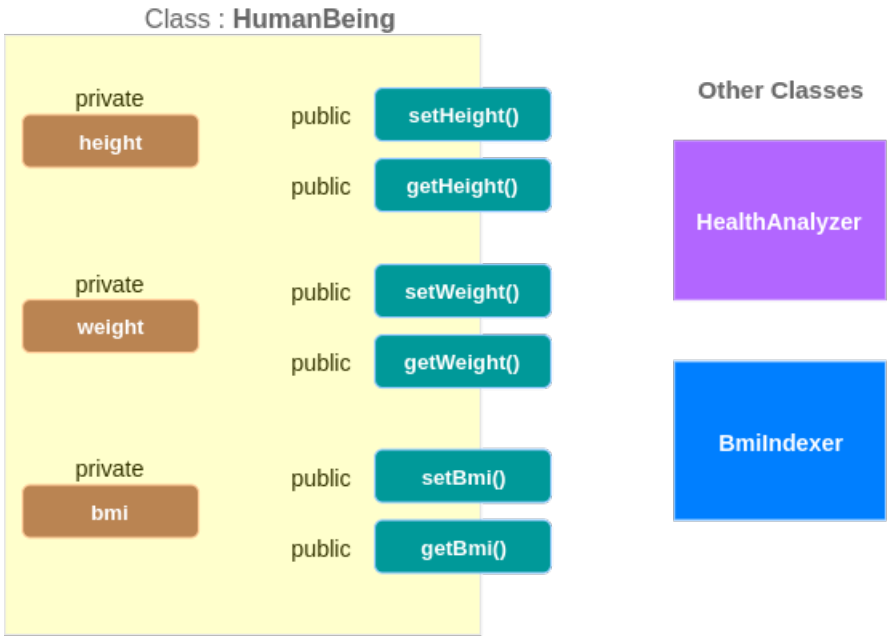
# Encapsulation in Java

## Encapsulation in Java

**Encapsulation in Java** – Hiding the variables of a class from other classes, and giving access to them only through methods(setters and getters). Hence, Encapsulation in Java language means binding the data (variables) with the code(methods – setters and getters). In this tutorial, we shall learn the intuition and realization of encapsulation in Java language with example programs.

The below diagram gives an idea about Encapsulation in Java




Points to be made from the above diagram are

- Variables (in the example: height, weight, bmi) are declared private, and hence not visible to other classes.
- For each variable, there is a setter method and getter method, which sets a value to the variable and gets the value of the variable respectively. Example : For variable `height`, setter method is `setHeight()`, getter method is `getHeight()`.
- Setter and Getter methods are public, hence visible to other classes.

Based on the presence of getter and setter for a variable, the read and write permissions to the class variable could be set. Following table elaborates this idea.

| Setter | Getter | Permission to the variable is |
|---|---|---|
| Present | Not Present | write-only |

| Setter | Getter | Permission to the variable is |
|---|---|---|
| Not Present | Present | read-only |
| Present | Present | write and read |

## For a variable in the class

- If setter is present and getter is not present, the variable is write-only, which means, other classes could only modify the value of the variable, but cannot read its value.
- If setter is not present and getter is present, the variable is read-only, which means, other classes can only read value of the variable, but cannot modify its value.
- If both setter and getter are present, the variable could be read or modified by the other classes

## Example 1 – Encapsulation in Java

In the following example, Human.java class has variables `height`, `weight` and `bmi` which are private. For each variable, there is a setter and getter.

```
package com.tutorialkart.java;
/**
 * @author tutorialkart
 */
class Human{
    private float weight;
    private float height;
    private float bmi;
    public float getWeight() {
        return weight;
    }
    public void setWeight(float weight) {
        this.weight = weight;
    }
    public float getHeight() {
        return height;
    }
    public void setHeight(float height) {
        this.height = height;
    }
    public float getBmi() {
        return bmi;
    }
    public void setBmi(float bmi) {
        this.bmi = bmi;
    }
}

public class EncapsulationExample {
    public static void main(String[] args) {
        Human h1 = new Human();
        // using setters of Human
        h1.setHeight(1.65f);
        h1.setWeight(68);
        h1.setBmi(calculateBmi(h1));

        // using getters of Human
        System.out.println("Person has "+h1.getWeight()+" kgs and is "+h1.getHeight()+"
```

```
        System.out.println("Person has "+h1.getWeight()+" kgs and is "+h1.getHeight()+
    }

    public static float calculateBmi(Human h1){
        return h1.getWeight()/(h1.getHeight()*h1.getHeight());
    }
}
```

**Output**

```
Person has 68.0 kgs and is 1.65 meters in height, which results in BMI of 24.977045
```

## Conclusion

In this Java Tutorial – Encapsulation in Java, we have learned to encapsulate data inside a class. The effects of getters and setters could be leveraged to set permissions to variables. We have learned all the Object Oriented Concepts in Java Programming Language. In our next tutorial, we shall start learning Java Programming Concepts starting with Java Data Types.

- Java File Operations
- Java Date & Time
- Java MySQL
- Java Random
- Java Math