MOBILE APPLICATION DEVELOPMENT (EI-333)

Lecture:06

"Activities"

ANDROID

ACTIVITIES

ANDROID

App components

- App components are the essential building blocks of an Android app. [1]
- Each component is an entry point through which the system or a user can enter your app.
- Some components depend on others.
- Each type serves a distinct purpose and has a distinct lifecycle that defines how the component is created and destroyed.
- There are four different types of app components:
  - Activities.
  - Services.
  - Broadcast receivers.
  - Content providers.

Activities

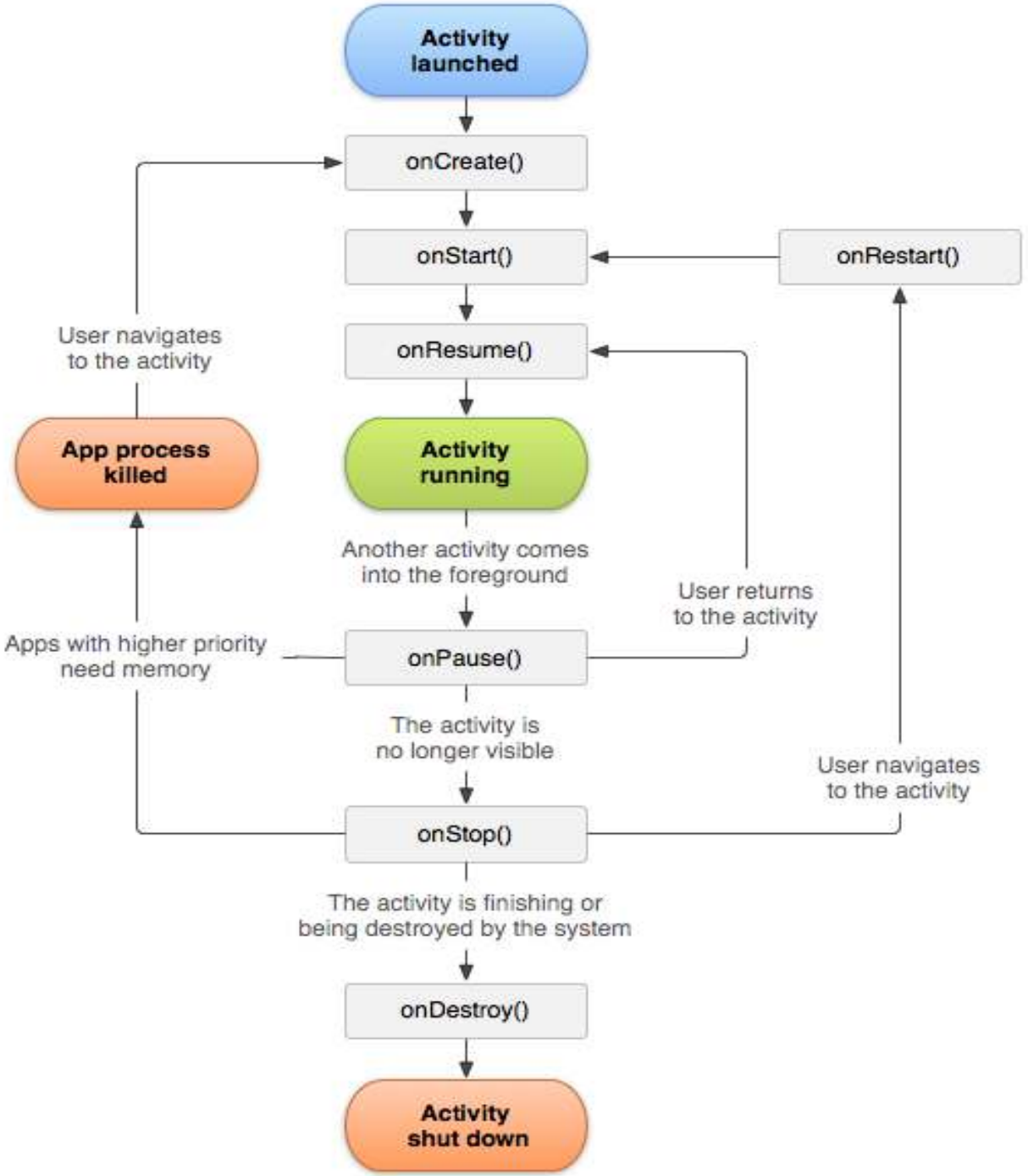
- Activities are one of the fundamental building blocks of apps on the Android platform. [1][2]
- An *activity* is the entry point for interacting with the user.
- They are also central to how a user navigates within an app (as with the Back button) or between apps (as with the Recents button).
- It represents a single screen with a user interface. For example, an email app might have
  - One activity that shows a list of new emails
  - Another activity to compose an email
  - Another activity for reading emails

Declare Activities

- For your app to be able to use activities, you must declare the activities, and certain of their attributes, in the manifest File.
- To declare your activity, open your manifest file and add an <activity> element as a child of the <application> element.
- **<manifest ... >**
  <application ... >
  <activity
  android:name=".ExampleActivity" />
  ...
  </application ... >
  ...
  </manifest >

Managing the activity lifecycle

- Over the course of its lifetime, an activity goes through a number of states.
- You use a series of callbacks to handle transitions between states.
- Within the following lifecycle callback methods, you can declare how your activity behaves when the user leaves and re-enters the activity.
- To navigate transitions between stages of the activity lifecycle, the Activity class provides a core set of six callbacks: onCreate(), onStart(),onResume(),onPause(), onStop(), and onDestroy().


Activity launched
onCreate()
onStart()
onRestart()
onResume()
User navigates to the activity
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

onCreate()

- You must implement this callback, which fires when the system first creates the activity.
- On activity creation, the activity enters the *Created* state.
- In the onCreate() method, you perform basic application startup logic that should happen only once for the entire life of the activity. For example, your implementation of onCreate() might bind data to lists, initialize background threads, and instantiate some class-scope variables.
- This method receives the parameter savedInstanceState, which is a Bundle object containing the activity's previously saved state. If the activity has never existed before, the value of the Bundle object is null.

onStart()

- When the activity enters the Started state, the system invokes this callback.
- The onStart() call makes the activity visible to the user, as the app prepares for the activity to enter the foreground and become interactive.
- For example, this method is where the app initializes the code that maintains the UI.
- The onStart() method completes very quickly and, as with the Created state, the activity does not stay resident in the Started state.
- Once this callback finishes, the activity enters the *Resumed* state, and the system invokes the onResume() method

onResume()

- When the activity enters the Resumed state, it comes to the foreground, and then the system invokes the onResume() callback.
- This is the state in which the app interacts with the user.
- The app stays in this state until something happens to take focus away from the app. Such an event might be, for instance, receiving a phone call, the user's navigating to another activity, or the device screen's turning off.
- When an interruptive event occurs, the activity enters the *Paused* state, and the system invokes the onPause() callback.
- If the activity returns to the Resumed state from the Paused state, the system once again calls onResume() method.
- implement onResume() to initialize components that you release during onPause().

onPause()

- The system calls this method as the first indication that the user is leaving your activity
- Use the onPause() method to pause operations such animations and music playback that should not continue while the Activity is in the Paused state

onStop()

- When your activity is no longer visible to the user, it has entered the *Stopped* state, and the system invokes the onStop() callback.
- This may occur, for example, when a newly launched activity covers the entire screen.
- The system may also call onStop() when the activity has finished running, and is about to be terminated.
- In the onStop() method, the app should release almost all resources that aren't needed while the user is not using it.
- For example, if you registered a BroadcastReceiver in onStart() to listen for changes that might affect your UI, you can unregister the broadcast receiver in onStop(), as the user can no longer see the UI. It is also important that you use onStop() to release resources that might leak memory, because it is possible for the system to kill the process hosting your activity without calling the activity's final onDestroy() callback.

onDestroy()

- Called before the activity is destroyed.
- This is the final call that the activity receives.
- The system either invokes this callback because the activity is finishing due to someone's calling finish(), or because the system is temporarily destroying the process containing the activity to save space.

THANK YOU ☺