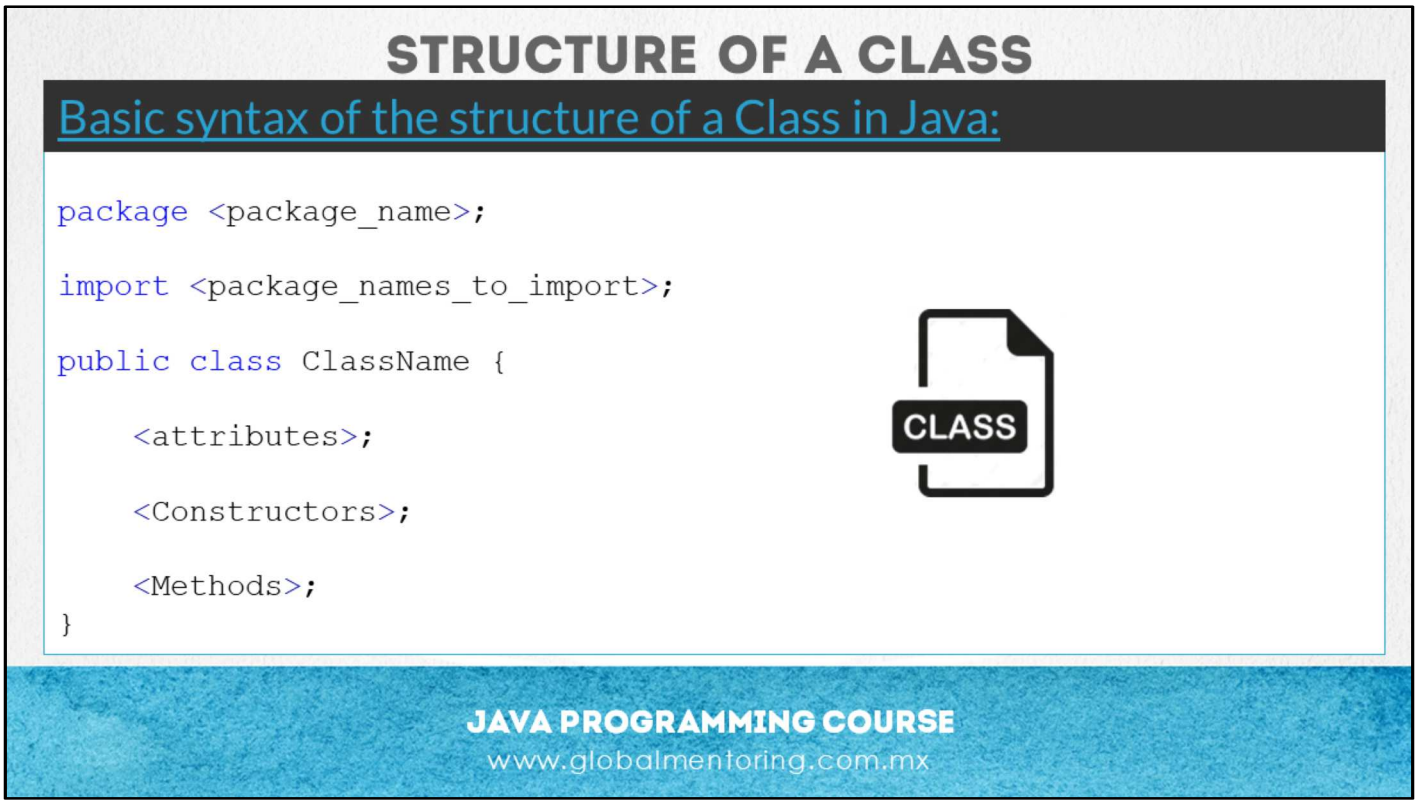
Hello, Ubaldo Acosta greets you. Welcome again to this Java Programming Course.

In this first lesson we will review the basic syntax, which we studied in detail in the previous course, this is just a brief review so that we start to warm up. We will see from the basic definition of a Class, among other examples of syntax.

So, if you're ready, so are we. Let's start immediately.



This is a brief review of the elements of a class.

First we specify the package to which the class belongs. A package is basically like a folder that groups several classes that have something in common, classes are usually grouped by their functionality.

Later, if necessary, we carry out the import of the classes that we will use. Remember that there are import of classes, or also import static which import attributes or static methods to be used directly in our code.

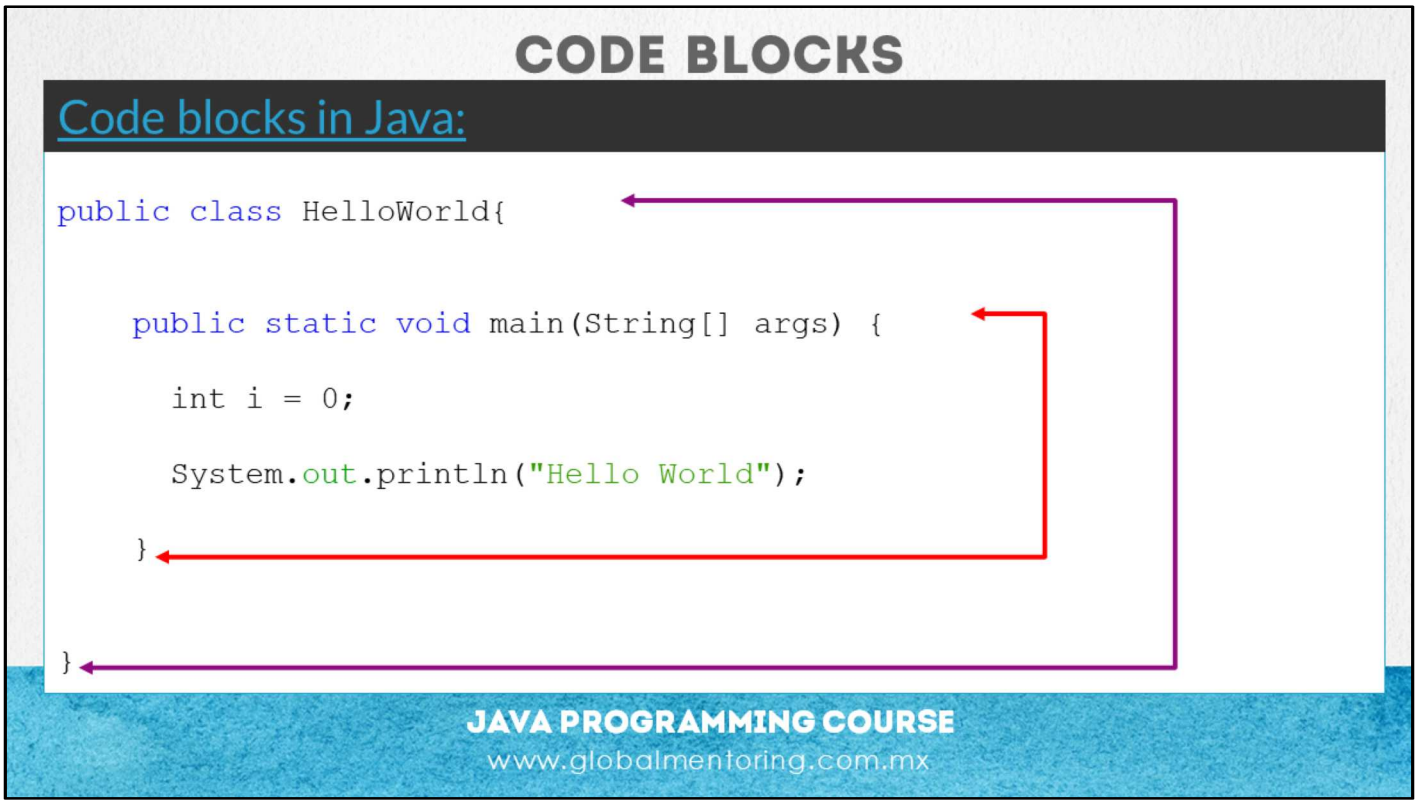
Then we indicate the name of our class. Remember that in a file can only declare a class of type public, and this should be called exactly like the name of the file that has .java extension, however there may be more classes in the same file, which are no longer public, but only classes defined within the same file.

Following the name of the class, we define the attributes of the class. The order of the elements does not affect the class, because if we remember the diagram of donating an object, we can understand that the attributes and methods can be defined anywhere within the class, but as a good practice it is recommended to do it in the order shown. The attributes can be of any type of data that we have previously studied, in addition to containing several modifiers, such as access modifiers, or static, final among several others that we will study later.

After the attributes it is recommended to declare the constructors, in the same way the order does not affect, but it is a good practice to define them after the attributes of our class. There can be several constructors, both private and public, with several arguments or without arguments. If we do not define a constructor, then the compiler automatically adds the empty constructor so that objects of the class we define can be created. If a constructor other than vacuum is defined, then the compiler no longer adds the empty constructor to our class, and one of the defined constructors must be used to create an object of the defined class.

Finally we define the methods of our class, they can exist of several types, such as simple methods of type get or set for the properties of the class, or static methods, public, private, etc. Everything depends on the functionality that we need from the defined method. This is known as the interface of the method, and they have nothing to do with the topic of interfaces that we will see later. The interface of a class has to do with the methods that we have available to be used in a class.

With this in broad strokes we have a summary of the creation of classes in Java. Remember that there are several other topics, such as the concept of inheritance and more themes, but let's leave simple the basic structure of the class to start studying other topics, and in turn we will go back to each of the topics studied in the previous course, as well as the new topics included in this course.



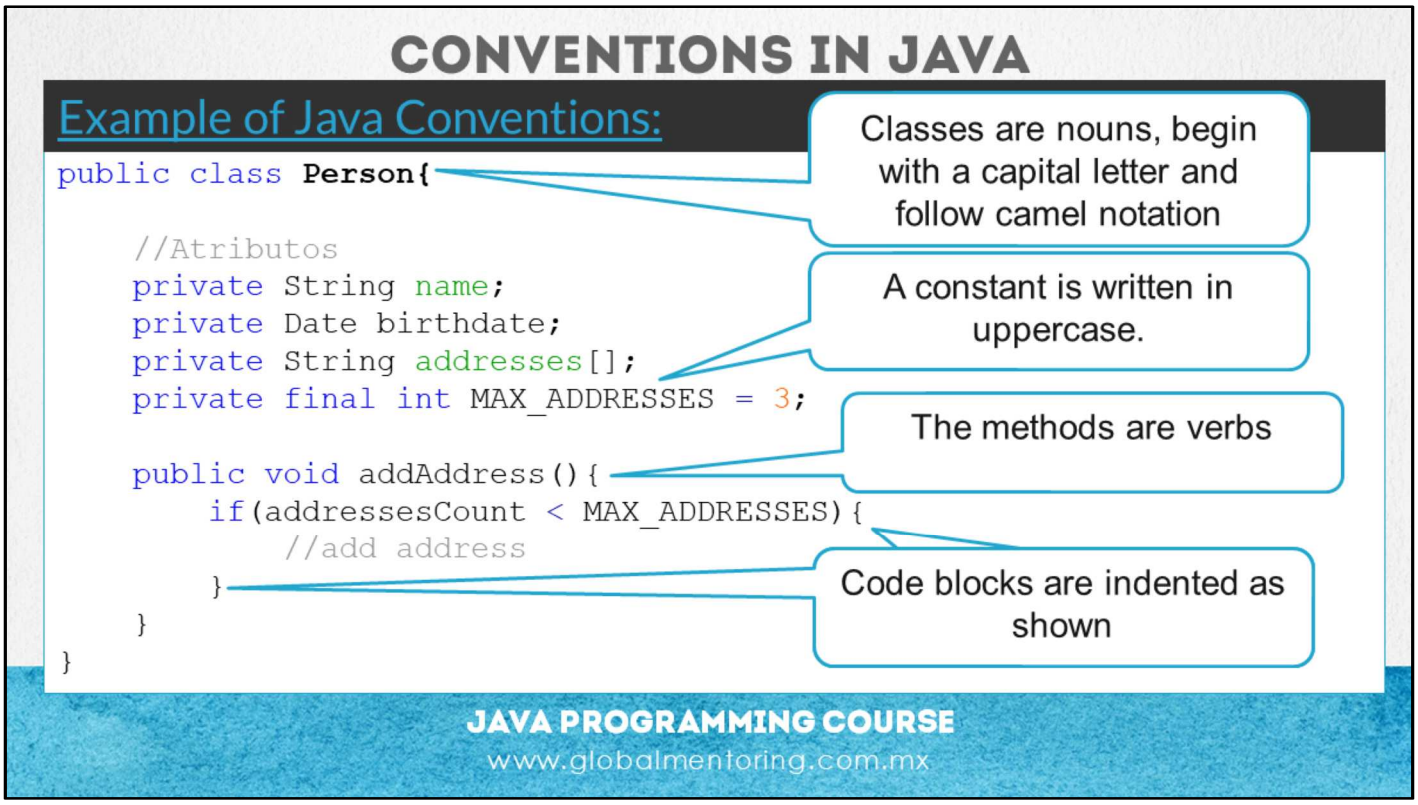
Classes, as well as methods, have a beginning and an end, the symbol used to mark the beginning and the end are the keys {}. This is known as a code block, there are even anonymous code blocks that do not have a name, this we will study later.

The variables defined within these code blocks have precisely the duration to which the code block where it is declared ends.

For example, if a variable is declared inside the main method, it will only exist during the execution of this method, and it will end when this block of code ends, that is, when the execution of the main method ends.

The Java compiler ignores any space or line, so we can use precisely this feature to make and write a more readable code, as we go we will see several good practices to write the code, since the compiler will not indicate that there is a error, but we must use these practices to make our code more readable and understandable for both us and the team working on the project in question.

This is just a brief review of the variable scope topic studied in the Java Fundamentals course.



Let's see now the subject of conventions. A convention is an agreement, so we must understand that the compiler will not throw out any errors or tell us that something needs to be corrected or changed. Therefore, we must take this as good practices and it is finally the work team that will define the conventions to be used in the code we create. However, these are some general conventions that we suggest you follow with the aim that your code is more readable and understandable for both us and our team working on the respective project.

- The classes are substantive, a noun is basically the name we give to a person or thing. It can exist physically or be something abstract, like a created concept. For that reason the name that we give to our classes will be substantive, and the name will have to follow some conventions. The first letter must be written in capital letters, and later use the camel notation. This notation means that if our class uses two names, we must join them, without separation and each word must start in capital letters, for example: CreditCard.

- The methods must be verbs, and the first letter must be written in lowercase, and later use the camel notation, for example: removeMoney(), or changeState (), etc.

- The variables should be brief but with substantial meaning, for example birthdate, firstName, etc. However it is preferable not to abbreviate their names too much so that just reading the code we can understand the use of the defined variable. Variables with a single letter must be avoided, except when they are variables that are executed in a block of small code and temporarily, such as a for loop when defining the variable i that serves as a counter for the iteration of this loop.

- The constants, which are the variables defined with the final word, must be written in uppercase and if more than one word is used, each word must be separated by a low guide.

- The indentation means the use of blank spaces or line breaks in order to make our code easier to read and therefore easier to make changes. Therefore, the methods will be devised according to the described sheet, that is, the key of the code block is opened at the end of the method, and at the end of the last line of the method the closing of the key of the respective method will be added on the next line.

For more information about the Java conventions, a document was created, which you can see in the following link:
http://icursos.net/cursos/ProgramacionJava/Leccion02/doc/convencionesJava.pdf