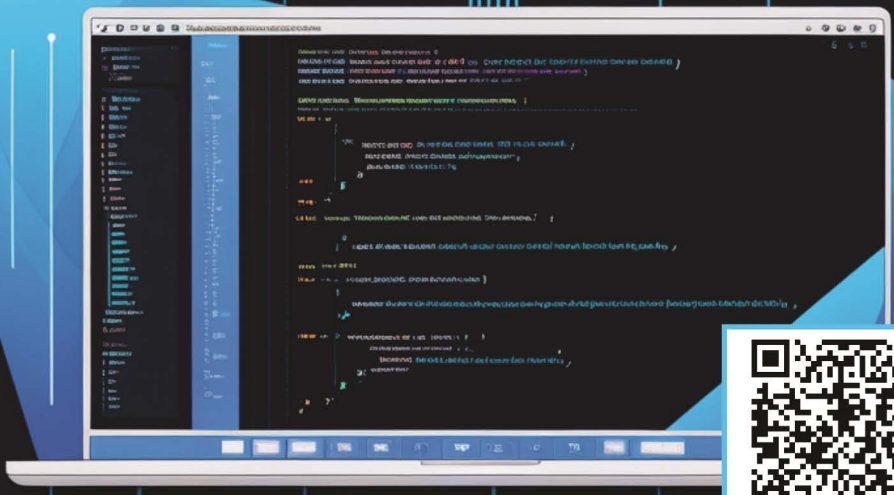
penerbitwidina@gmail.com

PENGANTAR PEMROGRAMAN DART

Panduan Cepat Untuk Pemula



QR Code Book Source

Yudha Islami Sulistya, S.Kom., M.Cs., Maie Istighosah, S.Kom., M.Kom.,
Ir. Huzain Azis, S.Kom, M.Cs., MTA.Abednego Dwi Septiadi, S.Kom., M.Kom.,
Ariq Cahya Wardhana, S.Kom., M.Kom.Rifki Adhitama, S.Kom., M.Kom.,
Arif Amrulloh, S.Kom., M.Kom., Maryona Septiara, S.Pd., M.Kom.

penerbitwidina@gmail.com

penerbitwidina@gmail.com

PENGANTAR PEMROGRAMAN

DART

Panduan Cepat Untuk Pemula

Yudha Islami Sulistya, S.Kom., M.Cs., Maie Istighosah, S.Kom., M.Kom.,
Ir. Huzain Azis, S.Kom, M.Cs., MTA.Abednego Dwi Septiadi, S.Kom., M.Kom.,
Ariq Cahya Wardhana, S.Kom., M.Kom.Rifki Adhitama, S.Kom., M.Kom.,
Arif Amrulloh, S.Kom., M.Kom., Maryona Septiara, S.Pd., M.Kom.

Penerbit widina

penerbitwidina@gmail.com

PENGANTAR PEMROGRAMAN DART
PANDUAN CEPAT UNTUK PEMULA

Tim Penulis:
Yudha Islami Sulistya, S.Kom., M.Cs.
Maie Istighosah, S.Kom., M.Kom.
Ir. Huzain Azis, S.Kom, M.Cs., MTA.
Abednego Dwi Septiadi, S.Kom., M.Kom.
Ariq Cahya Wardhana, S.Kom., M.Kom.
Rifki Adhitama, S.Kom., M.Kom.
Arif Amrulloh, S.Kom., M.Kom.
Maryona Septiara, S.Pd., M.Kom.

Desain Cover:
Septian Maulana

Sumber Ilustrasi:
www.freepik.com

Tata Letak:
Handarini Rohana

Editor:
Neneng Sri Wahyuni, S.H.

ISBN:
978-623-500-772-4

Cetakan Pertama:
Februari, 2025

PENERBIT:
WIDINA MEDIA UTAMA
Komplek Puri Melia Asri Blok C3 No. 17 Desa Bojong Emas
Kec. Solokan Jeruk Kabupaten Bandung, Provinsi Jawa Barat

Anggota IKAPI No. 360/JBA/2020
Website: www.penerbitwidina.com
Instagram: @penerbitwidina
Telepon (022) 87355370

KATA PENGANTAR

Buku ini hadir di tengah berkembangnya teknologi pemrograman yang terus berinovasi, terutama dalam membangun aplikasi yang cepat, modern, dan efisien. Dart, sebagai bahasa pemrograman yang dikembangkan oleh Google, telah menawarkan solusi bagi pengembang yang menginginkan fleksibilitas dalam membuat aplikasi berbasis web maupun mobile. Dengan sintaks yang bersih, pendekatan pemrograman berorientasi objek, dan dukungan kuat dari komunitasnya, Dart menjadi pilihan ideal, terutama dalam pengembangan front-end serta menjadi tulang punggung bagi framework Flutter.

Tim penulis terkesan dengan kesederhanaan sekaligus kekuatan yang ditawarkan oleh Dart. Bahasa ini mudah dipelajari oleh pemula dan menawarkan fitur-fitur modern seperti *null safety*, pemrograman asinkron, dan tools testing yang memudahkan proses pengembangan. Dart memungkinkan pengembang membangun aplikasi dengan performa yang optimal serta dapat dijalankan di berbagai platform, baik di web, server, maupun mobile.

Dengan mempelajari buku ini, pembaca akan dibimbing dari dasar-dasar pemrograman Dart hingga penerapan yang lebih kompleks. Buku ini dirancang untuk pemula yang baru mengenal dunia pemrograman, tetapi juga sangat relevan bagi pengembang berpengalaman yang ingin memperluas keterampilannya dalam pemrograman modern. Setiap bab disusun secara sistematis agar mudah diikuti, dilengkapi dengan contoh-contoh praktis yang relevan dengan kebutuhan pengembangan aplikasi saat ini.

Salah satu kekuatan Dart terletak pada dukungan komunitasnya yang terus berkembang, serta ketersediaan *package* yang memungkinkan pengembang menghemat waktu dan tidak perlu membangun semuanya dari nol. Dalam buku ini, saya juga membahas berbagai teknik dan alat bantu yang tersedia dalam ekosistem Dart, sehingga pembaca dapat sepenuhnya memanfaatkan potensi dari bahasa ini untuk membangun aplikasi yang robust dan berkinerja tinggi.

Tidak hanya teori, buku ini juga menyajikan berbagai contoh kasus dan proyek mini yang relevan, sehingga pembaca dapat langsung mempraktikkan apa yang dipelajari. Dengan demikian, pembaca akan mendapatkan gambaran nyata tentang cara menerapkan konsep-konsep yang ada dalam pengembangan aplikasi modern.

Tim penulis berharap buku ini dapat menjadi inspirasi bagi lebih banyak pengembang untuk berkreasi, serta menghasilkan aplikasi-aplikasi yang bermanfaat bagi masyarakat luas, dengan kualitas yang setara dengan aplikasi-aplikasi yang dibangun menggunakan bahasa lain.

Januari, 2025

Tim Penulis

PRAKATA

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa atas kesempatan, kesehatan, dan bimbingan-Nya sehingga buku ini dapat terselesaikan dengan baik. Buku ini ditulis dengan tujuan memberikan panduan yang lengkap dan praktis bagi para pengembang yang ingin mempelajari dan menguasai **Dart**, bahasa pemrograman yang dikembangkan oleh Google. Dart saat ini menjadi salah satu solusi utama dalam pengembangan aplikasi lintas platform, baik untuk web, server, maupun aplikasi mobile dengan satu basis kode.

Dalam buku ini, Tim penulis berupaya menjelaskan berbagai konsep dasar Dart hingga teknik-teknik pemrograman yang lebih mendalam, yang diperlukan untuk membangun aplikasi modern. Dengan pendekatan yang mudah dipahami, buku ini diharapkan dapat menjadi sumber referensi bagi mereka yang baru memulai perjalanan di dunia pemrograman, sekaligus bermanfaat bagi pengembang berpengalaman yang ingin memperluas keahlian mereka dalam menggunakan Dart.

Proses penulisan buku ini tentu tidak terlepas dari dukungan berbagai pihak. Oleh karena itu, Penulis pertama ingin menyampaikan terima kasih yang sebesar-besarnya kepada orang tua saya yang senantiasa memberikan dukungan dan doa tanpa henti. Kepada ayah saya yang telah berpulang, semoga selalu diberi tempat terbaik di sisi-Nya, dan kepada ibu saya yang selalu saya doakan agar diberikan kesehatan dan kekuatan dalam setiap langkah hidupnya. Dukungan mereka adalah sumber inspirasi dan semangat terbesar dalam hidup saya.

Tim penulis juga ingin mengucapkan terima kasih kepada masing-masing keluarga besar, rekan, dan kolega yang telah memberikan dukungan, masukan, serta kritik yang membangun selama proses penulisan ini berlangsung. Tidak lupa, saya juga mengapresiasi komunitas pengembang Dart yang terus berkembang dan berkontribusi besar dalam memperkaya ekosistem Dart melalui *package*, tutorial, dan bantuan teknis yang sangat berharga.

Harapan Tim penulis, buku ini dapat memberikan manfaat besar bagi pembaca, baik mereka yang baru memulai perjalanan di dunia pemrograman, maupun mereka yang sudah berpengalaman dan ingin memperdalam pemahaman mereka tentang Dart. Semoga buku ini dapat menjadi salah satu referensi yang bermanfaat dalam membangun aplikasi berkualitas tinggi, efisien, dan relevan dengan kebutuhan pengembangan modern.

Akhir kata, Tim penulis mengucapkan terima kasih kepada semua pihak yang terlibat, dan saya berharap buku ini dapat memenuhi harapan pembaca serta memberikan wawasan baru yang berguna bagi kemajuan di bidang pemrograman.

Januari, 2025

Tim Penulis

DAFTAR ISI

penerbitwidina@gmail.com

penerbitwidina@gmail.com

BAB 1
PENDAHULUAN DART

A. APA ITU DART?

Dart adalah bahasa pemrograman tingkat tinggi yang dikembangkan oleh Google. Pertama kali diperkenalkan pada tahun 2011, Dart dirancang dengan tujuan memberikan solusi pemrograman modern untuk pengembangan aplikasi lintas platform, baik untuk web, server, maupun aplikasi mobile. Bahasa ini memiliki sintaks yang sederhana dan mudah dipelajari, serta mendukung berbagai paradigma pemrograman, seperti pemrograman berorientasi objek, pemrograman asinkron, dan pemrograman fungsional.

Dart merupakan bahasa pemrograman yang **dikompilasi** (*compiled*), di mana kode sumber Dart diterjemahkan menjadi kode yang dapat dipahami dan dieksekusi oleh perangkat keras. Proses kompilasi ini dilakukan dalam dua mode utama, yaitu **just-in-time (JIT)** dan **ahead-of-time (AOT)**, yang masing-masing memiliki kelebihan tergantung pada tahap pengembangan atau produksi aplikasi.

Pada mode **just-in-time (JIT)**, kompilasi dilakukan pada saat aplikasi dijalankan (*runtime*). Hal ini memungkinkan pengembang untuk menguji dan mengubah kode secara dinamis tanpa perlu melakukan kompilasi ulang secara penuh. Mode JIT sangat berguna selama proses pengembangan, karena memungkinkan perubahan kode ditampilkan secara langsung, sehingga mempercepat siklus pengujian dan debugging.

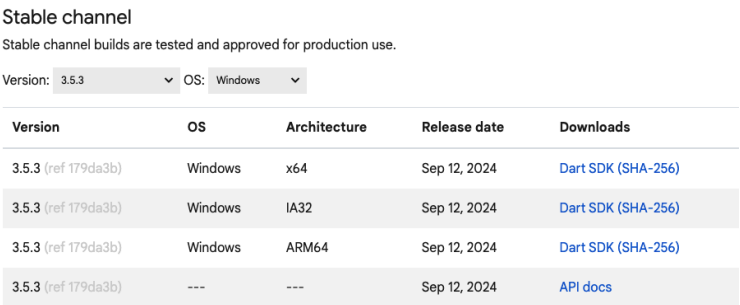
Sementara itu, dalam **ahead-of-time (AOT) compilation**, kompilasi dilakukan sebelum aplikasi dijalankan. Pada tahap ini, seluruh kode Dart diterjemahkan menjadi kode mesin yang dioptimalkan. Mode AOT sangat penting untuk kebutuhan produksi, karena menghasilkan aplikasi dengan performa yang lebih cepat, efisien, dan stabil, terutama pada platform seperti perangkat mobile atau server. Dengan AOT, aplikasi yang dikembangkan dapat dijalankan dengan lebih responsif dan konsumsi sumber daya yang lebih rendah.

Dengan demikian, Dart memanfaatkan keunggulan dari kedua mode kompilasi ini. Mode **JIT** menyediakan fleksibilitas tinggi selama pengembangan, sedangkan **AOT** memberikan performa optimal dalam lingkungan produksi. Kombinasi ini memungkinkan pengembang untuk bekerja secara efisien di berbagai tahap pengembangan dan distribusi aplikasi.

BAB 2
MEMULAI DENGAN DART

A. INSTALASI DART SDK

Untuk memulai pengembangan dengan Dart, langkah pertama yang perlu dilakukan adalah menginstal **Dart SDK** (Software Development Kit). SDK ini bisa diibaratkan sebagai kotak peralatan lengkap bagi seorang pengembang, berisi semua yang diperlukan untuk menulis, menjalankan, dan menguji kode Dart. Di dalam SDK, terdapat **Dart VM** yang bertindak sebagai mesin penggerak utama untuk menjalankan kode Dart, serta **compiler** yang berfungsi menerjemahkan kode agar bisa berjalan lebih cepat dan efisien. SDK ini juga mencakup alat penting seperti **Dart Analyzer** untuk memeriksa kesalahan kode secara otomatis. Dalam proses pengembangan, Dart SDK menjadi fondasi utama, sama halnya seperti seperangkat alat dasar yang diperlukan seorang tukang untuk membangun sebuah rumah.



Stable channel

Stable channel builds are tested and approved for production use.

Version: 3.5.3 OS: Windows

Version	OS	Architecture	Release date	Downloads
3.5.3 (ref 179da3b)	Windows	x64	Sep 12, 2024	Dart SDK (SHA-256)
3.5.3 (ref 179da3b)	Windows	IA32	Sep 12, 2024	Dart SDK (SHA-256)
3.5.3 (ref 179da3b)	Windows	ARM64	Sep 12, 2024	Dart SDK (SHA-256)
3.5.3 (ref 179da3b)	---	---	Sep 12, 2024	API docs

Untuk memulai instalasi Dart SDK dengan mudah, langkah-langkahnya dapat dilakukan melalui situs https://dart.dev/get-dart/archive, di mana tersedia berbagai versi Dart SDK yang sesuai dengan sistem operasi dan arsitektur komputer, seperti Windows, macOS, atau Linux. Berikut ini adalah panduan singkat untuk instalasi Dart SDK:

1. Akses Situs Dart

   Kunjungi halaman https://dart.dev/get-dart/archive untuk memilih versi stabil Dart SDK yang kompatibel dengan sistem operasi yang digunakan. Sebagai contoh OS yang digunakan adalah MacOS, kemudian pilih versi terbaru, seperti **Dart 3.5.3**, dan sesuaikan dengan arsitektur sistem (x64, IA32, atau ARM64).

BAB 3
SINTAKS DASAR DAN TIPE DATA

Sintaks dalam Dart bisa kita bayangkan seperti tata bahasa dalam percakapan sehari-hari. Ketika berbicara, kita mengikuti aturan tertentu agar pesan yang ingin kita sampaikan dapat dipahami dengan baik. Begitu juga dalam Dart, ada aturan atau "tata bahasa" yang harus kita ikuti untuk memastikan program dapat berjalan dengan benar. Misalnya, setiap program dimulai dari fungsi main(), yang bisa kita analogikan sebagai pintu depan rumah. Semua aktivitas dalam rumah (atau program) dimulai dari pintu ini. Kita juga harus menutup setiap pernyataan dengan tanda titik koma ;, seperti ketika kita mengakhiri kalimat dengan tanda titik, agar program tahu kapan satu perintah selesai dan kapan harus lanjut ke perintah berikutnya.

Dalam Dart, tipe data diibaratkan sebagai wadah yang berbeda untuk menyimpan informasi. Seperti di dapur, kita punya gelas untuk air, mangkuk untuk makanan, dan kotak untuk barang-barang kecil. Di Dart, tipe data int digunakan untuk bilangan bulat (seperti gelas yang hanya bisa diisi air), double untuk bilangan desimal, dan String untuk teks. Selain itu, ada bool yang menyimpan nilai benar atau salah, seperti saklar lampu yang hanya punya dua posisi: hidup (true) atau mati (false). Dart juga memiliki tipe var, yang memungkinkan kita untuk tidak menentukan tipe data secara eksplisit—Dart akan otomatis menyesuaikan sesuai isi variabel. Tipe dynamic lebih fleksibel lagi, memungkinkan perubahan tipe data selama eksekusi program, namun penggunaan dynamic perlu hati-hati karena Dart tidak akan lagi memeriksa tipe secara otomatis, yang bisa menyebabkan kesalahan jika tidak dikelola dengan baik.

Dart juga menawarkan beberapa fitur canggih lainnya seperti null safety, yang memastikan variabel tidak bernilai null kecuali secara eksplisit diizinkan, sehingga membantu menghindari kesalahan umum yang terjadi saat mencoba mengakses nilai null. Kemudian, fitur string interpolation mempermudah kita dalam menyisipkan variabel ke dalam teks dengan tanda $, misalnya, kita bisa menulis print('Halo, $nama!') dan Dart akan menggantikan variabel nama dengan nilainya di dalam string. Untuk menampilkan hasil, kita bisa menggunakan fungsi print(), yang mencetak output ke konsol, sangat berguna saat kita ingin melihat hasil sementara atau memeriksa apakah program berjalan sesuai harapan. Ketiga fitur ini—null safety, string interpolation, dan fungsi print()—adalah alat dasar yang

BAB 4
OPERATOR DAN STRUKTUR KONTROL

Dalam pemrograman, operator dan struktur kontrol adalah dua elemen utama yang membantu kita membangun logika dalam program. **Operator** adalah simbol-simbol khusus yang digunakan untuk melakukan operasi pada nilai atau variabel, seperti penjumlahan (+), pengurangan (-), atau perbandingan (==). Operator memungkinkan kita melakukan berbagai macam tugas, mulai dari perhitungan matematika hingga evaluasi logika. Sementara itu, **struktur kontrol** adalah alat yang kita gunakan untuk mengatur alur eksekusi program, seperti pengambilan keputusan dengan **if-else**, atau pengulangan dengan **for** dan **while**.

Bayangkan operator sebagai alat yang kita gunakan untuk bekerja dengan bahan mentah (data), seperti kalkulator yang membantu kita menghitung. Sedangkan struktur kontrol seperti peta jalan yang menentukan ke mana program akan bergerak berdasarkan kondisi tertentu. Dengan menggabungkan keduanya, kita bisa menciptakan program yang dinamis dan responsif terhadap berbagai input dan situasi. Misalnya, kita bisa menggunakan operator untuk mengevaluasi suatu kondisi, dan struktur kontrol untuk menjalankan blok kode tertentu berdasarkan hasil evaluasi tersebut. Dalam Dart, kombinasi keduanya sangat kuat dan memungkinkan kita untuk menulis kode yang lebih terorganisir dan efisien.

A. OPERATOR ARITMATIKA

Ketika kita memprogram, sering kali kita perlu melakukan perhitungan, seperti menjumlahkan angka, mengurangi, atau membagi. Dart menyediakan berbagai operator aritmatika yang memudahkan kita untuk melakukan operasi matematika. Operator ini bekerja seperti alat hitung, di mana kita bisa menambahkan, mengurangi, mengalikan, membagi, atau bahkan mencari sisa hasil pembagian (*modulus*).

Dart memiliki beberapa operator aritmatika dasar yang bisa kita gunakan. Berikut adalah tabel lengkap operator aritmatika beserta fungsinya:

Operator	Nama	Deskripsi	Contoh	Hasil
+	Penjumlahan	Menambahkan dua nilai	5 + 3	8
-	Pengurangan	Mengurangi satu nilai dari nilai lainnya	5 - 3	2
*	Perkalian	Mengalikan dua nilai	5 * 3	15

BAB 5
FUNGSI DALAM DART

A. PENDAHULUAN FUNGSI

Fungsi adalah komponen penting dalam setiap bahasa pemrograman, termasuk Dart. Fungsi memungkinkan pengelompokan logika yang dapat digunakan kembali dan membantu dalam pengorganisasian kode. Dengan menggunakan fungsi, proses pengembangan menjadi lebih modular dan lebih mudah dipahami. Setiap fungsi dalam Dart memiliki struktur dasar yang meliputi deklarasi nama, parameter (jika ada), tipe kembalian, dan tubuh fungsi. Tipe kembalian menunjukkan jenis data yang akan dikembalikan oleh fungsi. Jika fungsi tidak mengembalikan nilai, tipe kembalian dapat diindikasikan dengan void.

Berikut adalah contoh fungsi sederhana dalam Dart:

```
void greet() {
  print("Hello, Dart!");
}
```

Pada contoh di atas, fungsi greet tidak memiliki parameter dan tipe kembalian adalah void, menunjukkan bahwa fungsi ini tidak mengembalikan nilai apa pun.

B. PARAMETER: POSITIONAL DAN NAMED

Dart mendukung dua jenis parameter dalam fungsi: positional dan named. Kedua jenis parameter ini memberikan fleksibilitas dalam cara fungsi dipanggil.

1. Parameter Positional

Parameter positional adalah parameter yang diidentifikasi berdasarkan posisinya dalam daftar argumen. Dalam Dart, parameter ini dapat bersifat wajib atau opsional. Parameter opsional positional ditentukan dengan menggunakan tanda kurung siku [].

BAB 6
PEMOGRAMAN BERORIENTASI OBJEK (OOB)

Pemrograman Berorientasi Objek (**Object-Oriented Programming** atau OOP) adalah paradigma pemrograman yang berfokus pada objek sebagai unit utama untuk membangun aplikasi. Dalam OOP, kita memodelkan masalah dunia nyata ke dalam kode dengan memanfaatkan konsep seperti **kelas** dan **objek**. Sebuah **kelas** dapat dianggap sebagai cetak biru atau template, sedangkan **objek** adalah realisasi konkret dari cetak biru tersebut. Misalnya, jika kita memiliki kelas bernama "Mobil," maka objeknya bisa berupa "Mobil Toyota" atau "Mobil Honda." Pendekatan ini mempermudah kita untuk mengorganisasi kode, meningkatkan skalabilitas, dan mengurangi duplikasi.

OOP juga memperkenalkan empat pilar utama yang membuatnya sangat kuat: **enkapsulasi**, **inheritance (pewarisan)**, **polimorfisme**, dan **abstraksi**. Dengan **enkapsulasi**, kita dapat melindungi data agar hanya dapat diakses melalui metode tertentu, seperti menjaga privasi informasi dalam sebuah sistem. **Inheritance** memungkinkan kita untuk membuat kelas baru dengan mewarisi properti dan metode dari kelas yang sudah ada, yang membantu meminimalkan pengulangan kode. **Polimorfisme** memungkinkan kita menggunakan metode yang sama tetapi dengan cara yang berbeda, mirip seperti seseorang yang bisa menjadi mahasiswa di kelas tetapi juga bekerja di luar kelas. Terakhir, **abstraksi** membantu kita menyederhanakan masalah kompleks dengan hanya fokus pada aspek yang relevan, seperti memahami cara menggunakan mobil tanpa harus tahu detail mesin di dalamnya. Dengan pendekatan ini, OOP membuat pengembangan perangkat lunak lebih mudah dipelihara dan diperluas.

A. PENDAHULUAN OOP

Sebagai paradigma yang berfokus pada kelas dan objek, OOP mengubah cara kita memandang pengembangan perangkat lunak. Dalam OOP, segala sesuatu dipecah menjadi objek, yang memiliki data (atribut) dan perilaku (metode). Hal ini berbeda dengan pendekatan prosedural tradisional yang hanya berfokus pada fungsi dan urutan langkah. OOP memberi kita alat untuk menangani proyek besar dengan lebih terorganisir, di mana setiap bagian dapat dikelola secara terpisah tetapi tetap saling terhubung.

Pendekatan ini sangat relevan dalam pengembangan perangkat lunak modern, di mana kompleksitas aplikasi semakin meningkat. Dengan OOP, kita dapat membangun komponen yang dapat digunakan kembali, seperti

BAB 7
COLLECTIONS DAN ITERASI

Dalam pemrograman, **collections** adalah struktur data yang digunakan untuk menyimpan banyak nilai dalam satu variabel, seperti daftar nama, angka, atau bahkan objek. Dart menyediakan berbagai jenis collections seperti **List**, **Set**, dan **Map**, yang memungkinkan kita untuk menyimpan, mengelola, dan memanipulasi data dengan cara yang efisien. Misalnya, jika kita ingin menyimpan nama-nama teman kita, kita dapat menggunakan **List** untuk menyimpan nama tersebut dalam satu tempat. Collections adalah tulang punggung dalam pengelolaan data di banyak aplikasi, karena memungkinkan kita untuk bekerja dengan kelompok data yang besar dan dinamis.

Agar kita bisa bekerja dengan collections secara efektif, Dart menyediakan berbagai teknik **iterasi** untuk menjelajahi elemen di dalamnya. Iterasi adalah proses menelusuri setiap elemen dalam collections, seperti membaca daftar nama satu per satu. Dart mendukung iterasi melalui perulangan seperti **for**, **for-in**, dan metode modern seperti **forEach**. Bayangkan iterasi seperti membuka buku telepon; kita membaca satu nama pada satu waktu hingga mencapai akhir. Dengan kombinasi collections dan iterasi, kita dapat mengelola dan memproses data secara efisien, menjadikan pengembangan aplikasi lebih terstruktur dan mudah dipahami.

A. LIST: MANIPULASI DAN ITERASI

Ketika kita bekerja dengan data berurutan dalam Dart, **List** adalah salah satu alat yang paling berguna dan fleksibel. Bayangkan **List** sebagai rak buku, di mana setiap rak memiliki urutan dan tempatnya masing-masing. Kita dapat menambahkan buku baru, mengganti buku lama, atau bahkan memindahkan buku dari satu tempat ke tempat lain. Dalam bab ini, kita akan membahas berbagai cara manipulasi dan iterasi pada **List** yang lebih mendalam daripada yang telah kita pelajari sebelumnya.

Dart menyediakan berbagai metode untuk memanipulasi **List**, mulai dari menambahkan elemen baru hingga menghapus atau memfilter data. Kita akan menjelajahi metode-manipulasi yang sering digunakan.

1. Manipulasi List

Dart menyediakan berbagai metode untuk memanipulasi **List**, mulai dari menambahkan elemen baru hingga menghapus atau memfilter data. Kita akan menjelajahi metode-manipulasi yang sering digunakan.

BAB 8
ASYNCHRONOUS PROGRAMMING DENGAN DART

Dalam pengembangan aplikasi modern, banyak tugas yang membutuhkan waktu untuk selesai, seperti mengambil data dari internet, membaca file besar, atau menunggu respons pengguna. Jika kita menangani tugas-tugas ini secara **synchronous** (berurutan), aplikasi kita bisa menjadi lambat dan tidak responsif. Di sinilah konsep **asynchronous programming** menjadi penting. Dart dirancang dengan fitur asinkron bawaan yang memungkinkan kita untuk menangani tugas-tugas ini dengan lebih efisien, tanpa mengorbankan performa atau pengalaman pengguna.

Asynchronous programming di Dart menggunakan konsep **Future** dan **Stream** untuk mengelola tugas yang berjalan secara asinkron. Sebuah **Future** mewakili hasil dari operasi yang mungkin selesai di masa depan, seperti data yang dikembalikan oleh API. Dart juga menyediakan kata kunci **async** dan **await** untuk membuat kode asinkron lebih mudah dipahami dan ditulis, sehingga mirip seperti kode sinkron. Sebagai contoh, kita dapat menggunakan **await** untuk "menunggu" penyelesaian suatu tugas tanpa menghentikan seluruh aplikasi, layaknya seorang pelayan restoran yang melayani tamu lain sambil menunggu pesanan kita selesai dimasak. Dengan pendekatan ini, Dart memungkinkan kita untuk menulis aplikasi yang lebih responsif dan efisien.

A. PENDAHULUAN ASYNC-AWAIT

Ketika kita mengembangkan aplikasi modern, kita sering kali dihadapkan pada tugas-tugas yang membutuhkan waktu untuk selesai. Misalnya, mengambil data dari internet, menyimpan informasi ke dalam database, atau menunggu respons dari API pihak ketiga. Jika kita menangani tugas-tugas ini secara berurutan (sinkron), aplikasi kita mungkin menjadi lambat atau bahkan berhenti sementara menunggu proses selesai. Di sinilah **async-await** dalam Dart hadir untuk menyelamatkan kita.

Bayangkan kita sedang memesan makanan di restoran cepat saji. Setelah memesan, kita tidak perlu berdiri di depan kasir menunggu pesanan selesai. Sebagai gantinya, kita diberi nomor antrean dan bisa duduk santai atau melakukan aktivitas lain sampai nomor kita dipanggil. Itulah gambaran sederhana tentang cara kerja **async-await**: kita meminta Dart untuk menangani tugas asinkron di latar belakang, sementara kita melanjutkan

BAB 10
TESTING DAN DEBUGGING DART

A. UNIT TESTING DI DART

Unit testing adalah metode pengujian perangkat lunak yang berfokus pada pengujian unit terkecil dari sebuah aplikasi, seperti fungsi atau kelas, secara independen. Dalam Dart, unit testing memainkan peran penting dalam memastikan kode berjalan sesuai harapan, sekaligus mempermudah proses debugging dan pemeliharaan program.

Unit testing membantu mengidentifikasi dan memperbaiki bug pada tahap awal pengembangan perangkat lunak. Selain itu, pengujian ini meningkatkan kualitas kode dan mencegah regresi pada fungsi yang sudah ada ketika kode baru ditambahkan.

Manfaat utama dari unit testing meliputi:

- Deteksi dini kesalahan: Pengujian secara spesifik memeriksa fungsi atau modul individu.
- Pemeliharaan kode yang lebih mudah: Kode yang diuji cenderung lebih terstruktur dan terdokumentasi dengan baik.
- Mendukung refactoring: Unit test memastikan bahwa perubahan pada kode tidak mengganggu fungsi yang sudah berjalan.

1. Menggunakan Paket test

Dart menyediakan paket test untuk menulis dan menjalankan unit test. Paket ini memberikan berbagai fitur untuk membuat pengujian yang kuat dan efisien.

a. Langkah-Langkah Implementasi Unit Test

Menambahkan Paket test ke Proyek Tambahkan dependensi berikut ke file pubspec.yaml:

```
dev_dependencies:
  test: ^1.24.0
```

Setelah itu, jalankan perintah berikut untuk memperbarui dependensi:

```
dart pub get
```

GLOSARIUM

A

AOT (Ahead-Of-Time): Kompilasi yang dilakukan sebelum aplikasi dijalankan, menghasilkan kode mesin yang dioptimalkan untuk performa tinggi. Umumnya digunakan pada lingkungan produksi.

Abstract Class: Kelas yang tidak dapat diinstansiasi secara langsung dan berisi metode abstrak (tanpa implementasi) atau konkret (dengan implementasi). Digunakan sebagai kerangka untuk kelas turunan.

Async-Await: Konsep pemrograman asinkron di Dart untuk menangani operasi non-blokir seperti panggilan API atau I/O.

Asynchronous Programming: Pemrograman yang memungkinkan eksekusi tugas tanpa menunggu tugas sebelumnya selesai. Di Dart menggunakan async, await, Future, dan Stream.

B

Boolean (bool): Tipe data primitif di Dart yang hanya memiliki dua nilai: true (benar) atau false (salah).

C

Closures: Fungsi yang dapat mengakses variabel dari lingkup luar tempat fungsi tersebut didefinisikan, bahkan setelah lingkup tersebut selesai dieksekusi.

Constructor: Metode khusus dalam kelas untuk menginisialisasi objek. Di Dart, constructor dapat memiliki parameter opsional atau named parameters.

D

Dart: Bahasa pemrograman modern yang dikembangkan Google untuk membangun aplikasi web, mobile, dan desktop dengan sintaks sederhana dan dukungan null safety.

Dart DevTools: Alat pengembangan untuk debugging, profiling, dan analisis performa aplikasi Dart/Flutter.

Dart Frog: Framework backend minimalis untuk membangun API dengan cepat menggunakan Dart.

Dart SDK: Software Development Kit yang berisi compiler, Dart VM, dan library standar untuk mengembangkan aplikasi Dart.

Dart VM: Mesin virtual untuk menjalankan kode Dart secara efisien, terutama dalam mode JIT (Just-In-Time) selama pengembangan.

Dynamic: Tipe data di Dart yang memungkinkan variabel diubah tipenya selama runtime. Tidak direkomendasikan untuk keamanan tipe.

E

Encapsulation: Prinsip OOP untuk menyembunyikan detail implementasi dengan membatasi akses ke data melalui getter dan setter.

F

Flutter: Framework UI Google untuk membangun aplikasi lintas platform (mobile, web, desktop) menggunakan Dart.
Function: Blok kode yang melakukan tugas spesifik. Di Dart, fungsi bisa memiliki parameter positional, named, atau opsional.

G

Getter: Metode untuk membaca nilai atribut privat dalam kelas, biasanya digunakan untuk enkapsulasi.

H

Higher-Order Function: Fungsi yang menerima fungsi lain sebagai parameter atau mengembalikan fungsi sebagai hasil.

I

Inheritance: Mekanisme OOP di mana kelas turunan mewarisi atribut dan metode dari kelas induk. Di Dart menggunakan kata kunci extends.
Isolates: Mekanisme konkurensi di Dart untuk menjalankan kode di thread terpisah tanpa berbagi memori.
IDE (Integrated Development Environment): Perangkat lunak untuk menulis, menguji, dan debugging kode. Contoh: Visual Studio Code, IntelliJ IDEA.

J

JIT (Just-In-Time): Kompilasi kode saat runtime, memungkinkan hot reload dan debugging dinamis selama pengembangan.

L

List: Struktur data untuk menyimpan koleksi nilai berurutan. Contoh: List<int> numbers = [1, 2, 3];.

penerbitwidina@gmail.com

M

Map: Struktur data yang menyimpan pasangan key-value. Contoh: Map<String, int> ages = {'Alice': 30, 'Bob': 25};.

N

Null Safety: Fitur Dart untuk mencegah null pointer exception dengan memaksa variabel dideklarasikan sebagai nullable (?) atau non-nullable.

O

Object-Oriented Programming (OOP): Paradigma pemrograman berbasis objek dengan empat pilar: enkapsulasi, inheritance, polimorfisme, dan abstraksi.

Operator: Simbol untuk operasi matematika, logika, atau perbandingan. Contoh: +, ==, &&.

P

Package (pub.dev): Repositori pustaka dan alat pihak ketiga untuk mempercepat pengembangan aplikasi Dart/Flutter.

Polymorphism: Kemampuan objek untuk memiliki banyak bentuk. Di Dart diimplementasikan melalui method overriding.

pub.dev: Repositori resmi untuk paket Dart, tempat pengembang berbagi dan menemukan library.

R

Recursion: Teknik di mana fungsi memanggil dirinya sendiri hingga kondisi basis terpenuhi.

S

Serverpod: Framework backend Dart untuk membangun API dengan fitur database, otentikasi, dan manajemen sesi.

Setter: Metode untuk mengubah nilai atribut privat dengan validasi atau logika tambahan.

Shelf: Framework minimalis untuk membangun server HTTP di Dart.

String Interpolation: Teknik menyisipkan nilai variabel atau ekspresi ke dalam string menggunakan $ atau ${}. Contoh: 'Halo, $nama!'.

Super: Kata kunci untuk mengakses metode, atribut, atau konstruktor kelas induk dari kelas turunan.

T

This: Kata kunci yang merujuk ke objek saat ini dalam kelas, sering digunakan untuk menghindari ambiguitas nama.
Try-Catch: Mekanisme penanganan error dengan menjalankan blok kode (try) dan menangkap exception (catch).

V

Var: Kata kunci untuk mendeklarasikan variabel dengan tipe yang ditentukan secara implisit berdasarkan nilai awal.

INDEK

Map
Memory Management
Method
Mixin
Namespace
Null Safety
Object-Oriented Programming (OOP)
Optional Parameter
Override
Package
Parameter
Parsing
Polymorphism
Primitive Type
Procedural Programming
Pub.dev
Recursion
Reflection
Return Type
Runtime
Scope
SDK
Serialization
Set
Singleton
Stack
Static Typing
Stream
String Interpolation
Synchronous Execution
Template
Ternary Operator
Thread
Token
Try-Catch-Finally
Type Inference
Type Safety
Unit Testing

PROFIL PENULIS

Yudha Islami Sulistya, S.Kom., M.Cs.

Seorang dosen dan praktisi di Telkom University Purwokerto dengan keahlian di bidang Flutter development, full-stack web, dan machine learning. Dengan pandangan bahwa teknologi adalah seni, ia menciptakan pengalaman digital yang tidak hanya fungsional tetapi juga berkesan. Lulusan Magister Ilmu Komputer ini terus menjelajahi batasan teknologi, mengintegrasikan kreativitas dan ilmu untuk membangun solusi digital yang berdampak. Bagi Saya, setiap baris kode adalah cerita, dan setiap inovasi adalah langkah menuju masa depan yang lebih cerah.

Maie Istighosah, S.Kom., M.Kom.

Seorang dosen dan praktisi di Telkom University Purwokerto dengan keahlian dalam manajemen data, klasifikasi, serta penerapan machine learning. Dengan pengalaman lebih dari dua tahun di PT Tokopedia dan latar belakang akademik di bidang Informatika dari AMIKOM University Yogyakarta, saya terbiasa dalam memverifikasi, mengelola, dan mengklasifikasikan data dengan tingkat akurasi tinggi. Kemampuan saya dalam pemecahan masalah dan berpikir kritis menjadikan setiap tantangan sebagai peluang untuk menciptakan solusi inovatif. Sebagai pendidik, saya berkomitmen untuk menginspirasi dan membekali generasi masa depan dengan wawasan teknologi yang relevan dan berdampak.

Ir. Huzain Azis, S.Kom., M.Cs., MTA.

Seorang dosen dan peneliti di bidang Teknik Informatika yang telah menjadi bagian dari Fakultas Ilmu Komputer, Universitas Muslim Indonesia sejak 2014. Meraih gelar Magister Ilmu Komputer dari Universitas Gadjah Mada dan saat ini menempuh pendidikan S3 di MIIT University of Kuala Lumpur, beliau mengajar mata kuliah seperti Struktur Data, Data Mining, dan Keamanan Sistem Komputer, dengan pendekatan berbasis teori dan praktik di laboratorium. Selain itu, beliau aktif melakukan penelitian di bidang Data Science, Machine

Learning, Computer Vision, dan Computer Security, serta berupaya mengembangkan teknologi inovatif melalui startup berbasis riset yang bertujuan untuk memberikan manfaat nyata serta meningkatkan standar pendidikan dan keilmuan di bidangnya.

Abednego Dwi Septiadi, S.Kom., M.Kom.

Saya seorang Dosen di Telkom University Purwokerto, 5 tahun sebagai Programmer dan pernah menjadi Project Manager selama 2 tahun serta menguasai bidang Proses Pembangunan Perangkat Lunak, Pengujian Perangkat Lunak dan Manajemen Proyek. Saya lulusan Sarjana Sistem Informasi Amikom Purwokerto dan Magister Teknik Infomatika Universitas Amikom Yogyakarta. Bagi saya pemrograman adalah seni merangkai kode yang dapat menjadikan pengguna lebih mudah dalam menjalankan proses bisnisnya dan kepuasan dapat tercipta saat tujuan tersebut dapat tercapai.

Ariq Cahya Wardhana, S.Kom., M.Kom.

Seorang dosen dan praktisi di Telkom University Purwokerto dengan keahlian di bidang User Experience, Knowledge Management System, dan Software Engineering. Memiliki pengalaman sebagai Software Engineer di berbagai perusahaan rintisan teknologi, saya terbiasa mengembangkan solusi inovatif dengan pendekatan berbasis pengalaman pengguna. Kemampuan saya dalam Web Programming, IONIC, JavaScript, dan Startup Development mendukung perancangan sistem yang efisien dan berorientasi pada kebutuhan pengguna. Dengan semangat kewirausahaan yang kuat, saya berkomitmen untuk menciptakan teknologi yang tidak hanya canggih, tetapi juga memberikan dampak nyata bagi industri dan masyarakat.

Rifki Adhitama, S.Kom., M.Kom.

Seorang dosen dan peneliti di Telkom University Purwokerto (TUP) dengan keahlian dalam Natural Language Processing (NLP), dan software engineering methodology. Seorang peneliti yang meyakini bahwa rekayasa perangkat lunak (software engineering) adalah elemen penting dalam menciptakan kehidupan yang lebih baik bagi masyarakat. Bagi saya, rekayasa

perangkat lunak bukan sekadar disiplin ilmu, tetapi juga sebuah filosofi yang menghubungkan inovasi, efisiensi, dan solusi teknologi untuk menjawab tantangan dunia modern.

Arif Amrulloh, S.Kom., M.Kom.

Seorang dosen dan praktisi di Telkom University Purwokerto dengan keahlian di bidang Kotlin dan Web Development. Bagi saya, programming adalah seni, di mana setiap baris kode bukan sekadar instruksi, tetapi juga medium untuk bercerita dan berkarya. Dengan keahlian dalam mengembangkan aplikasi berbasis web dan mobile, saya selalu berusaha menciptakan solusi yang tidak hanya fungsional, tetapi juga estetis dan bermanfaat. Dengan semangat inovasi dan eksplorasi, saya terus mengasah kemampuan untuk menghadirkan teknologi yang membawa dampak positif bagi banyak orang.

Maryona Septiara S.Pd., M.Kom.

Penulis Bernama lengkap Maryona Septiara, S.Pd., M.Kom. Lahir pada tanggal 27 September 1993 di sebuah desa Simabur, kabupaten Tanah Datar, Provinsi Sumatera Barat. Ia lulusan Sarjana Pendidikan Teknik Informatika dan Komputer Universitas Bung Hatta dan Magister Ilmu Komputer Institut Pertanian Bogor. Saat ini ia telah mengabdi selama 1 tahun sebagai Dosen di Telkom University Purwokerto, dan buku ini merupakan karya pertamanya. Baginya pendidikan merupakan hal yang penting dan harus diutamakan, karena pendidikan merupakan seni untuk membuat manusia makin berkarakter. “Pendidikan adalah senjata paling ampuh yang dapat kamu gunakan untuk mengubah dunia.” – Nelson Mandela

Buku ini adalah panduan komprehensif bagi pemula yang ingin mempelajari bahasa pemrograman Dart secara cepat dan praktis. Dimulai dengan pendahuluan mengenai Dart, buku ini memberikan wawasan tentang sejarah bahasa ini, keunggulannya dibandingkan bahasa lain, serta alasan mengapa Dart merupakan pilihan tepat untuk pengembangan aplikasi modern, terutama untuk aplikasi web dan mobile dengan Flutter.

Buku ini dirancang untuk membantu pembaca memahami konsep dasar hingga tingkat lanjut dengan cara yang mudah diikuti. Di bab pertama, pembaca diperkenalkan dengan Dart, cara instalasi Dart SDK, dan pengaturan lingkungan pengembangan (IDE). Kemudian, pembaca diajak untuk memahami struktur dasar program Dart, serta menggunakan DartPad sebagai editor online.

Melangkah ke bab selanjutnya, pembaca akan mempelajari sintaks dasar dan tipe data, mulai dari variabel, tipe data primitif, hingga konsep penting seperti Null Safety dan manipulasi string. Buku ini juga mencakup penjelasan mengenai operator aritmatika, logika, dan perbandingan yang sangat berguna dalam pengembangan aplikasi sehari-hari.

Selanjutnya, bab-bab yang lebih mendalam menjelaskan tentang fungsi dalam Dart, pemrograman berorientasi objek (OOP), serta penggunaan koleksi dan iterasi dalam pengolahan data. Pembaca juga akan diperkenalkan dengan konsep pemrograman asinkron menggunakan async-await, stream, dan isolates untuk menangani operasi asinkron.

Buku ini tidak hanya fokus pada pemrograman dasar, tetapi juga mengajarkan cara membuat dan menggunakan libraries, testing dan debugging, serta pengoptimalan performa dengan teknik deferred loading dan profiling menggunakan Dart Devtools. Dalam setiap bab, pembaca diberikan contoh kode praktis yang dapat langsung diterapkan untuk memperkuat pemahaman.

Dengan struktur yang jelas, pembaca dapat belajar Dart secara bertahap, dari dasar hingga penerapan dalam proyek nyata, menjadikan buku ini sebagai panduan yang sangat berguna bagi siapa saja yang ingin menguasai bahasa pemrograman Dart.

SCANME
www.penerbitwidina.com
@penerbitwidina
penerbit widina
penerbitwidina@gmail.com
widina store
widina bookstore
Layanan Pembaca & Penerbitan Buku
0815-7000-699

Teknologi dan Komputer - Rp. 77.400
ISBN 978-623-500-772-4
9 786235 007724

penerbitwidina@gmail.com