TIERS LIMITED SUMMER INTERNSHIP 2024

MOBILE APP DEVELOPMENT

DART LANUAGE

CLASS # 2

TIERS
Limited

Overview

In this class, we will dive into the fundamentals of the Dart programming language, which is essential for Flutter development. We will cover variables, data types, control flow, functions, error handling, classes, and practice some basic Dart exercises to reinforce these concepts.

Objectives

- Understand the basic syntax and structure of Dart.
- Learn how to declare and use variables and data types.
- Explore control flow statements like if-else and loops.
- Write and use functions.
- Learn about classes and object-oriented programming in Dart.
- Complete basic exercises to practice these concepts.

Variables and Data Types

1. Variables

- Variables are used to store data. In Dart, you can declare variables using `var`, `final`, or `const`.
- Example:

```
var name = 'John';
final age = 25;
const PI = 3.14;
```

2. Data Types

- Dart supports various data types including:
  - **Numbers:** `int`, `double`
  - **Strings:** `String`
  - **Booleans:** `bool`
  - **Lists:** `List`
  - **Maps:** `Map`
- Example:

```
int score = 90;
double percentage = 93.5;
String message = 'Hello, Dart!';
bool isValid = true;
List<String> fruits = ['Apple', 'Banana', 'Mango'];
Map<String, int> ages = {'John': 30, 'Jane': 25};
```

Control Flow

1. If-Else Statements

- Used to execute code based on conditions.
- Example:

```
int number = 10;
if (number > 0) {
  print('Positive number');
} else if (number < 0) {
  print('Negative number');
} else {
  print('Zero');
}
```

2. Loops

- **For Loop:** Used to iterate over a range or collection.

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

- **While Loop:** Repeats a block of code while a condition is true.

```
int count = 0;
while (count < 5) {
  print(count);
  count++;
}
```

Functions

1. Defining Functions

- Functions are blocks of code that perform specific tasks.
- Example:

```
void greet(String name) {
  print('Hello, $name');
}

greet('John'); // Output: Hello, John
```

2. Returning Values

- Functions can return values using the `return` statement.

- Example:

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

int result = add(5, 3); // result is 8
```

Classes in Dart

1. Defining Classes

- A class is a blueprint for creating objects. A class encapsulates data for the object and methods to manipulate that data.
- Example:

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

  // Constructor
  Person(this.name, this.age);

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

void main() {
  Person person = Person('Alice', 30);
  person.introduce(); // Output: Hello, my name is Alice and I am 30 years
old.
}
```

2. Inheritance

- Inheritance is a way to create a new class from an existing class.
- Example:

```
class Animal {
  void sound() {
    print('Animal makes a sound');
  }
}

class Dog extends Animal {
  @override
  void sound() {
    print('Dog barks');
  }
}
```

```
void main() {
  Dog dog = Dog();
  dog.sound(); // Output: Dog barks
}
```

3. Getters and Setters

- Getters and setters are special methods that provide read and write access to an object's properties.
- Private variables in Dart are declared by prefixing the variable name with an underscore _. These variables are only accessible within the library in which they are declared.
- Example:

```
class Rectangle {
  double _width;
  double _height;

  Rectangle(this._width, this._height);

  // Getter
  double get area => _width * _height;

  // Setter
  set width(double value) => _width = value;
  set height(double value) => _height = value;
}

void main() {
  Rectangle rect = Rectangle(5, 10);
  print('Area: ${rect.area}'); // Output: Area: 50
  rect.width = 7;
  print('New Area: ${rect.area}'); // Output: New Area: 70
}
```

Exercises:

Exercise 1: Declare Variables

- Declare a variable for your name, age, and a list of your favorite colors.

Exercise 2: Control Flow

- Write a program that checks if a number is even or odd.

Exercise 3: Functions

- Create a function that calculates and return the area of a circle given its radius.

Exercise 4: Classes

- Create a class `Car` with properties `make`, `model`, and `year`. Add a method to display the car details. Instantiate the class and call the method.

Exercise 5: Getters and Setters

- Create a class `Student` with private properties `_name` and `_age`. Provide getters and setters for these properties and create a method to display the student's details.

Exercise 6: Inheritance

- Create a base class `Vehicle` with a method `move()`. Create a derived class `Bicycle` that overrides the `move()` method. Instantiate the derived class and call its method.

Assignment:

- Complete the exercises provided and submit your code for review.
- Read more about Dart language features from the Dart Official Documentation .