Table of Contents

[**Introduction to Dart Language:** 2](#_Toc129224492)

[**Installing Dart** 2](#_Toc129224493)

[**Features in Dart Language:** 2](#_Toc129224494)

[Object-oriented programming: 2](#_Toc129224495)

[Strong typing: 2](#_Toc129224496)

[Optional typing: 2](#_Toc129224497)

[Asynchronous programming: 2](#_Toc129224498)

[Garbage collection: 3](#_Toc129224499)

[Libraries and packages: 3](#_Toc129224500)

[Interoperability: 3](#_Toc129224501)

[Just-in-time (JIT) and ahead-of-time (AOT) compilation: 3](#_Toc129224502)

[Flutter framework: 3](#_Toc129224503)

[**Hello World** 3](#_Toc129224504)

[**Data Types** 4](#_Toc129224505)

[**Numbers** 4](#_Toc129224506)

[**Booleans** 4](#_Toc129224507)

[**Strings** 4](#_Toc129224508)

[**Lists** 5](#_Toc129224509)

[**Maps** 5](#_Toc129224510)

[**Variables** 5](#_Toc129224511)

[**Control Structures** 5](#_Toc129224512)

[**If Statements** 5](#_Toc129224513)

[**For loop** 6](#_Toc129224514)

[**While Loops** 7](#_Toc129224515)

[**Switch Statements** 7](#_Toc129224516)

[**Functions** 8](#_Toc129224517)

[**Classes** 9](#_Toc129224518)

[**Conclusion** 10](#_Toc129224519)

[**References** 10](#_Toc129224520)

# **Introduction to Dart Language:**

Dart is a client-optimized programming language developed by Google. It is an object-oriented language with C-style syntax that can be compiled to native code or JavaScript. Dart was designed to be fast, easy to learn, and flexible.

In this tutorial, we will cover the basics of Dart programming, including data types, variables, control structures, functions, classes, and more.

# **Installing Dart**

Before we start, you need to install the Dart SDK (Software Development Kit) on your system. You can download the latest version of Dart from the official website.

# **Features in Dart Language:**

Dart is a general-purpose programming language that includes a range of features and capabilities, including:

Object-oriented programming:

Dart is an object-oriented language, which means that it supports concepts such as classes, objects, inheritance, and encapsulation.

Strong typing:

Dart is a statically typed language, which means that variable types are checked at compile-time. This helps catch errors early in the development process.

Optional typing:

Dart also supports optional typing, which allows developers to choose whether or not to include type annotations in their code.

Asynchronous programming:

Dart includes built-in support for asynchronous programming using Futures and Streams, which makes it easy to write code that runs concurrently without blocking the main thread.

Garbage collection:

Dart includes automatic memory management through a garbage collector, which helps manage memory usage and prevent memory leaks.

Libraries and packages:

Dart has a rich ecosystem of libraries and packages that can be used to extend its functionality. These packages can be easily installed and managed using the Dart Package Manager (Pub).

Interoperability:

Dart can be used with other languages and technologies through its support for interoperation. For example, it can be used to write native extensions for the Flutter framework or to integrate with JavaScript code running in a web browser.

Just-in-time (JIT) and ahead-of-time (AOT) compilation:

Dart supports both JIT and AOT compilation, which allows developers to choose between fast development and fast execution depending on their needs.

Flutter framework:

Dart is the primary language used to develop apps on the Flutter framework, which is a popular open-source toolkit for building high-performance, high-fidelity, apps for iOS, Android, and the web.

# **Hello World**

Let's start with a simple "Hello, World!" program in Dart:

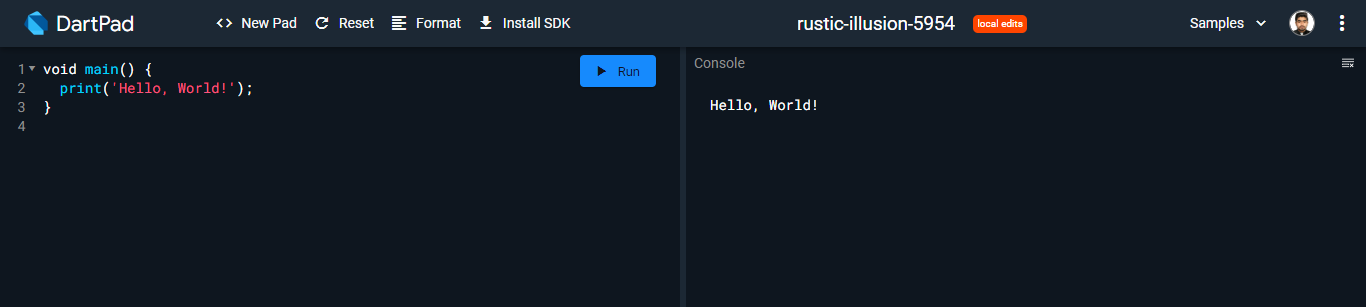
**Code**

void main() {

print('Hello, World!');

}

**Output**



Save this code in a file called hello. Art. To run the program, open your command-line interface and type:

This should output Hello, World! to the console.

# **Data Types**

Dart has several built-in data types, including

* int
* double
* bool
* String
* List
* Map

## **Numbers**

Dart has two types of numbers int and double.

**int age = 42;**

**double price = 9.99;**

## **Booleans**

Dart has a boolean type, which can only have two values: true and false.

**bool isRaining = true;**

**bool isSunny = false;**

## **Strings**

Dart strings can be defined with either single or double quotes.

String firstName = 'John';

String lastName = "Doe";

Dart also supports string interpolation:

String fullName = '$firstName $lastName';

## **Lists**

Dart lists are ordered collections of objects. They can contain objects of any type.

List<int> numbers = [1, 2, 3, 4];

List<String> names = ['Alice', 'Bob', 'Charlie'];

## **Maps**

Dart maps are collections of key-value pairs. The keys and values can be of any type.

Map<String, String> contacts = {

'Alice': '555-1234',

'Bob': '555-5678',

};

# **Variables**

In Dart, variables can be declared using the var keyword or by explicitly specifying their type.

var name = 'Alice';

String message = 'Hello, $name!';

Variables can be reassigned to new values:

var age = 42;

age = 43;

# **Control Structures**

Dart has several control structures, including if, for, while, and switch.

## **If Statements**

void main() {

var marks = 74;

if(marks > 85)

{

print("Excellent");

}

else if(marks>75)

{

print("Very Good");

}

else if(marks>65)

{

print("Good");

}

else

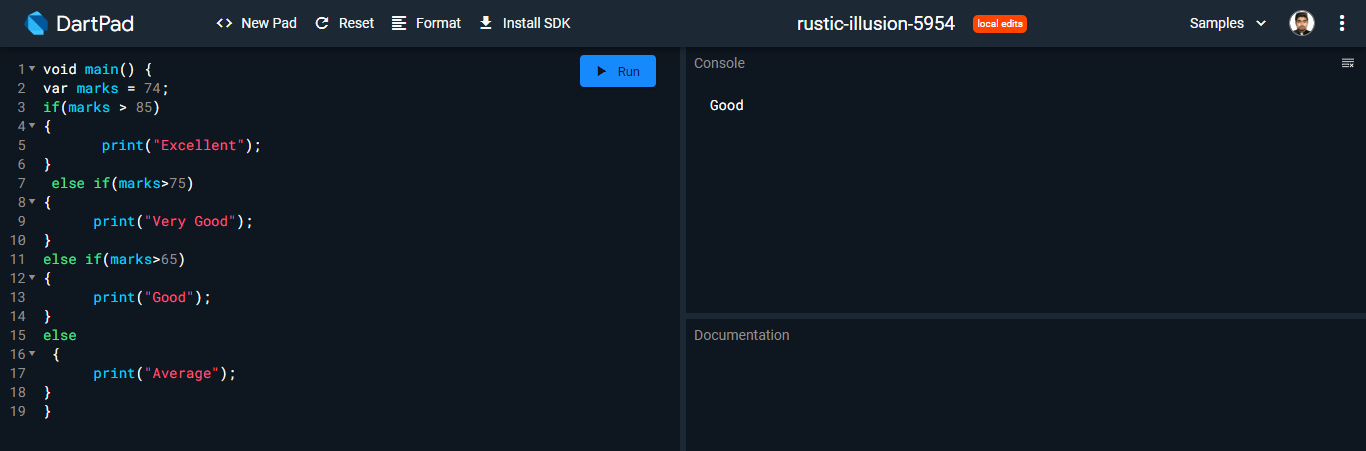
{

print("Average");

}

}

**Output**

****

## **For loop**

**code**

void main()

{

int num = 1;

for(num; num<=5; num++)

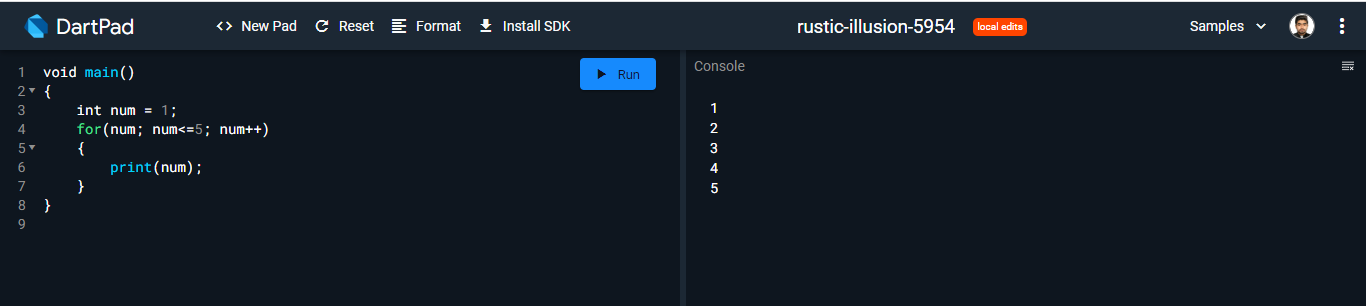
{

print(num);

}

}

**Output**



## **While Loops**

**Code**

void main() {

int current = 0;

while (current < 5) {

current++;

print(current);

}

}

**Output**

****

## **Switch Statements**

void main()

{

int n = 1;

switch (n) {

case 1: {

print("NAEEM");

} break;

case 2: {

print("ALI");

} break;

case 3: {

print("ABDULLAH");

} break;

default: {

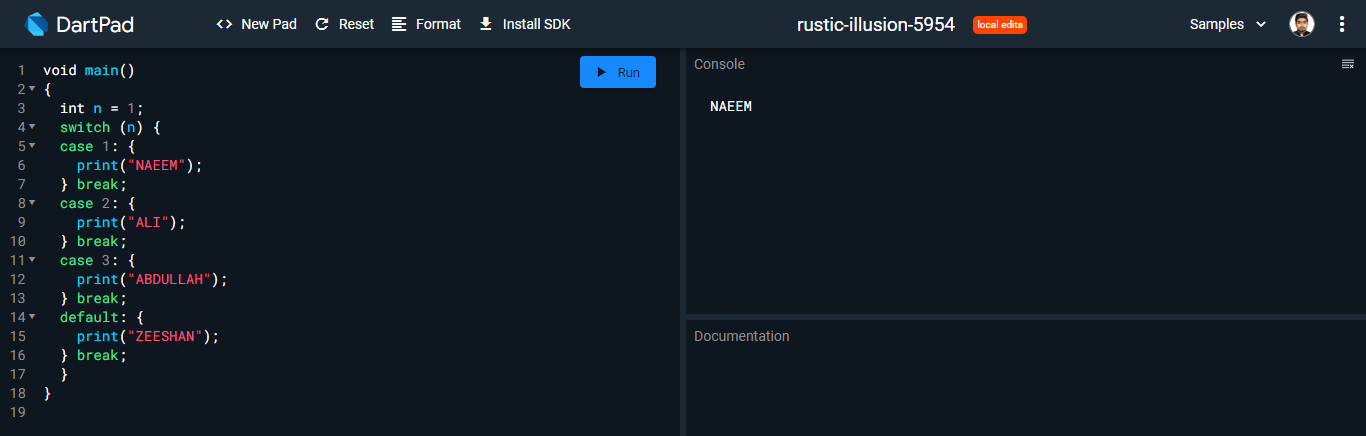
print("ZEESHAN");

} break;

}

}

**Output**

****

# **Functions**

Dart functions are defined using the void keyword for functions that do not return a value, or by specifying the return type of the function using the arrow (=>) syntax.

**Code**

void main() {

int sum = addNumbers(3, 5);

print(sum);

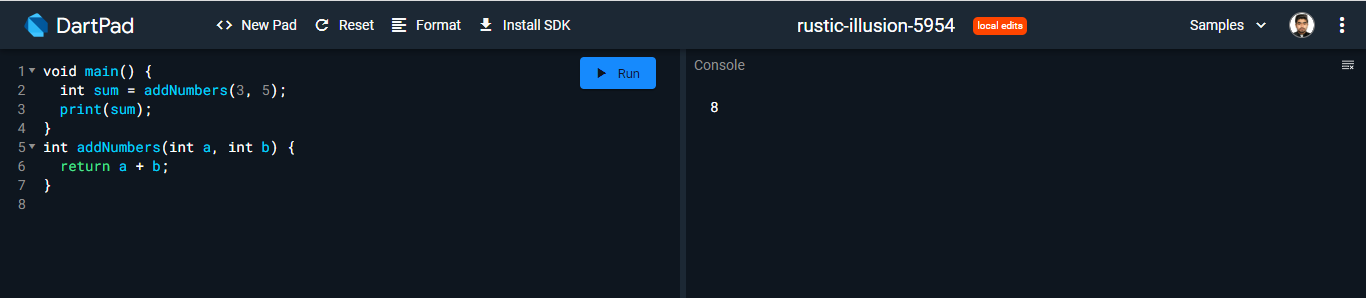
}

int addNumbers(int a, int b) {

return a + b;

}

**Output**

****

# **Classes**

Dart is an object-oriented language, so it has classes for defining objects and their behavior. Here is an example of a simple class:

**Code**

class Person {

String name;

int age;

Person(this.name, this.age);

void sayHello() {

print("Hello, my name is $name and I'm $age years old.");

}

void haveBirthday() {

age++;

print("It's my birthday! Now I'm $age years old.");

}

}

void main() {

Person person = Person("NAEEM", 22);

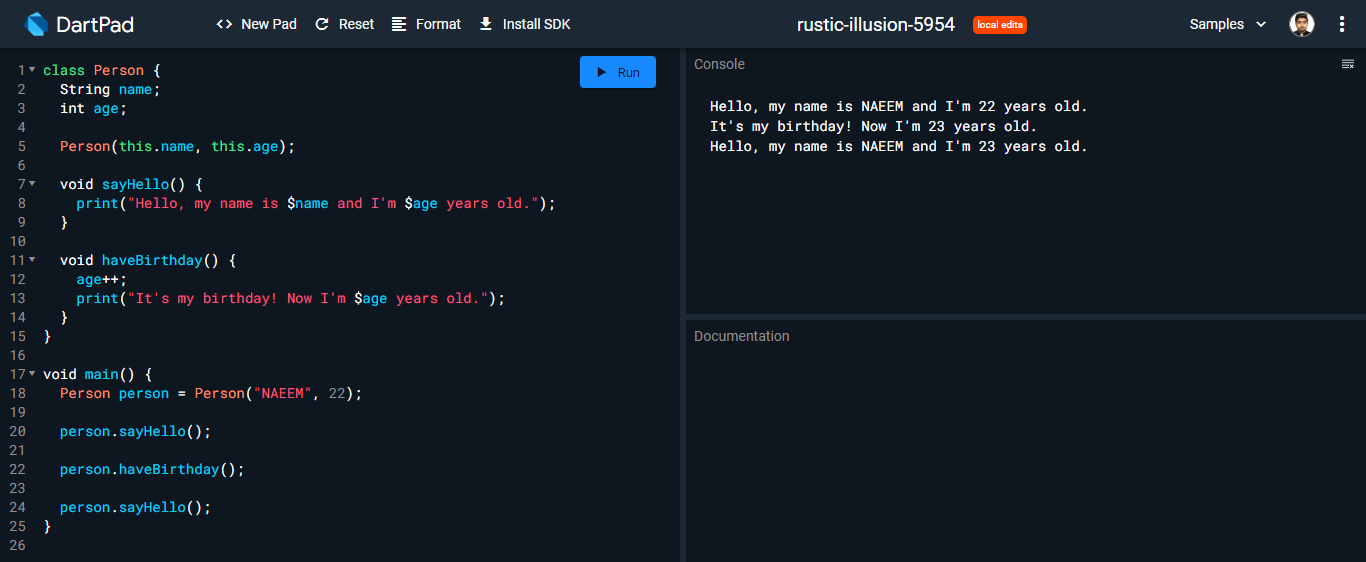
person.sayHello();

person.haveBirthday();

person.sayHello();

}

**Output**

****

# **Conclusion**

That's a brief introduction to the Dart programming language! There are many more features and libraries in Dart, so be sure to check out the official documentation for more information. Happy coding!

# **References**

https://chat.openai.com/chat