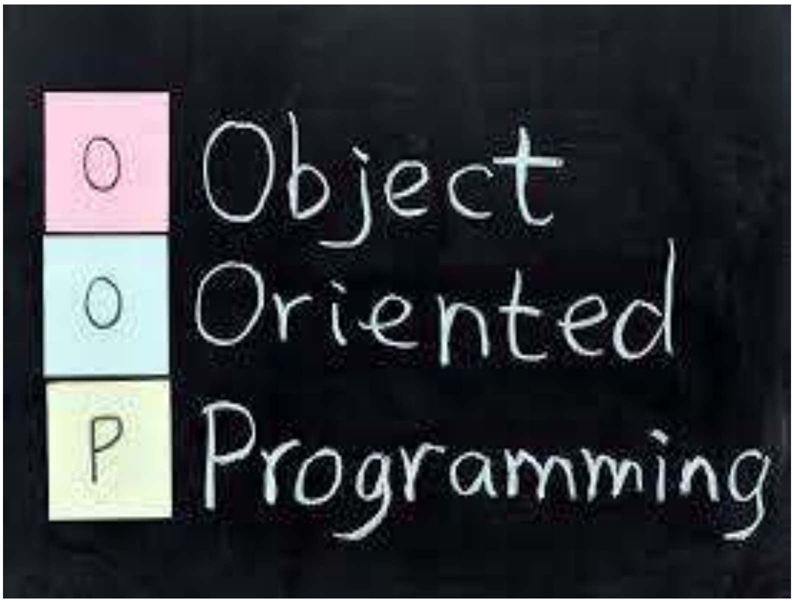


What is Object Oriented Programming?

Object-oriented programming (OOP) is a fundamental programming paradigm used by nearly every developer at some point in their career. OOP is the most popular programming paradigm and is taught as the standard way to code for most of a programmer's educational career.

Its relies on the concept of classes and objects. It is used to structure a software program into simple, reusable pieces of code blueprints (usually called classes), which are used to create individual instances of objects. There are many object-oriented programming languages including JavaScript, C++, Java, and Python.

The core concept of the object-oriented approach is to break complex problems into smaller objects.

Building blocks of OOP

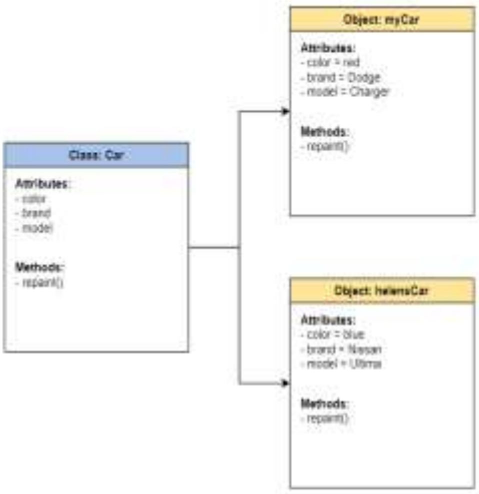
Next, we’ll take a deeper look at each of the fundamental building blocks of an OOP program used above:

Classes

Objects

Methods

Attributes



Class blueprint being used to create two Car type objects, myCar and helensCar

An object is any entity that has a state and behavior. For example, a bicycle is an object. It has

States: idle, first gear, etc

Behaviors: braking, accelerating, etc.

Before we learn about objects, let's first know about classes in Java.

Classes

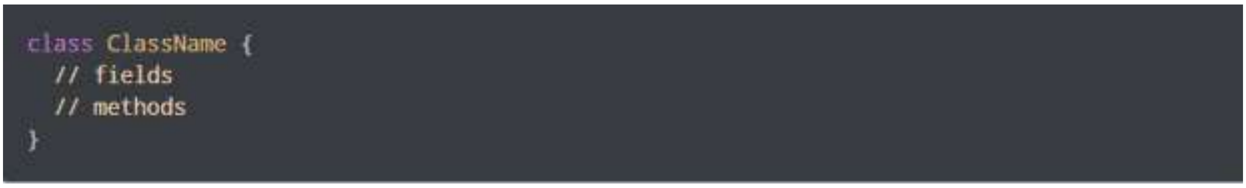
A class is a blueprint for the object. Before we create an object, we first need to define the class.

We can think of the class as a sketch (prototype) of a house. It contains all the details about the floors, doors, windows, etc. Based on these descriptions we build the house. House is the object.

Since many houses can be made from the same description, we can create many objects from a class.

Create a class in Java.

We can create a class in Java using the class keyword. For example,



```
class ClassName {
  // fields
  // methods
}
```

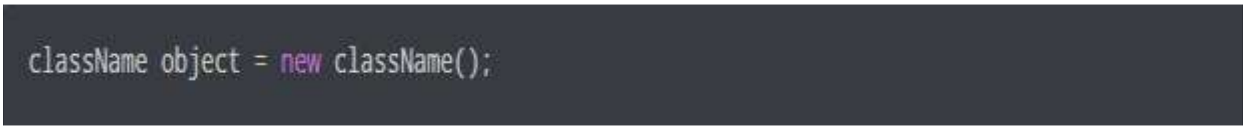
Here, fields (variables) and methods represent the state and behavior of the object respectively.

fields are used to store data.

methods are used to perform some operations.

Objects

Of course, OOP includes objects! Objects are instances of classes created with specific data, for example:



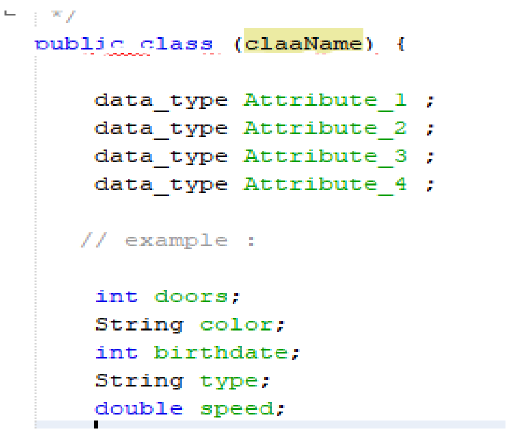
```
className object = new className();
```

We have used the **new** keyword along with the constructor of the class to create an object. Constructors are similar to methods and have the same name as the class.

Attributes

Attributes are the information that is stored. Attributes are defined in the Class template. When objects are instantiated, individual objects contain data stored in the Attributes field.

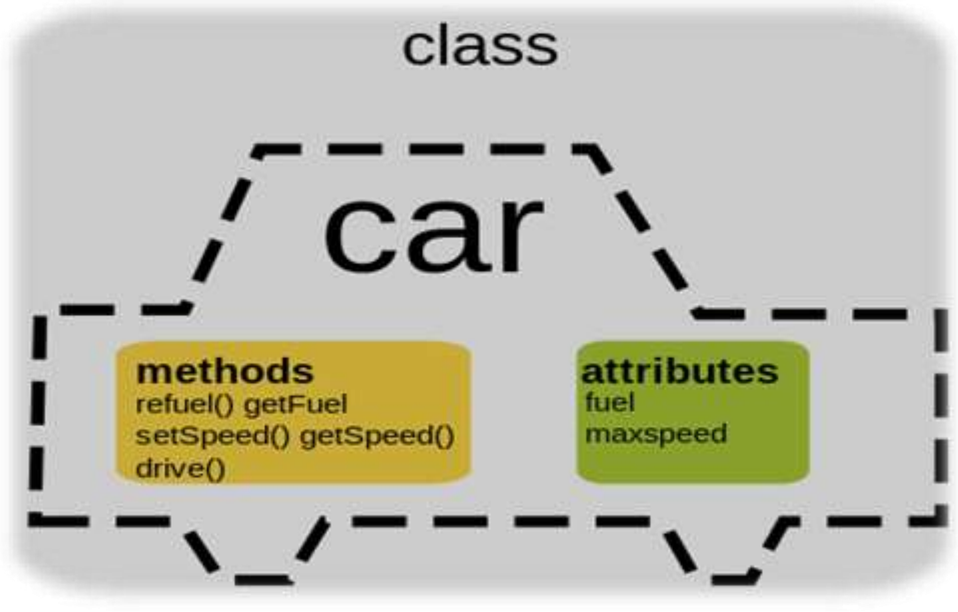
The state of an object is defined by the data in the object's attributes fields.



```
 */
public class (claaName) {

    data_type Attribute_1 ;
    data_type Attribute_2 ;
    data_type Attribute_3 ;
    data_type Attribute_4 ;

   // example :

    int doors;
    String color;
    int birthdate;
    String type;
    double speed;
```

Methods

Methods represent behaviors. Methods perform actions; methods might return information about an object, or update an object's data. The method's code is defined in the class definition.



Benefits of OOP

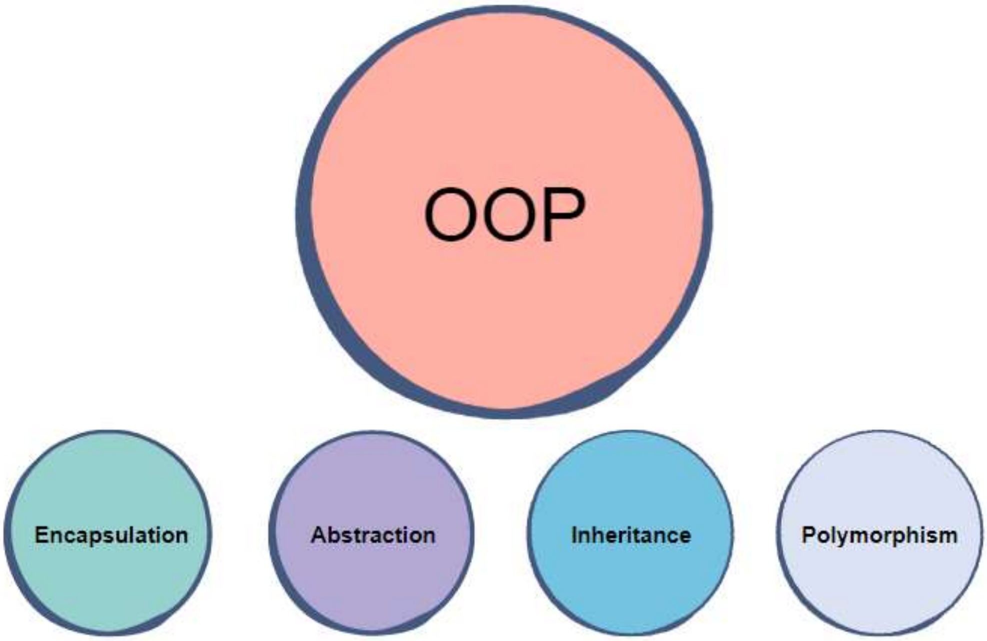
OOP models complex things as reproducible, simple structures

Reusable, OOP objects can be used across programs.

Allows for class-specific behavior through polymorphism.

Easier to debug, classes often contain all applicable information to them.

Secure, protects information through encapsulation.



Four Principles of OOP

The four pillars of object oriented programming are:

Inheritance: child classes inherit data and behaviors from parent class

Encapsulation: containing information in an object, exposing only selected information

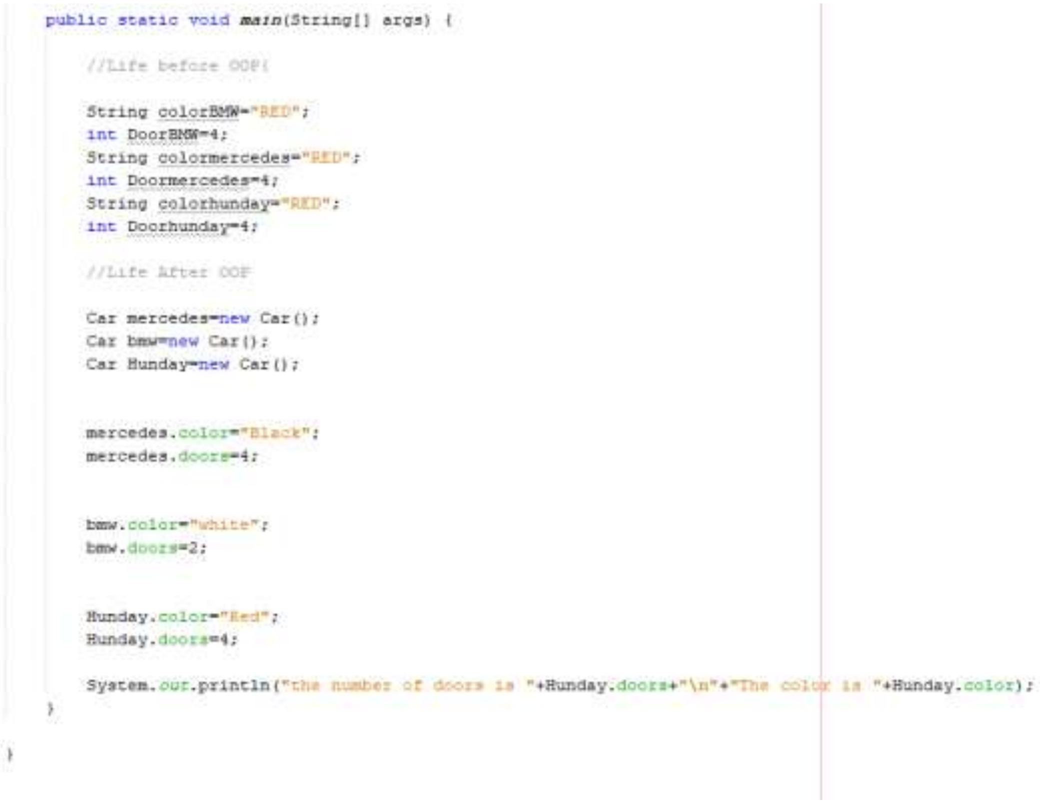
Abstraction: only exposing high level public methods for accessing an object

Polymorphism: many methods can do the same task

How to structure OOP programs

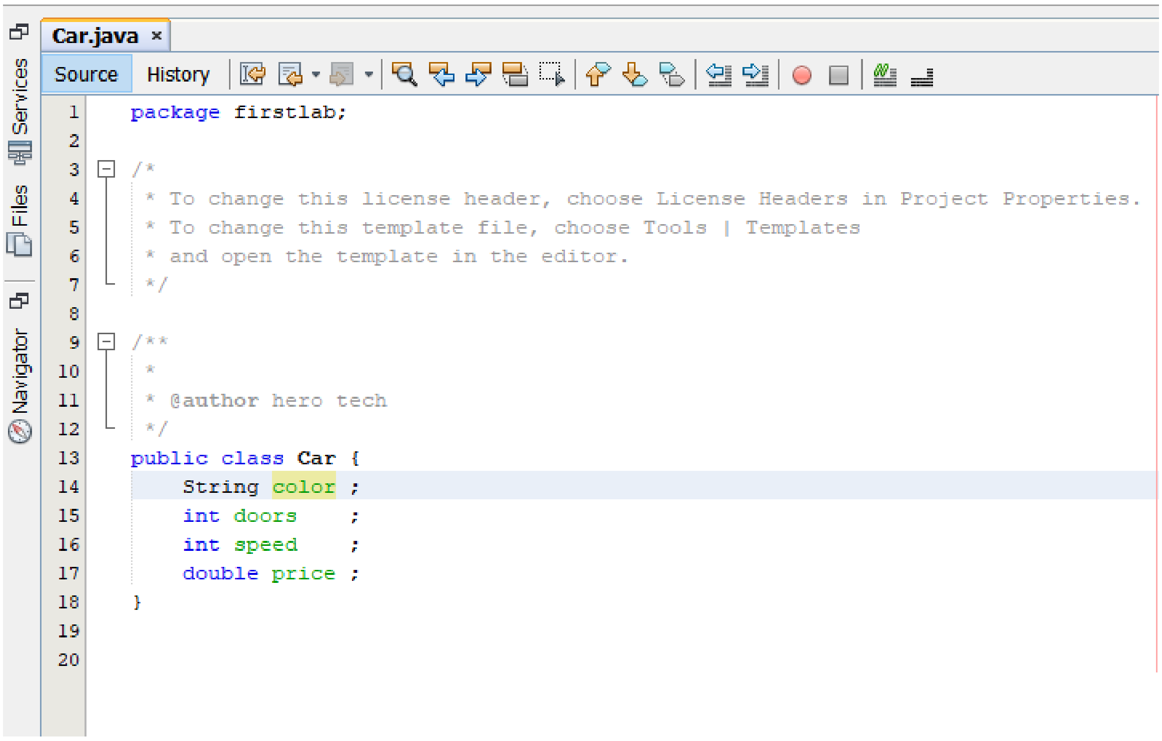
Let’s take a real-world problem, and conceptually design an OOP software program.

Package:



```
    public static void main(String[] args) {

        //Life before OOP(

        String colorBMW="RED";
        int DoorBMW=4;
        String colormercedes="RED";
        int Doormercedes=4;
        String colorhunday="RED";
        int Doorhunday=4;

        //Life After OOP

        Car mercedes=new Car();
        Car bmw=new Car();
        Car Hunday=new Car();


        mercedes.color="Black";
        mercedes.doors=4;


        bmw.color="white";
        bmw.doors=2;


        Hunday.color="Red";
        Hunday.doors=4;

        System.our.println("the number of doors is "+Hunday.doors+"\n"+"The color is "+Hunday.color);
    }

}
```

Class:



```
package firstlab;

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**
 *
 * @author hero tech
 */
public class Car {
    String color ;
    int doors    ;
    int speed    ;
    double price ;
}
```