This post was published to Code Blog at 2:05:00 PM 10/24/2016

Xamarin Forms - Tips

Category Xamarin

​

## Intro

xamarin works really well to create cross platform native apps, especially iOS, Android and UWP (Surface Tablet/Win 10) apps. Here are some tips I learned along the way.

* Use PCL's for xamarin forms projects. install this vsix to avoid errors;  <https://visualstudiogallery.msdn.microsoft.com/e1d736b0-5531-4eee-a27a-30a0318cac45>
* File > new project > Cross plarform > Xamarin.Forms (UWP/Android/iOS)

## Android

* Start your emulator each morning, it tends to be slow. Sometimes if you debug without the emulator started already it errors out.

## iOS

* If you use the Mac Debug bridge make sure visual studio on windows and xamarin studio on mac are on the same channel (stable). Make sure xcode is installed and the sdk locations are valid

## User Interface

**Scalable Vector Icons using Font Awesome**

<http://www.johankarlsson.net/2015/10/fontawesome-using-xamarin-forms-and-ios.html>



Floating Action Button

[FAB.Forms nuget package](https://www.nuget.org/packages/FAB.Forms) - Simplest FAB replacement for Android & iOS.

* + Need to make sure all Android packages are version 23.3.0 (it will yell at you if you forget).
  + Include this line in your xml header on any page you want to use a FAB:  
     xmlns:fab="clr-namespace:FAB.Forms;assembly=FAB.Forms"
  + And then to use it, wrap the rest of the page in a RelativeLayout, and include this anywhere:

<fab:FloatingActionButton

          x:Name="filterButton"

          Source="filter.png"

          NormalColor="Gray"

          Clicked="OnClicked"          RelativeLayout.XConstraint="{ConstraintExpression Type=RelativeToParent, Property=Width, Factor=1, Constant=-75}"

RelativeLayout.YConstraint="{ConstraintExpression Type=RelativeToParent, Property=Height, Factor=1, Constant=-75}" />

* + Android specifics:
    - Need to use appcompat theme. Create styles.xml:

<?xml version="1.0" encoding="utf-8"?>

<resources>

  <style name="AppTheme" parent="AppTheme.Base"/>

  <style name="AppTheme.Base" **parent="Theme.AppCompat">**

    <item name="windowActionBar">false</item>

    <item name="windowNoTitle">true</item>

  </style>

</resources>

* + - Then use it in MainActivity.cs:

[Activity (Label = "MyApp", Icon = "@drawable/icon", MainLauncher = true, **Theme = "@style/AppTheme",** ConfigurationChanges = ConfigChanges.ScreenSize | ConfigChanges.Orientation)]

* + - Also include this line in OnCreate after forms init:  
      FAB.Droid.FloatingActionButtonRenderer.InitControl();
  + iOS specifics:
    - In AppDelegate.cs, include the following line after forms init:

FAB.iOS.FloatingActionButtonRenderer.InitControl();

* That's it. ToDo is to find a package that works on windows as well.

## Creating Custom Controls

<https://blog.xamarin.com/extending-xamarin-forms-with-control-plugins/>

## Errors / Troubleshooting

Xamarin / Visual Studio / Xcode all need to be on the same version. It's very picky and sometimes fails if you are not in sync. For example, the new mac with the LED touch pad came out and Apple released v 8.1 of xcode, a day later Visual Studio and Xamarin Studio had updates to install. You must install updates on all the computers, update xcode on the mac to the latest, update Xamarin for Visual Studio, Update the iOS Simulator in VS by installing the lastedst bits from here: https://developer.xamarin.com/guides/cross-platform/windows/ios-simulator/ and lastly update Xamarin Studio on the mac to the lastest stable channel.

-If Breakpoints don't work, right click on the iOS Project -> Properties -> iOS Build  
For Linker Behavior, select Don't Link

[To be continued.... this is a running list]