UNIT III: ACTIVITIES, FRAGMENTS, AND INTENTS

3.1 Intents

It is quite usual for users to witness a jump from one application to another as a part of the whole process

Intents are used for navigating among various activities within the same application

Android application components can connect to other Android applications. This connection is based on a task description represented by an Intent object.

Intents are asynchronous messages which allow application components to request functionality from other Android components. Intents allow you to interact with components from the same applications as well as with components contributed by other applications

Uses of Intent

1. Sending the User to Another App
2. Setting a Launcher Activity
3. Switch Between Current Activity to Next Activity

Intents are objects of the android.content.Intent type. Your code can send them to the Android system defining the components you are targeting. For example, via the startActivity() method you can define that the intent should be used to start an activity.

There are two types of intents in android:

1. Explicit Intent
2. Implicit Intent.

Explicit Intent

Explicit Intent specifies the component. In such a case, intent provides the external class to be invoked.

Explicit Intents are used to connect the application internally.

In Explicit we use the name of component which will be affected by Intent. For Example: If we know class name then we can navigate the app from One Activity to another activity using Intent. In the similar way we can start a service to download a file in background process.

Explicit Intent work internally within an application to perform navigation and data transfer.

```
Intent intent = new Intent(getApplicationContext(), SecondActivity.class);
startActivity(intent);
```

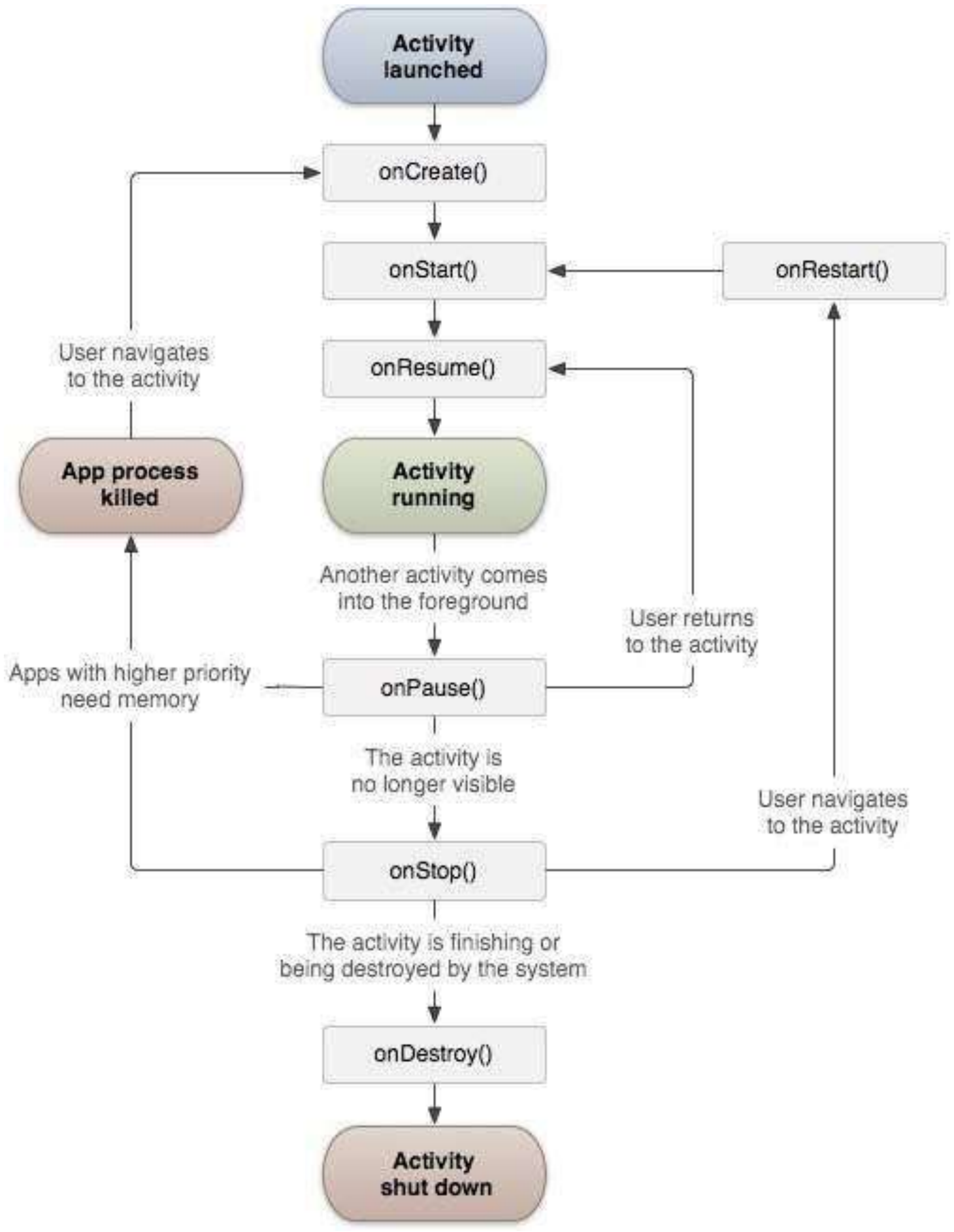
Implicit Intent

In Implicit Intents we do need to specify the name of the component. We just specify the Action which has to be performed and further this action is handled by the component of another application.

The basic example of implicit Intent is to open any web page

```
Intent intentObj = new Intent(Intent.ACTION_VIEW);
intentObj.setData(Uri.parse("https://www.google.com"));
startActivity(intentObj);
```

3.2. Activity life cycle


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

An activity represents a single screen with a user interface just like window or frame of Java.Android activity is the subclass of ContextThemeWrapper class. By the help of activity, you can place all your UI components or widgets in a single screen.
The 7 lifecycle method of Activity describes how activity will behave at different states.

1. **onCreate :** called when activity is first created.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
}
```

2. **onStart :** called when activity is becoming visible to the user.

```
@Override
protected void onStart() {
   super.onStart();
}
```

3. **onResume :** called when activity will start interacting with the user.

```
@Override
protected void onResume() {
   super.onResume();
}
```

4. **onPause :** called when activity is not visible to the user.

```
@Override
protected void onPause() {
   super.onPause();
}
```

5. **onStop :** called when activity is no longer visible to the user.

```
@Override
protected void onStop() {
   super.onStop();
}
```

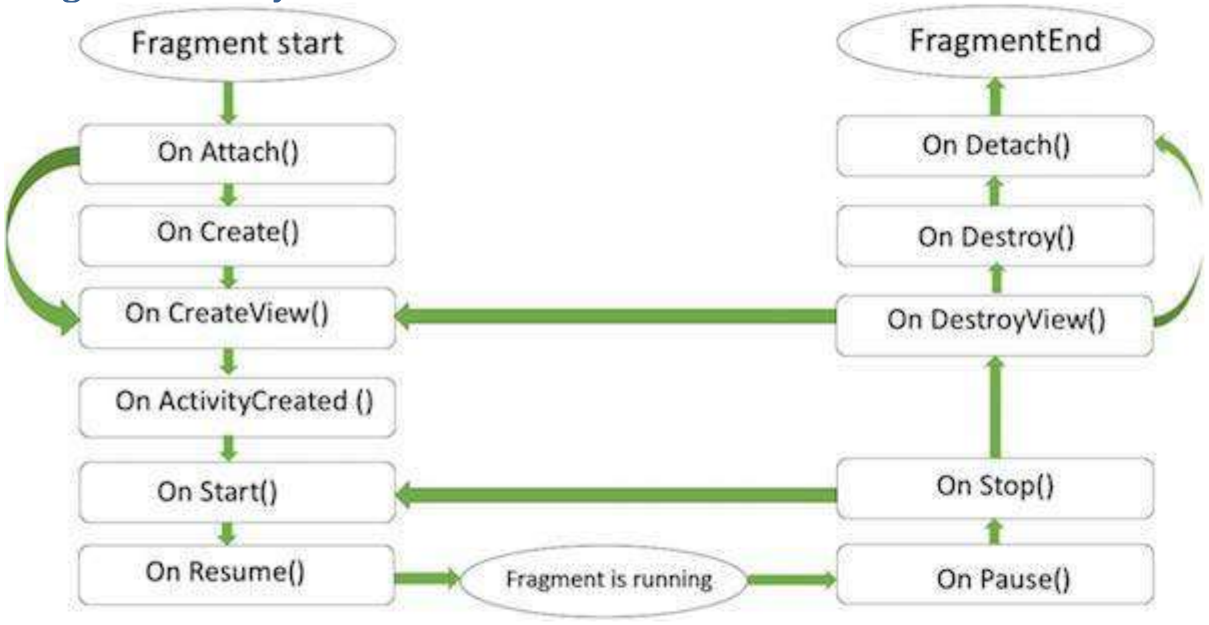
6. **onRestart :** called after your activity is stopped, prior to start.
7. **onDestroy :** called before the activity is destroyed.

3.3 Fragments

A Fragment **is a piece of an activity which enable more modular activity design. It will not be wrong if we say, a fragment is a kind of** sub-activity**.**

- A fragment has its own layout and its own behaviour with its own life cycle callbacks.
- You can add or remove fragments in an activity while the activity is running.
- You can combine multiple fragments in a single activity to build a multi-pane UI.
- A fragment can be used in multiple activities.
- Fragment life cycle is closely related to the life cycle of its host activity which means when the activity is paused, all the fragments available in the activity will also be stopped.
- A fragment can implement a behaviour that has no user interface component.
- Fragments were added to the Android API in Honeycomb version of Android which API version 11.

Fragment Life Cycle



onAttach()
The fragment instance is associated with an activity instance.The fragment and the activity is not fully initialized.

onCreate()
The system calls this method when creating the fragment.

onCreateView()
The system calls this callback when it's time for the fragment to draw its user interface for the first time. To draw a UI for your fragment, you must return a **View** component from this method that is the root of your fragment's layout.

onActivityCreated()
The onActivityCreated() is called after the onCreateView() method when the host activity is created.

onStart()

The onStart() method is called once the fragment gets visible.

onResume()

Fragment becomes active.

onPause()

The system calls this method as the first indication that the user is leaving the fragment.

onStop()

Fragment going to be stopped by calling onStop()

onDestroyView()

Fragment view will destroy after call this method

onDestroy()

onDestroy() called to do final clean-up of the fragment's state but Not guaranteed to be called by the Android platform.

Types of Android Fragments

1. **Single Fragment:** Display only one single view on the device screen. This type of fragment is mostly used for mobile phones.
2. **List Fragment:** This Fragment is used to display a list-view from which the user can select the desired sub-activity. The menu drawer of apps like Gmail is the best example of this kind of fragment.
3. **Fragment Transaction:** This kind of fragments supports the transition from one fragment to another at run time. Users can switch between multiple fragments like switching tabs.

```
FirstFragment fragment=new FirstFragment();
FragmentTransaction
transaction=getSupportFragmentManager().beginTransaction();
transaction.replace(R.id.layout1,fragment);
transaction.commit();
```

Thank You