

A seagull is shown in flight, wings spread wide, against a dramatic sky at sunset or sunrise. The sky is filled with clouds that are illuminated from below, creating a mix of deep blue, orange, and yellow hues. The seagull is white with dark wingtips and a yellow beak.

globalsyn
Taking People To The Next Level

globalsyn



globsyn 
finishing school

Activities



Apps Components

♣ Topics to be covered in this session:

- Activities
- Services

Activities

- 1 An activity is an application component that provides a screen with which users can interact in order to perform some activity.
- 2 Each activity is given a window to draw its user interface.
- 3 An application usually consists of multiple activities that are loosely bound to each other.
- 4 But only one activity is specified as the main activity as it is running currently.
- 5 Each activity can start another activity in order to perform different actions as desired by the user.

Activities

- | Each time a new activity starts, the previous activity is stopped and pushed to a stack called the "back stack".
- | The back stack abides to the basic "last in, first out" stack mechanism.
- | when the user is done with the current activity with the Back button, the previous activity resumes by popping put from the stack.
- | When a running activity is stopped to start a new activity this change in state is notified with the help of the activity lifecycle callback method.
- | Each callback aids to perform a specific work that's appropriate to that state change.

Activities

Creating an activity:

- | To create an activity, a subclass of it has to be created.
- | Within the subclass, callback methods have to be implemented to manage the transitioning of the activity being created, stopped, resumed, or destroyed.
- | The two most important callback methods are:
 - ▣ **onCreate()**: The system calls this to create an activity. the essential components of the activity have to be initialized. This is where the layout for the activity's user interface is defined.

Activities

- **onPause():** The system calls this method to indicate that the user is leaving the activity.
This is where certain changes are committed that should be persist beyond the current user session.

Implementing a user interface:

- ⌋ The user interface for an activity is provided by a hierarchy of views from the view class.
- ⌋ Each view controls a particular rectangular space within the activity's window and responds to user interaction.
- ⌋ A number of ready-made views are available to design and organize the layout efficiently.

Activities

- | The most common way to define a layout using views is with an XML layout file saved in the application resources.
- | With this the design of user interface can be maintained separately from the source code defining the activity's behavior.
- | In order for the activity to be accessible to the system, it has to be declared in the manifest file .
- | To do this open the manifest file and add an activity element as a child of the application element
- | An activity element can also specify various intent filters, these declare how other application components may activate it.

Activities

Starting an activity

- | An activity can be started by calling `startActivity()` and passing it an `Intent`.
- | The intent describes the activity to be started.
- | It specifies either the exact activity or describes the type of action to perform.
- | In specific cases where the results for an activity are immediately required the function `startActivityForResult()` can be called.
- | In order to receive the result the function `onActivityResult()` can be called.

Activities

- | An activity can be shut down by calling its finish() method.
- | A separate activity, started previously, can be shut off by calling finishActivity().
- | An activity can exist in essentially three states:
 - **Resumed or Running:** The activity is in the foreground of the screen and has user focus.)
 - **Paused:** Another activity is in the foreground and is visible whereas this one is partially visible.

A paused activity is completely alive but can be killed by the system in extremely low memory situations.

Activities

- ♣ **Stopped:** The activity is completely obscured by another activity but is retained in memory and is not attached to the window manager.

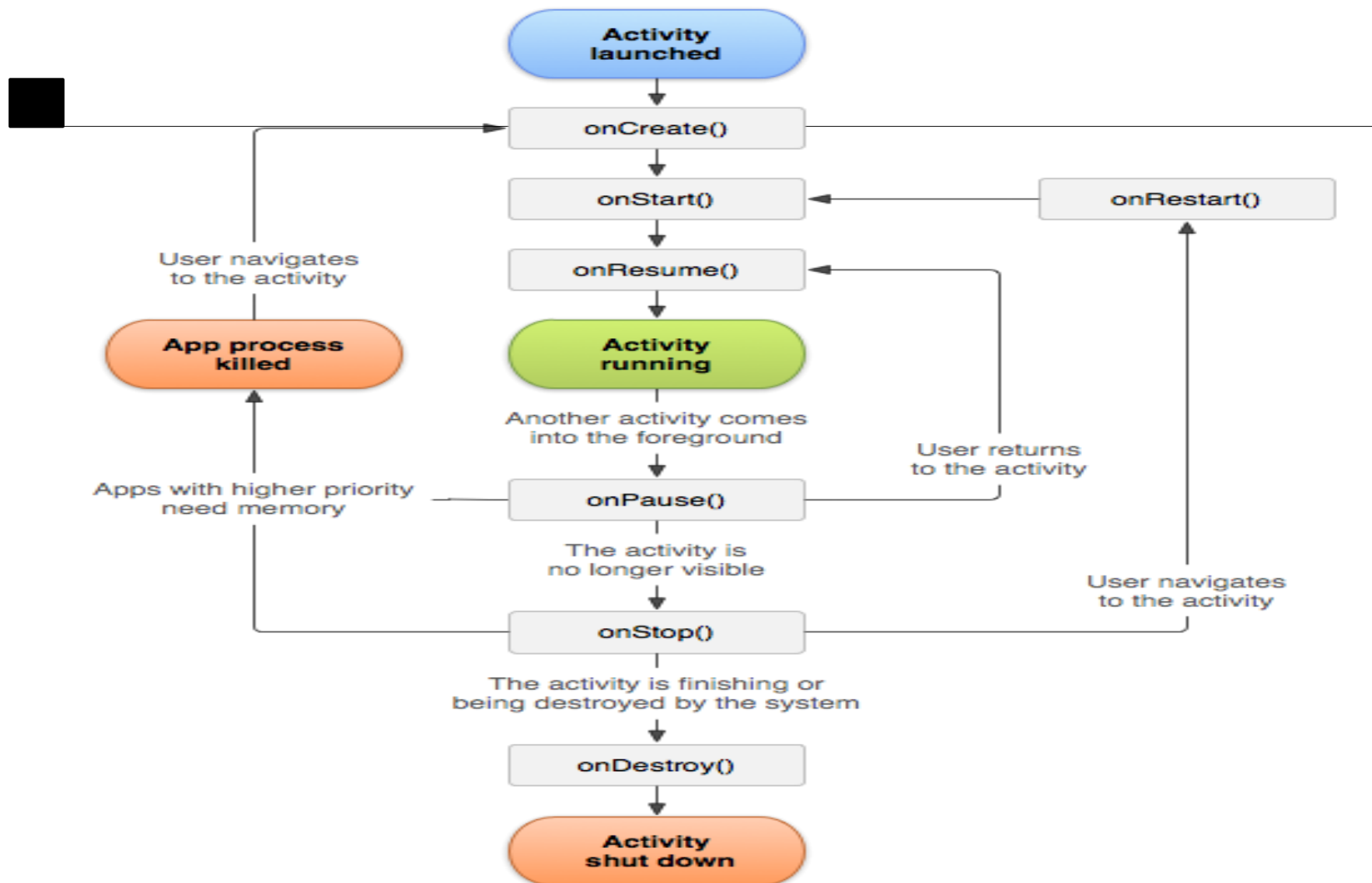
However, it is no longer visible to the user and it can be killed by the system when memory is needed elsewhere.

Activity Lifecycle:

- ♣ The activity lifecycle begins with launching the activity and creating it.
- ♣ Once the activity is created, it is started by the system as a result the activity begins to run.
- ♣ If another new activity interrupts the running activity, the current activity is paused.

Activities

- When this activity is not resumed by the user for a longer time and the system is in shortage of memory that activity is killed immediately.
- Else the activity remains and resumes when the user switches back to it.
- When the activity is no longer visible, it is destroyed by the system completely.
- The following diagram illustration explains the activity lifecycle in a better tone:



Activities

- It is said already that the state of the activity is saved before pausing or stopping it.
- Even while destroying the system stores the state of the activity with the help of the call back method `onSaveInstanceState()`.
- With this even when the system destroys an activity it can be restored back.
- Some device configurations change during runtime, to handle these the Android recreates the running activity by destroying and creating back instantaneously.
- This behaviour is designed to help the application adapt to newer configurations.

Activities

- Another important case is to manage the coordination of the activities.
- When one activity (A) wishes to start another activity (B) the process executes in the below mentioned fashion.
- Activity A's onPause() method executes.
- Activity B's onCreate(), onStart(), and onResume() methods execute in sequence.
- Then, if Activity A is no longer visible on screen, its onStop() method executes.
- This predictable sequence facilitates to manage the transition of information from one activity to another.

Activities


- 1 A Fragment represents a behavior or a portion of user interface in an Activity.
- 2 It decomposes the application functionality and UI into reusable modules.
- 3 Switching between activities can be limited with the help of multiple fragments.
- 4 Fragments have their own lifecycle, state, and back stack
- 5 They require an API Level 11 or greater.

Activities

- | Loaders are those that asynchronously load data in an activity or fragment.
- | They have the following characteristics:
- | They are available to every Activity and Fragment.
- | They provide asynchronous loading of data.
- | They monitor the source of their data and deliver new results when the content changes.
- | They automatically reconnect to the last loader's cursor when being recreated after a configuration change.
- | Thus, they don't need to re-query their data.

Activities

- | A task is a collection of activities in the user interaction order.
- | All the activities belong to a task.
- | Tasks enable an activity to move to the background and retain the state of each activity.
- | This retention is performed with the help of a back stack.



Taking People To The Next Level...