DART PROGRAMMING BASICS

Dart is a programming language developed by Google. It is designed for building a wide range of applications, from web and mobile apps to server-side and desktop applications. Dart is known for its simplicity, speed, and productivity. It is used as the primary language for developing apps with **Flutter**, which is Google's open-source UI toolkit for building natively compiled applications for mobile, web, and desktop from a single codebase.

Here are some key points about Dart and why it is used in Flutter:

1. **Efficient and Fast:** Dart is designed for high performance. It features a just-in-time (JIT) compiler for development and a ahead-of-time (AOT) compiler for production, making Flutter apps highly efficient and fast.

2. **Single Codebase:** Dart allows developers to write a single codebase that can run on multiple platforms. This is a crucial advantage for Flutter, as it enables the development of mobile, web, and desktop applications with shared code.

3. **Expressive Syntax:** Dart has a clear and expressive syntax, making it easy to learn and write code. Its syntax is similar to other C-style languages, so developers familiar with languages like JavaScript, Java, or C# will find Dart relatively easy to pick up.

4. **Strongly Typed:** Dart is a statically typed language, which means that variable types are known at compile-time. This helps catch type-related errors early in development, improving code quality.

5. **Garbage Collection:** Dart includes a garbage collector that automatically reclaims memory, reducing the risk of memory leaks and improving the stability of applications.

6. **Modern Features:** Dart includes modern language features like asynchronous programming, null safety, and a rich standard library, making it well-suited for developing complex applications.

7. **Flutter Integration:** Flutter was developed with Dart as its primary language. Dart's features are well-suited for building the user interface and application logic in Flutter, making the development process more efficient and productive.

8. **Community Support:** Dart has a growing community of developers and an active ecosystem with libraries and packages that can be used in Flutter development.

BASICS SYNTAX AND EXAMPLES

Let's go over some basic Dart syntax examples and provide explanations for each. Dart is the programming language used in Flutter for building mobile, web, and desktop applications. After you have downloaded flutter SDKs and configued them to your computer you can copy the examples below and try them out. For you to run the file in your code editor (VS Code) you need to save the file with the .dart extension e.g HelloWord.dart. Here are some fundamental Dart concepts and syntax:

1. **Hello, World! Program:**

```
void main() {
  print('Hello, World!');
}
```

- `void main()`: The entry point of a Dart program. The `main` function is where your program starts.
- `print('Hello, World!')`: The `print` function is used to display text in the console.

2. Variables and Data Types:
Starting from here untill the end, the code should be in the void main(){ //functions,variable ,if statemenst etc }

```
String name = 'Alice';
int age = 30;
double height = 5.7;
bool isStudent = true;
```

- Dart is statically typed, so you specify the data type when declaring variables (`String`, `int`, `double`, `bool`).

3. **Conditional Statements (if-else)**:

```
int temperature = 25;

if (temperature > 30) {
  print('It\'s hot outside.');
} else if (temperature > 20) {
  print('It\'s a pleasant day.');
} else {
  print('It\'s cold outside.');
}
```

- Dart supports common conditional statements like `if`, `else if`, and `else`.

4. **Loops (for loop):**

```
for (int i = 0; i < 5; i++) {
  print('Iteration $i');
}
```

- A `for` loop is used for iteration. In this example, it runs five times, printing the current iteration number.

5. **Lists:**

```
List<String> fruits = ['Apple', 'Banana', 'Cherry'];

for (String fruit in fruits) {
  print('I like $fruit.');
}
```

- Dart lists are ordered collections. Here, we declare a list of strings and iterate through it.

6. **Functions:**

```
int add(int a, int b) {
  return a + b;
}

void main() {
  int result = add(3, 5);
  print('The sum is $result');
}
```

- Functions in Dart are declared using the `returnType functionName(parameters) { ... }` syntax. In this example, we define an `add` function to calculate the sum of two numbers.

7. **Classes and Objects:**

```
class Person {
  String name;
  int age;

  Person(this.name, this.age);
```

```
  void greet() {
    print('Hello, my name is $name, and I am $age years old.');
  }
}

void main() {
  var person = Person('Alice', 30);
  person.greet();
}
```

- Dart supports object-oriented programming. Here, we define a `Person` class with a constructor and a `greet` method.

8. Comments:
Comments are use for documentation they are two types /* */ or the //

```
/* your name  ,about program etc*/
//your  name ,about the program
```

Summary

Dart is used in Flutter because it is a highly efficient, versatile, and modern programming language that complements Flutter's goals of providing a powerful and consistent framework for building native mobile, web, and desktop applications. Its ability to compile code for multiple platforms from a single codebase simplifies cross-platform development, making it a popular choice for developers who want to create high-quality, performant applications.

Dart is a versatile language with many features for building complex applications. Learning Dart is a crucial step in becoming proficient in Flutter development. You can explore more advanced Dart features and Flutter-specific concepts as you progress in your learning journey.

Prepared by: Kondwani Nyirenda
Contact: +260960322980
GDG Kitwe