Lab 2: Layout

Submission Due: 02/02/2021

Objective

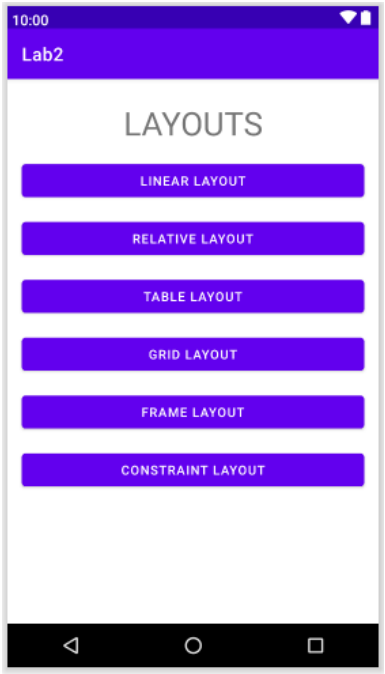
1. Learn how to design UI layouts and the elements they contain using XML.
2. Understand the common layout types: LinearLayout, RelativeLayout, TableLayout, GridLayout, FrameLayout, and ConstraintLayout.

Before Lab

1. Learn how to use Android Studio Layout Editor
   (Refer https://developer.android.com/studio/write/layout-editor)
2. Download the lab2 starter code from Camino (Files -> Lab -> Lab2_StarterCode).
3. Unzip and open the Lab2_StarterCode project in Android Studio.

Lab2_StarterCode

This starter code contains an Activity for each of the layout types and MainActivity. MainActivity contains code to navigate to other Activities (LayoutExamples).



StarterCode contains the code for all the Activities. It also contains some starter code for layouts in XML files.

You only need to complete the corresponding UIs using different layouts. For instance, complete activity_relative_example.xml file to build UI with Relative Layout.

Tasks

Task 1: Build UI with Linear Layout (Complete activity_linear_example.xml)

Build the UI as shown:

Tips: Some useful attributes.

Attributes	Description
android:layout_gravity	One or more constants separated by "|" such as top, bottom, start, end, center, and many others
android: orientation	Horizontal or vertical
android:layout_margin	Dimension values usually in dp. (sets margin on all 4 sides)
android:layout_marginStart	Dimension values usually in dp. (sets margin on start side)

For more details, Refer

https://developer.android.com/reference/android/widget/LinearLayout.LayoutParams

Task 2: Build UI with Relative Layout (Complete activity_relative_example.xml)

Build the UI as shown:

Tips: Some useful attributes.

Attributes	Description
android:layout_centerHorizontal	If true, centers this child horizontally within its parent
android:layout_centerVertical	If true, centers this child vertically within its parent
android:layout_centerInParent	If true, centers this child horizontally and vertically within its parent.
android:layout_alignParentStart	If true, makes the start edge of this view match the start edge of the parent.
android:layout_above	Positions the bottom edge of this view above the given anchor view ID
android:layout_toStartOf	Positions the end edge of this view to the start of the given anchor view ID

For more details, Refer

https://developer.android.com/reference/android/widget/RelativeLayout.LayoutParams

Task 3: Build UI with Table Layout (Complete activity_table_example.xml)

Build the UI as shown:

A TableLayout consists of TableRow objects, each defining a row.

Add table row to table layout using <TableRow> tag.

Tips: Some useful attributes.

Attributes	Description
android:layout_column	The index of the column in which this child should be.
android:layout_span	Defines how many columns this child should span.

For more details, Refer

https://developer.android.com/reference/android/widget/TableLayout.LayoutParams

https://developer.android.com/reference/android/widget/TableRow.LayoutParams

Task 4: Build UI with Grid Layout (Complete activity_grid_example.xml)

Build the UI as shown:

Tips: Some useful attributes.

Attributes	Description
android:layout_row	The row boundary delimiting the top of the group of cells occupied by this view. (Integer)
android:layout_rowWeight	The relative proportion of vertical space that should be allocated to this view during excess space distribution.
android:layout_rowSpan	The difference between the top and bottom boundaries delimiting the group of cells occupied by this view. (Integer)
android:rowCount	Number of rows

(Similar attributes for column)

For more details, Refer

https://developer.android.com/reference/android/widget/GridLayout.LayoutParams

Task 5: Build UI with Frame Layout (Complete activity_frame_example.xml)

Build UI as shown

1. Change ConstraintLayout to FrameLayout
2. Add ImageView and TextView inside Frame Layout.
3. Set ImageView width and height attributes to match parent (covers the entire screen)
4. Set Background of ImageView to worldmap.jpg as follows (worldmap.jpg is placed under res/drawable)

```
android:background="@drawable/worldmap"
```

5. Position the textView as shown above
6. Set textColor – red, textSize – 24sp, textStyle – bold

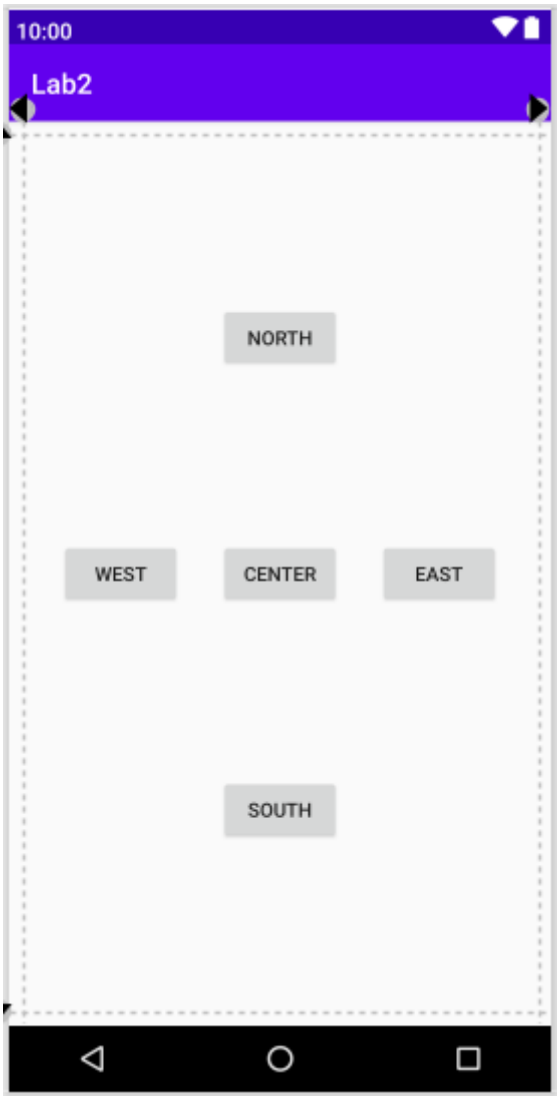
```
android:textColor="@android:color/holo_red_dark"
```

Task 6: Build UI with Constraint Layout (Complete activity_constraint_example.xml)

Guide to build a layout with Constraint layout:

https://developer.android.com/training/constraint-layout

Build UI as shown



1. Add Guidelines at the top, bottom, start and end and set space between screen edge and guideline to 10dp
2. Add Button 'Center' with Centering Positioning constraint on it to place it at the center. Centering Positioning constraint: constraint top to bottom of *top guideline*, bottom to top of *bottom guideline*, start to end of *left guideline*, end to start of *right guideline*. For instance:

```
app:layout_constraintBottom_toTopOf="@id/bottomGuideline"
```

3. Add buttons 'North', 'South', 'East' and 'West' with relative positioning constraints. Relative positioning w.r.t button 'Center' and guidelines.

Tips: Other useful attributes.

Attributes	Description
android:orientation	Horizontal or vertical (to add horizontal or vertical guideline)
app:layout_constraintGuide_begin	specifying a fixed distance from the left or the top of a layout
app:layout_constraintGuide_end	specifying a fixed distance from the right or the bottom of a layout

For more details, Refer

https://developer.android.com/reference/androidx/constraintlayout/widget/ConstraintLayout

https://developer.android.com/reference/androidx/constraintlayout/widget/Guideline

Task 7: Execute & Observe

1. Run the app on an emulator or an Android device.
2. Check whether UIs are as expected.

Deliverables

1. Final code. (File Format: zip)