[Dart Programming Language] - Lecture [3]

[Dart Program Structure + If statement]

1. Introduction

Before we start writing code, we ***must*** understand what is the structure of a program in dart, we need to understand what are we doing before we start doing it. The program in dart is divided into multiple section and writing the appropriate code in each section is necessary for our code to be readable, clear and understood by other programmers.

In this lecture we will cover the sections of any dart program. Then we will start going through the basics of writing code and the most common instructions and their use cases like if statements and loops (next lecture).

Learning Objectives

1. Understand the structure of a dart program.
2. Be able to write dart code following this structure.
3. To have knowledge about what is the purpose of ***if*** statement.
4. Developing the skill to spot when to use ***if*** statements.

2. Context

Technical Background

Writing code in a standard way (following the theoretical rules) is crucial to be a successful programmer. When publishing code into the internet it is not easy for others to understand your code, and writing a messy, unstructured code will make this much worse, ***so we must follow these rules.***

When it comes to if statements and loops, the majority of our code throughout this course will use these tools, so mastering them and when to use them, and perfecting them, is one of the main goals of this course. Becoming a master in these two concepts will enhance your problem solving greatly.

Key Terms and Definitions:

1. Program: code that performs a specific task.
2. Structure: the organization and arrangement of the code in a specific way.
3. Condition: a statement that is either true or false

4. Function: a block of code that performs a specific task (it is much more than this, but this is enough for now).
5. Library: a ready to use set of code.
6. Syntax: the way we write code.
7. Terminal: the interface where users can interact with a computer system using commands, it is usually called CMD (command)

3. Main Content

1. Program Structure

Let's take a look at the first section of each dart program.

1. ***Imports***: Imports are used to bring in ***libraries*** and ***packages*** that your Dart program depends on. These can be Dart's core libraries, third-party packages, or even other Dart files in your project.
2. ***Global variables***: variables that is accessed everywhere on our code.
3. ***Main Function***: The ***main*** function is the entry point of every Dart program. This function is where the execution starts.
4. ***Comments:*** Comments are used to add notes or explanations within the code. They are ignored when we run the program. They can be used anywhere.
5. ***Other functions and classes:*** this will be studied later.

These are the sections of every dart program, the only required section is the main function. Here is a sample program with these sections.

```
// Import statements
 import 'dart:math';

// Global variables (optional)
const String appName = 'My Dart App';
// Main function - Entry point of the program
 void main() {
 print("Hello World!");
}
// Additional functions and classes
```

Now, let's analyze the code as much as we can.

- **Comments**: are highlighted in yellow and the give us information on the code that we can write ourselves in any language we want, to make a comment we just add // in the beginning of the line.
- **Import 'dart:math'**: before we take a look at import, we should realize that when we start writing code, we are using the default built-in library of dart. This library contains the basic instructions we use, however there are situations where we need to use instructions that do not exist in this library, so we can use other built-in (or packages) libraries using the import statement. In this example we imported the math library which includes a more advanced instructions regarding math.
- ***(;)***: the semicolon is a very famous symbol used to indicate the end of an instruction, we must use it or the program will not work.
- ***Const string appName***: this a global variable the can be used everywhere in this program, and it has the value "My Dart App"
- ***void***: this is a keyword that indicates the absence of a ***data type***.
- ***main():*** this is name of the function that all of our program is written inside, the program execution starts from here. And it finishes when the execution of main is finished.
- ***{ } :*** these are called brackets and are used to indicate the beginning and the end of a block of code, functions (like main) should always followed by **{ }**, the opening bracket. (**{**) indicated the beginning of the main block, and the closing bracket (**}**) indicates the end of the main block, so whatever is outside these two brackets ***does not belong to main.***
- ***print("Hello World!"):*** print is a function that is defined in the dart built-in library. It is used to show its input into the terminal. We notice that the input in this case is "Hello World!
- **Additional functions and classes:** this will be explored in later lectures**.**

From now on we will start writing our code inside the main function, now let's look at the if statement.

2. If statement:

The ***if*** statement is a fundamental control flow structure in Dart, as it is in many other programming languages. It allows us to make decisions in our code, executing certain blocks of code only when specific conditions are met.

First let's take a look at the structure of ***if*** (syntax):

```
if (condition)
 {
// Code to execute if the condition is true
}
```

The is the most basic form of if, it has a ***condition***, a condition is a statement that is either true or false, and there are multiple ways we can write conditions, we will learn them with time throughout this course. If has two cases, one is when the condition is true and the other is when the condition is false. If it is true, the block of code inside its brackets will be executed, if it is false, the block will be ignored.

Let's take a look at another form of if:

```
if (condition)
 {
// Code to execute if the condition is true
}
else {
// Code to execute if the condition is false
}
```

Now we notice adding a new instruction called ***else, this keyword is only used after if.*** Now if the condition is true what is inside if well be executed and if the condition is false, what is inside the else will be executed.

Let's is look at a new form:

```
if (condition1)
{
// Code to execute if condition1 is true
}
else if (condition2)
{
// Code to execute if condition2 is true
 }
else
 {
// Code to execute if all previous conditions are false
 }
```

Using if – else if is very useful when we have multiple conditions we have to evaluate.

4. Practical Examples

Example 1: [Check if a String is Empty]

Use if else statement with string attributes to check if a string is empty or not, print a message to show the result.

Example 2: [Check if a String Starts with a Specific Prefix]

Use if else if statements to check if a string (choose this string as you want) starts with (@) or with (#), print a message to show the result. Note, search for a behavior that checks the beginning of a text.

Example 3: [Check if a number is odd or even]

Use if else statement to check if a number (choose this number as you want) is odd or even. Print a message to show the result. Use integer attributes.

Example 4: [Check if a number is positive or negative]

Use if else if statements to check if a number (choose this number as you want) is positive, negative or zero. Print a message to show the result. Use integer attributes.

5. Homework/Assignments

Do the practical examples.

Due Date: 29/8/2024