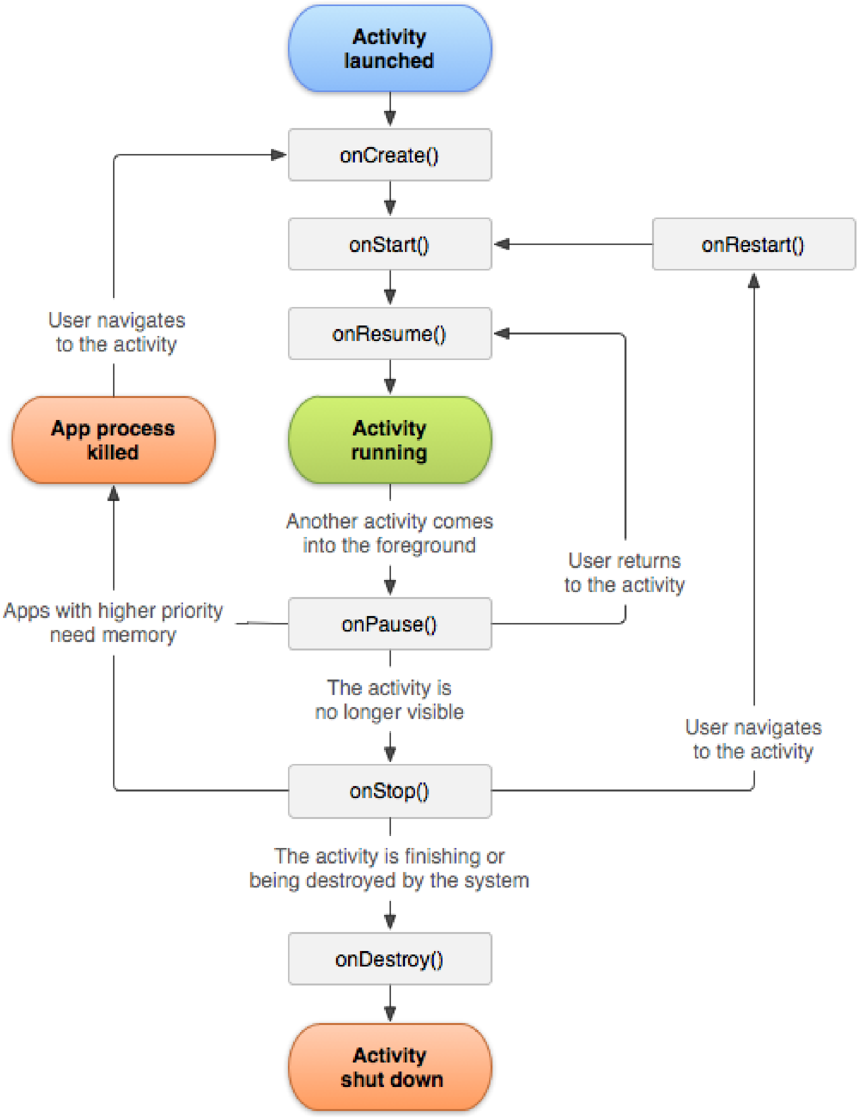
Android life Cycles and their importance

What is Android Life Cycle ?

Activity is the part of an Android app that appears in the frontend and runs in the backend. Activities allows the user to connect and use the frontend and background parts of the application. Android activity lifecycle consists of Callback methods such as onCreate(), onStart(), onResume(), onPause(), onStop()etc. While the user is using the application, the application switches between these and other life cycles. The following table shows a diagram of the Android lifecycles and the relationship between them.



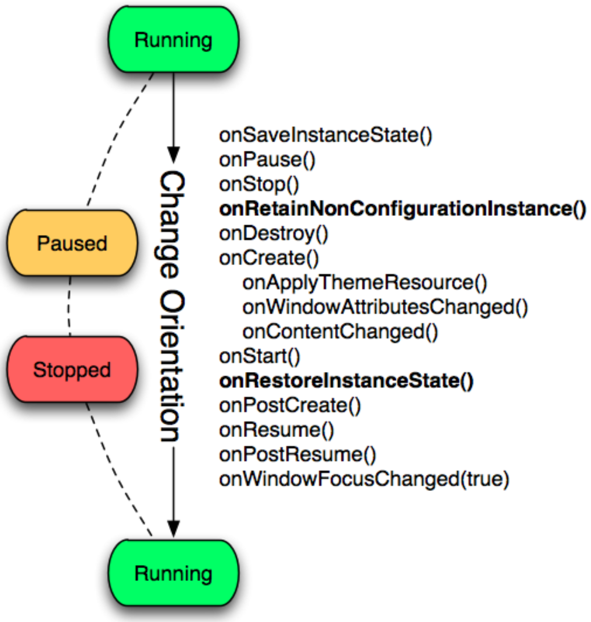
Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to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

How do we manage Android Life Cycle 's ?

We can manage Android lifecycles by using callback methods correctly and appropriately. For example, when designing the application, we should think about the possible behaviors and scenarios of the user in the application and use callback methods in the right places. As an example of this situation, when the user is using the application and his phone rings, the application can be stopped with the onPause() callback method. In addition, if the user presses the back button or if the system needs more memory and the phone rings or the application is stopped, the system stores the data from the activity in the Bundle object. This allows the System to create a new Activity and load this data when the user wants to return previous Activity. As it stores which step of the activity the user is in, it returns to that state and shows the last screen the user was on. The data protected by the system is stored in the Bundle object with key-value pairs.

What are the problems of incorrect use of Android life cycles and how can we avoid them ?

One of the most common examples of Android LifeCycles is rotateing the screen. When a user rotates their phone the configuration of the Activity changes, so android OS destroys the Activity and recreates it again. Behind the scenes; when the device orientation changes, first the Activity will disappear for a millisecond when the onPause, onStop, and onDestroy methods are called. After a few milliseconds, the activity will be restarted and the onCreate, onStart, and onResume methods are called.



As can be seen in the diagram, the onCreate method is called again when the screen is returned, which causes the previous data in the activity to be lost. To prevent this, onSaveInstanceState() can be used to save the data and the old data can be reused when the activity is called again with the onRestoreInstanceState method. Apart from rotating the screen, there can be many different scenarios, such as the phone ringing, switching from one application to another, etc. In order to prevent this and similar situations, scenarios should be designed in advance and the application should be designed in such a way that Android Lifecycle architectures and the methods used in them are used in the right places.