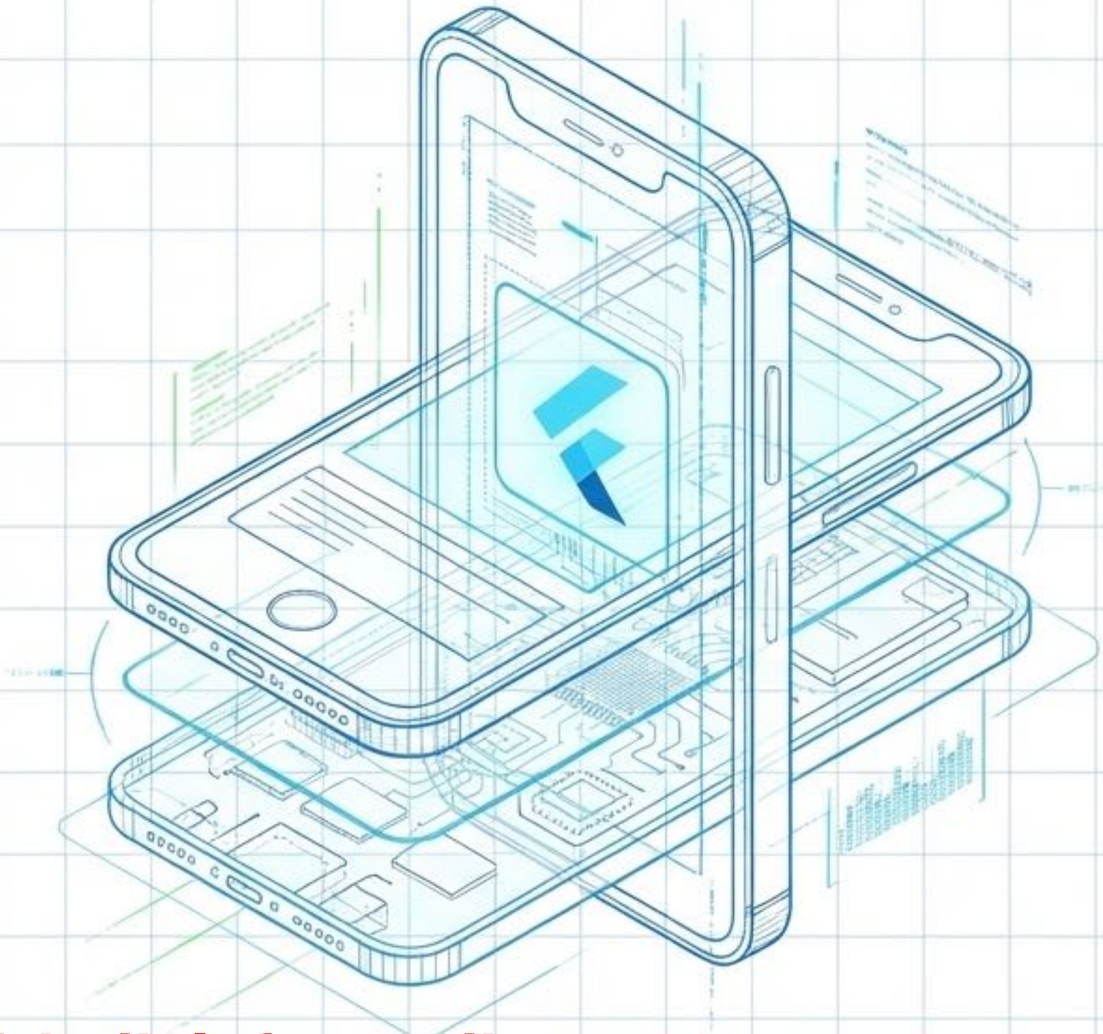
Pemrograman Mobile: Pertemuan 2

Anatomi Organisme Digital — Android Activity, Event Handling, Intent & Flutter Lifecycle

Dosen: Rizki Muliono, S.Kom, M.Kom



Teknik Informatika
Universitas Medan Area

Tujuan Instruksional Khusus (TIK)

1. Memahami State Life Cycle pada aplikasi mobile (fase hidup sebuah activity).

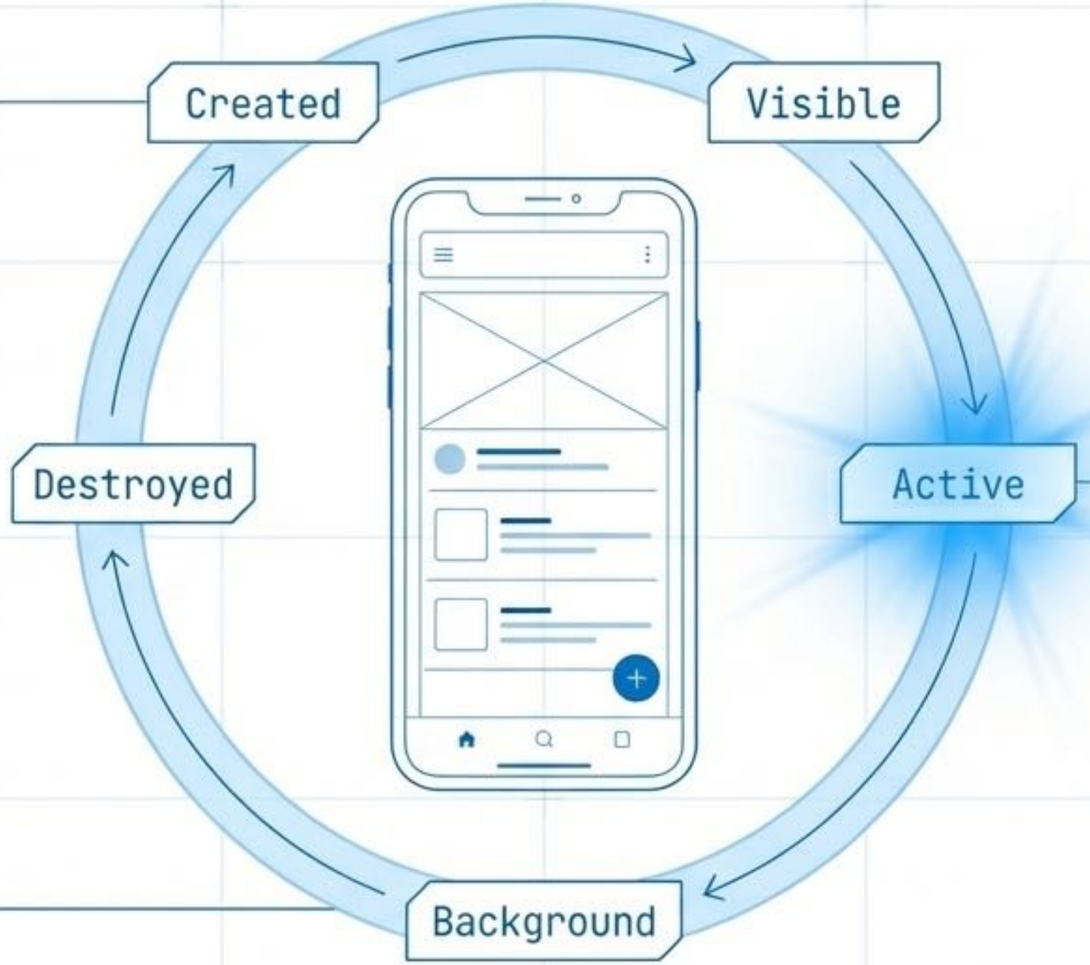
2. Membuat instruksi pemrograman berbasis Event yang dieksekusi pada keadaan tertentu.

3. Menghubungkan antar-activity dan melakukan Passing Data (Intent) secara mulus.

Activity Life Cycle: Jantung Antarmuka Aplikasi

Apa itu Activity?

Komponen utama yang menampung User Interface (UI) dan memberikan ruang interaksi antara user dan aplikasi.



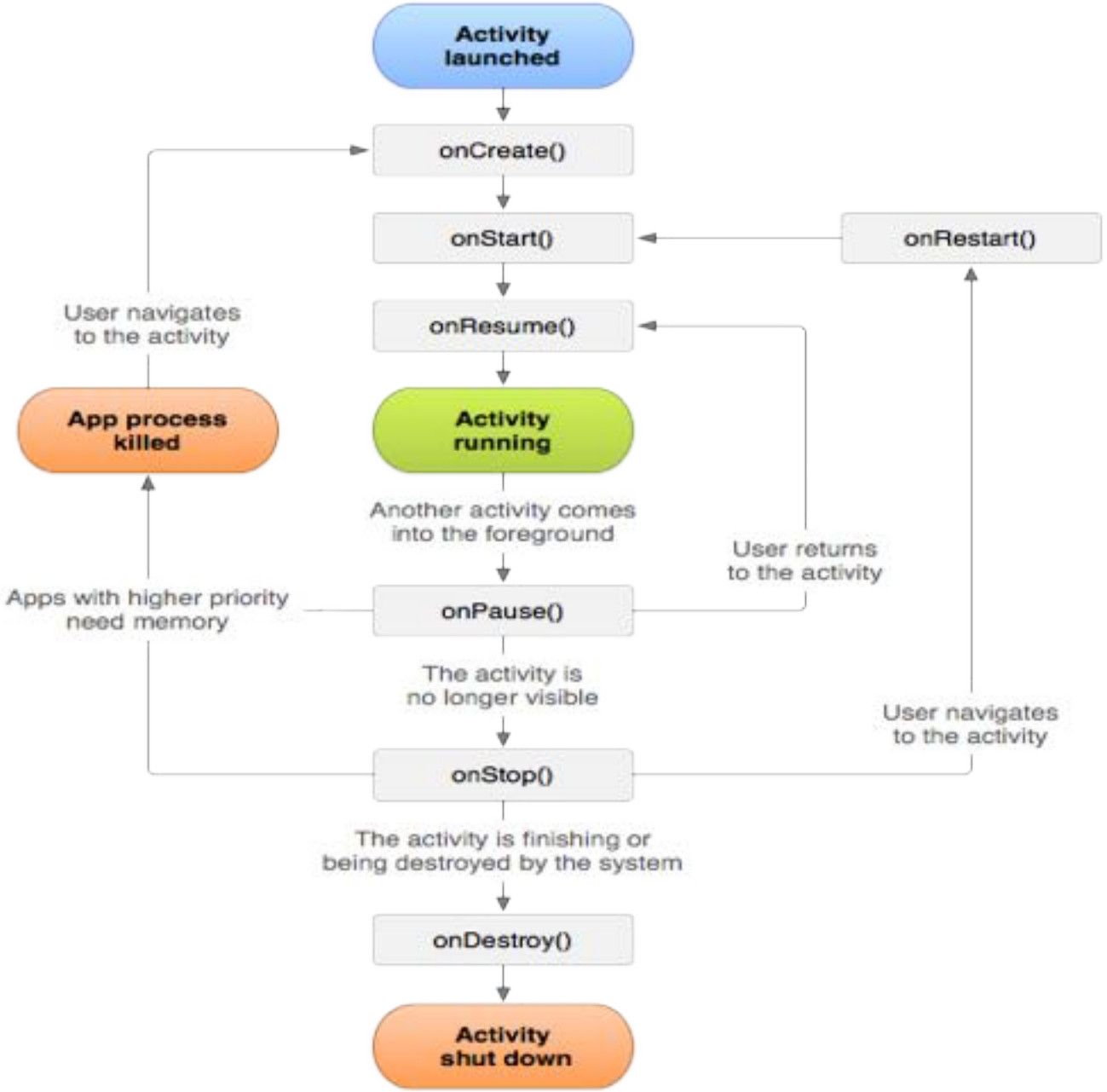
Sistem Callback

Metode yang dipanggil otomatis oleh sistem ketika terjadi perubahan status (event), ibarat detak jantung aplikasi.

Transisi Primer:

onCreate(), onStart(), onResume(), onPause(), onStop(), onDestroy()

Android Activity Lifecycles



Mobile Development Essentials

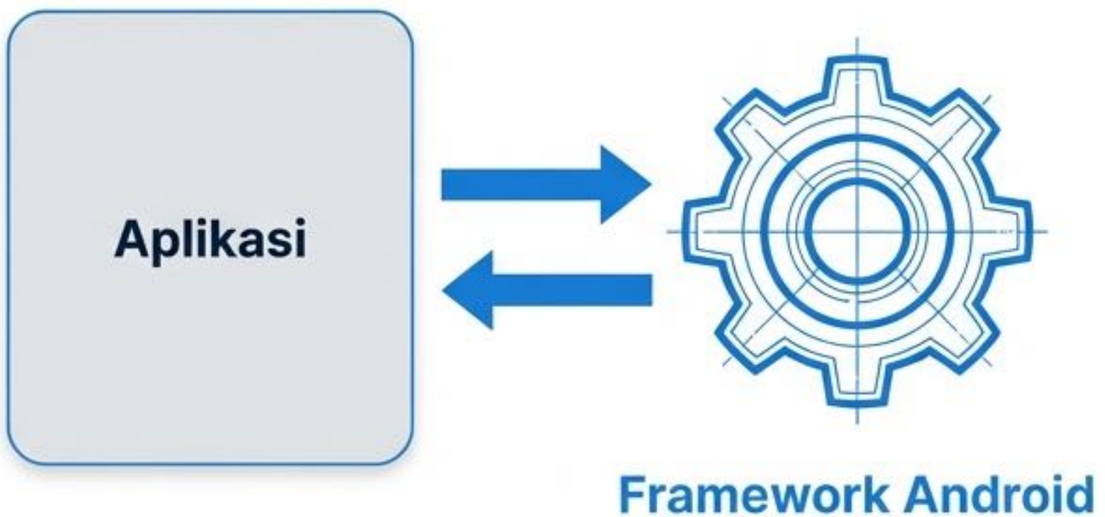
Mastering Android & Flutter Activity Lifecycles

Panduan Visual Arsitektur Daur Hidup Aplikasi

Panduan Komprehensif

Mengapa Aplikasi Membutuhkan Daur Hidup?

Daur hidup Android Activity adalah sistem callback yang dirancang oleh framework Android untuk mengelola transisi status sebuah activity dari saat diciptakan hingga dihancurkan.

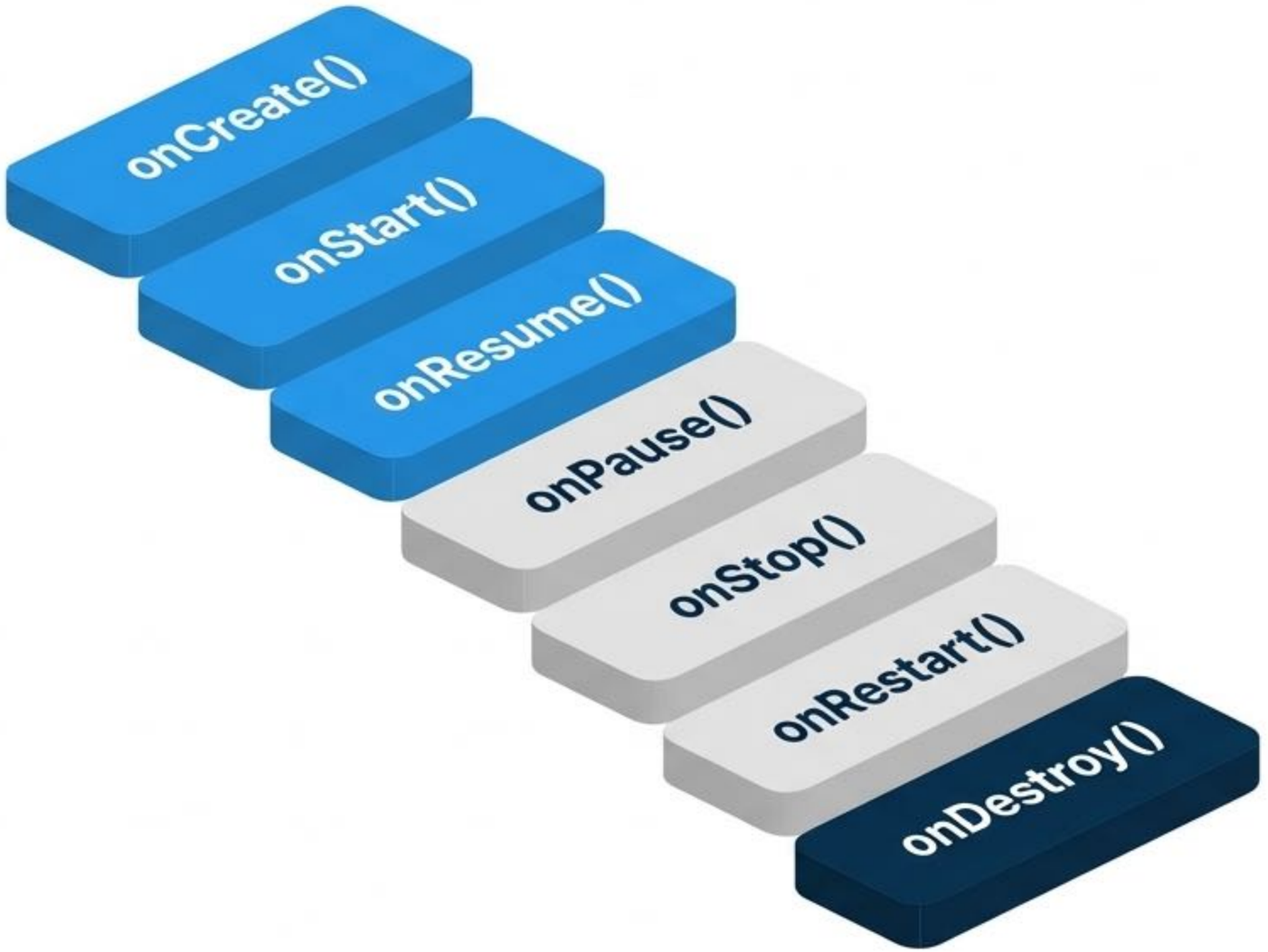


Manajemen Memori:
Mencegah kebocoran (memory leaks) dan mengoptimalkan RAM.

Pengalaman Pengguna:
Memastikan transisi mulus saat pengguna berpindah antar aplikasi.

Efisiensi Baterai:
Menghentikan proses berat yang tidak lagi terlihat di layar.

Anatomi Android Activity: 7 Pilar Callback



Fase 1: Penciptaan & Visibilitas Awal

onCreate()

Dipanggil pertama kali saat activity dibuat oleh sistem.

- **Tugas utama**: Menginisialisasi status awal dan mengeksekusi setContentView() untuk menggambar antarmuka pengguna (UI) secara visual.

onStart()

Dipanggil tepat setelah onCreate().

- **Status**: Activity mulai terlihat (visible) secara fisik oleh pengguna di layar, namun belum memiliki fokus untuk berinteraksi penuh.

Fase 2: Interaksi Pengguna (The Prime Time)

Dipanggil saat activity sepenuhnya siap berinteraksi dengan pengguna pengguna. Aplikasi kini berada di garis depan (foreground) dan terkunci dalam status Running.

Konteks: Semua input sentuhan, gestur, dan ketikan keyboard diproses secara eksklusif pada fase puncak ini.

Fase 3: Kehilangan Fokus & Latar Belakang

onPause()

Aplikasi kehilangan fokus utama tetapi masih terlihat sebagian (misal: tertimpa alert dialog transparan).

Aksi Wajib: Waktu yang tepat untuk menghentikan animasi atau menahan sumber daya yang menguras baterai.

onStop()

Aplikasi sudah tidak terlihat sama sekali (pengguna menekan tombol home atau membuka layar baru). UI sepenuhnya disembunyikan.

onRestart()

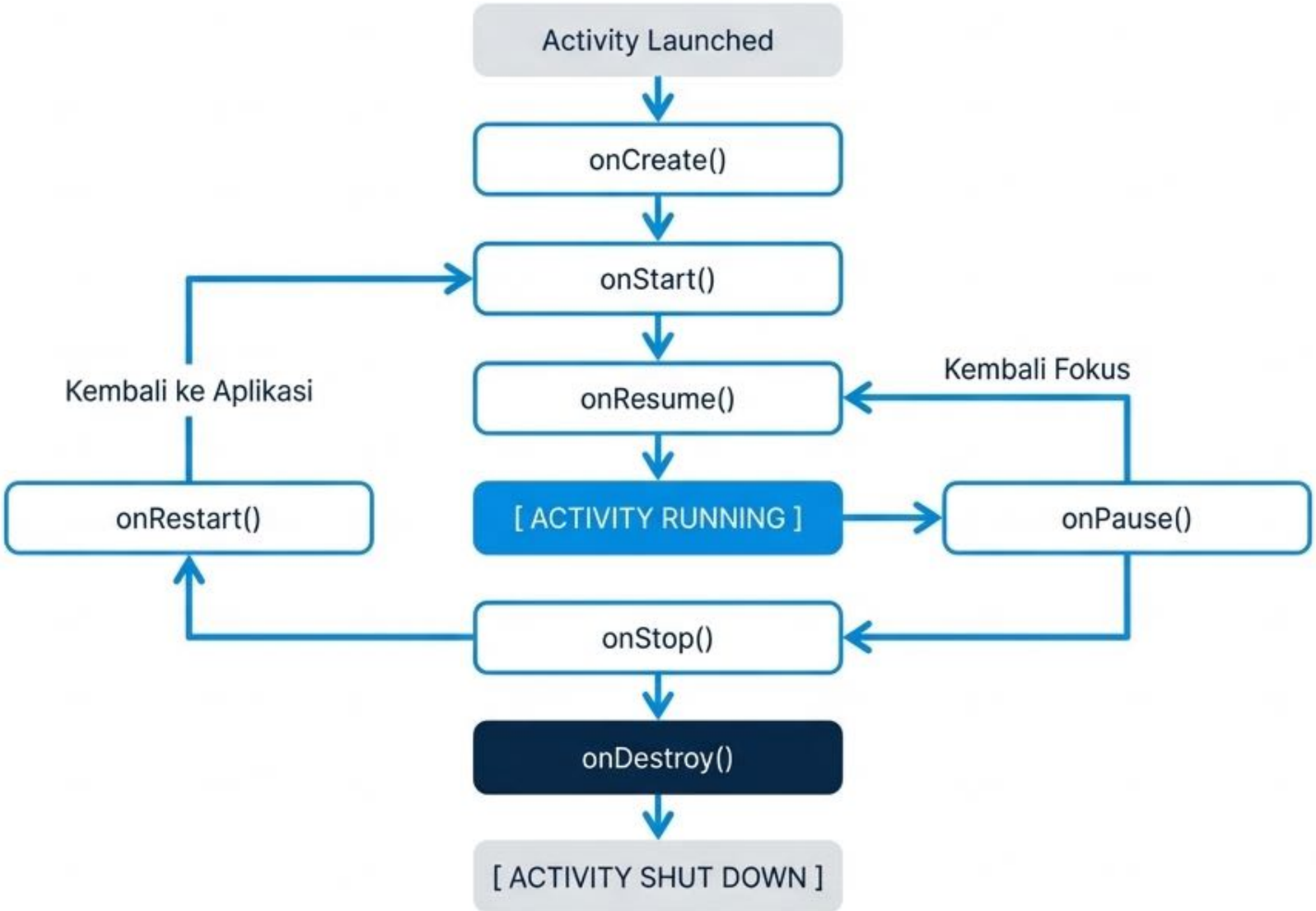
Jembatan penghubung kembali. Dipanggil jika pengguna tiba-tiba memutuskan untuk kembali ke activity yang berstatus stopped.

Fase 4: Pelepasan Sumber Daya Sistem

onDestroy()

- Ini adalah callback terakhir yang dieksekusi sebelum activity dihancurkan sepenuhnya dari memori sistem.
- **Pemicu:** Terjadi karena pengguna secara eksplisit menutup aplikasi, atau sistem operasi secara paksa membersihkan memori (kill process).
- **Fungsi Kritis:** Membersihkan semua resource, thread, atau koneksi database yang tersisa untuk mencegah kebocoran memori fatal.

Peta Arsitektur: Alur Transisi Status Activity



Skenario Nyata 1: Interupsi Telepon Masuk

Aksi 1: Aplikasi Dibuka Pertama Kali

Trace: onCreate() –> onStart() –> onResume()

Aksi 2: Menerima Panggilan Telepon (Aplikasi tergeser ke Background)

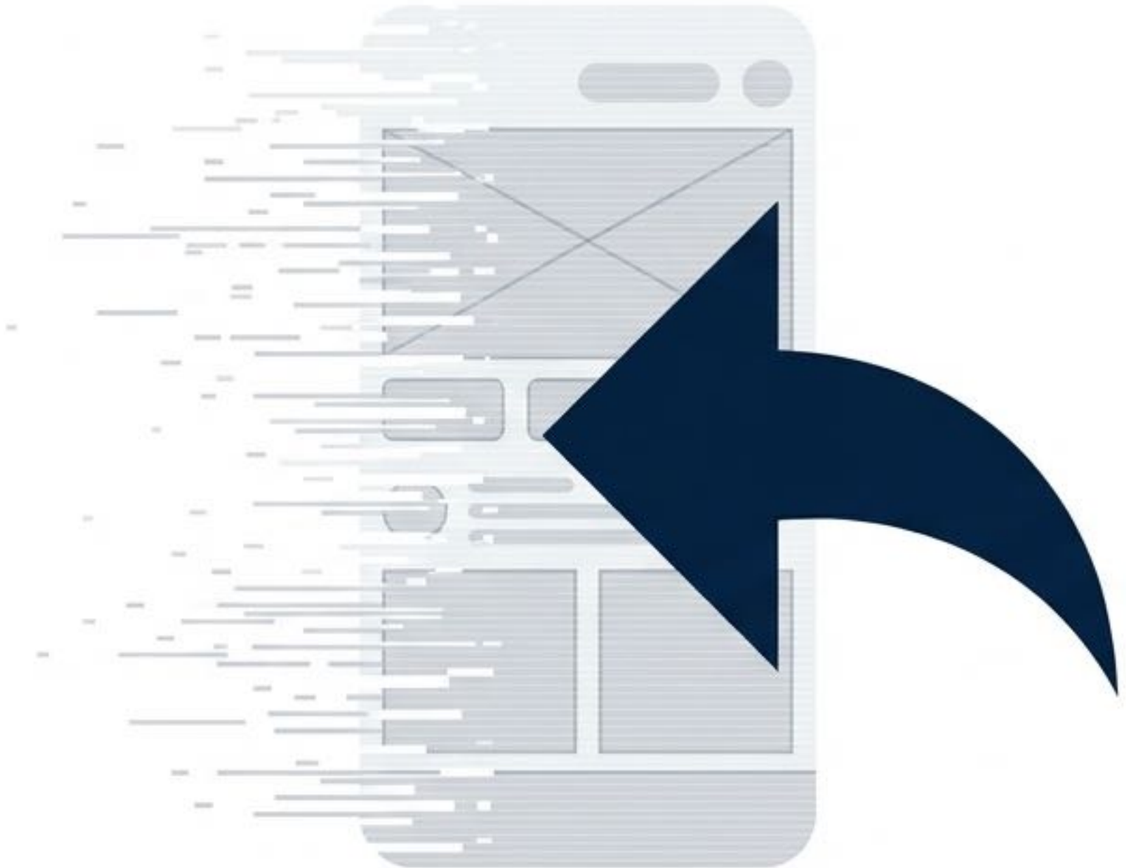
Trace: onPause() (Aplikasi utama kehilangan fokus) -> onStop() (Aplikasi utama tertutup total oleh layar panggilan).

Aksi 3: Panggilan Selesai (Pengguna kembali ke Aplikasi)

Trace: onRestart() -> onStart() -> onResume() (Aplikasi kembali aktif).

Skenario Nyata 2: Menyelesaikan Sesi Penggunaan

Konteks: Pengguna selesai beraktivitas dan menekan tombol Back atau melakukan gestur Exit.



Alur Kausalitas Sistem:

1. **onPause()** : Interaksi dengan layar dihentikan seketika.
2. **onStop()** : Layar aplikasi menghilang dari pandangan pengguna, digantikan oleh Home Screen.
3. **onDestroy()** : Sesi secara resmi diakhiri, memori sistem dibebaskan, dan arsitektur activity dihancurkan secara permanen.

Evolusi Paradigma: Memasuki Dunia Flutter

Dalam ekosistem Flutter, antarmuka pengguna tidak lagi dikelola oleh entitas Activity, melainkan sepenuhnya diatur melalui StatefulWIdget.

Insight Utama:
Meskipun mekanisme dan bahasanya ta berbeda dari Android Native, konsep fundamental mengenai manajemen sumber daya dan logika transisi status tetap identik.

4 Pilar Utama Lifecycle Flutter (StatefulWidget)

initState()

Fase inisialisasi awal (Setara dengan onCreate). Dipanggil hanya satu kali saat widget dimasukkan ke dalam widget tree.

didChangeDependencies()

Dipanggil segera setelah initState(). Digunakan khusus saat widget perlu mengambil data dari dependensi luar (seperti InheritedWidget).

build()

Membangun struktur UI secara visual. Ini adalah method yang paling sering dieksekusi ulang setiap kali status (state) berubah.

dispose()

Fase eliminasi (Setara dengan onDestroy). Membersihkan controller, animasi, dan memori sesaat sebelum widget dihapus permanen.

Implementasi Kode: Anatomi StatefulWIdget

```dart
class MyWidgetState extends State<MyWidget> {
  @override
  void initState() {
    super.initState();
  }

  @override
  void didChangeDependencies() {
    super.didChangeDependencies();
  }

  @override
  Widget build(BuildContext context) {
    return Container();
  }

  @override
  void dispose() {
    super.dispose();
  }
```

Inisialisasi state awal (Satu kali eksekusi)

Menangkap perubahan dependensi

Menggambar ulang UI jika ada perubahan state

Pembersihan memori & listener (Mencegah leak)

Konvergensi Konsep: Matriks Android vs Flutter

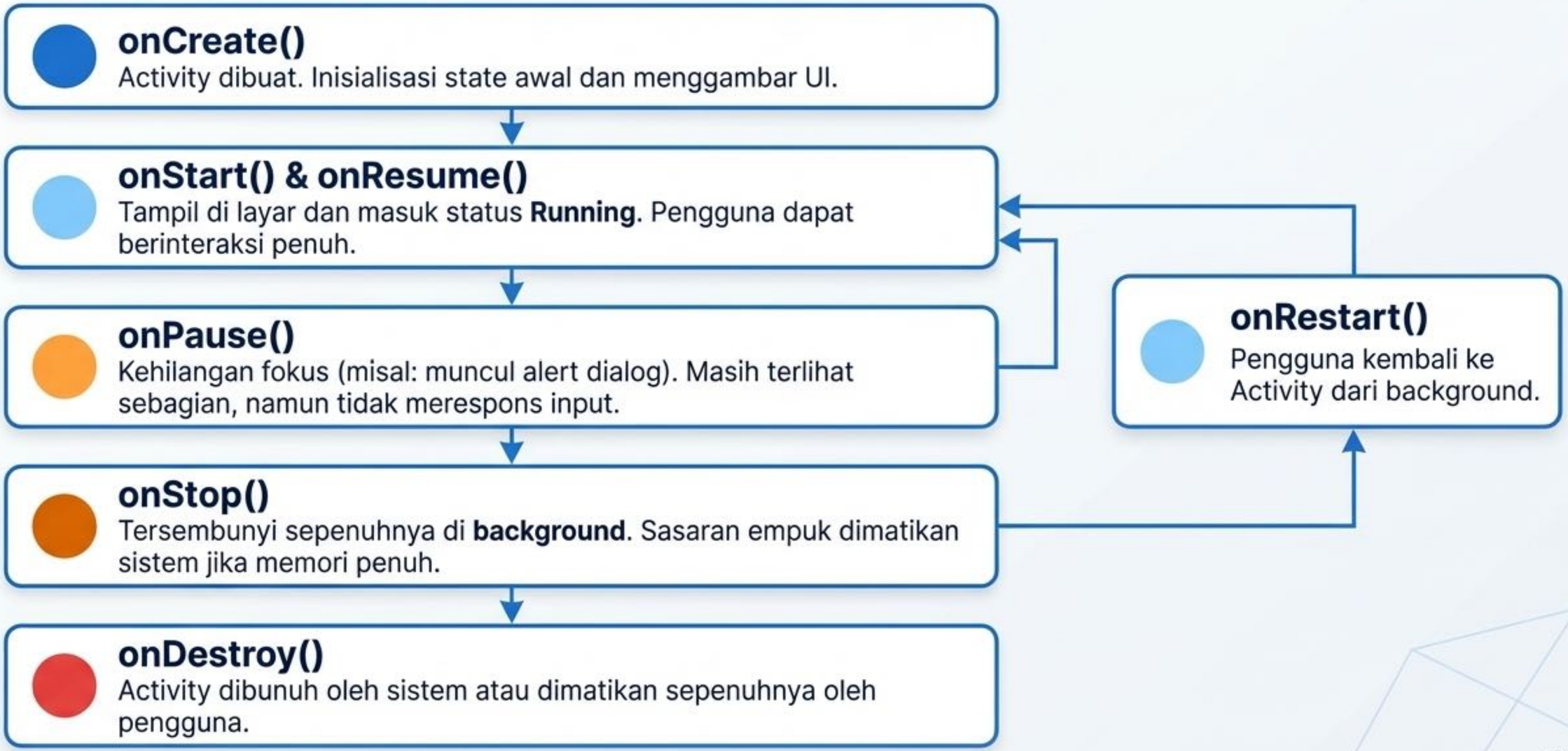
Tujuan Sistem	Android Native	Flutter
Inisialisasi Awal State	onCreate()	initState()
Menggambar Layar / UI	setContentView()	build()
Pelepasan Memori (Akhir)	onDestroy()	dispose()

Kesimpulan: Sintaks dan bahasa pemrograman mungkin berevolusi, tetapi esensi optimalisasi sumber daya perangkat seluler selalu bertumpu pada arsitektur yang sama.

Kuasai Daur Hidupnya, Kuasai Aplikasinya.

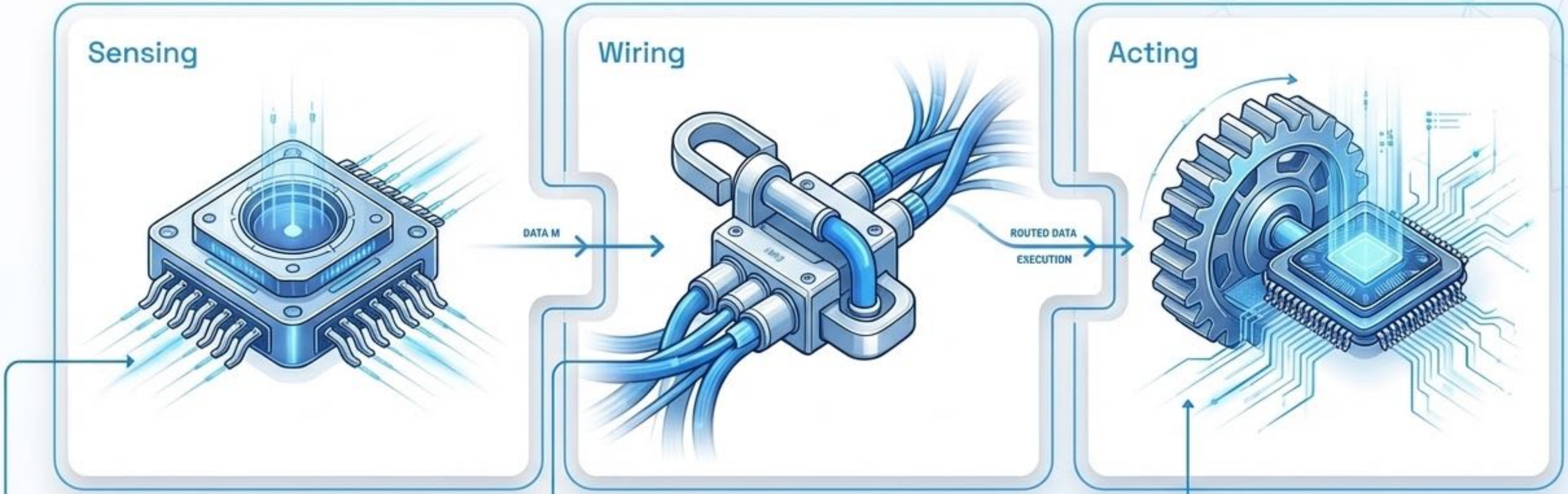
- **Antisipasi State**: Pahami siklus ini untuk mencegah aplikasi crash dan mengeliminasi memory leak.
- **Maksimalkan Efisiensi**: Pastikan proses berat hanya berjalan saat aplikasi benar-benar sedang aktif di layar pengguna.
- **Agnostik Platform**: Penguasaan arsitektur Native Android adalah fondasi paling sempurna untuk menguasai framework modern seperti Flutter.

Anatomi Android Activity Lifecycle: Dari Lahir Hingga Mati



Event Handi Handling: Sistem Saraf Aplikasi

Event adalah cara aplikasi mengumpulkan data dari interaksi pengguna menggunakan antrian FIFO (First In First Out).



1. Event Listener: Objek/Antarmuka yang bertugas 'mendengarkan' dan menerima pemberitahuan saat peristiwa terjadi.

2. Event Registration: Proses mendaftarkan handler ke listener (menyambungkan kabel), sehingga sistem tahu siapa yang harus dipanggil.

3. Event Handler: Metode aktual berisi logika yang dieksekusi sebagai respons terhadap peristiwa tersebut.

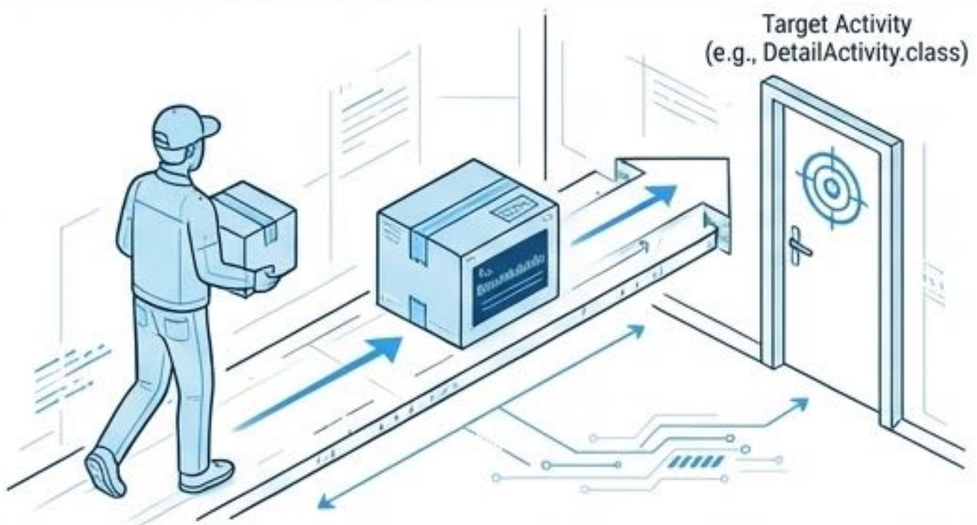
Matriks Respons Interaksi (Common Event Handlers)

Trigger / Interaksi	Event Listener	Event Handler
Klik Tunggal / Sentuh	OnClickListener()	onClick()
Tahan Layar (1 detik+)	OnLongClickListener()	onLongClick()
Pindah Fokus UI	OnFocusChangeListener()	onFocusChange()
Tekan Tombol Hardware	OnKeyListener()	onKey()
Gesture / Gerakan Layar	OnTouchListener()	onTouch()
Pilih Item Menu	OnMenuItemClickListener()	onMenuItemClick()

Intent: Sistem Sirkulasi & Pesan Asinkron

Intent adalah pesan asinkron untuk 'meminta' fungsi dari komponen lain, dikirim via metode **startActivity()**.

Explicit Intent



Memanggil Activity lain secara langsung di dalam aplikasi yang sama dengan menyebutkan nama class-nya secara spesifik.

Implicit Intent

Mendelegasikan tugas ke sistem Android untuk mencari aplikasi internal/eksternal yang cocok untuk menangani aksi tersebut (misal: membuka browser, menelepon).

Eksekusi Intent & Konfigurasi Flutter

Flutter Dart Code

```
### Explicit Intent (Navigator)
Navigator.push(context,
  MaterialPageRoute(builder: (context) =>
      SecondScreen()));

### Implicit Intent (url_launcher)

final Uri url =
  Uri.parse('https://flutter.dev');
if (!await launchUrl(url))
  throw 'Could not launch $url';
```

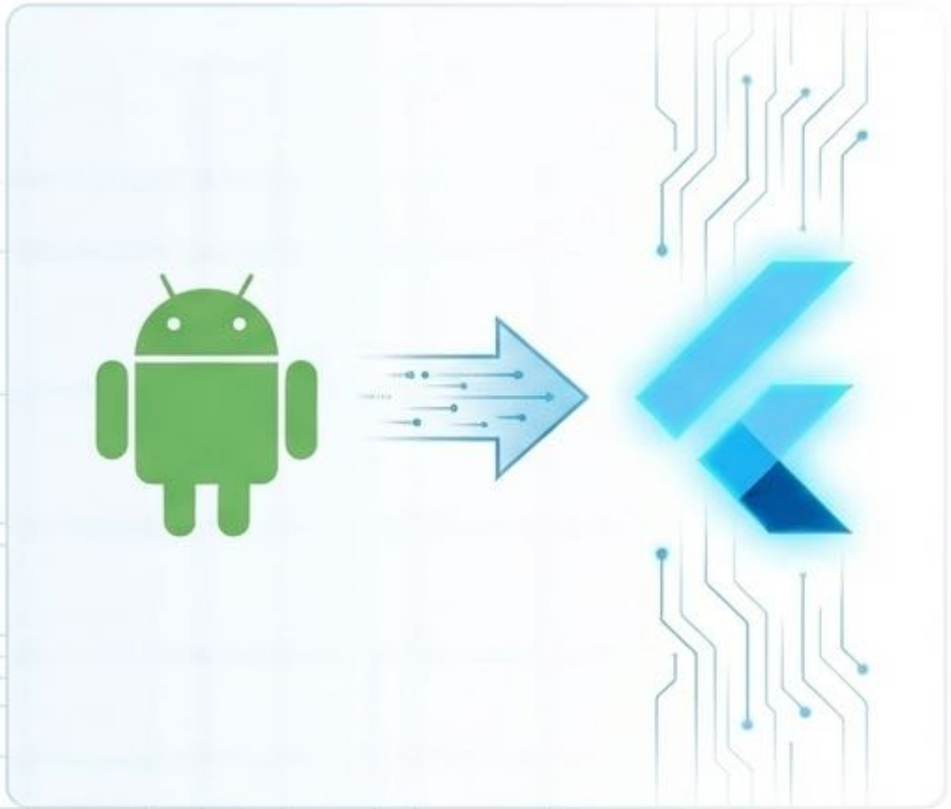
Flutter Android Manifest (for Intents)

```
<intent-filter>
   <action android:name="android.intent.action.VIEW" />
   <category
android:name="android.intent.category.DEFAULT" />
   <category
android:name="android.intent.category.BROWSABLE" />
   <data android:scheme="https"
   android:host="flutter.dev" />
</intent-filter>
```

Menjelaskan bagaimana aplikasi Flutter mendeklarasikan kemampuan untuk menangani URL di Android.

Evolusi ke Flutter: AppLifecycleState

Aplikasi Flutter berjalan pada engine terisolasi. State siklus hidupnya mengamati status aplikasi secara global dari perspektif sistem operasi.



Detached

Aplikasi ditampung di engine, namun terlepas dari host views.

Inactive

Aplikasi dalam keadaan tidak aktif, tidak menerima input pengguna (fase transisi).

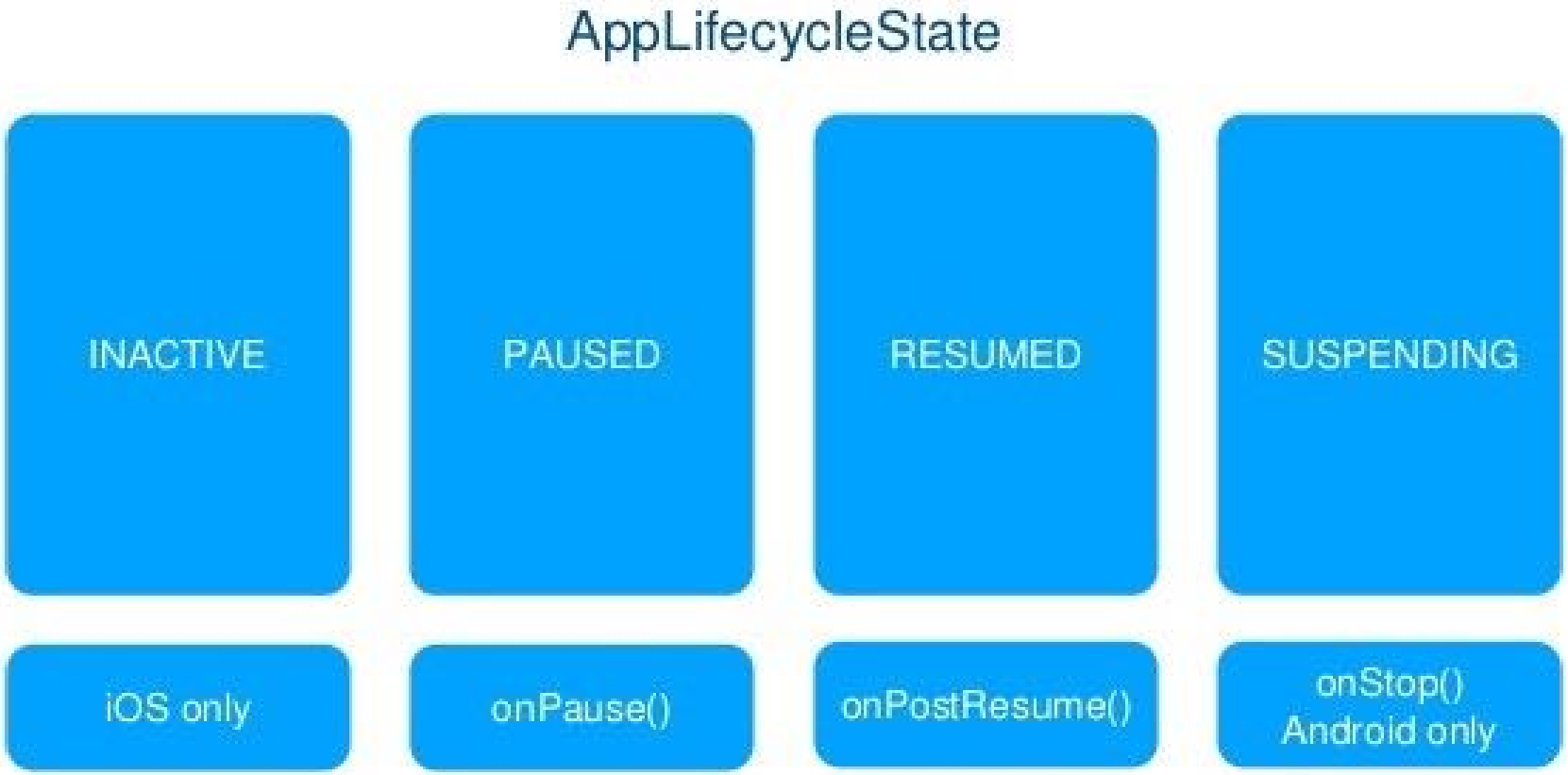
Paused

Berjalan di background, tidak terlihat, tidak merespons. (Setara dengan onPause ( ) di Android).

Resumed

Terlihat penuh di layar dan siap merespons input user. (Setara dengan onResune ( ) di Android).

Life cycle


AppLifecycleState
INACTIVE
PAUSED
RESUMED
SUSPENDING
iOS only
onPause()
onPostResume()
onStop()
Android only

Siklus Hidup StatlvfulWidget

1. initState()

Initiate the apostives dangan from didChendencit cenai

2. didChangeDependencies()

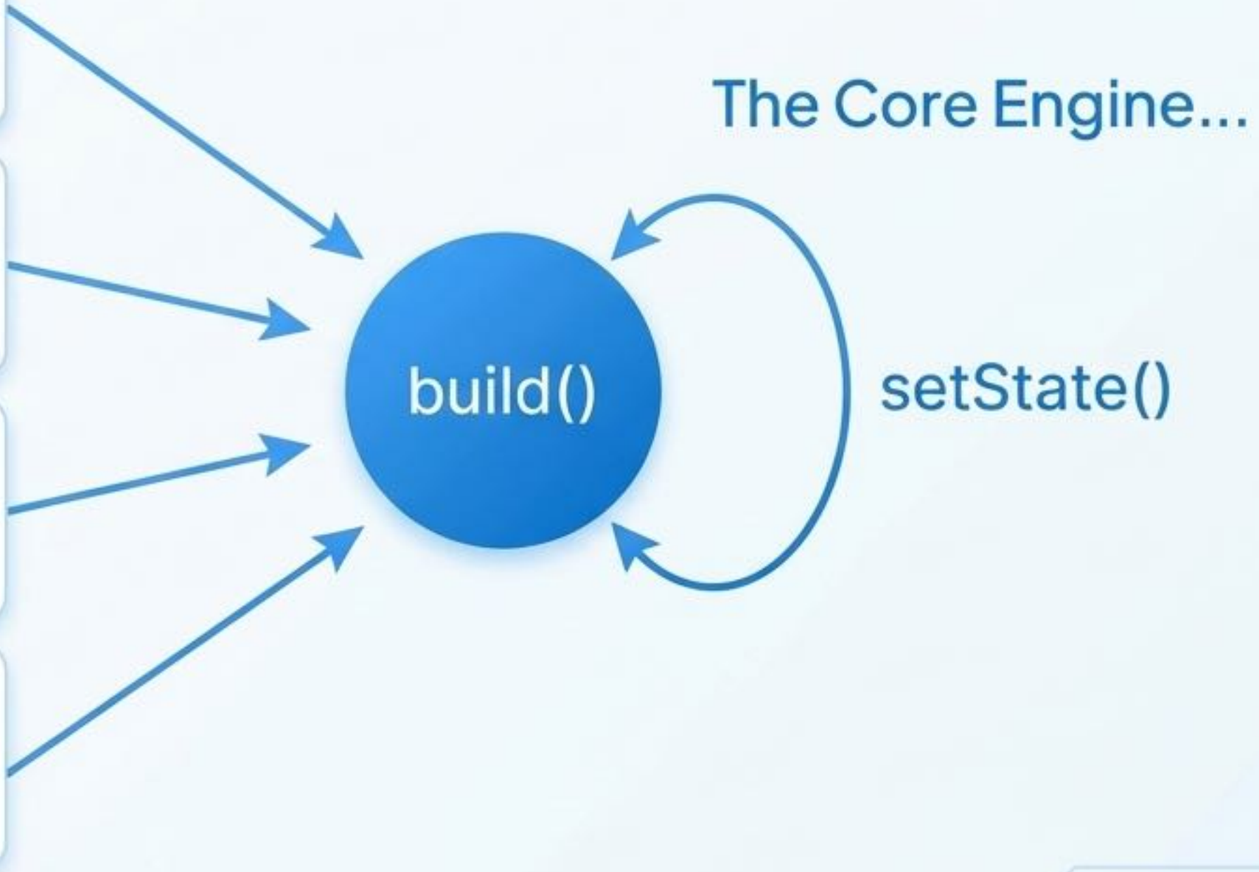
Confirm ine ranon stottant

3. didUpdateWidget()

SetState/:nednted didUpdateWidget

4. deactivate() & dispose()

Deactivate & dimose oreress & dispose

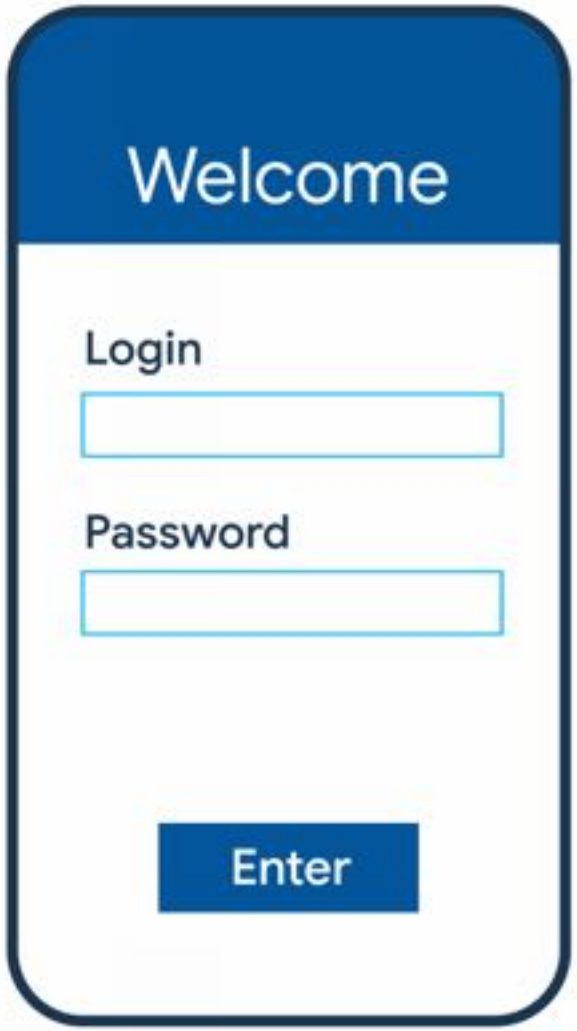


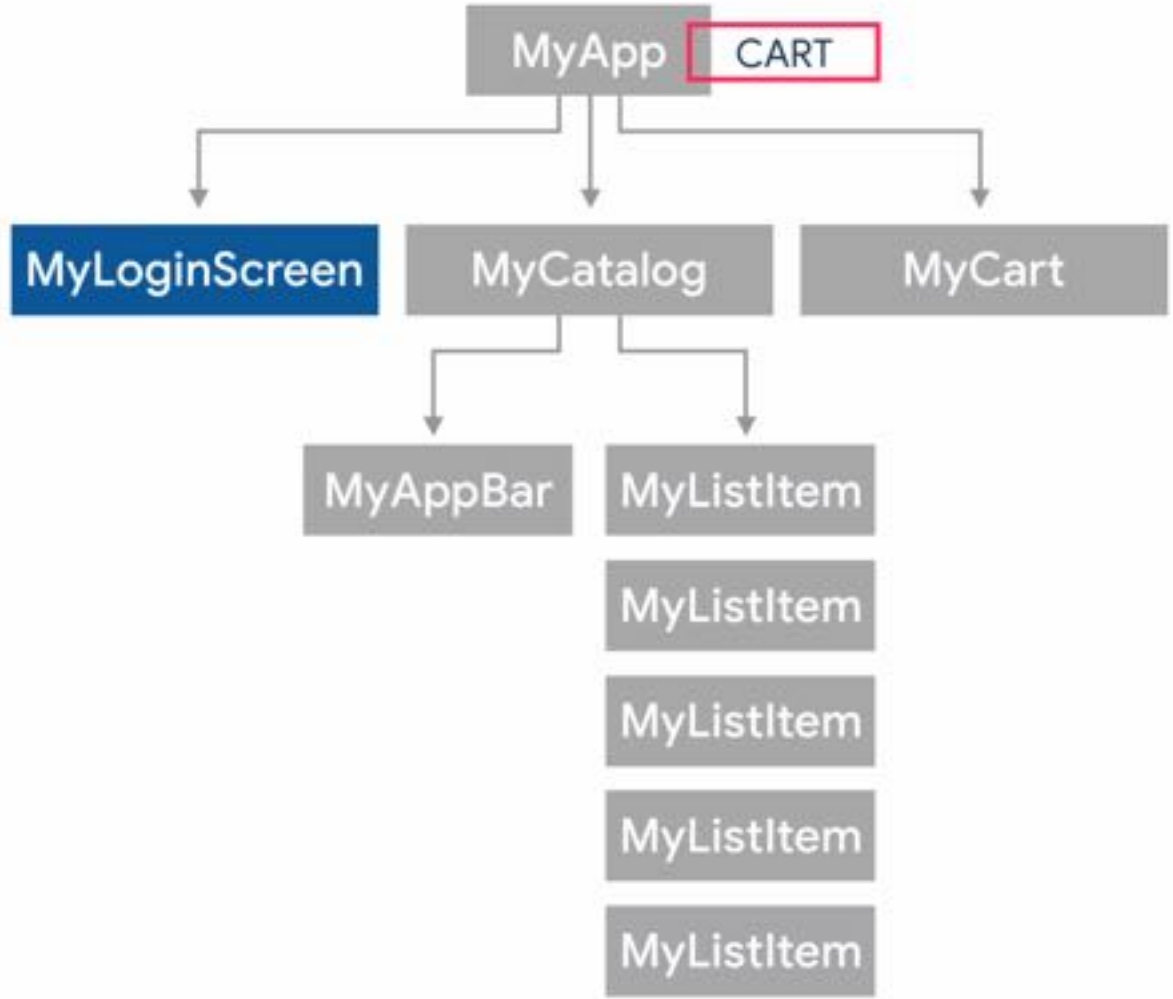

UI = f(state)
The layout on the screen
Your build methods
The application state

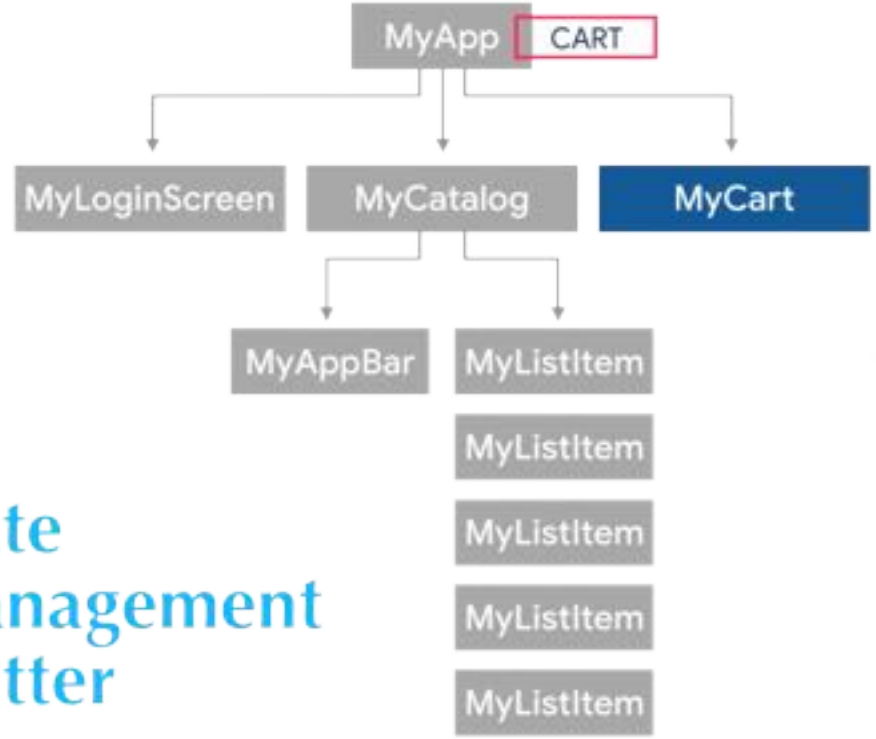
Arsitektur UI: Stateless vs Stateful

UI = f(state)

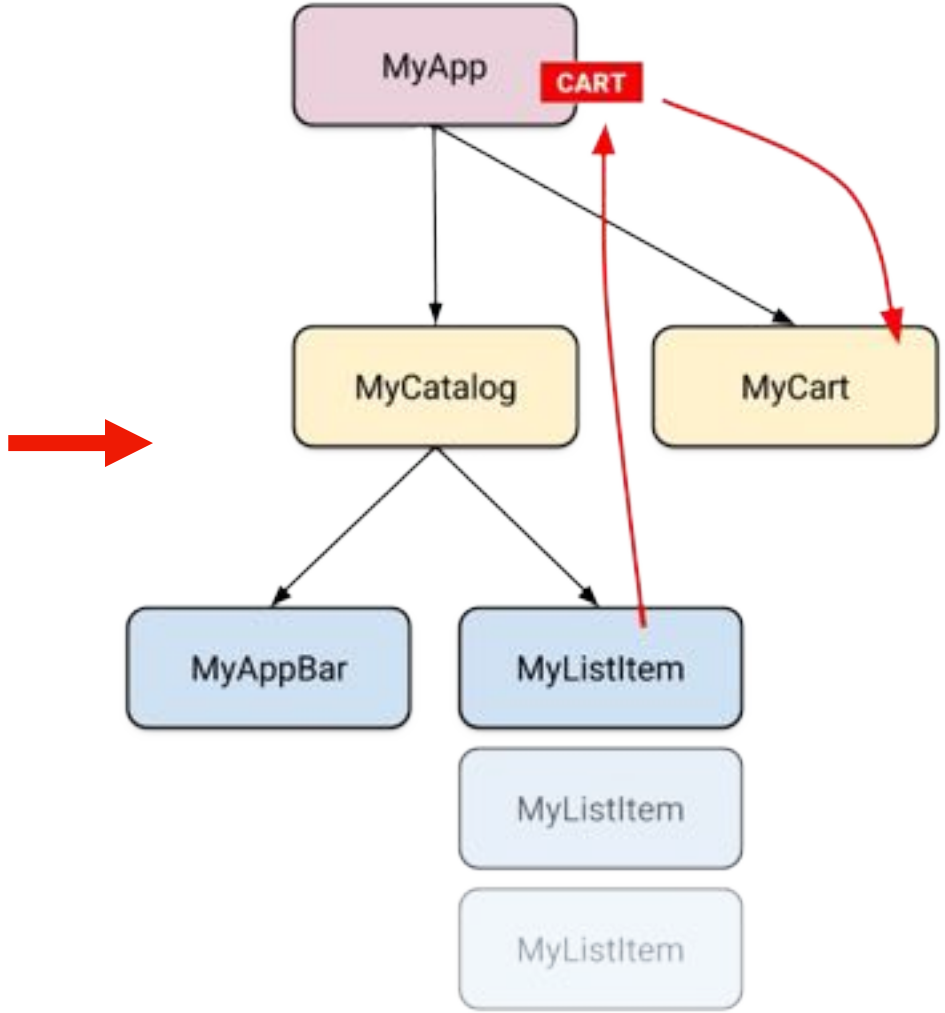
Stateless Widget	Stateful Widget
Tidak memiliki state yang dapat berubah (Statis).	Memiliki state yang dinamis dan reaktif.
Direnderer ulang dari awal secara keseluruhan jika induk berubah.	Hanya merender bagian (subset) UI yang berubah.
Sederhana, memori ringan, dan eksekusi sangat cepat.	Lebih kompleks, membutuhkan manajemen memori dan state.
Label Teks, Ikon statis, Gambar dekoratif.	Formulir Login, Counter Angka, Animasi Interaktif.


Welcome
Login
Password
Enter


MyApp
CART
MyLoginScreen
MyCatalog
MyCart
MyAppBar
MyListItem
MyListItem
MyListItem
MyListItem
MyListItem



State Management Flutter



Ada Pertanyaan, Tinggal kan di Comentar Group WA

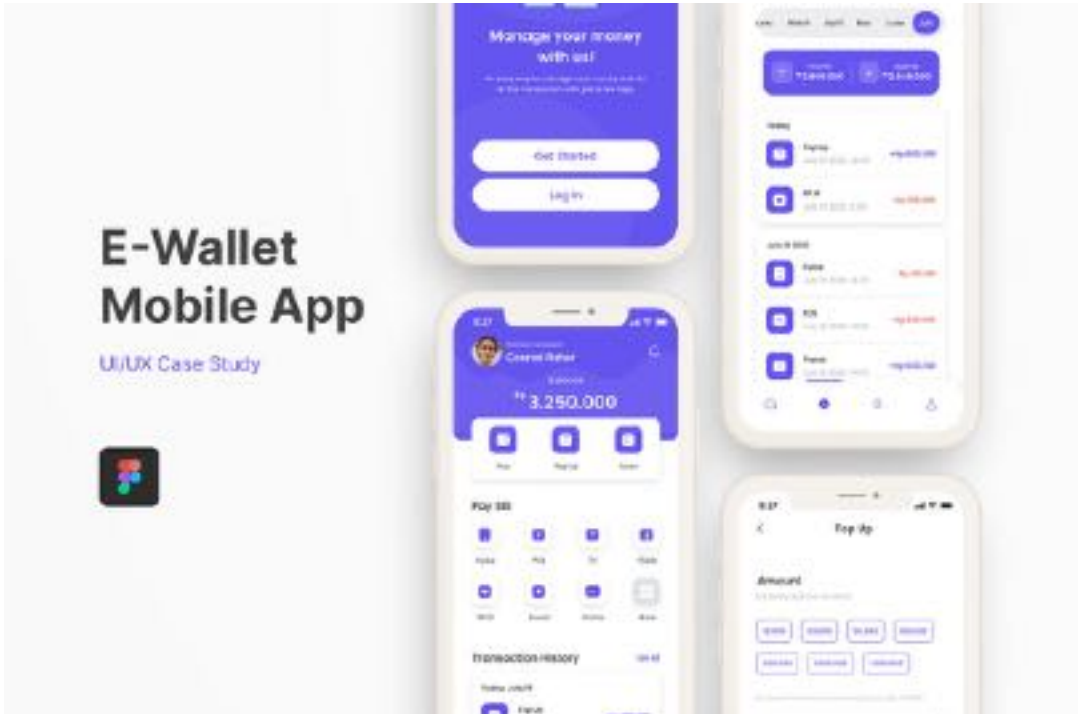
?

TUGAS #1 App Mobile – IDE APLIKSI MOBILE ANDROID

- Buat lah desain tampilan aplikasi mobile (mockup), berikut dengan keterangan fungsi tombol atau menu di masing2 page activity dan deskirpsi aplikasi.
- Minimal 5 page, tiap page deskripsikan future dan fungsinya tiap element tombol
- Dikerjakan Masing2 **Kelompok (5 orang)**, karya akan menjadi project akhir mahasiswa.
- Batas Pengumpulan **30-03-2026 23.50 wib. (elearning)**
- Contoh :

TUGAS #1 App Mobile – IDE APLIKSI MOBILE ANDROID

- **Ketentuan**:
- Tema Aplikasi Tentukan Sendiri dan **TIDAK ADA YANG BOLEH SAMA ANTAR KELOMPOK**
- Aplikasi Memiliki Sistem Saldo / E-Wallet
- Aplikasi memiliki fungsi Menampilkan dan melakukan Transaksi dengan QR CODE
- Tools desain page bisa menggunakan : **FIGMA**, Sketch, Whimsical atau Miro



Tugas Proyek Akhir : KEBANGKAN IDE DIGITAL ANDA

Tujuan Utama

Rancang Mockup UI Aplikasi Mobile beserta deskripsi fungsional fitur dan tombol. Mockup ini akan menjadi dasar coding Proyek Akhir.

Cakupan & Aturan Emas

- Minimal 5 Halaman Activity yang saling terhubung.
- Dikerjakan berkelompok.
- Tema aplikasi ditentukan sendiri. **TIDAK BOLEH SAMA antar kelompok.**

Tenggat Waktu: 30 Maret 2026, pkl 23.50 WIB (Pengumpulan via E E-Learning).

Tools Direkomendasikan:

Figma Sketch miro Whimsical

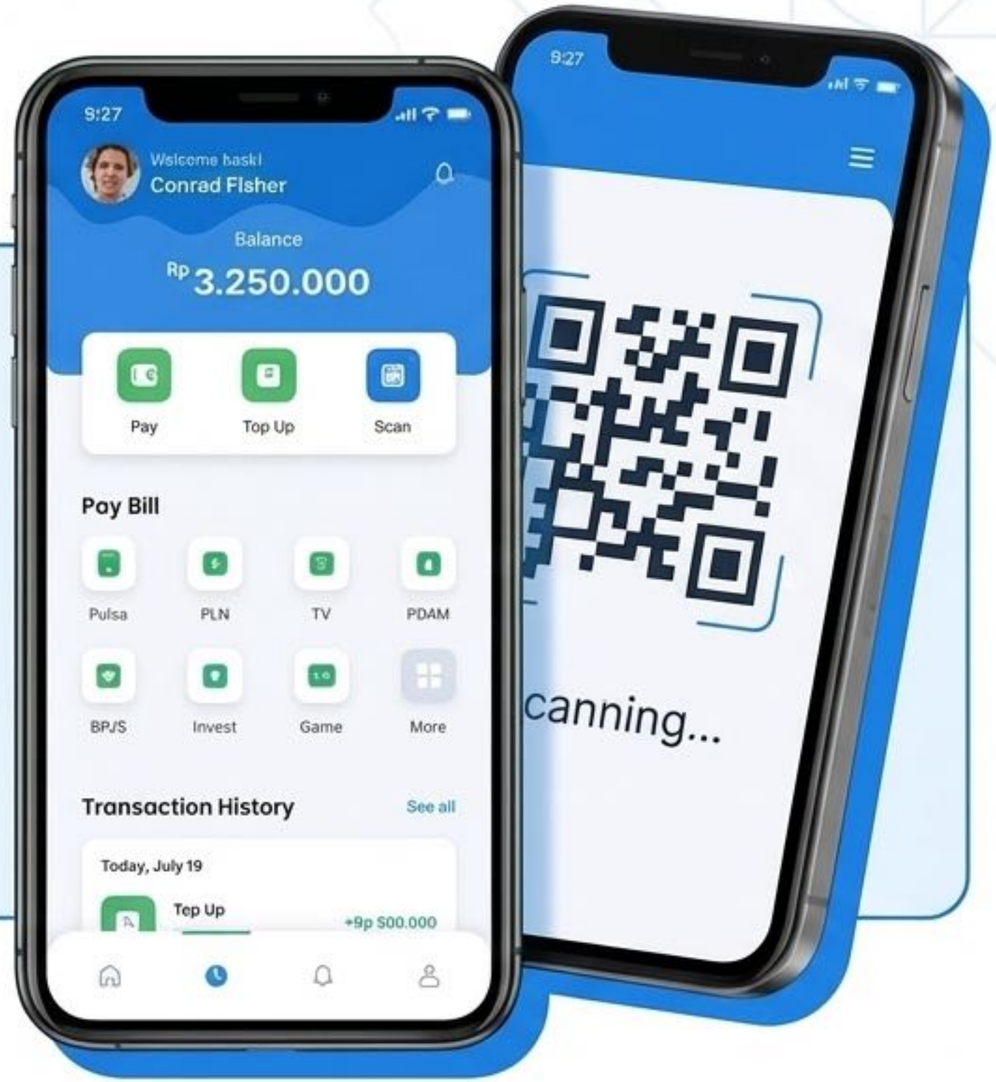
Studi Kasus Wajib: E-Wallet & QR Code

Syarat Kelulusan Mockup Aplikasi:

- ☑ Aplikasi harus memiliki Sistem Saldo / E-Wallet yang terstruktur.
- ☑ Aplikasi harus memiliki fitur untuk menampilkan dan melakukan transaksi dengan QR CODE.

Call to Action

Diskusi ide, pertanyaan, dan pembentukan tim dilanjutkan di Grup WhatsApp.



Terima Kasih!

Mari mulai mendesain masa depan.


NOTIFICATION
Leveled up!
Terima Kasih!