What is Java?

Java is a popular programming language, created in 1995.

It is owned by Oracle, and more than 3 billion devices run Java.

It is used for:

Mobile applications (specially Android apps)
Desktop applications
Web applications
Web servers and application servers
Games
Database connection
And much, much more!

Why Use Java?

- Java works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.)
- It is one of the most popular programming language in the world
- It has a large demand in the current job market
- It is easy to learn and simple to use
- It is open-source and free
- It is secure, fast and powerful
- It has a huge community support (tens of millions of developers)
- Java is an object oriented language which gives a clear structure to programs and allows code to be reused, lowering development costs
- As Java is close to C++ and C#, it makes it easy for programmers to switch to Java or vice versa

Java Syntax

In the previous chapter, we created a Java file called Main.java, and we used the following code to print "Hello World" to the screen:

Main.java

```
public class Main {
  public static void main(String[] args) {
    System.out.println("Hello World");
  }
}
```

Example explained

Every line of code that runs in Java must be inside a class. In our example, we named the class Main. A class should always start with an uppercase first letter.

Note: Java is case-sensitive: "MyClass" and "myclass" has different meaning.

The name of the java file must match the class name. When saving the file, save it using the class name and add ".java" to the end of the filename. To run the example above on your computer, make sure that Java is properly installed: Go to the Get Started Chapter for how to install Java. The output should be:

The main Method

The main() method is required and you will see it in every Java program:

public static void main(String[] args)

Any code inside the main() method will be executed. Don't worry about the keywords before and after main. You will get to know them bit by bit while reading this tutorial.

For now, just remember that every Java program has a class name which must match the filename, and that every program must contain the main() method.

System.out.println()

Inside the main() method, we can use the println() method to print a line of text to the screen:

```
public static void main(String[] args) {
  System.out.println("Hello World");
}
```

Note: The curly braces {} marks the beginning and the end of a block of code.

System is a built-in Java class that contains useful members, such as out, which is short for "output". The println() method, short for "print line", is used to print a value to the screen (or a file).

Don't worry too much about System, out and println(). Just know that you need them together to print stuff to the screen.

You should also note that each code statement must end with a semicolon (;).

Java Output / Print

Print Text

You learned from the previous chapter that you can use the println() method to output values or print text in Java:

```
System.out.println("Hello World!");
```

You can add as many println() methods as you want. Note that it will add a new line for each method:

Example

```
System.out.println("Hello World!");
System.out.println("I am learning Java.");
System.out.println("It is awesome!");
```

Double Quotes

When you are working with text, it must be wrapped inside double quotations marks "".

If you forget the double quotes, an error occurs:

Example

```
System.out.println("This sentence will work!");
System.out.println(This sentence will produce an error);
```

The Print() Method

There is also a print() method, which is similar to println().

The only difference is that it does not insert a new line at the end of the output:

Example

```
System.out.print("Hello World! ");
System.out.print("I will print on the same line.");
```

Note that we add an extra space (after "Hello World!" in the example above), for better readability.

In this tutorial, we will only use println() as it makes it easier to read the output of code.

Java Output Numbers

Print Numbers

You can also use the println() method to print numbers.

However, unlike text, we don't put numbers inside double quotes:

```
System.out.println(3);
System.out.println(358);
System.out.println(50000);
```

You can also perform mathematical calculations inside the println() method:

System.out.println(3 + 3);

Java Comments

Java Comments

Comments can be used to explain Java code, and to make it more readable. It can also be used to prevent execution when testing alternative code.

Single-line Comments
Single-line comments start with two forward slashes (//).

Any text between // and the end of the line is ignored by Java (will not be executed).

This example uses a single-line comment before a line of code:

This example uses a single-line comment at the end of a line of code:

Example
System.out.println("Hello World"); // This is a comment

Java Multi-line Comments

Multi-line comments start with /* and ends with */.

Any text between /* and */ will be ignored by Java.

This example uses a multi-line comment (a comment block) to explain the code:

Example

```
/* The code below will print the words Hello World
to the screen, and it is amazing */
System.out.println("Hello World");
```