Introduction to dart

Dart is a modern, object-oriented programming language developed by **Google**. It is specifically designed for building high-performance, cross-platform applications. Dart offers a clean syntax, strong typing, and a wide range of built-in libraries, making it a powerful tool for app development.

Why Learn Dart ?

Learning Dart can be beneficial for several reasons, particularly if you are interested in mobile app development, web development, or server-side development. Here are some reasons why learning Dart might be a good idea:

1. **Flutter Framework:** Dart is the primary language used for developing mobile applications with Flutter. Flutter is a UI toolkit developed by Google that allows you to create natively compiled applications for mobile, web, and desktop from a single codebase. Flutter has gained popularity for its hot reload feature, expressive UI, and excellent performance.

2. **Cross-Platform Development:** Dart, in conjunction with Flutter, enables you to build cross-platform applications. This means you can write code once and run it on both iOS and Android platforms, saving time and effort compared to developing separate codebases for each platform.

3. **Fast Development:** Dart's hot reload feature in Flutter allows developers to instantly see the effect of the changes made in the code without restarting the entire application. This significantly speeds up the development process and makes it easier to experiment with different UI designs and features.

4. **Strongly Typed Language:** Dart is a statically-typed language, which means that it helps catch errors during development before the code is executed. This can lead to more reliable and maintainable code, especially in larger projects.

5. **Asynchronous Programming:** Dart has built-in support for asynchronous programming, making it well-suited for tasks that involve handling multiple operations simultaneously. This is particularly useful for developing responsive and efficient applications, especially in scenarios like web development.

6. **Growing Ecosystem:** Dart's ecosystem is growing, with an increasing number of packages and libraries available through the Dart Package Manager (pub.dev). This can save developers time by leveraging existing solutions for common tasks.

7. **Google's Support:** Dart is backed by Google, and the company actively uses it in various projects, including some high-profile ones.

This support provides a level of confidence in the language's stability and future development.

8. **Server -Side Development:** While Dart is widely known for client-side development with Flutter, it can also be used for server-side development. The Dart SDK includes libraries for building server-side applications, making it a versatile language for full-stack development.

Getting Started with Dart:

To start writing Dart code, you'll need to set up your development environment:

1. Install Dart SDK:

Visit the official **Dart website** and download the **Dart SDK** for your operating system. The SDK includes the Dart compiler and various tools necessary for running Dart programs.

2. Choose an IDE:

While Dart can be written in any text editor, using an **Integrated Development Environment** (IDE) like **Visual Studio Code** (VS Code) or Android Studio can enhance your coding experience.

Your First Dart Program:

Now that your environment is set up, let's write your first Dart program. Open your preferred text editor or IDE and follow these steps:

1. Create a new Dart file with the **'.dart'** extension, such as **'hello_world.dart'**.

2. In your Dart file, start with the following code:

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

3. Save the file and navigate to its location using the terminal or command prompt.

4. Run the Dart program by executing the command:

dart hello_world.dart.

You should see the output **"Hello, World!"** displayed in the console.

Dart Syntax Basics:

Dart has a clean and intuitive syntax that makes it beginner-friendly. Here are a few key concepts to keep in mind:

- **Statements:** Dart programs consist of a series of statements, each ending with a **semicolon (;).**

  The print statement in the previous example is used to output text to the console.

- **Functions:** In Dart, functions are blocks of reusable code. The **main** function serves as the entry point for Dart programs.
- **Variables:** Variables hold values that can be used and manipulated throughout your code. In Dart, you can declare variables using the **"var"** keyword or specify their types explicitly.
- **Comments:** Comments are used to provide explanations or annotate your code. In Dart, single-line comments start with **//**, while multi-line comments are enclosed in /* ... */.

Dart Features:

- **Open Source:**

  Dart is an **open-source** programming language, which means it is freely available. It is developed by Google, approved by the ECMA standard, and comes with a BSD license.

- **Platform Independent:**

  Dart supports all primary operating systems such as **Windows, Linux, Macintosh,** etc. The Dart has its own Virtual Machine which known as Dart VM, that allows us to run the Dart code in every operating system.

- **Object-Oriented:**

  Dart is an object-oriented programming language and supports all oops concepts such as classes, inheritance, interfaces and optional typing features. It also supports advance concepts like mixin, abstract, classes, reified generic, and robust type system.

- **Concurrency:**

  Dart is an asynchronous programming language, which means it supports multithreading using Isolates. The isolates are the independent entities that are related to threads but don't share memory and establish the communication between the processes by the message passing.

- **Extensive Libraries:**

  Dart consists of many useful inbuilt libraries including **SDK (Software Development Kit), core, math, async, math, convert, html, IO**, etc. It also provides the facility to organize the Dart code into libraries with proper namespacing. It can reuse by the import statement.

- **Easy to learn:**

  As we discussed in the previous section, learning the Dart is not the Hercules task as we know that Dart's syntax is similar to Java, C#, JavaScript, kotlin, etc. if you know any of these languages then you can **learn easily the Dart.**

- **Flexible Compilation:**

  Dart provides the **flexibility** to **compile** the code and fast as well. It supports two types of compilation processes, AOT (Ahead of Time) and JIT (Just-in-Time). The Dart code is transmitted in the other language that can run in the modern web-brewers.

- **Type Safe:**

  The Dart is **the type safe language**, which means it uses both static type checking and runtime checks to confirm that a variable's value always matches the variable's static type, sometimes it known as the sound typing.

- **Objects:**

  The Dart treats everything as an **object**. The value which assigns to the variable is an object. The functions, numbers, and strings are also an object in Dart. All objects inherit from Object class.

- **Browser Support:**

  The **Dart supports** all modern **web-browser**. It comes with the dart2js compiler that converts the Dart code into optimized JavaScript code that is suitable for all type of web-browser.

- **Community:**

  Dart has a large community across the world. So if you face problem while coding then it is easy to find help. The dedicated developers' team is working towards enhancing its functionality.

Applications of Dart Language

- **Web Development:** Dart shines in **web development**, offering frameworks like Flutter development and AngularDart. Flutter allows you to build high-performance, cross-platform mobile and web applications with a single codebase. AngularDart is a Dart version of the popular Angular framework, providing a seamless experience for building dynamic and responsive web applications.
- **Mobile App Development:** Dart, with the Flutter framework, has gained significant popularity in the mobile app development space. Flutter allows developers to create beautiful and native-like apps for both iOS and Android platforms using a single codebase, thereby reducing development time and effort applications of dart programming language

- **Server-Side Development:** You can use Dart for server-side development to build scalable and performant backends. Dart's strong typing and asynchronous capabilities make it well-suited for creating efficient server applications.
- **Internet of Things (IoT):** You can also utilize Dart in IoT development, providing an efficient and flexible language for building applications that connect and interact with IoT devices.
- **Command-Line Tools:** Dart's flexibility and ease of use make it a great choice for building command-line tools and utilities, enabling developers to create efficient and scalable automation scripts.