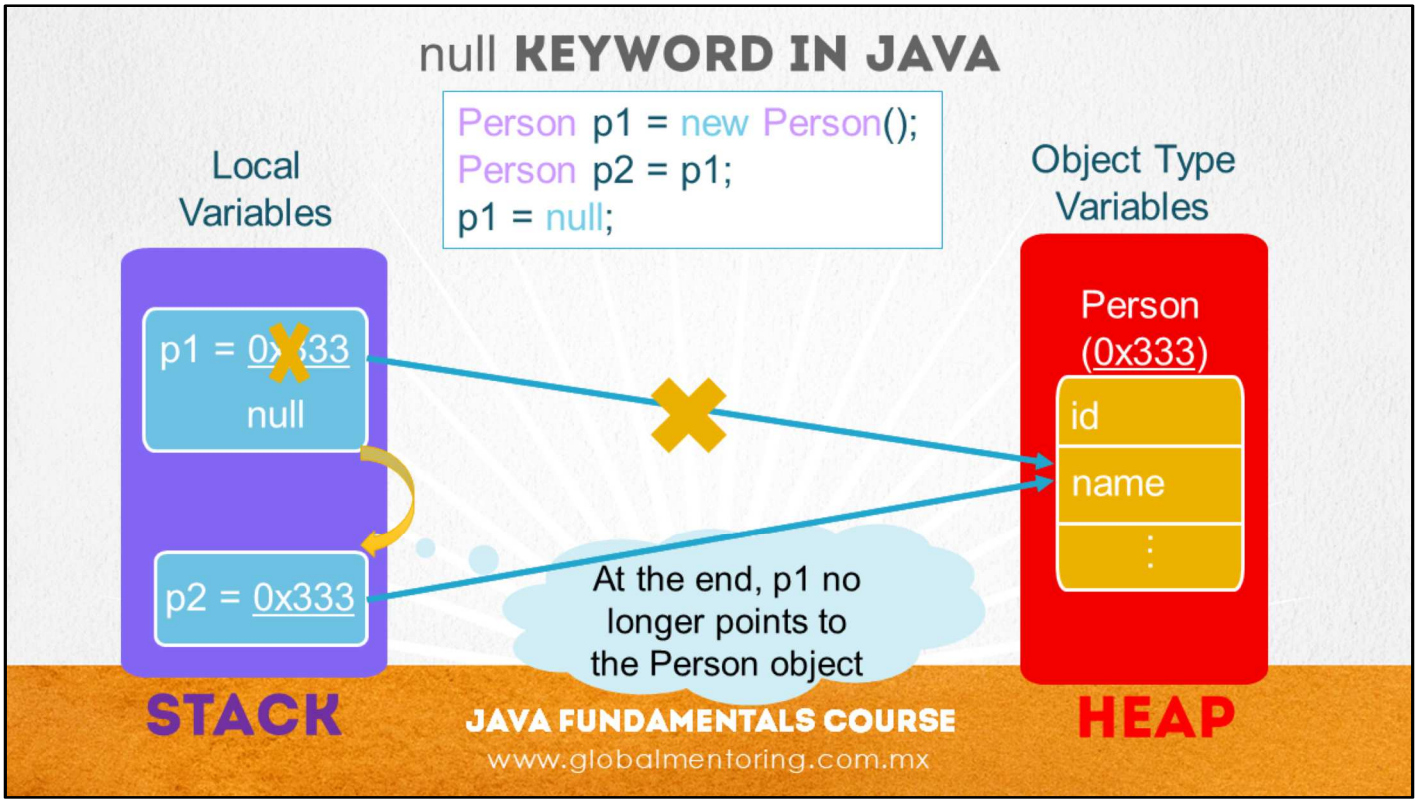
Hello, Ubaldo Acosta greets you. Welcome again. I hope you're ready to start with this lesson.

We are going to study the topic of the null keyword in Java.

Are you ready? OK let's go!



The word null in Java is used to indicate that no object reference has yet been assigned to a variable of type Object.

It is not possible to assign the null value to a variable of primitive type.

As we can see in the figure, first we create the variable p1 of type Person, which is created in the heap memory and assigned the memory location 0x333.

Then we create the variable p2 and assign the reference stored by the variable p1, that is, now both the variable p1 and the variable p2 point to the same object and both variables can access it.

Finally the variable p1 decided that we will not use it anymore, and for this we assign the value of null, this means that it loses the reference of the created object and therefore only the variable p2 will now be able to access the Person assigned in the memory location. 0x333.

Now, what happens if we do p2 = null?

This would mean that the Person object is no longer pointed by any variable, and therefore remains inaccessible. From the point of view of the garbage collector of Java it would be a candidate object to be eliminated from the memory, since no variable can access it and therefore it is unusable, and it only remains that it is eliminated from the Heap memory by means of the process of garbage collector by calling the System.gc () method, that is, garbage collector or garbage collector.

It should be noted that the garbage collector can not force it to start its process of cleaning objects, we can only send it to call the method and wait for the same Java virtual machine to decide when it is more convenient to start this process, since which is a process that uses many resources. However, if we are not going to use an object, it is convenient and good practice to assign the value of null to the variable that was pointing to the previously created object, with this once the trash collection process starts the created object will be candidate to be eliminated.

Finally, we can realize that the variables created in the Stack memory are temporary and of shorter duration, and once the method and / or program in which those variables were created are destroyed, they are destroyed. However, variables created in Heap memory, which are Java objects, tend to last longer in memory, and are destroyed until the process of the entire Java virtual machine is completed.

That is why it is good practice that if we are no longer going to use an object, we eliminate any reference of any variable that points to it by assigning the value of null to said variable.

EXAMPLE NULL KEYWORD IN JAVA

Use of null keyword:

```
1  public class PalabraNull {
2
3      public static void main(String[] args) {
4          Person p1 = new Person("John");
5          System.out.println("p1 name: " + p1.getName());
6
7          Person p2 = p1; //p2 points to the same object as p1
8          System.out.println("p2 name: " + p2.getName());
9
10         //We make p1 no longer point to the object p1
11         p1 = null;
12
13         //Check that p2 is still accessing the object
14         System.out.println("p2 name: " + p2.getName());
15     }
16 }
17
18 class Person{
19     String name; //default value is null
20
21     public Person(String nameArg){
22         name = nameArg;
23     }
24
25     public String getName(){
26         return name;
27     }
28 }
```

We can observe the code, in which we are creating a Person type object, and the reference is initially assigned to the variable p1 (line 4).

In line 5 we print the name of the Person whose reference is stored in the variable p1. And we can see that the value is John.

Later in the line 7 we create a variable p2, this variable we assign the value of p1, that is, now p2 also points to the same object created in line 4. This is seen in line 8 where the variable p2 prints the same name of the person assigned on p1.

Later we use the word **null** to indicate that the variable p1 does NOT point to any object, and in fact if we wanted to execute the method of p1.getName () would throw us an error since this variable can no longer access the methods or attributes of the Person object. The issue of handling exceptions will be studied in detail in later courses, however it is important to understand the use of null and the reason why we mark this error known as NullPointerExcepcion, which will be one of the most common when we work with Java, and basically it means that we are trying to access a method or attribute of a class in which the variable has not yet been assigned a reference of a valid object and whose value is null.

Finally, on line 14 we verify that the variable p2 continues accessing the Person object without problems and can continue printing the name assigned to this object.