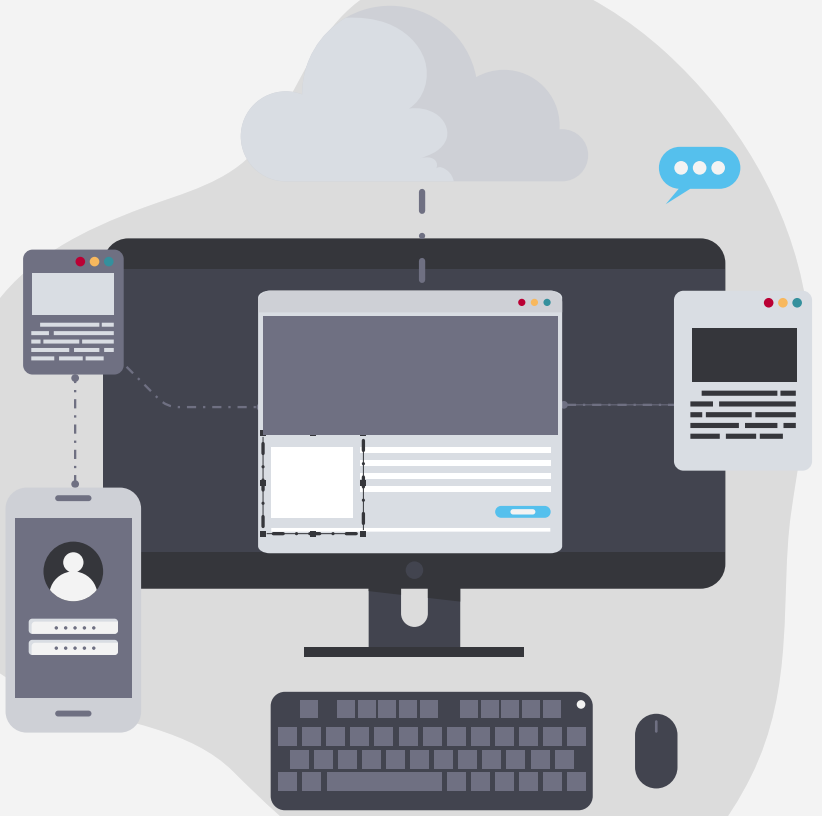
Pemrograman Mobile

Episode - 2



Apa bahasan kita hari ini?

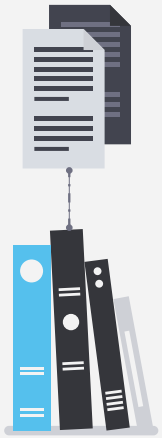
Pembahasan Hari Ini

01

Introduction
to Kotlin

02

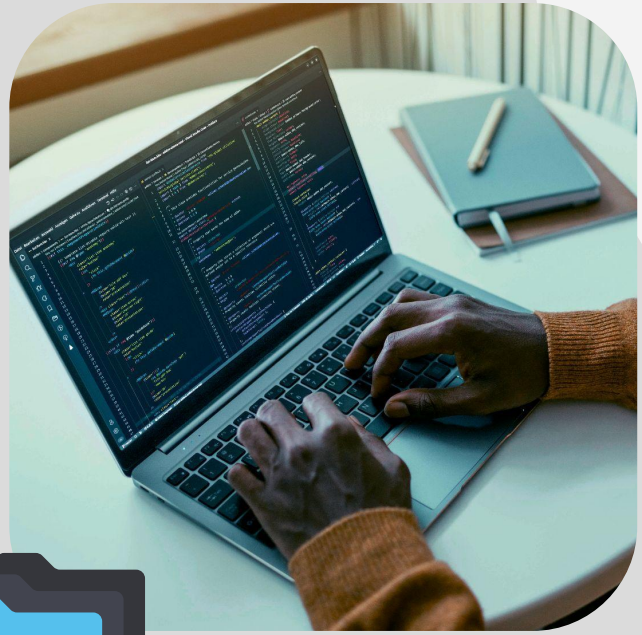
Struktur Aplikasi dan
Activity pada Android

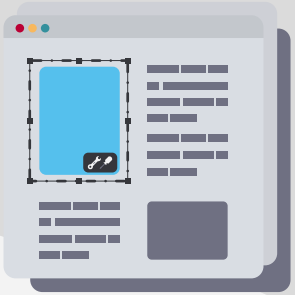


Clan’s Idea Presentation

Mempresentasikan ide *clan* di depan kelas dengan durasi **maksimal 5 menit** dengan bahasan:

- **Rancangan ide**
  (latar belakang, permasalahan, metodologi)
- Fokus **poin SDGs**





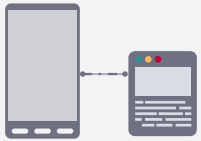
01

Introduction to Kotlin

Kotlin tidak diciptakan untuk ‘mematikan’ Java

Kotlin vs Java

	Kotlin	Java
Code Verbosity	Lebih ringkas dan sederhana	Banyak boilerplate
Data Class	Tidak membutuhkan method *setter* dan *getter*	Membutuhkan *method* setter dan *getter*
Null Safety	Memiliki penanganan kondisi ketika *null*	Tidak memiliki penanganan kondisi ketika *null*



```kotlin
data class Mahasiswa(val nama: String)
```

Kotlin

No more **boilerplate** in Kotlin

```java
public class Mahasiswa {
    private String nama;

    public Mahasiswa(String nama) {
        this.nama = nama;
    }

    public String getNama() {
        return nama;
    }
}
```

Java

```kotlin
var name: String? = null
println(name?.length) // Safe
```

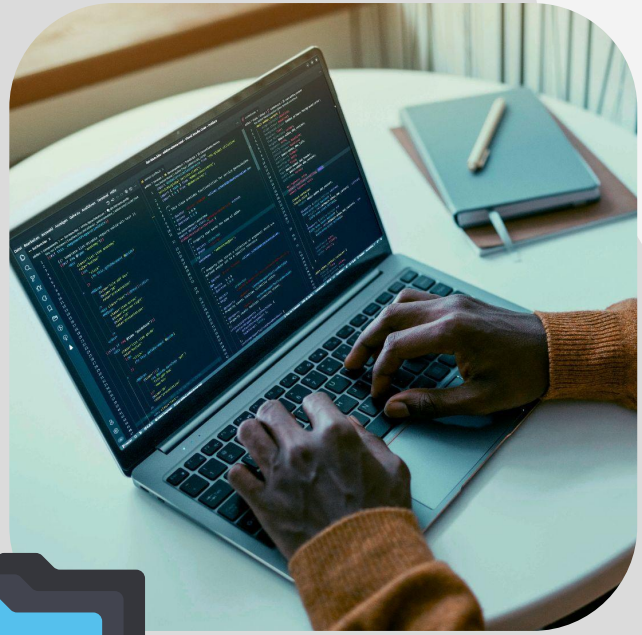
Kotlin

```java
String name = null;
System.out.println(name.length()); // Runtime crash
```

Java

No more null error handling

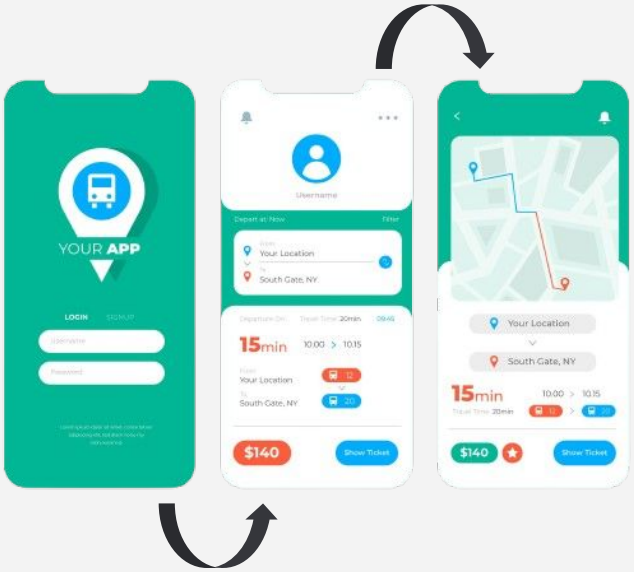
Kotlin tells us how to write code.
Android lifecycle tells us when code runs



02

Struktur Aplikasi dan *Activity* di Android

Fokus Pengembangan Aplikasi Android



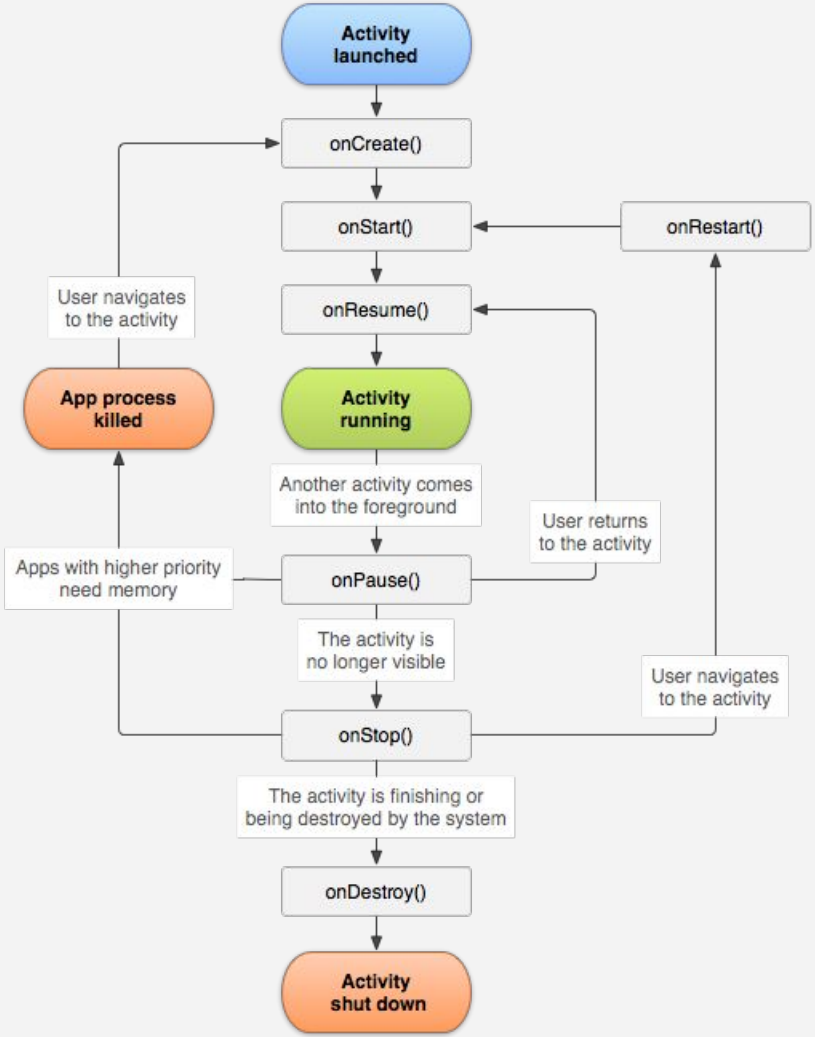
- Membangun rancangan User Interface (UI) di dalam file .xml
- Mengatur logika atau *flow* aplikasi menggunakan kotlin

Activity merepresentasikan **satu tampilan yang memungkinkan pengguna berinteraksi**

Android memiliki konsep yang menangani perubahan konfigurasi dari aktivitas pengguna, yaitu konsep ***Android Activity Lifecycle***

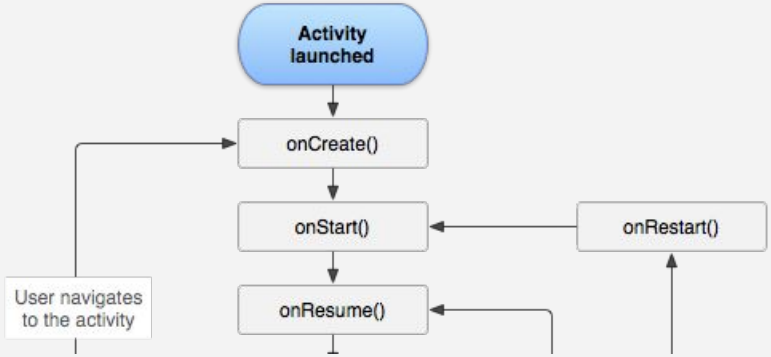
Android memiliki konsep ***Activity Lifecycle*** yang **menangani perubahan kondisi (*state*) suatu *Activity*** mulai dari awal sampai akhir

Android mengontrol statusnya menggunakan *callback method*



❑ onCreate()

- dipanggil ketika *activity* **pertama kali dijalankan**
- **‘Mengikat’** ***front-end*** yang dituliskan di file xml ke file *kotlin*



❑ onStart()

- dipanggil setelah onCreate()
- melakukan *observe* pada perubahan data

❑ onResume()

- Bekerja ketika *user* berinteraksi pada *interface*

❑ onPause()

- dipanggil terdapat *activity* lain yang bekerja dan sifatnya adalah *partial* (ex: membuka dialog)
- menyimpan *temporary state*

❑ onDestroy()

Bekerja ketika

- *user* menekan tombol *back* atau meninggalkan *interface*
- sistem menutup aplikasi

❑ onStop()

- dipanggil setelah *activity* tidak aktif
- Biasanya digunakan ketika menjalankan *task* berat seperti *disconnect* database atau mematikan GPS

Think an Android activity lifecycle as a classroom

- onCreate() → kelas sudah *ready*
- onStart() → mahasiswa masuk kelas
- onResume() → kuliah dimulai
- onPause() → kuliah di pending karena menyanyikan Indonesia Raya
- onStop() → kuliah diakhiri dan mahasiswa keluar ruangan
- onDestroy() → kelas dikunci

PoTD

Practice of The Day

“Track Your Run. Track Your Progress.”

Pak Levi ingin mengembangkan sebuah aplikasi berbasis Android yang memiliki fungsi pencatatan riwayat lari

Pengguna dapat memasukkan data aktivitas lari seperti jarak yang ditempuh, durasi waktu, dan waktu

PoTD Timeline

- Main Screen Setup
- Menambahkan form lari
- Validasi Form
- Data Passing
- Menampilkan riwayat data Part. 1
- Menampilkan riwayat data Part. 2

sekian.

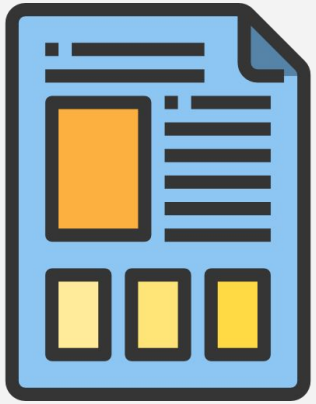
Oleh-Oleh buat di Rumah

What you need to do

Bersama *clan* masing-masing, **susunlah sebuah abstrak** atau gambaran ide yang telah diusulkan ke dalam **80-300 kata**. Abstrak harus memuat:

- Penjelasan secara singkat terkait **latar belakang**
- **Metodologi**
- Gambaran hasil dan kesimpulan yang akan diperoleh dan kesimpulan

Abstrak **ditulis secara langsung di *template*** yang sudah disediakan



Klik *icon* di atas untuk melihat *template*