**What is Java technology and why do I need it?**

* **Java is a programming language and computing platform first released by Sun Microsystems in 1995.**
* **It has evolved from humble beginnings to power a large share of today’s digital world, by providing the reliable platform upon which many services and applications are built.**
* **New, innovative products and digital services designed for the future continue to rely on Java, as well**
* **While most modern Java applications combine the Java runtime and application together.**
* **Coding Standards :**

[**https://www2.hawaii.edu/~walbritt/ics211/materials/standard.htm**](https://www2.hawaii.edu/~walbritt/ics211/materials/standard.htm)

* **Java Programming Rules :**
* **Classes begin with a capital letter.**
* **Packages are all lower case.**
* **Methods begin with a lower case letter.**
* **Multi-word identifiers are internally capitalized in methods (CamelCase).**

# **Java Syntax:**

## What is Java Syntax?

Java Syntax is a basic of the language, all the main rules, commands, constructions to write programs that the compiler and computer “understands”. Every programming language has its syntax as well as human language.

**Class in Java**

**Class** is a model or template or blueprint of the object. It describes the behavior and states that the object of its type supports. For example, the Cat class has its name, color, owner; cat also has behavior such as eating, purring, walking, sleeping.

**Methods in Java**

Methods are for describing the logics, manipulating data and executing all the actions. Every method defines behavior. A class can contain many methods. For example we can write a method sleep() for Cat class (to sleep) or purr() to purr.

**Instance Variables in Java**

Every object has a unique set of instance variables. Object state is usually generated by the values assigned to these instance variables. For example cat’s name or age could be a variable. We are going to start with the simplest Java program. Using this example, we will understand the basic concepts of Java syntax, and then take a closer look at them.

## Simple Java program: Hello, Java!

**class** HelloJava {

**public** **static** **void** main(String[] args) {

System.out.println("Hello, Java!");

}

}

**This program prints out a string “Hello, Java!” to console. I recommend you to install JDK and IntelliJ IDEA and try to write out the code you see above. Or for the first try find an online IDE to do the same.**

**class HelloJava**

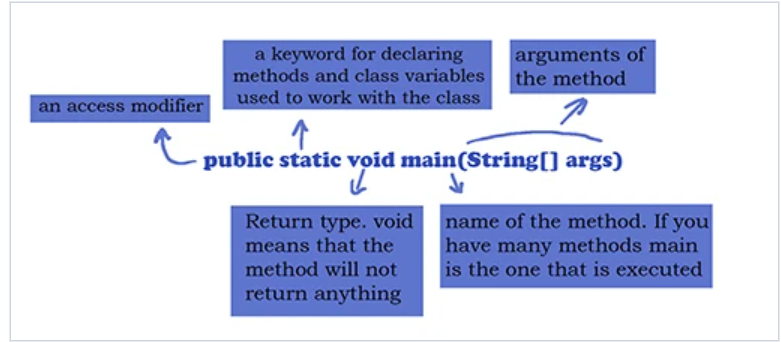
**Each program in Java is a class or more often many classes. The line class HelloJava means that here we create a new class and its name is HelloJava. As we defined above, class is a kind of template or blueprint, it describes the behavior and states of the class’ objects. It could be tough for novice programmers, you’ll learn this concept a little bit later. For now class HelloJava is just the beginning of your program.**

**You may have noticed the curly brace { on the same line and throughout the text. A pair of curly braces {} denotes a block, a group of programming statements that is treated as one single unit. Where { means the beginning of the unit and } its ending. Blocks can be nested within each other, or they can be sequential.**

**There are two nested blocks in the above program. The external one contains the body of the class Hello. The inner block contains the body of the main() method.**

**public static void main (String args []) {**

**Here is the beginning of the main method. A method is a behaviour, or the sequence of commands that allows you to perform an operation in a program. For example multiply 2 numbers or print out a string. In other words, a method is a function. In some other programming languages, methods are often referred to as "functions". Methods, just like all elements of a Java program, are located within a class. Each class can have one, many, or no methods.**

****

**public is an access modifier. A variable, method, or class marked with the public modifier can be accessed from anywhere in the program. There are four of them in Java: public, private, protected and default (empty). We talk about them a little bit later. For the first step it is better to make all your methods public.**

**void is the return type of the method. Void means that it doesn't return any value.**

**main represents the starting point of the program. This is the name of the method.**

**String[] args is a main method argument. For now it is enough to know that almost every Java program has the main method, it starts the program and it declares such as public static void main(String[] args)**

**Static methods are ones to work with the class. Methods that use the static keyword in their declaration can only work directly with local and static variables.**

**System.out.println("Hello, Java!");**

**Formally this line executes the println method of the out object. The out object is declared in the OutputStream class and statically initialized in the System class. However it’s a bit complicated for a total newbie. It is enough for a beginner to know that this line prints words "Hello, Java!" to the console.**

## **Java basic syntax rules**

**There are some main syntax rules to follow when programming in Java:**

* **the file name must be identical to the class name;**
* **most often each class is in a separate file with a .java extension. Class files are usually grouped into folders. These folders are called packages;**
* **characters are case sensitive. String is not equal to string;**
* **The beginning of Java program processing always starts in the main method: public static void main (String [] args). The main () method is a required part of any Java program;**
* **Method (procedure, function) is a sequence of commands. Methods define the behavior of on object;**
* **The order of the methods in the program file is irrelevant;**
* **Keep in mind that the first letter of a class name is in uppercase. If you are using multiple words, use uppercase for the first letter of each word (“MyFirstJavaClass”);**
* **the names of all methods in Java syntax begin with a lowercase letter. When using multiple words, the subsequent letters are capitalized ("public void myFirstMethodName ()");**
* **files are saved with class name and .java extension ("MyFirstJavaClass.java");**
* **In Java syntax, there are delimiters "{...}" that denote a block of code and a new area of ​​code;**
* **Each code statement must end with a semicolon.**

**Java variables and data typesVariables are special entities used to store data. Any data. In Java, all data is stored in variables. You may say a variable is a reserved place or a box to put variable in. Every variable has its data type, name (identifier) and value. Data types can be primitive and non-primitive or reference. Primitive data types could be:**

* **Integers: byte, short, int, long**
* **Fractionals: float and double**
* **Logical values: boolean**
* **Symbolic values (for representing letters and numerals): char**

#### **Java variables example:**

**int s;**

**s = 5;**

**char myChar = ‘a’;**

**n this code we created an integer variable s (an empty container) and then put a value 5 in it.**

**The same story with a variable named myChar. We created it with a char data type and defined it as a letter a. In this case we created a variable and simultaneously assigned a value into it. Java syntax lets you do it this way.**

**Reference types are some objects that keep references to values or other objects. They can also contain reference to the null. Null is a special value to denote the absence of value.**

**Among reference types are String, Arrays and every Class you want. If you have a Violin class, you can create a variable of this Class.**

**Among reference types are String, Arrays and every Class you want. If you have a Violin class, you can create a variable of this Class.**

**Java reference type variables example:**

**String s = “my words”;**

**Violin myViolin;**

**You will learn more about them later. Remember that non-primitive types of variables start from capital letters while primitive — from lowercase letters.**

**int i = 25;**

**String s = “Hello, Java!”;**

### Java Arrays

**Arrays are objects that store multiple variables of the same type. However, an array itself is an object on the heap. We will look into how to declare, construct, and initialize in the upcoming chapters.**

**Array example:**

**int[] myArray = {1,7,5};**

**Here we have an array that contains from the three integers (1,7 and 5)**

**Java Enums**

**In addition to primitive data types Java has such a type as enum or enumeration. Enumerations represent a collection of logically related constants. An enumeration is declared using the enum operator, followed by the name of the enumeration. Then comes a comma-separated list of enumeration elements:**

**enum DayOfWeek {**

**MONDAY,**

**TUESDAY,**

**WEDNESDAY,**

**THURSDAY,**

**FRIDAY,**

**SATURDAY,**

**SUNDAY**

**}**

**An enumeration actually represents a new type, so we can define a variable of that type and use it. Here is an example of using enumeration.**

#### **Java Enum Example**

**public class MyNum{**

**public static void main(String[] args) {**

**Day myDay = DayOfWeek.FRIDAY;**

**System.out.println(myDay); //print a day from the enum**

**}**

**}**

**enum DayOfWeek{**

**MONDAY,**

**TUESDAY,**

**WEDNESDAY,**

**THURSDAY,**

**FRIDAY,**

**SATURDAY,**

**SUNDAY**

**}**

**If you run the program, FRIDAY is printed in the console. You can put your Enum and MyNum class code in one file, but it is better to create two separate files: one for MyNum class and one for Day enum. IntelliJ IDEA lets you choose enum while creating.**

### Declaring Variables in Java

**Actually we have declared some variables above and even identified them. Declaration is a process of allocating memory for a variable of a certain type and naming it. Something like that:**

**int i;**

**boolean boo;**

**We can also declare to initialize a variable using assignment operator (=). That means we put a particular value into the memory we allocated. We can do it right in a moment of declaration or later.**

#### **declaring a variable example**

**String str;**

**int i = 5;**

**Str = “here is my string”;**

**How Java Program Works :**

