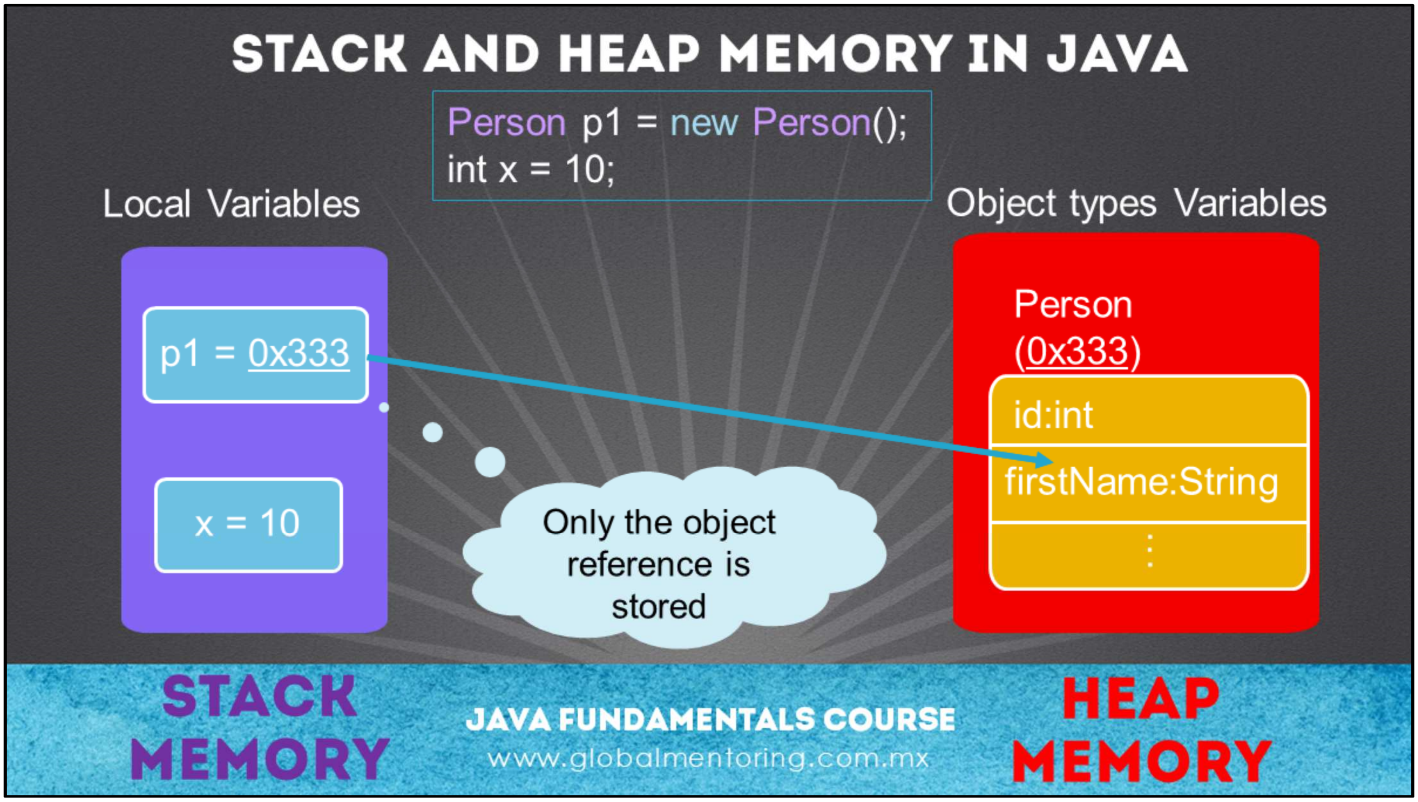
Hello, Ubaldo Acosta greets you again. I hope you're ready to start with this lesson.

We are going to study the subject of Stack and Heap Memory in Java.

Are you ready? Come on!



As in many programming languages, RAM (Random Access Memory) is used to store the information of our program while it is running. Java executes a process, and internally there are two classifications to store the values of our programs, known as Stack memory and Heap memory.

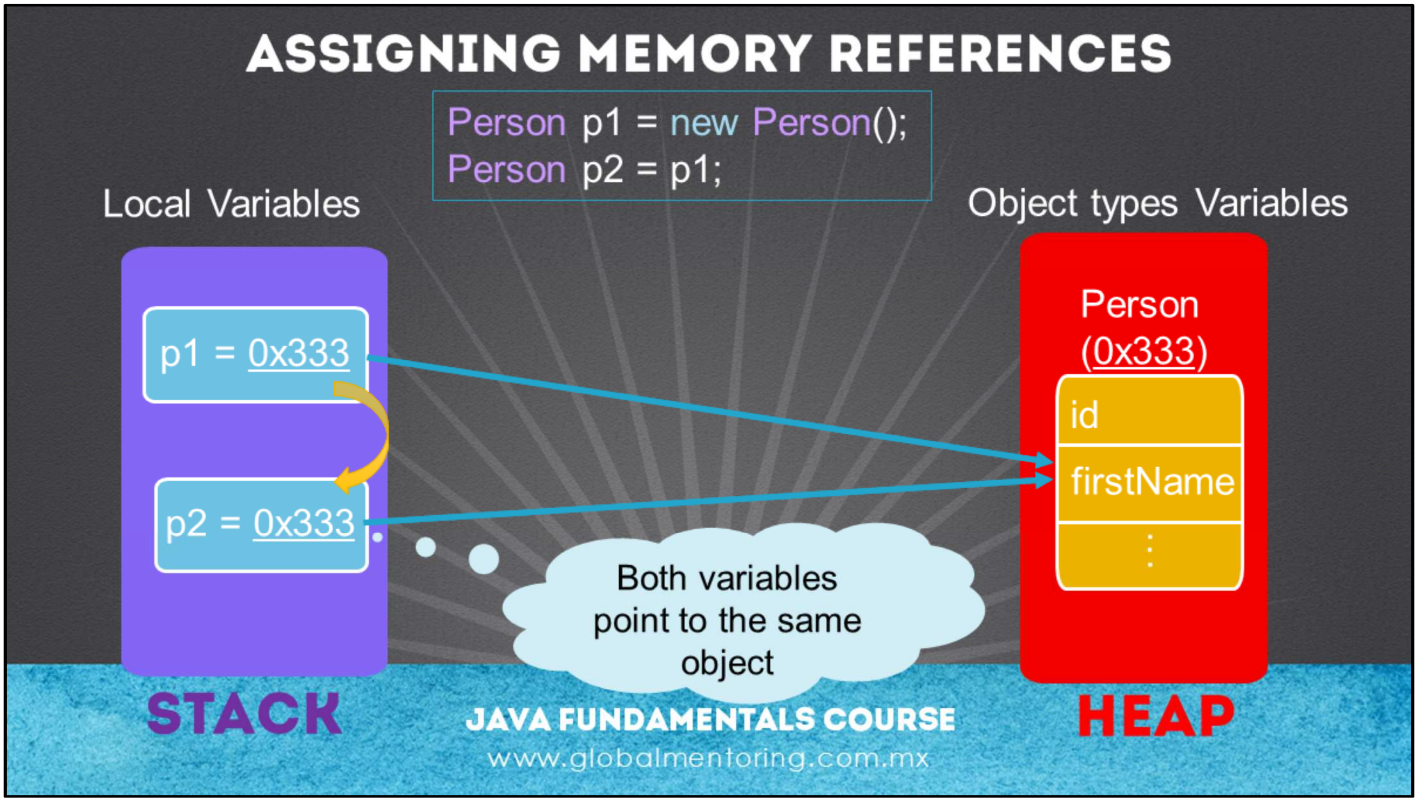
Stack memory is used to store local variables and function calls in Java. The variables stored in this memory space usually have a short life span, only while the function or method in which they are running ends.

Heap memory is used to store Java objects, including their attributes. Objects stored in this memory space usually have a longer duration time.

This means that the varibles that we create, as we can observe in the figure, do not store the object in itself, but only keep the reference of the object. In the figure we can see that the reference of the object is represented by a hexadecimal value (0x333), which contains the memory address where the object is stored, and therefore the local variable p1 stores only this memory reference.

This is very important for issues such as the collection of objects in Java, because the garbage collector can only remove objects that are not being pointed by any variable. In this way the garbage collector can search for those objects in the heap memory that are no longer being referenced by any other variable and finally free space in memory that occupied that object.

With this we can add another important distinction between class and object. A class defines a type to be later instantiated, that is, with the objective of creating objects. This means that an object occupies a real space in memory, and it is the objects that we will use in most of our programs to send and receive messages through the use of their methods.



Unlike what we might think, in Java a variable of type Object only stores a reference to the objects that are created, so the second line of the code shown, only what it does is pass the reference that is storing the variable p1 . And therefore now there are two variables that are pointing to the same object, that is, that both variables can access what the object allows according to the defined class, for example its attributes or methods that have been defined.

It is important to note that a second object is not being created, since there is only one call from the word new, and therefore there is only one created object. What the variable p2 stores is only the value of the memory reference of the created Person object. Every time we use the word new we will receive a new memory reference, it is important to note this, since until that moment a new object will be created and not before.

If we execute the following line:

p1 = null;

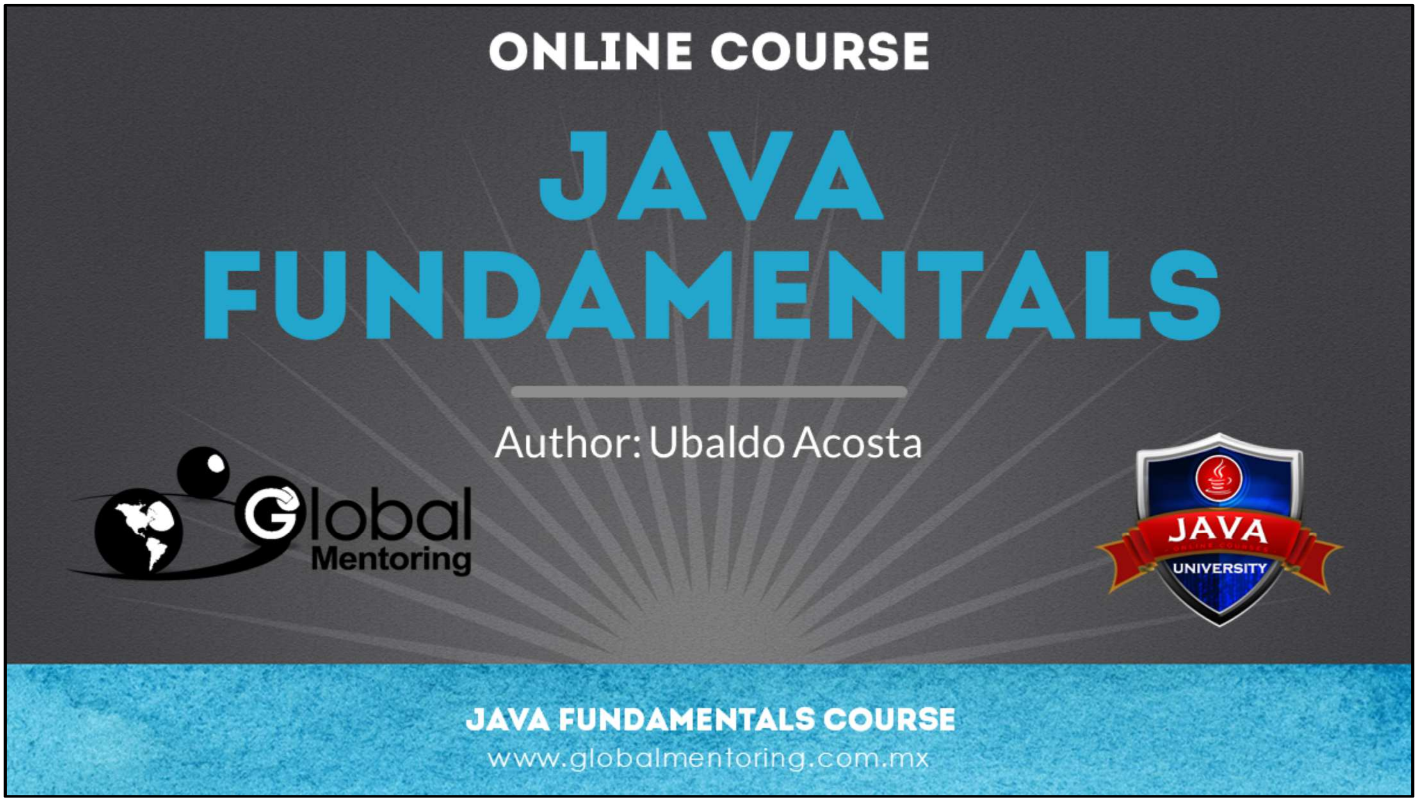
What would happen is that the variable p1 would no longer have the reference of the created Person object located in the reference 0x333, and therefore only the variable p2 is the one that could continue to access this created object.

If we finally executed:

p2 = null;

What would happen is that p2 would no longer have the object reference and therefore no variable would have the reference of the created Person object, and therefore the Java garbage collector could dispose of this object and remove it from memory. An object is a candidate to be removed from memory until no variable is pointing to the object, that is, that no variable contains its reference.

The garbage collector process is a process that demands a lot of resources, so calling System.gc () does not guarantee that the garbage collector will be executed. However, with the call to this function we are telling Java that as soon as possible, it executes the collector and therefore deletes the memory objects that are no longer being used, that is, there is no longer any variable pointing to these objects.


ONLINE COURSE
JAVA
FUNDAMENTALS
Author: Ubaldo Acosta
Global
Mentoring
JAVA
UNIVERSITY
JAVA FUNDAMENTALS COURSE
www.globalmentoring.com.mx