Xamarin Fundamentals:

Module 3, Lesson 9

Build a Basic Android app using Xamarin.Forms Lab

Overview

Xamarin is a powerful tool that allows you to develop mobile applications across all major mobile platforms. This lab will cover how to build a very basic Android, iOS, and UWP application using Xamarin.Forms.

Note on the UWP project: A Xamarin.Forms solution created with the Windows 10 SDK installed on the machine should automatically include a UWP project. If that was not the case and a UWP project is missing, add one using Lab 3a.

Objectives

In this hands-on lab you will learn how to:

* Interact with Xamarin.Forms, specifically in the context of building a simple Android app
* Create a text input field
* Display a dialog box on the click of a button

Prerequisites

The following are required to complete this hands-on lab:

* Visual Studio with Xamarin installed or Xamarin Studio for OS X
* Completion of [Module 3 Lessons 1-10](https://github.com/MSFTImagine/computerscience/tree/master/Complimentary%20Course%20Content/Module3/Lessons) and [Module 3 Lesson 5 and 6 Labs](https://github.com/MSFTImagine/computerscience/tree/master/Complimentary%20Course%20Content/Module3/Labs)

Exercises

This hands-on lab includes the following exercises:

* Exercise 1: Create a First Name Entry UI

Exercise 1: Create a First Name Entry UI

Create a First Name Entry UI that displays the name in a popup when the button is clicked.

1. Create a Visual Studio Visual C#> Cross-platform > Blank App (Xamarin.Forms Portable) Project.
2. Right-click the Android, iOS, or UWP project, and then click Set As Startup Project. This step determines the platform that will run.
3. On the Build menu, click Configuration Manager. In the Configuration Manager dialog box, select the Build and Deploy checkboxes of the Android, iOS, or UWP project (same as your startup project platform).
4. Create a custom page in the Xamarin.Forms core project and set it to be the app’s main page. Create a class inherited from ContentPage and call it NamePage.

using Xamarin.Forms;

namespace Mobile3\_Lab3\_Forms

{

class NamePage : ContentPage

{

public NamePage()

{

// views/controls will go here

}

}

}

1. Now instantiate the new page. In Xamarin.Forms project’s App.cs, update the App constructor to set an instance of NamePage as the MainPage:

namespace Mobile3\_Lab3\_Forms

{

public class App : Application

{

public App()

{

MainPage = new NamePage();

}

1. Now add some views to NamePage.cs using C# (you could also use XAML but this is a C# example).
   * Add a Label with a with a text value of “First Name”.

Label labelFirst = new Label

{

Text = "First Name",

FontSize = Device.GetNamedSize(NamedSize.Large, typeof(Label)),

HorizontalOptions = LayoutOptions.FillAndExpand

};

* + Add an Entry control called firstName with a Placeholder of “enter name”.

Entry firstName = new Entry

{

Placeholder = "enter name",

FontSize = Device.GetNamedSize(NamedSize.Medium, typeof(Entry)),

HorizontalOptions = LayoutOptions.FillAndExpand,

Keyboard = Keyboard.Text

};

* + Add a Button called submitName with Text of “Submit”.

Button submitName = new Button

{

Text = "Submit",

FontSize = 25,

HorizontalOptions = LayoutOptions.FillAndExpand

};

1. Place the views on a layout. A Layout view acts as a container for other views. Since a ContentPage can have only one child, all the views on our page must be placed in a single container that is made the child of the ContentPage. Use a StackLayout, a subclass of Layout that can “stack” child views vertically:

StackLayout stackLayout = new StackLayout

{

Children =

{

labelFirst,

firstName,

submitName

},

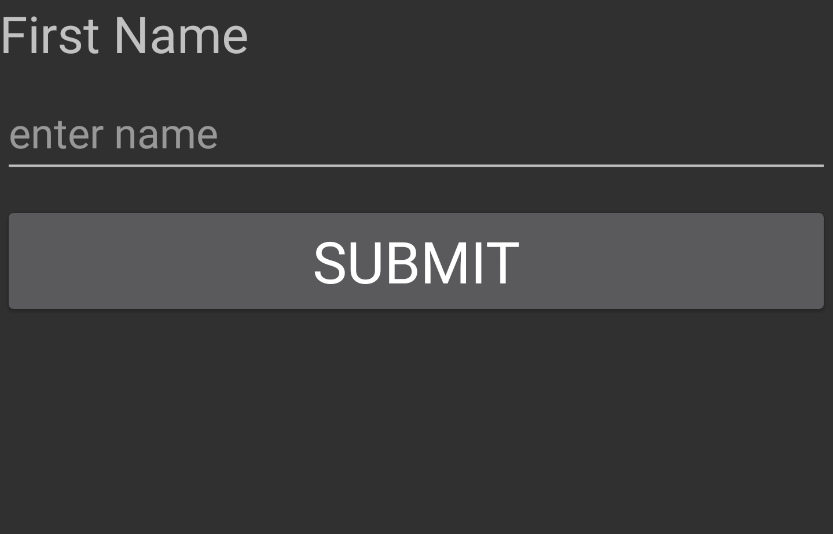
HeightRequest = 1500

};

1. To get the StackLayout to display on our page, we must assign it to the Content property of the ContentPage:

this.Content = stackLayout;

The views look like this on the page and tapping “enter name” will invoke a keyboard.



1. In the NamePage, make the button click event show the firstName value in a popup.

Handle the button’s Click event.

submitName.Click += delegate { … }

OR

submitName.Click += (object sender, EventArgs e) => {…}

In the button’s event, use DisplayAlert to create a popup displaying the name in the firstName view. Using async/await with popups is good practice to help prevent the UI from locking up.

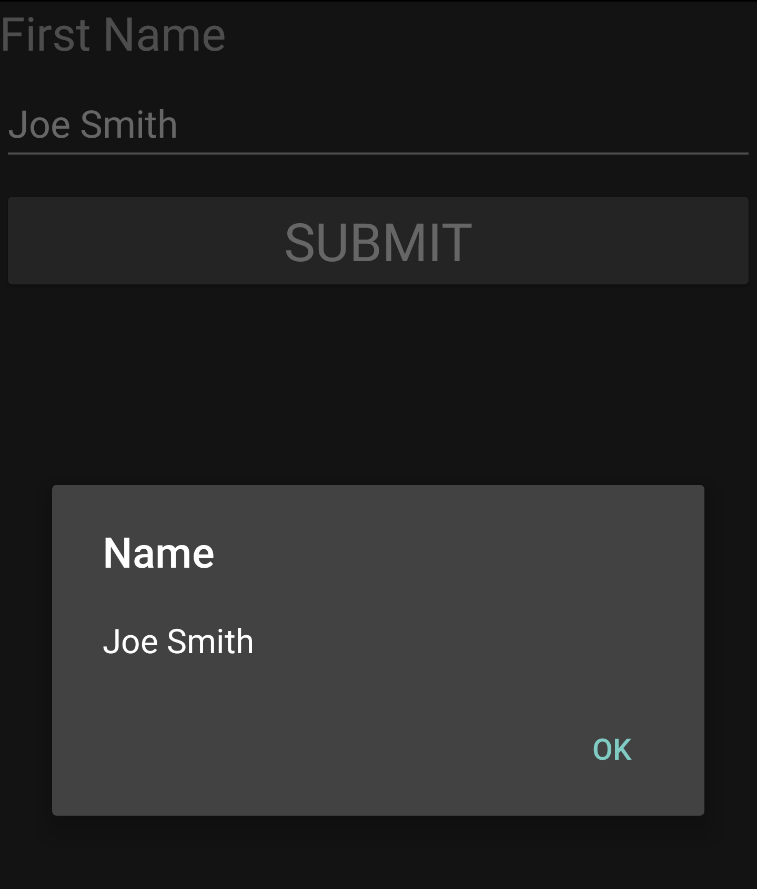
submitName.Clicked += async (sender, e) =>

{

await DisplayAlert("Name", firstName.Text, "OK");

};

Entering a name and tapping the button raises the alert.



Build and deploy to Android.

Summary

In this hands-on lab, you learned how to:

* Create a Xamarin.Forms project
* Implement a First Name Entry UI using Xamarin.Forms