Scala Tutorial Pt. 1

Scala is both an object-oriented and functional programming language. It is based on the Java Virtual Machine (JVM) and supports seamless integration with Java. Visit the Getting Started page to correctly install Scala on your computer, based on the operating system you have.

Once the installation is complete, run the following command in the terminal to confirm whether Scala has been installed correctly or not.

```
$ scala --version
Scala code runner version 3.3.0 -- Copyright 2002-2023, LAMP/EPFL
```

We're ready to go further!

Create a new file named `hello.scala` with the extension `.scala`, as it is the file extension used for Scala programs. We want to print the text *hello, world* on the terminal. To do this, let's write the *hello, world* text within double quotes, as strings in Scala are enclosed in double quotes. Single quotes are reserved for characters.

```
"hello, world"
```

`print()` method is used to print something on terminal.

```
print("hello, world")
```

In Scala, the execution of a program starts from the `main()` method, similar to other programming languages. However, defining the `main()` method in Scala is slightly different. Instead of a conventional `main()` method, you can define *any* method you prefer and annotate it as the entry point using the `@main` annotation.

To define a method in Scala, `def` keyword is used. Let's define the `hello()` method and put this `print()` statement within this `hello()` method.

```
def hello() = print("hello, world")
```

In Scala, the `=` sign is used to separate the method definition from the method body. Everything after the `=` sign represents the body of the `hello()` method.

Mark this as *main* using `@main` annotation.

```
@main def hello() = print("hello, world")
```

Here we have the *hello, world* program in Scala!

In Scala, the execution of the `hello.scala` program starts from the `hello()` method, which is designated as the main method for Scala.

In Scala, the execution of a program occurs in two steps. In the first step, we compile the program using the Scala compiler (`scalac`). After successful

compilation, the Scala compiler generates the `hello.class` file (along with a few other files). In the second step, the created class is interpreted by the Scala interpreter (`scala`).

Following is the command for the step one.

```
$ scalac hello.scala
```

Following is the command for the step two.

```
$ scala hello
hello, world
```

We just wrote and run the *hello, world* program in Scala!

Instead of using the `print()` method, you can use the `println()` method in Scala to format the output nicely. This method appends the `\n` character at the end of the printed text, creating a new line.

```
@main def hello() = println("hello, world")
```

Run the program again with above two commands and confirm the output.

In Scala, whitespace does not affect the code's functionality. The above code can be written in different ways, and the output will remain the same. For example,

```
@main def hello() =
  println("hello, world")
```

When your method contains more than one statement in its body, it is recommended to format the code in this way to achieve readability.

Run the program again and confirm the output. You should see the same identical *hello, world* text as the output.

You can rename the `hello()` method to `main()` (or any other name) in Scala, and there won't be any issues with this change. Scala considers it as the *main* method. The only difference is that after compilation, `scalac` will generate `main.class` (along with a few other files) instead of `hello.class`.

```
@main def main() =
  println("hello, world")
```

Following are the commands we need to run in order to see the output after this new change.

```
$ scalac hello.scala
```

```
$ scala main
hello, world
```