

What is Class and Object in Java OOPS? Learn with Example

Classes and Objects in Java are the fundamental components of OOP's. Often there is a confusion between classes and objects. In this tutorial, we try to tell you the difference between Class and Object in Java.

First, let's understand what they are,

- [What is Class in Java?](#)
- [What is an Object in Java?](#)
- [What is the Difference Between Object and Class in Java?](#)
- [Concept of Classes and Objects](#)
- [Classes and Objects in Java Example Programs](#)
- [Java Object and Class Example: main outside class](#)

What is Class in Java?

Class are a blueprint or a set of instructions to build a specific type of object. It is a basic concept of Object-Oriented Programming which revolve around the real-life entities. Class in Java determines how an object will behave and what the object will contain.

Syntax

```
class <class_name>{  
    field;  
    method;  
}
```

What is Object in Java?

Object is an instance of a class. An object in OOPS is nothing but a self-contained component which consists of methods and properties to make a particular type of data useful. For example color name, table, bag, barking. When you send a message to an object, you are asking the object to invoke or execute one of its methods as defined in the class.

From a programming point of view, an object in OOPS can include a data structure, a variable, or a function. It has a memory location allocated. Java Objects are designed as class hierarchies.

Syntax

```
ClassName ReferenceVariable = new ClassName();
```

What is the Difference Between Object and Class in Java?

A **Class** in object oriented programming is a blueprint or prototype that defines the variables and the methods (functions) common to all Java Objects of a certain kind.

An **object** in OOPS is a specimen of a class. Software objects are often used to model real-world objects you find in everyday life.

Classes and Objects in Java: Tutorial for Begin...

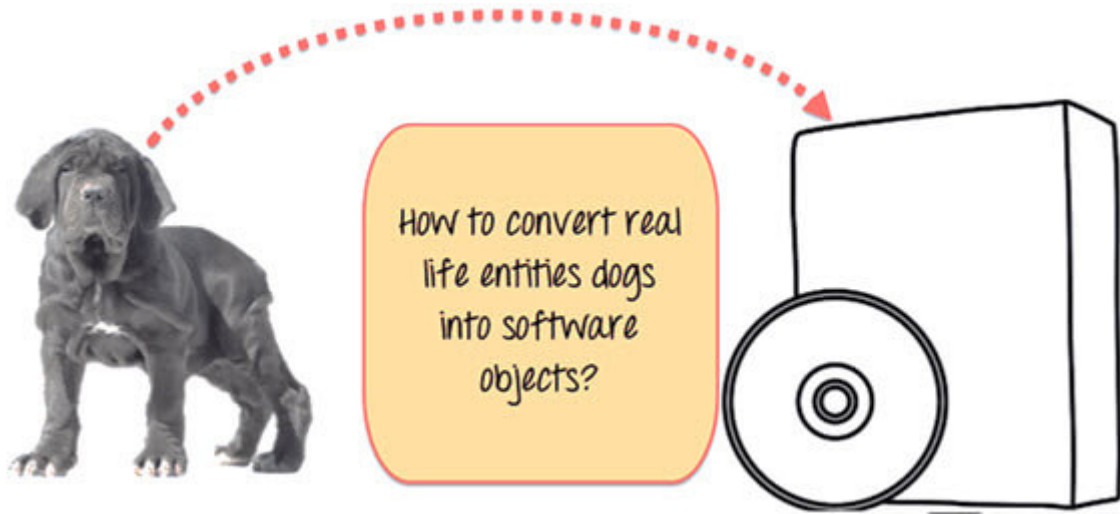


Click [here \(/faq.html#1\)](/faq.html#1) if the video is not accessible

Understand the concept of Java Classes and Objects with an example.

Let's take an example of developing a pet management system, specially meant for dogs. You will need various information about the dogs like different breeds of the dogs, the age, size, etc.

You need to model real-life beings, i.e., dogs into software entities.



[./images/java/052016_0704_ObjectsandC1.jpg](#)

Moreover, the million dollar question is, how you design such software?

Here is the solution-

First, let's do an exercise.

You can see the picture of three different breeds of dogs below.



[./images/java/052016_0704_ObjectsandC2.jpg](#)

Stop here right now! List down the differences between them.

Some of the differences you might have listed out maybe breed, age, size, color, etc. If you think for a minute, these differences are also some common characteristics shared by these dogs. These characteristics (breed, age, size, color) can form a data members for your object.

COMMON CHARACTERISTICS

- ☒ Breed
- ☒ Size
- ☒ Age
- ☒ Color



[./images/java/052016_0704_ObjectsandC3.jpg](http://images/java/052016_0704_ObjectsandC3.jpg)

Next, list out the common behaviors of these dogs like sleep, sit, eat, etc. So these will be the actions of our software objects.

COMMON ACTIONS

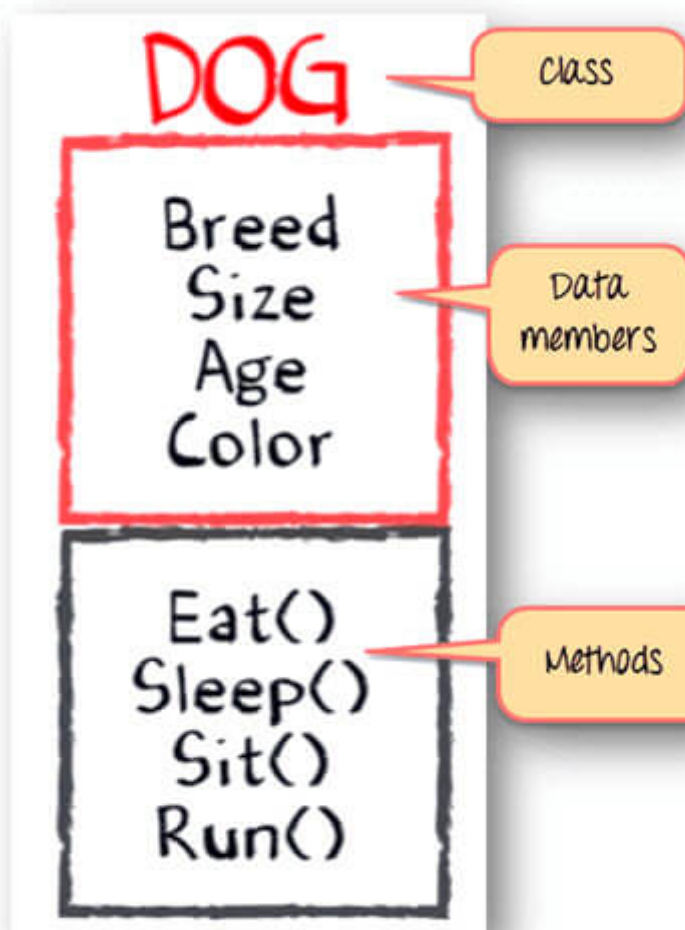
- ☒ Eat
- ☒ Sleep
- ☒ Sit
- ☒ Run



[./images/java/052016_0704_ObjectsandC4.jpg](http://images/java/052016_0704_ObjectsandC4.jpg)

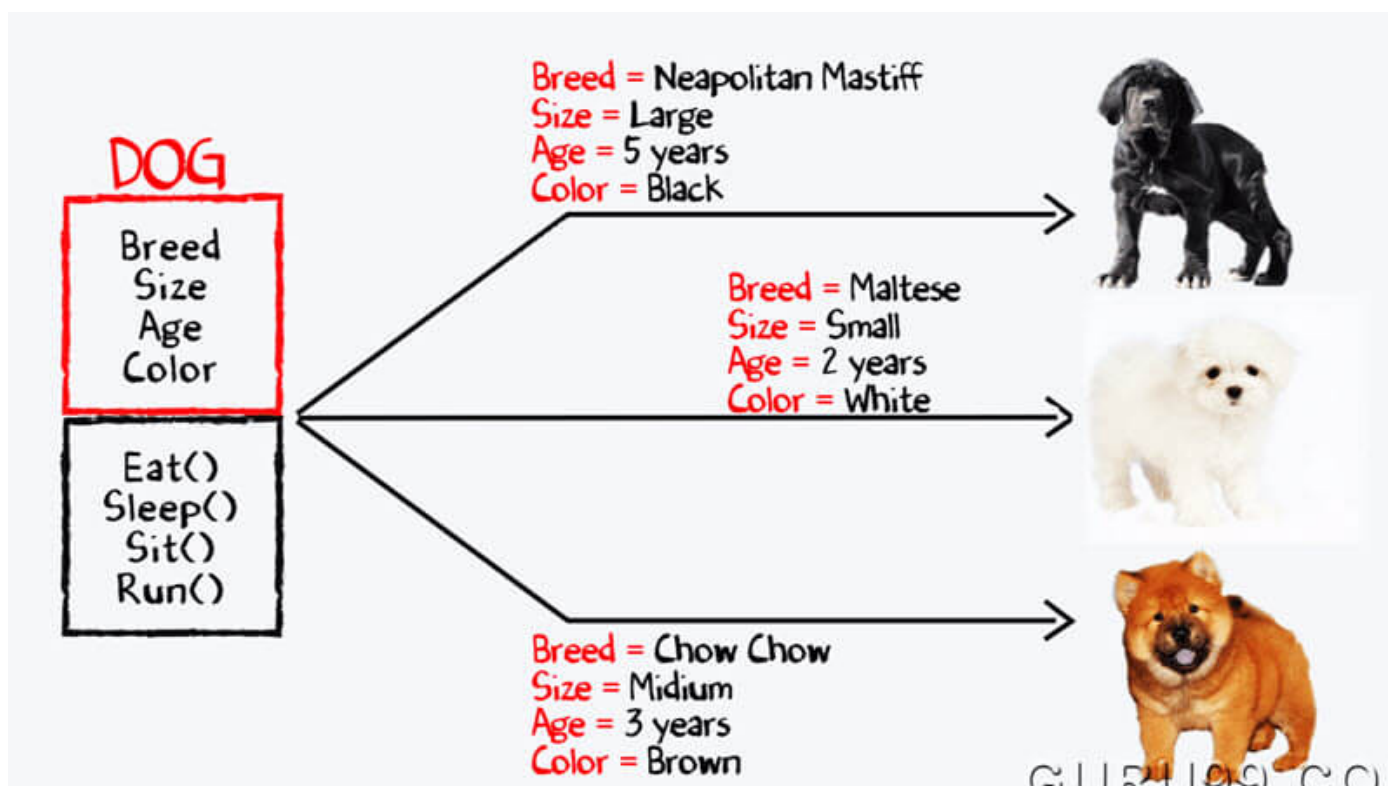
So far we have defined following things,

- **Class** - Dogs
- **Data members** or **objects**- size, age, color, breed, etc.
- **Methods**- eat, sleep, sit and run.



(./images/java/052016_0704_ObjectsandC5.jpg).

Now, for different values of data members (breed size, age, and color) in Java class, you will get different dog objects.



(./images/java/052016_0704_ObjectsandC6.jpg).

You can design any program using this OOPs approach.

While creating a class, one must follow the following principles.

- **Single Responsibility Principle (SRP)**- A class should have only one reason to change
- **Open Closed Responsibility (OCP)**- It should be able to extend any classes without modifying it
- **Liskov Substitution Responsibility (LSR)**- Derived classes must be substitutable for their base classes
- **Dependency Inversion Principle (DIP)**- Depend on abstraction and not on concretions
- **Interface Segregation Principle (ISP)**- Prepare fine grained interfaces that are client specific.

Classes and Objects in Java Example Programs

```
// Class Declaration
public class Dog {
    // Instance Variables
    String breed;
    String size;
    int age;
    String color;

    // method 1
    public String getInfo() {
        return ("Breed is: "+breed+" Size is:"+size+" Age is:"+age+" color
is: "+color);
    }

    public static void main(String[] args) {
        Dog maltese = new Dog();
        maltese.breed="Maltese";
        maltese.size="Small";
        maltese.age=2;
        maltese.color="white";
        System.out.println(maltese.getInfo());
    }
}
```

Output:

```
Breed is: Maltese Size is:Small Age is:2 color is: white
```

Java Object and Class Example: main outside class

In previous program, we are creating main() method inside the class. Now, we create classes and define main() method in another class. This is a better way than previous one.

```
// Class Declaration
class Dog {
    // Instance Variables
    String breed;
    String size;
    int age;
    String color;

    // method 1
    public String getInfo() {
        return ("Breed is: "+breed+" Size is:"+size+" Age is:"+age+" color
is: "+color);
    }
}

public class Execute{
    public static void main(String[] args) {
        Dog maltese = new Dog();
        maltese.breed="Maltese";
        maltese.size="Small";
        maltese.age=2;
        maltese.color="white";
        System.out.println(maltese.getInfo());
    }
}
```

Output:

```
Breed is: Maltese Size is:Small Age is:2 color is: white
```

Summary:

- Java Class is an entity that determines how Java Objects will behave and what objects will contain
- A Java object is a self-contained component which consists of methods and properties to make certain type of data useful
- A class system allows the program to define a new class (derived class) in terms of an existing class (superclass) by using a technique like inheritance, overriding and augmenting.