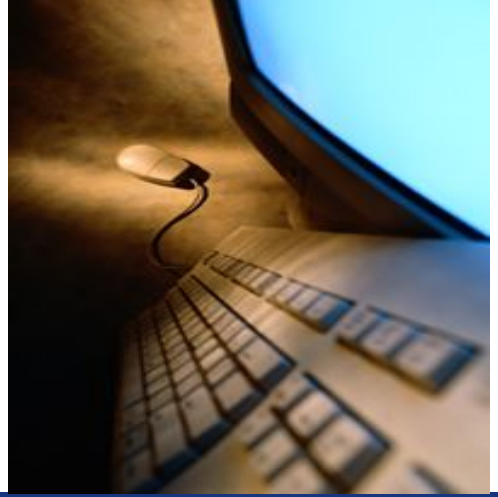
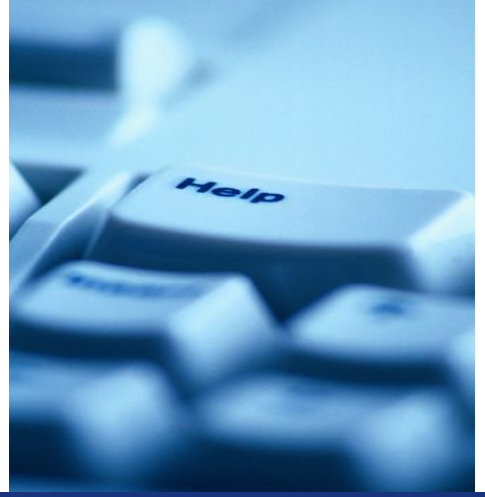
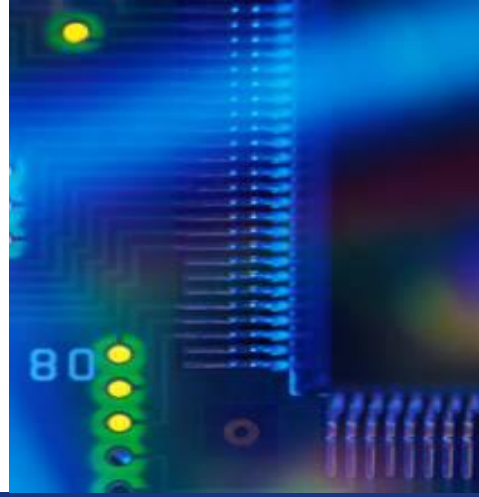
UNIVERSITAS
BUDI LUHUR



FAKULTAS TEKNOLOGI INFORMASI

Mobile Programming

[PG119/ 3 SKS]

Riskiana Wulan, MKom


Activity Recap

- **An *activity* is the entry point for interacting with the user. It represents a single screen with a user interface.**

The Activity which you want it to be the very first screen if your app is opened, then mention it as LAUNCHER in the intent category and remaining activities mention Default in intent category.

Responsibilities:

- Managing the lifecycle (e.g., onCreate(), onPause(), onResume(), etc.).
- Handling navigation between different screens or other activities.
- Hosting views (like buttons, text fields) and responding to user input.

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.StudentApp"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

MainActivity.java

Here you can see the MainActivity is bind to activity_main layout

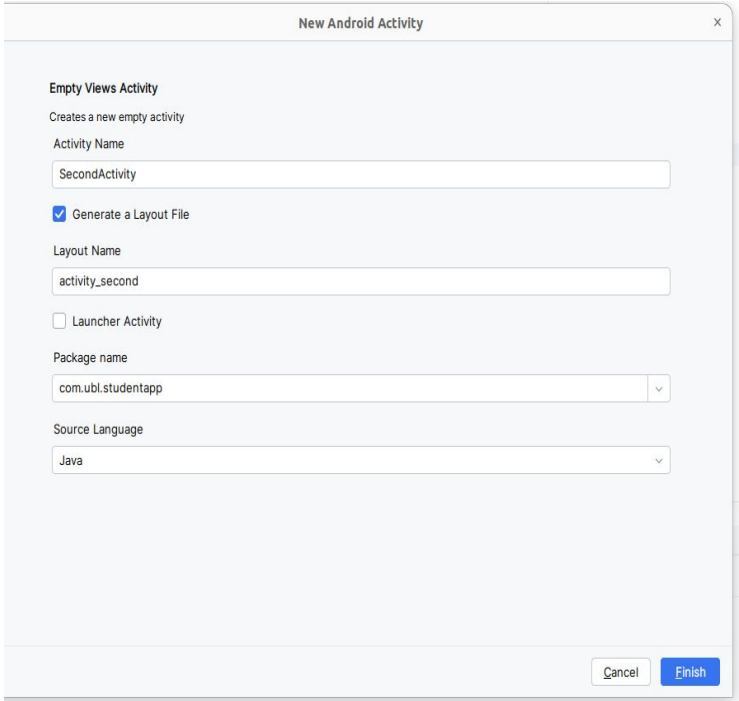
```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable( $this$enableEdgeToEdge: this);
        setContentView(R.layout.activity_main);

    }
}
```

Create new empty view activity

Right click at the java folder->New->Activity->Empty views activity



It will produce

- **Activity class (SecondActivity.java)**
- **Layout file (activity_second.xml)**

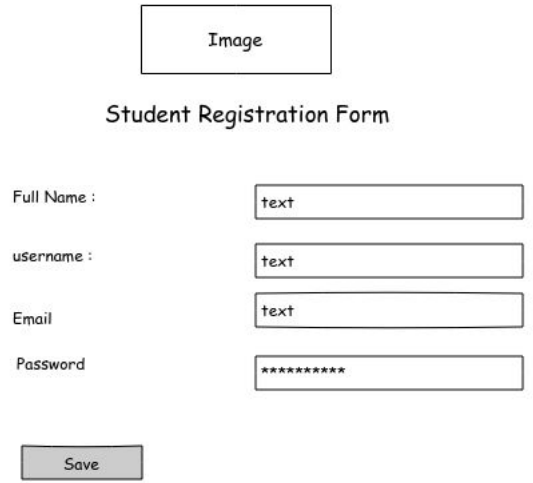
See the new Activity automatically added to AndroidManifest.xml

```
<application
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@mipmap/ic_launcher"
    android:label="StudentApp"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.StudentApp"
    tools:targetApi="31">
    <activity
        android:name=".SecondActivity"
        android:exported="false" />
    <activity
        android:name=".MainActivity"
        android:exported="true">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
```

Layout

Design Registration Form



Create layout from scratch using Linear Layout

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
----------------------
Add the view here
-----------------------
</LinearLayout>
```

Structure layout


Size of a View

- **wrap_content**

  android:layout_width="wrap_content"
- **match_parent**

  android:layout_width="match_parent"
- **Fixed value (use dp units)**

  android:layout_width="48dp"

Android has several ways to specify the width and height of a `View`. These examples are for the layout_width of a View, but the same applies for the `layout_height` attribute of a View.

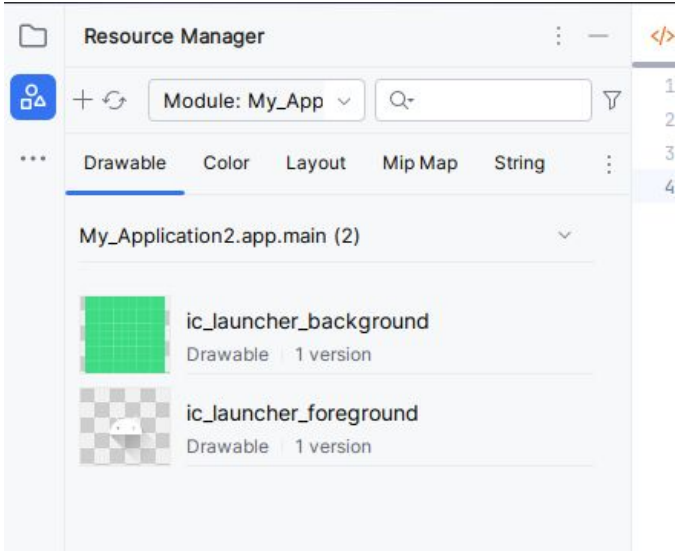
- Use `wrap_content` to use only as much space as needed to display the content within the `View`. For example, if you want the `View` to be as wide as the text within the `TextView`, use `wrap_content`.
- Use `match_parent` to use the dimension of the parent (such as matching the width of the parent `View`). For example, if you want the `ImageView` to take up the full size of the parent, set `match_parent` for its width and height.
- Lastly, if you want a fixed size for the `View`, then set a specific dp (density-independent pixels) value. We will be revisiting the concept of dps in the next lesson when we learn more about layouts.

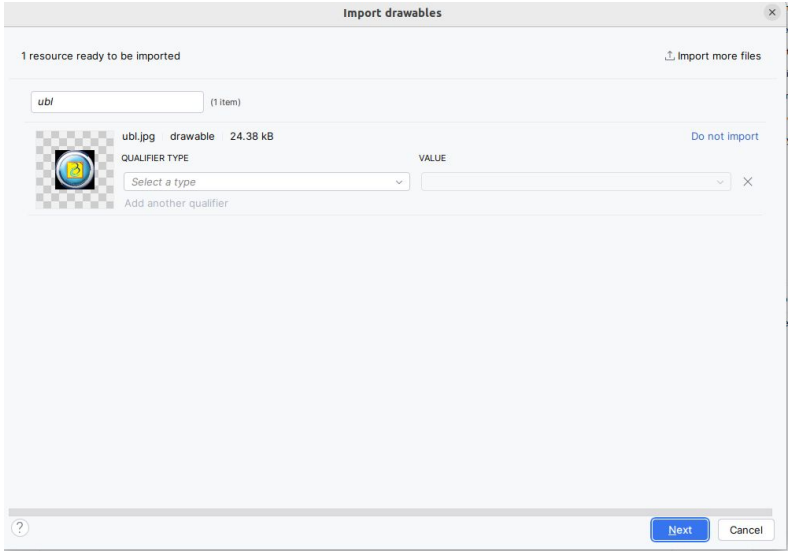
Create image view

`<ImageView/>`

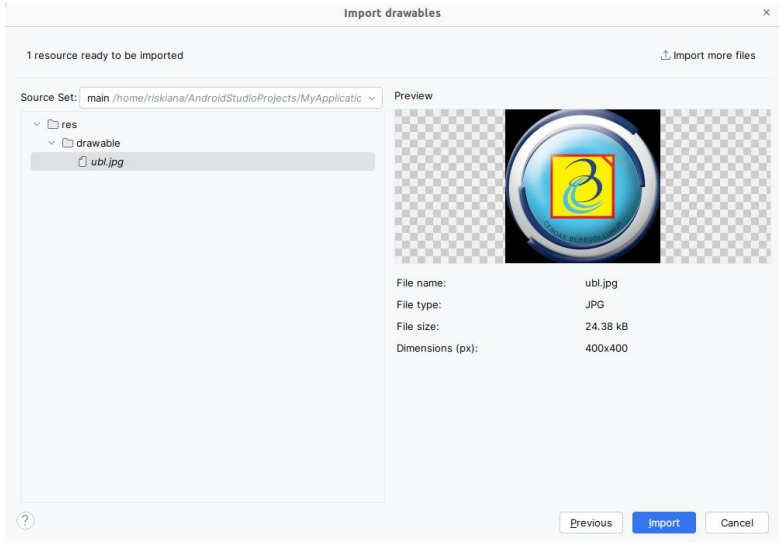
Use attribute :src to set the image source

Upload image file to the Drawable





Click Import




```
<ImageView
    android:src="@drawable/ubl"
    android:layout_width="100dp"
    android:layout_height="100dp"/>
```


Create Text view for Student Registration Form

```
<TextView
    android:text="Student Registration Form"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

Create Text view for full name

```
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">
    <TextView
        android:text="Full name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
    <EditText
        android:hint="input full name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

Use EditText for input type

Notice that We put the TextView and EditText inside the LinearLayout

Same thing is applied for input username


Create Email input

```
<TextView
    android:text="Email"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
<EditText
    android:hint="input email address"
    android:inputType="textEmailAddress"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

Notice that we use inputType=”textEmailAddress”

Create password input

```
<TextView
   android:text="Password"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"/>
<EditText
   android:hint="input password"
   android:inputType="textPassword"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"/>
```

Notice that we use inputType=”textPassword”

Create button save

```
<Button
    android:text="Save"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```


Adjust Layout

At the parent layout,add padding and gravity

```
android:padding="24dp"
android:gravity="center_horizontal"
```

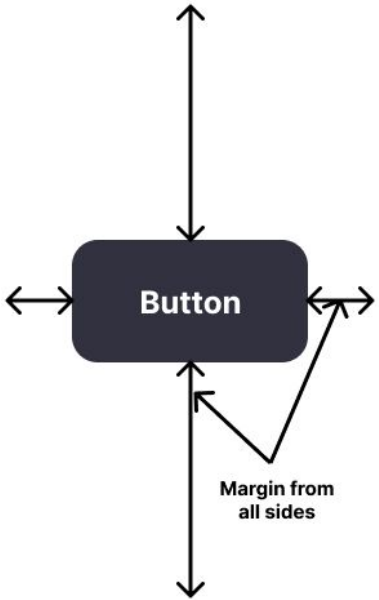
Each view
Add the layout margin

```
android:layout_margin="10dp"
```

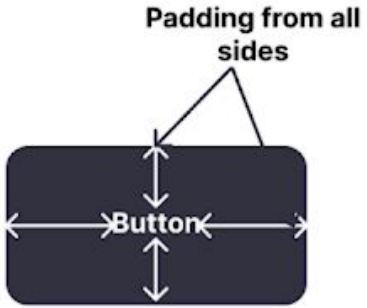
You could adjust other views such as text font, size, style

Margin Vs Padding

Margin is the distance between the view and its immediate sibling or a parent view. Margin is used to create a visual separation between two different views.

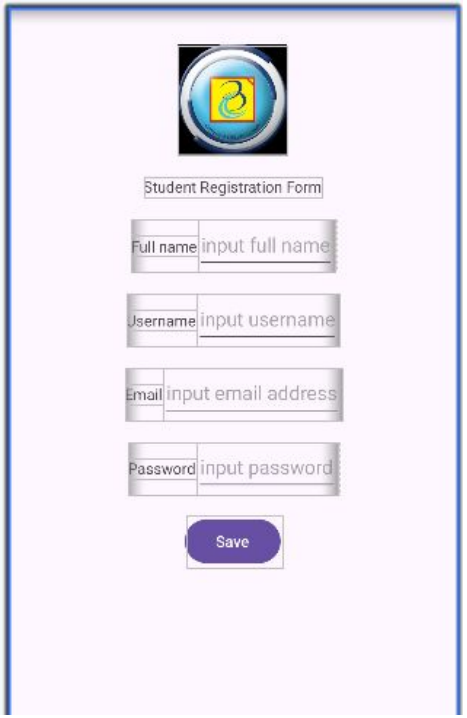


Padding is a space which is present between the content which we have to display and the border of that content. Padding is used to create the extra space within the content.



https://www.tutorialspoint.com/difference-between-a-view-s-margin-and-padding-in-android

Full Layout



Exercise:

- Add the Mobile Number with type number
- Add the Gender option with Male and Female

Views Id

```
android:id="@+id/identifier"
```

Naming convention

- **Lower case**
- **Use underscore for more than 1 word**
-

Sample for input fullname

```
<EditText
    android:id="@+id/txt_fullname"
    android:hint="input full name"
    android:minHeight="48dp"
    android:padding="8dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

Now, Create id for other views

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    EditText fullname =
findViewById(R.id.txt_fullname);
    EditText username =
findViewById(R.id.txt_username);
    TextView resultText =
findViewById(R.id.txt_result);
    Button btnSave = findViewById(R.id.btn_save);
```

Use the findViewById to determine which view id


Note

1. **We can ‘reformat code’ to tidy up the line breaks, indents, and order of our files (both XML and Java Code). Either Code Menu-> Reformat Code , or CTRL+ALT+L**

Hands on Lab

Tampilkan Nama, email, mobile phone pada text view result pada saat button save diclick