COURSEWARE

Class Members

Contents

Overview

`Class members` are either `variables` or `methods` within a `class`.
Each class member can be an `instance` member, a `static` class member, and can also be `final`.

Tutorial

Instance Class Members

Instance members are attributes or methods which need to have an instance of the class instantiated before it can be used and are unique to that instance.

```
public class Customer {
    private String firstName;
    private String surname;

    public Customer(String firstName, String surname) {
        this.firstName = firstName;
        this.surname = surname;
    }

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
}
```

In the above example, we have a class, Customer, which has two variables and `getter` and `setter` methods for the variables.
Since these class members are not static an instance of Customer would need to be instantiated in order to access the class members.
Once we instantiate an object of Customer, we can still instantiate a second object of Customer with different data stored for firstName and surname as the class members are instance members and are unique to the instance of the class.

```
public class Main {
    public static void main(String[] args) {
        Customer firstCustomer = new Customer("Arnold","Rimmer");
        Customer secondCustomer = new Customer("David","Lister");

        firstCustomer.setFirstName("Arnold J.");
        secondCustomer.setFirstName("Dave");

        System.out.println(firstCustomer.getFirstName());   // output: Arnold J.
        System.out.println(secondCustomer.getFirstName());  // output: Dave
    }
}
```

Static Class Members

Static class members are attributes or methods which can be accessed without instantiating an object of the class first because they belong to the class, not the object.
Any static class members are universal across all instances of a class.

```
public class Customer {

    private String firstName;
    private String surname;
    private static int numberOfPeople = 0;

    public Customer(String firstName, String surname) {
        this.firstName = firstName;
        this.surname = surname;
        numberOfPeople++;
    }

    public static int getNumberOfPeople() {
        return numberOfPeople;
    }

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
}
```

In the above example, we now have two new `static` class members, the variable numberOfPeople and the method getNumberOfPeople.
Because they are `static` they belong to the class, not the instance of the class, meaning that we can access them without having to instantiate an object.

```
public class Main {
    public static void main(String[] args) {
        System.out.println(Customer.getNumberOfPeople());   // output: 0    No
objects instantiated

        Customer firstCustomer = new Customer("Arnold","Rimmer");
        Customer secondCustomer = new Customer("David","Lister");

        System.out.println(firstCustomer.getFirstName());   // output: Arnold
        System.out.println(secondCustomer.getFirstName());  // output: David

        System.out.println(firstCustomer.getNumberOfPeople());  // output: 2
        System.out.println(secondCustomer.getNumberOfPeople()); // output: 2
        System.out.println(Customer.getNumberOfPeople());       // output: 2
    }
}
```

Final Class Members

Final is a keyword in Java that deserves it's own module.
The Final module can be found here.

Exercises

There are no exercises for this module.