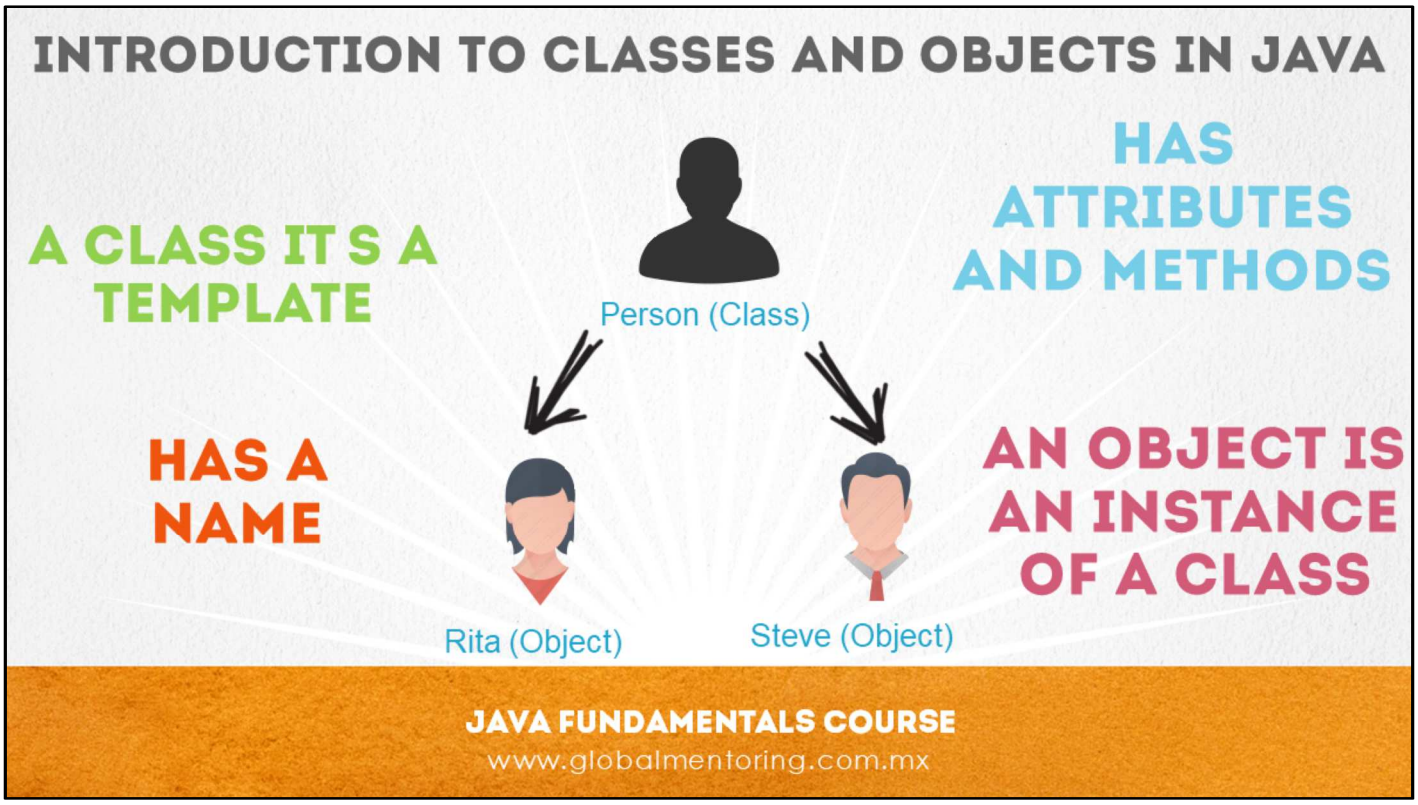
Hello, Ubaldo Acosta greets you. Welcome or welcome again. I hope you're ready to start with this lesson.

We are going to study the subject of Classes and Objects, this is the basis of object-oriented programming.

Are you ready? OK let's go!



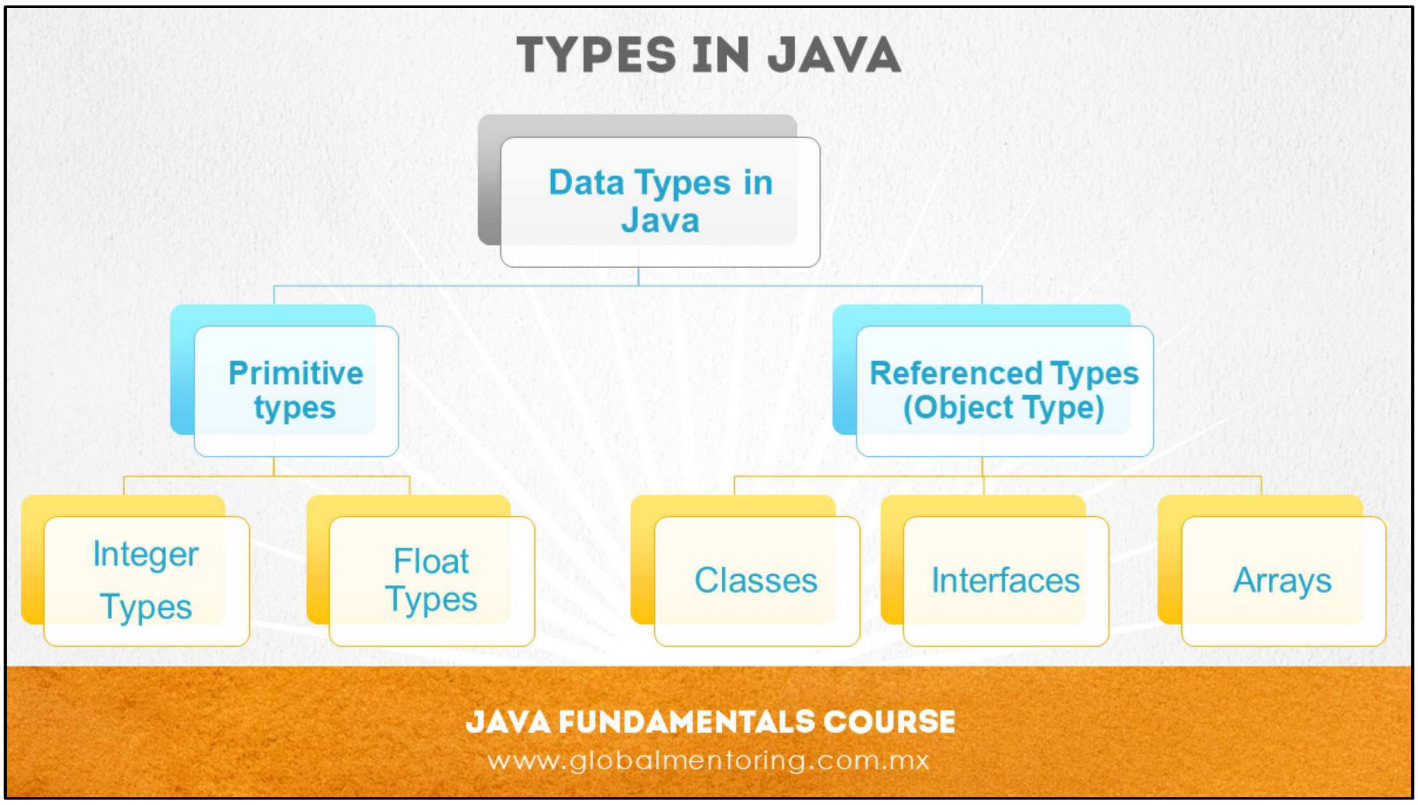
In Java the concept of Classes is fundamental to be able to handle this language in an appropriate way, since it is the foundation of this incredible programming language.

A class defines the nature of an object, therefore a class provides the bases of object-oriented programming. So any concept that is going to be implemented in Java is wrapped in a class. In this chapter we will see how to create a class and how we can create objects from our class.

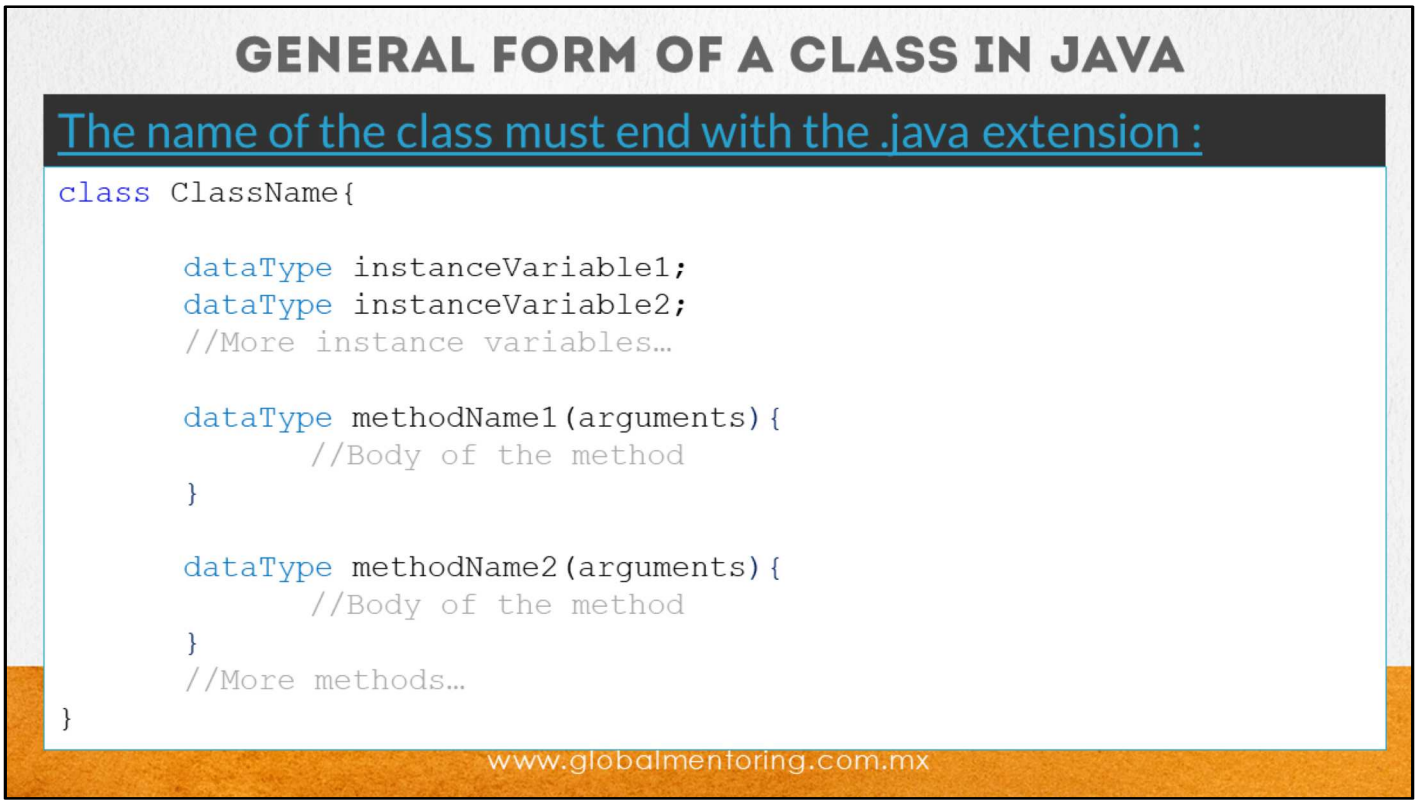
We have used the classes since the beginning of the course, but until now we will explain this concept. So far the classes that we have created have been created with the sole objective of encapsulating the main method and being able to execute our programs. However, we will begin to see what a class consists of and how we can benefit from this paradigm of Object Oriented Programming, also known as OOP.

Perhaps one of the most important points to understand is that a class generates a new type of data in Java. Once this type of data is defined, then we can use it to create objects of the previously defined type. Therefore a class is a template to be able to create objects. Because an object is an instance of a class, we will normally use "object" or "instance of a class" in an equivalent way.

For example, we can see in the figure a template of a Person, this person can have attributes (name, gender, occupation, etc), and methods to execute the actions that this person can perform (eating, sleeping, jumping, etc). Next we will see how to define these elements of a class in more detail.



Recall quickly that the types in Java, in addition to having primitive types, there are also referenced types. These types can come from Classes, Interfaces or Arrays. In this lesson we will study the Object types that are instances of a class, and later we will study the types Interface and Arrays.



In the code we can see the general structure of a class in Java. As we have said, a Class is a template, from which we can generate objects (instances) so that these new types of data can be finally used in our programs.

There are more variants, but in this code we are going to simplify the creation of a class, so that we have a more concrete idea of what it is and how we can start working with them. Let's see the general steps to create a class:

1) Define the name of the class, prefixing the reserved word class. This class should be saved in a file with the same name (remember that Java is case sensitive) with the extension .java

2) Define the attributes or variables of our class. These are known as instance variables of our class.

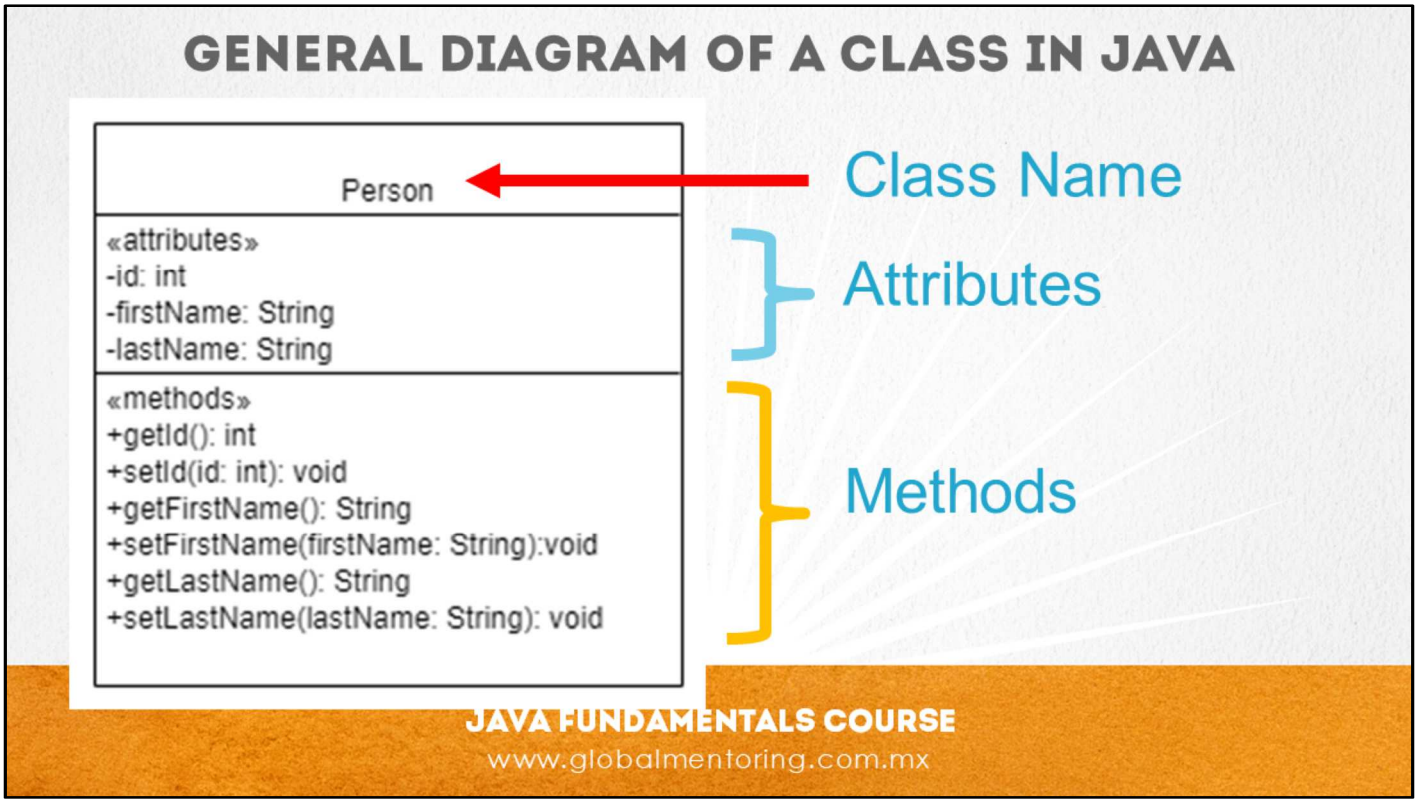
3) Define the methods of our class The methods are those that contain the code of our class, that is, the functionality and purpose of being of our class.

It is important to say that the definition of a class is precisely to define our template, from which we will create objects, and we'll finally work with those objects in our application.

On many occasions, when we begin with the process of creating a class is usually somewhat complicated to understand, since we must add variables and methods to our class, of which we still do not know exactly its functionality, then the question is: Should we define all the attributes and / or methods from the beginning to our class, or should we wait for the needs or requirements to complement the code of the class?

The answer to this question depends in many occasions on the methodology of software development that we are using, but in general, we will add the methods that arise from the analysis of our system. So possibly, the class will have more methods and attributes that we will use at the beginning, but our class or template will be ready to support future changes in our system. That is why sometimes we will find attributes or methods in our classes that are little or never used when creating the objects of these templates.

Later, we will see how to create objects from our class.



In the figure we can see the general structure of a class in Java. As we have said a Class is a template, from which we can generate objects (instances) so these new types of data can be finally used in our programs.

There are more variants, but in this figure we are going to simplify the creation of a class, so we have a more concrete idea of what it is and how we can start working with them. Let's see the general steps to create a class:

1) Define the name of the class

2) Define the attributes or variables of our class. This is known as instance variables of our class.

3) Define the methods of our class

Each of these points are those that we will be studying in the following lessons, from the creation of classes, definition of attributes of a class, as well as the creation of methods in a class.

CODE EXAMPLE OF A CLASS CALLED PERSON

Person.java Class:

```
//Class Name
class Persona {

    //Class atributes
    int id;
    String firstName;
    String lastName;

}
```

JAVA FUNDAMENTALS COURSE
www.globalmentoring.com.mx

Let's start the study of classes, with the definition of a very simple class. We can observe the definition of the Person class, which contains 3 attributes, one of int type and two attributes of String type, and without any method.

As we have said, when defining this class, we are creating a new data type in Java. In this case, the new data type is called Person. We will use this name to create Person objects. It is important to remember that when defining a class, we are only defining the template with which we are going to work, but in order to use this template we need to create objects from this template. Defining the class does not create any object automatically, we have to create them. This is what we will see next.