Difference between Abstraction and Encapsulation in Java with Examples

Difficulty Level : Easy Last Updated : 08 Nov, 2019

Encapsulation in Java

Encapsulation is defined as the wrapping up of data under a single unit. It is the mechanism that binds together code and the data it manipulates. Another way to think about encapsulation is, it is a protective shield that prevents the data from being accessed by the code outside this shield.

Technically in encapsulation, the variables or data of a class is hidden from any other class and can be accessed only through any member function of own class in which they are declared.
As in encapsulation, the data in a class is hidden from other classes, so it is also known as data-hiding.
Encapsulation can be achieved by Declaring all the variables in the class as private and writing public methods in the class to set and get the values of variables.

```
// Java program to demonstrate encapsulation

public class Encapsulate {

    // private variables declared
    // these can only be accessed by
    // public methods of class
    private String geekName;
    private int geekRoll;
    private int geekAge;

    // get method for age to access
    // private variable geekAge
    public int getAge()
    {
        return geekAge;
    }

    // get method for name to access
    // private variable geekName
    public String getName()
    {
        return geekName;
    }

    // get method for roll to access
    // private variable geekRoll
    public int getRoll()
    {
        return geekRoll;
    }

    // set method for age to access
    // private variable geekage
    public void setAge(int newAge)
    {
        geekAge = newAge;
    }

    // set method for name to access
    // private variable geekName
    public void setName(String newName)
    {
        geekName = newName;
    }

    // set method for roll to access
    // private variable geekRoll
    public void setRoll(int newRoll)
    {
        geekRoll = newRoll;
    }
}

// Class to access variables
// of the class Encapsulate
class TestEncapsulation {
    public static void main(String[] args)
    {
        Encapsulate obj = new Encapsulate();

        // setting values of the variables
        obj.setName("Harsh");
        obj.setAge(19);
        obj.setRoll(51);

        // Displaying values of the variables
        System.out.println("Geek's name: " + obj.getName());
        System.out.println("Geek's age: " + obj.getAge());
        System.out.println("Geek's roll: " + obj.getRoll());

        // Direct access of geekRoll is not possible
        // due to encapsulation
        // System.out.println("Geek's roll: " + obj.geekName);
    }
}
```

Output:

```
Geek's name: Harsh
Geek's age: 19
Geek's roll: 51
```

Abstraction in Java

Data Abstraction is the property by virtue of which only the essential details are displayed to the user. The trivial or the non-essentials units are not displayed to the user. Ex: A car is viewed as a car rather than its individual components.

Data Abstraction may also be defined as the process of identifying only the required characteristics of an object ignoring the irrelevant details. The properties and behaviours of an object differentiate it from other objects of similar type and also help in classifying/grouping the objects.

```java
// Java program to illustrate the concept of Abstraction

abstract class Shape {
    String color;

    // these are abstract methods
    abstract double area();
    public abstract String toString();

    // abstract class can have a constructor
    public Shape(String color)
    {
        System.out.println("Shape constructor called");
        this.color = color;
    }

    // this is a concrete method
    public String getColor()
    {
        return color;
    }
}
class Circle extends Shape {
    double radius;

    public Circle(String color, double radius)
    {
        // calling Shape constructor
        super(color);
        System.out.println("Circle constructor called");
        this.radius = radius;
    }

    @Override
    double area()
    {
        return Math.PI * Math.pow(radius, 2);
    }

    @Override
    public String toString()
    {
        return "Circle color is "
            + super.color
            + "and area is : "
            + area();
    }
}

class Rectangle extends Shape {

    double length;
    double width;

    public Rectangle(String color,
                     double length,
                     double width)
    {

        // calling Shape constructor
        super(color);
        System.out.println("Rectangle constructor called");
        this.length = length;
        this.width = width;
    }

    @Override
    double area()
    {
        return length * width;
    }

    @Override
    public String toString()
    {
        return "Rectangle color is "
            + super.color
            + "and area is : "
            + area();
    }
}

public class Test {
    public static void main(String[] args)
    {
        Shape s1 = new Circle("Red", 2.2);
        Shape s2 = new Rectangle("Yellow", 2, 4);

        System.out.println(s1.toString());
        System.out.println(s2.toString());
    }
}
```

Output:

```
Shape constructor called
Circle constructor called
Shape constructor called
Rectangle constructor called
Circle color is Redand area is : 15.205308443374602
Rectangle color is Yellowand area is : 8.0
```

Difference between Abstraction and Encapsulation:

Abstraction	Encapsulation
Abstraction is the process or method of gaining the information.	While encapsulation is the process or method to contain the information.
In abstraction, problems are solved at the design or interface level.	While in encapsulation, problems are solved at the implementation level.
Abstraction is the method of hiding the unwanted information.	Whereas encapsulation is a method to hide the data in a single entity or unit along with a method to protect information from outside.
We can implement abstraction using abstract class and interfaces.	Whereas encapsulation can be implemented using by access modifier i.e. private, protected and public.
In abstraction, implementation complexities are hidden using abstract classes and interfaces.	While in encapsulation, the data is hidden using methods of getters and setters.
The objects that help to perform abstraction are encapsulated.	Whereas the objects that result in encapsulation need not be abstracted.