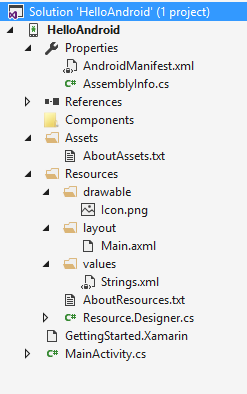
**Learning Outcomes**

1. Understanding android UI and controls
2. Working with xml layout files

**Understanding Folder Structure in AndroidProject**



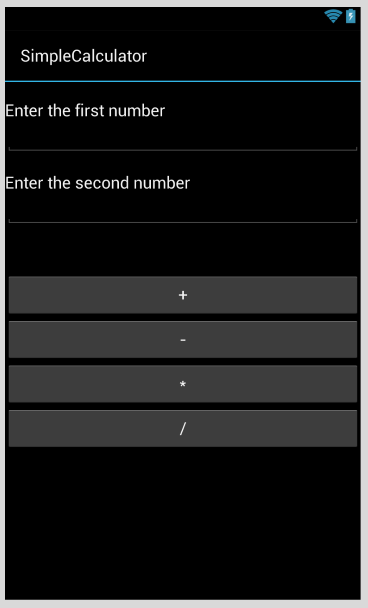
**Assets folder** is the location where the assets (image, xml, any file) are kept. These files would be copied along with the package file (APK) and copied to the device.

**Drawable folder** is the location where the images used for background, custom control, etc… and the xml layout for custom control.

**Layout folder** is the location for storing the layouts. Typically an AXML file could be designed in the designer, which has an XML view. But along with these AXML files, XML layouts are also kept here. Later we would see how we load a layout dynamically (from XML).

**Values folder** should contain Strings.xml and Styles.xml.

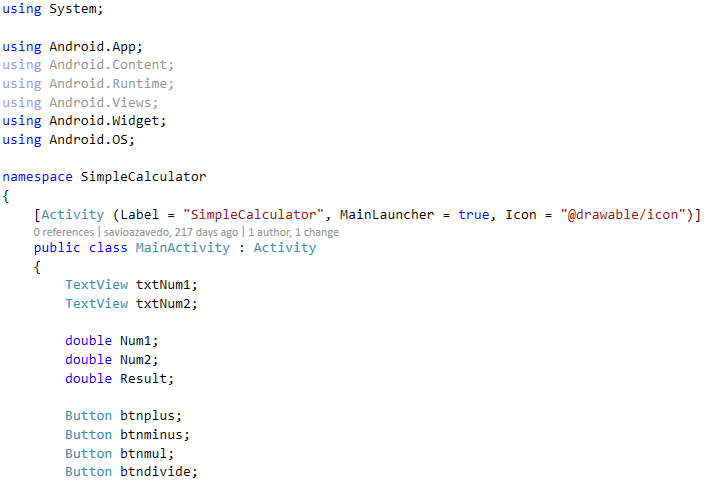
**Creating a Simple Calculator**

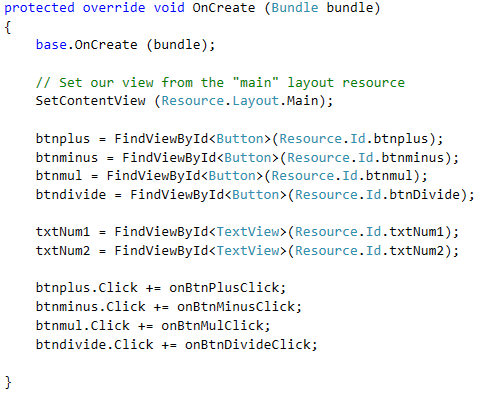


Create a layout as with two textboxes, labels and four buttons as shown above.

<?xml version="1.0" encoding="utf-8"?>  
<AbsoluteLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:id="@+id/widget0"  
    android:layout\_width="fill\_parent"  
    android:layout\_height="fill\_parent">  
    <Button  
        android:id="@+id/btnplus"  
        android:layout\_width="50dp"  
        android:layout\_height="wrap\_content"  
        android:text="+"  
        android:layout\_x="26dp"  
        android:layout\_y="290dp" />  
    <Button  
        android:id="@+id/btnminus"  
        android:layout\_width="50dp"  
        android:layout\_height="wrap\_content"  
        android:text="-"  
        android:layout\_x="97dp"  
        android:layout\_y="291dp" />  
    <Button  
        android:id="@+id/btnmul"  
        android:layout\_width="50dp"  
        android:layout\_height="wrap\_content"  
        android:text="\*"  
        android:layout\_x="163dp"  
        android:layout\_y="291dp" />  
    <Button  
        android:id="@+id/btnDivide"  
        android:layout\_width="50dp"  
        android:layout\_height="wrap\_content"  
        android:text="/"  
        android:layout\_x="231dp"  
        android:layout\_y="290dp" />  
    <EditText  
        android:id="@+id/txtNum1"  
        android:layout\_width="192dp"  
        android:layout\_height="wrap\_content"  
        android:textSize="18sp"  
        android:numeric="decimal"  
        android:layout\_x="118dp"  
        android:layout\_y="68dp" />  
    <EditText  
        android:id="@+id/txtNum2"  
        android:layout\_width="194dp"  
        android:layout\_height="wrap\_content"  
        android:textSize="18sp"  
        android:layout\_x="116dp"  
        android:layout\_y="146dp" />  
    <TextView  
        android:id="@+id/widget44"  
        android:layout\_width="wrap\_content"  
        android:layout\_height="wrap\_content"  
        android:text="First Number"  
        android:layout\_x="6dp"  
        android:layout\_y="77dp" />  
    <TextView  
        android:id="@+id/widget45"  
        android:layout\_width="wrap\_content"  
        android:layout\_height="wrap\_content"  
        android:text="Second Number"  
        android:layout\_x="7dp"  
        android:layout\_y="157dp" />  
</AbsoluteLayout>

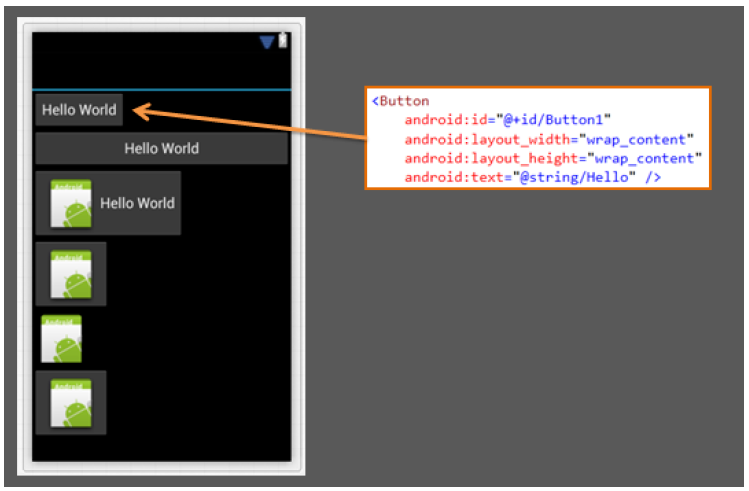
Before writing the code **build** your project so that the resource file is set up.

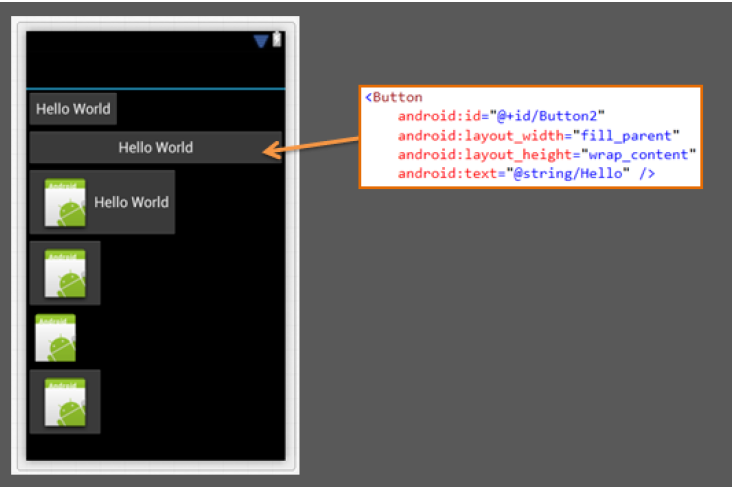


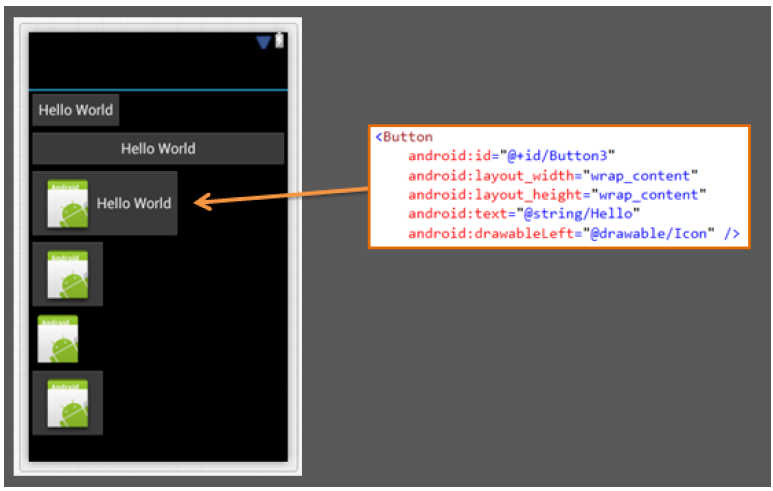


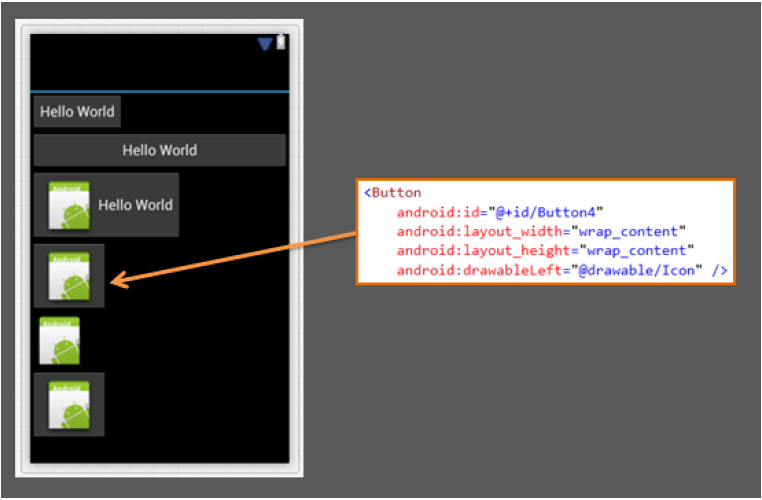


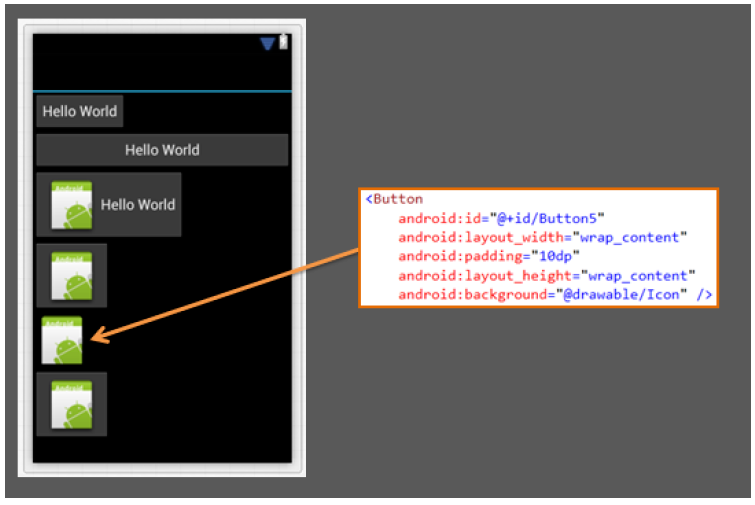
**Styles of Button**

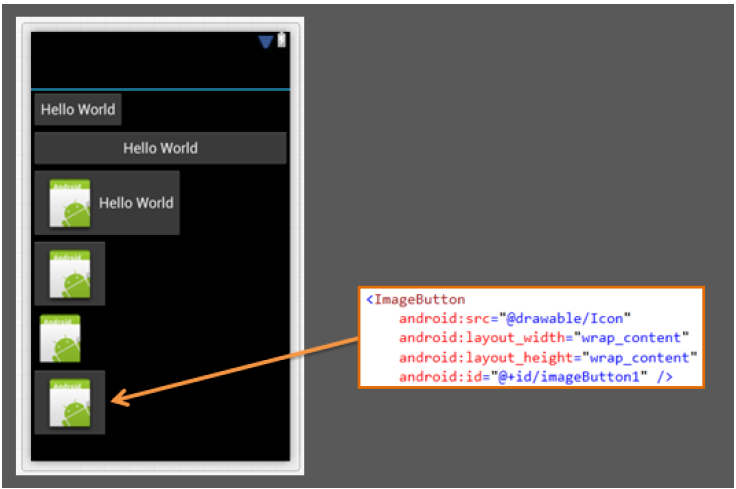




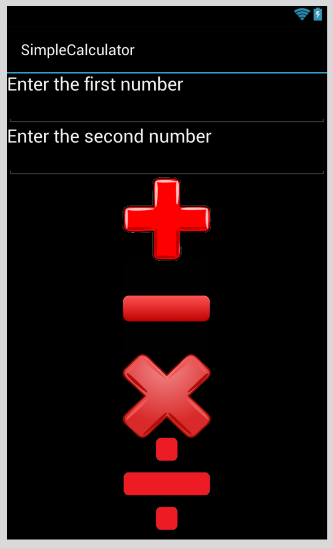








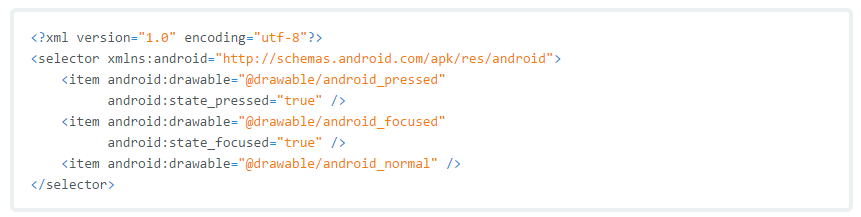
Change your calculator to Image buttons



**Custom Buttons** [(https://developer.xamarin.com/guides/android/user\_interface/form\_elements/custom\_button/)]((https:/developer.xamarin.com/guides/android/user_interface/form_elements/custom_button/))

In this section, you will create a button with a custom image instead of text, using the Button widget and an XML file that defines three different images to use for the different button states. When the button is pressed, a short message will be displayed.

* Copy the images below into the *Resources*\drawable\ directory of your project. These will be used for the different button states.  
  
* Create a new file in the *Resources*\drawable\ directory named android\_button.xml. Insert the following XML:



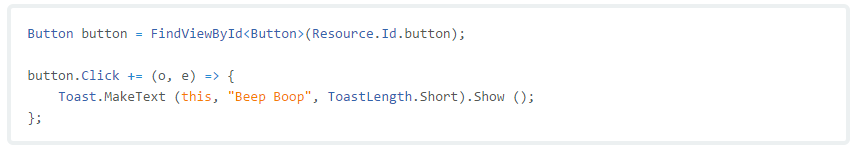
This defines a single drawable resource, which will change its image based on the current state of the button. The first<item> defines android\_pressed.png as the image when the button is pressed (it's been activated); the second <item>defines android\_focused.png as the image when the button is focused (when the button is highlighted using the trackball or directional pad); and the third <item> defines android\_normal.png as the image for the normal state (when neither pressed nor focused). This XML file now represents a single drawable resource and when referenced by a [Button](http://androidapi.xamarin.com/?link=T%3aAndroid.Widget.Button) for its background, the image displayed will change based on these three states.

Open the Resources\layout\main.xml file and add the [Button](http://androidapi.xamarin.com/?link=T%3aAndroid.Widget.Button) element:



The android:background attribute specifies the drawable resource to use for the button background (which, when saved at *Resources\*drawable android.xml, is referenced as @drawable/android). This replaces the normal background image used for buttons throughout the system. In order for the drawable to change its image based on the button state, the image must be applied to the background.

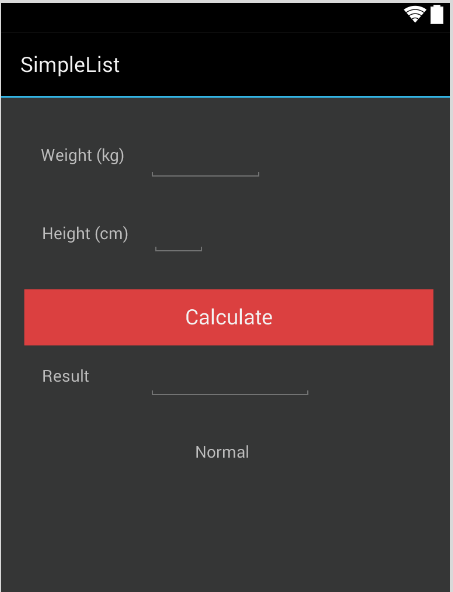
To make the button do something when pressed, add the following code at the end of the [OnCreate()](http://androidapi.xamarin.com/?link=M%3aAndroid.App.Activity.OnCreate(Android.OS.Bundle)" \t "_blank) method:



This captures the [Button](http://androidapi.xamarin.com/?link=T%3aAndroid.Widget.Button) from the layout, then adds a [Toast](http://androidapi.xamarin.com/?link=T%3aAndroid.Widget.Toast) message to be displayed when the [Button](http://androidapi.xamarin.com/?link=T%3aAndroid.Widget.Button) is clicked.Now run the application.

**Practice**

**BMI Calculator**



Design an app as shown above using DroidDraw

The formula for BMI goes like this

**BMI = Weight / (Height \* Height)**

|  |  |
| --- | --- |
| BMI | Result |
| < 18.5 | Underweight |
| 18.6 to 24.9 | Normal |
| 25 to 29.9 | Overweight |
| > 30 | Obese |

1. Design a Math Tester using Random Numbers