**PRACTICAL # 03**

**OBJECT:**

Understanding activity lifecycle

**THEORY:**

Android applications run in their own processes. Processes are started and stopped as needed to run an application's components. Processes may be killed to reclaim system resources for other applications.

Activities are fundamental to Android applications. An activity typically represents a UI screen of an application. Activities can exist in different states within an application. When an activity changes state, corresponding lifecycle event callback method is invoked, allowing to execute code in order to adapt to that change. Figure 1 shows the flow diagram of these lifecycle events.

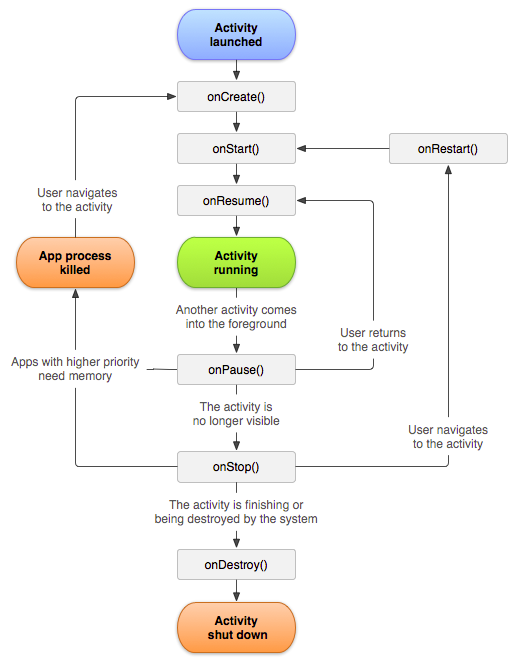


Figure 1: Activity lifecycle

**Activity CallBacks**

**OnCreate:** First method called when an activity is created, to perform any startup initializations that may be required by an Activity such as: Creating views, Initializing variables. Bundle parameter is a dictionary for storing and passing state information and objects between activities. A non-null bundle indicates the activity is restarting and it should restore its state from the previous instance.

*@Override protected void onCreate(Bundle savedInstanceState) {*

*super.onCreate(savedInstanceState);*

*setContentView(R.layout.activity\_main);*

*}*

**OnStart:** After OnCreate is finished, OnStart is called. Activities may override this method to perform any specific tasks right before an activity becomes visible, such as refreshing current values of views within the activity.

**OnResume:**

OnResume is called immediately after OnStart.

This method when the Activity is ready to start interacting with the user. Activities should override this method to perform tasks such as: Starting animations, Listening for GPS updates, Display any relevant alerts or dialogs.

*public void OnResume() {*

*super.OnResume(); // Always call the superclass first.*

*if (\_camera==null) {*

*// Do camera initializations here*

*}*

*}*

**OnPause:**

Called just before the system puts an activity into the background or when the activity becomes partially obscure. Override this method to: Commit unsaved changes to persistent data, Destroy or clean up other objects consuming resources.

*public void OnPause() {*

*super.onPause(); // Always call the superclass first*

*// Releasing the camera as other activities might need it*

*if (\_camera != null) {*

*\_camera.Release();*

*\_camera = null;*

*}*

*}*

**OnStop:**

This method is called when the activity is no longer visible to the user. This happens when one of the following occurs:

- A new activity is being started and is covering up this activity.

- An existing activity is being brought to the foreground.

- The activity is being destroyed.

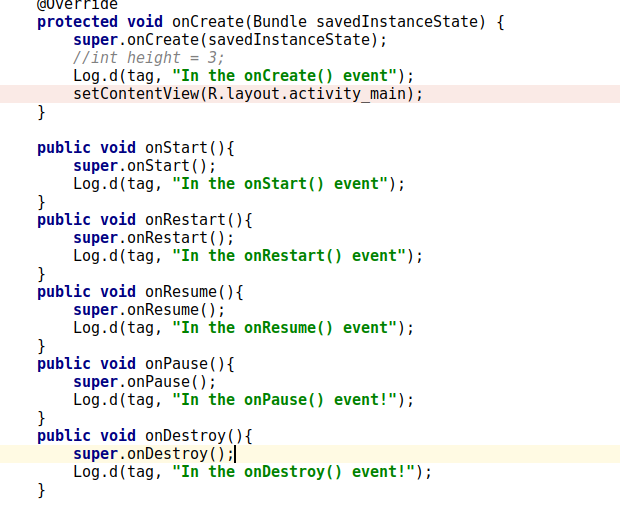
**OnDestroy:**

The final method that is called on an Activity instance before it's destroyed and removed from memory. In some extreme situation (e.g application crash) when application process is killed by system, this method is not invoked. The method is typically overridden to clean up long running resources that might leak resources. An example of this might be background threads that were started in OnCreate.

**OnRestart:**

This method is called after an activity has been stopped, prior to it being started again. For instance, when user presses home button while on an activity in the application, OnPause and then OnStop methods are called, and the Activity is moved to the background but is not destroyed. If the user were then to restore the application by the task manager or a similar application, Android will call the OnRestart method of the activity.

These activity lifecycle events are shown in the code below.



**ACTIVITIES**

**Activity 1**

Create Hello World Project, and override the onCreate method to show your name on TextView.

**REVIEW QUESTIONS**

1. What is an Android activity?
2. What is activity lifecycle?
3. What are different lifecycles of an activity?
4. Is it necessary to override all event callbacks?