Mobile Application

Pertemuan 13

Dart Language

- Bahasa Dart, yang dikembangkan oleh Google, adalah bahasa pemrograman yang dapat digunakan untuk mengembangkan aplikasi web, desktop, sisi server, dan seluler.
- Dart adalah bahasa pemrograman yang digunakan untuk membuat kode aplikasi Flutter, memungkinkannya memberikan pengalaman terbaik kepada pengembang untuk pembuatan aplikasi seluler tingkat tinggi.

Fitur Dart

- Alat produktif: Ini termasuk alat untuk menganalisis kode, plugin lingkungan pengembangan terintegrasi (IDE), dan ekosistem paket besar.
- Pengumpulan sampah: Ini mengelola atau menangani dealokasi memori (terutama memori yang ditempati oleh objek yang tidak lagi digunakan).
- Ketik anotasi (opsional): Ini untuk mereka yang menginginkan keamanan dan konsistensi untuk mengontrol semua data dalam aplikasi.

Lanjutan

- Diketik secara statis: Dart aman untuk tipe dan menggunakan inferensi tipe untuk menganalisis tipe saat runtime. Fitur ini penting untuk menemukan bug selama waktu kompilasi.
- Portabilitas: Ini tidak hanya untuk web (diterjemahkan ke JavaScript), tetapi dapat dikompilasi secara asli ke kode ARM dan x86.

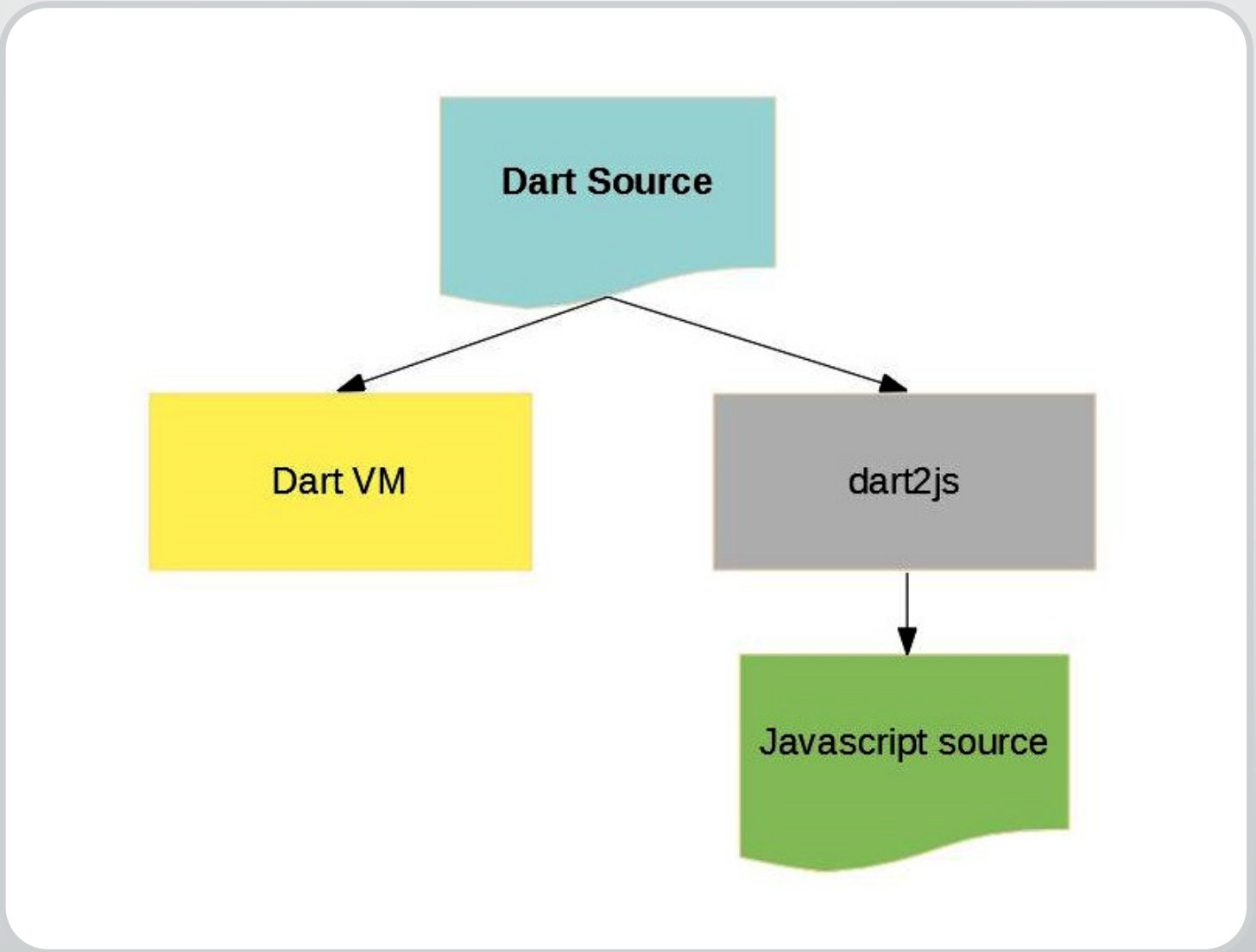
Cara Kerja Dart

- Untuk memahami dari mana fleksibilitas bahasa itu berasal, kita perlu tahu bagaimana caranya menjalankan kode Dart.
- Ini dilakukan dengan dua cara:
  - Mesin Virtual Dart (VM)
  - kompilasi JavaScript

Dart

Diagram

- Eksekusi kode Dart beroperasi dalam dua mode
- kompilasi Just-In-Time (JIT) atau
- kompilasi Ahead-Of-Time (AOT).



Fitur : Just In Time (JIT)

- Kompilasi JIT adalah tempat kode sumber dimuat dan dikompilasi ke kode mesin asli oleh Dart VM dengan cepat.
- Ini digunakan untuk menjalankan kode di antarmuka baris perintah atau saat mengembangkan aplikasi seluler untuk menggunakan fitur seperti debugging dan hot reload

Ahead-of-Time

- Kompilasi AOT adalah tempat Dart VM dan kode dikompilasi sebelumnya dan VM bekerja lebih seperti sistem runtime Dart, menyediakan pengumpul sampah dan berbagai metode asli dari kit pengembangan perangkat lunak Dart (SDK) ke aplikasi.

Flutter

Mengapa Flutter menggunakan Dart

- Kerangka kerja Flutter bertujuan untuk menjadi pengubah permainan dalam pengembangan aplikasi seluler, menyediakan semua alat yang dibutuhkan oleh pengembang untuk membuat aplikasi yang luar biasa tanpa kekurangan dalam kinerja dan skalabilitas.

Masalah SDK Lain yang Flutter Akhiri

- **Siklus pengembangan yang panjang/lebih mahal**
  - Untuk dapat mengatasi permintaan pasar, Anda harus memilih untuk membangun untuk satu platform, atau membuat beberapa tim. Ini memiliki beberapa konsekuensi dalam hal biaya, beberapa tenggat waktu, dan kemampuan kerangka kerja asli yang berbeda.

Lanjutan

- **Berbagai bahasa untuk dipelajari**
  - Jika seorang pengembang ingin mengembangkan untuk beberapa platform, mereka harus belajar bagaimana melakukan sesuatu dalam satu OS dan bahasa pemrograman, dan kemudian, hal yang sama pada OS dan bahasa pemrograman lain.

Lanjutan

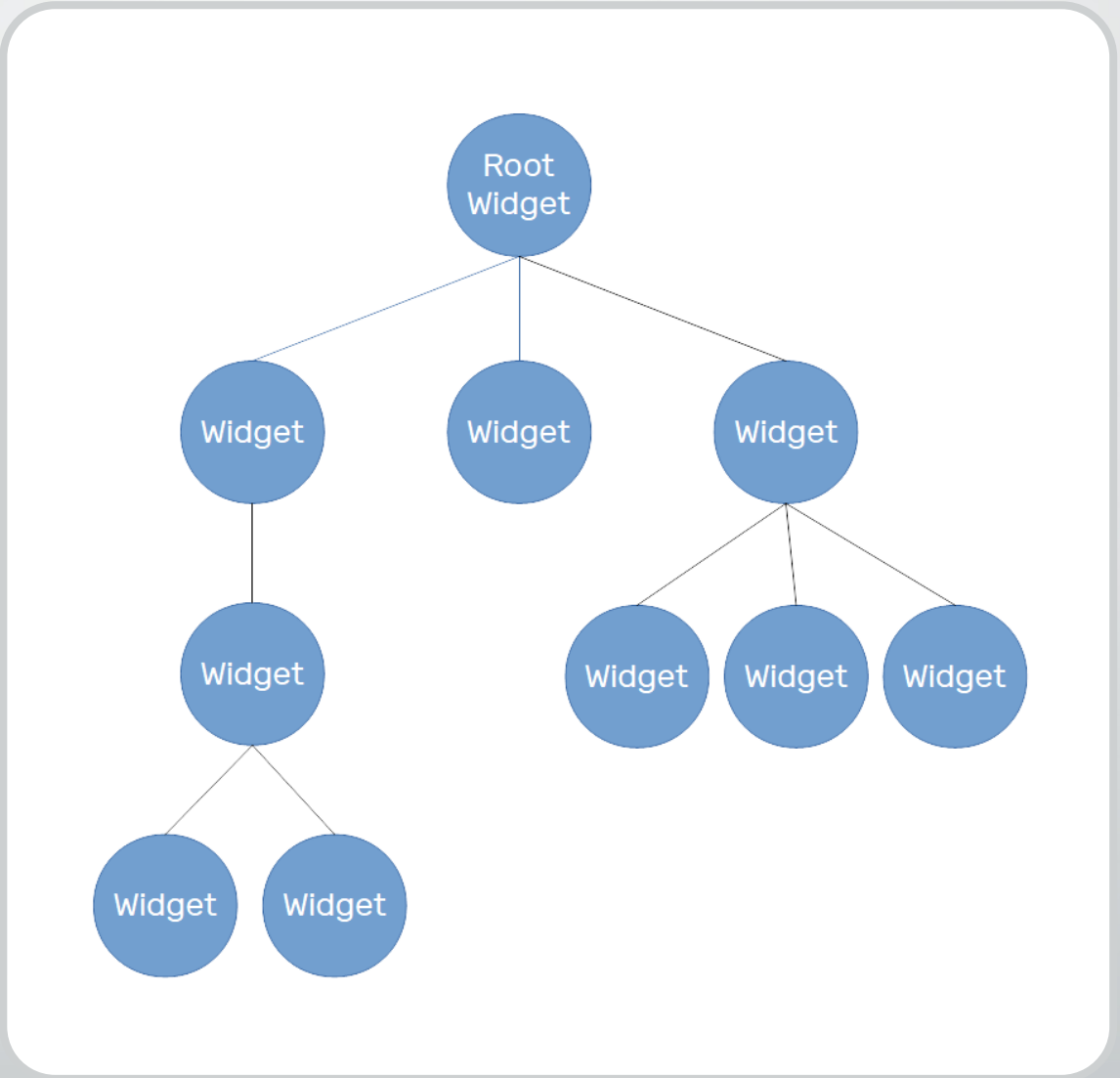
- **Waktu pembuatan/kompilasi yang lama**
  - Beberapa pengembang mungkin telah mengalami bagaimana waktu pembuatan dapat berdampak pada produktivitas.
  - Di Android, misalnya, beberapa pengembang mengalami beberapa kali waktu pembuatan yang lama setelah beberapa menit pengkodean

Lanjutan

- **Efek samping solusi lintas platform yang ada**
  - Pengembang mengadopsi kerangka kerja lintas platform yang ada dalam upaya untuk mengatasi masalah sebelumnya, tetapi ini dapat memengaruhi kinerja, desain, atau pengalaman pengguna.

Konsep Pohon Widget

- Ini adalah konsep penting lainnya dalam tata letak Flutter. Di situlah widget menjadi hidup. Pohon widget adalah representasi logis dari semua widget UI. Itu dihitung selama tata letak



Lanjutan

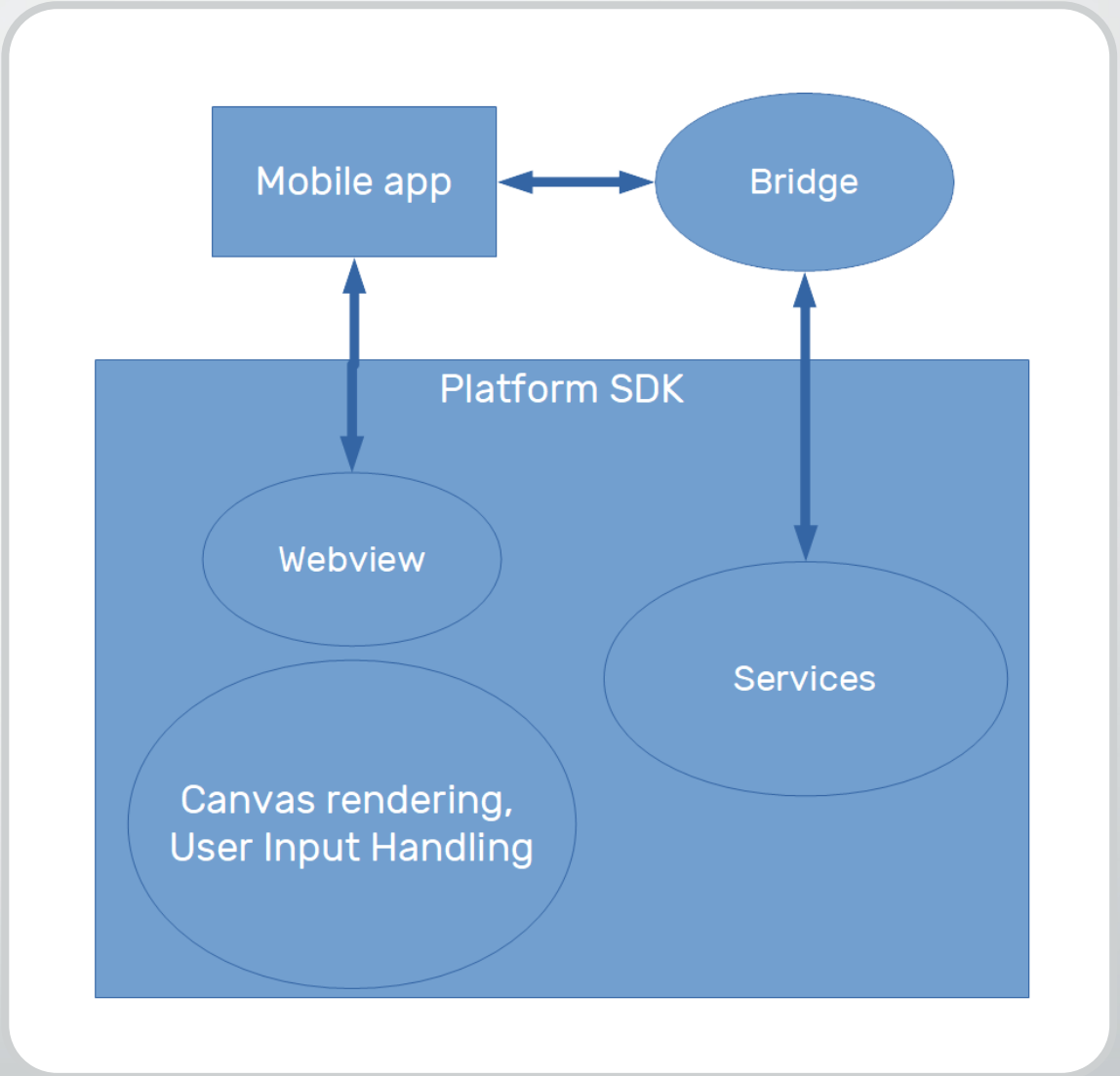
- Widget direpresentasikan di pohon sebagai node. Ini mungkin memiliki status yang terkait dengannya; setiap perubahan pada statusnya menghasilkan pembangunan kembali widget dan anak yang terlibat.

Rendering

- Salah satu aspek utama yang membuat Flutter unik adalah caranya menggambar komponen visual ke layar. Perbedaan besarnya adalah bagaimana aplikasi berkomunikasi dengan SDK platform, apa yang diminta SDK untuk dilakukan, dan apa yang dilakukannya dengan sendirinya:

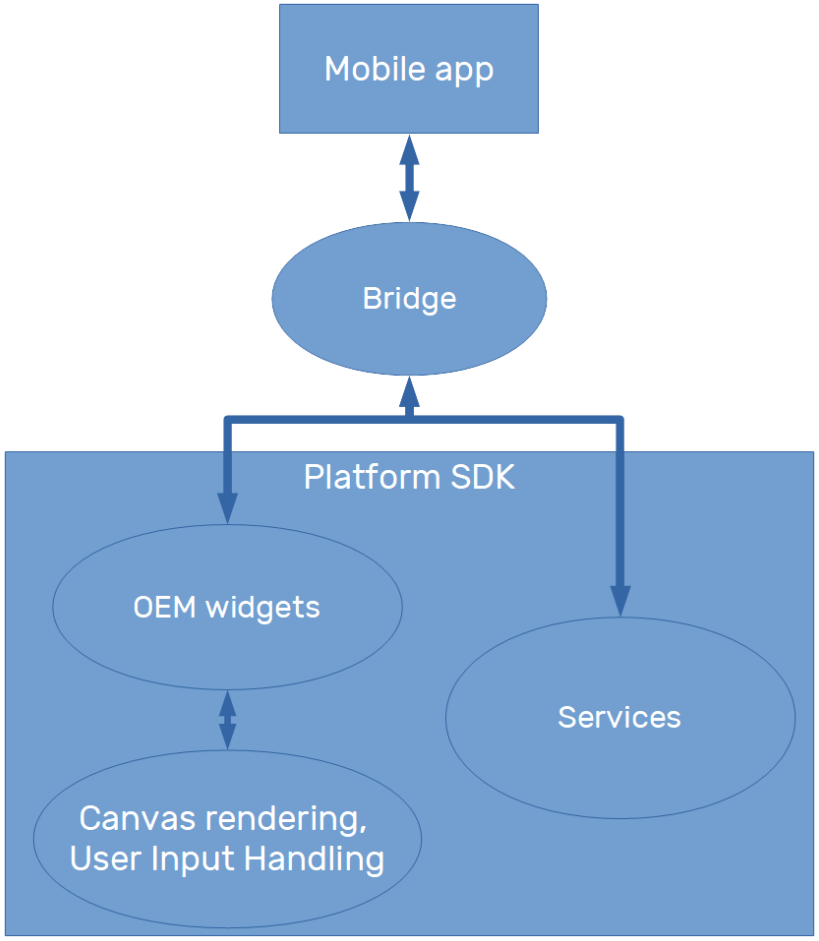
Web-based Rendering

- Aplikasi tidak mengetahui bagaimana rendering dilakukan oleh platform; satu-satunya hal yang dibutuhkan adalah widget webview yang akan membuat kode HTML dan CSS.



Framework and OEM widgets

- Dalam mode rendering ini, pekerjaan dilakukan oleh SDK seperti aplikasi asli biasa, tetapi sebelum itu, tata letak ditentukan oleh langkah tambahan dalam bahasa kerangka kerja.



Rendering Flutter Sendiri

- Flutter memilih untuk melakukan semua kerja keras sendirian. Satu-satunya hal yang dibutuhkan dari SDK platform adalah akses ke API Layanan dan kanvas untuk menggambar UI di: