Activity Lifecycle in Android with Demo App

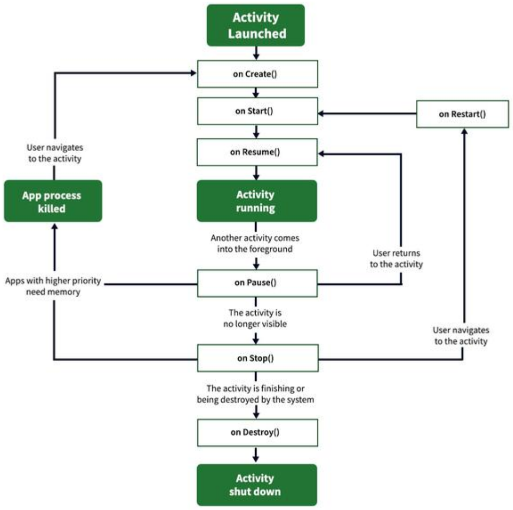
Last Updated : 08 Oct, 2024

In Android, an **activity** is referred to as one screen in an application. It is very similar to a single window of any desktop application. An Android app consists of one or more screens or activities.
Each activity goes through various stages or a lifecycle and is managed by activity stacks. So when a new activity starts, the previous one always remains below it. There are four stages of an activity.

1. If an activity is in the foreground of the screen i.e at the top of the stack, then it is said to be active or running. This is usually the activity that the user is currently interacting with.
2. If an activity has lost focus and a non-full-sized or transparent activity has focused on top of your activity. In such a case either another activity has a higher position in multi-window mode or the activity itself is not focusable in the current window mode. Such activity is completely alive.
3. If an activity is completely hidden by another activity, it is stopped or hidden. It still retains all the information, and as its window is hidden thus it will often be killed by the system when memory is needed elsewhere.
4. The system can destroy the activity from memory by either asking it to finish or simply killing its process. When it is displayed again to the user, it must be completely restarted and restored to its previous state.

For each stage, android provides us with a set of 7 methods that have their own significance for each stage in the life cycle. The image shows a path of migration whenever an app switches from one state to another.

For a deeper understanding and practical experience of handling lifecycle events in Kotlin, the Android Mastery with Kotlin: Beginner to Advanced course provides hands-on projects and in-depth lessons


Activity Launched
on Create()
on Start()
on Restart()
on Resume()
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
on Pause()
The activity is no longer visible
User navigates to the activity
on Stop()
The activity is finishing or being destroyed by the system
on Destroy()
Activity shut down

Activity Lifecycle in Android

Detailed introduction on each of the method is stated as follows:

1. onCreate()

It is called when the activity is first created. This is where all the static work is done like creating views, binding data to lists, etc. This method also provides a Bundle containing its previous frozen state, if there was one.

Example:

Java Kotlin

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

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

        // Bundle containing previous frozen state
        setContentView(R.layout.activity_main);

        // The content view pointing to the id of layout
        // in the file activity_main.xml
        Toast toast = Toast.makeText(getApplicationContext(),
"onCreate Called", Toast.LENGTH_LONG).show();
    }
}
```

2. onStart()

It is invoked when the activity is visible to the user. It is followed by onResume() if the activity is invoked from the background. It is also invoked after onCreate() when the activity is first started.

Example:

Java Kotlin

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

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

        // Bundle containing previous frozen state
        setContentView(R.layout.activity_main);

        // The content view pointing to the id of layout
        // in the file activity_main.xml
        Toast toast = Toast.makeText(getApplicationContext(),
"onCreate Called", Toast.LENGTH_LONG).show();
    }

    protected void onStart()
    {
        // It will show a message on the screen
        // then onStart is invoked
        Toast toast = Toast.makeText(getApplicationContext(),
"onStart Called", Toast.LENGTH_LONG).show();
    }
}
```

3. onRestart()

It is invoked after the activity has been stopped and prior to its starting stage and thus is always followed by onStart() when any activity is revived from background to on-screen.

Example:

Java Kotlin

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

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

        // Bundle containing previous frozen state
        setContentView(R.layout.activity_main);

        // The content view pointing to the id of layout
        // in the file activity_main.xml
        Toast toast = Toast.makeText(getApplicationContext(),
"onCreate Called", Toast.LENGTH_LONG).show();
    }

    protected void onRestart() {
        // It will show a message on the screen
        // then onRestart is invoked
        Toast toast = Toast.makeText(getApplicationContext(),
"onRestart Called", Toast.LENGTH_LONG).show();
    }
}
```

4. onResume()

It is invoked when the activity starts interacting with the user. At this point, the activity is at the top of the activity stack, with a user interacting with it. Always followed by onPause() when the activity goes into the background or is closed by the user.

Example:

Java Kotlin

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

import com.example.share.R;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        // Bundle containing previous frozen state
        super.onCreate(savedInstanceState);

        // The content view pointing to the id of layout
        // in the file activity_main.xml
        setContentView(R.layout.activity_main);

        Toast toast = Toast.makeText(getApplicationContext(),
"onCreate Called", Toast.LENGTH_LONG).show();
    }

    protected void onResume() {
        // It will show a message on the screen
        // then onResume is invoked
        Toast toast = Toast.makeText(getApplicationContext(),
"onResume Called", Toast.LENGTH_LONG).show();
    }
}
```

5. onPause()

It is invoked when an activity is going into the background but has not yet been killed. It is a counterpart to onResume(). When an activity is launched in front of another activity, this callback will be invoked on the top activity (currently on screen). The activity, under the active activity, will not be created until the active activity's onPause() returns, so it is recommended that heavy processing should not be done in this part.

Example:

Java Kotlin

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        // Bundle containing previous frozen state
        super.onCreate(savedInstanceState);

        // The content view pointing to the id of layout
        // in the file activity_main.xml
        setContentView(R.layout.activity_main);

        Toast toast = Toast.makeText(getApplicationContext(),
"onCreate Called", Toast.LENGTH_LONG).show();
    }

    protected void onPause() {
        // It will show a message on the screen
        // then onPause is invoked
        Toast toast = Toast.makeText(getApplicationContext(),
"onPause Called", Toast.LENGTH_LONG).show();
    }
}
```

6. onStop()

It is invoked when the activity is not visible to the user. It is followed by **onRestart()** when the activity is revoked from the background, followed by onDestroy() when the activity is closed or finished, and nothing when the activity remains on the background only. Note that this method may never be called, in low memory situations where the system does not have enough memory to keep the activity's process running after its onPause() method is called.

Example:

Java Kotlin

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        // Bundle containing previous frozen state
        super.onCreate(savedInstanceState);

        // The content view pointing to the id of layout
        // in the file activity_main.xml
        setContentView(R.layout.activity_main);

        Toast toast = Toast.makeText(getApplicationContext(),
"onCreate Called", Toast.LENGTH_LONG).show();
    }
    protected void onStop() {
        // It will show a message on the screen
        // then onStop is invoked
        Toast toast = Toast.makeText(getApplicationContext(),
"onStop Called", Toast.LENGTH_LONG).show();
    }
}
```

7. onDestroy()

The final call received before the activity is destroyed. This can happen either because the activity is finishing (when finish() is invoked) or because the system is temporarily destroying this instance of the activity to save space. To distinguish between these scenarios, check it with **isFinishing()** method.

Example:

Java Kotlin

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        // Bundle containing previous frozen state
        super.onCreate(savedInstanceState);

        // The content view pointing to the id of layout
        // in the file activity_main.xml
        setContentView(R.layout.activity_main);

        Toast toast = Toast.makeText(getApplicationContext(),
"onCreate Called", Toast.LENGTH_LONG).show();
    }

    protected void onDestroy() {
        // It will show a message on the screen
        // then onDestroy is invoked
        Toast toast = Toast.makeText(getApplicationContext(),
"onDestroy Called", Toast.LENGTH_LONG).show();
    }
}
```

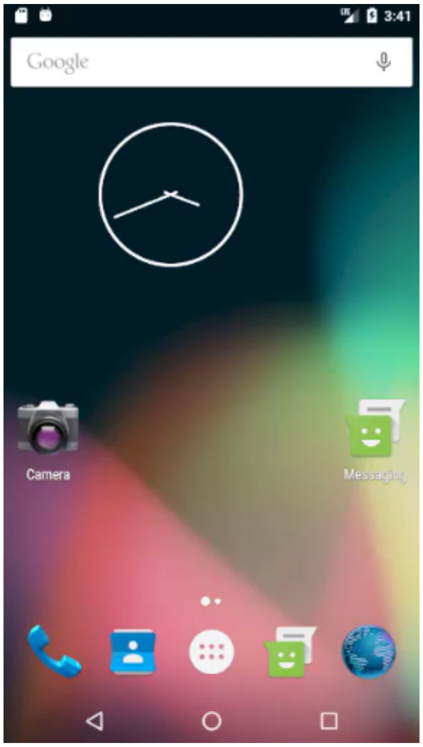
Demo Android App to Demonstrate Activity Lifecycle in Android

Java   Kotlin

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toast toast = Toast.makeText(getApplicationContext(),
"onCreate Called", Toast.LENGTH_LONG).show();
    }

    protected void onStart() {
        super.onStart();
        Toast toast = Toast.makeText(getApplicationContext(),
"onStart Called", Toast.LENGTH_LONG).show();
    }

    @Override
    protected void onRestart() {
        super.onRestart();
        Toast toast = Toast.makeText(getApplicationContext(),
"onRestart Called", Toast.LENGTH_LONG).show();
    }

    protected void onPause() {
        super.onPause();
        Toast toast = Toast.makeText(getApplicationContext(),
"onPause Called", Toast.LENGTH_LONG).show();
    }

    protected void onResume() {
        super.onResume();
        Toast toast = Toast.makeText(getApplicationContext(),
"onResume Called", Toast.LENGTH_LONG).show();
    }

    protected void onStop() {
        super.onStop();
```

```
            Toast toast = Toast.makeText(getApplicationContext(),
"onStop Called", Toast.LENGTH_LONG).show();
        }

        protected void onDestroy() {
            super.onDestroy();
            Toast toast = Toast.makeText(getApplicationContext(),
"onDestroy Called", Toast.LENGTH_LONG).show();
        }
    }
```

Output:



Looking to become an expert in Android App Development? Whether you're a student or a professional aiming to advance your career in mobile app development, our course, "[Android App Development with Kotlin](#)," available exclusively on GeeksforGeeks, is the perfect fit for you. Gain hands-on experience with **Kotlin**, the modern language preferred by Android developers worldwide. This course will guide you through the essentials of Android app development, from the **basics to advanced techniques**, using practical projects and real-world scenarios. Ideal for beginners and those looking to enhance their existing skills, this course will equip you with the expertise needed to build high-quality, piyus... 34

Previous Article

How Does Android App Work?

Next Article

Introduction to Gradle

Video | The Activity Lifecycle

Similar Reads

Understanding Activity Lifecycle to Retain UI Data when Back Presse...

What is onBackPressed() in Android? This is an override function called when the user presses the back button on an Android device. It has great...

5 min read

How to Pass a Serializable Object from One Activity to Another Activi…

While developing an application in Android, the developer can create more than one activity in the application. So, there can be many activities and...

4 min read

How to Send Data From One Activity to Second Activity in Android?

This article aims to tell and show how to "Send the data from one activity to second activity using Intent" . In this example, we have two activities,...

7 min read

How to Use Android Sliding Activity Library in Android App?

Sliding activities allow you to easily set header content, menus, and data onto slidable screens. Easily create activities that can slide vertically on t...

3 min read

How to Send Image File from One Activity to Another Activity?

In this article, we are going to send an image from one activity to another. We will be using putExtra to send image and we will be using bundle to...

3 min read

Intent Filter in Android with Demo App

The intent is a messaging object which tells what kind of action to be performed. The intent's most significant use is the launching of the activit...

5 min read

Processes and Application Lifecycle in Android

As an android developer, if one does not know the application lifecycle of android application or does not have in-depth knowledge about it, there...

7 min read

Lifecycle in Android Architecture Components

Lifecycle is one of the Android Architecture Components which was released by Google to make it easier for all the Android developers. The...

4 min read

Fragment Lifecycle in Android

In Android, the fragment is the part of the Activity that represents a portion of the User Interface(UI) on the screen. It is the modular section o...

8 min read

Android - You need to use a Theme.AppCompat theme (or descenda...

You need to use a Theme.AppCompat theme (or descendant) with this activity is a common error message that appears when trying to run an...

4 min read

Article Tags : Android Java

Practice Tags : Java

Corporate & Communications Address:-
A-143, 9th Floor, Sovereign Corporate
Tower, Sector- 136, Noida, Uttar Pradesh
(201305) | Registered Address:- K 061,
Tower K, Gulshan Vivante Apartment,

Sector 137, Noida, Gautam Buddh
Nagar, Uttar Pradesh, 201305

Company

- About Us
- Legal
- Careers
- In Media
- Contact Us
- Advertise with us
- GFG Corporate Solution
- Placement Training Program

Explore

- Job-A-Thon Hiring Challenge
- Hack-A-Thon
- GfG Weekly Contest
- Offline Classes (Delhi/NCR)
- DSA in JAVA/C++
- Master System Design
- Master CP
- GeeksforGeeks Videos
- Geeks Community

Languages

- Python
- Java
- C++
- PHP
- GoLang
- SQL
- R Language
- Android Tutorial

DSA

- Data Structures
- Algorithms
- DSA for Beginners
- Basic DSA Problems
- DSA Roadmap
- DSA Interview Questions
- Competitive Programming

Data Science & ML

- Data Science With Python
- Data Science For Beginner
- Machine Learning
- ML Maths
- Data Visualisation
- Pandas
- NumPy
- NLP
- Deep Learning

Web Technologies

- HTML
- CSS
- JavaScript
- TypeScript
- ReactJS
- NextJS
- NodeJs
- Bootstrap
- Tailwind CSS

Python Tutorial

- Python Programming Examples
- Django Tutorial
- Python Projects

Computer Science

- GATE CS Notes
- Operating Systems
- Computer Network

Python Tkinter
Web Scraping
OpenCV Tutorial
Python Interview Question

Database Management System
Software Engineering
Digital Logic Design
Engineering Maths

DevOps

Git
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

System Design

High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns
OOAD
System Design Bootcamp
Interview Questions

School Subjects

Mathematics
Physics
Chemistry
Biology
Social Science
English Grammar

Commerce

Accountancy
Business Studies
Economics
Management
HR Management
Finance
Income Tax

Databases

SQL
MYSQL
PostgreSQL
PL/SQL
MongoDB

Preparation Corner

Company-Wise Recruitment Process
Resume Templates
Aptitude Preparation
Puzzles
Company-Wise Preparation
Companies
Colleges

Competitive Exams

JEE Advanced
UGC NET
UPSC
SSC CGL
SBI PO
SBI Clerk
IBPS PO
IBPS Clerk

More Tutorials

Software Development
Software Testing
Product Management
Project Management
Linux
Excel
All Cheat Sheets
Recent Articles

Free Online Tools

Typing Test

Write & Earn

Write an Article