Let's consider a basic scenario of modeling a "Car" in Java.

java

Copy code

// Car class

class Car {

// Fields

String make;

String model;

int year;

boolean isRunning;

// Constructor

public Car(String make, String model, int year) {

this.make = make;

this.model = model;

this.year = year;

this.isRunning = false; // By default, the car is not running

}

// Methods

public void start() {

if (!isRunning) {

System.out.println("The " + year + " " + make + " " + model + " is starting.");

isRunning = true;

} else {

System.out.println("The car is already running.");

}

}

public void stop() {

if (isRunning) {

System.out.println("The " + year + " " + make + " " + model + " is stopping.");

isRunning = false;

} else {

System.out.println("The car is already stopped.");

}

}

public void displayInfo() {

System.out.println("Car Information:");

System.out.println("Make: " + make);

System.out.println("Model: " + model);

System.out.println("Year: " + year);

System.out.println("Is Running: " + (isRunning ? "Yes" : "No"));

}

}

// Main class

public class CarDemo {

public static void main(String[] args) {

// Creating instances of the Car class

Car myCar = new Car("Toyota", "Camry", 2020);

Car anotherCar = new Car("Honda", "Civic", 2021);

// Using methods to interact with the objects

myCar.start();

myCar.displayInfo();

myCar.stop();

System.out.println(); // Adding a newline for clarity

anotherCar.displayInfo();

anotherCar.start();

anotherCar.displayInfo();

}

}

In this example, we have a Car class with fields such as make, model, year, and isRunning. The class has a constructor for initializing the car, methods for starting and stopping the car, and a method for displaying car information.

In the CarDemo class, we create instances of the Car class (objects) and demonstrate how to use methods to interact with these objects. This example can help students understand the basic concepts of classes, objects, fields, constructors, and methods in Java.