Activities

Tushar B. Kute,
http://tusharkute.com

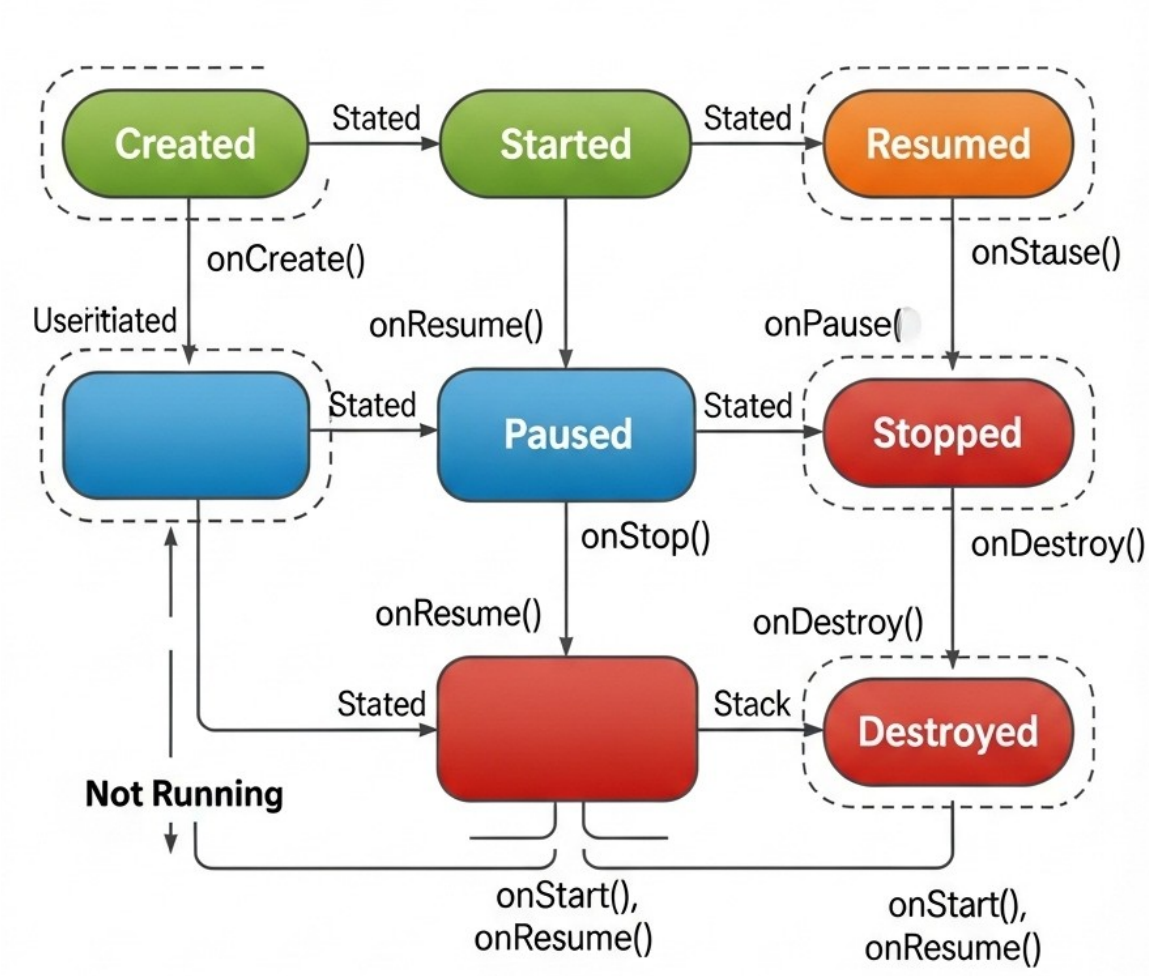
Activity

- An Activity is a single screen in an Android app with a user interface, like a login screen or a home page.
- The lifecycle of an activity refers to the different states it can be in, from being created to being destroyed.
- Android manages this lifecycle to ensure the app runs smoothly and efficiently.

Activity : Life Cycle

- onCreate(): The activity is being created for the first time. You set up your layout here.
- onStart(): The activity is becoming visible to the user.
- onResume(): The activity is now in the foreground and the user can interact with it.
- onPause(): The activity is partially hidden, like when a pop-up dialog appears.
- onStop(): The activity is no longer visible to the user.
- onDestroy(): The activity is being completely destroyed and removed from memory.

Activity : Life Cycle



Kotlin Activity

- A View is the basic building block for the user interface. Think of it as a single widget or component on the screen, like a button, a text box, or an image.
- A ViewGroup is a special kind of view that can contain other views and view groups. It acts as a container or layout manager, organizing and arranging the child views within it.
- Common examples are LinearLayout which arranges views in a single row or column, and RelativeLayout which positions views relative to each other.

Kotlin Activity

```kotlin
package com.example.myfirstapp

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import android.widget.TextView // Import TextView if using it directly in

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        // Set the content view for this Activity from the layout XML fil
        setContentView(R.layout.activity_main)

        // Example: Accessing a TextView from the layout and setting its
        // val myTextView: TextView = findViewById(R.id.myTextView)
        // myTextView.text = "Hello from Kotlin Activity!"
    }
}
```

Kotlin Activity

- package com.example.myfirstapp:
  - Defines the package for your application.
- import android.os.Bundle:
- Imports the Bundle class, used to pass data between activities and save/restore activity state.
- import androidx.appcompat.app.AppCompatActivity:
  - Imports the base class for activities, providing compatibility features across different Android versions.

Kotlin Activity

- class MainActivity : AppCompatActivity():
  - Declares MainActivity as a class that inherits from AppCompatActivity, making it an Android Activity.
  - override fun onCreate(savedInstanceState: Bundle?):
- This is the primary lifecycle method of an Activity. It's called when the Activity is first created.
  - super.onCreate(savedInstanceState): Calls the onCreate method of the parent class.
  - setContentView(R.layout.activity_main): This line connects your Kotlin code to the user interface defined in the activity_main.xml layout file.

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://
    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">

    <TextView
        android:id="@+id/myTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

activity_main.xml

- This XML defines the layout of your MainActivity.
- It uses a ConstraintLayout as the root element and contains a TextView with the ID myTextView and the initial text "Hello World!".

AndroidManifest.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.myfirstapp">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyFirstApp">

        <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>
```

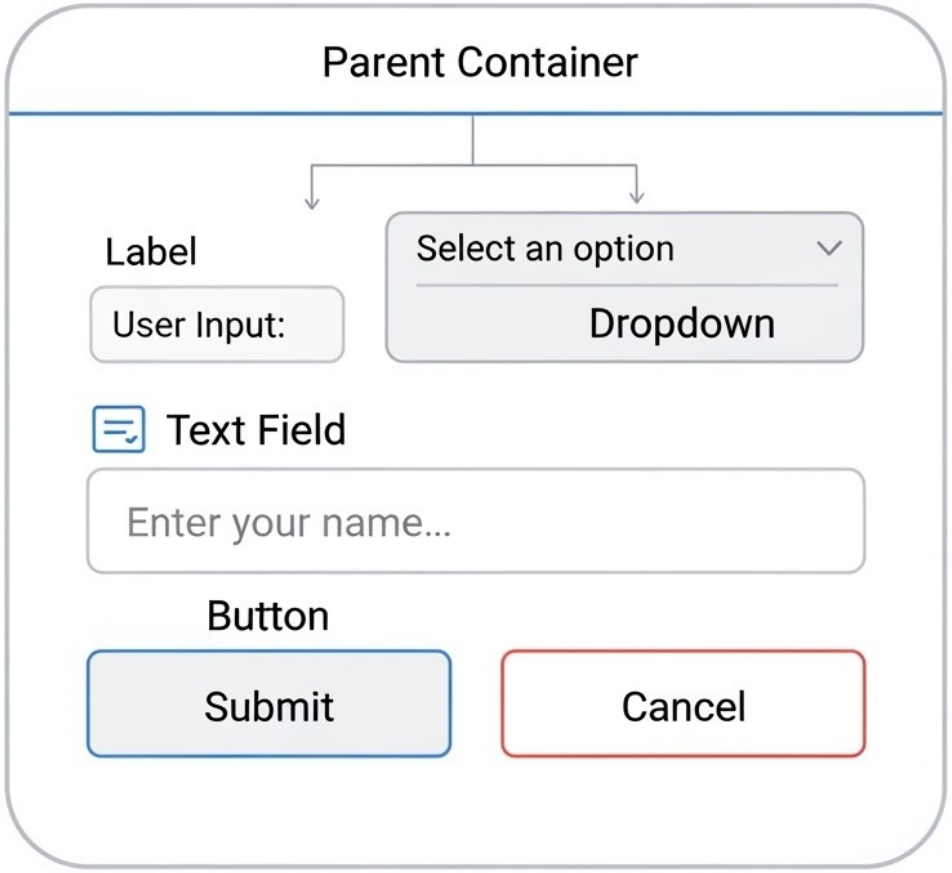
AndroidManifest.xml

- This file declares the components of your application to the Android system.
- The <activity> tag registers MainActivity as an activity in your app.
- The <intent-filter> with android.intent.action.MAIN and android.intent.category.LAUNCHER specifies that MainActivity is the entry point of your application and should appear in the device's app launcher.

View

- A View is the basic building block for the user interface. Think of it as a single widget or component on the screen, like a button, a text box, or an image.
- A ViewGroup is a special kind of view that can contain other views and view groups. It acts as a container or layout manager, organizing and arranging the child views within it.
- Common examples are LinearLayout which arranges views in a single row or column, and RelativeLayout which positions views relative to each other.

View


Parent Container
Label
User Input:
Select an option
Dropdown
Text Field
Enter your name...
Button
Submit
Cancel

Fragment

- A Fragment represents a reusable, modular portion of an activity's user interface.
- Fragments are often used to create flexible layouts for different screen sizes, like a tablet where you might have two fragments side-by-side on one screen, but on a phone, they might be in separate activities.
- A fragment must always be hosted by an activity.
- Fragments have their own lifecycle that is closely tied to the lifecycle of the activity they are in. For instance, when an activity is resumed, all of its fragments are also resumed.

Fragment : Life Cycle

- onAttach(): The fragment has been associated with an activity.
- onCreateView(): The fragment's user interface is being created.
- onActivityCreated(): The hosting activity has completed its onCreate() method.
- onDestroyView(): The fragment's UI is being removed.
- onDetach(): The fragment is no longer associated with its activity.

Example

- Fragment

Thank you

@mitu_skillologies

@mITuSkillologies

@mitu_group

@mitu-skillologies

@MITUSkillologies

kaggle

@mituskillologies

Web Resources
https://mitu.co.in
http://tusharkute.com

@mituskillologies

contact@mitu.co.in

tushar@tusharkute.com