

Lab 4 Report
Melvin Viado
100899671
INFT 3101-02
November 1, 2024



Contents

No table of contents entries found.

Main Content

Research Findings

Topic

My last name is Viado, therefore my topic is about Packages.

Definition

Dart packages are code libraries or collections of pre-written code that you are able to use to perform certain tasks, or to implement certain types of functionalities or features. Therefore, dart packages consist of highly reusable code that you are able to utilize across your dart projects.

Reason for Usage

Using packages can greatly improve your efficiency and effectiveness in terms of working and developing. Dart packages create this effect because it allows you to use very useful pre-existing code that offers a variety of functionality to your projects.

Comparison

Dart Packages in compared to Dart classes. Dart is an object-oriented programming language that uses principles such as inheritance. With inheritance, existing classes with properties and methods can be used to create new classes, allowing you to reuse code from that close without duplicating code. Using packages are similar to Dart classes in the sense that you can reuse code. The difference between these concepts is that you are not duplicating code during runtime with packages compared to when using class inheritance.

Advantages and Limitations

Advantages

Using packages increases your productivity through functional existing code. Developers should focus more on the logic of their projects rather than wasting time not using packages. Valuable time is wasted by trying to make something that has been already successfully made by someone else. In this case, code has been already made and has been collected into these Dart packages.

Limitations

Limitations for using Dart packages are that some packages may not work well together and have some interferences. Developers may also hide how these packages work and how they perform tasks. This can prevent developers from learning how they can implement these functionalities themselves which can hinder the entire development process of a given project.

References

A comprehensive guide to dart pub. The Ultimate Guide to Dart Pub: Mastering Dart Packages. (n.d.). <https://www.dhiwise.com/post/dart-pub-a-guide-to-dart-packages-and-flutter-pub>

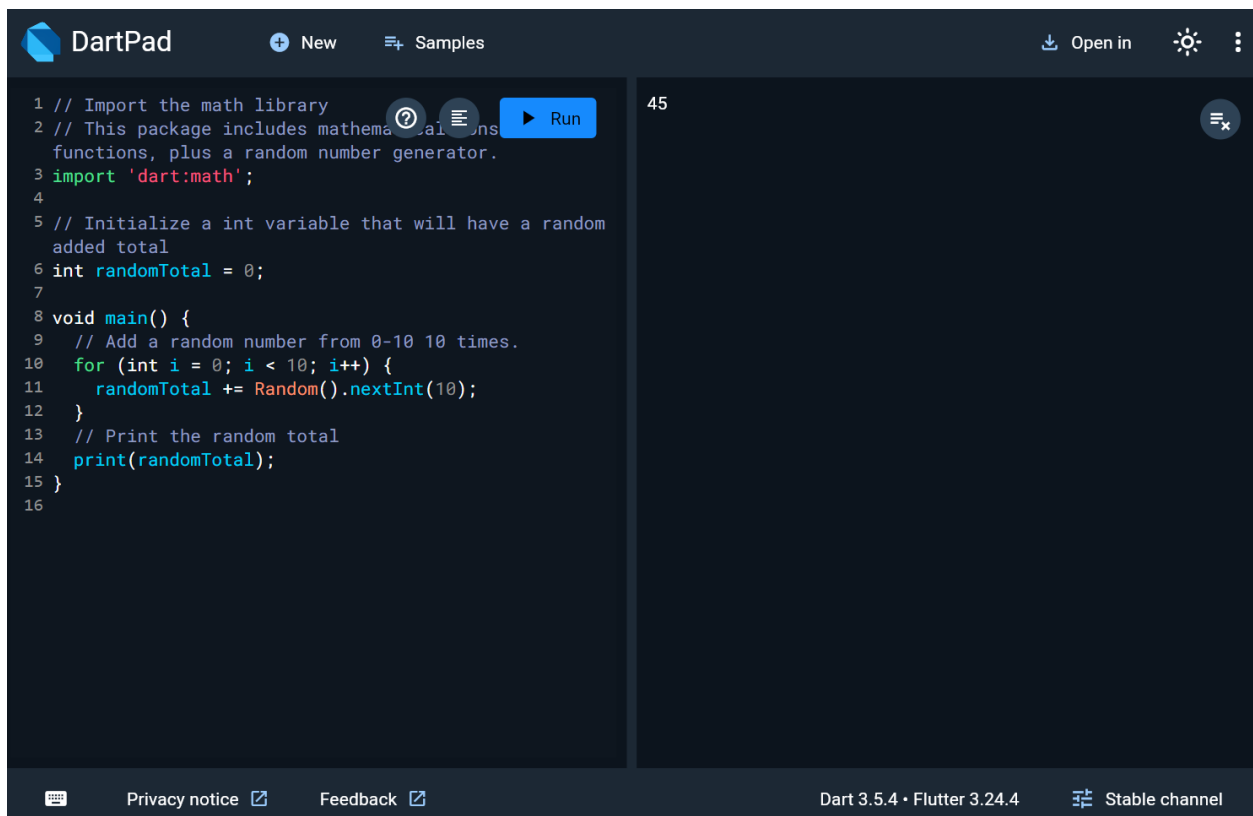
What is code reuse and why is it important?. OpsLevel. (n.d.).

<https://www.opslevel.com/resources/what-is-code-reuse-and-why-is-it-important#:~:text=Code%20reusability%20refers%20to%20writing,codebase%20size%2C%20and%20improves%20maintainability.>

Examples

Example Code One

This simple code utilizes the math package from dart that includes math constants and functions, and a random number generator. This code uses the random number generator to add to a total that would be completely random. This piece of code is useful for game applications that want to use chance as a feature of the game. On the right of screenshot, we can see the result of running the code and the display of the random number.



The screenshot shows the DartPad web interface. The left pane contains Dart code that imports the 'dart:math' package, initializes a 'randomTotal' variable to 0, and then uses a 'for' loop to add 10 random integers (generated by 'Random().nextInt(10)') to the total. The right pane displays the output '45'. The bottom status bar indicates 'Dart 3.5.4 • Flutter 3.24.4' and 'Stable channel'.

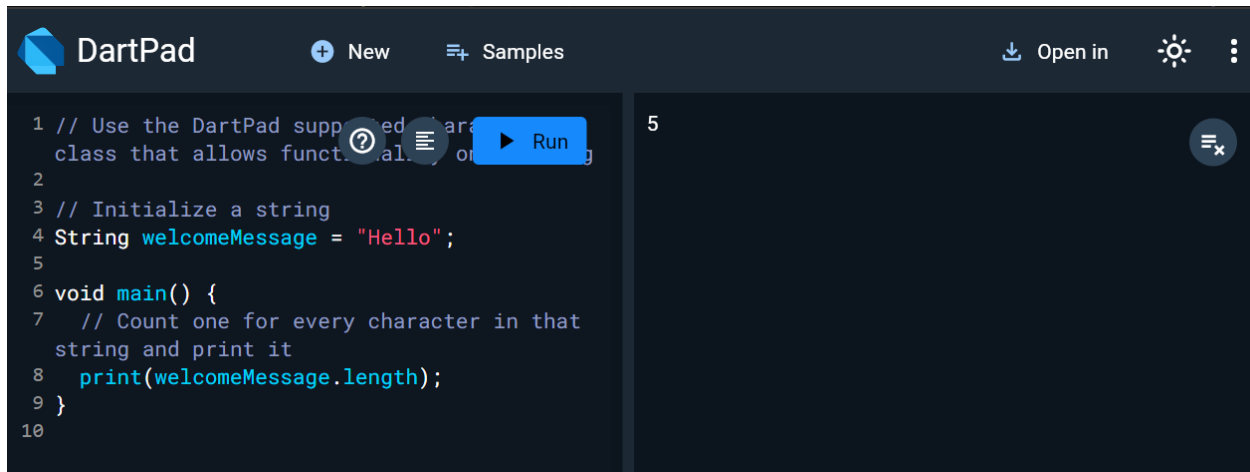
```
1 // Import the math library
2 // This package includes mathematical constants,
  functions, plus a random number generator.
3 import 'dart:math';
4
5 // Initialize a int variable that will have a random
  added total
6 int randomTotal = 0;
7
8 void main() {
9   // Add a random number from 0-10 10 times.
10  for (int i = 0; i < 10; i++) {
11    randomTotal += Random().nextInt(10);
12  }
13  // Print the random total
14  print(randomTotal);
15 }
16
```

45

Privacy notice Feedback Dart 3.5.4 • Flutter 3.24.4 Stable channel

Example Code Two

The library used in this code snippet is the character package included in the core library of dart. This snippet code uses the length function that iterates over each character in a string. For each character, it counts one to a total and returns a number. This snippet code prints the length of the string after the code is ran. This snippet of code is particularly useful in cases that utilize the length of the string. For example, if a user wants to send a message to another user but the message cannot exceed a certain limit, the length function can be used to check the length of the message and prevent the user from sending that too long of a message.



The screenshot shows the DartPad web interface. On the left, a code editor displays the following Dart code:

```
1 // Use the DartPad supported class that allows functional programming
2
3 // Initialize a string
4 String welcomeMessage = "Hello";
5
6 void main() {
7   // Count one for every character in that string and print it
8   print(welcomeMessage.length);
9 }
10
```

On the right, the output area shows the number 5, which is the length of the string "Hello". The interface includes a "Run" button and a "New" button.

Real-World Scenario

Flutter, like Dart, has Flutter packages such as a package of widgets. These widgets are a collection of ways you can view different kinds of information with visually appealing components. At the back end of a given project, you can use Dart packages to process information before it is viewed on these widgets with ease. For example, you can use an API package for Dart that gets the value of stored credits information and view it on the money widget shown at the top right of the following screenshot. Similar functions from different packages can be used for every widget shown below.

