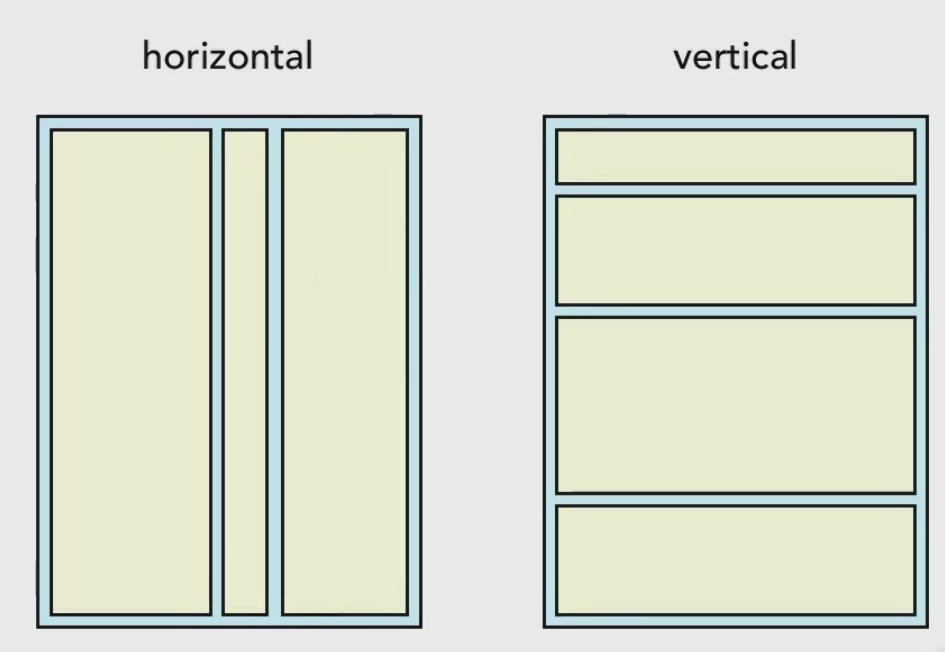
**Learning Outcomes**

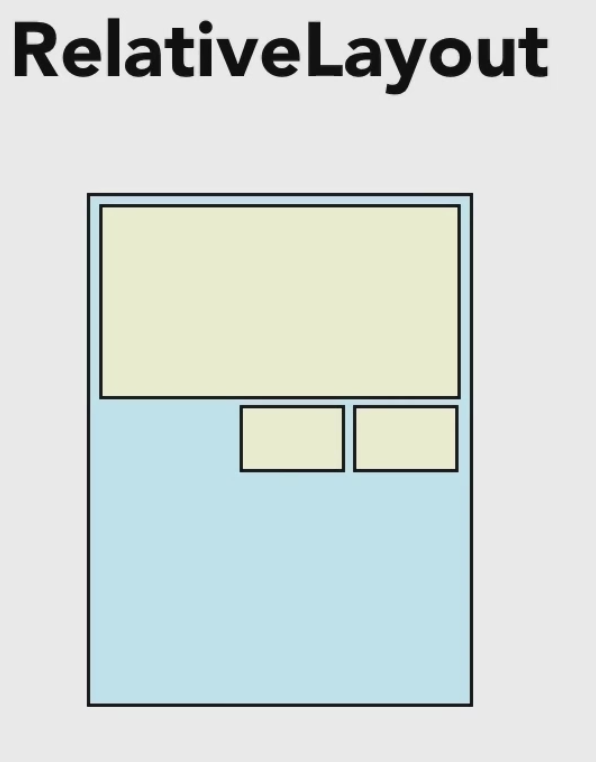
1. Layout fundamentals
2. Creating Responsive Layouts
3. Handling Device Orientation

**Basic Layouts**

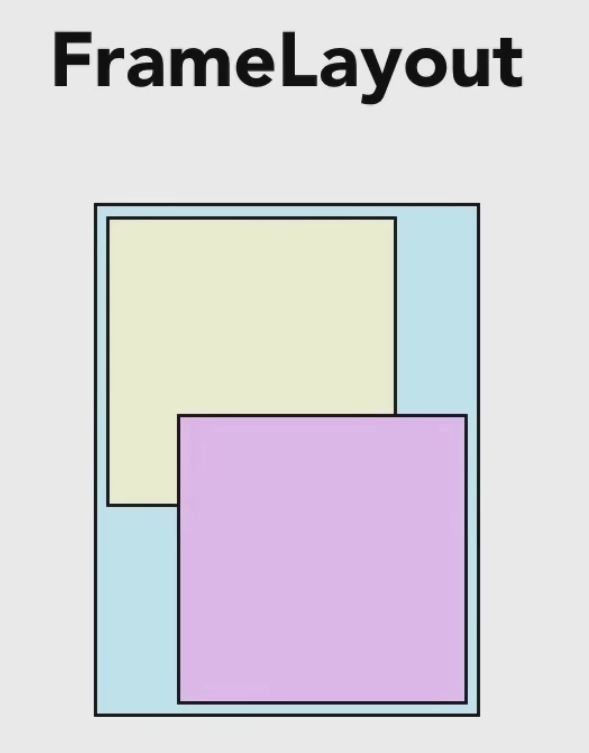
## **Linear Layout**



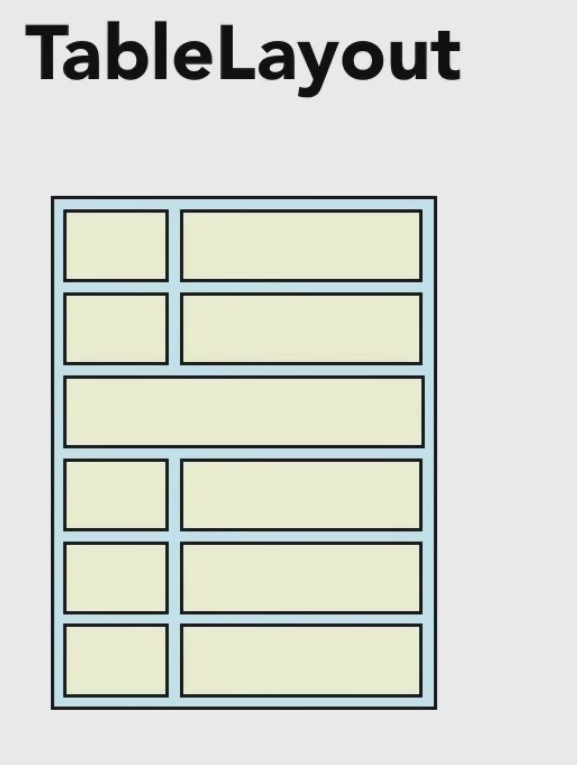
**Relative Layout**



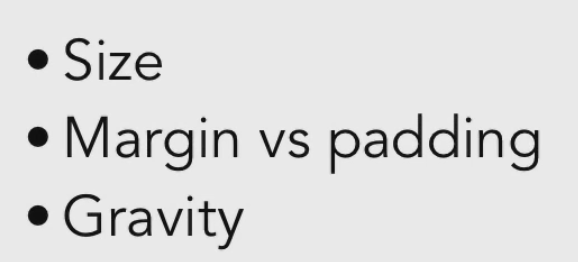
**Frame Layout**



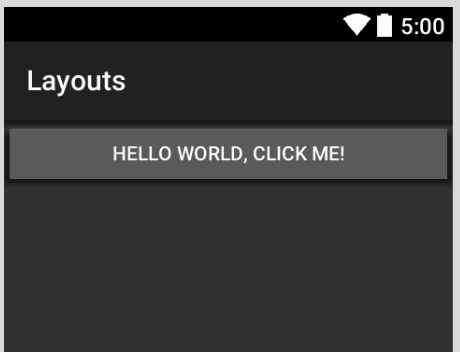
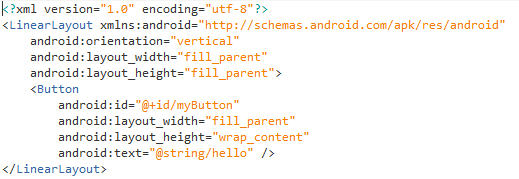
**Table Layout**

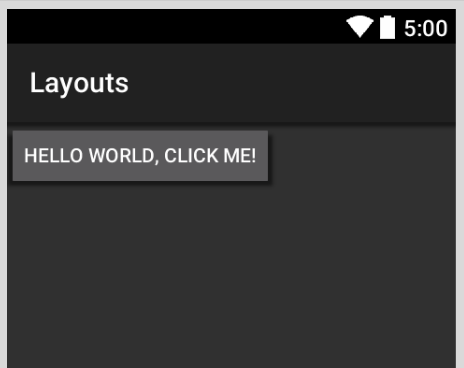
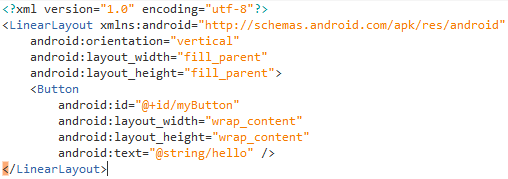


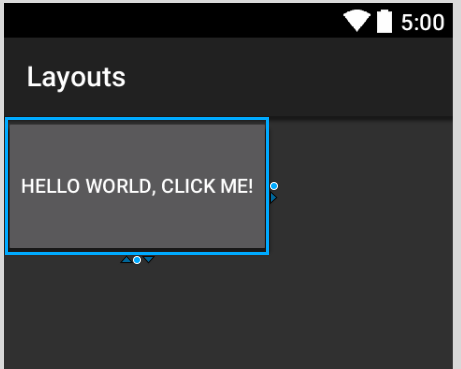
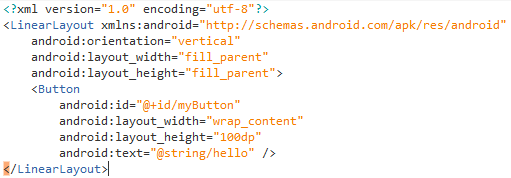
**Basic Attributes**



Fill\_Parent and Wrap\_Content

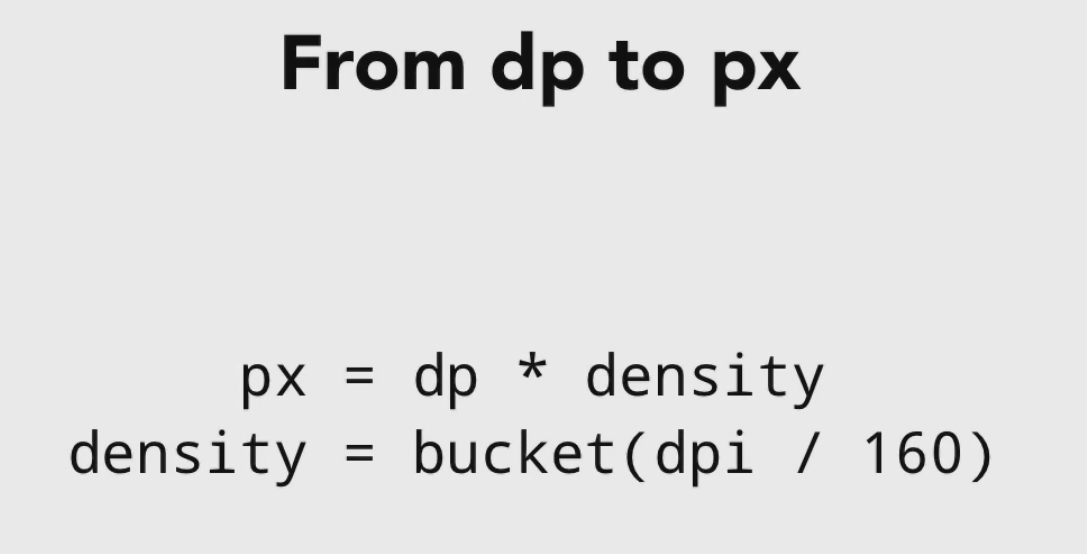






**What is “DP”**





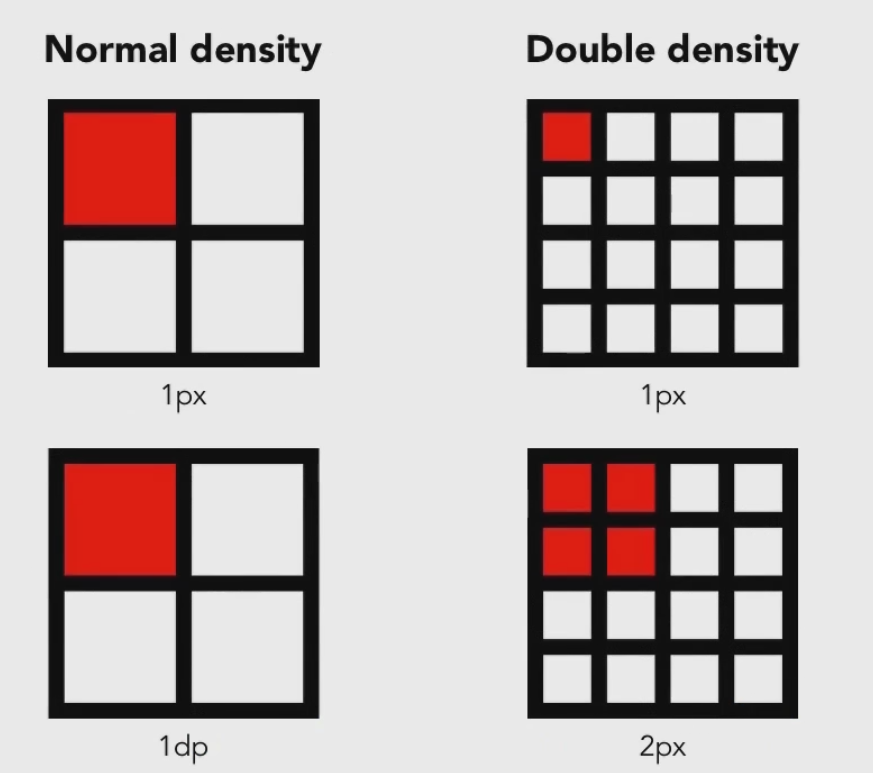
When Android was first launched it was on a device with 160 dpi, later different devices came into the market with different density and dpi values.

160 became the base value for density which is mdpi (medium density) with a multiplication factor of 1.

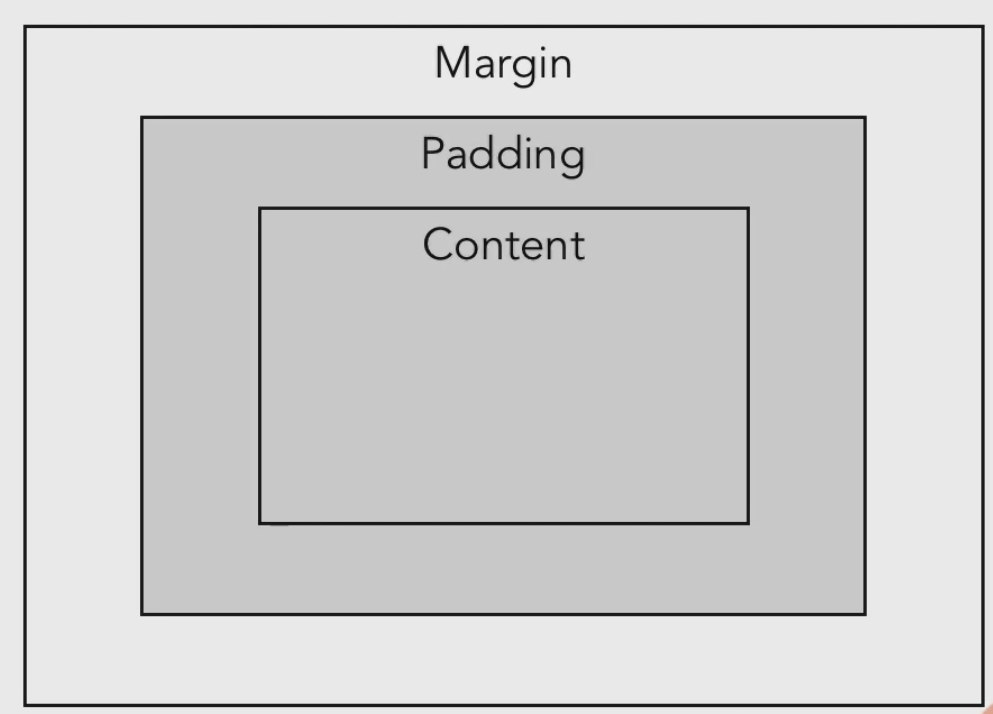
So 100 dp on mdpi device is 100 \* 1 = 100 px

100 dp on hdpi device is 100 \* 1.5 = 150 px



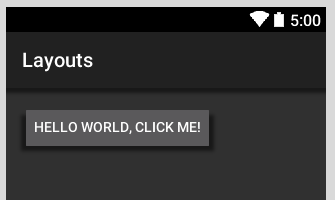


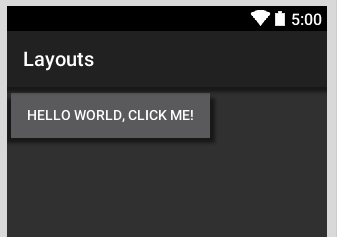
**Margin and Padding**



Padding - Space within the view

Margin - Space between the parent and the view

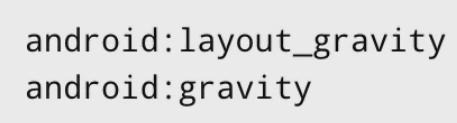
 





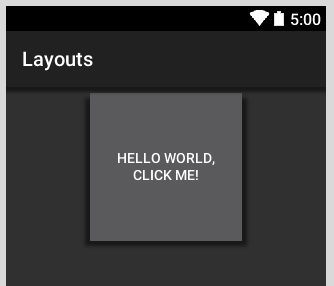
If the attribute has the keyword layout, it specifies it’s relation with the parent

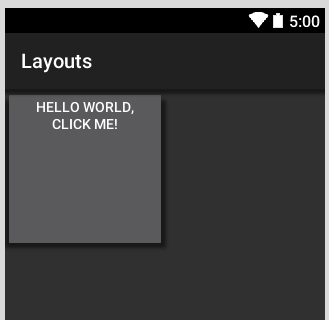
**Layout Gravity and Gravity**



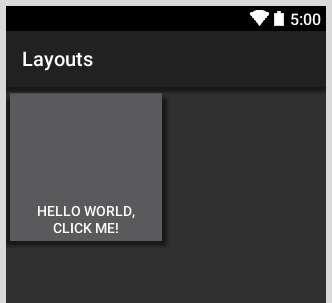
Layout\_Gravity – position of the view with respect to its parent

Gravity – position of the content within the view

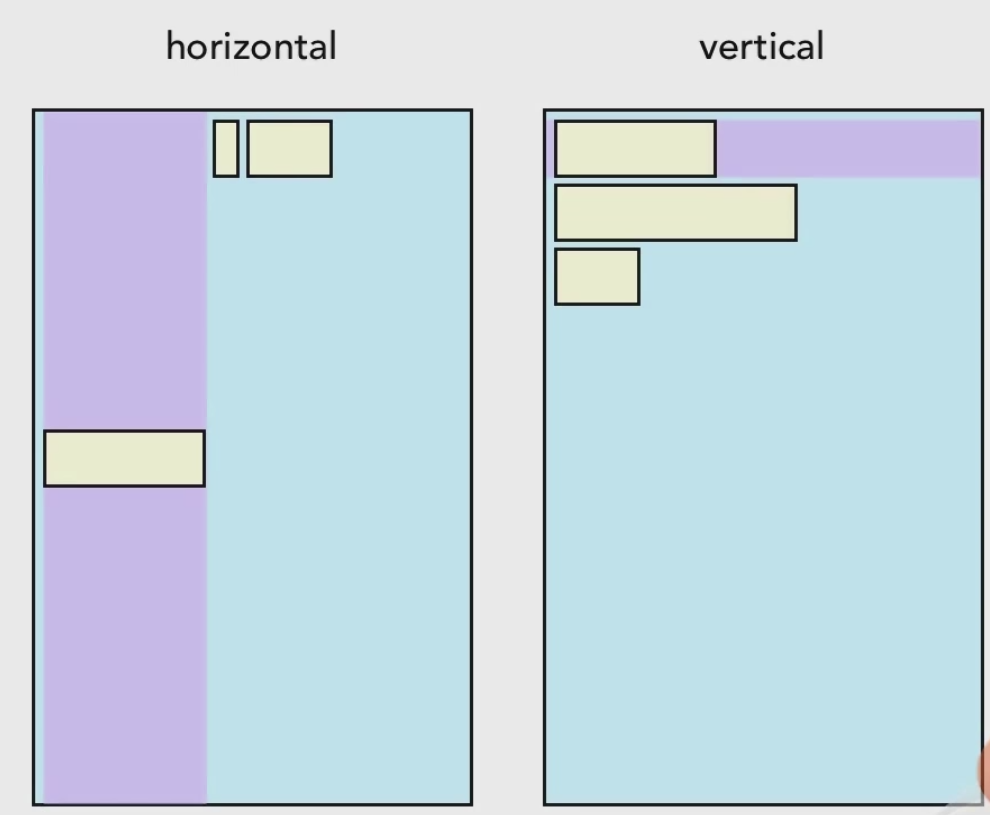




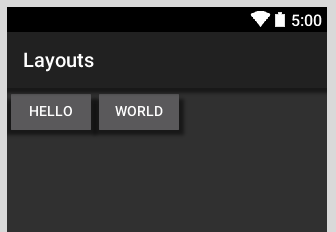
**Specifying more than one attribute**



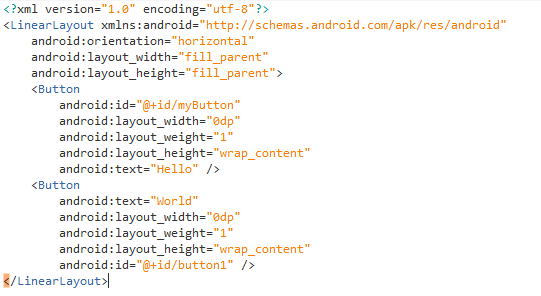
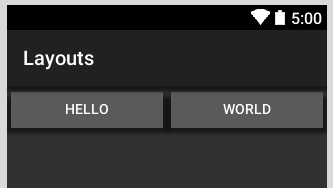
**Linear Layout**



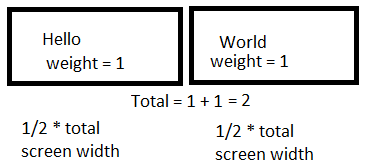
android:orientation = “horizontal” or “verticle”

Assigning weights to make the view proportionate and take space with respect to the layout.

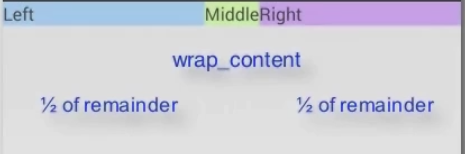
 

So how does android calculate the width from the weight?

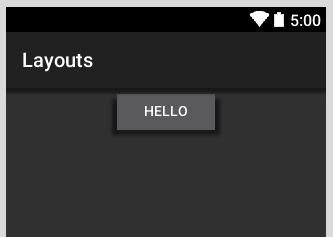
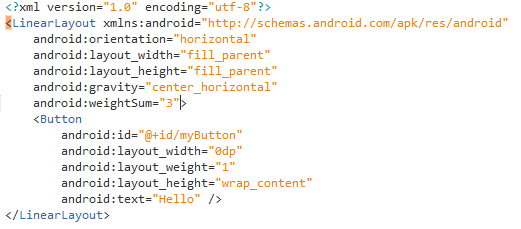


Similarly for three views with layout\_weight = “1”

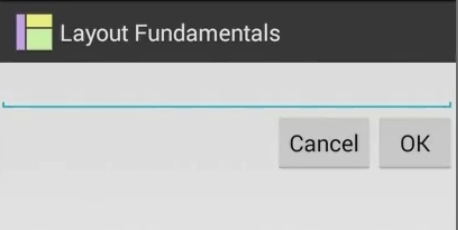




**weight\_sum**

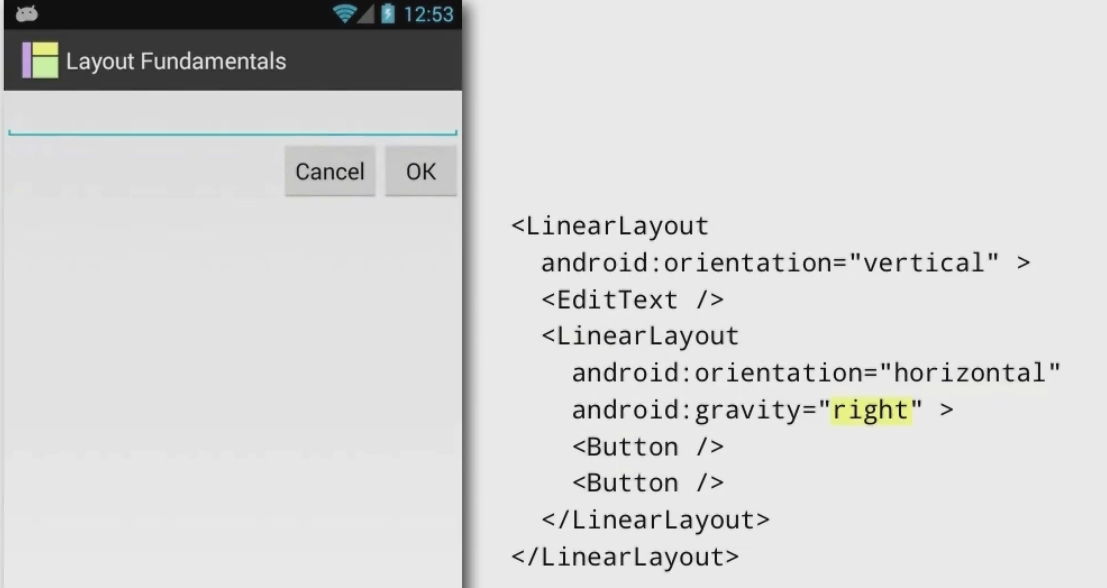


**Nested Linear layout**

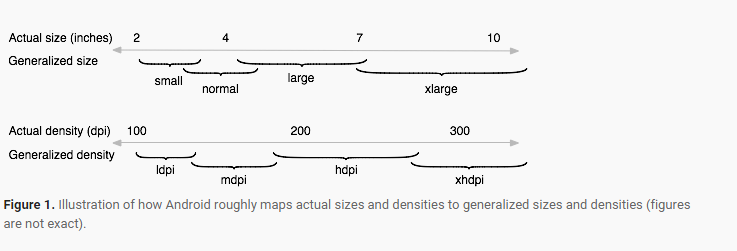


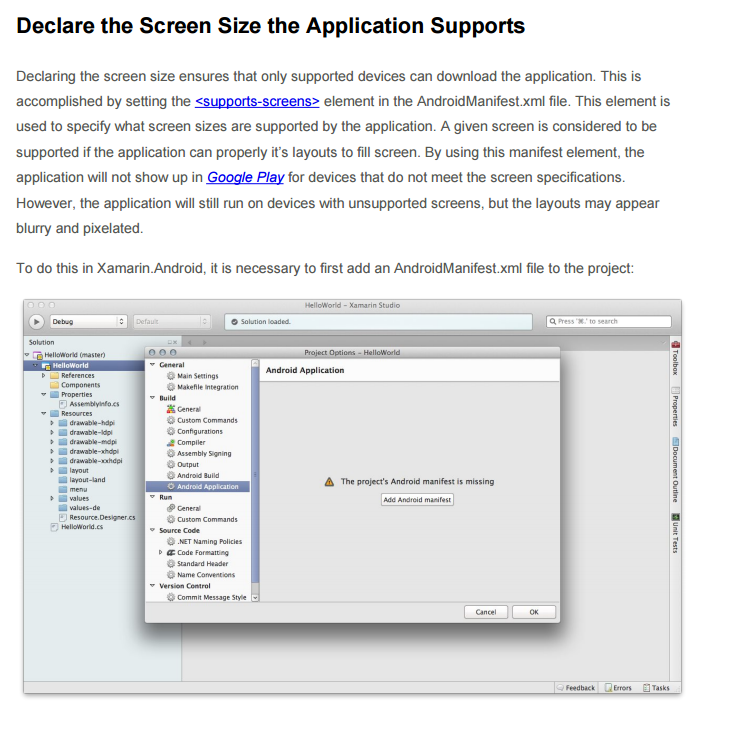
Solution a nested linear layout

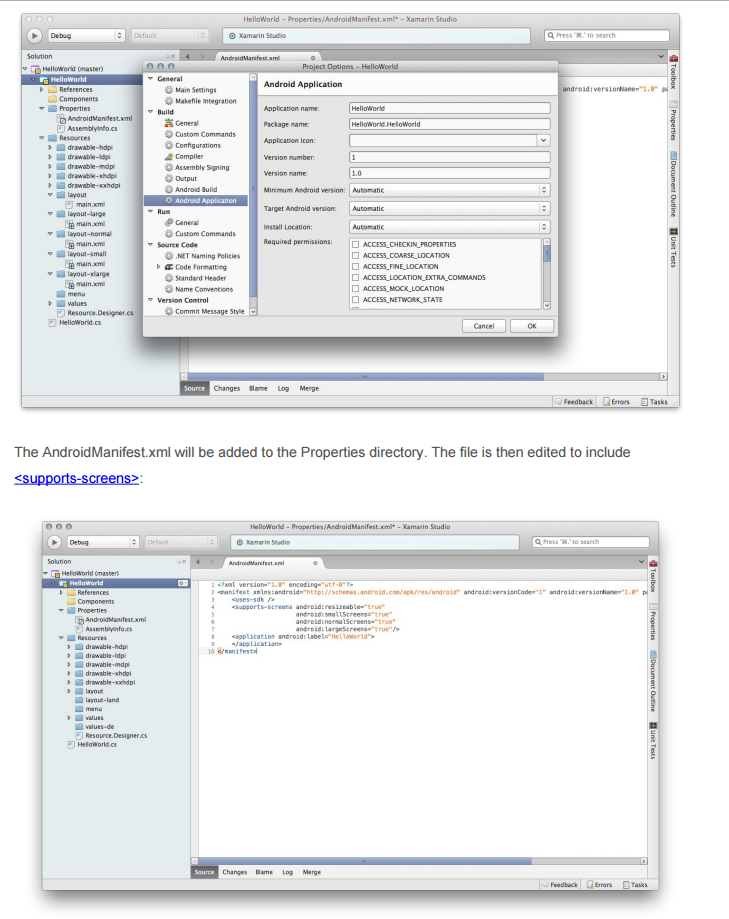


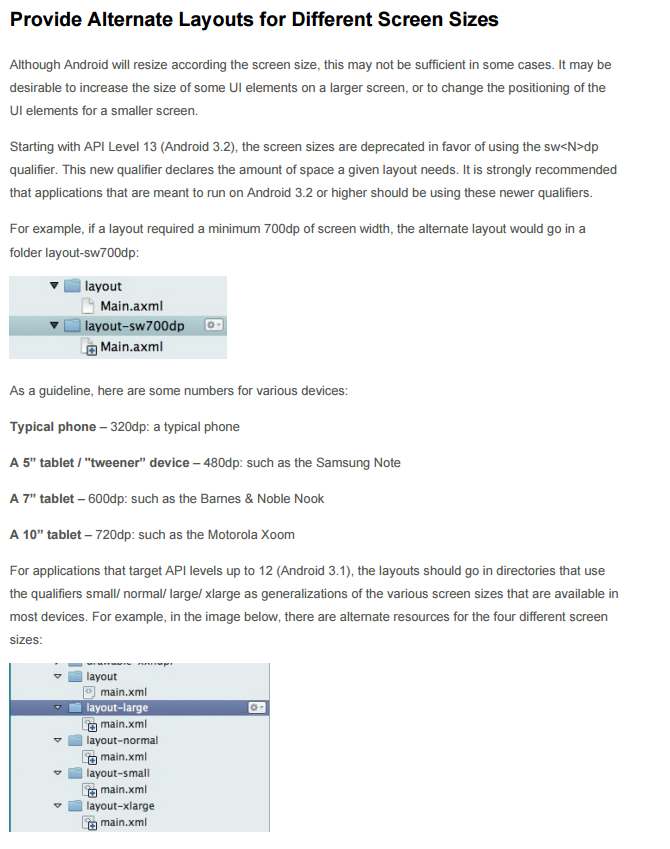


**Supporting multiple screen sizes**

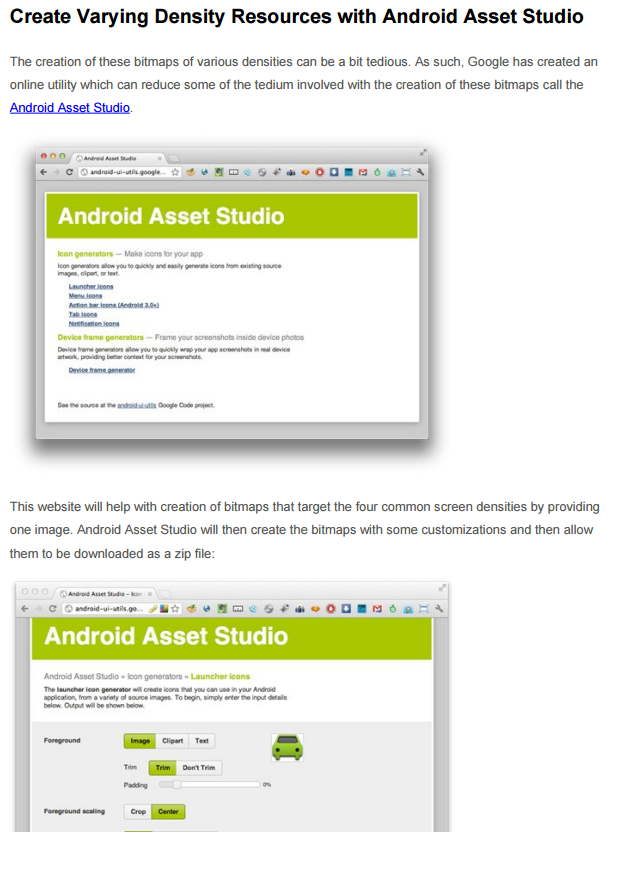


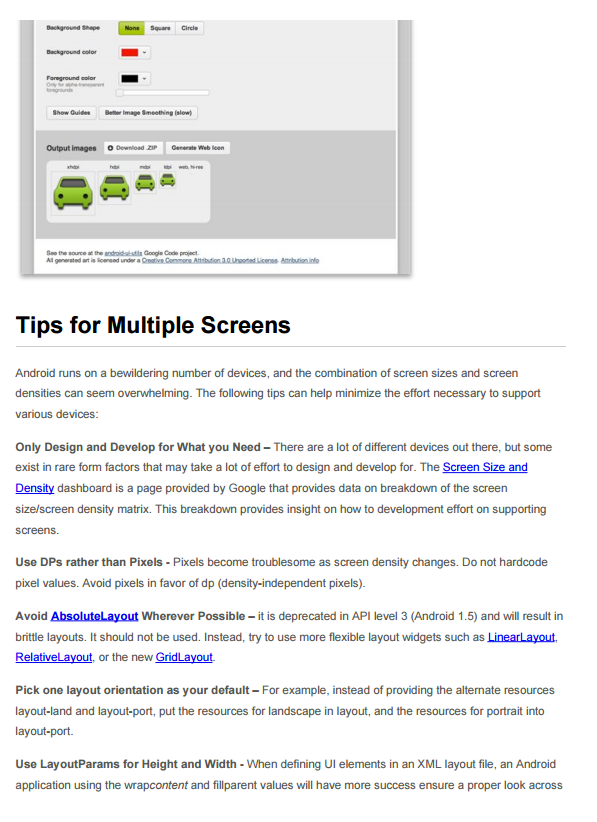


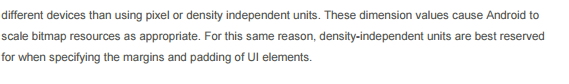




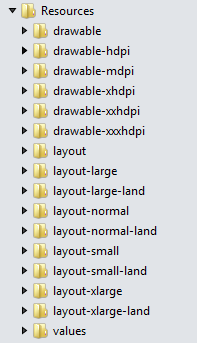






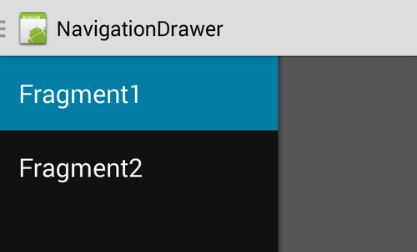
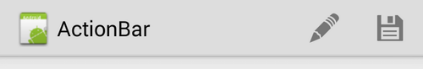
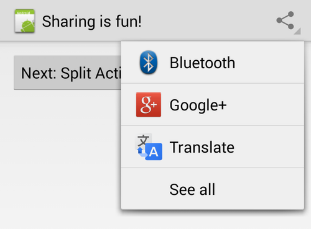


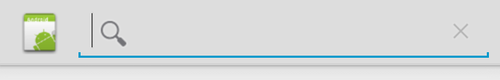
Following is a list of project folders that you would normally have



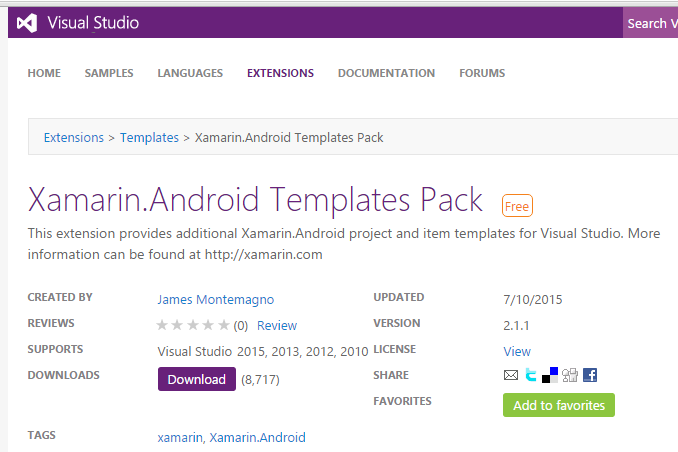
**Visual Studio Template pack**

The Xamarin Android template pack features some of the most common app templates that you would use such as a navigation drawer.

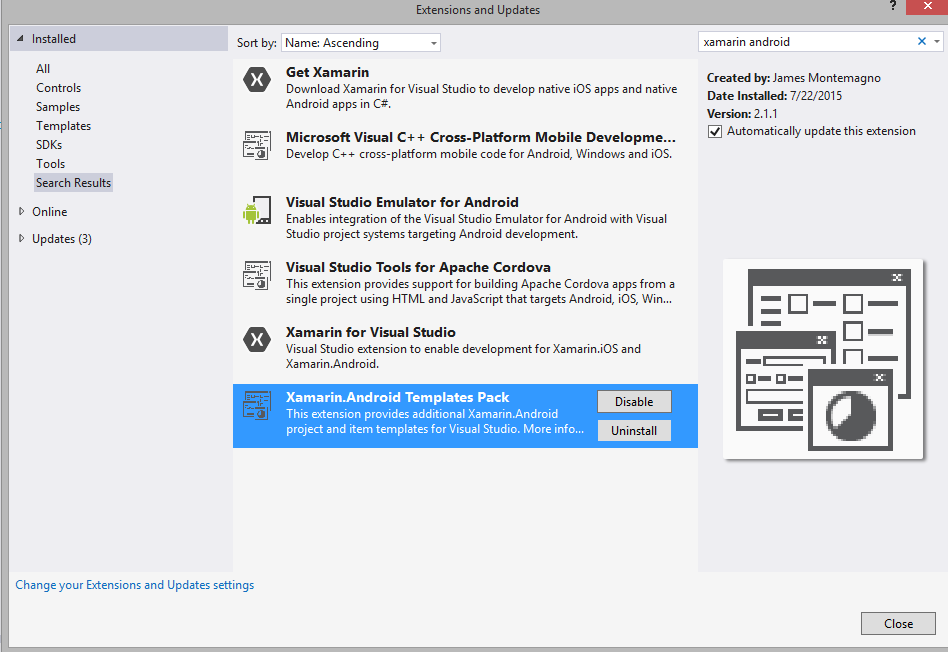
Navigation Drawer:   
  
  
Menu Item:  
  
  
Share Menu:  


Search Menu:  


Download the template pack from the Visual Studio gallery online or



Get it from the Visual Studio Gallery under “Tools->Extensions and Updates”



**Watch the video below to get more information about the templates**

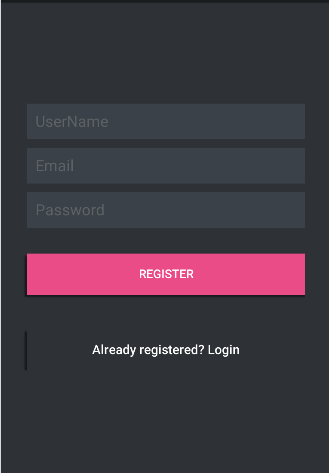
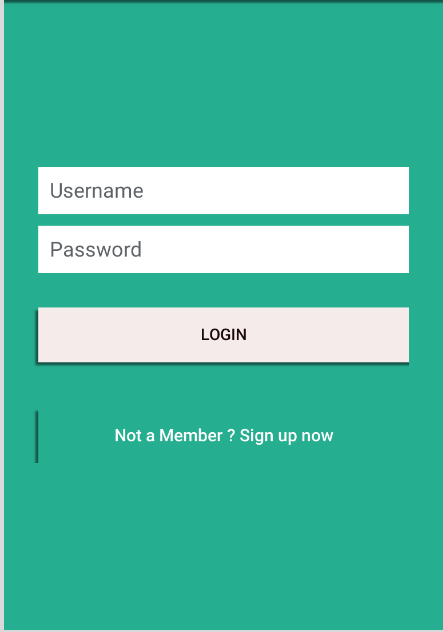
<https://youtu.be/Ebsiq1sIiBM>

These templates are only for Visual studio.

**Exercise**

Design a neat looking Login and Registration screen that fits all screen sizes.

You can use your own color combination

* Find out how you can fix the orientation of an app to just portrait
* Try to get rid of the title-bar from your app