UI Element Sizing

## Background

This module demonstrates the importance of sizing of UI elements..

A working version of this app is available at: <https://github.com/milk-modules/Apps/tree/master/accessible/DemoApp03>

Further reading: <https://material.io/guidelines/layout/metrics-keylines.html#metrics-keylines-touch-target-size>

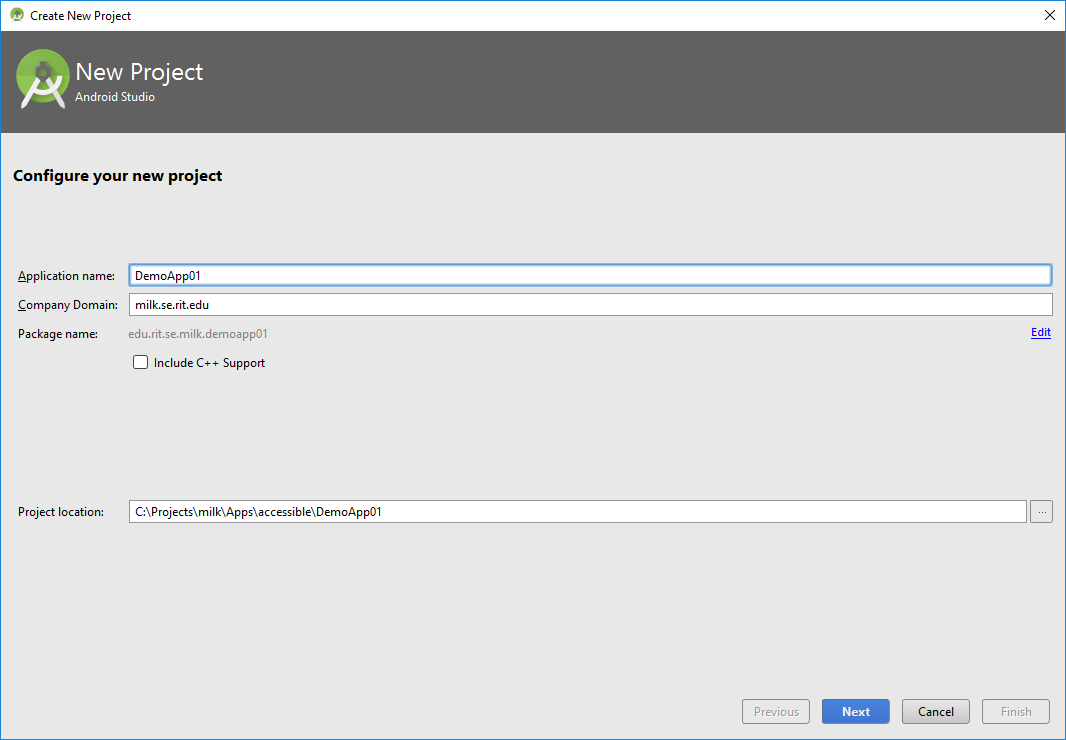
## Prerequisite

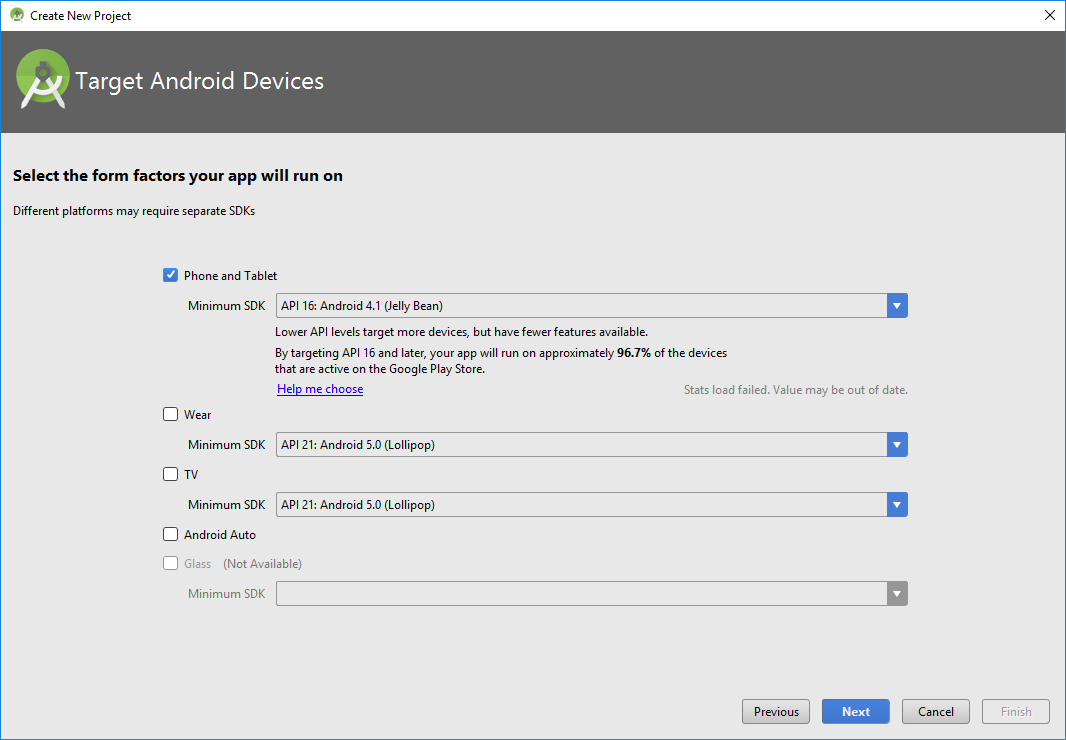
1. Android studio is installed on the development workstation
2. A working Android emulator is available for testing

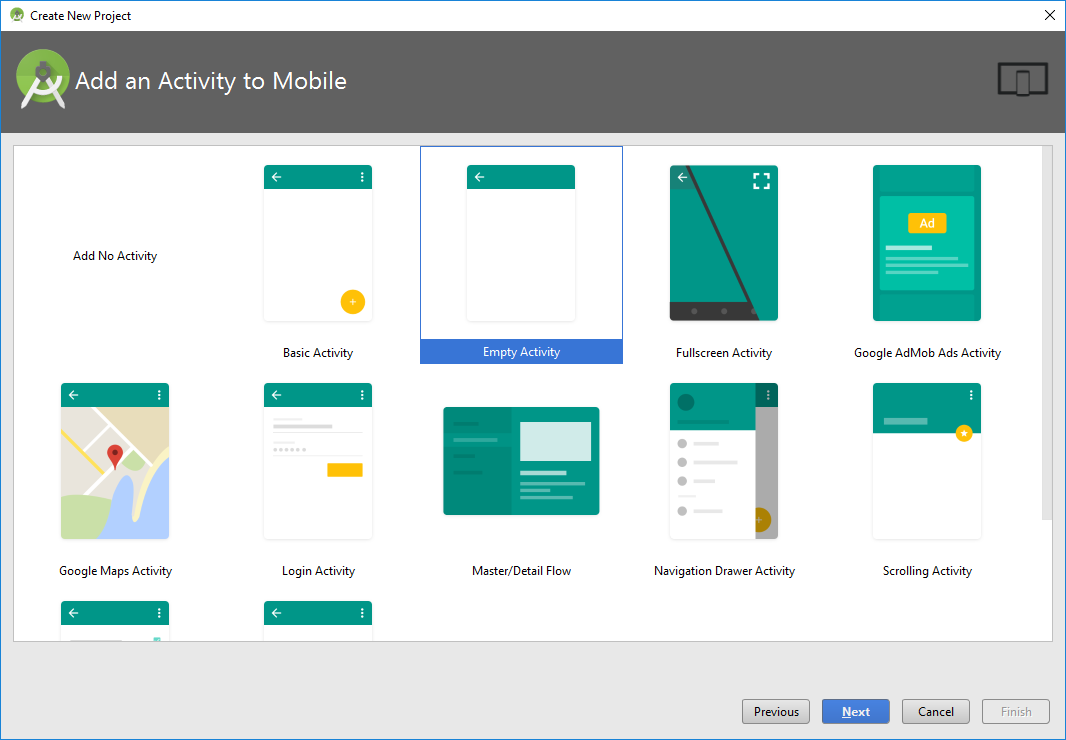
## Activity Instructions

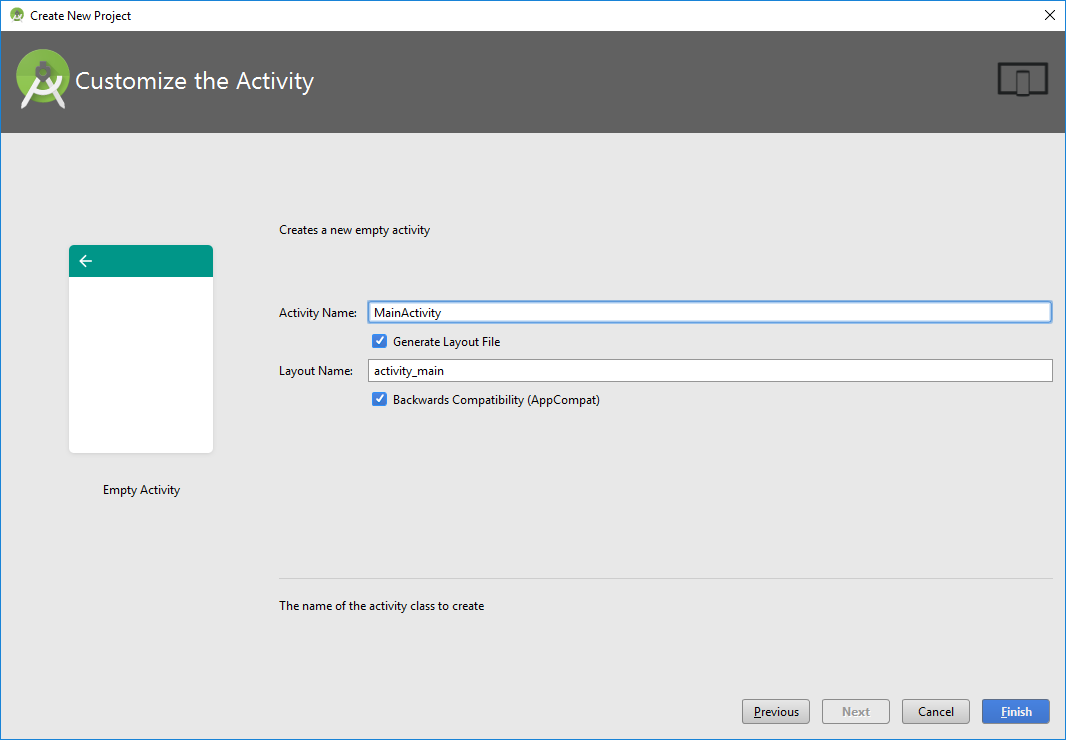
There two approaches that you can take to perform this activity:

1. End-to-End development of the app by following all the below steps
2. Using a pre-created version of this project and only resizing of the UI elements:
   1. Download the code for DemoApp03 from: <https://github.com/milk-modules/Apps/tree/master/non-accessible>
   2. Perform ONLY step #3
3. Project Creation
   1. Follow the screens below to create a new project:









1. Construct User Interface
   1. From the Palette tool window, add the following UI controls into the screen layout.
      1. Update the following properties of the existing Relative Layout:
         * layout\_width="match\_parent"
         * layout\_height="match\_parent"
      2. Within the existing Relative Layout add:
         1. **Switch**
            * Update the following properties:

text="Alternate Rendering"

layout\_width="match\_parent"

layout\_height="wrap\_content"

id="@+id/switchAccessibility"

focusable="false"

* + - 1. **Relative Layout**:
         * Update the following properties:

layout\_width="match\_parent"

layout\_height="wrap\_content"

id="@+id/layoutContents"

layout\_weight="100"

* + - * + Add the following UI controls:

**Button**:

Update the following properties:

android:text="Yes"

android:layout\_width="50dp"

android:layout\_height="35dp"

android:layout\_alignParentTop="true"

android:layout\_alignParentLeft="true"

android:layout\_alignParentStart="true"

android:layout\_marginTop="146dp"

android:id="@+id/buttonLeft"

android:background="@android:drawable/btn\_default"

android:gravity="center"

android:layout\_gravity="left|center"

android:layout\_marginLeft="130dp"

**TextView:**

* Update the following properties:

android:text="Is this the greatest app of all time?"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/textView"

android:textAppearance="@android:style/TextAppearance.DeviceDefault.Medium"

android:layout\_above="@+id/buttonRight"

android:layout\_marginBottom="36dp"

android:textStyle="normal|bold"

android:textAlignment="center"

android:layout\_alignParentLeft="false"

android:layout\_alignParentStart="false"

android:layout\_alignParentRight="false"

android:layout\_alignParentEnd="false"

android:gravity="center\_horizontal""

**Button:**

Update the following properties:

android:text="No"

android:layout\_width="45dp"

android:layout\_height="35dp"

android:id="@+id/buttonRight"

android:layout\_gravity="right"

android:background="@android:drawable/btn\_default"

android:gravity="center"

android:layout\_alignTop="@+id/buttonLeft"

android:layout\_alignParentRight="true"

android:layout\_alignParentEnd="true"

android:layout\_marginRight="130dp"

* + - 1. **Relative** **Layout**
         * Update the following properties:

layout\_width="match\_parent"

layout\_height="wrap\_content"

layout\_weight="1"

* + - * + Add the following UI controls:

**TextView**:

Update the following properties:

layout\_width="match\_parent"

layout\_height="wrap\_content"

id="@+id/textViewStatus"

layout\_alignParentTop="true"

layout\_alignParentLeft="true"

layout\_alignParentStart="true"

textAlignment="center"

textStyle="normal|bold"

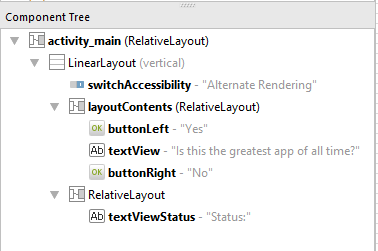
layout\_alignParentRight="true"

layout\_alignParentEnd="true"

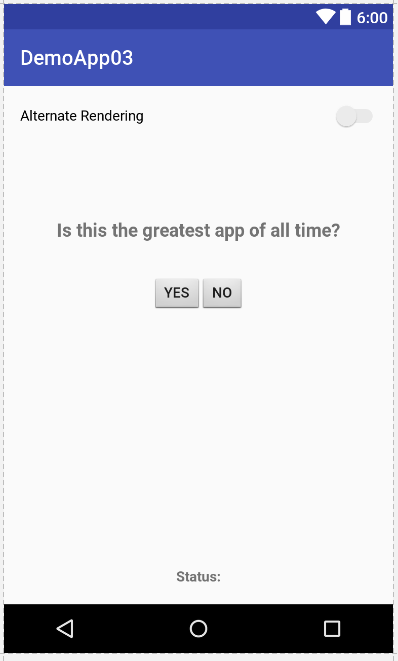
text="Status:"

gravity="bottom"

Following is the hierarchical layout of the controls on the screen:



Following is the rendering of controls on the screen:



1. Resize & Respace Buttons

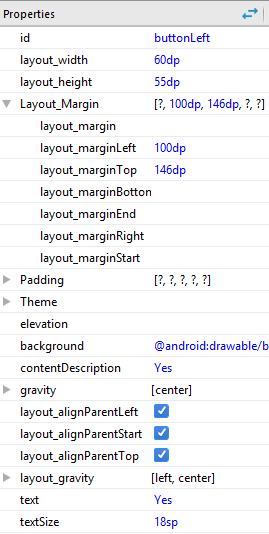
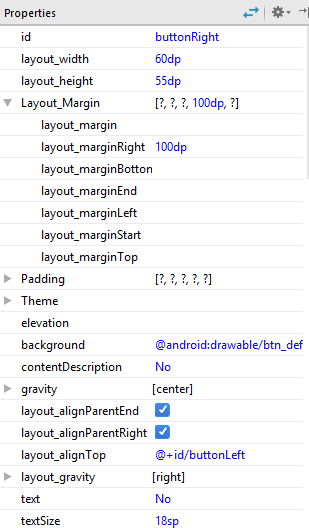
The recommend size for touch targets should be at least 48 x 48 dp. There should also be a spacing of 8dp or more between the targets.

Select the Button with id “buttonLeft”. Update the following properties:

* layout\_width = 60dp
* layout\_height = 55dp
* layout\_marginLeft = 100dp
* textSize = 18sp

Select the Button with id “buttonRight”. Update the following properties:

* layout\_width = 60dp
* layout\_height = 55dp
* layout\_marginRight = 100dp
* textSize = 18sp

1. Code

Open MainActivity.java and add the following code:

**package** edu.rit.se.milk.demoapp03;  
  
**import** android.graphics.Color;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.support.v7.widget.ButtonBarLayout;  
**import** android.view.Gravity;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.Button;  
**import** android.widget.CompoundButton;  
**import** android.widget.LinearLayout;  
**import** android.widget.RelativeLayout;  
**import** android.widget.Switch;  
**import** android.widget.TextView;  
**import** android.widget.RelativeLayout.LayoutParams;  
  
**import** java.util.Random;  
**import** java.util.Timer;  
**import** java.util.TimerTask;  
  
**import static** android.R.attr.***button***;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 Button **buttonLeft**, **buttonRight**;  
 TextView **textStatus**;  
 Switch **switchRendering**;  
  
 RelativeLayout **layoutCover**;  
  
 Timer **buttonMoveTimer**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 **buttonLeft** = (Button) findViewById(R.id.***buttonLeft***);  
 **buttonRight** = (Button) findViewById(R.id.***buttonRight***);  
 **textStatus** = (TextView) findViewById(R.id.***textViewStatus***);  
 **switchRendering** = (Switch) findViewById(R.id.***switchAccessibility***);  
 **layoutCover** = (RelativeLayout) findViewById(R.id.***layoutContents***);  
  
 setupEventHandlers();  
 }  
  
 **private void** setupEventHandlers() {  
 **buttonLeft**.setOnClickListener(  
 **new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **textStatus**.setText(**"Button Tapped: \"Yes\""**);  
 }  
 }  
 );  
  
 **buttonRight**.setOnClickListener(  
 **new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **textStatus**.setText(**"Button Tapped: \"No\""**);  
 }  
 }  
 );  
  
 **switchRendering**.setOnCheckedChangeListener(**new** CompoundButton.OnCheckedChangeListener() {  
 @Override  
 **public void** onCheckedChanged(CompoundButton buttonView, **boolean** isChecked) {  
 **if** (isChecked) {  
 *// reset();* **buttonMoveTimer** = **new** Timer();  
 **buttonMoveTimer**.schedule(**new** TimerTask() {  
 @Override  
 **public void** run() {  
 runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 moveButton();  
 }  
 });  
 }  
 }, 0, 50);  
 } **else** {  
 reset();  
 **buttonMoveTimer**.cancel();  
 }  
 }  
 });  
 }  
  
 **private void** reset() {  
 setContentView(R.layout.***activity\_main***);  
  
 **buttonLeft** = (Button) findViewById(R.id.***buttonLeft***);  
 **buttonRight** = (Button) findViewById(R.id.***buttonRight***);  
 **textStatus** = (TextView) findViewById(R.id.***textViewStatus***);  
 **switchRendering** = (Switch) findViewById(R.id.***switchAccessibility***);  
 **layoutCover** = (RelativeLayout) findViewById(R.id.***layoutContents***);  
  
 setupEventHandlers();  
 }  
  
 **private void** moveButton() {  
 Random rand = **new** Random();  
 **int** leftX = rand.nextInt(1) + 1;  
 **int** rightX = rand.nextInt(2) + 1;  
 **int** rightY = rand.nextInt(3) + 1;  
 **int** leftY = rand.nextInt(4) + 1;  
  
 **int** directionX = rand.nextInt(2) + 1;  
 **int** directionY = rand.nextInt(2) + 1;  
  
 **if** (directionX == 1) {  
 **buttonLeft**.setX(**buttonLeft**.getX() + leftX);  
 **buttonLeft**.setY(**buttonLeft**.getY() + leftY);  
  
 **buttonRight**.setX(**buttonRight**.getX() - rightX);  
 **buttonRight**.setY(**buttonRight**.getY() - rightY);  
 } **else** {  
 **buttonLeft**.setX(**buttonLeft**.getX() - leftX);  
 **buttonLeft**.setY(**buttonLeft**.getY() - leftY);  
  
 **buttonRight**.setX(**buttonRight**.getX() + rightX);  
 **buttonRight**.setY(**buttonRight**.getY() + rightY);  
 }  
  
  
 }  
}

The above code achieves the following:

1. A timer to move the buttons
2. Handles the onCheckedChange event of the Switch control to:
   1. Move the buttons when the switch is checked (i.e. set to “On”)
   2. Reset the layout of the buttons when unchecked (i.e. set to “Off”)