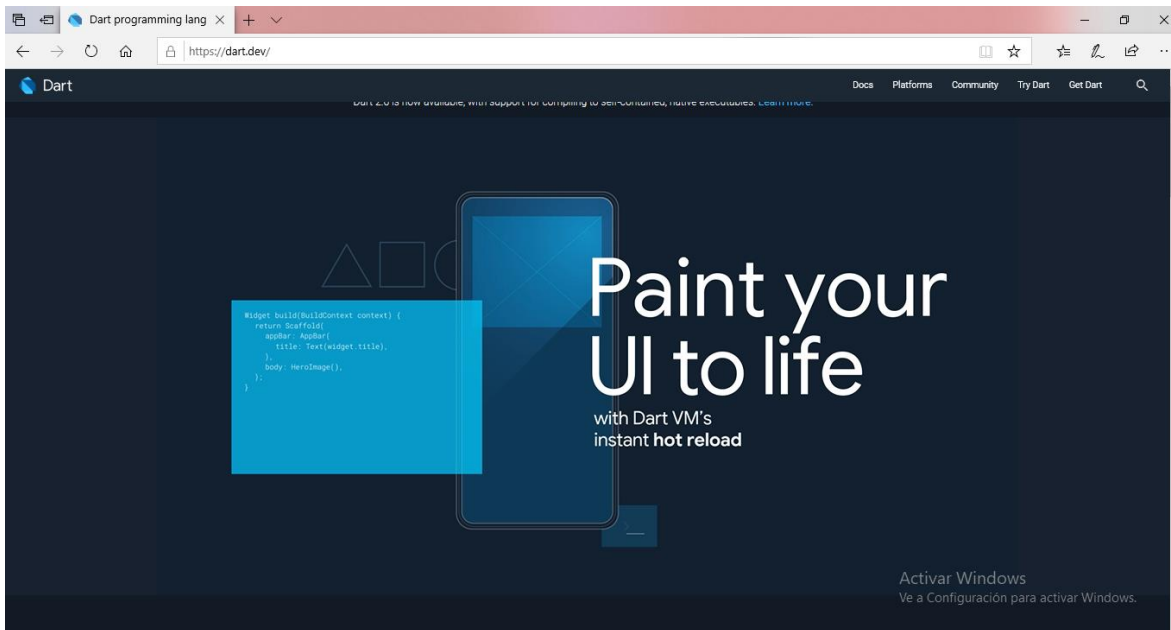




<https://dart.dev/>



Dart lenguaje de proposito general

Codigo Abierto

Desarrollado por Google

Aprobado como estandar por ECMA

Lenguaje de programacion orientado al servidor como al navegador

Contiene un SDK

Transpila equivalente a JavaScript de un Dart Script

Lenguaje Orientado a Objetos


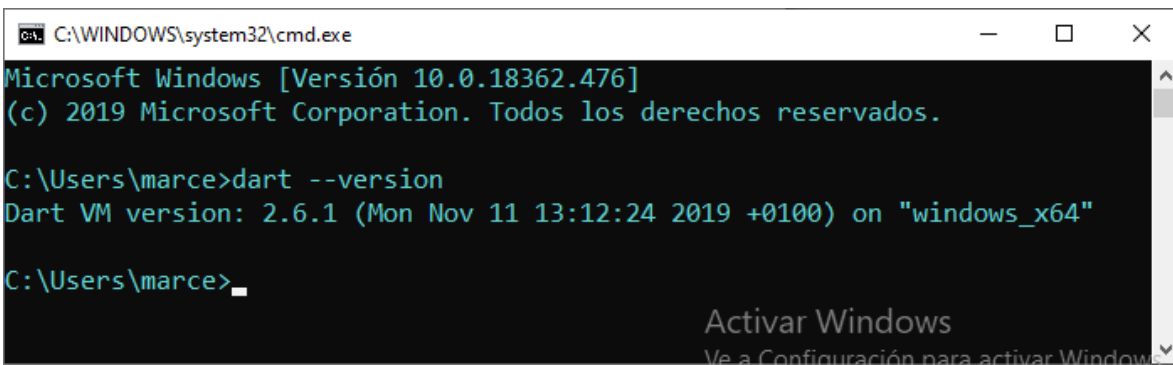
Dart web crea aplicaciones SPA

Chromium basado en Dart VM

Instalación:

<https://dart.dev/get-dart>

<https://gekorm.com/dart-windows/>

 **Dart_x64.stable.setup.exe**

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Versión 10.0.18362.476]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\marce>dart --version
Dart VM version: 2.6.1 (Mon Nov 11 13:12:24 2019 +0100) on "windows_x64"

C:\Users\marce>
```



Lenguaje, palabras clave:

abstract ₂	dynamic ₂	implements ₂	show ₁
as ₂	else	import ₂	static ₂
assert	enum	in	super
async ₁	export ₂	interface ₂	switch
await ₃	extends	is	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	rethrow	while
deferred ₂	hide ₁	return	with
do	if	set ₂	yield ₃

Las palabras con el superíndice **1** son **palabras clave contextuales**, que tienen significado solo en lugares específicos. Son identificadores válidos en todas partes.

Las palabras con el superíndice **2** son **identificadores incorporados**. Para simplificar la tarea de portar código JavaScript a Dart, estas palabras clave son identificadores válidos en la mayoría de los lugares, pero no pueden usarse como nombres de clase o tipo, o como prefijos de importación.

Las palabras con el superíndice **3** son palabras reservadas más nuevas y limitadas relacionadas con el soporte de asincronía que se agregó después de la versión 1.0 de Dart. No se puede utilizar `await` o `yield` como un identificador en cualquier cuerpo de la función marcado con `async`, `async*` o `sync*`.

Un programa en Dart está compuesto de:

- **Variables y Operadores**
- **Las clases**
- **Funciones**
- **Expresiones y construcciones de programación**
- **Toma de decisiones y construcciones en bucle**
- **Comentarios**
- **Bibliotecas y Paquetes**
- **Typedefs**
- **Estructuras de datos representadas como colecciones / genéricos**

Dart soporta:

Numbers

Strings

Booleans

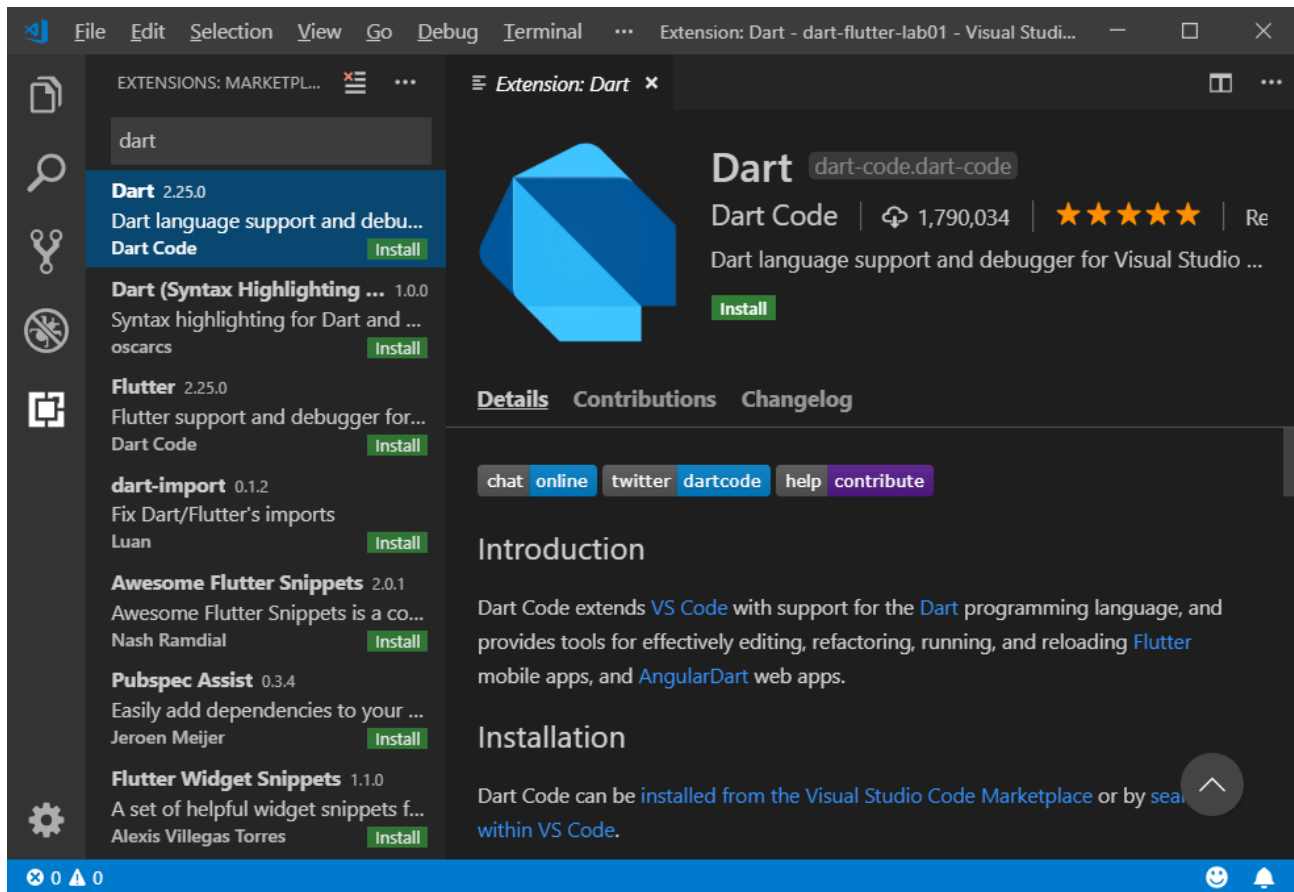
Lists

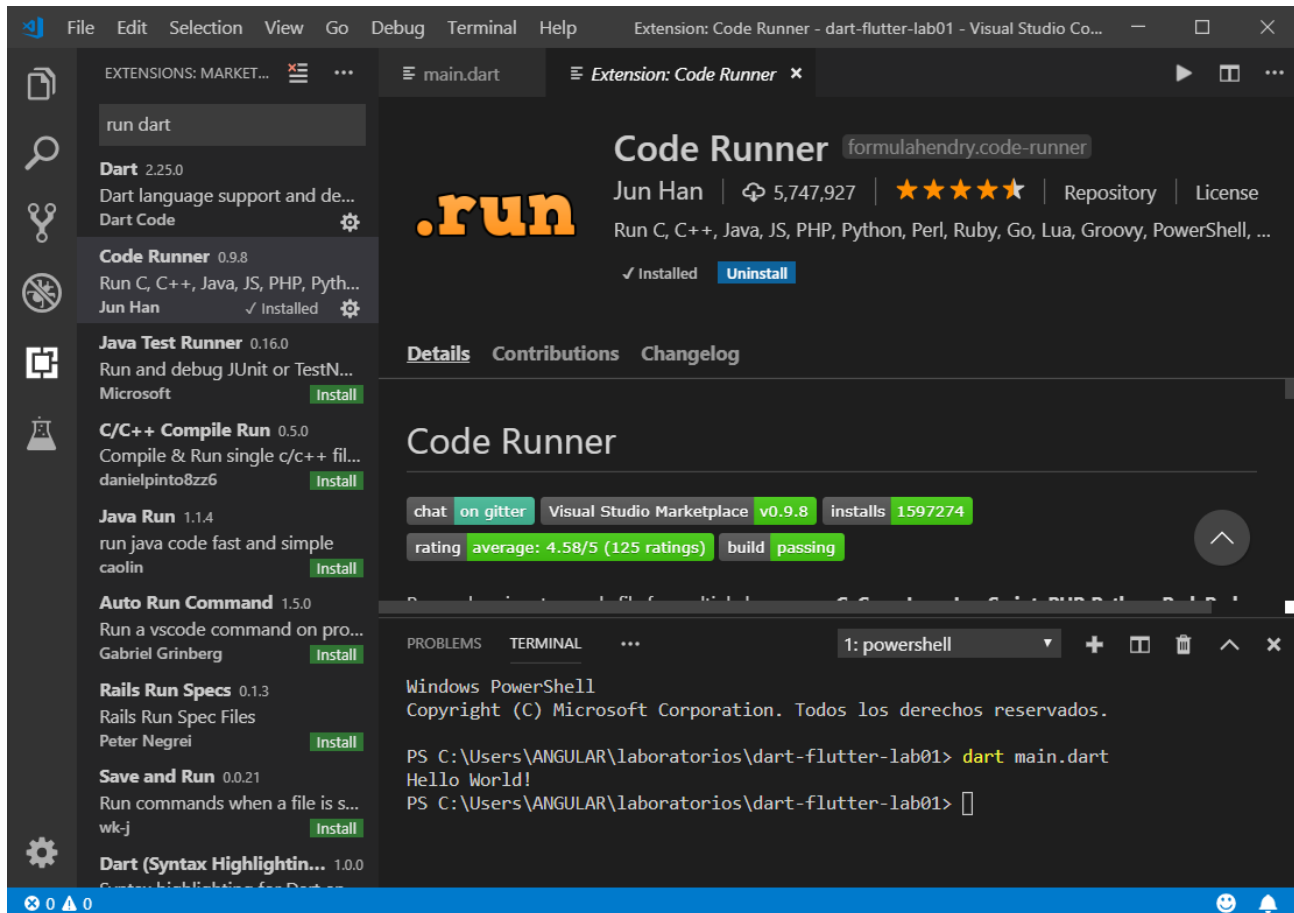
Maps

Comentarios en Dart

Comentarios de una sola línea (//)

Comentarios de varias líneas (/ ** /)





The screenshot displays the Visual Studio Code interface. The left sidebar shows the 'EXTENSIONS: MARKET...' view with a search for 'run dart'. The 'Code Runner' extension by Jun Han is highlighted, showing it is installed. The main panel displays the 'Code Runner' extension page, including its details, contributions, and changelog. The terminal window at the bottom shows the execution of a command in a PowerShell environment.

EXTENSIONS: MARKET...

- run dart**
- Dart** 2.25.0
Dart language support and de...
Dart Code
- Code Runner** 0.9.8
Run C, C++, Java, JS, PHP, Pyth...
Jun Han ✓ Installed
- Java Test Runner** 0.16.0
Run and debug JUnit or TestN...
Microsoft **Install**
- C/C++ Compile Run** 0.5.0
Compile & Run single c/c++ fil...
danielpinto8zz6 **Install**
- Java Run** 1.1.4
run java code fast and simple
caolin **Install**
- Auto Run Command** 1.5.0
Run a vscode command on pro...
Gabriel Grinberg **Install**
- Rails Run Specs** 0.1.3
Rails Run Spec Files
Peter Negrei **Install**
- Save and Run** 0.0.21
Run commands when a file is s...
wk-j **Install**
- Dart (Syntax Highlightin...** 1.0.0
Syntax highlighting for Dart...

Code Runner formulahendry.code-runner

Jun Han | 5,747,927 | ★★★★★ | Repository | License

Run C, C++, Java, JS, PHP, Python, Perl, Ruby, Go, Lua, Groovy, PowerShell, ...

✓ Installed **Uninstall**

Details Contributions Changelog

Code Runner

chat on gitter Visual Studio Marketplace v0.9.8 installs 1597274

rating average: 4.58/5 (125 ratings) build passing

PROBLEMS TERMINAL

1: powershell

Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

PS C:\Users\ANGULAR\laboratorios\dart-flutter-lab01> dart main.dart
Hello World!
PS C:\Users\ANGULAR\laboratorios\dart-flutter-lab01>



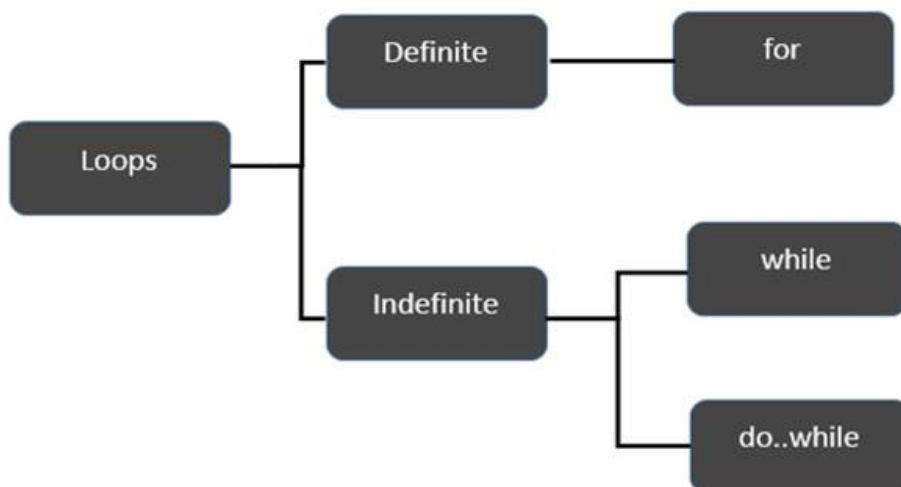
Operadores de Relación

Operator	Description	Example
>	Greater than	(A > B) is False
<	Lesser than	(A < B) is True
>=	Greater than or equal to	(A >= B) is False
<=	Lesser than or equal to	(A <= B) is True
==	Equality	(A==B) is True
!=	Not equal	(A!=B) is True

Operadores Logicos

Operator	Description	Example
&&	And – The operator returns true only if all the expressions specified return true	(A > 10 && B > 10) is False.
	OR – The operator returns true if at least one of the expressions specified return true	(A > 10 B > 10) is True.
!	NOT – The operator returns the inverse of the expression's result. For E.g.: !(7>5) returns false	!(A > 10) is True.

Loops



Tipos incorporados

El lenguaje Dart tiene soporte especial para los siguientes tipos:

- **numbers**
- **strings**
- **booleans**
- **lists**
- **sets**
- **maps**
- **runes** (for expressing Unicode characters in a string)
- **symbols**

Control de Flujos

- **if and else**
- **for loops**
- **while and do-while loops**
- **break and continue**
- **switch and case**
- **assert**

Control de Excepciones

```
Try {  
  } catch (e) {  
  } finally {  
  }  
}
```

</laboratorio>

```
C:\WINDOWS\system32\cmd.exe
C:\Users\marce>dart --version
Dart VM version: 2.6.1 (Mon Nov 11 13:12:24 2019 +0100) on "windows_x64"

C:\Users\marce>cd laboratorio

C:\Users\marce\laboratorio>mkdir dart-lab1

C:\Users\marce\laboratorio>cd dart-lab1

C:\Users\marce\laboratorio\dart-lab1>code .

C:\Users\marce\laboratorio\dart-lab1>_
```

/main.dart

```
void main() {
  print('Hola mundo Dart');
}
```

```
C:\WINDOWS\system32\cmd.exe
C:\Users\marce\laboratorio\dart-lab1>dart main.dart
Hola mundo Dart

C:\Users\marce\laboratorio\dart-lab1>
```



/main.dart

```
void main() {
  print('Hola mundo Dart');

  // variables
  var nombre = 'Ana Gomez';
  var anio = 2011;
  var diametro = 3.7;
  var cursos = ['Flutter', 'Angular', 'Nodejs', 'Reactjs'];

  print('Nombre completo es: ${nombre} de cursos ${cursos}');

  // control y declaracion de flujos

  if (anio >= 2020) {
    print('Futuro ya esta aqui');
  } else if (anio >= 2000) {
    print('Pasado');
  }

  for (var curso in cursos) {
    print(curso);
  }

  for (int mes = 1; mes <= 12; mes++) {
    print(mes);
  }

  while (anio < 2020) {
    anio += 1;
  }
  print('variable anio: ${anio}');

  // funciones
  var num = 5;
  var result = factorial(num);

  print('el factorial de ${num} es ${result}');
```

```
// expresion condicional
var compara =
    result > 100 ? 'El numero es mayor a 100' : 'El numero es menor a 100';
print(compara);

// listas
var lista = new List(3);
lista[0] = 'Sql Server';
lista[1] = 'Oracle';
lista[2] = 'mysql';

print(lista);

lista.forEach((e) => print(e));

// Sets
var linux = {'Ubuntu', 'Centos', 'Debian'};

// Maps
var so = <String>{};
so.add('Windows Server');
so.addAll(linux);

print(so);

var monedas = {
    'BOB' : 'Boliviano',
    'USD' : 'Dolar Americano',
    'EUR' : 'Euro'
};

var estadoHttp = {
    404 : 'No encontrado',
    200 : 'OK',
    401 : 'No autorizado'
};

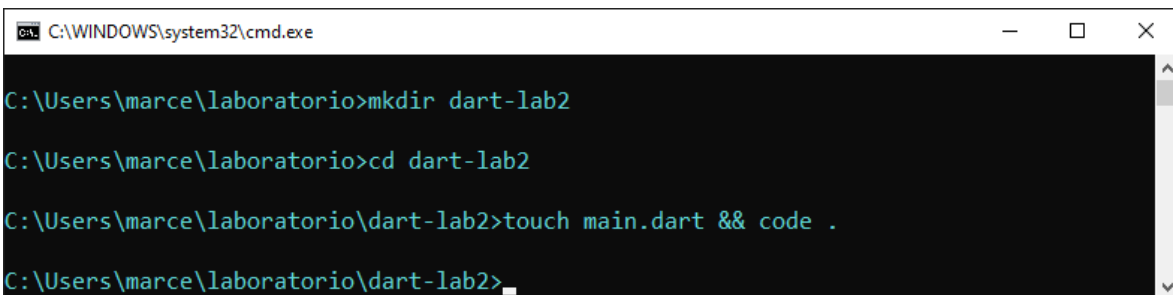
print(monedas['BOB']);
}
```



```
int factorial(int n) {  
  if (n <= 0)  
    return 1;  
  else  
    return (n * factorial(n - 1));  
}
```

Class

```
class class_name {  
  <fields>  
  <getters/setters>  
  <constructors>  
  <functions>  
}
```



```
C:\WINDOWS\system32\cmd.exe  
  
C:\Users\marce\laboratorio>mkdir dart-lab2  
  
C:\Users\marce\laboratorio>cd dart-lab2  
  
C:\Users\marce\laboratorio\dart-lab2>touch main.dart && code .  
  
C:\Users\marce\laboratorio\dart-lab2>_
```



/main.dart

```
class Alumno {  
  String nombre;  
  String apellido;  
  
  String nombreCompleto() {  
    return '${nombre} ${apellido}';  
  }  
}  
  
void main () {  
  
  Alumno alumno = new Alumno();  
  
  alumno.nombre = 'Ana';  
  alumno.apellido = 'Gomez';  
  
  print(alumno.nombreCompleto());  
  
}
```

Librerías

/operaciones.dart

```
import 'dart:math';

class Operaciones {
  num x;
  num y;

  Operaciones(num x,num y){
    this.x = x;
    this.y = y;
  }

  num suma(){
    return this.x + this.y;
  }

  num aleatorio(int n){
    return new Random().nextInt(n);
  }
}
```




/main.dart

```
import 'operaciones.dart';

class Alumno {
  String nombre;
  String apellido;

  String nombreCompleto() {
    return '${nombre} ${apellido}';
  }
}

void main () {

  Alumno alumno = new Alumno();

  alumno.nombre = 'Ana';
  alumno.apellido = 'Gomez';

  print(alumno.nombreCompleto());

  Operaciones operaciones = new Operaciones(14.5, 7);
  print(operaciones.suma());

  print('Numero aleatorios es ${operaciones.aleatorio(10)}');
}
```