

**Resume of thinking in Java**

Name: Sebastian Landazuri

Date: 18/11/2018

Within a class, or a method, we can define fields or variables, respectively, that can be of simple types, or complex classes, either of the Java API, whether we have defined ourselves, or that we have copied from another place.

Like the names of the classes, it is usually convenient to use nouns that describe the meaning of the field, and may also be formed by several words. In this case, the first word will begin with a lowercase letter, and the rest will be capitalized. For example, surnames, birthdate, numIterations.

On the other hand, the constants are declared as final static, and their names are written in uppercase, separating the different words that form them by underscore characters ('\_'). For example, WINDOW\_WIDE, MSG\_ERROR\_FICH.

Both the classes and the fields and methods admit access modifiers, to indicate if said elements have public, protected or private scope. These modifiers are marked with the reserved words public, protected and private, respectively, and are placed at the beginning of the declaration

The protected modifier implies that the elements that carry it are visible from the class, its subclasses, and the other classes of the same package as the class.

If no modifier is specified, the element will be considered a packet type. This type of elements may be visible from the class or from classes in the same package, but not from the subclasses.

Each Java file that we create must have one and only one public class (which will be the main class of the file). This class must be called the same as the file. Apart, the file may have other internal classes, but may no longer be public.

The protected modifier implies that the elements that carry it are visible from the class, its subclasses, and the other classes of the same package as the class.

If no modifier is specified, the element will be considered a packet type. This type of elements may be visible from the class or from classes in the same package, but not from the subclasses.

Each Java file that we create must have one and only one public class (which will be the main class of the file). This class must be called the same as the file. Apart, the file may have other internal classes, but may no longer be public.

In addition to the access modifiers seen before, in classes, methods and / or fields these modifiers can also be used:

abstract: base element for inheritance (subtype objects must define this element). It is used to define abstract classes, and abstract methods within those classes, so that they can be implemented by subclasses that inherit from it.

static: element shared by all objects of the same class. With this modifier, a copy of the element is not created in each object that is created of the class, but all share a single copy in memory of the element, which is created without the need to create an object of the class that contains it. As it has been seen previously, it can also be applied on classes, with a different meaning in this case.

final: final object, not modifiable (used to define constants) or inheritable (in case of applying it to classes).

synchronized: for elements that can not be accessed at the same time from different threads.