Differences Between Class and Object in Java

If you just started programming in Java, you might be confused about the difference between class and object.

In this tutorial, we'll explain the difference between them.

Classes in Java

A class represents a basic building block in all object-oriented programming languages. In Java, every code we write should be placed inside a class.

We can imagine classes as blueprints or templates for object creation. In other words, the class describes characteristics an object will have (once we create it) and what actions we could perform on it.

The main purpose of having classes is to create objects from them.

Within a class, we usually define an object's characteristics as instance variables and behavior as methods

.

For instance, suppose we’d like to create and use objects representing a dog. We’d like to be able to have more than one dog, each having different characteristics.

Firstly, let’s define a class. When defining a class, we should consider the characteristics and behaviors an object will have. For example, a dog could be described by its name, type, age, and color. Furthermore, each dog could perform actions like barking, sleeping, eating, and playing.

Secondly, based on the diagram, let’s define a Dog class:

```
public class Dog {
    private String name;
    private String type;
    private double age;
    private String color;

    public void makeSound(){
        System.out.println("Wuf");
    }

    public void sleep(){
        System.out.println("I'm sleeping");
    }

    public void play(){
        System.out.println("Playing is super fun");
    }
    public void eat(){
        System.out.println("Om nom nom");
    }

    // getters and setters
}
```

Finally, the Dog class represents a template for every object of type Dog we’ll create in the application. In addition, each dog is defined with the same characteristics (name, type, age, color), but we’ll put different values on them.

Objects in Java

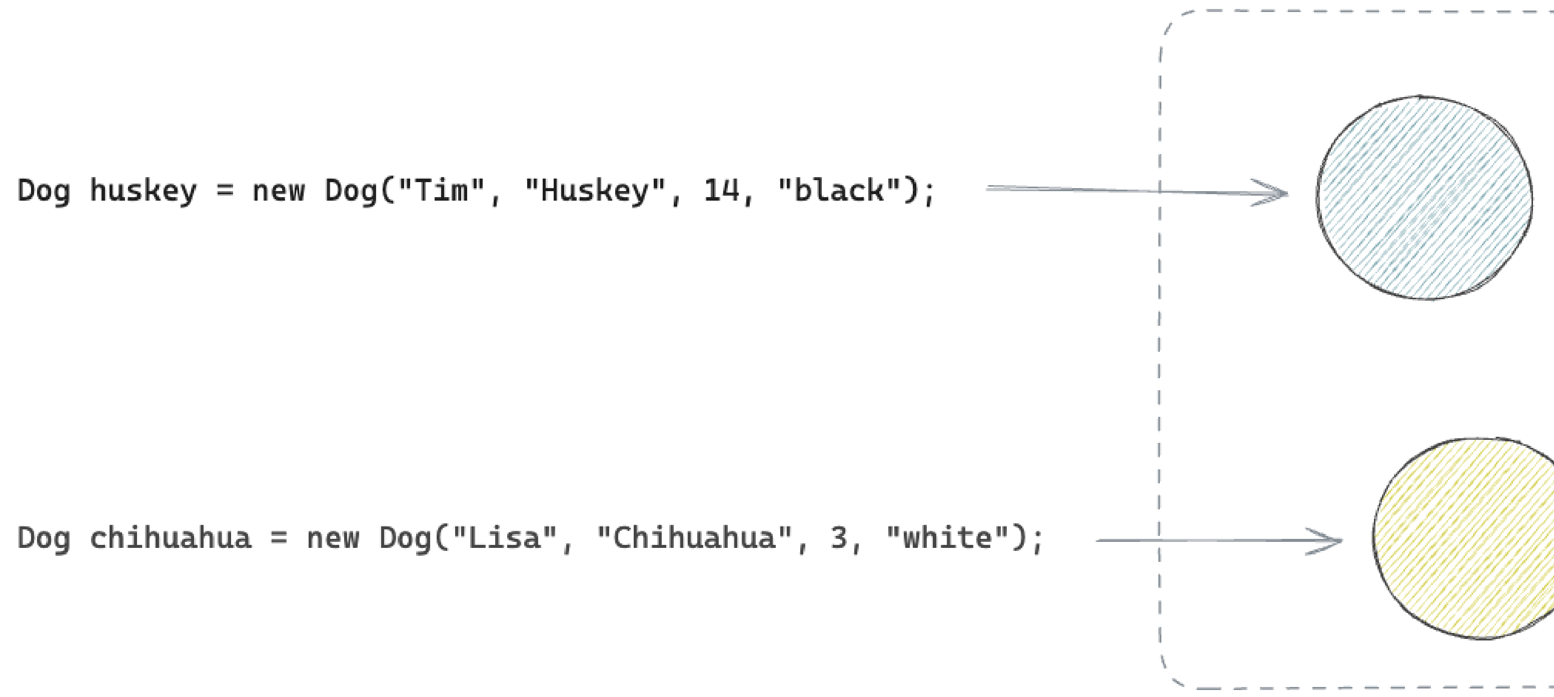
Simply put, an object is an instance of a class. While the class represents a logical entity, an object represents a physical entity. In other words, we use a logical entity (class) to create a physical entity (object).

Let’s create objects using our Dog class:

```
Dog huskey = new Dog("Tim", "Huskey", 14, "black");
Dog chihuahua = new Dog("Lisa", "Chihuahua", 3, "white");
```

Here, we created two instances from the Dog class. Each object has a different set of characteristics. We can create one or more objects using the same class.

Furthermore, once created, objects are stored and live inside a memory:



```
Dog huskey = new Dog("Tim", "Huskey", 14, "black");

Dog chihuahua = new Dog("Lisa", "Chihuahua", 3, "white");
```

They will stay inside a memory until they become eligible for garbage collection.

Differences Between Class And Object

Let’s list the differences between classes and objects in a table view:

Class	Object
Blueprint for object creation.	An instance of a class.
Describes which characteristics and actions the object will have.	Has characteristics and actions defined within the class.
Logical entity.	Physical entity.
Doesn’t allocate memory.	Allocates memory when created.
Each class is created only once in an application.	We can create more than one object using the same class.

5. Conclusion

In this short article, we learned the differences between classes and objects in Java.

To summarize, we create objects from classes. Classes represent a blueprint, describing an object – what characteristics it will have and what it will be able to do. On the other hand, an object is just a physical manifestation of the class. Once we create it, it lives in memory until garbage is collected.

In this short tutorial, we’ve looked at the differences between classes and objects in Java.

Read More