

Nama : Fitria Ramadhani Prihandiva

Kelas : SIB 3D / 17

PEMROGRAMAN MOBILE

Selasa, 3 September 2024

Jobsheet 1 - <https://jti-polinema.github.io/flutter-codelab/02-pengantar-bahasa-pemrograman-dart-bag-1/#6>

Soal 1

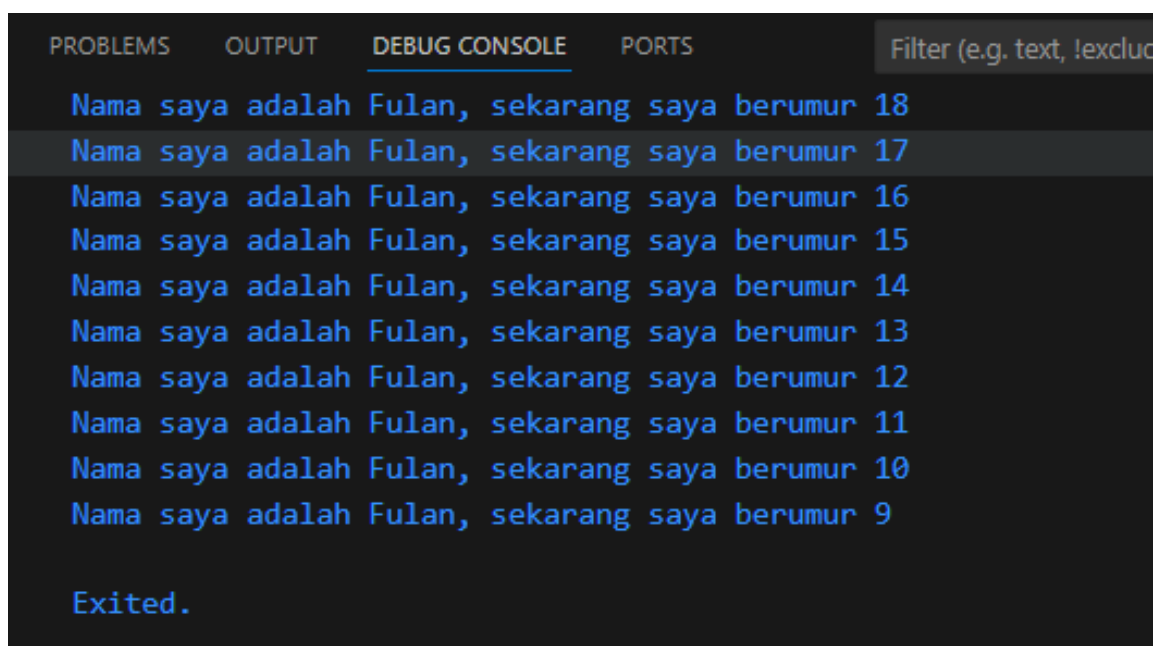
Modifikasilah kode pada baris 3 di VS Code atau Editor Code favorit Anda berikut ini agar mendapatkan keluaran (*output*) sesuai yang diminta!

Code :

```
//FITRIA RAMADHANI PRIHANDIVA
//JS1 - P.MOBILE
void main() {
  //
  //for (int i = 0; i < 10; i++) {
  //  print('hello ${i + 1}');
  //}

  //Soal 1
  for (int i = 19; i > 9; i--) {
    print('Nama saya adalah Fulan, sekarang saya berumur ${i - 1}');
  }
}
```

Hasil Running :



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  PORTS  Filter (e.g. text, !exclud

Nama saya adalah Fulan, sekarang saya berumur 18
Nama saya adalah Fulan, sekarang saya berumur 17
Nama saya adalah Fulan, sekarang saya berumur 16
Nama saya adalah Fulan, sekarang saya berumur 15
Nama saya adalah Fulan, sekarang saya berumur 14
Nama saya adalah Fulan, sekarang saya berumur 13
Nama saya adalah Fulan, sekarang saya berumur 12
Nama saya adalah Fulan, sekarang saya berumur 11
Nama saya adalah Fulan, sekarang saya berumur 10
Nama saya adalah Fulan, sekarang saya berumur 9

Exited.
```

Soal 2

Mengapa sangat penting untuk memahami bahasa pemrograman Dart sebelum kita menggunakan framework Flutter ? Jelaskan!

Jawaban :

Flutter dibangun di atas bahasa pemrograman Dart, sehingga semua kode yang ditulis dalam Flutter sebenarnya adalah kode Dart. Untuk memahami cara kerja Flutter secara mendalam, tentu harus memahami Dart terlebih dahulu. Ini mencakup konsep dasar seperti tipe data, fungsi, kelas, dan struktur kontrol seperti loop dan kondisi.

Soal 3

Rangkumlah materi dari codelab ini menjadi poin-poin penting yang dapat Anda gunakan untuk membantu proses pengembangan aplikasi mobile menggunakan framework Flutter.

Jawaban :

Pentingnya dart

Semua pengembangan framework Flutter melibatkan pengetahuan/fitur mendalam dengan bahasa Dart; Kode aplikasi, kode plugin, dan manajemen dependensi semuanya menggunakan bahasa Dart beserta fitur-fiturnya. Terdapat 2 cara mengeksekusi kode Dart yaitu

1. Dart Virtual Machines (VMs)
2. Javascript Compilations

Eksekusi dart

Kode Dart dapat dieksekusi pada lingkungan yang mendukung bahasa Dart. Lingkungan yang mendukung bahasa Dart perlu memperhatikan fitur-fitur penting seperti berikut:

- *Runtime systems*
- *Dart core libraries*
- *Garbage collectors*

Eksekusi kode Dart dapat beroperasi dalam dua mode — kompilasi **Just-In-Time (JIT)** atau Kompilasi **Ahead-Of-Time (AOT)**. Dijelaskan secara lebih rinci sebagai berikut:

- Kompilasi JIT adalah tempat kode sumber dikompilasi sesuai kebutuhan—*Just in time*. Dart VM memuat dan mengkompilasi kode sumber ke kode mesin asli (*native*). Pendekatan ini digunakan untuk menjalankan kode pada *command line* atau selama proses pengembangan aplikasi mobile yang dapat memanfaatkan fitur seperti debugging dan *hot reload*.
- Kompilasi AOT adalah dimana Dart VM dan kode Anda dikompilasi sebelumnya, VM bekerja lebih seperti sistem runtime Dart, yang menyediakan *garbage collector* dan metode-metode *native* dari Dart **software development kit (SDK)** pada aplikasi.

Pendekatan ini memiliki keuntungan performa yang sangat besar dibandingkan kompilasi JIT, tetapi fitur lain seperti debugging dan hot reload tidak tersedia.

Bahasa dart

Bahasa dart memiliki banyak kesamaan dengan Bahasa Java yang pernah dipelajari sebelumnya. Dart menyediakan sebagian besar operator standar untuk memanipulasi variabel; *built-in types* adalah yang paling umum ditemukan dalam bahasa pemrograman tingkat tinggi. *Control flow* dan *function* sangat mirip dengan bahasa pemrograman lainnya. Dart dirancang untuk Object-Oriented (OOP). Sesuai prinsip OO memastikan bahwa Dart memiliki fitur encapsulation, inheritance, composition, abstraction, dan polymorphism. jika kita telah belajar konsep OO dalam bahasa lain seperti Java, maka sebagian besar desain OO pada Dart akan sangat mirip

Function versus method

Function dan *method* memiliki sintaks yang identik (aturan tentang struktur kodenya), dan sering kali penggunaan ***function*** dan ***method*** digunakan bergantian, lalu apa bedanya? Secara khusus, sebuah *function* berada di luar *class* (kita akan mempelajari tentang *class* pada pertemuan berikutnya). *Function* main adalah contoh di sini. Sebaliknya, sebuah *method* terikat pada turunan *class* dan memiliki referensi secara implisit ke *instance class* melalui keyword *this*.

Soal 4

Buatlah slide yang berisi penjelasan dan contoh eksekusi kode tentang perbedaan Null Safety dan Late variabel ! (**Khusus soal ini kelompok berupa link google slide**)

Kumpulkan jawaban Anda di spreadsheet pada tautan yang telah disediakan di grup telegram. Untuk soal nomor 1 sampai 3 push repo GitHub Anda.

Jawaban :

Link Google Slide : <https://docs.google.com/presentation/d/1pRMMD3Dp-zn9UfwPUqsoycnS2KclXx3G1VjIRZWv61Y/edit?usp=sharing>

Link Github : <https://github.com/prihandiva/PMobile>