simulator and the list of speakers will be displayed:


iOS Simulator – iPhone Retina (3.5-inch) / iOS 7.0.3 (11B508)
Carrier
10:11 AM

Miguel de Icaza

Nat Friedman

Bart Decrem

Scott Hanselman

Craig Dunn

5. Stop debugging.

Summary

In this lab we learned the difference between sharing code using a Portable Class Library (PCL) versus code-linking the same physical C# file into multiple platform specific application projects to maximize code reuse, while maintaining high quality native applications on each platform supported.