# Xamarin Teaching Notes

## Creating a Xamarin project

* In VS select – New Project – Visual C# - Cross Platform – Cross Platform App (Xamarin)
* Use a blank app and shared project

## First Solution and Projects Layout

There are three projects in the solution:

1. Shared Project: Contains a file for laying out the interface (for now) called MainPage.xaml
   1. This is a content page, which we can also define in code
2. Android Project: Activity that contains the app from the shared project
3. iOS Project: AppDelegate class that launches the app from shared project

Start a project and fire it up. Be prepared to for this taking a while due to general VS issues. Android should be pretty fast.

## Layouts in Xamarin

Layout and content can be added both programmatically and using XAML.

## Accessing screen size – Just to show an example that needs the three projects

No standard way to access, but can setup a public variable in App and then update it from Android and iOS: (notice there is a variable in the main project that can be accessed from the platform projects)

In App:

public static Size ScreenSize;

public static Label label;

public App ()

{ InitializeComponent();

label = new Label { Text = "Size?" };

MainPage = new ContentPage { Content = label};

}

In Android MainActivity.cs onCreate before LoadApplication() line: App.ScreenSize = new Xamarin.Forms.Size( Resources.DisplayMetrics.WidthPixels, Resources.DisplayMetrics.HeightPixels); // can divide each by Density if wanted

After loadapplication line:

App.label.Text = App.ScreenSize.ToString();

In iOS FinishedLaunching: App.ScreenSize = new Xamarin.Forms.Size(UIScreen.MainScreen.Bounds.Width, UIScreen.MainScreen.Bounds.Height);

Run the app to see screensize. We’re accessing the label in the App class from the Android project and showing something Android “knows about” (screen size). There are other ways to get this info, but this shows one way the two projects can interact.

## Example to show grid of squares

Add constants to App class. Const int padding = 10; const int spacing = 5; const int count = 5

In the App constructor, calculate the appropriate box size based on screen size:

var boxSize = Math.Abs((ScreenSize.Width - 2 \* padding + spacing) / count – spacing);

Programatically add a ColorGrid as the MainPage (replace current MainPage = line):

MainPage = new ContentPage {

Padding = padding,

Content = new ColorGrid(boxSize, count, count) { //this will be a grid of BoxViews

RowSpacing = spacing,

ColumnSpacing = spacing,

VerticalOptions = LayoutOptions.CenterAndExpand,

},

};

ColorGrid class:

public class ColorGrid: Grid

{

public ColorGrid(double boxSize, int rows, int columns)

{

//just a loop to create a bunch of boxes in the grid with dynamic colors based on position

for (var row = 0; row < rows; row++)

for (var column = 0; column < columns; column++) {

var box = new BoxView {

Color = Color.FromRgb(row \* 256 / rows, column \* 256 / columns, 127),

WidthRequest = boxSize,

HeightRequest = boxSize,

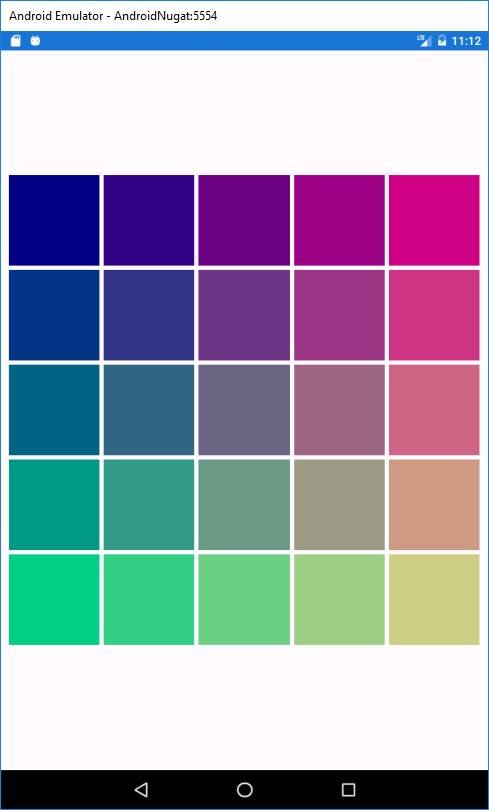
};

Children.Add(box, row, column);

}

}

}



# XAML Introduction

Quick example using a stack layout, a button, and a label. Clicking the button changes the label.  
Inside ContentPage tag inside MainPage.xaml file:

<StackLayout HorizontalOptions="CenterAndExpand" VerticalOptions="CenterAndExpand">

<Label x:Name="helloLabel" Text="Hello, XAML!" />

<Button x:Name="clickButton" Text="Click me!" />

</StackLayout>

Access the button and label by their names in the code after MainPage has been initialized in the App constructor (and add an event handler):

var label = MainPage.FindByName<Label>("helloLabel");

var button = MainPage.FindByName<Button>("clickButton");

button.Clicked += (sender, e) => { label.Text = "You did it!"; MainPage.BackgroundColor = new Color(150, 0, 150); } ;

# Same thing using C#

Comment out the stuff in App constructor, then (can also add events but won’t bother):

StackLayout layout = new Stacklayout { HorizontalOptions= LayoutOptions.CenterAndExpand, VerticalOptions=LayoutOptions.CenterAndExpand };

Layout.Children.Add(new Label { Text = “Hello XAML”});

Layout.Children.Add(new Button { Text = “Click Me!”});

MainPage = new ContentPage { Content = layout};

# Exercise: Simple Calculator

Create a simple calculator with 2 text boxes, a spinner, a button, and a label. Use XAML.