

Pemrograman Dart - Sintaks

Sintaks mendefinisikan sekumpulan aturan untuk menulis program. Setiap spesifikasi bahasa menentukan sintaksnya sendiri. Program Dart terdiri dari -

- Variabel dan Operator
- Kelas
- Fungsi
- Ekspresi dan Konstruksi Pemrograman
- Pembuatan Keputusan dan Konstruksi Pendaaran
- Komentar
- Perpustakaan dan Paket
- Typedefs
- Struktur data direpresentasikan sebagai Koleksi / Generik

Kode Dart Pertama Anda

Mari kita mulai dengan contoh "Hello World" tradisional -

[Demo Langsung](#)

```
main() {  
  print("Hello World!");  
}
```

Fungsi **main ()** adalah metode standar di Dart. Metode ini bertindak sebagai titik masuk ke aplikasi. Skrip Dart membutuhkan metode **main ()** untuk eksekusi. **print ()** adalah fungsi pradeфинisi yang mencetak string atau nilai tertentu ke output standar, misalnya terminal.

Output dari kode di atas akan menjadi -

```
Hello World!
```

Jalankan Program Dart

Anda dapat menjalankan program Dart dengan dua cara -

- Melalui terminal
- Melalui WebStorm IDE

Melalui Terminal

Untuk menjalankan program Dart melalui terminal -

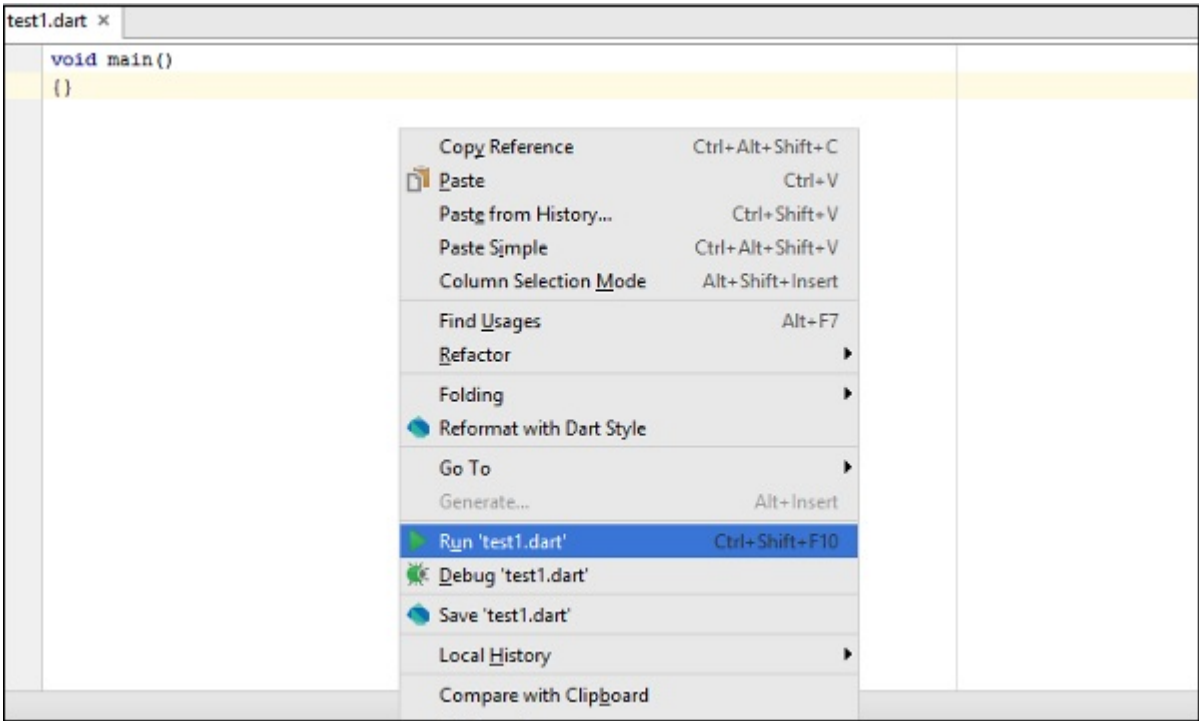
- Arahkan ke jalur proyek saat ini
- Ketik perintah berikut di jendela Terminal


```
dart file_name.dart
```

Melalui WebStorm IDE

Untuk menjalankan program Dart melalui WebStorm IDE -

- Klik kanan file skrip Dart pada IDE. (File harus berisi fungsi **main ()** untuk memungkinkan eksekusi)
- Klik pada opsi '**Jalankan <file_name>**'. Tangkapan layar yang sama diberikan di bawah ini -



Sebagai alternatif, Anda dapat mengklik  tombol atau menggunakan pintasan **Ctrl + Shift + F10** untuk menjalankan Skrip Dart.

Opsi Baris Perintah Dart

Opsi baris perintah Dart digunakan untuk memodifikasi eksekusi Dart Script. Opsi baris perintah umum untuk Dart termasuk yang berikut ini -

Sr.No	Opsi & Deskripsi Baris Perintah
1	-c atau --c Mengaktifkan pernyataan dan pemeriksaan jenis (mode dicentang).
2	--Versi: kapan Menampilkan informasi versi VM.
3	--paket <path> Menentukan jalur ke file konfigurasi resolusi paket.
4	-p <path> Menentukan di mana menemukan perpustakaan yang diimpor. Opsi ini tidak dapat digunakan dengan --packages.
5	-h atau --help Menampilkan bantuan.

Mengaktifkan Mode yang Dicentang

Program Dart berjalan dalam dua mode yaitu -

- Mode Tercentang
- Mode Produksi (Default)

Direkomendasikan untuk menjalankan VM Dart dalam **mode dicentang** selama pengembangan dan pengujian, karena ini menambahkan peringatan dan kesalahan untuk membantu proses pengembangan dan debugging. Mode yang dicentang memberlakukan berbagai pemeriksaan seperti pemeriksaan jenis dll. Untuk mengaktifkan mode yang dicentang, tambahkan opsi -c atau -- yang dicentang sebelum nama file skrip saat menjalankan skrip.

Namun, untuk memastikan keuntungan kinerja saat menjalankan skrip, disarankan untuk menjalankan skrip dalam **mode produksi** .

Pertimbangkan file skrip **Test.dart** berikut -

Demo Langsung

```
void main() {  
  int n = "hello";  
  print(n);  
}
```

Jalankan skrip dengan memasukkan -

```
dart Test.dart
```

Meskipun ada ketidakcocokan jenis, skrip berhasil dijalankan saat mode yang dicentang dimatikan. Skrip akan menghasilkan keluaran berikut -

```
hello
```

Sekarang coba jalankan skrip dengan opsi "- - diperiksa" atau "-c" -

```
dart -c Test.dart
```

Atau,

```
dart - - checked Test.dart
```

VM Dart akan menampilkan kesalahan yang menyatakan bahwa ada jenis ketidakcocokan.

```
Unhandled exception:  
type 'String' is not a subtype of type 'int' of 'n' where  
  String is from dart:core  
  int is from dart:core  
#0  main (file:///C:/Users/Administrator/Desktop/test.dart:3:9)  
#1  _startIsolate.<anonymous closure> (dart:isolate-patch/isolate_patch.dart:  
#2  _RawReceivePortImpl._handleMessage (dart:isolate-patch/isolate_patch.dart:
```

Pengidentifikasi di Dart

Pengenal adalah nama yang diberikan ke elemen dalam program seperti variabel, fungsi, dll. Aturan untuk pengenal adalah -

Pengidentifikasi dapat mencakup keduanya, karakter dan angka. Namun, pengenal tidak boleh dimulai dengan digit.

- Pengidentifikasi tidak dapat menyertakan simbol khusus kecuali untuk garis bawah (_) atau tanda dolar (\$).
- Pengenal tidak boleh menjadi kata kunci.
- Mereka harus unik.

- Pengenal peka huruf besar kecil.
- Pengenal tidak boleh berisi spasi.

Tabel berikut mencantumkan beberapa contoh pengenal yang valid dan tidak valid -

Pengenal yang valid	ID tidak valid
nama depan	Dimana
nama depan	nama depan
num1	nama depan
\$ result	1 angka

Kata kunci di Dart

Kata kunci memiliki arti khusus dalam konteks suatu bahasa. Tabel berikut mencantumkan beberapa kata kunci di Dart.

abstrak 1	terus	Salah	baru	ini
sebagai 1	default	terakhir	batal	melemparkan
menegaskan	ditangguhkan 1	akhirnya	operator 1	benar
asinkron 2	melakukan	untuk	Bagian 1	mencoba
asinkron * 2	dinamis 1	dapatkan 1	melempar kembali	typedef 1
menunggu 2	lain	jika	kembali	dimana
istirahat	enum	mengimplementasikan 1	set 1	kosong
kasus	ekspor 1	impor 1	statis 1	sementara
menangkap	eksternal 1	di	super	dengan
kelas	meluas	adalah	beralih	hasil 2
const	pabrik 1	perpustakaan 1	sinkronisasi * 2	hasil * 2

Spasi Putih dan Jeda Baris

Dart mengabaikan spasi, tab, dan baris baru yang muncul di program. Anda dapat menggunakan spasi, tab, dan baris baru secara bebas dalam program Anda dan Anda bebas untuk memformat dan membuat indentasi program Anda dengan cara yang rapi dan konsisten yang membuat kode mudah dibaca dan dipahami.

Dart peka huruf besar-kecil

Dart peka huruf besar kecil. Ini berarti Dart membedakan karakter huruf besar dan huruf kecil.

Pernyataan diakhiri dengan titik koma

Setiap baris instruksi disebut pernyataan. Setiap pernyataan panah harus diakhiri dengan titik koma (;). Satu baris bisa berisi banyak pernyataan. Namun, pernyataan ini harus dipisahkan dengan titik koma.

Komentar di Dart

Komentar adalah cara untuk meningkatkan keterbacaan program. Komentar dapat digunakan

Komentar adalah cara untuk meningkatkan keterbacaan program. Komentar dapat digunakan untuk memasukkan informasi tambahan tentang program seperti pembuat kode, petunjuk tentang fungsi / konstruksi, dll. Komentar diabaikan oleh kompiler.

Dart mendukung jenis komentar berikut -

- **Komentar satu baris (*//*)** - Teks apa pun di antara *//* dan akhir baris dianggap sebagai komentar
- **Komentar banyak baris (*/* */*)** - Komentar ini dapat mencakup beberapa baris.

Contoh

```
// this is single line comment

/* This is a
   Multi-line comment
*/
```

Pemrograman Berorientasi Objek di Dart

Dart adalah bahasa Berorientasi Objek. Object Orientation adalah paradigma pengembangan perangkat lunak yang mengikuti pemodelan dunia nyata. Orientasi Objek menganggap program sebagai kumpulan objek yang berkomunikasi satu sama lain melalui mekanisme yang disebut metode.

- **Objek** - Objek adalah representasi real-time dari entitas apa pun. Sesuai dengan Grady Brooch, setiap objek harus memiliki tiga fitur -
 - **Negara** - dijelaskan oleh atribut suatu objek.
 - **Perilaku** - menjelaskan bagaimana objek akan bertindak.
 - **Identitas** - nilai unik yang membedakan suatu objek dari sekumpulan objek serupa.
- **Kelas** - Kelas dalam istilah OOP adalah cetak biru untuk membuat objek. Kelas merangkum data untuk objek tersebut.
- **Metode** - Metode memfasilitasi komunikasi antar objek.

Contoh: Orientasi Dart dan Objek

Demo Langsung

```
class TestClass {
  void disp() {
    print("Hello World");
  }
}

void main() {
  TestClass c = new TestClass();
  c.disp();
}
```

Contoh di atas mendefinisikan kelas **TestClass**. Kelas memiliki metode **disp ()**. Metode mencetak string "Hello World" di terminal. Kata kunci **new** membuat objek kelas. Objek memanggil metode **disp ()**.

Kode harus menghasilkan **keluaran** berikut -

```
Hello World
```