

Title: Understanding Dart: Collections.

Student Name: Talent Nyota

Course Code: INFT 3101

Institution Name: Durham College

Date: 29/10/2024

Logo: The logo for Durham College features a green shield-like emblem on the left, composed of four squares arranged in a 2x2 grid. To the right of the emblem, the words "DURHAM" and "COLLEGE" are stacked vertically in a bold, green, sans-serif font. Below "COLLEGE", the phrase "SUCCESS MATTERS" is written in a smaller, green, sans-serif font, separated by a thin horizontal line.

Table of Contents

3. Introduction

3. Collections in Dart

- Definition
- Usage
- Comparisons
- Advantages and Limitations

4. Code Examples

- List Example
- Map Example

5. Real-World Scenario

6. References

Introduction

Dart is a contemporary programming language that is highly effective for both client and server-side development. It is particularly notable in the realm of mobile app development when used in conjunction with the Flutter framework. Collections in Dart serve as robust tools, enabling developers to efficiently organize and manipulate groups of data.

Collections in Dart

In Dart, collections are specialized classes designed to store multiple items. There are three primary types of collections: Lists, Maps, and Sets, each offering unique methods for managing data.

Usage:

- Lists: These are ordered collections of items that can be accessed by their indices.
- Maps: These consist of key-value pairs and are ideal for associating related items.
- Sets: These are collections of unique items, where each item appears only once.

Comparisons:

- Lists in Dart are like arrays in other languages like JavaScript, but they offer a more comprehensive array of methods, making them more versatile.
- Maps in Dart function similarly to objects in JavaScript or dictionaries in Python.
- Sets in Dart provide a uniqueness guarantee, like certain data structures in Java.

Advantages and Limitations:

Advantages:

- Offers flexibility in handling data.
- Provides a rich API that supports a variety of operations such as sorting, filtering, and searching.
- Ensures efficient management of memory.

Limitations:

- Lists and Maps may consume a significant amount of memory if not managed carefully.
- Sets do not preserve the order of items as they were inserted, which may be important for some applications.

Code Examples

This example uses a Map to track and manage student grades, showcasing how to utilize key-value pairs for efficient data handling:

```

bin > % dart_application_1.dart > ...
1 // Example 1: Using a List to Manage a Dynamic Task List
2 // This example will demonstrate how to use a List in Dart to manage a dynamic task list where tasks can be added and marked as done.
3 void main() {
4   List<String> tasks = [
5     "Complete Dart assignment",
6     "Attend team meeting",
7     "Read Dart documentation"
8   ];
9
10  // Adding a new task
11  tasks.add("Plan project milestones");
12  print("Current Tasks: $tasks");
13
14  // Marking the first task as done
15  String doneTask = tasks.removeAt(0);
16  print("Completed Task: $doneTask");
17  print("Remaining Tasks: $tasks");
18
19  // Example 2: Using a Map to Track Student Grades
20  // This example uses a Map to track student grades on a course, demonstrating how keys and values work in Dart maps.
21  Map<String, double> studentGrades = {
22    "Alice": 85.5,
23    "Bob": 92.0,
24    "Charlie": 88.0
25  };
26
27  // Adding a new student's grade
28  studentGrades["Diana"] = 91.5;
29  print("Student Grades: $studentGrades");
30
31  // Updating a grade
32  studentGrades["Alice"] = 87.0;
33  print("Updated Alice's Grades: $studentGrades");
34
35  // Removing a student
36  studentGrades.remove("Bob");
37  print("Grades after removing Bob: $studentGrades");
38
39

```

```

bin > % dart_application_1.dart > ...
1 // Example 1: Using a List to Manage a Dynamic Task List
2 // This example will demonstrate how to use a List in Dart to manage a dynamic task list where tasks can be added and marked as done.
3 void main() {
4   List<String> tasks = [
5     "Complete Dart assignment",
6     "Attend team meeting",
7     "Read Dart documentation"
8   ];
9
10  // Adding a new task
11  tasks.add("Plan project milestones");
12  print("Current Tasks: $tasks");
13
14  // Marking the first task as done
15  String doneTask = tasks.removeAt(0);
16  print("Completed Task: $doneTask");
17  print("Remaining Tasks: $tasks");
18
19  // Example 2: Using a Map to Track Student Grades
20  // This example uses a Map to track student grades on a course, demonstrating how keys and values work in Dart maps.
21  Map<String, double> studentGrades = {
22    "Alice": 85.5,
23    "Bob": 92.0,
24    "Charlie": 88.0
25  };
26
27  // Adding a new student's grade
28  studentGrades["Diana"] = 91.5;
29  print("Student Grades: $studentGrades");
30
31  // Updating a grade
32  studentGrades["Alice"] = 87.0;
33  print("Updated Alice's Grades: $studentGrades");
34
35  // Removing a student
36  studentGrades.remove("Bob");
37  print("Grades after removing Bob: $studentGrades");
38
39

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ZONE EXPLORER ZONE CONSOLE

```

home123@chris-MBP:~$ dart run bin/dart_application_1.dart
The Dart CLI developer tool uses Google Analytics to report usage and diagnostic data along with package dependencies, and crash reporting to send basic crash reports. This data is used to help improve the Dart platform, Flutter framework, and related tools.

Telemetry is not sent on the very first run. To disable reporting of telemetry, run this terminal command:

  dart --disable-analytics

If you opt out of telemetry, an opt-out event will be sent, and then no further information will be sent. This data is collected in accordance with the Google Privacy Policy (https://policies.google.com/privacy).

Current Tasks: [Complete Dart assignment, Attend team meeting, Read Dart documentation, Plan project milestones]
Remaining Tasks: [Attend team meeting, Read Dart documentation, Plan project milestones]
Student Grades: {Alice: 85.5, Bob: 92.0, Charlie: 88.0, Diana: 91.5}
Updated Alice's Grades: {Alice: 87.0, Bob: 92.0, Charlie: 88.0, Diana: 91.5}
Grades after removing Bob: {Alice: 87.0, Charlie: 88.0, Diana: 91.5}
home123@chris-MBP:~$ dart run bin/dart_application_1.dart

```

Real-World Scenario

A mobile application where users can personalize their settings for notifications, themes, and privacy options. By utilizing a Map, each preference can be stored as a key-value pair. This structure allows for straightforward updates and easy retrieval of each setting, enhancing the user's experience by enabling quick and efficient customization.

5. References

- Official Dart Documentation: Dart Collections, <https://dart.dev/>
- Flutter & Dart - The Complete Guide [Udemy Course]
- "Learning Dart - Second Edition" by Ivo Balbaert, Dzenan Ridjanovic

