PEMROGRAMAN MOBILE

MENGENAL BAHASA PEMROGRAMAN: DART

By. ASEP INDRA SYAHYADI M.KOM

01

PENGANTAR

MENGENAL BAHASA PEMROGRAMAN DART

DART

- Dart adalah sebuah bahasa pemrograman yang dikembangkan oleh Google dan merupakan bahasa pemrograman resmi untuk Flutter, sebuah UI toolkit dan aplikasi multiplatform dari Google yang digunakan untuk menembangkan aplikasi multiplatform.
- Dart menawarkan bahasa pemrograman yang produktif dalam pengembangan multiplatfom dengan disandingkan platform runtime eksekusi yang fleksibel (*flexible execution runtime platform*)
- Dart diperioritaskan untuk pengembangandari sisi client dan target kompilasi yang multiplatform (web, seluler, dan desktop).

DART

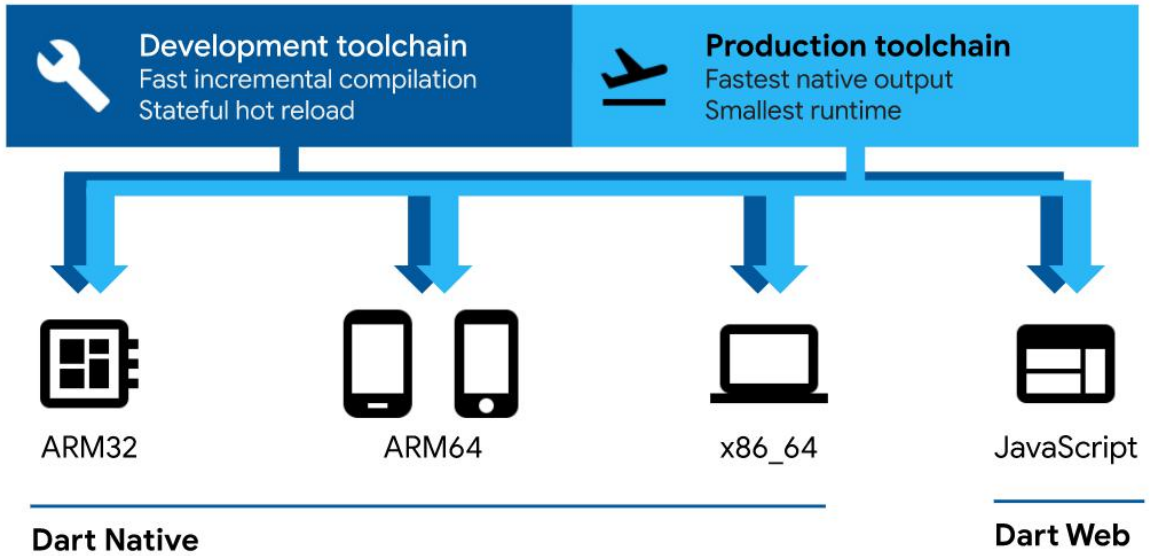
- Dart adalah bahasa pemrograman “type safe”. Patuh terhadap tipe data variabel yang telah dideklarasikan.
- Dart adalah bahasa pemrograman “null safety”. Akan memberikan peringatan terhadapa nilai variabel yang “null” atau tidak didefinisikan
- Dart adalah bahasa pemrograman yang berorientasi objek. mendukung class, objek dan method
- Dart adalah bahasa pemrograman yang “open source” dan didukung banyak librari yang “built-in”

DART: Library

- Dart memiliki banyak kumpulan librari. Berikut librari utama yang sering digunakan
  - Built-in types, collections, and other core functionality for every Dart program (**dart:core**)
  - Richer collection types such as queues, linked lists, hashmaps, and binary trees (**dart:collection**)
  - Encoders and decoders for converting between different data representations, including JSON and UTF-8 (**dart:convert**)
  - Mathematical constants and functions, and random number generation (**dart:math**)
  - File, socket, HTTP, and other I/O support for non-web applications (**dart:io**)
  - Support for asynchronous programming, with classes such as Future and Stream (**dart:async**)
  - Lists that efficiently handle fixed-sized data (for example, unsigned 8-byte integers) and SIMD numeric types (**dart:typed_data**)
  - Foreign function interfaces for interoperability with other code that presents a C-style interface (**dart:ffi**)
  - Concurrent programming using isolates — independent workers that are similar to threads but don’t share memory, communicating only through messages (**dart:isolate**)
  - HTML elements and other resources for web-based applications that need to interact with the browser and the Document Object Model (DOM) (**dart:html**)

DART: Platform

- Teknologi kompiler Dart memungkinkan Anda menjalankan kode dengan cara yang berbeda:
  - **Platform asli** : Untuk aplikasi yang menargetkan perangkat seluler dan desktop, Dart menyertakan Dart VM dengan kompilasi just-in-time (JIT) dan ahead-of-time (AOT) untuk memproduksi kode mesin.
  - **Platform web** : Untuk aplikasi yang menargetkan web, Dart menyertakan kompiler waktu pengembangan (dartdevc) dan kompiler waktu produksi (dart2js). Kedua kompiler menerjemahkan Dart ke dalam JavaScript.



DART: Platform

- Dart Native (kode mesin JIT dan AOT)
  - Selama pengembangan, siklus pengembang yang cepat sangat penting untuk iterasi. Dart VM menawarkan kompiler just-in-time (JIT) dengan kompilasi ulang tambahan (mengaktifkan hot reload), koleksi metrik langsung (menghidupkan DevTools), dan dukungan debugging yang kaya
  - Saat aplikasi siap di-deploy ke produksi — baik Anda memublikasikan ke app store atau men-deploy ke backend produksi — compiler Dart AOT memungkinkan kompilasi sebelumnya ke kode mesin ARM atau x64 asli. Aplikasi yang dikompilasi AOT Anda diluncurkan dengan waktu startup yang singkat dan konsisten.
- Dart Web (pengembang & prod JavaScript)
  - Dart Web memungkinkan menjalankan kode Dart pada platform web yang didukung oleh JavaScript (dart2js). Dengan Dart Web, Anda mengompilasi kode Dart ke kode JavaScript, yang selanjutnya berjalan di browser.

DART: Platform

- Waktu operasi Dart (Runtime). mengeksekusi kode memerlukan runtime Dart. Runtime ini bertanggung jawab untuk tugas-tugas penting berikut:
  - Mengelola memori: Dart menggunakan model memori terkelola, di mana memori yang tidak digunakan diambil kembali oleh pengumpul sampah (GC)
  - Menegakkan sistem tipe Dart: Meskipun sebagian besar pemeriksaan tipe di Dart bersifat statis (waktu kompilasi), beberapa jenis pemeriksaan bersifat dinamis (runtime). Misalnya, waktu proses Dart memberlakukan pemeriksaan dinamis berdasarkan jenis pemeriksaan dan operator transmisi .
  - Waktu proses Dart mengontrol isolat utama (tempat kode biasanya berjalan) dan isolat lain yang dibuat aplikasi

02

MENGENAL BAHASA DART

MENGENAL BAHASA PEMROGRAMAN DART

DART: Dasar

- Program Dart dasar

```
// Define a function.
void printInteger(int aNumber) {
  print('The number is $aNumber.'); // Print to console.
}

// This is where the app starts executing.
void main() {
  var number = 42; // Declare and initialize a variable.
  printInteger(number); // Call a function.
}
```

DART: Dasar

- Konsep Utama
  - Segala sesuatu yang dapat Anda tempatkan dalam variabel adalah objek , dan setiap objek adalah turunan dari kelas. bilangan genap, fungsi, dan null juga merupakan objek. Dengan pengecualian null(jika Anda mengaktifkan sound null safety), semua objek mewarisi dari Objectkelas
  - Jika kita mengaktifkan null safety , variabel tidak dapat berisi null kecuali kita mendeklarasikannya. Kita dapat membuat variabel nullable dengan meletakkan tanda tanya (?) di akhir jenisnya. Misalnya, variabel tipe int? mungkin bilangan bulat, atau mungkin null
  - Dart mendukung tipe generik, seperti List<int>(daftar bilangan bulat) atau List<Object>(daftar objek dari tipe apa pun).
  - Dart mendukung fungsi tingkat atas (seperti main()), serta fungsi yang terkait dengan kelas atau objek (masing-masing metode statis dan instan). Anda juga dapat membuat fungsi di dalam fungsi (fungsi bersarang atau lokal).
  - Tidak seperti OOP lain seperti Java, Dart tidak memiliki kata kunci public, protected, dan private.
  - Alat Dart dapat melaporkan dua jenis masalah: peringatan dan kesalahan

DART: Keyword

- Tabel disamping mencantumkan kata-kata yang diperlakukan secara khusus oleh bahasa Dart.

abstract [2]	else	import [2]	show [1]
as [2]	enum	in	static [2]
assert	export [2]	interface [2]	super
async [1]	extends	is	switch
await [3]	extension [2]	late [2]	sync [1]
break	external [2]	library [2]	this
case	factory [2]	mixin [2]	throw
catch	false	new	true
class	final	null	try
const	finally	on [1]	typedef [2]
continue	for	operator [2]	var
covariant [2]	Function [2]	part [2]	void
default	get [2]	required [2]	while
deferred [2]	hide [1]	rethrow	with
do	if	return	yield [3]
dynamic [2]	implements [2]	set [2]	

DART: Keyword

- Hindari menggunakan kata-kata ini sebagai pengenal (penamaan: variabel, method, class). Namun, jika perlu, kata kunci yang ditandai dengan superskrip dapat menjadi pengidentifikasi:
  - Kata-kata dengan superskrip 1 adalah kata kunci kontekstual , yang hanya memiliki makna di tempat-tempat tertentu. Mereka adalah pengidentifikasi yang valid di mana-mana.
  - Kata-kata dengan superskrip 2 adalah pengidentifikasi bawaan . Kata kunci ini adalah pengidentifikasi yang valid di sebagian besar tempat, tetapi tidak dapat digunakan sebagai nama kelas atau jenis, atau sebagai awalan impor.
  - Kata-kata dengan superskrip 3 adalah kata-kata yang dicadangkan terbatas yang terkait dengan dukungan asinkron . Anda tidak dapat menggunakan “await” atau “yield” sebagai pengenal di badan fungsi apa pun yang ditandai dengan async, async*, atau sync*.

DART: Variabel

- Contoh Membuat Variabel:

```
var nama = 'Bob' ;
```

  - Variabel menyimpan referensi. Variabel “nama” berisi referensi ke sebuah objek **String** dengan nilai "Bob". Nilai dari variabel “name” tidak bisa berisi selain String.
  - Untuk menggunakan variabel dengan tipe data yg dinamis bisa menggunakan type Objek.

```
Object name = 'Bob';
```

DART: Default value

- variabel yang tidak diberikan nilai maka nilai defaultnya adalah *null* (Jika tidak aktif null safety , maka setiap variabel memiliki tipe nullable.):

```
int lineCount
```

- Jika kita mengaktifkan null safety, maka variabel yanf didefinisikan harus memiliki nilai awal

```
int lineCount = 0;
```

DART: late variabel

- variabel yang tidak diberikan nilai maka nilai defaultnya adalah *null* (Jika tidak aktif null safety , maka setiap variabel memiliki tipe nullable.):

```
int lineCount
```

- Jika kita mengaktifkan null safety, maka variabel yanf didefinisikan harus memiliki nilai awal

```
int lineCount = 0;
```

DART: Final & Const variabel

- Jika variabel yang dideklarasikan dan tidak ingin mengubah variabel maka gunakan **final** atau **const ,** baik sebagai pengganti var atau sebagai tambahan untuk suatu tipe

```
final name = 'Bob'; // Without a type annotation
final String nickname = 'Bobby';
```

- Kata const kunci tidak hanya untuk mendeklarasikan variabel konstan. Anda juga dapat menggunakannya untuk membuat nilai konstan , serta mendeklarasikan konstruktor yang membuat nilai konstan. Setiap variabel dapat memiliki nilai konstan.

```
var foo = const [];
final bar = const [];
const baz = []; // Equivalent to `const []`
```

DART: Built in Type

Bahasa Dart memiliki dukungan khusus untuk hal-hal berikut:

- Numbers (int, double)
- Strings (String)
- Booleans (bool)
- Lists (List, also known as arrays)
- Sets (Set)
- Maps (Map)
- Runes (Runes; often replaced by the characters API)
- Symbols (Symbol)
- Nilai null (Null)

Dukungan ini mencakup kemampuan untuk membuat objek menggunakan literal. Misalnya, 'this is a string' adalah string literal, dan “true” literal boolean.

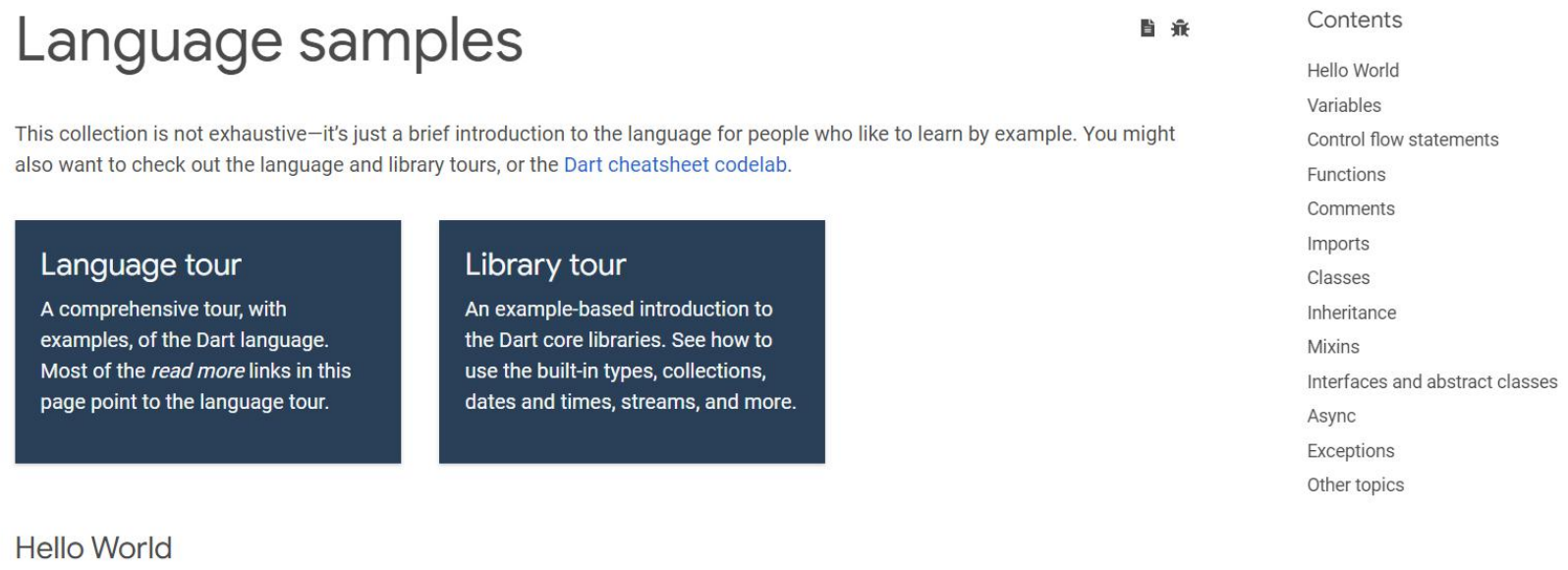
DART: Built in Type

Beberapa jenis lain juga memiliki peran khusus dalam bahasa Dart:

- **Object**: Superclass dari semua kelas Dart kecuali Null.
- **Future** dan **Stream**: Digunakan dalam dukungan asinkron .
- **Iterable**: Digunakan dalam loop for-in dan dalam fungsi generator sinkron .
- **Never**: Menunjukkan bahwa ekspresi tidak akan pernah berhasil menyelesaikan evaluasi. Paling sering digunakan untuk fungsi yang selalu melempar pengecualian.
- **dynamic**: Menunjukkan bahwa Anda ingin menonaktifkan pemeriksaan statis. Biasanya Anda harus menggunakan “Object” atau “Object?” sebagai gantinya.
- **void**: Menunjukkan bahwa suatu nilai tidak pernah digunakan. Sering digunakan sebagai tipe pengembalian.

Tugas Individu

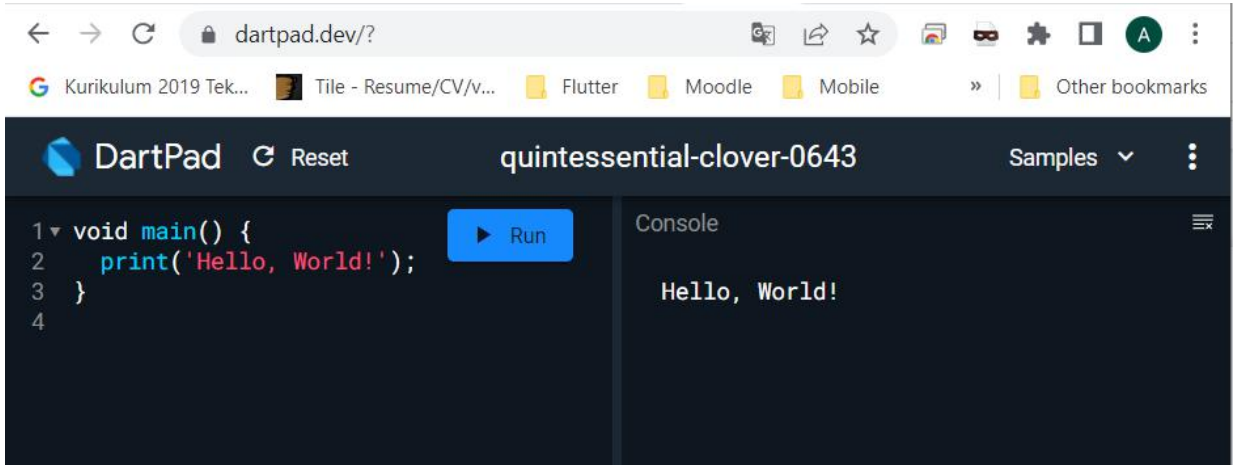
- Silakan akses laman berikut => https://dart.dev/samples
- Akan tampil halaman sebagai berikut



Tugas Individu

- Silakan mempelajari setiap contents yang ada. Lihat menu yg ada di sebelah kanan
- Jalankan Contoh skrip yang ada disetiap konten melalui halaman

  https://dartpad.dev/
- Seperti contoh pada gambar berikut:



Contents

Tugas Individu

- Silakan Rekam Proses ujicoba yang anda lakukan pada setiap konten. Dan jelaskan maksud dari proses yang anda lakukan dan jelaskan skrip yang ditampilkan.
- Hasil Rekaman Di Upload di Youtube dan Link youtube dikirim di lentera.

NEXT

LANJUT PERTEMUAN BERIKUTNYA....


Thank You