DEVELOPED YOUR FIRST XAMARIN.FORMS APP

Updated 2016.12.18

Developed your First Xamarin.Forms App

Prepared by Eng Soon Cheah, Microsoft MVP

Objective

Build your first mobile app for iOS,Android, and Windows with Xamarin.Forms

Problems and Solutions After installation of Xamarin in Visual Studio

- http://bit.ly/XamarinP1
- http://bit.ly/XamarinP2

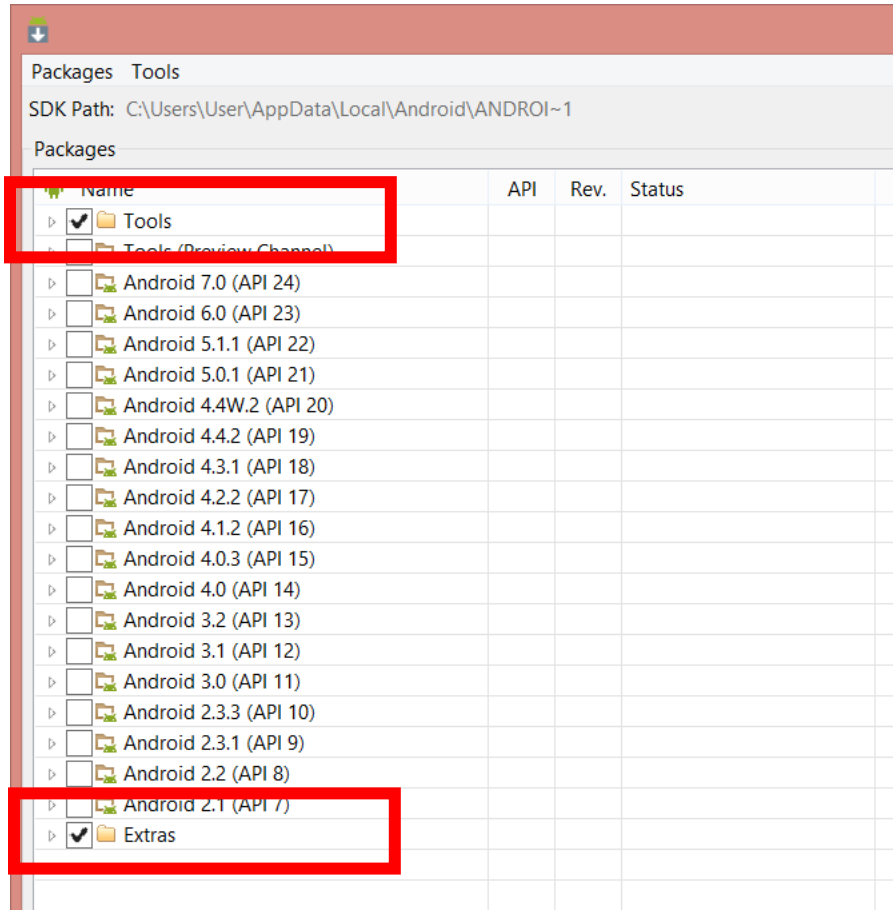
Requirements

1. Install Visual Studio 2015 Enterprise

2. Install Visual Studio Emulator for Android

3. Install Windows Standalone SDK for Windows 10 & Enable Developer Mode

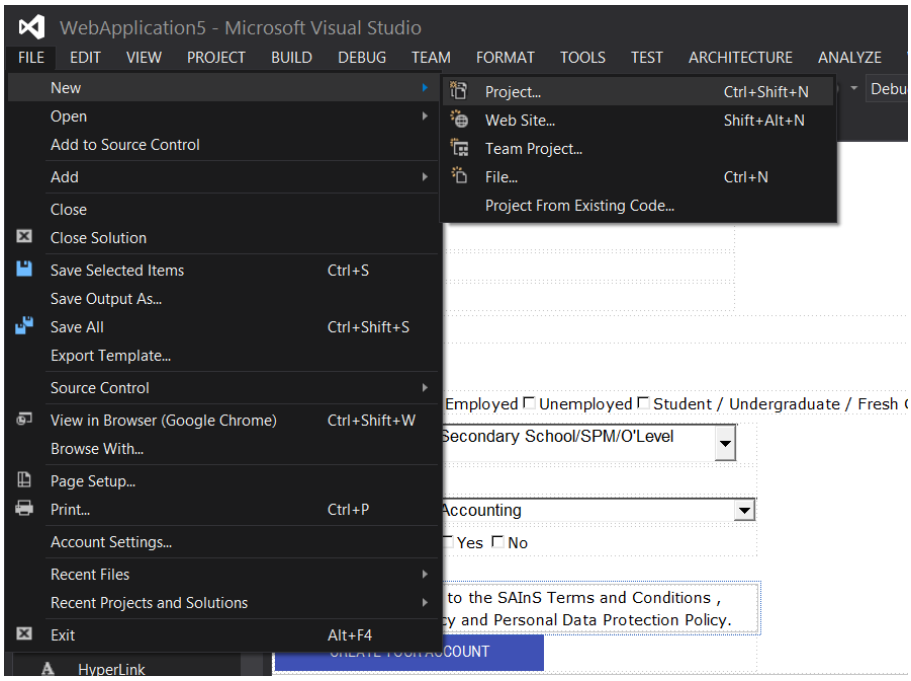
4. Install Android SDK Manager for Tools and Extras



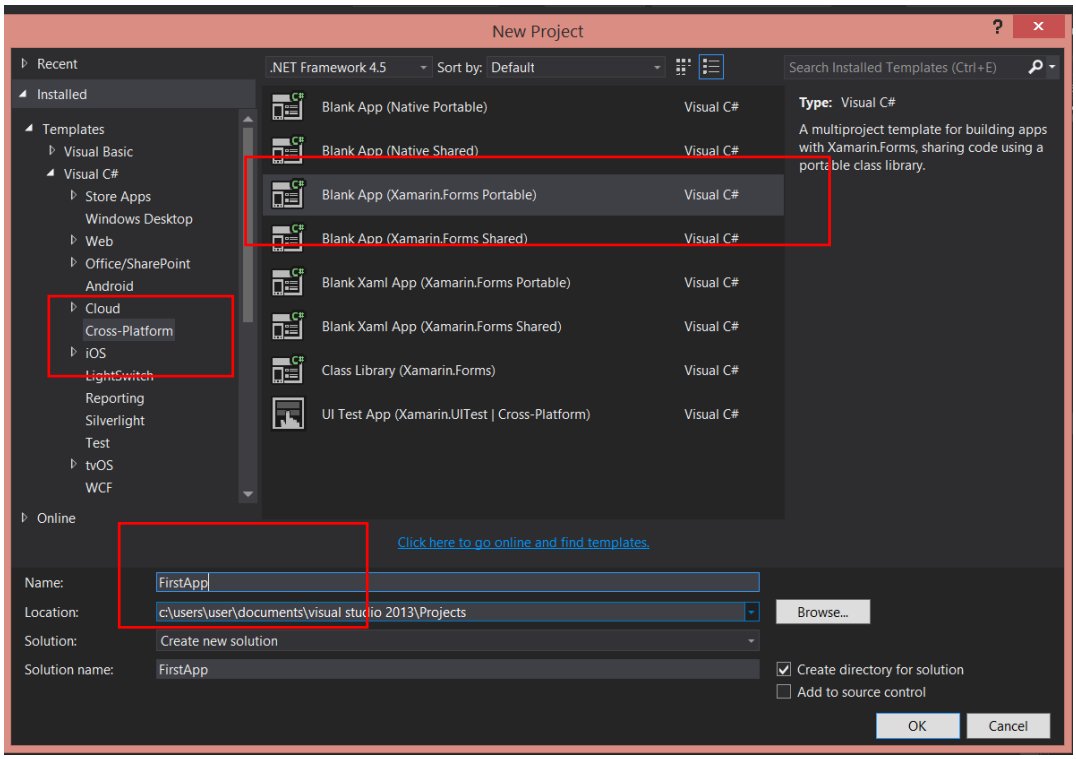
Content

Chapter 1: Create Your First Project

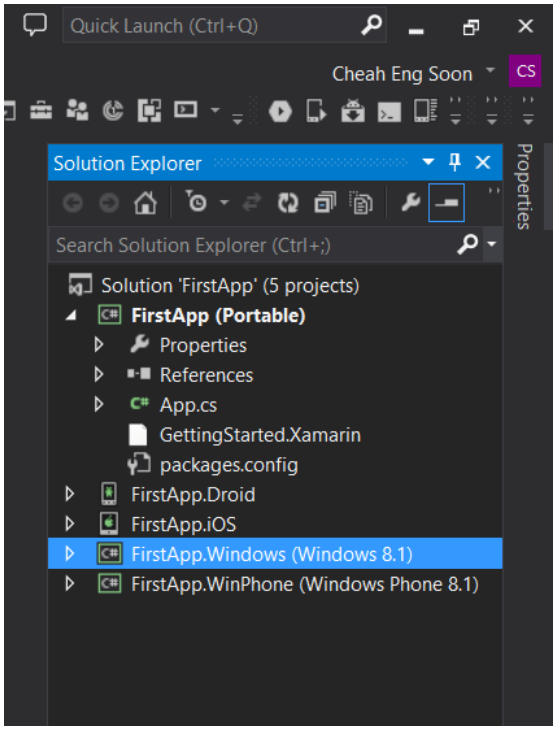
1. Open your Visual Studio 2015/ Visual Studio Enterprise.
2. Go to **File** > **New** > **Project**



3. Select "**Cross-Platform**" > Select " **Blank App (Xamarin.Forms Portable)** "& Name Your Project (App Name)

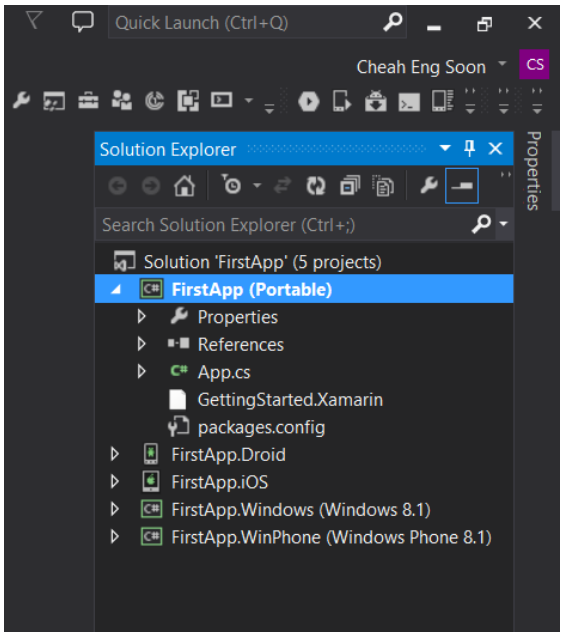


4. Your Xamarin.Forms App Solutions Successful Created.

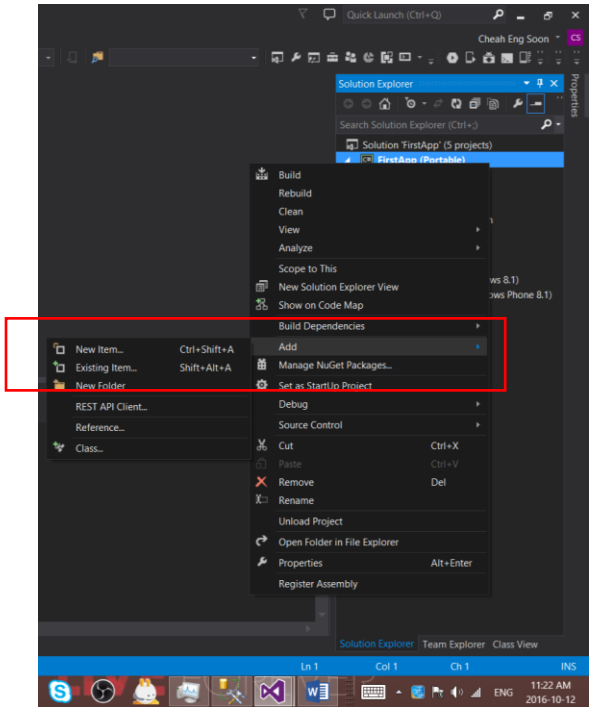


* If you use Windows 10 & install Windows 10 SDK, You will saw the Universal Windows Platform (UWP) App.

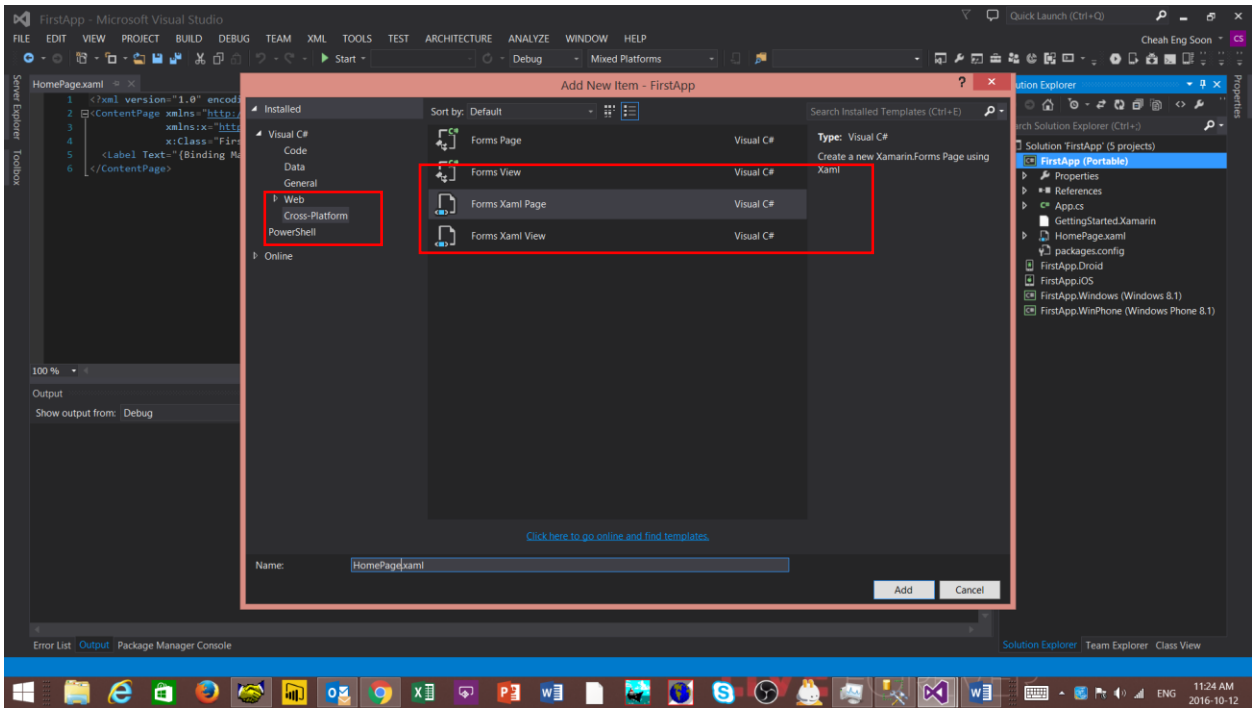
5. Select the Solution “FirstApp(Portable)”



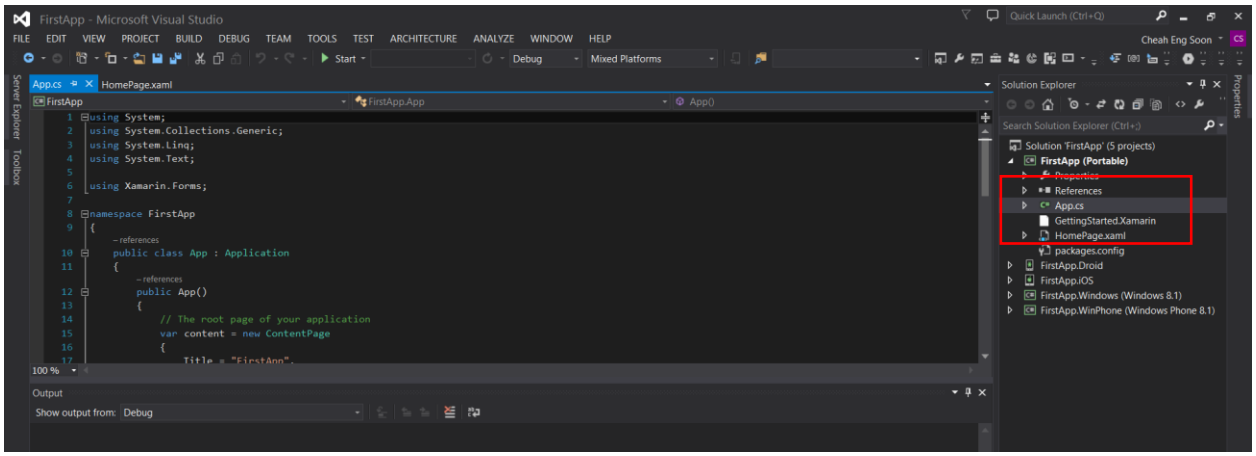
6. Right- Click the Solution “**FirstApp(Portable)**” > Select “**Add**” > Select “**New Item**”



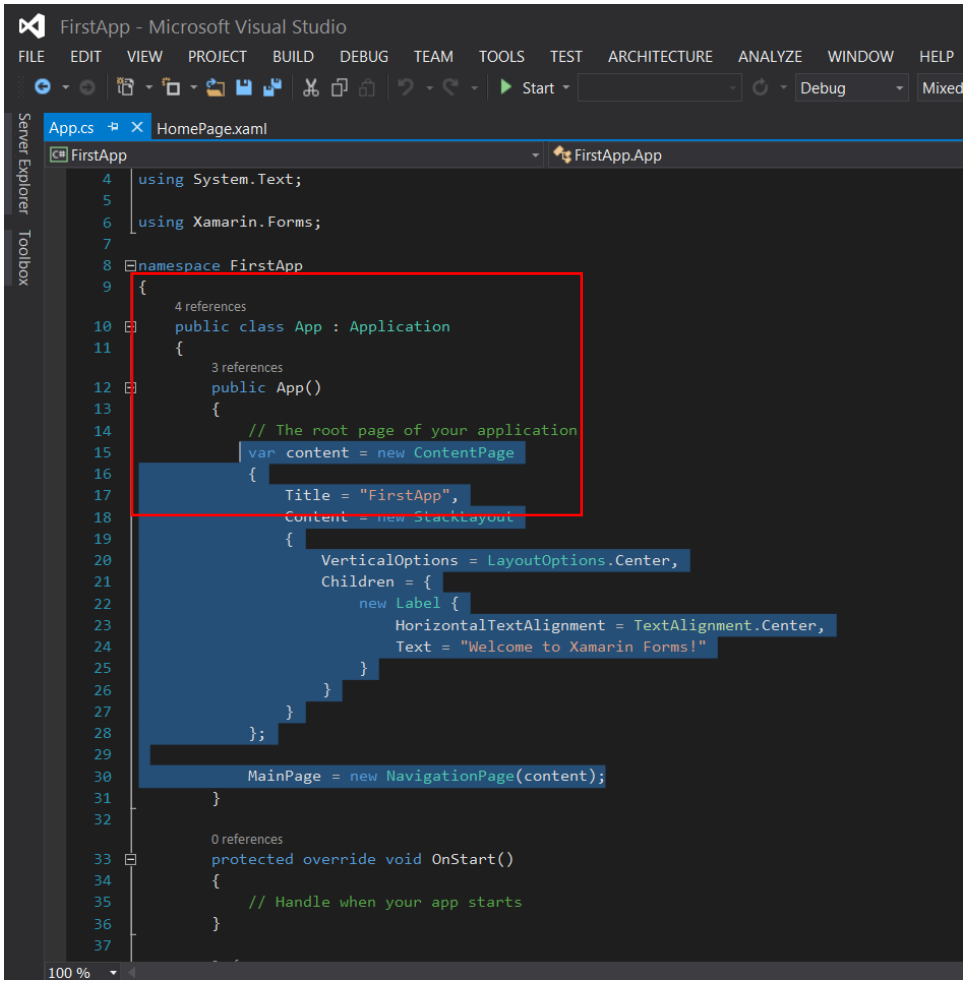
7. Select “**Cross-Platform**” > Select “**Forms Xaml Page**” > Name the Page “**HomePage.xaml**”



8. Go to “**FirstApp (Portable)**” > Select “**App.cs**”



9. Highlight and Remove the Source Code in **App()**

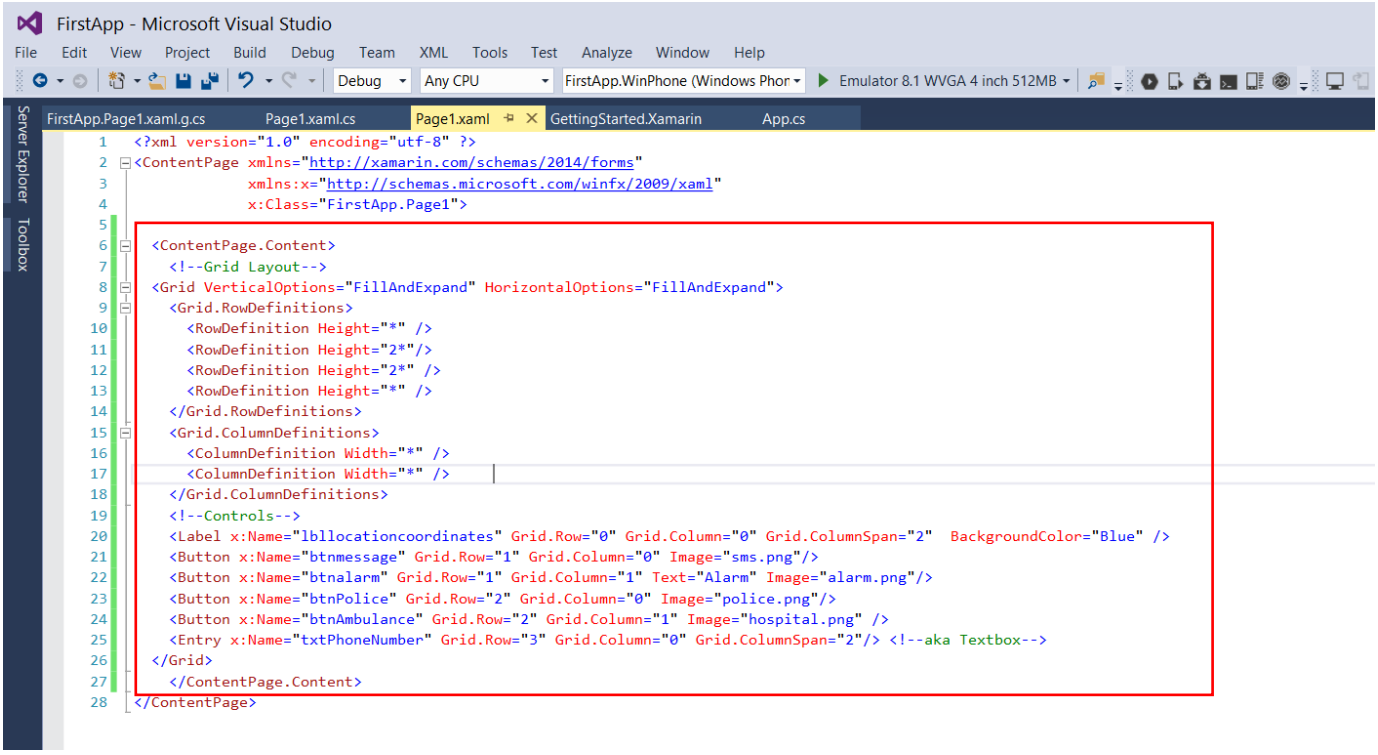


10. And Replace with “MainPage = new HomePage();”

```
using System.Text;

using Xamarin.Forms;

namespace FirstApp
{
    4 references
    public class App : Application
    {
        3 references
        public App()
        {


            MainPage = new HomePage();
        }

        0 references
```

Chapter 2: Layout and Control

1. Go to "**HomePage.xaml**" add Layout and Control

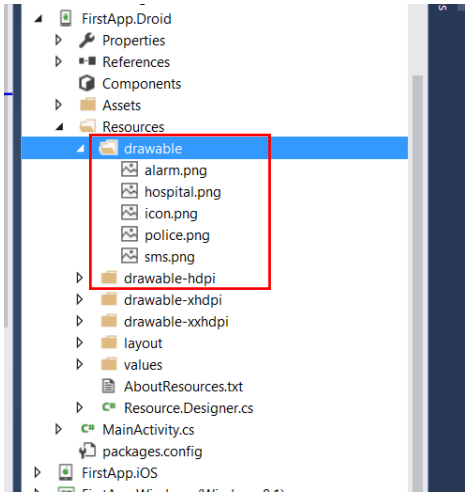


2. Add Image to each Platform (iOS,Android,Windows/UWP)

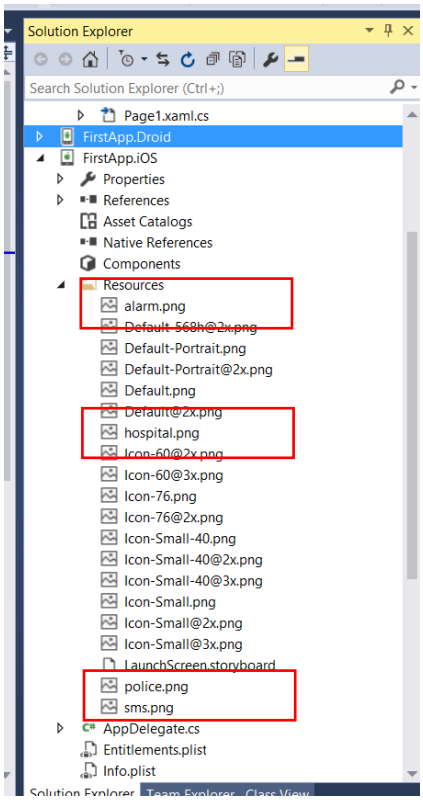
- iOS – Add Image to " **Resources**" Folder , **Build Action: BundleResource** in Properties Tab.
- Android – Add Image to "**Resources/drawable**" Folder , **Build Action:AndroidResource**.
- Windows /UWP – Place images in root directory , **Build Action: Content**.

Download the image and MP3 file from:

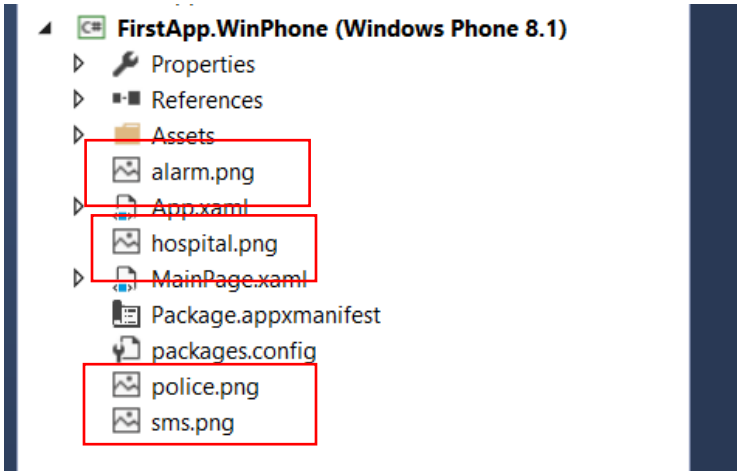
https://github.com/cheahengsoon/FirstXamarinFormsApp



Android Platform



iOS Platform



Windows Platform

Result:

Layout Done. But still not had any event or functions.

3. Add Event handler to the Entry for PhoneNumber.

```
    <Button x:Name="btnAmbulance" Grid.Row="2" Grid.Column="1" Image="hospital.png" />
    <Entry x:Name="txtPhoneNumber" Grid.Row="3" Grid.Column="0" Grid.ColumnSpan="2"
           Placeholder="Phone Number" TextChanged="PhoneNo_TextChanged" Keyboard="Numeric" /> <!--aka Textbox-->
  </Grid>
```

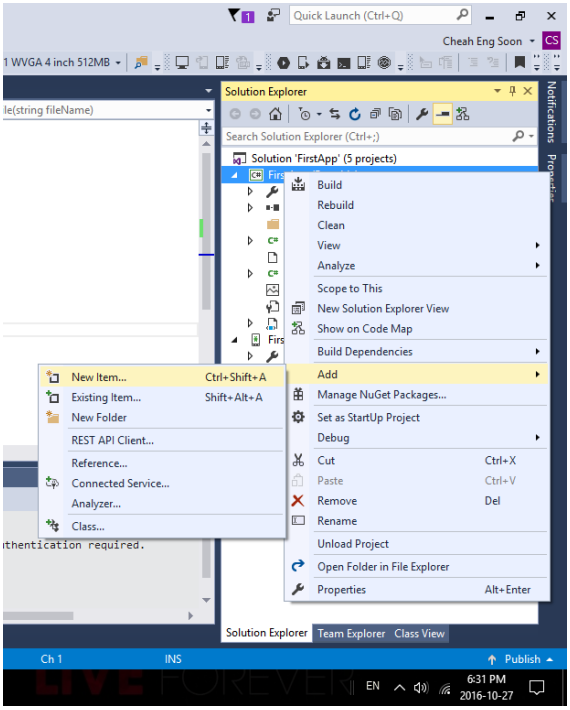
And

Go to HomePage.xaml.cs , add the C# Code

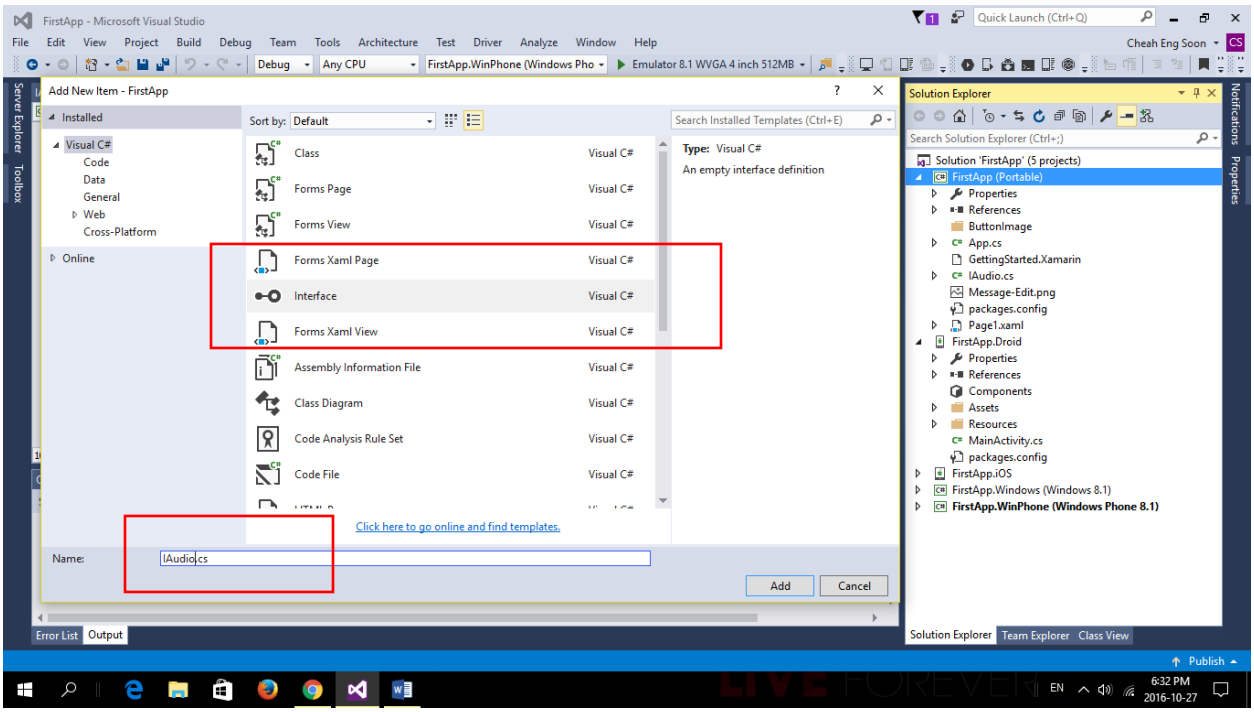
```
public partial class Page1 : ContentPage
{
    public Page1()
    {
        InitializeComponent();

        //Read the Phone Number from Temporary Storage
        if (Application.Current.Properties.ContainsKey("PhoneNo"))
        {
            var phonenumber = (string)Application.Current.Properties["PhoneNo"];
            txtPhoneNumber.Text = phonenumber;
        }
    }

    private void PhoneNo_TextChanged(object sender,TextChangedEventArgs e)
    {
        //Temporary Storage for Phone Number from txtxPhoneNumber
        var phonenumber = txtPhoneNumber.Text;
        Application.Current.Properties["PhoneNo"] = phonenumber;
    }
```

4. Play Audio for Alarm

1. Right Click "**FirstApp(Portable)**" Solution, select "**Add**", and "**New item**".

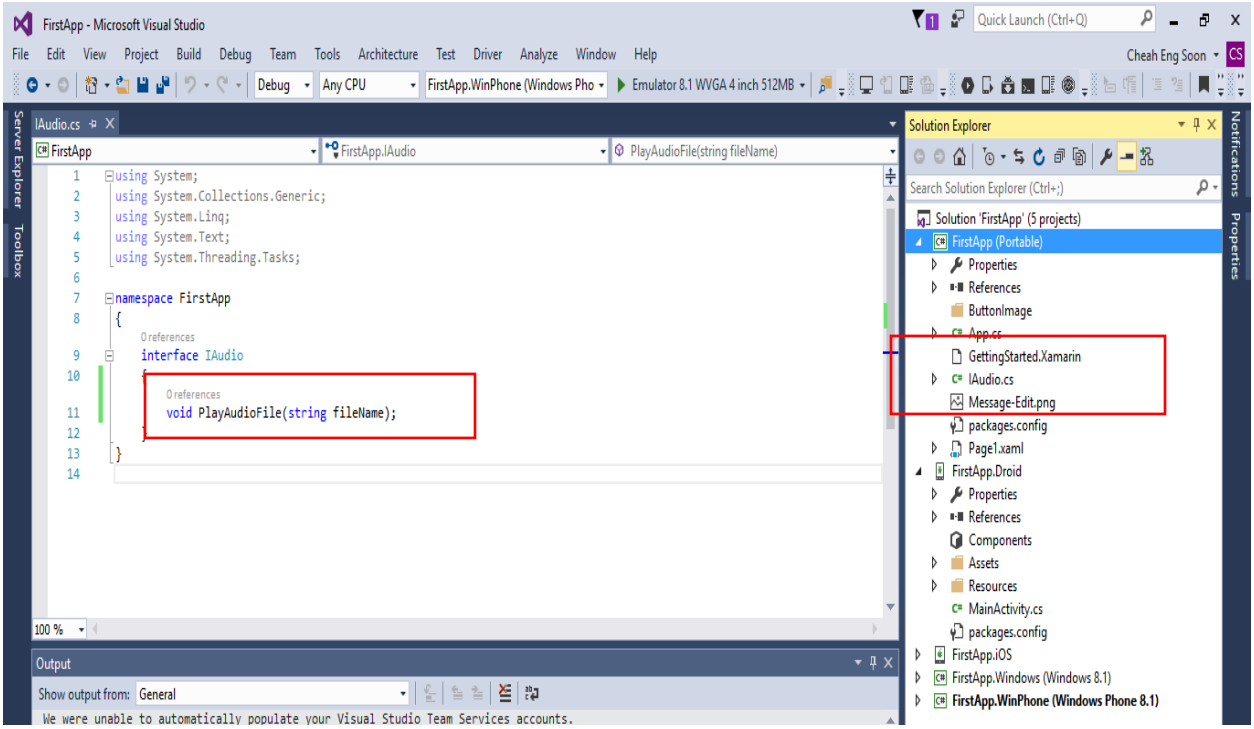


2. Select "**Interface**" and name the class as "**IAudio.cs**"



3. "**IAudio.cs**" Success created, And Insert this line of code as below.

void PlayAudioFile(string filename);



4. Go to "**HomePage.xaml"** add event handler for "**btnalarm**"

```
<Button x:Name="btnalarm" Grid.Row="1" Grid.Column="1" Text="Alarm" Image="alarm.png"
Clicked="OnAlertYesNoClicked" />
```

And

Add the backend function to play the alarm sound.

```
0 references
async void OnAlertYesNoClicked(object sender, EventArgs e)
{
    DependencyService.Get<IAudio>().PlayAudioFile("MySong.mp3");
}
```

5. Now add the "policeSiren.mp3" to iOS & Android.

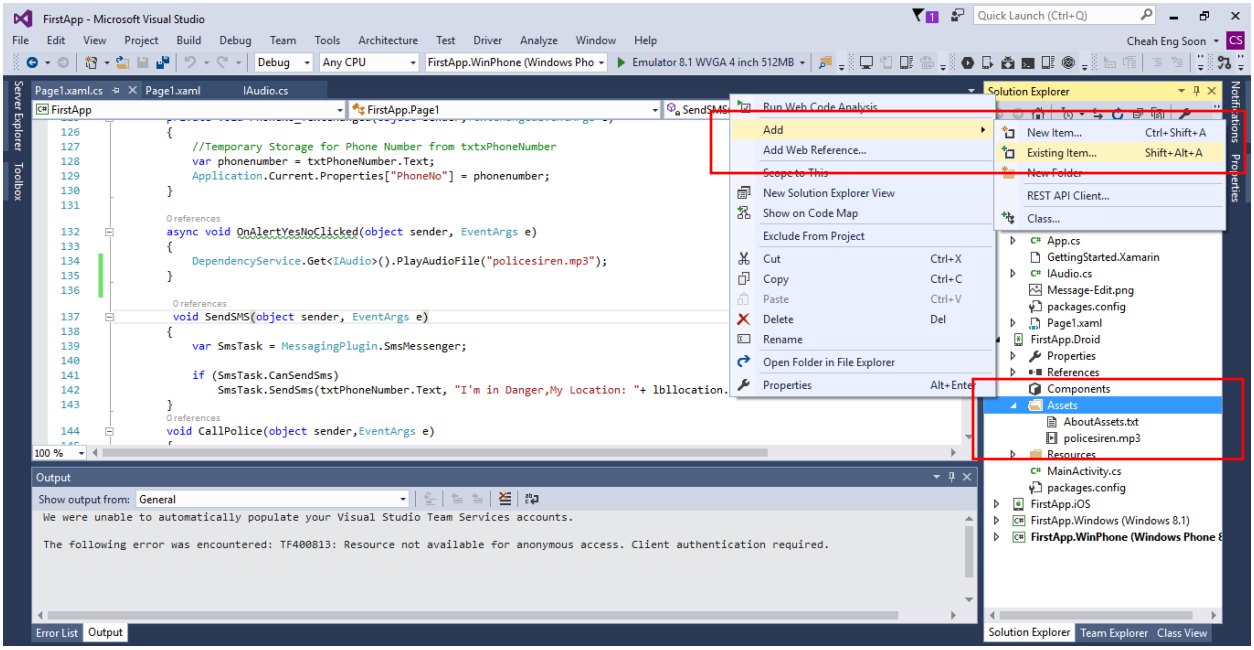
**iOS: file should be added to the Resources folder

**Android: file should be added to the Assets folder

For Android

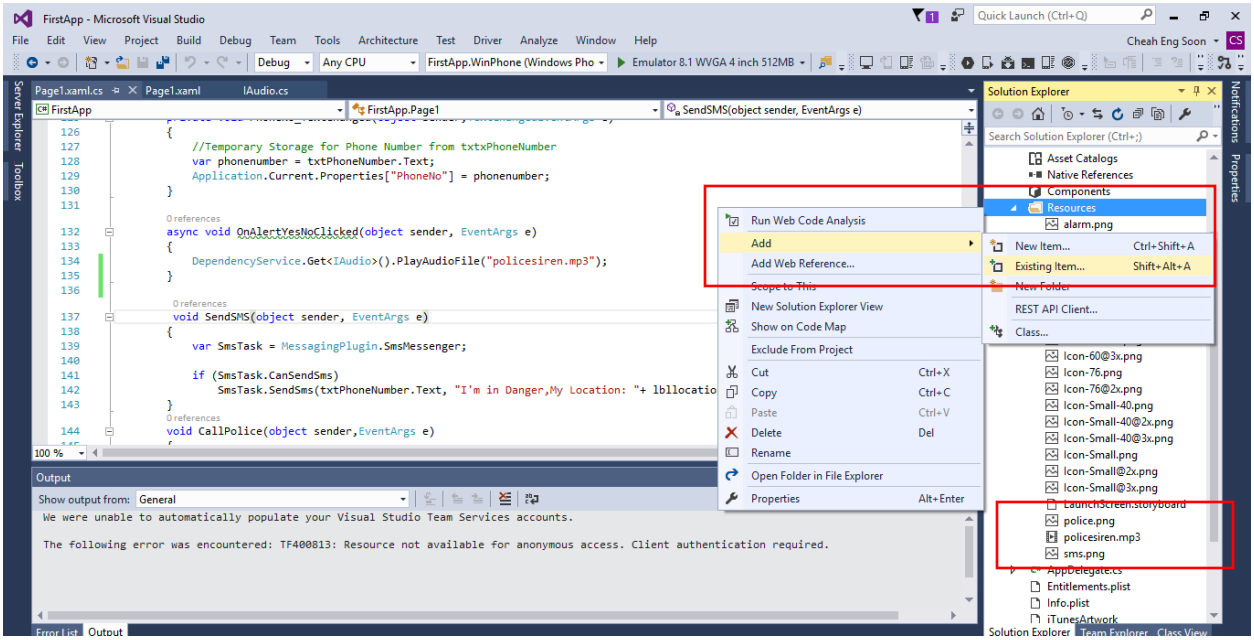
==========

Go to "**FirstApp.Droid**", Select "**Assets**" folder and select "**Add**" and "**Existing item**". Add "**policesiren.mp3**"

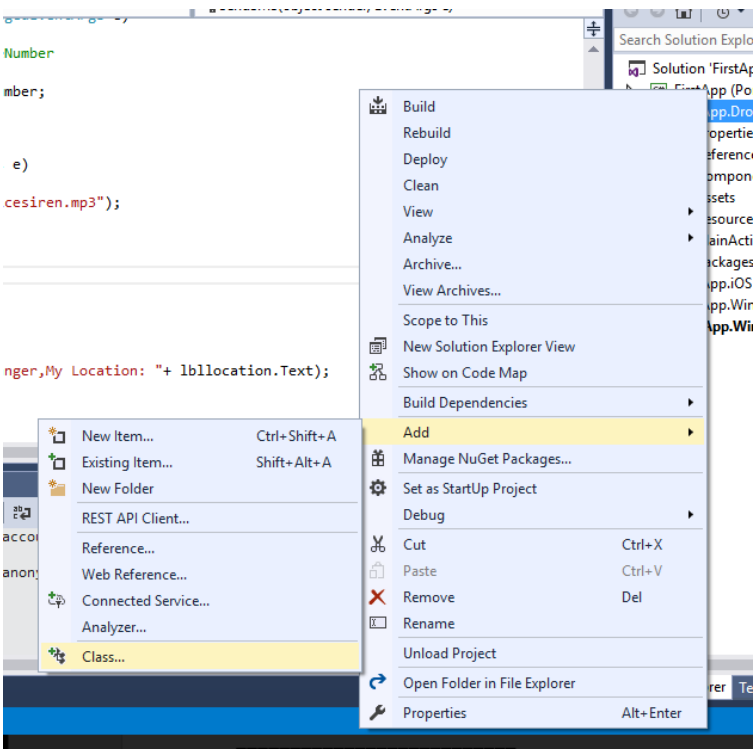


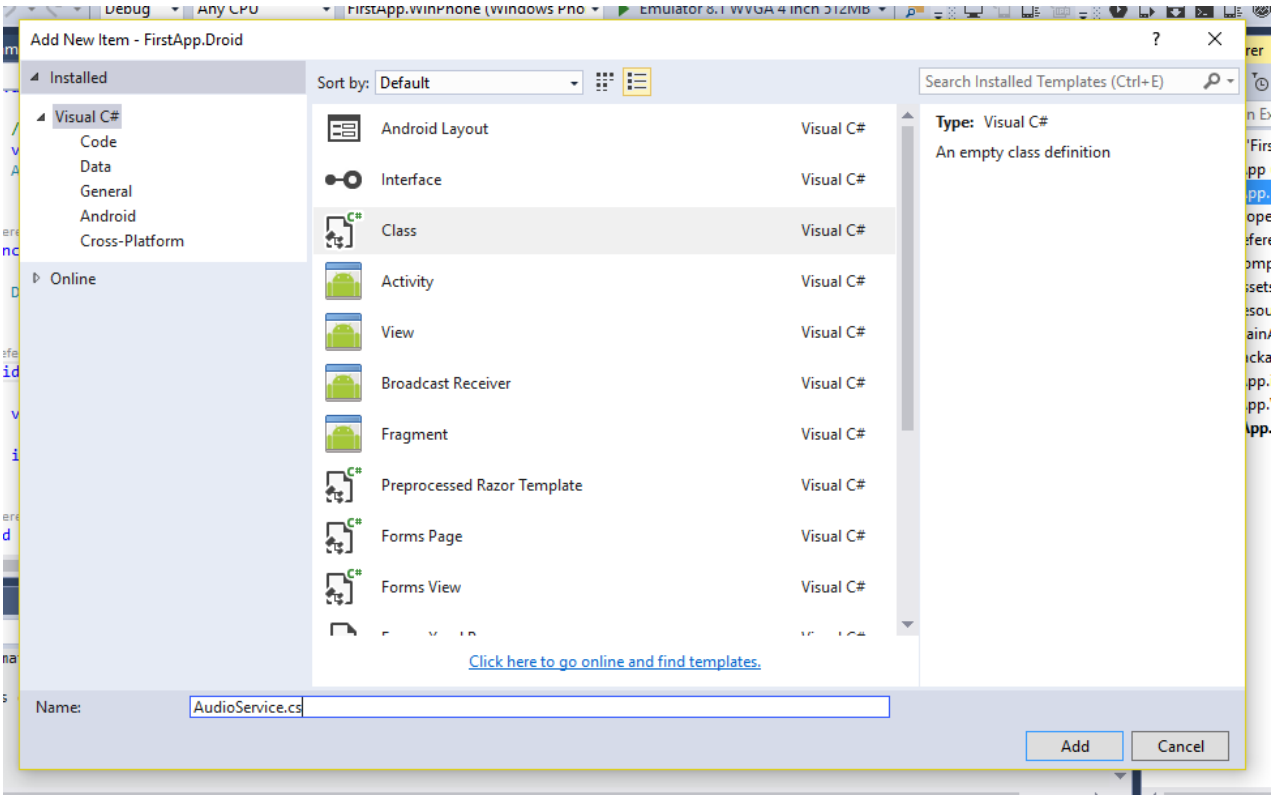
For iOS

Go to "**FirstApp.iOS**", Select "**Resources**" folder, select "**Add**" and "**Existing Item**". Add "**policesiren.mp3**"



6. Add class “**AudioService.cs**” to **FirstApp.Droid**.

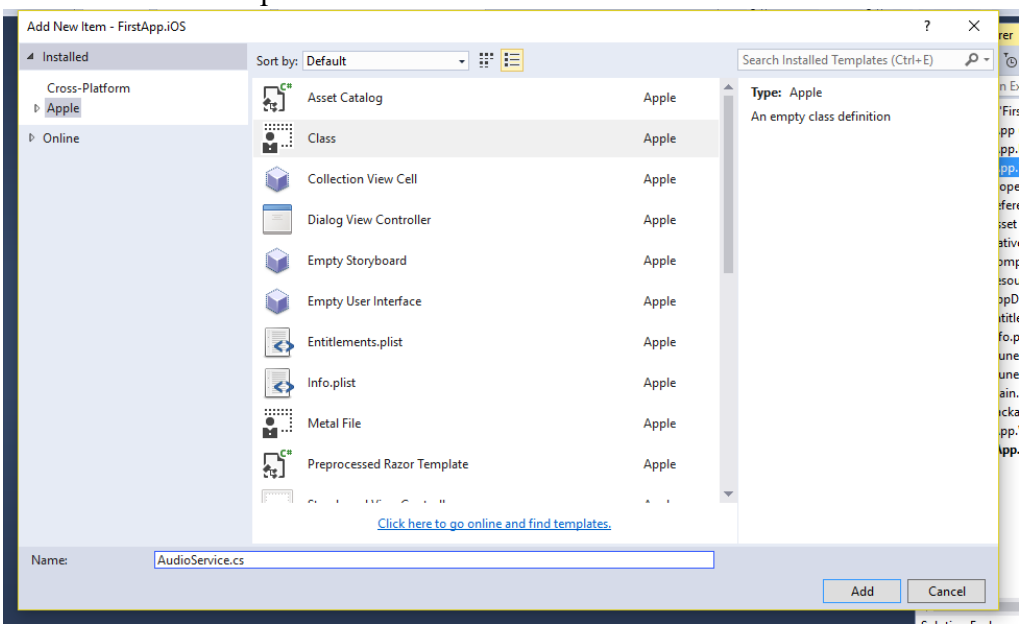




And add the Code to "**AudioServices.cs**"

```
using Android.Views;
using Android.Widget;

using Android.Media;

[assembly: Dependency(typeof(AudioService))]

namespace FirstApp.Droid
{
    class AudioService: IAudio
    {
        public AudioService()
        {
        }

        public void PlayAudioFile(string fileName)
        {
            var player = new MediaPlayer();
            var fd = global::Android.App.Application.Context.Assets.OpenFd(fileName);
            player.Prepared += (s, e) =>
            {
                player.Start();
            };
            player.SetDataSource(fd.FileDescriptor, fd.StartOffset, fd.Length);
            player.Prepare();
        }
    }
}
```

7. In iOS still same steps with Android add the "**AudioServices.cs"**



And Add the code to "**AudioServices.cs**"

```
IAudio.cs    AudioService.cs*
FirstApp.iOS    FirstApp.iOS.AudioService    AudioService
11  //using MonoTouch.Foundation;
12  //using MonoTouch.UIKit;
13
14  [assembly: Dependency(typeof(AudioService))]
15
16  namespace FirstApp.iOS
17  {
        2 references
18      public class AudioService : IAudio
19      {
            0 references
20          public AudioService()
21          {
22          }
23
            0 references
24          public void PlayAudioFile(string fileName)
25          {
26              string sFilePath = NSBundle.MainBundle.PathForResource
27              (Path.GetFileNameWithoutExtension(fileName), Path.GetExtension(fileName));
28              var url = NSUrl.FromString(sFilePath);
29              var _player = AVAudioPlayer.FromUrl(url);
30              _player.FinishedPlaying += (object sender, AVStatusEventArgs e) => {
31                  _player = null;
32              };
33              _player.Play();
34          }
35      }
36  }
```

UWP/Windows/Windows Phone 8.1

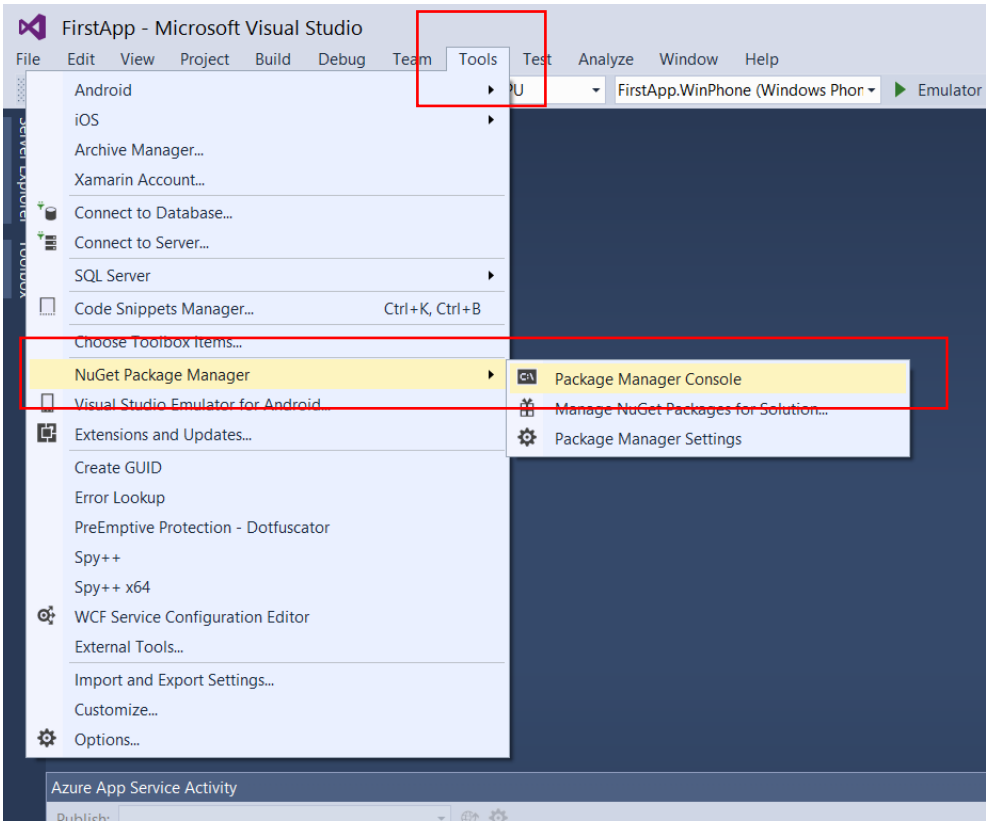
==========

Temporary unavailable

Chapter 3: Plugin

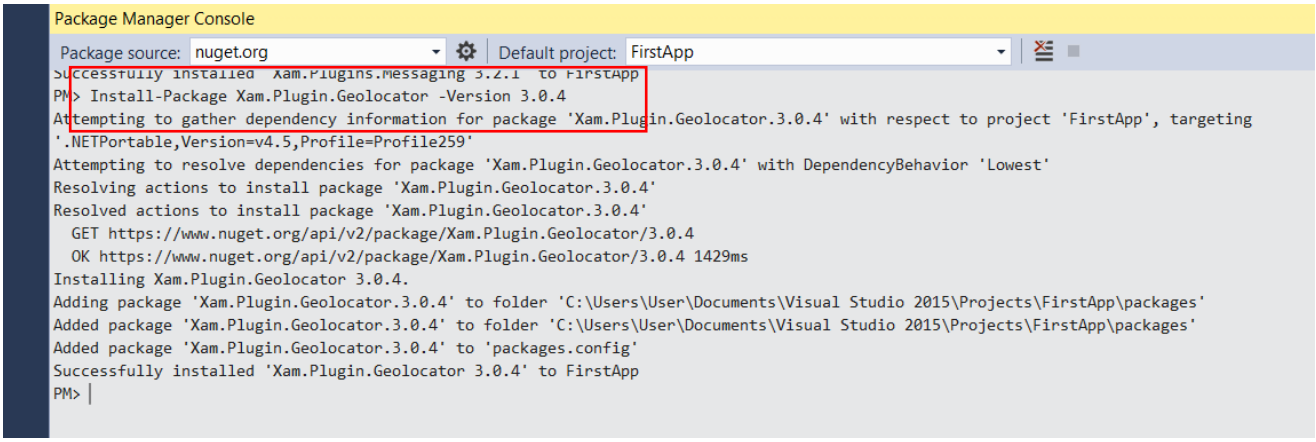
Geolocator Plugin

1. Go to Tools > Nuget Package Manager > Package Manager Console.



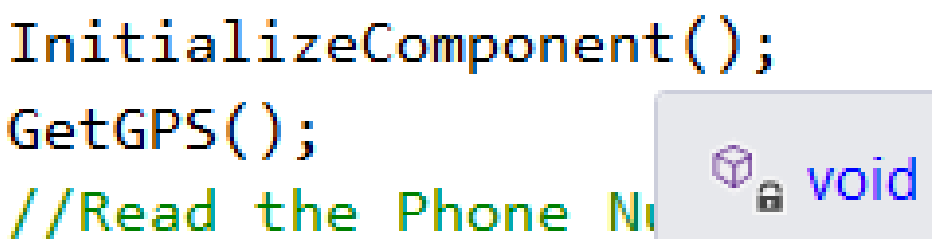
2. Key In the Command & Enter for installation of Library.

```
Install-Package Xam.Plugin.Geolocator -Version 3.0.4
```



3. Go to **HomePage.xaml.cs** add the Function for retrieve the location details

GetGPS(); at Below **InitializeComponent();**



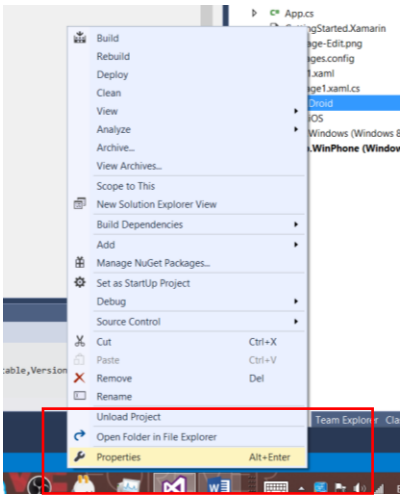
And Enter the Code Below

```
private async void GetGPS()
    {
        try
        {
            var locator = CrossGeolocator.Current;
            locator.DesiredAccuracy = 1000;
            lbllocation.Text = "Getting gps";

            var position = await locator.GetPositionAsync(timeoutMilliseconds: 10000);

            if (position == null)
            {
                lbllocation.Text = "null gps :(";
                return;
            }
            lbllocation.Text = string.Format("Time: {0} \nLat: {1} \nLong: {2}"+
              " \nAltitude: {3} \nAltitude Accuracy: {4} \nAccuracy: {5} \nHeading: {6} \nSpeed:
{7}",
                position.Timestamp, position.Latitude, position.Longitude,
                position.Altitude, position.AltitudeAccuracy, position.Accuracy, position.Heading,
position.Speed);

        }
        catch //(Exception ex)
        {
            // Xamarin.Insights.Report(ex);
            // await DisplayAlert("Uh oh", "Something went wrong, but don't worry we captured it
in Xamarin Insights! Thanks.", "OK");
        }
    }
```

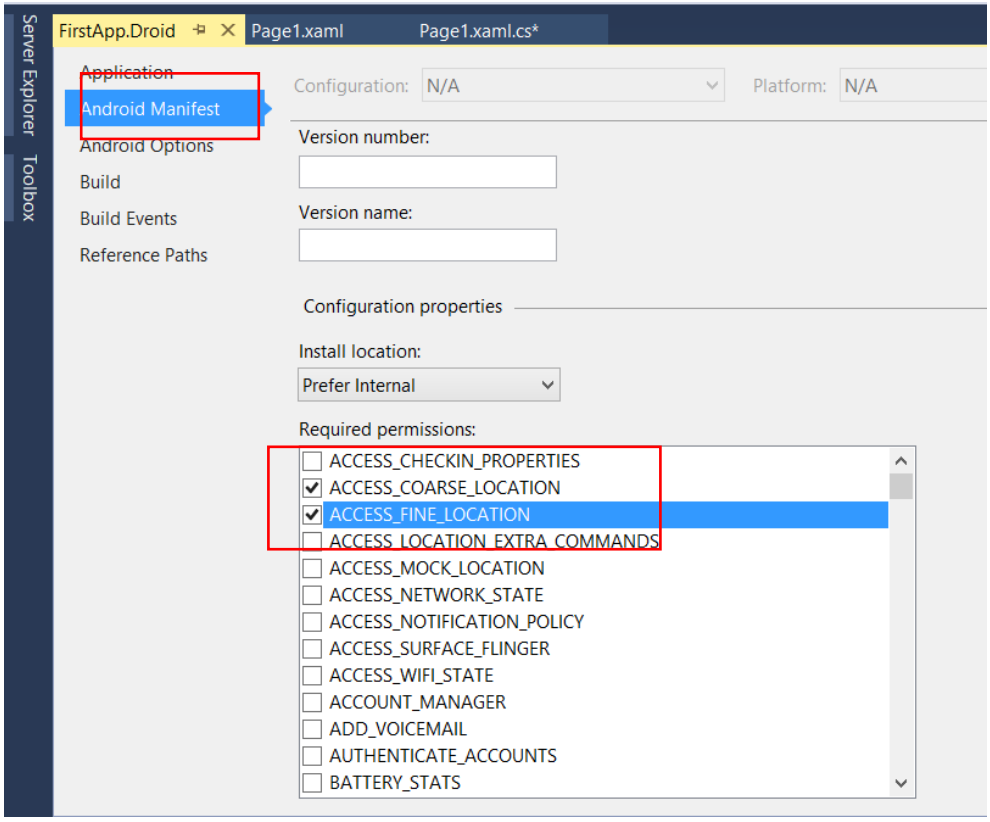
```
        protected override void OnAppearing()
        {
            base.OnAppearing();
            try
            {
                CrossGeolocator.Current.PositionChanged +=
CrossGeolocator_Current_PositionChanged;
                CrossGeolocator.Current.PositionError += CrossGeolocator_Current_PositionError;
            }
            catch
            {
            }
        }

        void CrossGeolocator_Current_PositionError(object sender,
Plugin.Geolocator.Abstractions.PositionErrorEventArgs e)
        {

            lbllocation.Text = "Location error: " + e.Error.ToString();
        }

        void CrossGeolocator_Current_PositionChanged(object sender,
Plugin.Geolocator.Abstractions.PositionEventArgs e)
        {
            var position = e.Position;
            lbllocation.Text = string.Format("Time: {0} \nLat: {1} \nLong: {2} \nAltitude: "
                +"{3} \nAltitude Accuracy: {4} \nAccuracy: {5} \nHeading: {6} \nSpeed: {7}",
                position.Timestamp, position.Latitude, position.Longitude,
                position.Altitude, position.AltitudeAccuracy, position.Accuracy, position.Heading,
position.Speed);


        }

        protected override void OnDisappearing()
        {
            base.OnDisappearing();
            try
            {
                CrossGeolocator.Current.PositionChanged -=
CrossGeolocator_Current_PositionChanged;
                CrossGeolocator.Current.PositionError -= CrossGeolocator_Current_PositionError;
            }
            catch
```

```
    {
    }
}
```

4. Go to **FirstApp.Droid** , Right Click > Select Properties

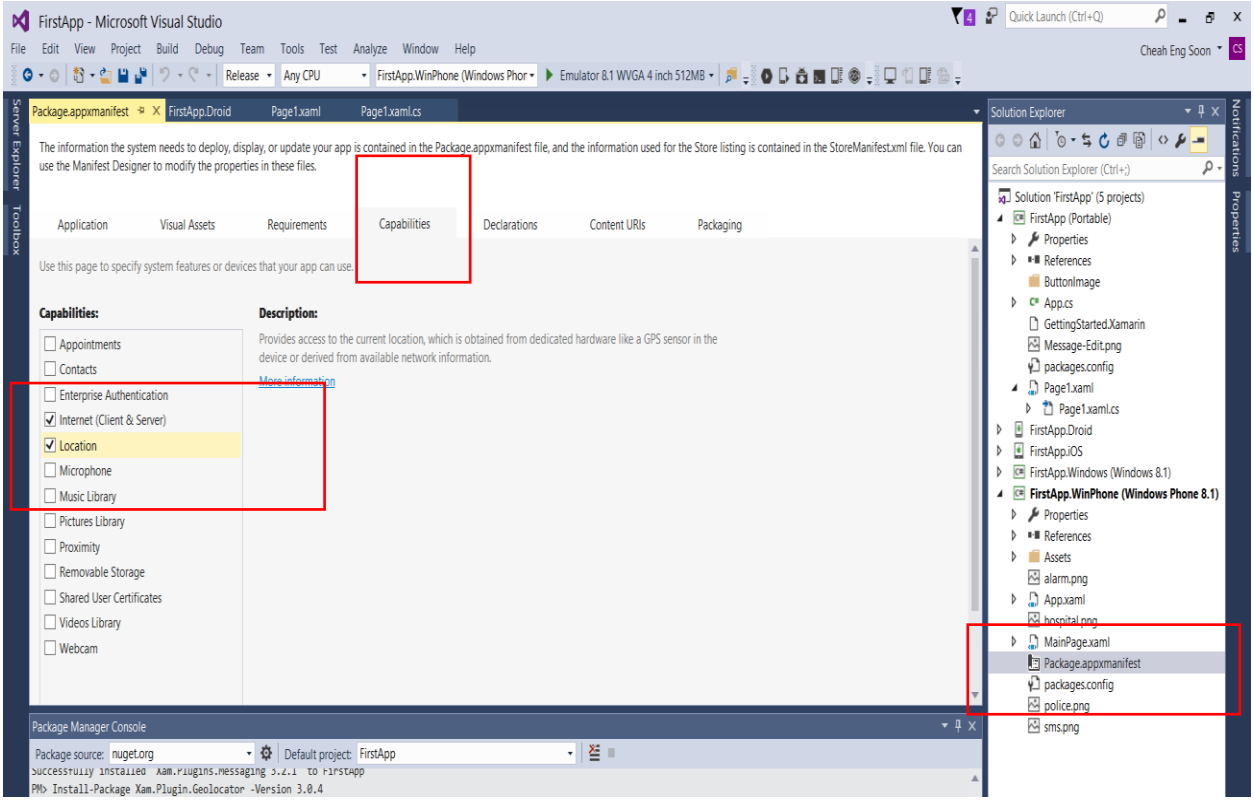


5. Go to **Android Manifest** Enable Location Permission



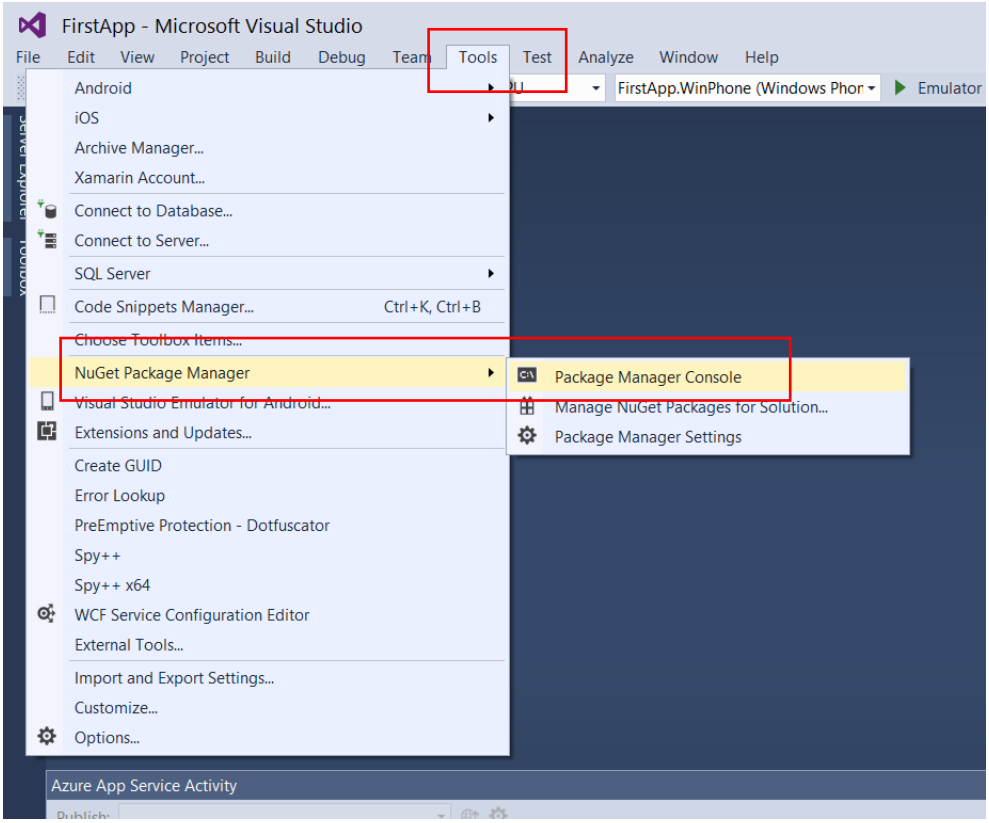
For **Windows Phone/Windows/UWP**

Go to "**Package.appxmanifest**" to Enable Location Services



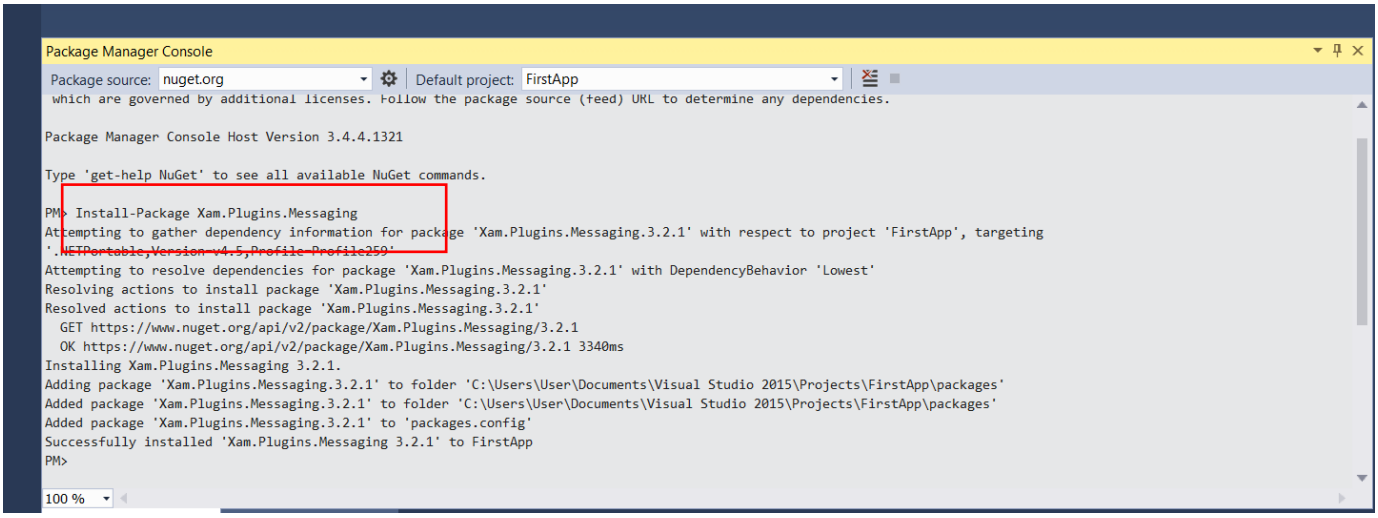
SMS & Call Plugin

1. Go to Tools > Nuget Package Manager > Package Manager Console.



2. Enter the Command & Enter for installation of Library.

```
Install-Package Xam.Plugins.Messaging
```



3. Go to **HomePage.xaml** create an event handler for Call & SMS Function

```
    <!--Controls-->
    <Label x:Name="lbllocation" Grid.Row="0" Grid.Column="0" Grid.ColumnSpan="2"  BackgroundColor="Blue" />
    <Button x:Name="btnmessage" Grid.Row="1" Grid.Column="0" Image="sms.png"  Clicked="SendSMS"/>
    <Button x:Name="btnalarm" Grid.Row="1" Grid.Column="1" Text="Alarm" Image="alarm.png" Clicked="PlayAlarmOnClicked"/>
    <Button x:Name="btnPolice" Grid.Row="2" Grid.Column="0" Image="police.png" Clicked="CallPolice"/>
    <Button x:Name="btnAmbulance" Grid.Row="2" Grid.Column="1" Image="hospital.png" Clicked="CallHospital"/>

    <Entry x:Name="txtPhoneNumber" Grid.Row="3" Grid.Column="0" Grid.ColumnSpan="2"
           Placeholder="Phone Number" TextChanged="PhoneNo_TextChanged" Keyboard="Numeric" /> <!--aka Textbox-->
  </Grid>
    </ContentPage.Content>
</ContentPage>
```

4. Go to **HomePage.xaml.cs** add the Backend Code.

```
    void SendSMS(object sender, EventArgs e)
    {
        var SmsTask = MessagingPlugin.SmsMessenger;

        if (SmsTask.CanSendSms)
            SmsTask.SendSms(txtPhoneNumber.Text, "I'm in Danger,My Location: "+ lbllocation.Text);
    }
    void CallPolice(object sender,EventArgs e)
    {
        //Don't forgot to enable ID_CAP_PHONEDAILER on manifest file
        var PhoneCallTask = MessagingPlugin.PhoneDialer;
        if (PhoneCallTask.CanMakePhoneCall)
            PhoneCallTask.MakePhoneCall("999");
    }
    void CallHospital(object sender, EventArgs e)
    {
        //Don't forgot to enable ID_CAP_PHONEDAILER on manifest file
        var PhoneCallTask = MessagingPlugin.PhoneDialer;
        if (PhoneCallTask.CanMakePhoneCall)
            PhoneCallTask.MakePhoneCall("999");
    }
```

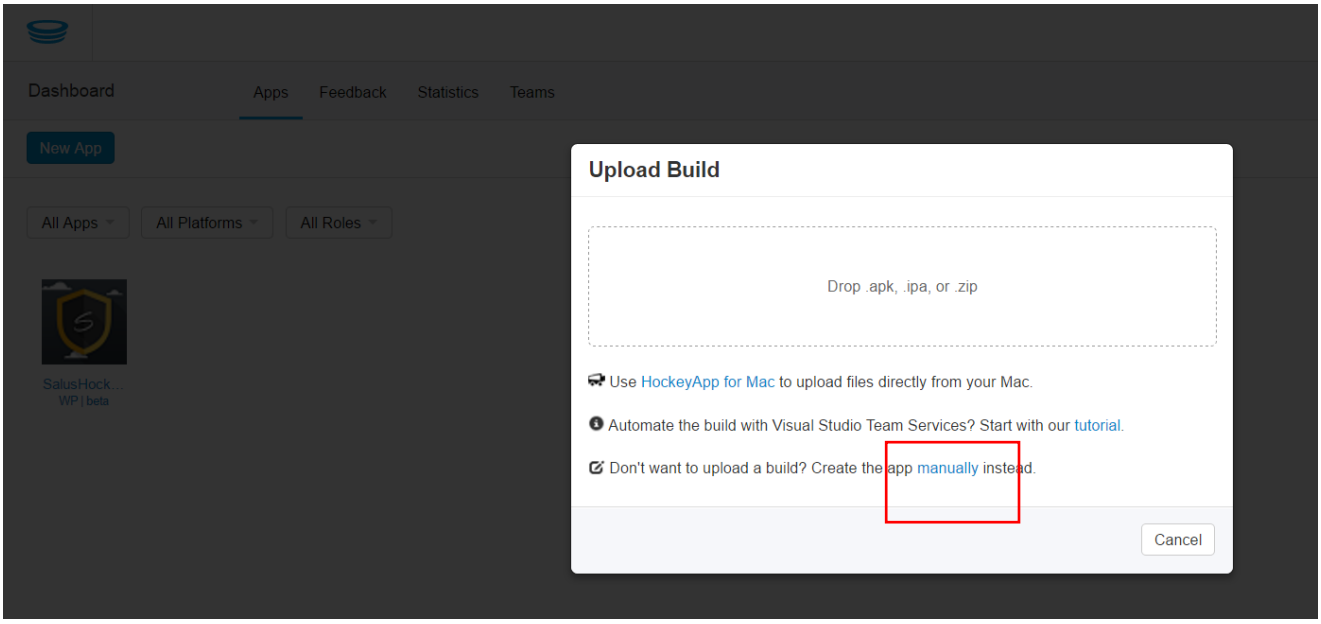
5. Enable the **CALL_PHONE** and **SEND_SMS** in Android Manifest . For WindowsPhone/UWP enable **ID_CAP_PHONEDAILER**

CHAPTER 4: Integrate with HockeyApp

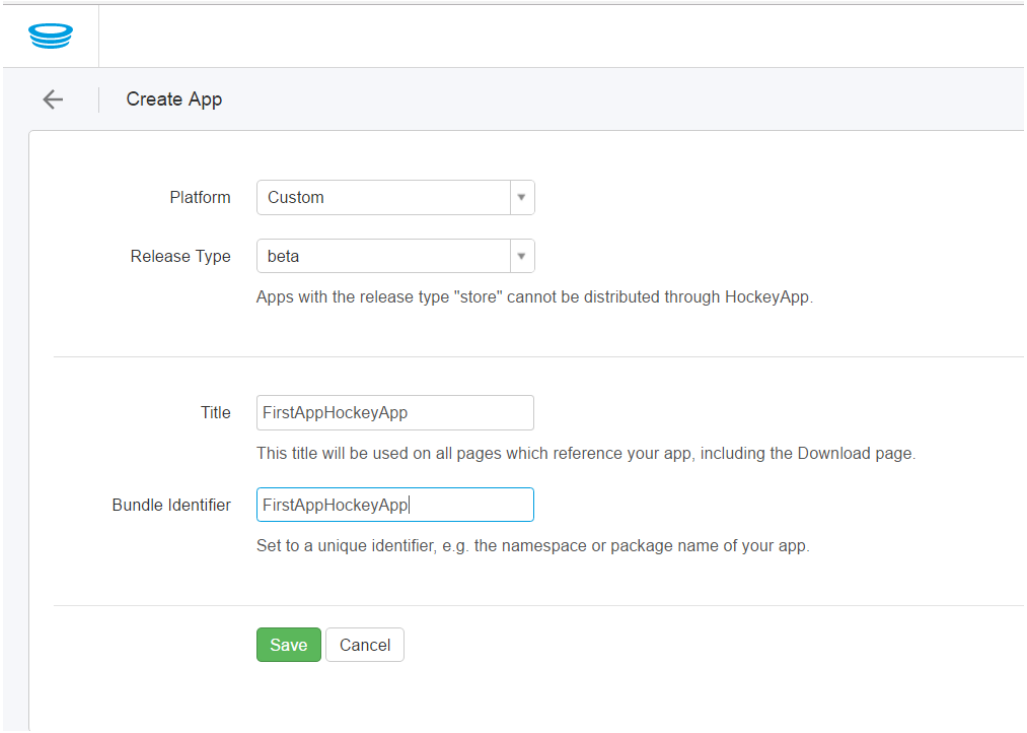
1. Go to https://**hockeyapp**.net/

2. Sign Up with your Microsoft Account

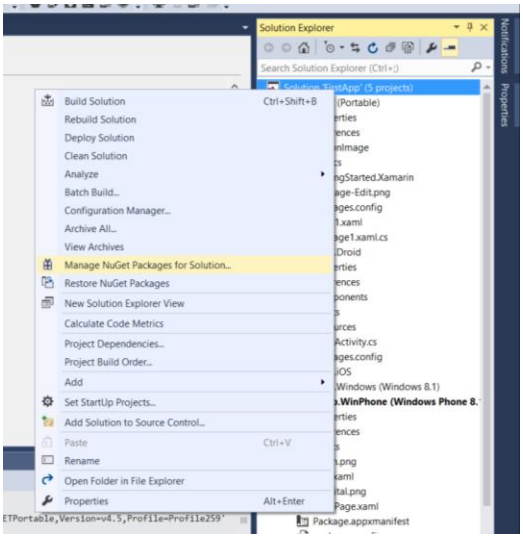
3. Create a New App and Select “manually”

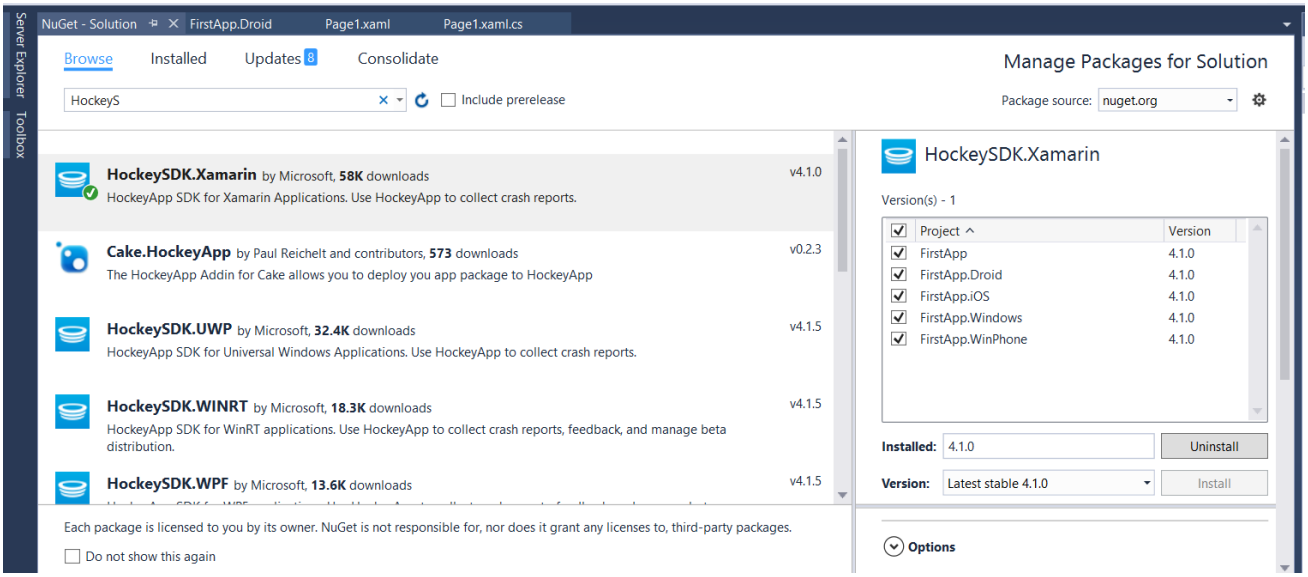


4. Fill in the Details & Save.



5. Go to Project > Manage NuGet Packages …. And Search HockeySDK.Xamarin



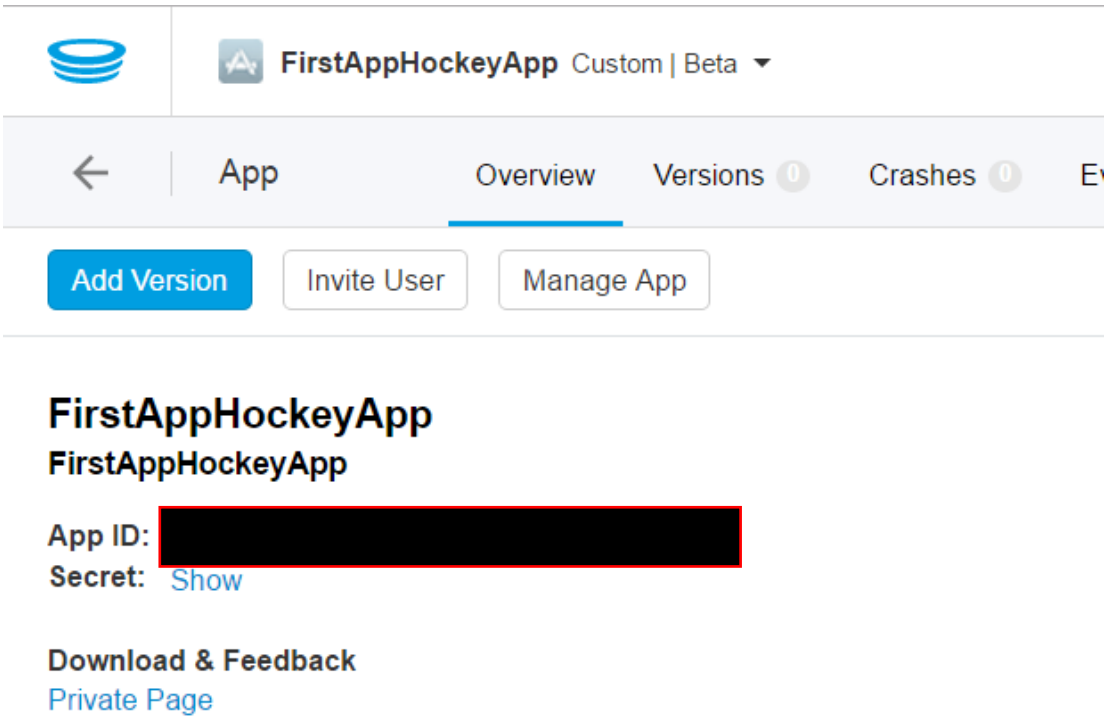


For Android

Open **MainActivity.cs**

```
using System;

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

namespace FirstApp.Droid
{
    [Activity(Label = "FirstApp", Icon = "@drawable/icon", Theme = "@style/MainTheme", MainLauncher = true, ConfigurationChanges = ConfigChanges.Scr
    public class MainActivity : global::Xamarin.Forms.Platform.Android.FormsAppCompatActivity
    {
        protected override void OnCreate(Bundle bundle)
        {
            TabLayoutResource = Resource.Layout.Tabbar;
            ToolbarResource = Resource.Layout.Toolbar;

            base.OnCreate(bundle);

            global::Xamarin.Forms.Forms.Init(this, bundle);
            LoadApplication(new App());

            CrashManager.Register(this, "$Your_App_Id");
        }
    }
```

Replace **"$Your_App_Id"** That you can get in Hockey App.



For iOS

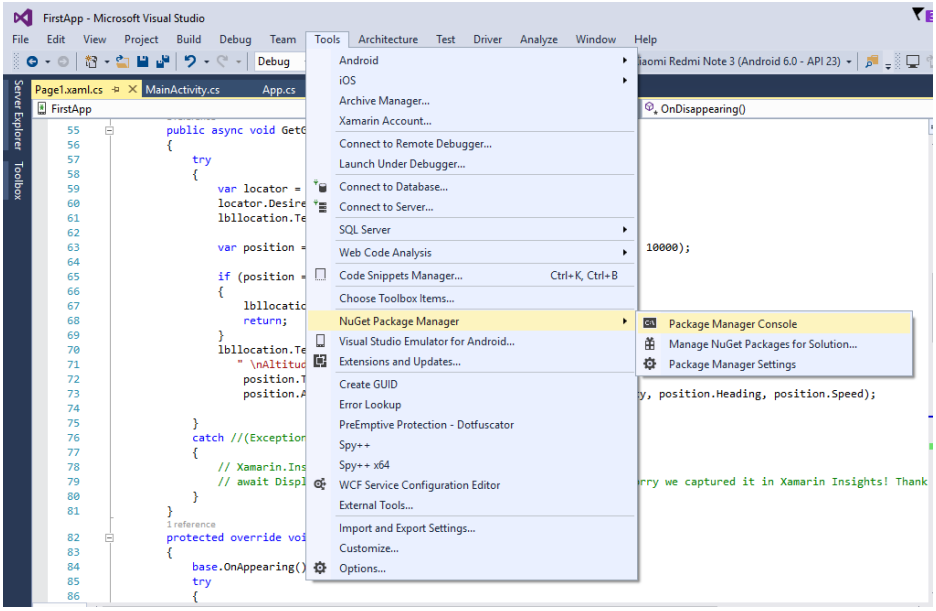
1. Open **AppDelegate.cs** and add this new line of code.

```
// The UIApplicationDelegate for the application. This class is responsible for launching the
// User Interface of the application, as well as listening (and optionally responding) to
// application events from iOS.
[Register("AppDelegate")]
public partial class AppDelegate : global::Xamarin.Forms.Platform.iOS.FormsApplicationDelegate
{
    //
    // This method is invoked when the application has loaded and is ready to run. In this
    // method you should instantiate the window, load the UI into it and then make the window
    // visible.
    //
    // You have 17 seconds to return from this method, or iOS will terminate your application.
    //
    public override bool FinishedLaunching(UIApplication app, NSDictionary options)
    {
        global::Xamarin.Forms.Forms.Init();
        LoadApplication(new App());

        return base.FinishedLaunching(app, options);

        var manager = BITHockeyManager.SharedHockeyManager;
        manager.Configure("$Your_App_Id");
        manager.StartManager();
    }
}
}
```

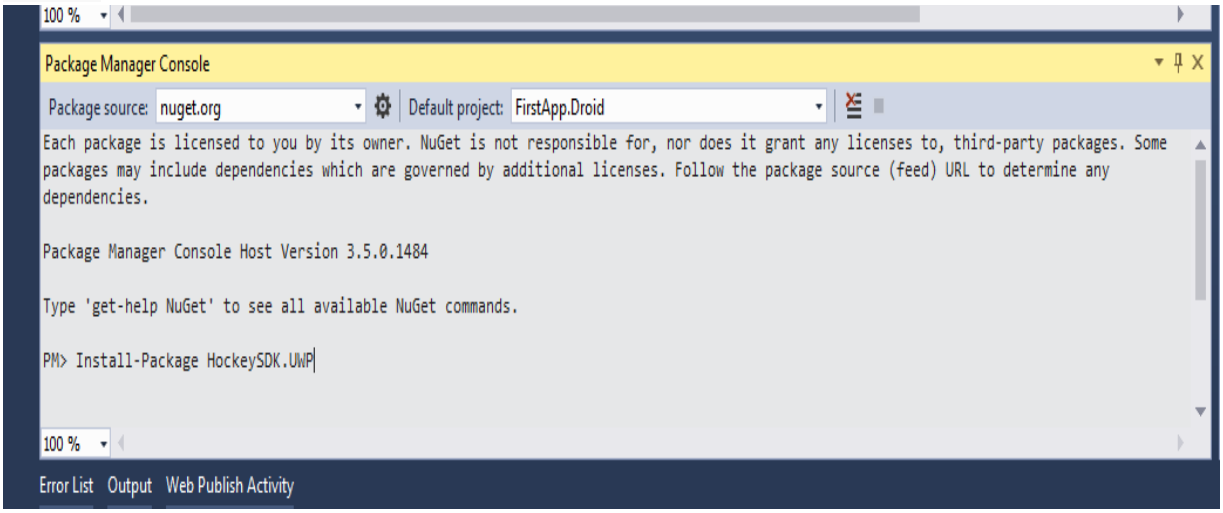
For UWP

https://www.hockeyapp.net/blog/2016/10/18/the-latest-on-hockeysdk-for-uwp.html

1. Go to Tools > Nuget Package Manager > Package Manager Console



2. Enter the Command "**Install-Package HockeySDK.UWP**" in the PowerShell Host. And Press "**Enter**".



3. In the ***App.xaml.cs*** > ***App*** *class constructor*, add the following line using the App ID from the overview page of your app on the HockeyApp web app:

```
Microsoft.HockeyApp.HockeyClient.Current.Configure("Your-App-ID");
```

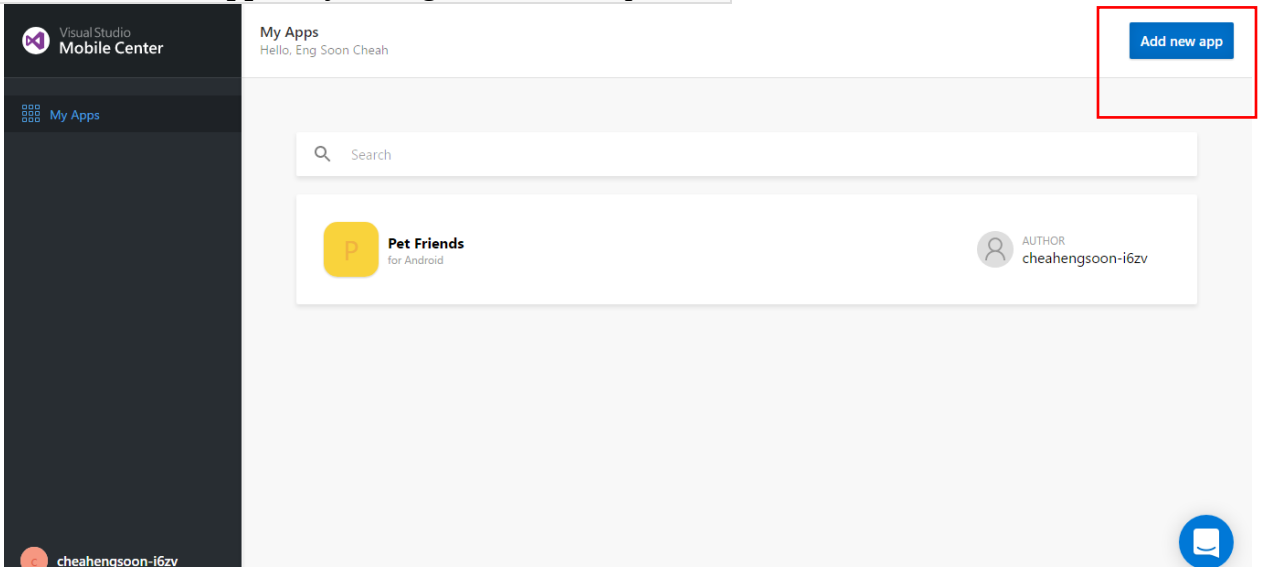
4. Make sure **Internet(Client)** capability is enabled in the package manifest

Chapter 5 : Integrate your App with Visual Studio Mobile Center

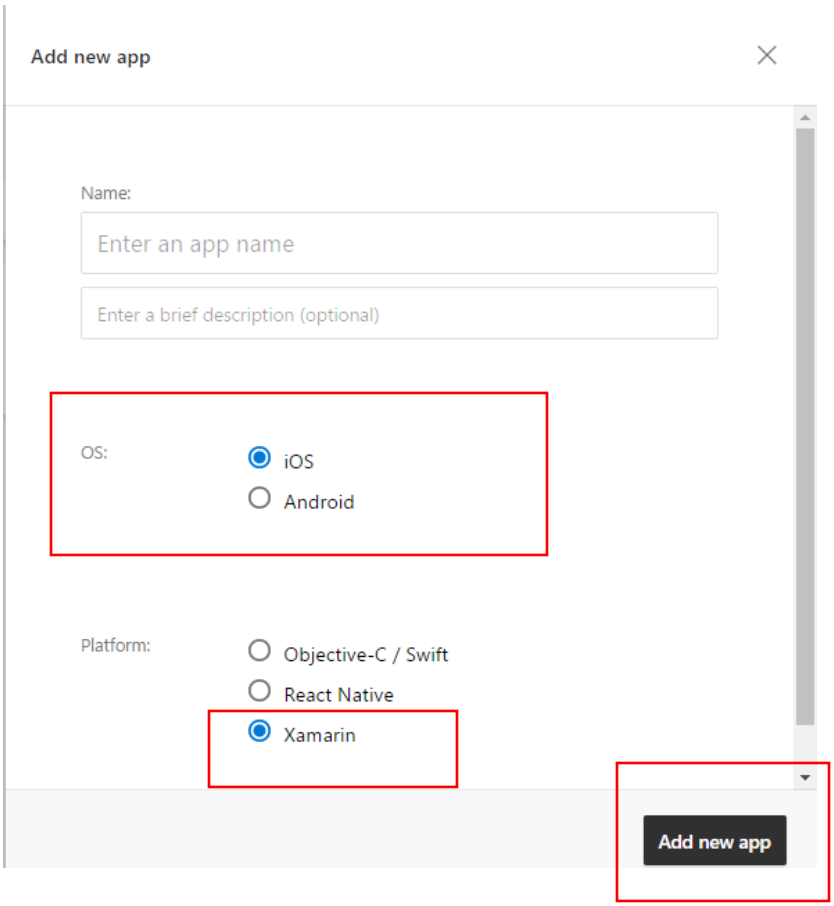
Go to Visual Studio Mobile Center : https://mobile.azure.com/login

Login with your **Microsoft Account** or **GitHub Account**

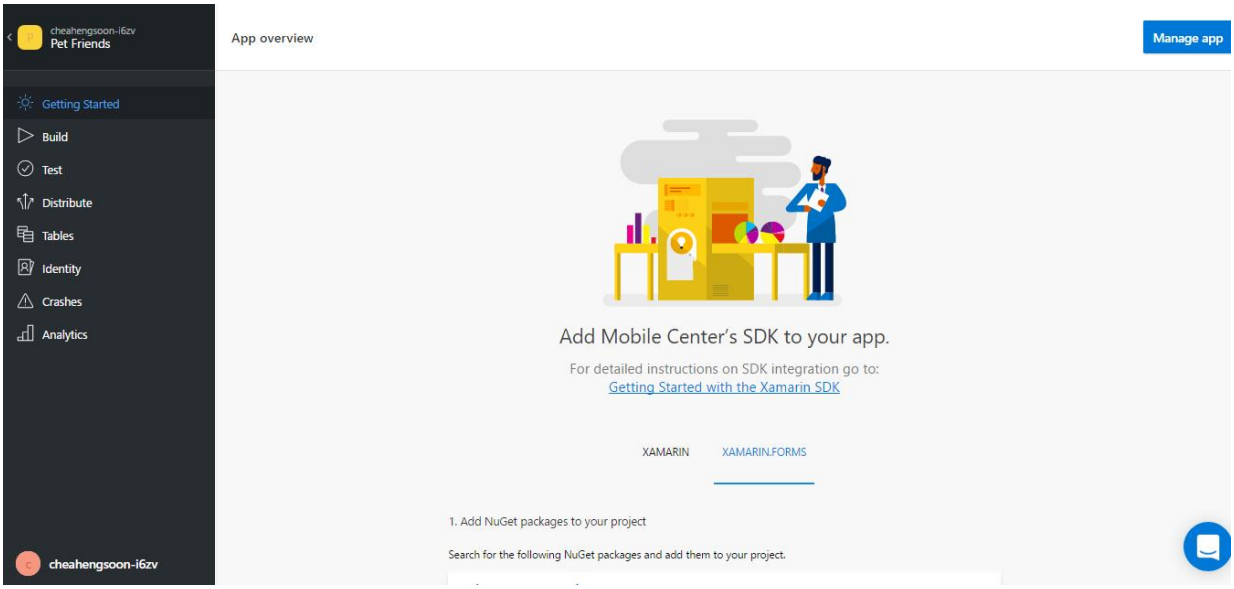
Click "**Add a new App**" on your Right hand side top corner



Enter your App information and select your OS : iOS or Android. And the platforms choose "Xamarin" and Click "Add new app"



Follow the instruction of Visual Studio Mobile Center instructions for Android and iOS.



Setup

Mobile Center SDK is designed with a modular approach – a developer only needs to integrate the modules of the services that they're interested in. If you'd like to get started with just Analytics or Crashes, include their packages in your app. For each iOS, Android and Forms project, add the `Mobile Center Analytics` and `Mobile Center Crashes` packages.

For Xamarin.Forms

Multiplatform Xamarin.Forms app has three projects in your solution - portable class library or shared library, project.Droid, project.iOS . You need to add NuGet packages to each of these projects.

- Navigate to the Project -> 'Add NuGet Packages…'
- Search and select "Mobile Center Analytics" and "Mobile Center Crashes". Then Click 'Add Packages'

For Xamarin for Visual Studio

- Navigate Project -> Manage NuGet Packages…
- Search and select "Mobile Center Analytics" and "Mobile Center Crashes". Then Click 'Add Packages'

Now that you've integrated the SDK in your application, it's time to start the SDK and make use of Mobile Center services.

Start the SDK

To start the Mobile Center SDK in your app, follow these steps:

1. **Add using statements:** Add the appropriate namespaces befor eyou get started with using our APIs.

   Xamarin.iOS - Open AppDelegate.cs file and add the lines below the existing using statements

   Xamarin.Android - Open MainActivity.cs file and add the lines below the existing using statements

   Xamarin.Forms - Open App.xaml.cs file in your shared project and add these using statements

```
using Microsoft.Azure.Mobile;

using Microsoft.Azure.Mobile.Analytics;

using Microsoft.Azure.Mobile.Crashes;
```

2. **Start the SDK:** Mobile Center provides developers with two modules to get started – Analytics and Crashes. In order to use these modules, you need to opt in for the module(s) that you'd like, meaning by default no module is started and you will have to explicitly call each of them when starting the SDK.

   Xamarin.iOS

   Open AppDelegate.cs file and add the Start API in FinishedLaunching() method

```
MobileCenter.Start("{Your App Secret}", typeof(Analytics), typeof(Crashes));
```

   Xamarin.Android

   Open MainActivity.cs file and add the Start API in OnCreate() method

```
MobileCenter.Start("{Your App Secret}", typeof(Analytics), typeof(Crashes));
```

   Xamarin.Forms

   Start SDK call is split into two methods for Xamarin.Forms. That's because you need two different AppSecrets - one for iOS and other for your Android app. Open **App.xaml.cs** file in your shared project and add the API below in the **App()** constructor.

```
MobileCenter.Start(typeof(Analytics), typeof(Crashes));
```

In the **iOS** project of the Forms app, open **AppDelegate.cs** and add the API in FinishedLaunching() method

```
MobileCenter.Configure("{Your iOS App Secret}");
```

In the Droid project of the Forms app, open **MainActivity.cs** and add the API in **OnCreate()** method

```
MobileCenter.Configure("{Your Android App Secret}");
```

References: https://docs.mobile.azure.com/sdk/Xamarin/getting-started/

References:

Microsoft Virtual Academy

https://mva.microsoft.com/en-US/training-courses/xamarin-for-absolute-beginners-16182

Xamarin Malaysia Developers

https://www.facebook.com/groups/xamarinmydev/

HockeyApp

https://hockeyapp.net/

Source Code

https://github.com/cheahengsoon/FirstXamarinFormsApp

Visual Studio Mobile Center

https://mobile.azure.com/