.

SUKKUR IBA UNIVERSITY

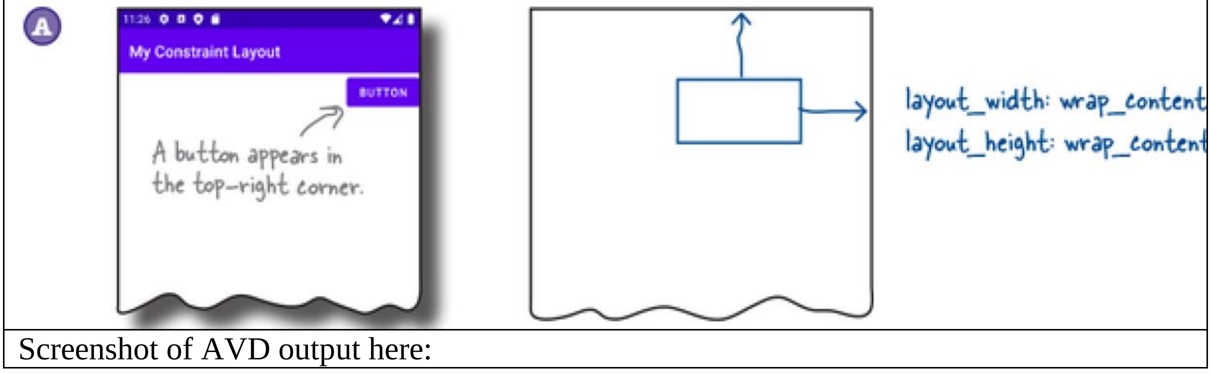
MERIT – QUALITY – EXCELLENCE

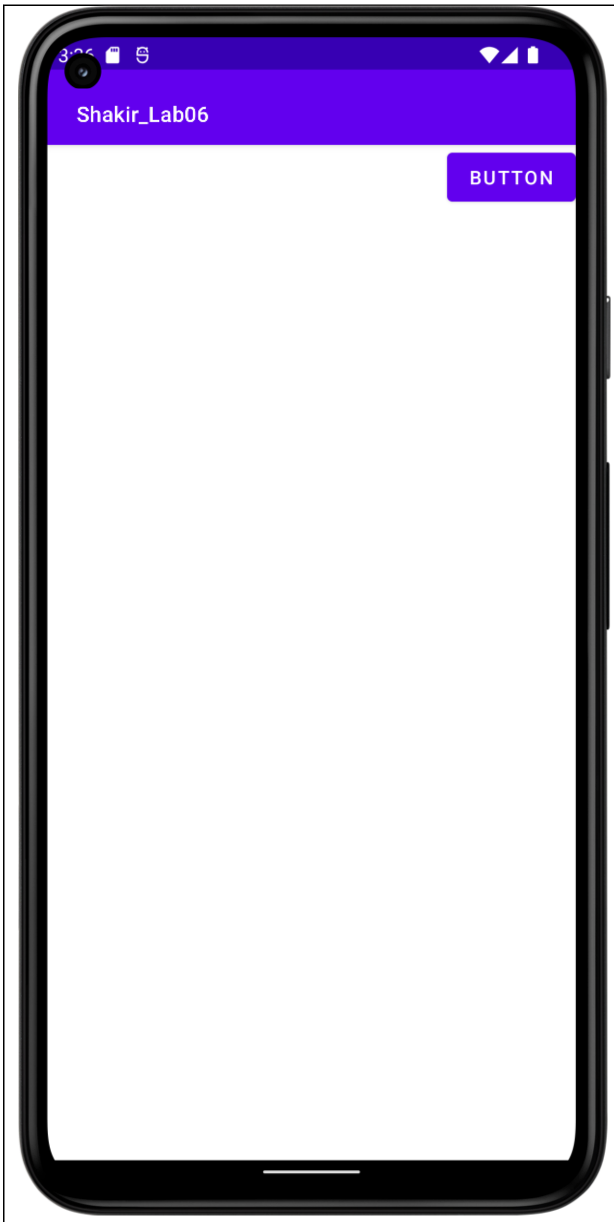
Lab handout 06

Mobile Application Development (CSE-426), Fall 2022, BE-EE(CS)-VIII		
Name: Shakir Ali	CMS ID#: 031-19-0009	Instructor: Dr. Abdul Aziz
Section: A Lab group: NA	Department: Electrical Engineering	Marks obtained out of 100%
NOTE: Must follow submission instructions		

Understanding and using constraint layout

Lab Exercise and Submission

Exercise 1
Develop following layout using constraint layout and mentioned attributes in the figure.  A 11:26 My Constraint Layout BUTTON A button appears in the top-right corner. layout_width: wrap_content layout_height: wrap_content
Screenshot of AVD output here:

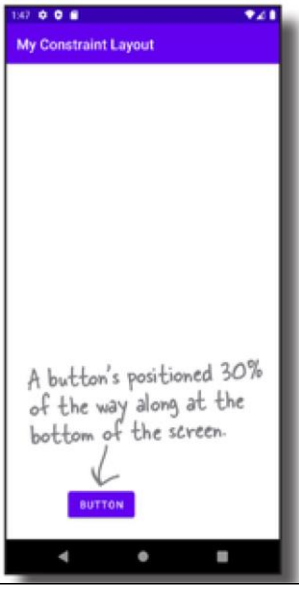


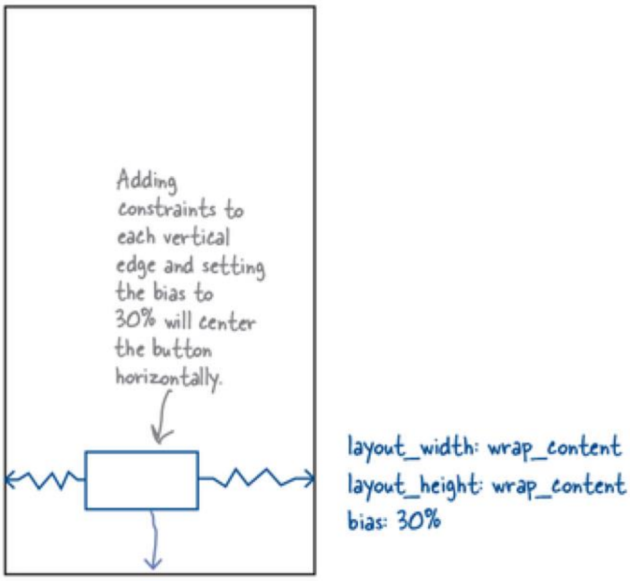
Insert XML codes here:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">


    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

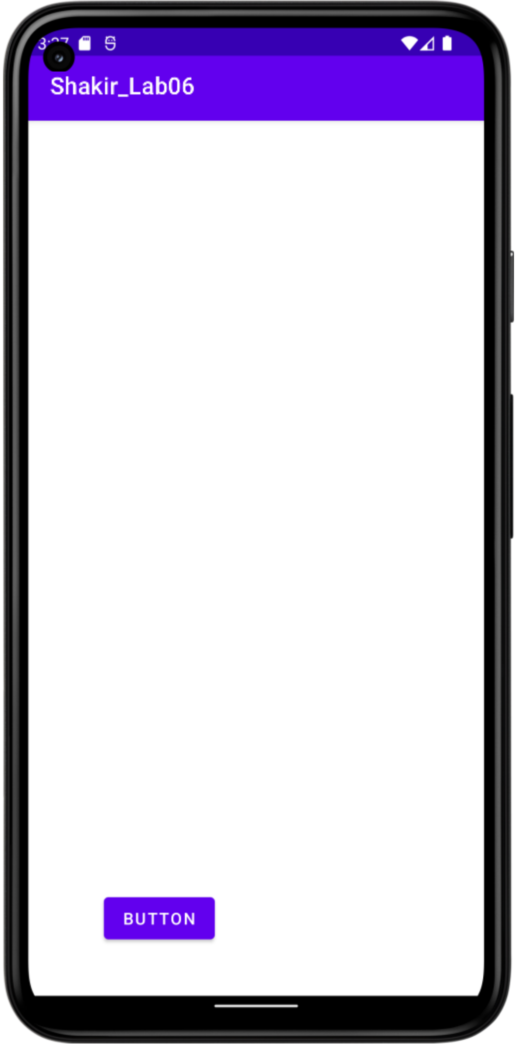
Exercise 2

Develop following layout using constraint layout and mentioned attributes in the figure.


My Constraint Layout
A button's positioned 30% of the way along at the bottom of the screen.
BUTTON


Adding constraints to each vertical edge and setting the bias to 30% will center the button horizontally.
layout_width: wrap_content
layout_height: wrap_content
bias: 30%

Screenshot of AVD output here:

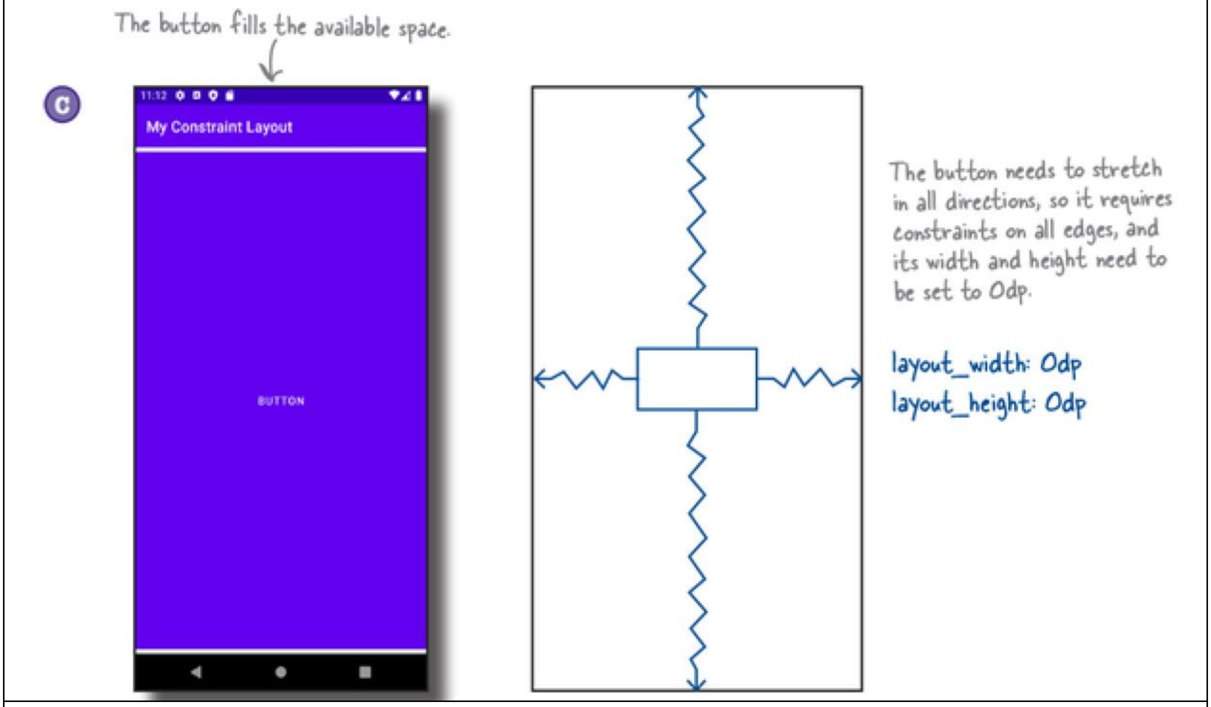


Insert XML codes here:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">


    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="664dp"
        android:layout_marginEnd="232dp"
        android:text="Button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Exercise 3

Develop following layout using constraint layout and mentioned attributes in the figure.

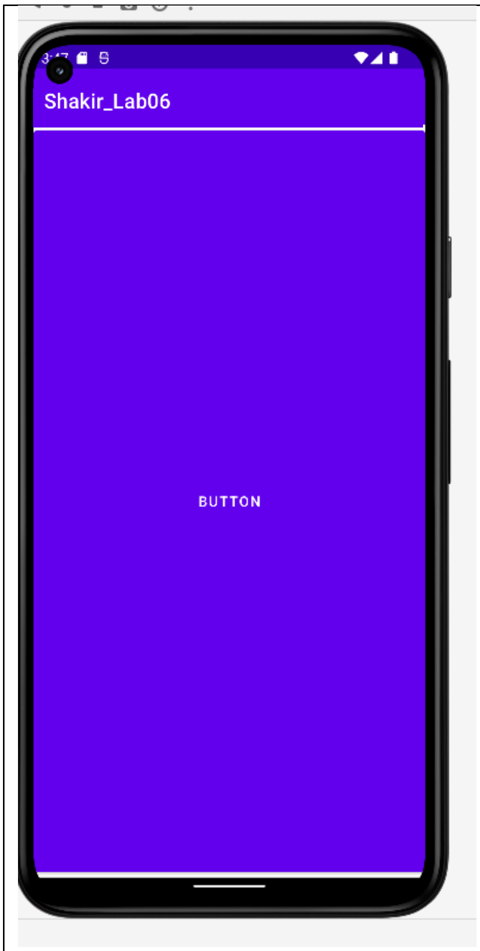


The button fills the available space.

The button needs to stretch in all directions, so it requires constraints on all edges, and its width and height need to be set to 0dp.

layout_width: 0dp
layout_height: 0dp

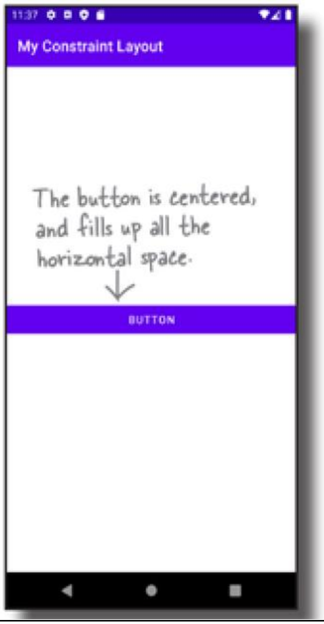
Screenshot of AVD output here:



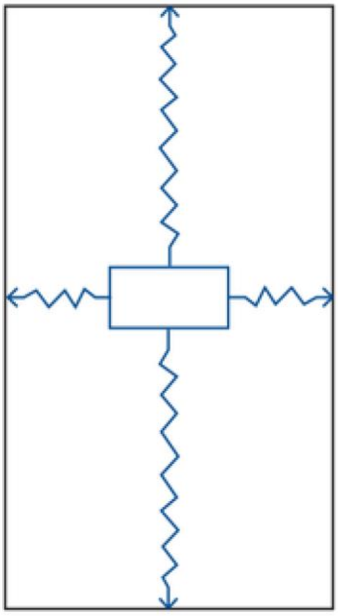
Insert XML codes here:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp"
    android:orientation="vertical"
    tools:context=".MainActivity">
  <Button
    android:id="@+id/button"
    android:layout_width="match_parent"
    android:layout_height=" match_parent "
    android:text="Button"/>
</LinearLayout>
```

Exercise 4
Develop following layout using constraint layout and mentioned attributes in the figure.



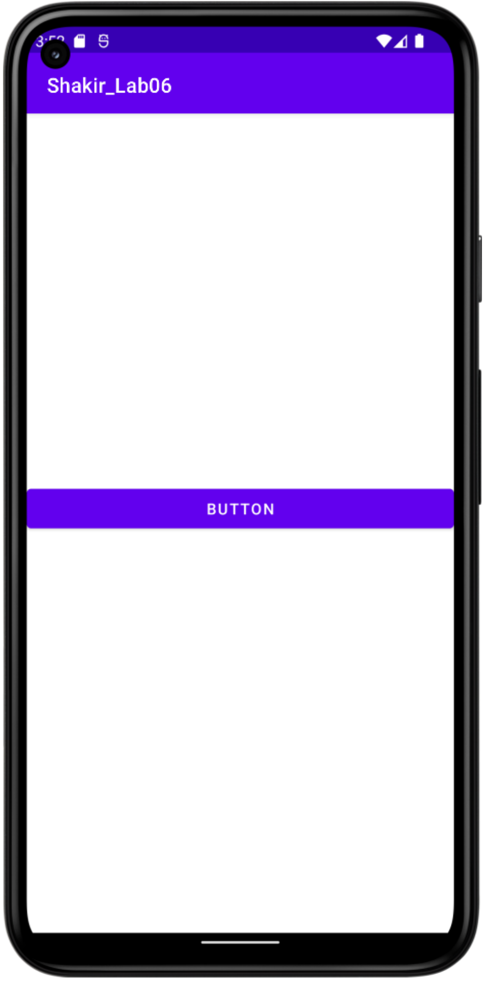
The button is centered, and fills up all the horizontal space.



layout_width: 0dp
layout_height: wrap_content

The button needs to stretch horizontally and be centered vertically. It therefore needs constraints on all edges as for C, but its width is set to 0dp, and its height is set to wrap_content.

Screenshot of AVD output here:

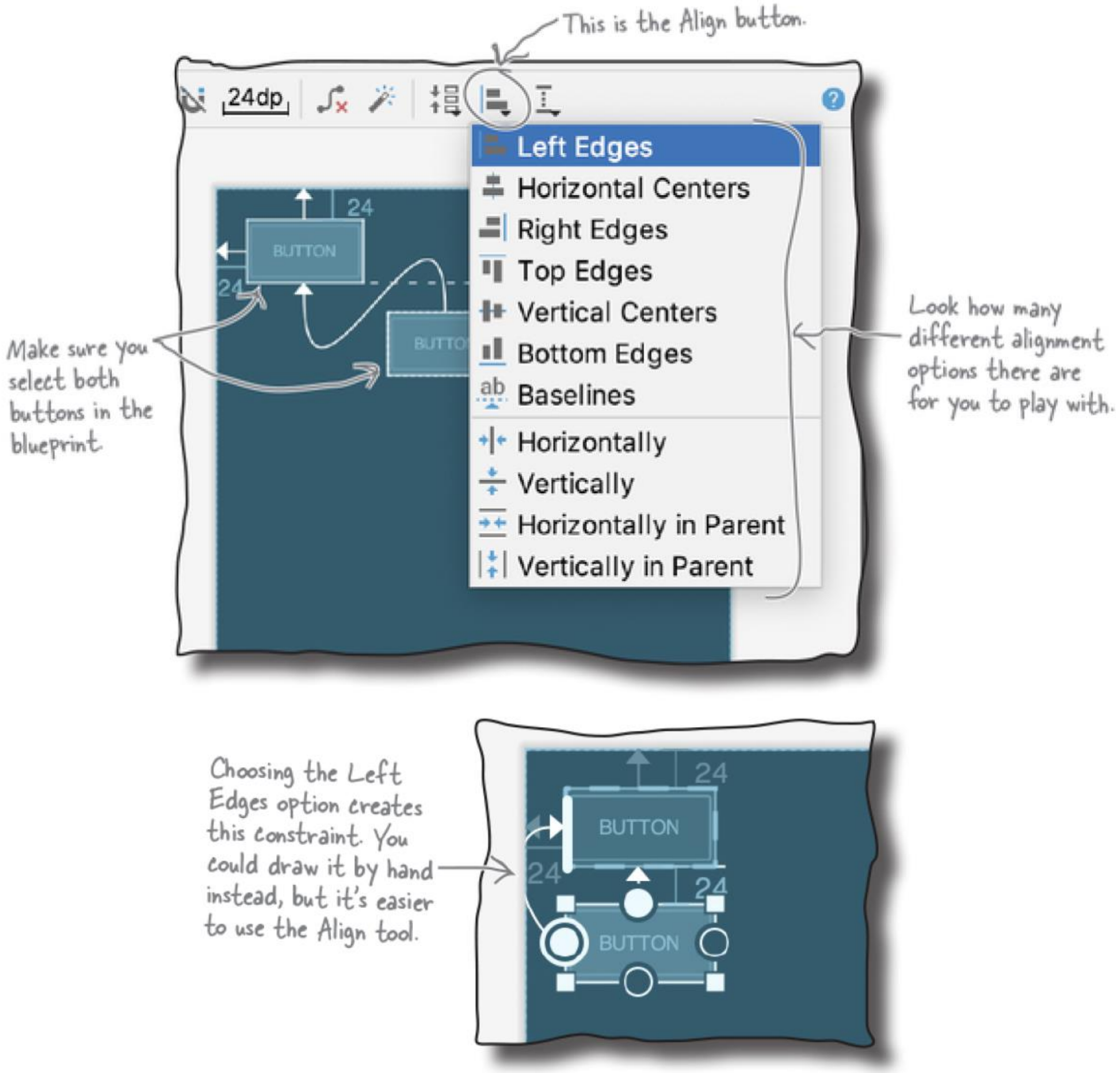


Insert XML codes here:

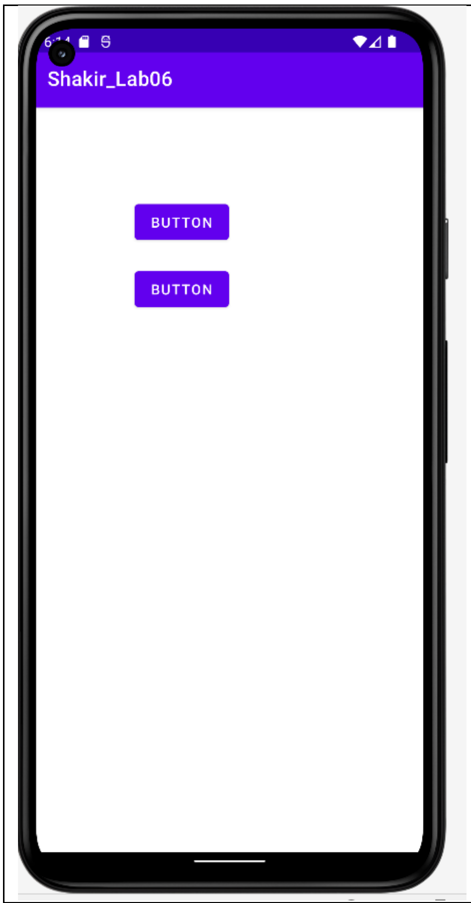
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <Button
        android:id="@+id/button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="340dp"
        android:layout_weight="1"
        android:text="Button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Exercise 5

Using constraint layout explore different alignment options and submit obtained results.


This is the Align button.
24dp
Left Edges
Horizontal Centers
Right Edges
Top Edges
Vertical Centers
Bottom Edges
Baselines
Horizontally
Vertically
Horizontally in Parent
Vertically in Parent
BUTTON
24
Make sure you select both buttons in the blueprint.
Look how many different alignment options there are for you to play with.
Choosing the Left Edges option creates this constraint. You could draw it by hand instead, but it's easier to use the Align tool.
BUTTON
24
BUTTON

Attach AVD application output here:

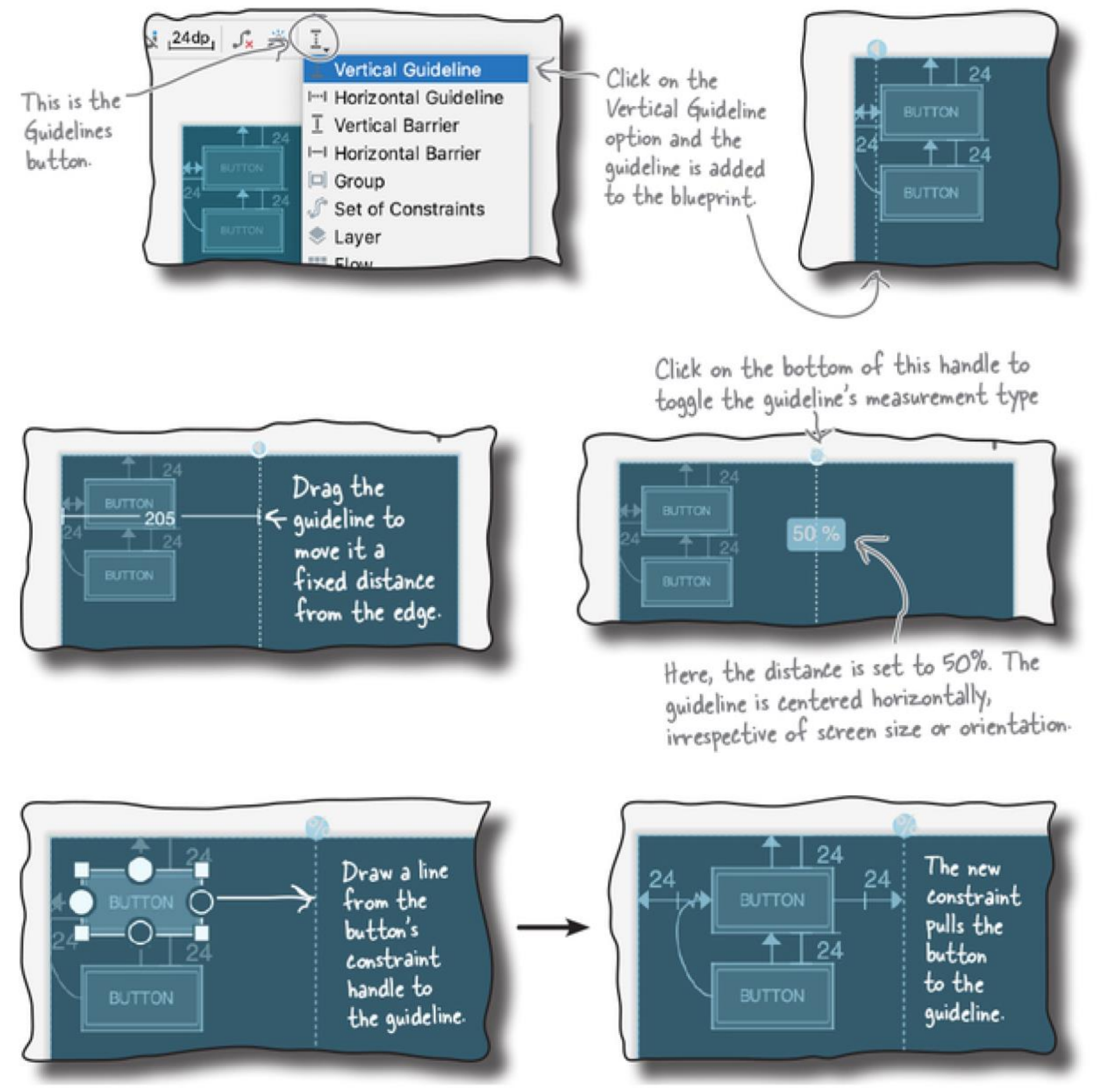


Insert XML file code here:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="100dp"
        android:layout_marginTop="92dp"
        android:text="Button"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:text="Button"
        app:layout_constraintStart_toStartOf="@+id/button"
        app:layout_constraintTop_toBottomOf="@+id/button" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Exercise 6

Align and partition views using guidelines.



This is the Guidelines button.

Click on the Vertical Guideline option and the guideline is added to the blueprint.

Drag the guideline to move it a fixed distance from the edge.

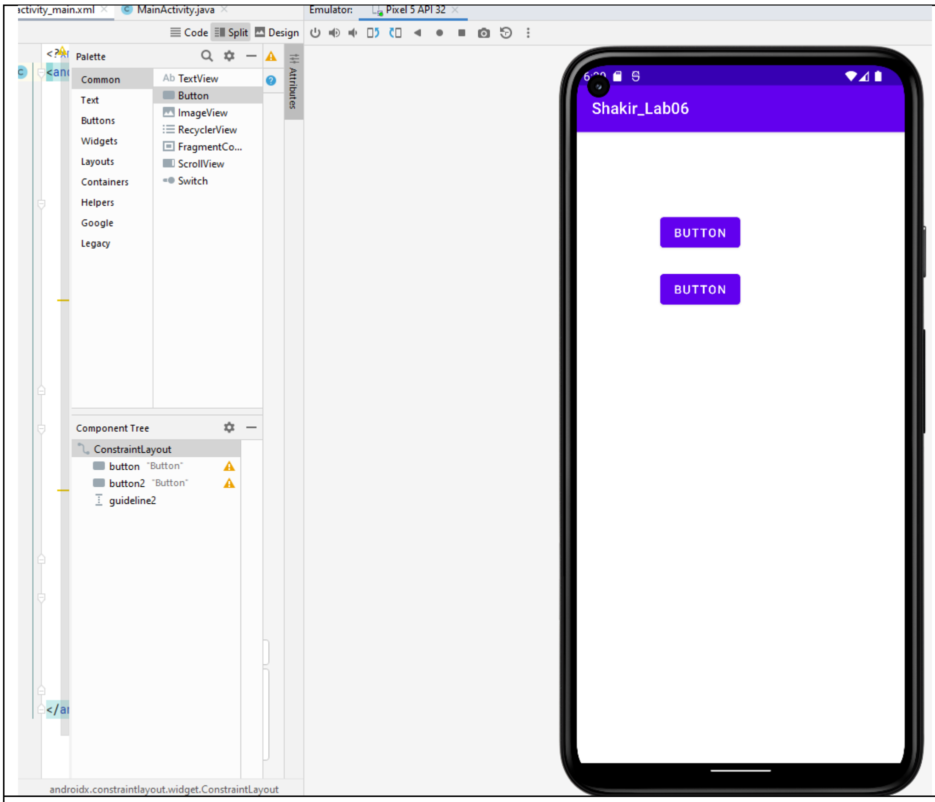
Click on the bottom of this handle to toggle the guideline's measurement type

Here, the distance is set to 50%. The guideline is centered horizontally, irrespective of screen size or orientation.

Draw a line from the button's constraint handle to the guideline.

The new constraint pulls the button to the guideline.

Attach AVD application output here:

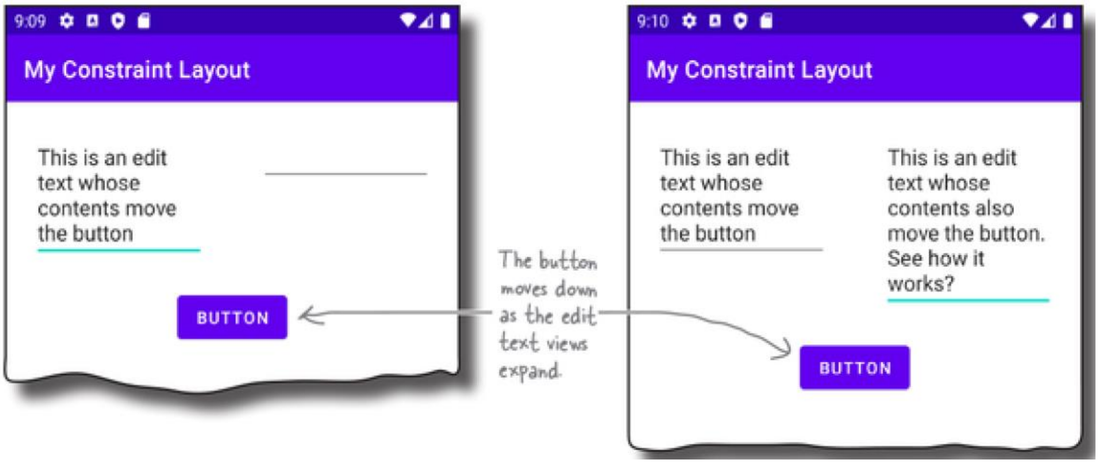


Insert XML file code here:

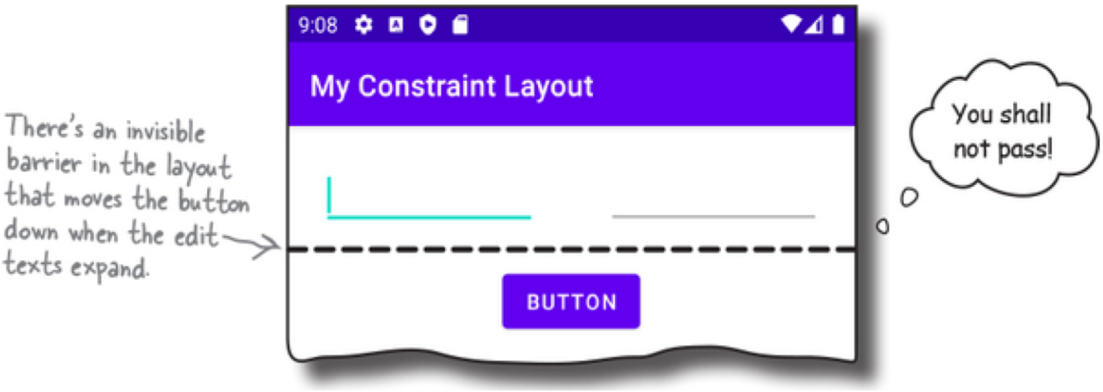
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="100dp"
        android:layout_marginTop="96dp"
        android:text="Button"
        app:layout_constraintEnd_toStartOf="@+id/guideline2"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:text="Button"
        app:layout_constraintStart_toStartOf="@+id/button"
        app:layout_constraintTop_toBottomOf="@+id/button" />

    <androidx.constraintlayout.widget.Guideline
        android:id="@+id/guideline2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        app:layout_constraintGuide_begin="226dp" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Exercise 7

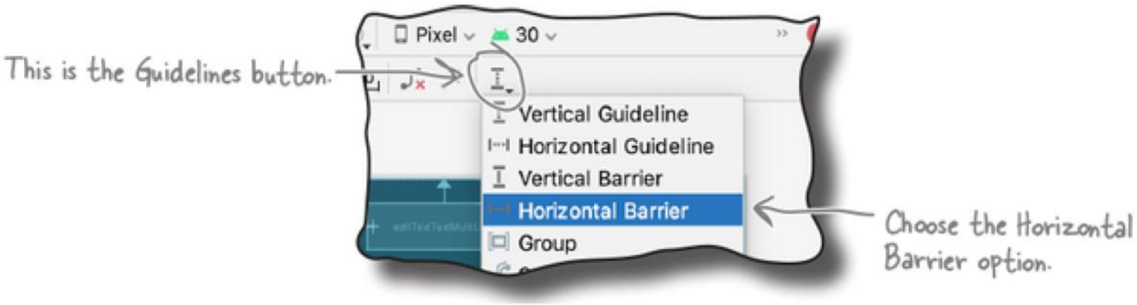
Make use of horizontal barrier to let be button go down while multiline text is taking more space.


9:09
My Constraint Layout
This is an edit text whose contents move the button
BUTTON
9:10
My Constraint Layout
This is an edit text whose contents move the button
This is an edit text whose contents also move the button. See how it works?
BUTTON

The button moves down as the edit text views expand.

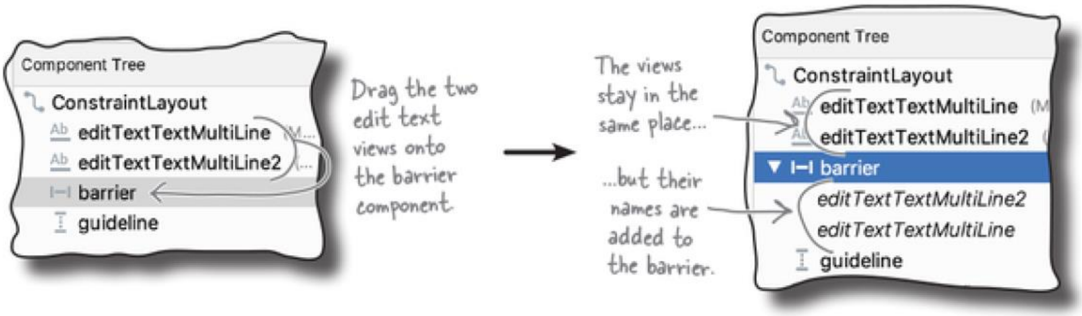


There's an invisible barrier in the layout that moves the button down when the edit texts expand.



This is the Guidelines button.

Choose the Horizontal Barrier option.

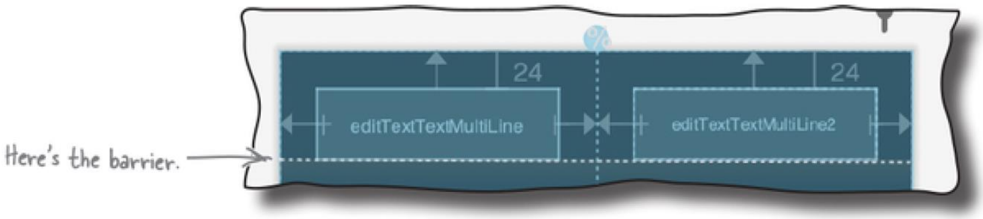


Drag the two edit text views onto the barrier component.

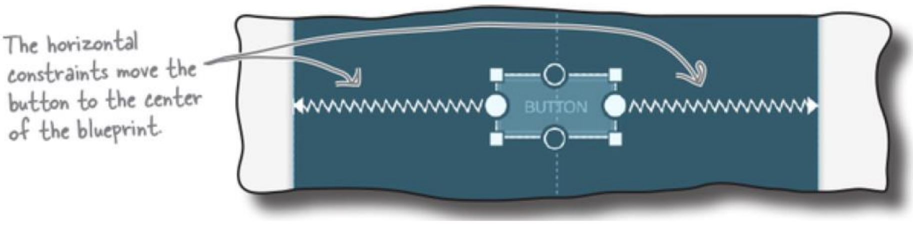
The views stay in the same place...

...but their names are added to the barrier.

Make sure that `app:barrierDirection="bottom"`

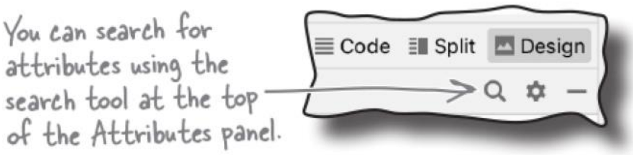


Here's the barrier.

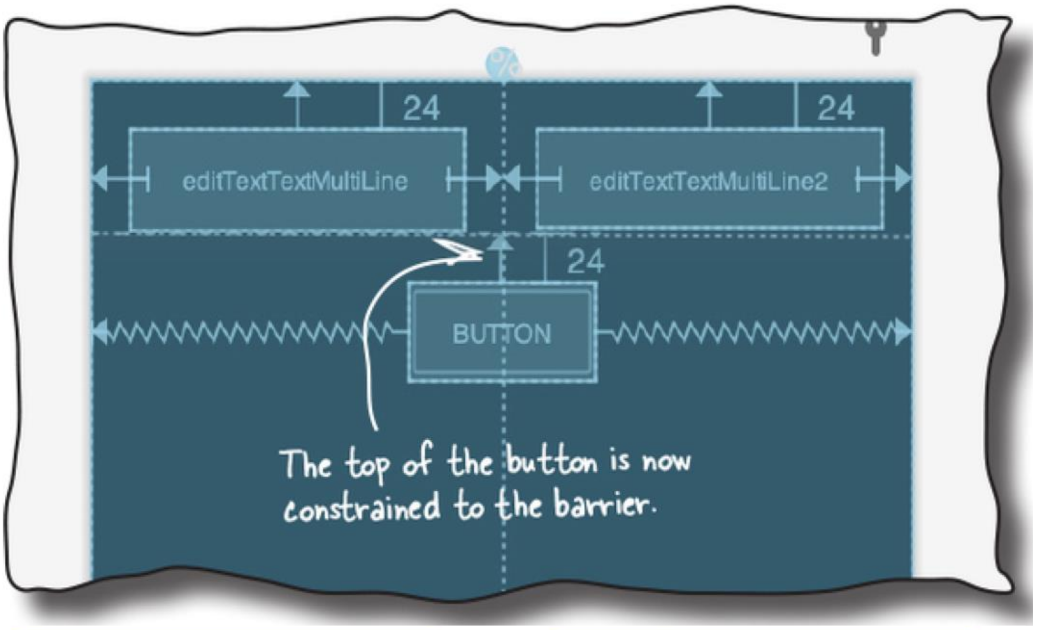


The horizontal constraints move the button to the center of the blueprint.

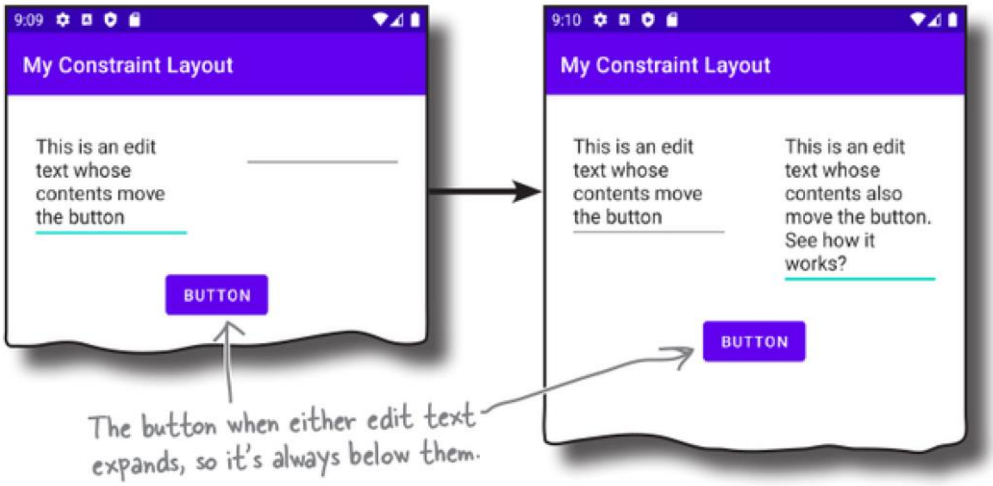
Search for button's `layout_constraintTop_toBottomOf="@id/barrier"` attribute in the Attributes panel, and change its value to the barrier's ID



You can search for attributes using the search tool at the top of the Attributes panel.



The top of the button is now constrained to the barrier.



My Constraint Layout

This is an edit text whose contents move the button

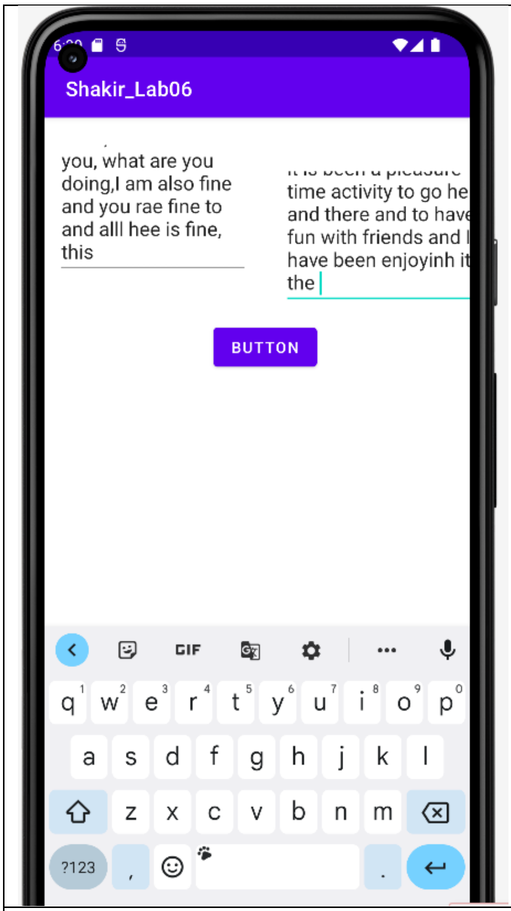
My Constraint Layout

This is an edit text whose contents move the button

This is an edit text whose contents also move the button. See how it works?

The button when either edit text expands, so it's always below them.

Screenshot of AVD output here:



Shakir_Lab06

you, what are you doing,I am also fine and you rae fine to and alll hee is fine, this

it is been a pleasure time activity to go he and there and to have fun with friends and I have been enjoyinh it the

BUTTON

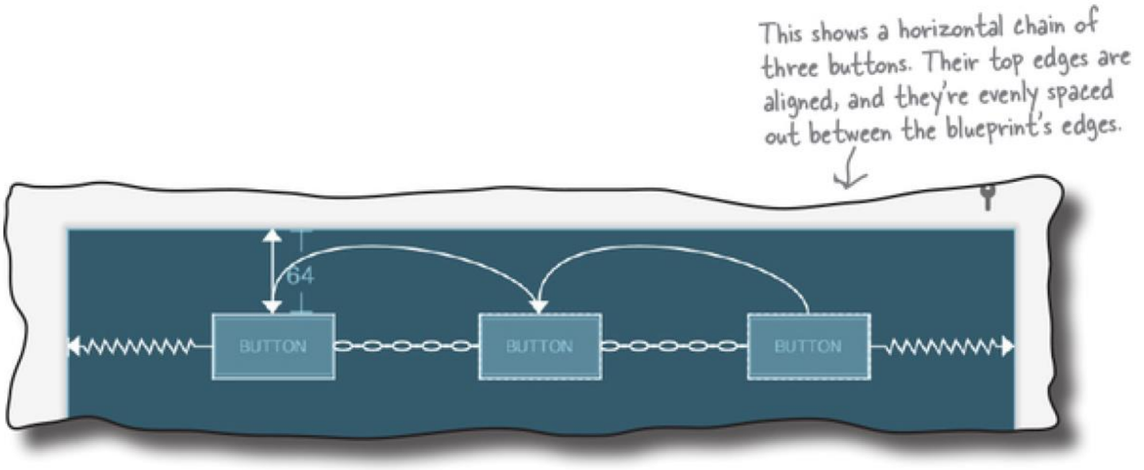
Insert XML file code here:

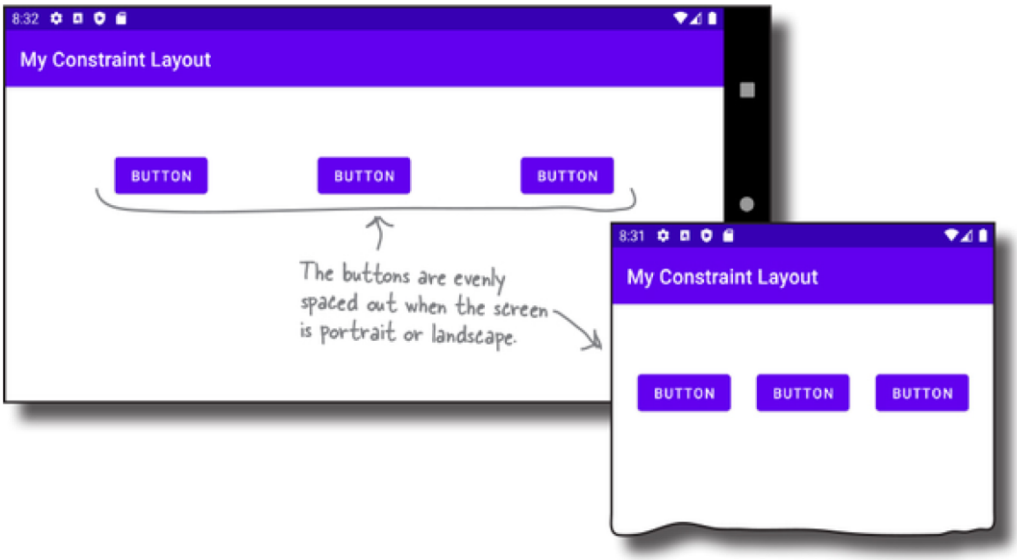
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/editTextTextMultiLine"
        android:layout_width="177dp"
        android:layout_height="131dp"
        android:ems="10"
        android:gravity="start|top"
        android:inputType="textMultiLine"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@+id/guideline3"
        app:layout_constraintHorizontal_bias="0.296"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.025" />

    <Button
        android:id="@+id/button4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="156dp"
        android:layout_marginTop="188dp"
        android:text="Button"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
```

```
        android:id="@+id/editTextTextMultiLine2"
        android:layout_width="202dp"
        android:layout_height="135dp"
        android:layout_marginStart="4dp"
        android:layout_marginBottom="580dp"
        android:ems="10"
        android:gravity="start|top"
        android:inputType="textMultiLine"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="@+id/guideline3" />

    <androidx.constraintlayout.widget.Barrier
        android:id="@+id/barrier2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:barrierDirection="bottom"

app:constraint_referenced_ids="editTextTextMultiLine2,editTextTextMultiLi
ne"
        tools:layout_editor_absoluteY="103dp" />

    <androidx.constraintlayout.widget.Guideline
        android:id="@+id/guideline3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        app:layout_constraintGuide_begin="216dp" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Exercise 8

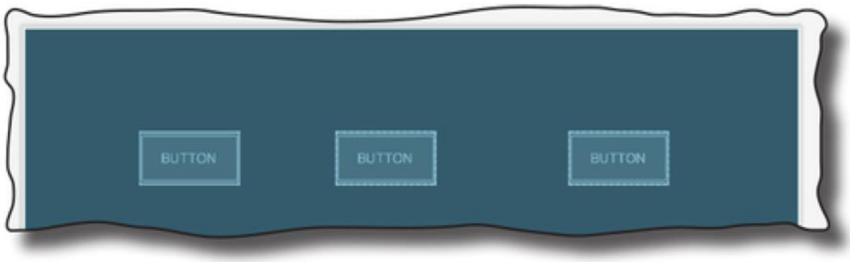
Create horizontal chain for views in constraint layout.

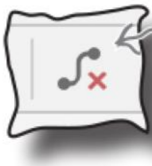

This shows a horizontal chain of three buttons. Their top edges are aligned, and they're evenly spaced out between the blueprint's edges.



The buttons are evenly spaced out when the screen is portrait or landscape.

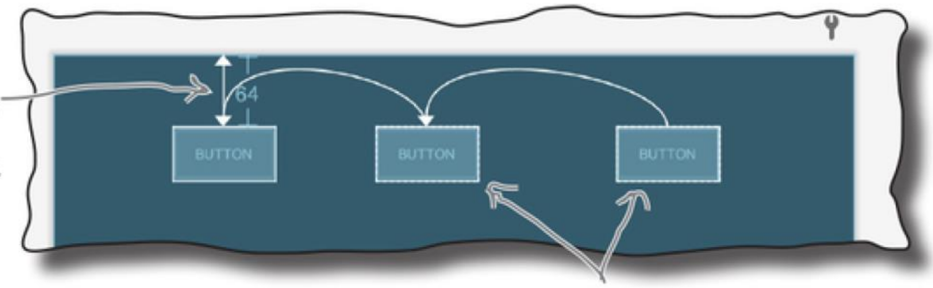
Make sure your blueprint includes three buttons, and no constraints or guidelines.





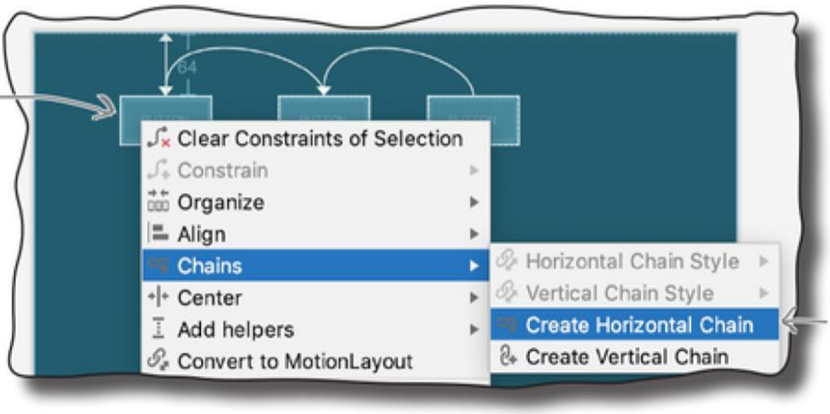
Use this button to clear all of the layout's constraints and start from scratch.

This constraint has its margin set to 64.

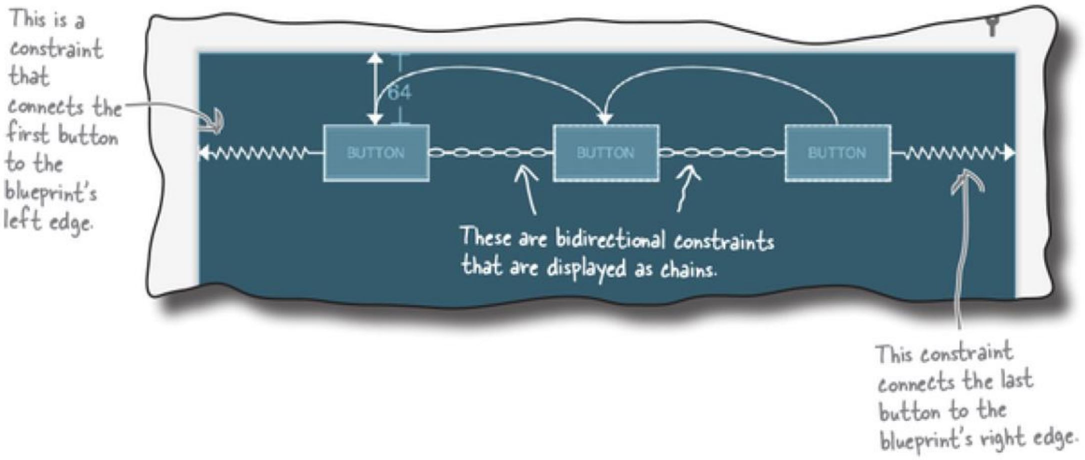


The second two buttons are aligned to the first.

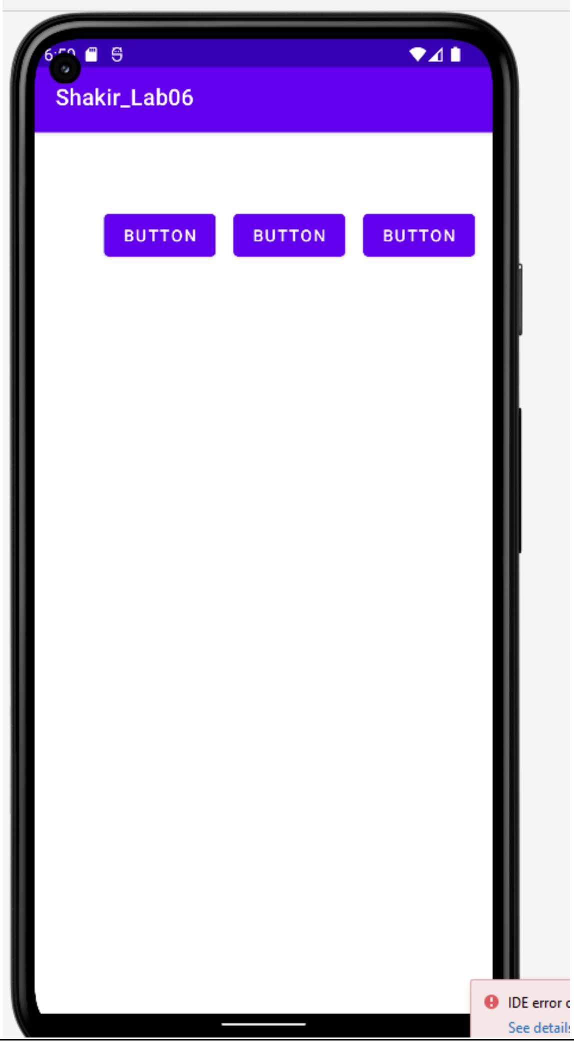
Select the views you want to chain, then right-click on one of them.



Choose the option to create a horizontal chain.


This is a constraint that connects the first button to the blueprint's left edge.
64
BUTTON
BUTTON
BUTTON
These are bidirectional constraints that are displayed as chains.
This constraint connects the last button to the blueprint's right edge.

Attach AVD application output here:

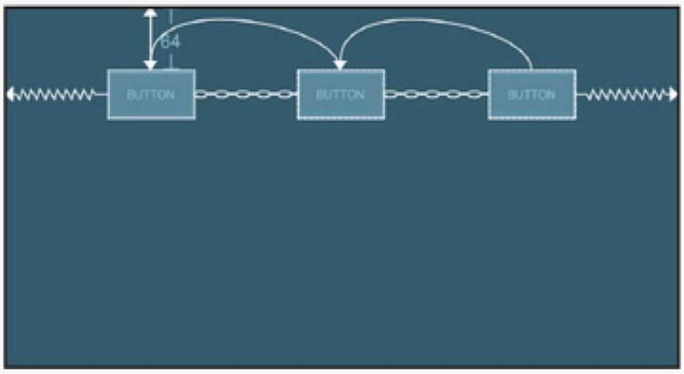


Insert XML file code here:

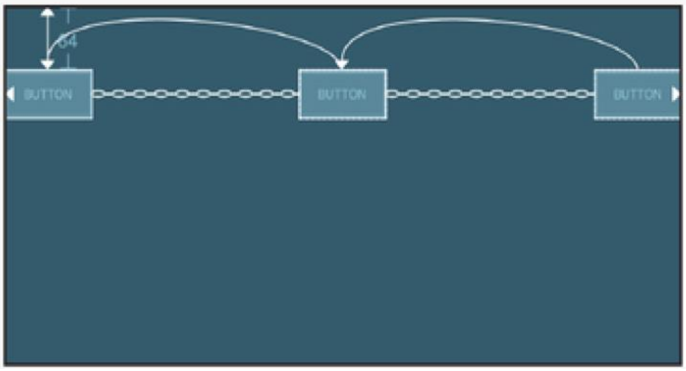
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/button5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="44dp"
        android:layout_marginTop="64dp"
        android:text="Button"
        app:layout_constraintEnd_toStartOf="@+id/button6"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        app:layout_constraintEnd_toStartOf="@+id/button7"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/button5"
        app:layout_constraintTop_toTopOf="@+id/button5" />

    <Button
        android:id="@+id/button7"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/button6"
        app:layout_constraintTop_toTopOf="@+id/button6" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Exercise 8

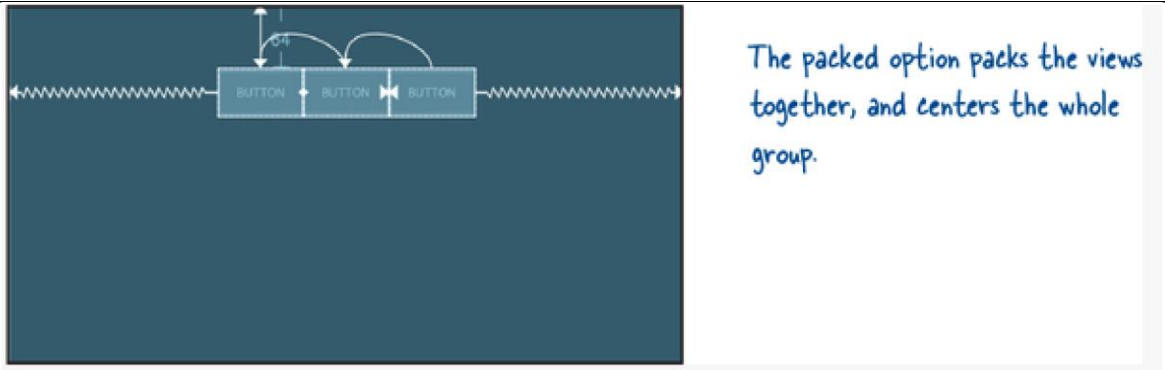
In the above horizontal chained constraint layout, right-click on one of the buttons in the horizontal chain, select the Chains option from the menu that appears, and then choose Horizontal Chain Style. What happens to the views when you choose (i) Spread (ii) Spread inside (iii) Packed? Draw the blueprint for each option below, and describe how it arranges the views.



The spread option arranges views so that they are evenly spaced out between the blueprint's edges.

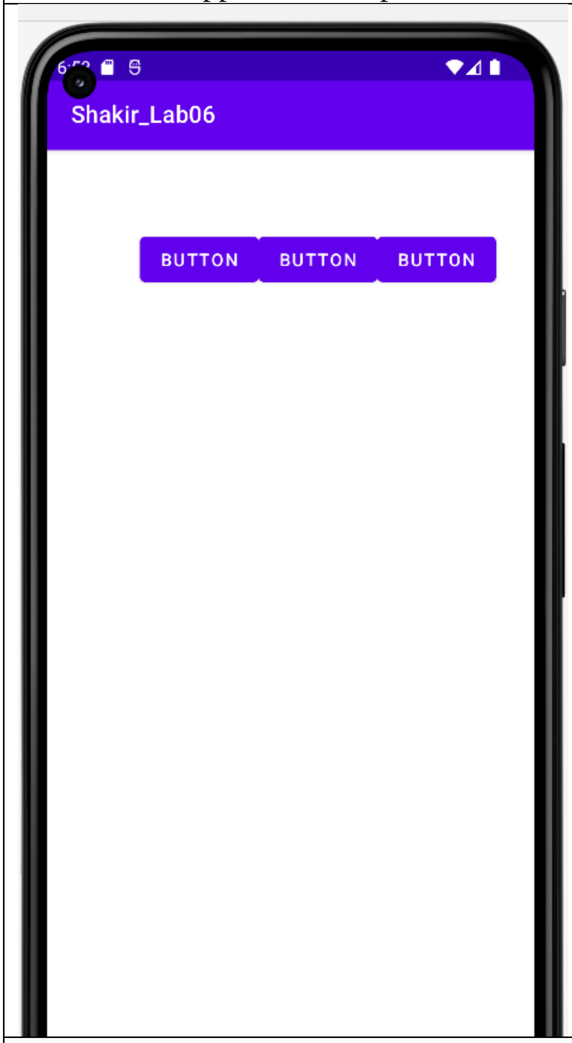


The spread inside option moves the left- and rightmost views to the blueprint's edges, then evenly spaces out the remaining views.



The packed option packs the views together, and centers the whole group.

Attach AVD application output here:



Insert XML file code here:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/button5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="44dp"
        android:layout_marginTop="64dp"
        android:text="Button"
        app:layout_constraintEnd_toStartOf="@+id/button6"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintHorizontal_chainStyle="packed"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        app:layout_constraintEnd_toStartOf="@+id/button7"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/button5"
        app:layout_constraintTop_toTopOf="@+id/button5" />

    <Button
        android:id="@+id/button7"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toEndOf="@+id/button6"
        app:layout_constraintTop_toTopOf="@+id/button6" />
</androidx.constraintlayout.widget.ConstraintLayout>
```