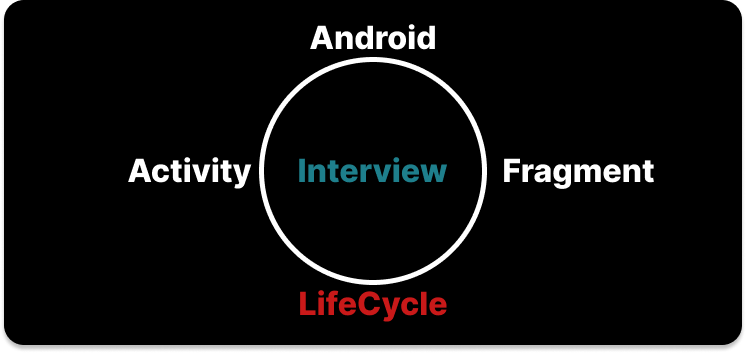
**Android Activity & Fragment Lifecycles, Most ASKED Interview Questions and Answers**



Questions about Lifecycles ( Activity and Fragment ) are asked in every interview for the Android development position. I will discuss some of the most asked questions here. I hope you will find these helpful during your interview preparation. So let’s start…

Q**uestion 1  
Tell about the Lifecycle of the Activity.**

***Answer :***  
· *OnCreate(): This is when the view is first created. This is normally where we create views, get data from bundles, etc.*

· *OnStart(): Called when the activity is becoming visible to the user. Followed by onResume() if the activity comes to the foreground, or onStop() if it becomes hidden.*

· *OnResume(): Called when the activity will start interacting with the user. At this point, your activity is at the top of the activity stack, with user input going to it.*

· *OnPause(): Called as part of the activity lifecycle when an activity is going into the background but has not (yet) been killed.*

· *OnStop(): Called when you are no longer visible to the user.*

· *OnDestroy(): Called when the activity is finishing*

· *OnRestart(): Called after your activity has been stopped, prior to it being started again*

Q**uestion 2  
Difference between onCreate() and onStart() in Activity Lifecycle**

***Answer :***  
· *The onCreate() method is called once during the Activity lifecycle, either when the application starts, or when the Activity has been destroyed and then recreated, for example during a configuration change.*

· *The onStart() method is called whenever the Activity becomes visible to the user, typically after onCreate() or onRestart().*

Q**uestion 3  
A scenario in which only onDestroy is called for an activity without onPause() and onStop()? / What will happen when you call finish() in onCreate?**

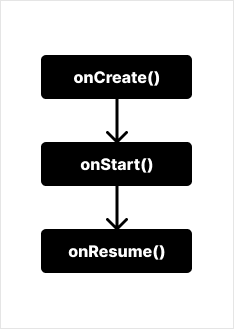
***Answer:****If finish() is called in the OnCreate method of an activity, the system will invoke onDestroy() method directly.*

Q**uestion 4  
Why would you do the setContentView() in onCreate() of Activity class?**

***Answer:****As onCreate() of an Activity is called only once, this is the point where most initialization should go. It is inefficient to set the content in onResume() or onStart() (which are called multiple times) as the setContentView() is a heavy operation.*

Q**uestion 5  
When activity A is opened -> It’s lifecycle**

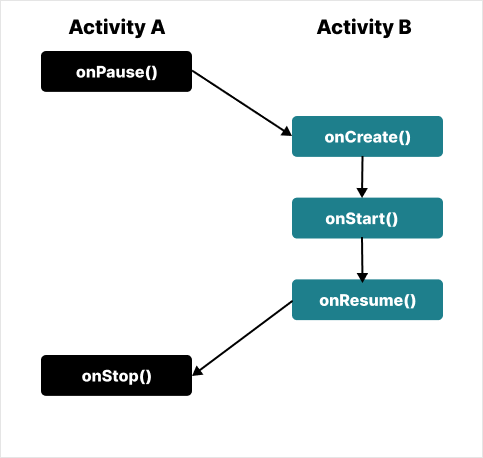
***Answer :***



Lifecycle: When an Activity is opened

Q**uestion 6  
When another activity B is opened from Activity A, the lifecycle of the previous and current activity.**

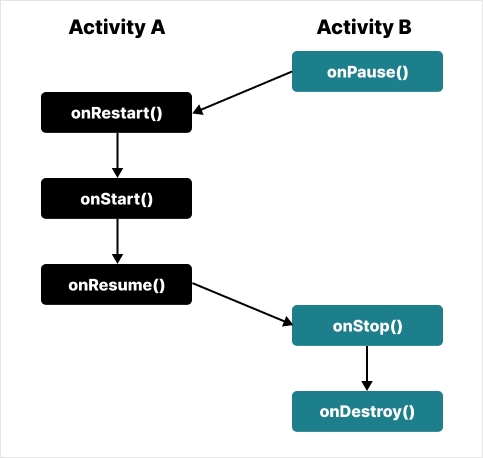
***Answer:***



Lifecycle: Activity B is opened from Activity A

Q**uestion 7  
When you press the back button on Activity B -> then the lifecycle of Activity B and Activity A in sequence.**

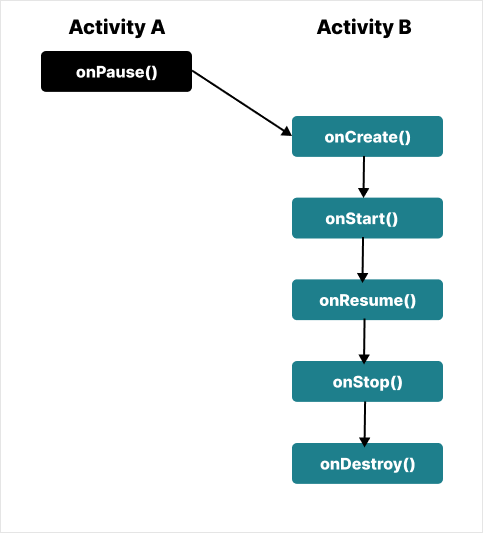
***Answer:***



Lifecycle: Backbuton press on Activity B and back to Activity A

Q**uestion 8  
LifeCycle -> From Activity A start Activity B by calling StartActivity(Intent) on button click and use finish() method inside onstart() method on Activity B**

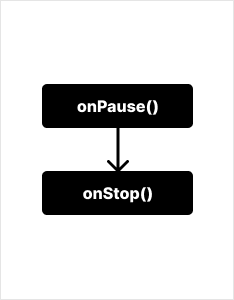
***Answer:***



LifeCycle : From Activity A start Activity B by calling StartActivity(Intent) on button click and use finish() method inside onstart() method on Activity B

Q**uestion 9  
Activity lifecycle when Clicking the Home button / Screen light off / Incoming call?**

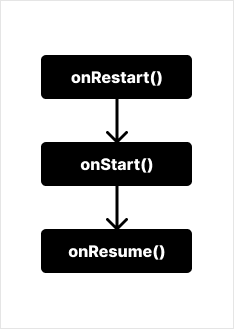
***Answer:***



Activity lifecycle: Click Home button / Screen light off / Incoming call

Q**uestion 10  
Activity lifecycle when open app via Recent / Screen light on / Close incoming call?**

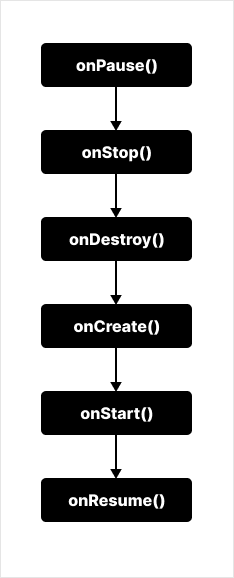
***Answer:***



Activity lifecycle: Open app via Recent / Screen light on / Close incoming call

Q**uestion 11  
Activity Lifecycle on Configuration change or Rotate the screen?**

***Answer:***



Activity Lifecycle: Configuration change or Rotate the screen

Q**uestion 12  
What is onSavedInstanceState() and onRestoreInstanceState() in activity?**

***Answer:  
OnRestoreInstanceState()****— When an activity is recreated after it was previously destroyed, we can recover the saved state from the Bundle that the system passes to the activity. Both the onCreate() and onRestoreInstanceState() callback methods receive the same Bundle that contains the instance state information. But because the onCreate() method is called whether the system is creating a new instance of your activity or recreating a previous one, you must check whether the state Bundle is null before you attempt to read it. If it is null, then the system is creating a new instance of the activity, instead of restoring a previous one that was destroyed.*

***onSaveInstanceState()****— is a method used to store data before pausing the activity.*

Q**uestion 13  
When onSaveInstance() is called?**

***Answer:****As the activity begins to stop, the system calls the [onSaveInstanceState()](https://developer.android.com/reference/android/app/Activity" \l "onSaveInstanceState(android.os.Bundle)" \t "_blank) method so the activity can save state information to an instance state bundle. onSaveInstanceState() called between onPause() and onStop().*

Q**uestion 14  
When OnRestoreInstanceState is called*?***

***Answer:****OnRestoreInstanceState() is called after onStart().*

Q**uestion 15  
Tell about the Fragment lifecycle**

***Answer:***· *onAttach(): The fragment instance is associated with an activity instance. The fragment and the activity are not fully initialized. Typically you get in this method a reference to the activity which uses the fragment for further initialization work.*

· *onCreate(): The system calls this method when creating the fragment. You should initialize essential components of the fragment that you want to retain when the fragment is paused or stopped, then resumed.*

· *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. You can return null if the fragment does not provide a UI.*

· *onActivityCreated() : The onActivityCreated() is called after the onCreateView() method when the host activity is created. Activity and fragment instances have been created as well as the view hierarchy of the activity. At this point, the view can be accessed with the findViewById() method. example. In this method, you can instantiate objects which require a Context object*

· *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. This is usually where you should commit any changes that should be persisted beyond the current user session.*

· *onStop() : Fragment going to be stopped by calling onStop()*

· *onDestroyView(): Fragment view will destroy after calling this method*

· *onDestroy(): called to do a final clean-up of the fragment’s state but Not guaranteed to be called by the Android platform.*

Q**uestion 16  
Activity & Fragment Lifecycle  
Scenario 1: Activity with Fragment starts and finishes  
Scenario 2: Activity with Fragment is rotated  
Scenario 3: Activity with retained Fragment is rotated**

***Answer:****You can read this awesome article to have a clear concept*<https://medium.com/androiddevelopers/the-android-lifecycle-cheat-sheet-part-iii-fragments-afc87d4f37fd>

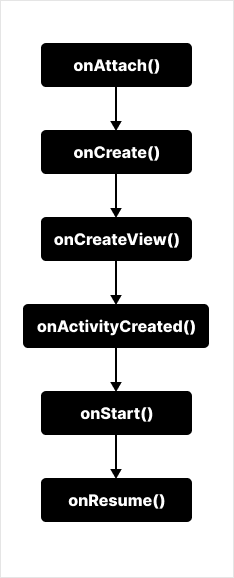
Q**uestion 17  
Difference between adding and replace fragment in backstack?**

***Answer:***· ***replace****removes the existing fragment and adds a new fragment. This means when you press the back button the fragment that got replaced will be created with its onCreateView being invoked.*

· ***add****retains the existing fragments and adds a new fragment which means the existing fragment will be active and they won't be in a ‘paused’ state hence when a back button is pressed onCreateView is not called for the existing fragment(the fragment which was there before new fragment was added).*

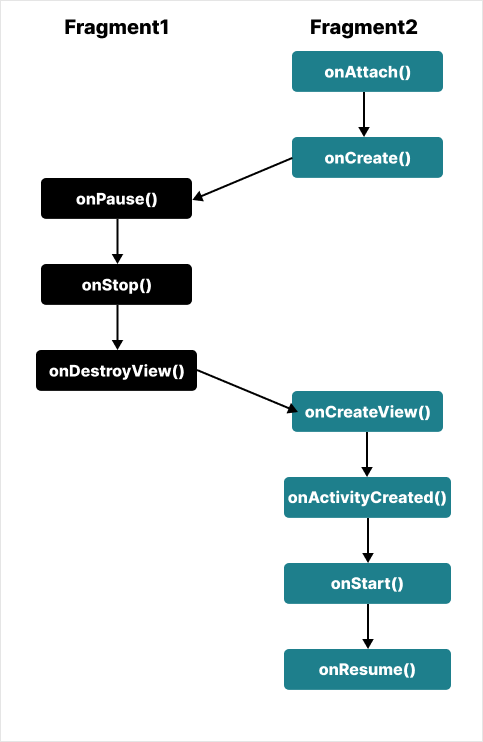
Q**uestion 18  
Suppose an activity has 2 fragments (Fragment1 & Fragment2) and we use FragmentManager to *replace* with addToBackstack each fragment to a layout in activity  
i) Lifecycle when Fragment1 is opened (using replace)?  
ii) Lifecycle when Fragment2 is opened from Fragment1 (using replace)?  
iii) Lifecycle when pop Fragment2?  
iv) Lifecycle when pop Fragment1?**

***Answer:****i) Lifecycle: Fragment1 is opened (using replace):*



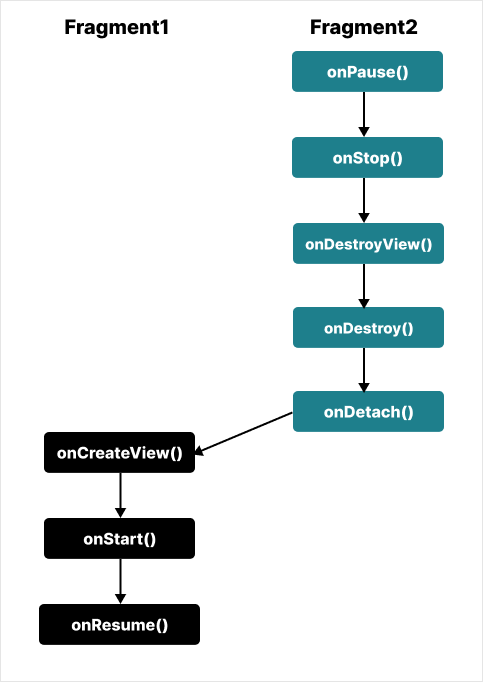
i) Lifecycle: Fragment1 is opened (using replace)

*ii) Lifecycle: Fragment2 is opened from Fragment1 (using replace):*



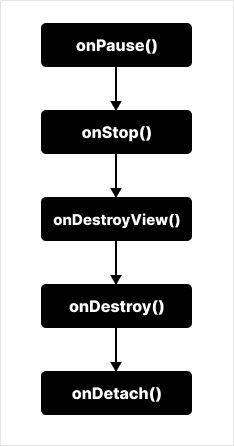
Lifecycle: Fragment2 is opened from Fragment1 (using replace)

*iii) Lifecycle: pop Fragment2:*



iii) Lifecycle: pop Fragment2

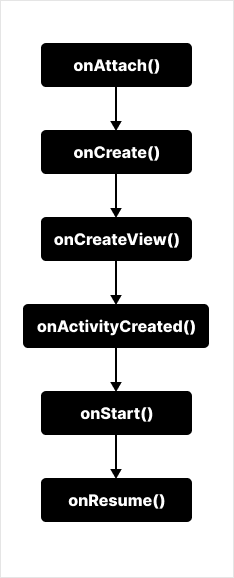
*iv) Lifecycle: pop Fragment1:*



Lifecycle: pop Fragment1

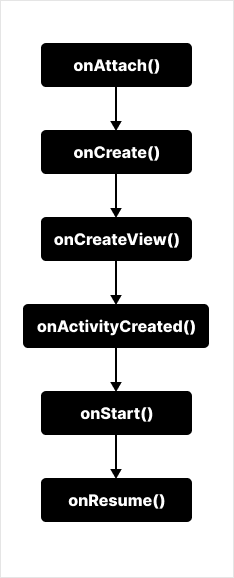
Q**uestion 19  
Suppose an activity has 2 fragments (Fragment1 & Fragment2) and we use FragmentManager to *add* with addToBackstack each fragment to a layout in activity  
i) Lifecycle when Fragment1 is opened (using add)?  
ii) Lifecycle when Fragment2 is opened from Fragment1 (using add)?  
iii) Lifecycle when pop Fragment2?  
iv) Lifecycle when pop Fragment1?**

***Answer:****i) Lifecycle when Fragment1 is opened (using add):*



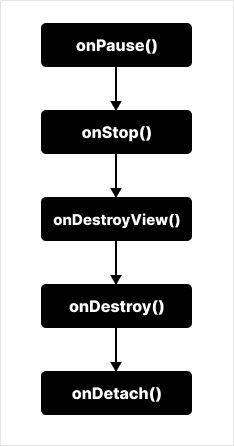
i) Lifecycle: Fragment1 is opened (using add)

*ii) Lifecycle : Fragment2 is opened from Fragment1 (using add):*



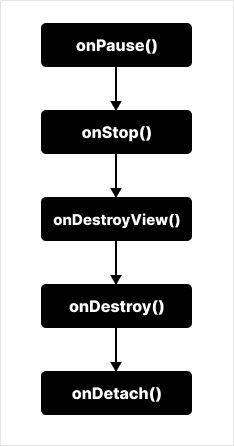
*ii) Lifecycle : Fragment2 is opened from Fragment1 (using add)*

*iii) Lifecycle : pop Fragment2:*



*iii) Lifecycle : pop Fragment2*

*iv) Lifecycle when pop Fragment1:*



*iv) Lifecycle when pop Fragment1*

Q**uestion 20  
When you show dialog on a fragment/activity which lifecycle method will be called?**

***Answer:***not lifecycle method is called