



**Android**

**Environment & Lifecycle**

## Daftar Isi

A. Pengenalan Android	3
B. Pengenalan JDK	3
C. Pengenalan Android Studio	3
D. Persyaratan Sistem Operasi untuk Instalasi Android Studio	4
E. Instalasi Android Studio pada Windows	6
F. Apa itu Gradle?	8
G. Instalasi Postman	9
H. Komponen Aplikasi Android	11
I. Activity	12
J. Lifecycle Activity	13
References	20

## A. Pengenalan Android

Android adalah sistem operasi berbasis Linux yang dimodifikasi untuk perangkat bergerak (mobile device) yang terdiri dari sistem operasi, middleware, dan aplikasi-aplikasi utama. Awalnya, Android dikembangkan oleh Android Inc. Perusahaan ini kemudian dibeli oleh Google pada tahun 2005.

Dalam pengembangannya, penamaan versi dari Android menggunakan nama-nama makanan manis. Bukan hanya itu, setiap perkembangan Android, berbagai konsep, tampilan, dan juga fungsi-fungsi yang baru juga ditambahkan pada sistem operasinya agar meningkatkan performa dan user experience. Saat ini sudah banyak produsen yang telah menggunakan Android sebagai sistem operasi untuk perangkat mobile yang mereka produksi.

## B. Pengenalan JDK

JDK merupakan singkatan dari Java Development Kit. JDK menyediakan compiler, library, API, dan tools lainnya. JDK menyediakan segala hal yang dibutuhkan untuk membuat aplikasi dengan bahasa pemrograman berbasis Java.

## C. Pengenalan Android Studio

Android Studio adalah Lingkungan Pengembangan Terpadu – Integrated Development Environment (IDE) untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA. Selain merupakan editor kode IntelliJ dan alat

pengembang yang berdaya guna, Android Studio menawarkan fitur lebih banyak untuk meningkatkan produktivitas developer saat membuat aplikasi Android, misalnya :

1. Sistem versi berbasis Gradle yang fleksibel
2. Emulator yang cepat dan kaya fitur sehingga developer dapat menjalankan aplikasi yang sedang mereka bangun tanpa harus menggunakan device Android secara fisik.
3. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat Android
4. Instant Run untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru
5. Template kode dan integrasi GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh
6. Alat pengujian dan kerangka kerja yang ekstensif
7. Alat Lint untuk meningkatkan kinerja, kegunaan, kompatibilitas versi, dan masalah-masalah lain
8. Dukungan C++ dan NDK
9. Dukungan bawaan untuk Google Cloud Platform, mempermudah pengintegrasian Google Cloud Messaging dan App Engine

## D. Persyaratan Sistem Operasi untuk Instalasi Android Studio

Berikut ini adalah persyaratan sistem untuk Android Studio di **Windows** :

- Microsoft® Windows® 8/10 (64-bit)

- x86\_64 CPU architecture; Intel Core 2nd Gen atau lebih, atau AMD CPU dengan support Windows Hypervisor
- RAM 8 GB atau lebih
- Ruang disk minimum yang tersedia 8 GB (IDE + Android SDK + Android Emulator)
- Resolusi layar minimum 1280 x 800

Berikut ini adalah persyaratan sistem untuk Android Studio di **macOS** :

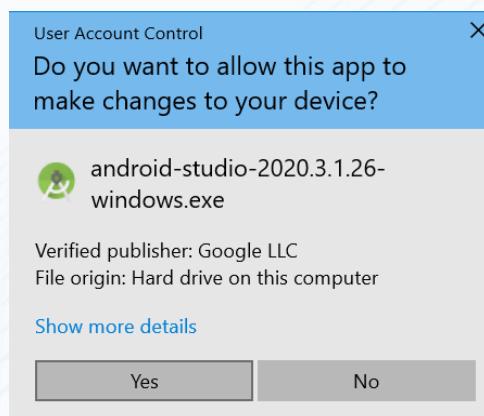
- MacOS® 10.14 (Mojave) atau lebih baru
- ARM-based chips, atau Intel Core 2nd Gen atau lebih dengan support Hypervisor.Framework
- RAM 8 GB atau lebih
- Ruang disk minimum yang tersedia 8 GB (IDE + Android SDK + Android Emulator)
- Resolusi layar minimum 1280 x 800

Berikut ini adalah persyaratan sistem untuk Android Studio di **Linux** :

- Linux 64-bit yang support GNOME, KDE, atau Unity DE, GNU C Library (glibc) 2.31 atau lebih
- x86\_64 CPU architecture; Intel Core atau lebih 2nd Gen, atau AMD processor dengan support AMD Virtualization (AMD-V) dan SSSE3
- RAM 8 GB atau lebih
- Ruang disk minimum yang tersedia 8 GB (IDE + Android SDK + Android Emulator)
- Resolusi layar minimum 1280 x 800

## E. Instalasi Android Studio pada Windows

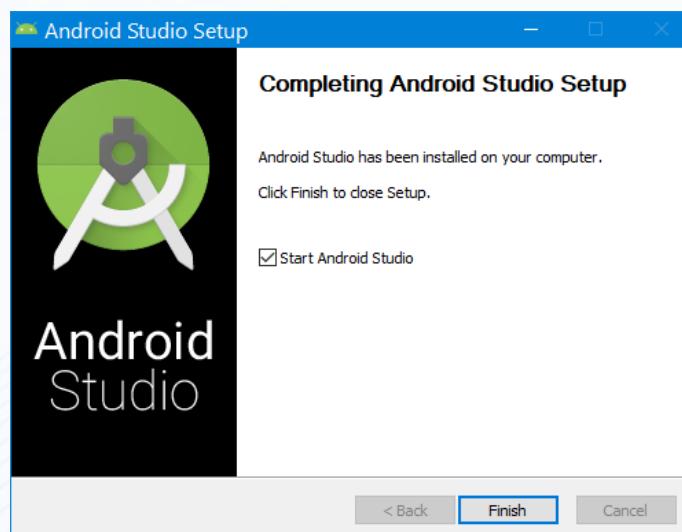
1. Buka web Android Studio pada halaman <https://developer.android.com/studio>. Halaman ini secara otomatis akan mendeteksi sistem operasi yang digunakan.
2. Klik “Download Android Studio” kemudian baca dan setujui license agreement yang ada di bagian bawah.
3. Klik “Download Android Studio .... for ...” untuk memulai proses download dan tentukan lokasi penyimpanan kemudian tunggu proses download selesai.
4. Setelah proses download selesai, install Android Studio dengan menjalankan file instalasi.
5. Izinkan penginstalan dengan mengklik Ya pada dialog “User Account Control seperti gambar di bawah ini.



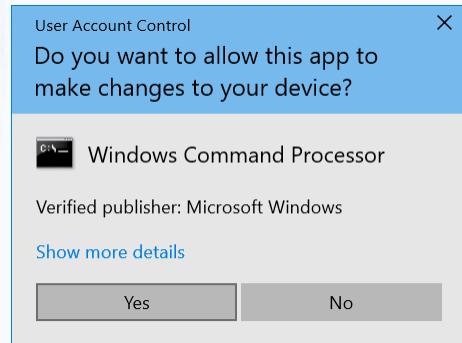
6. Dialog “Welcome to Android Studio Setup” muncul. Klik next untuk memulai proses instalasi.



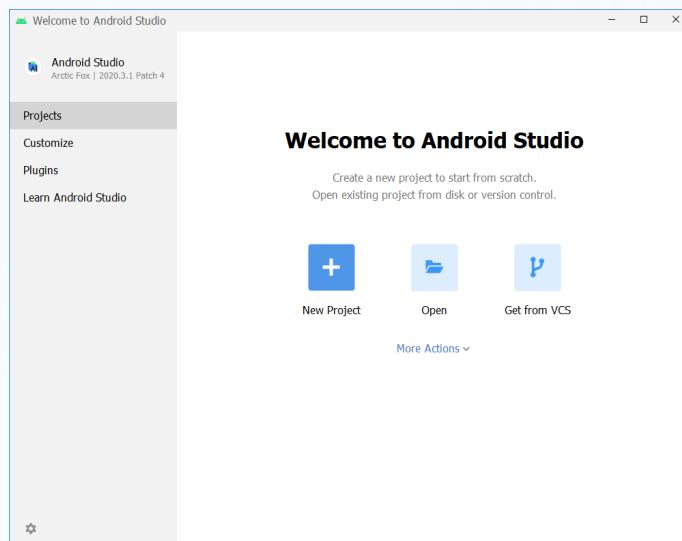
7. Setujui default instalasi untuk semua langkah.
8. Klik Finish ketika instalasi selesai untuk menjalankan Android Studio.



9. Pilih preferensi tema untuk Android Studio.
10. Selanjutnya Android Studio akan menginstal beberapa komponen pendukung seperti ADB dan SDK. Android SDK merupakan sekumpulan tools yang digunakan untuk mengembangkan aplikasi Android. Klik “Ya” pada dialog untuk memulai proses instalasi komponen-komponen tersebut.



11. Jika proses instalasi komponen-komponen tersebut sudah selesai, maka akan muncul window “Welcome to Android Studio” seperti gambar di bawah ini.



Untuk tahapan instalasi pada sistem operasi lain seperti MacOS dan Linux dapat dilihat pada halaman berikut :

<https://developer.android.com/codelabs/basic-android-kotlin-compose-install-android-studio#4>

## F. Apa itu Gradle?

Gradle merupakan sebuah program yang dirancang khusus untuk melakukan build secara otomatis. Oleh karena itu Gradle juga bisa disebut sebagai build-tool. Gradle didesain agar dapat membuat multi-project yang memiliki

skala besar. Selain Java gradle juga dapat dijalankan dengan bahasa pemrograman lainnya contohnya seperti C++, JavaScript, Kotlin, Groovy, dan juga lain-lain. Fungsi dari Gradle adalah sebagai berikut :

1. Gradle dapat membantu menghindari pekerjaan yang tidak perlu dengan hanya menjalankan pekerjaan yang perlu dikerjakan saja.
2. Gradle membantu mengatur dependencies atau library apa saja yang ingin dan dapat digunakan oleh programer.
3. Mengunduh berbagai macam pustaka atau library yang diperlukan untuk membangun suatu project.

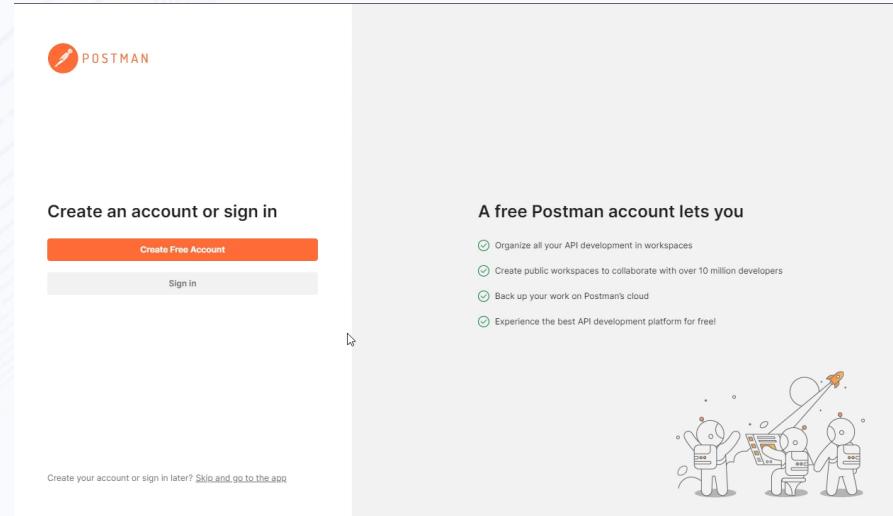
## G. Instalasi Postman

Postman adalah developing tools yang membantu penggunanya untuk membangun, menguji, dan memodifikasi API. Ia menawarkan para developer berbagai fitur dan fungsi yang penting sehingga kinerjanya dapat berlangsung mudah dan sederhana. Ketika menjalankan pengujian, Postman mengirim request API ke server web dan kemudian menerima segala jenis respons. Sesuai dengan namanya, ia berfungsi layaknya tukang pos.

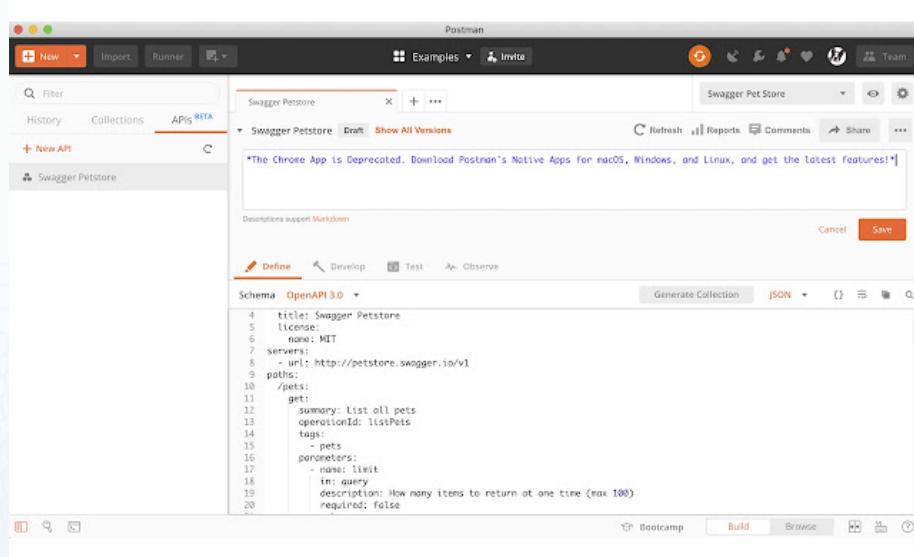
Aplikasi ini dapat mengakomodasi berbagai jenis request HTTP, menyimpan lingkungan untuk penggunaan selanjutnya. Postman biasa digunakan oleh developer pembuat API sebagai tools untuk menguji API yang telah mereka buat.

Untuk menggunakan Postman, download file instalasi pada halaman berikut : <https://www.postman.com/downloads/>. Kemudian, buka file installer yang berformat .exe dan jalankan.

Setelah proses instalasi selesai, jalankan Postman sehingga akan muncul window sebagai berikut.



Jika ingin membuat akun baru, maka isi data-data yang tersedia pada tampilan awal aplikasi atau masuk secara langsung. Namun, penggunaan tanpa masuk akun juga memungkinkan. Berikut ini merupakan tampilan Postman.



## H. Komponen Aplikasi Android

Aplikasi Android ditulis dengan bahasa pemrograman Java atau Kotlin. Semua file kode intermediate dari asset disatukan dalam satu paket berupa file berekstensi .APK atau .AAB, sebuah file yang dapat didistribusi. Tiap file .APK atau .ABB adalah sebuah aplikasi Android tunggal. Komponen aplikasi Android terdiri dari beberapa jenis, antara lain :

### 1. Activity

Activity adalah istilah yang digunakan dalam pemrograman Android untuk mengacu pada satuan interaksi dengan user melalui antarmuka grafis (graphical user-interface, GUI). Sebagai satuan interaksi, Activity adalah tampilan yang terlihat di layar seperti Windows atau kotak dialog pada pemrograman aplikasi desktop. Tiap aplikasi dapat terdiri dari nol atau lebih Activity. Selain sebagai satuan interaksi dengan user, Activity juga satuan eksekusi. Sebagai satuan eksekusi, Activity selalu memiliki paling tidak satu buah Thread, yakni Thread utama yang digunakan untuk memperbarui tampilan user interface (UI Thread).

### 2. Intent

Intent adalah istilah yang digunakan dalam pemrograman Android untuk mengacu pada mekanisme berbagai pesan pemberitahuan atau bertukar data Activity atau untuk menjalankan aplikasi lain.

### 3. Service

Service adalah komponen aplikasi yang berjalan di belakang layar tanpa user interface untuk menyediakan layanan tertentu seperti mengecek RSS feed secara kontinu atau memainkan musik. Service tetap berjalan

meski Activity yang mengendalikannya telah berhenti. Media player adalah sebuah contoh aplikasi yang menggunakan Service.

#### 4. Content Provider

Content Provider membuat suatu aplikasi dapat berbagi sejumlah data tertentu kepada aplikasi lain. Jika membutuhkan data nama-nama kontak, aplikasi tinggal meminta data tersebut.

#### 5. Broadcast Receiver

Broadcast Receiver adalah komponen yang memantau, menerima, dan bereaksi terhadap pesan yang disebarluaskan, baik oleh sistem maupun aplikasi lain. Misalnya, ketika baterai lemah, Android akan mengirim pesan “baterai lemah” kepada semua broadcast receiver yang ingin diberitahu pesan ini. Untuk menggunakan broadcast receiver, pada dasarnya, yang diperlukan hanya membuat turunan tipe broadcast receiver, melengkapi metode `onReceive()`, dan mendaftarkannya di `AndroidManifest.xml` atau dengan metode `Context.registerReceiver()`. Instance broadcast receiver hanya valid selama pemanggilan metode `onReceive()` sehingga referensi tidak boleh disimpan ke instance ini.

## I. Activity

Activity adalah istilah yang digunakan dalam pemrograman Android untuk mengacu pada satuan interaksi dengan user melalui antarmuka grafis (graphical user-interface, GUI). Sebagai satuan interaksi, Activity adalah tampilan yang terlihat di layar seperti Windows atau kotak dialog pada pemrograman aplikasi desktop. Tiap aplikasi dapat terdiri dari nol atau lebih Activity. Selain sebagai satuan interaksi dengan user, Activity juga satuan

eksekusi. Sebagai satuan eksekusi, Activity selalu memiliki paling tidak satu buah Thread, yakni Thread utama yang digunakan untuk memperbarui tampilan user-interface (UI Thread).

Ketika membuat project baru di Android Studio, biasanya terdapat dua file yang sudah terbuat otomatis, yaitu file MainActivity dan activity\_main.xml. MainActivity disebut sebagai class Activity karena mewarisi (extends) superclass Activity. Class ini menampilkan layout yang terdapat pada file activity\_main.xml dan mengelola interaksi yang ada di dalamnya.

Umumnya dalam sebuah aplikasi Android terdapat lebih dari satu Activity yang saling berhubungan dengan tugas yang beragam. Yang perlu diperhatikan yaitu setiap Activity harus terdaftar di file AndroidManifest.xml. Secara default, Activity akan didaftarkan jika Activity baru terbuat dengan cara otomatis. Caranya yaitu klik kanan pada nama package → New → Activity → pilih template Activity yang tersedia.

## J. Lifecycle Activity

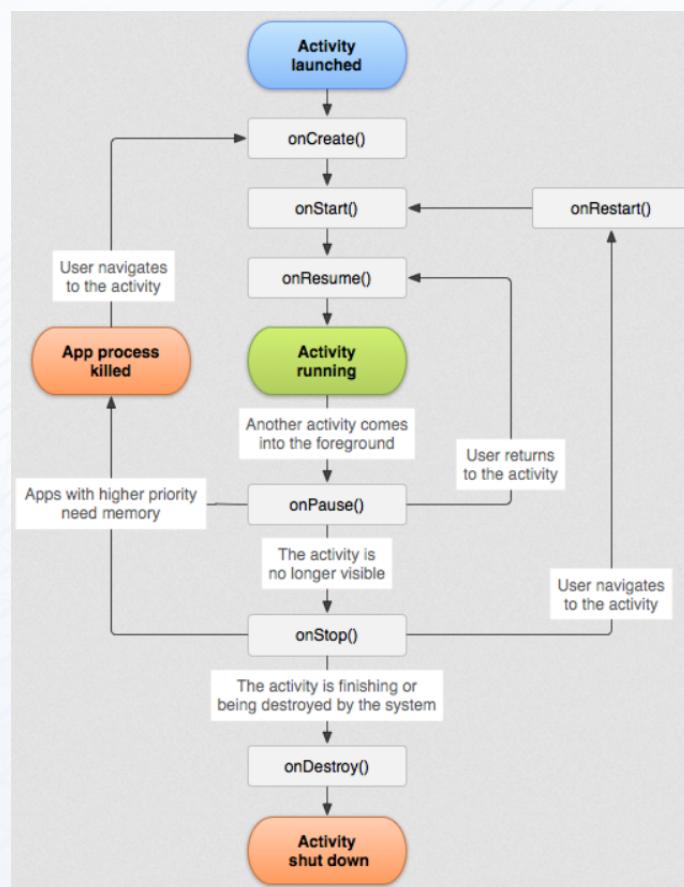
Saat user menelusuri, keluar, dan kembali ke aplikasi, instance Activity dalam aplikasi melakukan transisi ke berbagai status dalam siklus prosesnya. Class Activity menyediakan sejumlah callback yang memungkinkan activity mengetahui bahwa status telah berubah: bahwa sistem membuat, menghentikan, atau melanjutkan suatu activity , atau menutup proses tempat beradanya activity.

Dalam metode callback siklus proses, kita dapat mendeklarasikan cara activity berperilaku saat user meninggalkan dan memasuki kembali activity itu. Misalnya, jika sedang mem-build pemutar video streaming, kita dapat

menghentikan sementara video itu dan mengakhiri koneksi jaringan saat user beralih ke aplikasi lain. Saat user kembali ke aplikasi, kita dapat menghubungkan ulang ke jaringan dan memungkinkan user melanjutkan video tersebut dari titik yang sama. Dengan kata lain, setiap callback memungkinkan kita melakukan pekerjaan tertentu yang sesuai dengan perubahan status yang diberikan. Melakukan pekerjaan yang tepat dan pada waktu yang tepat, serta menangani transisi dengan benar membuat aplikasi yang kita kembangkan menjadi lebih andal dan efektif.

Untuk menavigasi transisi di antara tahap siklus proses activity, class Activity menyediakan set inti sebanyak enam callback: **onCreate()**, **onStart()**, **onResume()**, **onPause()**, **onStop()**, dan **onDestroy()**. Sistem memanggil masing-masing callback ini saat activity memasuki status baru.

Berikut ini merupakan gambaran dari siklus dan callback sebuah Activity :



## 1. Callback onCreate()

Kita harus menerapkan callback ini saat sistem pertama kali membuat activity. Pada pembuatan activity, activity memasuki status Dibuat. Dalam metode `onCreate()`, kita menjalankan logika startup aplikasi dasar yang hanya boleh terjadi sekali selama siklus activity. Misalnya, implementasi `onCreate()` mungkin mengikat data ke daftar, mengaitkan activity dengan `ViewModel`, dan membuat instance beberapa variabel lingkup class. Metode ini menerima parameter `savedInstanceState`, yang merupakan objek `Bundle` yang berisi status activity yang sebelumnya disimpan. Jika activity belum pernah ada sebelumnya, nilai objek `Bundle` adalah nol.

Jika kita memiliki komponen berbasis siklus proses yang terhubung dengan siklus proses activity kita, activity akan menerima peristiwa `ON_CREATE`. Metode yang dijelaskan dengan `@OnLifecycleEvent` akan dipanggil sehingga komponen berbasis siklus proses kita dapat melakukan kode penyiapan apa pun yang diperlukan untuk status yang dibuat.

Dalam metode `onCreate()` biasanya terdapat penyiapan dasar untuk activity, seperti mendeklarasikan antarmuka pengguna (didefinisikan dalam file tata letak XML), mendefinisikan variabel anggota, dan mengkonfigurasi beberapa UI.

## 2. Callback onStart()

Ketika activity memasuki status Dimulai, sistem memanggil callback ini. Panggilan `onStart()` membuat activity terlihat oleh user, saat aplikasi mempersiapkan activity untuk memasuki latar depan dan menjadi interaktif. Misalnya, metode ini adalah tempat aplikasi menginisialisasi kode yang mengelola UI.

Saat activity berpindah ke status dimulai, komponen berbasis siklus proses apa pun yang terkait dengan siklus proses activity akan menerima peristiwa ON\_START. Metode onStart() selesai dengan sangat cepat dan, seperti pada status Dibuat, activity tidak tetap berada dalam status Dimulai. Setelah callback ini selesai, activity memasuki status Dilanjutkan, dan sistem memanggil metode onResume().

### 3. Callback onResume()

Setelah activity memasuki status Dilanjutkan, activity tersebut masuk ke latar depan, kemudian sistem memanggil callback onResume(). Ini adalah status saat aplikasi berinteraksi dengan user. Aplikasi tetap dalam status ini sampai terjadi sesuatu untuk mengambil fokus dari aplikasi. Peristiwa yang terjadi misalnya adalah menerima panggilan telepon, user beralih ke activity lain, atau layar perangkat mati.

Saat activity berpindah ke status dilanjutkan, komponen berbasis siklus proses apa pun yang terkait dengan siklus proses activity akan menerima peristiwa ON\_RESUME. Di sinilah komponen siklus proses dapat mengaktifkan fungsi apa pun yang perlu dijalankan saat komponen terlihat dan berada di latar depan, seperti memulai pratinjau kamera.

Jika terjadi suatu peristiwa interupsi, activity memasuki status Dijeda, dan sistem memanggil callback onPause(). Jika activity kembali ke status Dilanjutkan dari status Dijeda, sistem akan memanggil metode onResume() sekali lagi. Untuk alasan ini, kita harus menerapkan onResume() untuk menginisialisasi komponen yang kita rilis selama onPause(), dan melakukan inisialisasi lainnya yang harus terjadi setiap kali activity memasuki status Dilanjutkan.

#### 4. Callback onPause()

Sistem akan memanggil metode ini sebagai indikasi pertama bahwa user meninggalkan activity (meskipun tidak selalu berarti activity sedang ditutup); hal ini menunjukkan bahwa activity tidak lagi di latar depan (meskipun mungkin masih terlihat jika user berada dalam mode multi-jendela). Gunakan metode onPause() untuk menjeda atau menyesuaikan operasi yang tidak boleh dilanjutkan (atau harus dilanjutkan dalam jumlah sedang) sementara Activity berada dalam status Dijeda, dan kita berharap untuk segera melanjutkan.

Saat activity berpindah ke status dijeda, komponen berbasis siklus proses yang terkait dengan siklus proses activity akan menerima peristiwa ON\_PAUSE. Di sinilah komponen siklus proses dapat menghentikan fungsi apa pun yang tidak perlu dijalankan saat komponen tidak ada di latar depan, seperti menghentikan pratinjau kamera.

Kita juga dapat menggunakan metode onPause() untuk melepaskan resource sistem, menangani sensor (seperti GPS), atau resource apa pun yang dapat memengaruhi masa pakai baterai saat activity dijeda dan user tidak membutuhkannya. Namun, seperti yang disebutkan di atas di bagian onResume(), activity yang Dijeda mungkin masih sepenuhnya terlihat jika berada dalam mode multi-jendela. Karena itu, kita harus mempertimbangkan menggunakan onStop() daripada onPause() untuk sepenuhnya melepaskan atau menyesuaikan resource dan operasi terkait UI untuk lebih optimal mendukung mode multi-jendela.

## 5. Callback onStop()

Jika activity tidak lagi terlihat oleh user, activity tersebut telah memasuki status Berhenti, dan sistem memanggil callback onStop(). Ini dapat terjadi, misalnya, ketika activity yang baru diluncurkan menutupi seluruh layar. Sistem juga dapat memanggil onStop() ketika activity telah selesai berjalan, dan akan segera dihentikan.

Saat activity beralih ke status berhenti, komponen berbasis siklus proses yang terkait dengan siklus proses activity akan menerima peristiwa ON\_STOP. Di sinilah komponen siklus proses dapat menghentikan fungsi apa pun yang tidak perlu dijalankan saat komponen tidak terlihat di layar.

Dalam metode onStop(), aplikasi harus melepaskan atau menyesuaikan resource yang tidak diperlukan saat aplikasi tidak terlihat oleh user. Misalnya, aplikasi dapat menjeda animasi atau beralih dari pembaruan lokasi yang sangat akurat ke kurang akurat. Menggunakan onStop(), dan bukannya onPause() akan memastikan bahwa pekerjaan terkait UI berlanjut, meski user melihat activity dalam mode multi-jendela.

## 6. Callback onDestroy()

onDestroy() dipanggil sebelum activity ditutup. Sistem memanggil callback ini karena:

aktivitas selesai (karena user benar-benar menutup activity atau karena finish() dipanggil pada activity tersebut), atau  
sistem sementara menutup activity karena perubahan konfigurasi (seperti rotasi perangkat atau mode multi-jendela)

Saat activity berpindah ke status ditutup, komponen berbasis siklus proses apa pun yang terkait dengan siklus proses activity akan menerima peristiwa

ON\_DESTROY. Di sinilah komponen siklus proses dapat membersihkan apa pun yang diperlukan sebelum activity ditutup.

Jika activity selesai, `onDestroy()` adalah callback siklus proses terakhir yang diterima activity. Jika `onDestroy()` dipanggil sebagai hasil dari perubahan konfigurasi, sistem akan segera membuat instance activity baru kemudian memanggil `onCreate()` pada instance baru dalam konfigurasi baru. Callback `onDestroy()` harus melepaskan semua resource yang belum dirilis oleh callback sebelumnya seperti `onStop()`.

## References

- [https://developer.android.com/codelabs/basic-android-kotlin-compose-inst  
all-android-studio](https://developer.android.com/codelabs/basic-android-kotlin-compose-install-android-studio)
- <https://www.petanikode.com/java-linux/>
- <https://developer.android.com/studio/intro/?hl=id>
- [https://binus.ac.id/bandung/2022/10/perkembangan-operating-system-an  
droid/](https://binus.ac.id/bandung/2022/10/perkembangan-operating-system-android/)
- [https://idmetafora.com/news/read/1064/Apa-itu-Gradle-Pada-Android-Stu  
dio-Mari-Simak-Pembahasannya.html](https://idmetafora.com/news/read/1064/Apa-itu-Gradle-Pada-Android-Stu<br/>dio-Mari-Simak-Pembahasannya.html)
- <https://www.sekawanmedia.co.id/blog/postman-api/>
- <https://developer.android.com/guide/components/activities/intro-activities>