

# GridLayout

- Layout that places its children in a **rectangular grid**.
- The grid is composed of a set of infinitely thin lines that separate the viewing area into cells.
- In **Android GridLayout**, we can specify the number of columns and rows that the grid will have.
- The number of rows and columns within the grid can be declared using the `android:rowCount` and `android:columnCount` properties.
- Typically, however, if the number of columns is declared the GridLayout will infer the number of rows based on the number of occupied cells making the use of the `rowCount` property unnecessary.
- Similarly, the orientation of the GridLayout may optionally be defined via the `android:orientation` property.
- The following example XML declares a 2 x 2 GridLayout configuration in horizontal orientation:

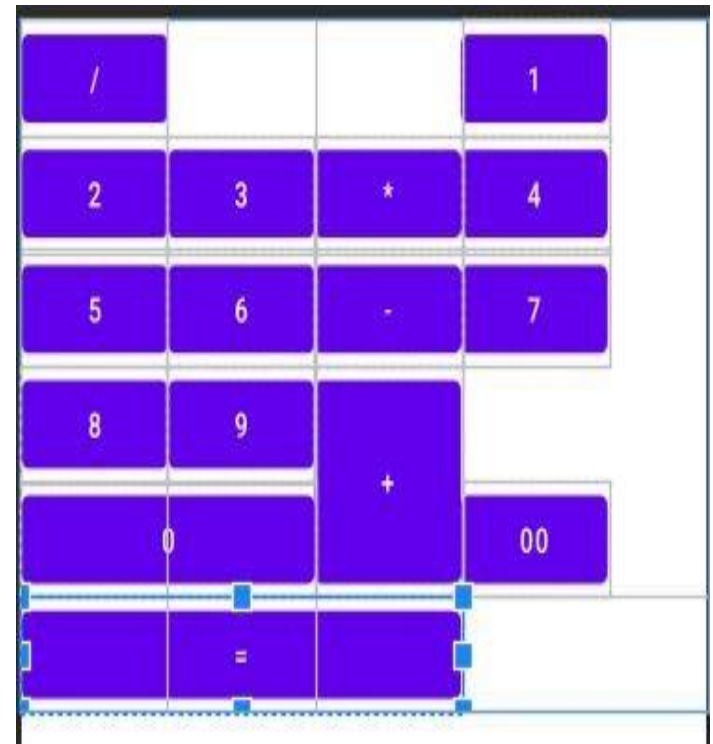
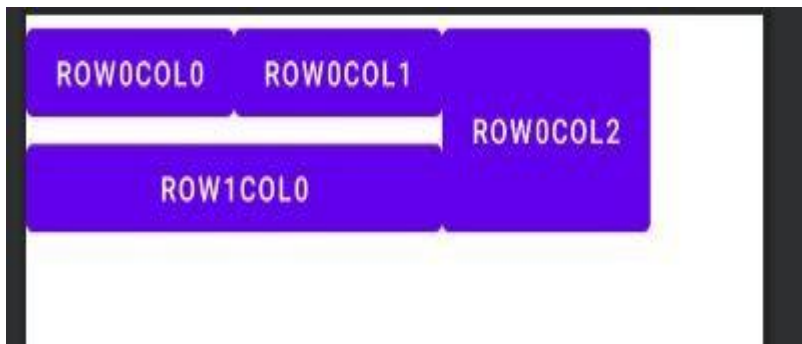


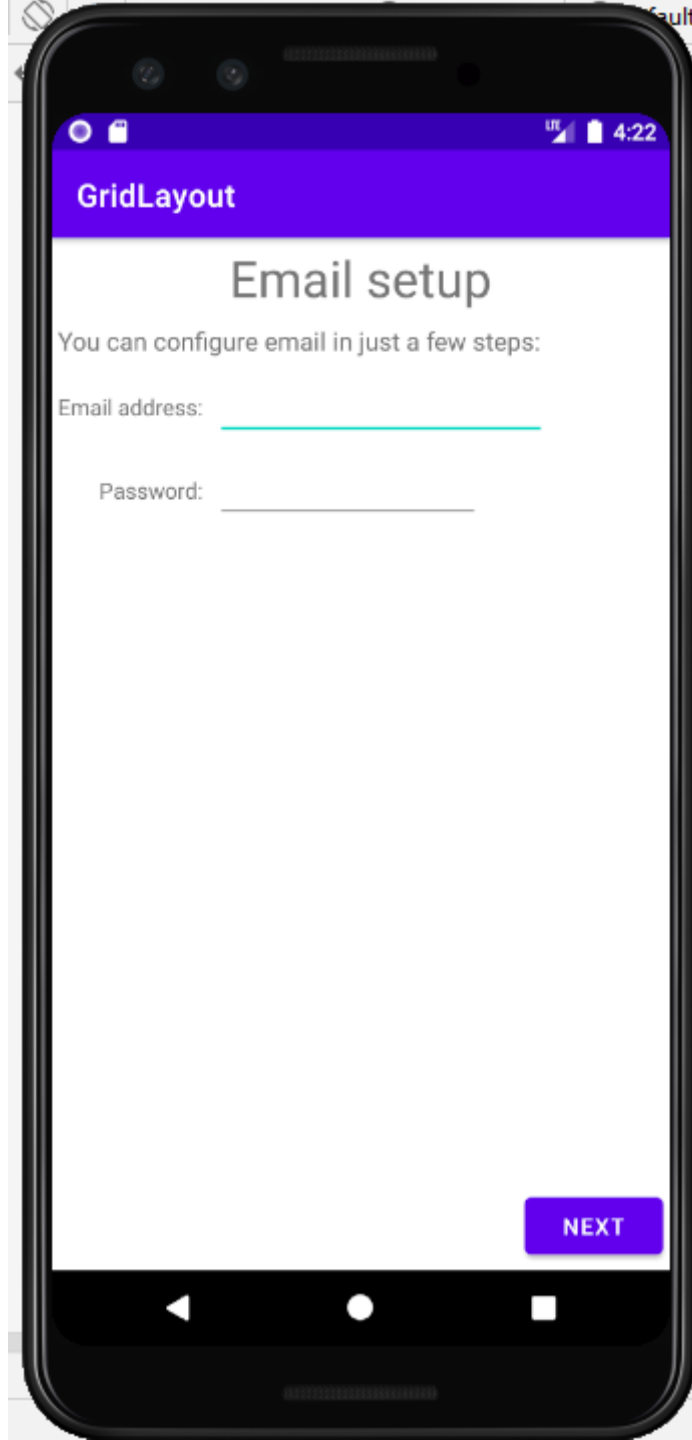
HELLO	GRIDLAYOUT	
ROW21	ROW31	

- A view can be placed within a specific cell by specifying the intersecting row and column number of the destination cell. The following Button view will be placed in the cell and row 1, column 2 of the parent GridLayout:

```
<Button
    android:id="@+id/button5"
    android:layout_column="2"
    android:layout_row="1"
    android:layout_gravity="left|top"
    android:text="Button" />
```

- `rowSpan` or `colspan` ?





```
<?xml version="1.0" encoding="utf-8"?>
<GridLayout 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"
    android:useDefaultMargins="true"
    android:alignmentMode="alignBounds"
    android:columnOrderPreserved="false"
    android:columnCount="4"
    tools:context=".MainActivity">

    <TextView
        android:text="Email setup"
        android:textSize="32dip"

        android:layout_columnSpan="4"
        android:layout_gravity="center_horizontal"
    />

    <TextView
        android:text="You can configure email in just a few steps:"
        android:textSize="16dip"

        android:layout_columnSpan="4"
        android:layout_gravity="left"
    />
```

```
<TextView
    android:text="Email address:"

    android:layout_gravity="right" />

<EditText
    android:ems="10" />

<TextView
    android:text="Password:"
    android:layout_column="0"
    android:layout_gravity="right" />

<EditText
    android:ems="8" />

<Space
    android:layout_row="4"
    android:layout_column="0"
    android:layout_columnSpan="3"
    android:layout_gravity="fill" />

<Button
    android:text="Next"
    android:layout_row="5"
    android:layout_column="3" />

</GridLayout>
```

# Attributes

## **ColumnCount**

- ColumnCount is used only to generate default column/column indices when they are not specified by a component's layout parameters.

# Attributes

## **RowCount**

- RowCount is used only to generate default row/column indices when they are not specified by a component's layout parameters.

# Attributes

## **RowOrderPreserved**

- When this property is true, GridLayout is forced to place the row boundaries so that their associated grid indices are in ascending order in the view.
- When this property is false GridLayout is at liberty to place the vertical row boundaries in whatever order best fits the given constraints.
- The default value of this property is true.



# Attributes

## **ColumnOrderPreserved**

- When this property is true, GridLayout is forced to place the column boundaries so that their associated grid indices are in ascending order in the view.
- When this property is false GridLayout is at liberty to place the horizontal column boundaries in whatever order best fits the given constraints.
- The default value of this property is true.

# Attributes

## UseDefaultMargins

- When true, GridLayout allocates default margins around children based on the child's visual characteristics.
- Each of the margins so defined may be independently overridden by an assignment to the appropriate layout parameter.
- When false, the default value of all margins is zero.
- When setting to true, consider setting the value of the **alignmentMode** property to ALIGN\_BOUNDS.
- The default value of this property is false.

# Attributes

## AlignmentMode

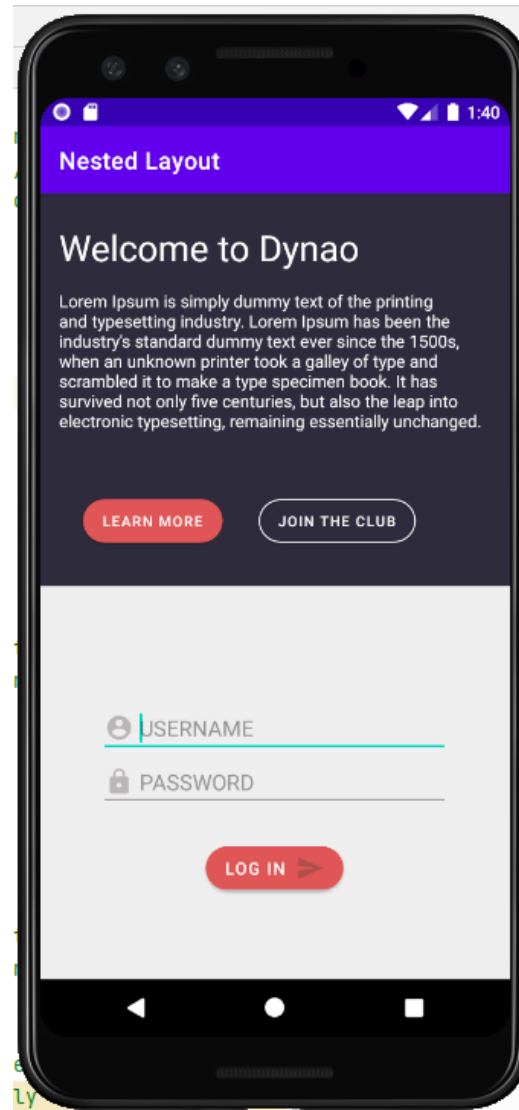
- Sets the alignment mode to be used for all of the alignments between the children of this container.
- The default value of this property is `ALIGN_MARGINS`
- **ALIGN\_MARGINS**
  - When the alignmentMode is set to `ALIGN_MARGINS`, the bounds of each view are extended outwards, according to their margins, before the edges of the resulting rectangle are aligned.
- **ALIGN\_BOUNDS**
  - When the alignmentMode is set to `ALIGN_BOUNDS`, alignment is made between the edges of each component's raw view boundary: i.e. the area delimited by the component's: top, left, bottom and right properties.

## Table layout vs Grid layout

- If the amount of data is low, fixed and don't require scrolling then `TableLayout` should be used but if the data is large and require scrolling to access, `GridLayout` with `ScrollView` should be used.

# Nested Layout

- Design the following screen



Activity\_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
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"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingLeft="16dp"
        android:paddingRight="16dp"
        android:layout_weight="0.5"
        android:background="#2f2c3d">
        <TextView
            android:id="@+id/welcomeText"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="25dp"
            android:textSize="30dp"
            android:textColor="#FFF"
            android:text="Welcome to Dynao"
        />
    </RelativeLayout>
</LinearLayout>
```

<TextView

```
    android:id="@+id/introText"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#FFF"
    android:paddingTop="15dp"
    android:layout_below="@+id/welcomeText"
    android:text="Lorem Ipsum is simply dummy text
```

of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged."

/>

<Button

```
    android:id="@+id/primaryButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/introText"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="50dp"
    android:text="LEARN MORE"
    android:textSize="12sp"
    app:backgroundTint="#e05555"
    app:cornerRadius="24dp" />
```

<Button

```
    android:id="@+id/secondButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/introText"
    android:layout_marginLeft="30dp"
    android:layout_marginTop="50dp"
    android:layout_toRightOf="@+id/primaryButton"
    android:textSize="12sp"
    android:text="JOIN THE CLUB"
    app:backgroundTint="@android:color/transparent"
    app:cornerRadius="24dp"
    app:strokeColor="#FFF"
    app:strokeWidth="1dp" />
```

</RelativeLayout>

<RelativeLayout

```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_weight="0.5"
    android:paddingLeft="50dp"
    android:paddingRight="50dp"
    android:background="#eeeeee"
    android:orientation="vertical">
```



```
<EditText
    android:id="@+id/username"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_above="@+id/password"
    android:drawableLeft="@drawable/ic_baseline_account_circle_24"
    android:drawablePadding="5dp"
    android:hint="USERNAME" />
<EditText
    android:id="@+id/password"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true"
    android:drawableLeft="@drawable/ic_baseline_lock_24"
    android:drawablePadding="5dp"
    android:hint="PASSWORD" />
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="25dp"
    android:drawableRight="@drawable/ic_baseline_send_24"
    android:drawableEnd="@drawable/ic_baseline_send_24"
    android:drawablePadding="5dp"
    android:layout_below="@+id/password"
    app:backgroundTint="#e05555"
    app:cornerRadius="24dp"
    android:text="Log In" />
```

```
</RelativeLayout>
```

```
</LinearLayout>
```