Nama : Muhammad Naufal Kurniawan
Kelas : TI-3C
NIM : 2241720214

Bahasa Dart adalah inti dari framework Flutter. Memahami Dart adalah dasar untuk bekerja dengan Flutter; pengembang perlu mengetahui asal-usul bahasa Dart, bagaimana komunitas mengerjakannya, kelebihannya.

Fitur-fitur Bahasa pemrograman Dart:

- **Productive tooling**: merupakan fitur kakas (*tool*) untuk menganalisis kode, plugin IDE, dan ekosistem paket yang besar.
- **Garbage collection**: untuk mengelola atau menangani dealokasi memori (terutama memori yang ditempati oleh objek yang tidak lagi digunakan).
- **Type annotations (opsional)**: untuk keamanan dan konsistensi dalam mengontrol semua data dalam aplikasi.
- **Statically typed**: Meskipun *type annotations* bersifat opsional, Dart tetap aman karena menggunakan fitur *type-safe* dan *type inference* untuk menganalisis *types* saat *runtime*. Fitur ini penting untuk menemukan *bug* selama kompilasi kode.
- **Portability**: bahasa Dart tidak hanya untuk web (yang dapat diterjemahkan ke JavaScript) tetapi juga dapat dikompilasi secara *native* ke kode **Advanced RISC Machines (ARM)** dan x86.

Evolusi Dart

Diluncurkan pada tahun 2011, Dart telah berkembang sejak saat itu. Dart merilis versi stabilnya pada tahun 2013, dengan perubahan besar termasuk dalam rilis Dart 2.0 menjelang akhir 2018, yang dapat diuraikan sebagai berikut:

- Awalnya berfokus pada pengembangan web, dengan tujuan utama menggantikan JavaScript, sekarang telah fokus pada mobile development, termasuk framework Flutter.
- **Mencoba memecahkan masalah pada JavaScript**: JavaScript tidak menyediakan ketahanan seperti banyak bahasa pemrograman lainnya, sehingga Dart ingin menjadi penerus daripada JavaScript.
- **Menawarkan performa terbaik dan alat yang lebih baik untuk proyek berskala besar:** Dart memiliki perkakas yang modern dan stabil yang telah disediakan oleh plugin IDE. Hal ini telah dirancang untuk mendapatkan performa terbaik dengan tetap menjaga nuansa bahasa yang dinamis.
- **Dibentuk agar kuat dan fleksibel:** Dengan tetap mempertahankan type annotations bersifat opsional dan menambahkan fitur OOP, Dart dapat menyeimbangkan dua fitur utama yaitu fleksibilitas dan ketangguhan.

Dart dapat dieksekusi dengan 2 cara, yaitu:

- Dart virtual machine (VMs)
- JavaScript compilation

Structure of the Dart language

- Object orientation
  Seperti kebanyakan bahasa modern, Dart dirancang untuk **object-oriented (OO)**. Secara singkat, Bahasa OOP didasarkan pada konsep **objek** yang menyimpan kedua data

(disebut **fields**) dan kode (disebut **methods**). Objek-objek ini dibuat dari cetak biru yang disebut **class** yang mendefinisikan *field* dan *method* yang akan dimiliki oleh sebuah objek.

- Dart operators
  Di Dart, operator tidak lebih dari method yang didefinisikan dalam class dengan sintaks khusus.
  Jadi, ketika Anda menggunakan operator seperti x == y, seolah-olah Anda sedang memanggil x.==(y) metode untuk melakukan perbandingan kesetaraan.
- Arithmetic operators
  Dart hadir dengan banyak operator *typical* yang bekerja seperti banyak bahasa pemrograman lainnya; yaitu sebagai berikut:
  - + untuk tambahan.
  - - untuk pengurangan.
  - * untuk perkalian.
  - / untuk pembagian.
  - ~/ untuk pembagian bilangan bulat. Di Dart, setiap pembagian sederhana dengan / menghasilkan nilai *double*. Untuk mendapatkan nilai bilangan bulat, Anda perlu membuat semacam transformasi (yaitu, *typecast*) dalam bahasa pemrograman lain; namun Dart sudah mendukung untuk operasi ini.
  - % untuk operasi modulus (sisa bagi dari bilangan bulat).
  - -expression untuk negasi (yang membalikkan suatu nilai).
- Increment and decrement operators
  Operator penambahan dan pengurangan juga merupakan operator umum dan diimplementasikan pada angka, sebagai berikut:
    - ++var atau var++ untuk menambah nilai variabel var sebesar 1
    - --var atau var-- untuk mengurangi nilai variabel var sebesar 1
- Equality and relational operators
  Persamaan operator Dart dijelaskan sebagai berikut:
  - == untuk memeriksa apakah operan sama
  - != untuk memeriksa apakah operan berbeda

  Untuk melakukan pengujian relasional, maka gunakan operator sebagai berikut:

  - > memeriksa apakah operan kiri lebih besar dari operan kanan
  - < memeriksa apakah operan kiri lebih kecil dari operan kanan
  - >= memeriksa apakah operan kiri lebih besar dari atau sama dengan operan kanan
  - <= memeriksa apakah operan kiri kurang dari atau sama dengan operan kanan
- Logical operators
    - `!expression` negasi atau kebalikan hasil ekspresi—yaitu, `true` menjadi `false` dan `false` menjadi `true`.
    - `||` menerapkan operasi logika `OR` antara dua ekspresi.
    - `&&` menerapkan operasi logika `AND` antara dua ekspresi.

DartPad

Dartpad adalah cara termudah untuk memulai pemrograman dart, kita dapat mengeksekusi program dart secara mudah pada platform online.

Main FunctioRingkasan:

Dalam Dart, fungsi dan metode digunakan untuk memisahkan kode. Fungsi menerima data, mengeksekusi kode, dan mengembalikan data. Contoh fungsi utama dalam Dart adalah `main()`, yang harus ada di setiap aplikasi Dart agar Dart VM tahu di mana memulai eksekusi. Fungsi `main()` tidak menerima atau mengembalikan data apa pun, sehingga menggunakan tipe data `void`. Kurung kurawal `{}` digunakan untuk menandai awal dan akhir dari fungsi, berbeda dengan beberapa bahasa seperti Python yang tidak menggunakan kurung kurawal.