WEEK 7

User Defined Class
(Class Member Accessibility)

Sharmila Mat Yusof

Outline

- Accessibility modifiers
  - `private` modifier
  - Example of `private` modifier
  - `public` modifier
  - Example of `public` modifier
  - `public` modifier and encapsulation issue
- Accessor and Mutator
  - Accessor method
  - Example of accessor method
  - Mutator method
  - Example of mutator method

Learning Objectives

- To describe the effect of `private` and `public` access to data and methods.
- To understand accessor and mutator methods.

Accessibility Modifiers

- Generally, the use of accessibility modifiers is to determine the right access for class, object's data and methods.
- By default, the class, variable and method can be accessed by any class in the same package.
- The two common accessibility modifiers used in a program are:
  - `public`
    - The data and method is visible to any class in any package.
  - `private`
    - The data and method can be accessed only by the declaring class.

Accessibility Modifiers

- `private` modifier is used to enforce information hiding by making instance variable(s) private.
- The instance variable(s) is/are encapsulated to prevent the client program to access it directly.
- Method(s) is/are declared as `private` when it is only to be accessible within the same class.
- We use public methods(i.e. accessor and mutator methods) to read and modify the private data.
- Another `protected` modifier will be discussed later in the inheritance topic.

`private` Modifier: Example

`Student` class[1]

```
1   package student;
2   public class Student {
3       String name;
4       int matricNo;
5       private String grade;
6       int noOfStudent = 0;
7
8       public Student(String studName)
9       {
10          name = studName;
11          noOfStudent++;
12      }
13       public Student(String studName,int matricNum)
14       {
15          name = studName;
16          matricNo = matricNum;
17          noOfStudent++;
18       }
19       public Student(String studName,int matricNum,double mark)
20       {
21          name = studName;
22          matricNo = matricNum;
23          grade = determineGrade(mark);
24          noOfStudent++;
25       }
```

private instance variable - `grade`

`private` Modifier: Example

`Student` class[2]

```
26
27      public int getNoOfStudent()
28      {
29        return noOfStudent;
30      }
31
32      private String determineGrade(double mark)
33      {
34       if (mark > 39)
35           grade = "PASS";
36       else
37           grade = "FAIL";
38       return grade;
39      }
40      public void displayInfo()
41      {
42        System.out.println("Name: "+name);
43        System.out.println("Matric Number: "+matricNo);
44        System.out.println("Grade: "+grade);
45      }
46    }
47
```

private method – `determineGrade`

`private` Modifier: Example

Client class

```
1   package student;
2
3   public class privateModifier {
4     public static void main(String[] args)
5     {
6         Student PGStudent = new Student("Basir",67890,90);
7         System.out.println("Name: "+PGStudent.name);
8         System.out.println("Matric Number: "+PGStudent.matricNo);
          System.out.println("Grade: "+PGStudent.grade);
          System.out.println("Grade: "+PGStudent.determineGrade(90));
11    }
12  }
13
```

grade has private access in Student

(Alt-Enter shows hints)

determineGrade(double) has private access in Student

(Alt-Enter shows hints)

The **PGStudent** object cannot access variable **`grade`** and method **`determineGrade`** as they have been declared as private.

public Modifier: Example

Student class[1]

```
1   package student;
2   public class Student {
3       String name;
4       int matricNo;
5       public String grade;
6       int noOfStudent = 0;
7
8       public Student(String studName)
9       {
10          name = studName;
11          noOfStudent++;
12      }
13       public Student(String studName,int matricNum)
14       {
15          name = studName;
16          matricNo = matricNum;
17          noOfStudent++;
18       }
19       public Student(String studName,int matricNum,double mark)
20       {
21          name = studName;
22          matricNo = matricNum;
            grade = determineGrade(mark);
24          noOfStudent++;
25       }
```

public instance variable - grade

public Modifier: Example

Student class[2]

```
26
27      public int getNoOfStudent()
28      {
29        return noOfStudent;
30      }
31
32      public String determineGrade(double mark)
33      {
34       if (mark > 39)
35           grade = "PASS";
36       else
37           grade = "FAIL";
38       return grade;
39      }
40      public void displayInfo()
41      {
42        System.out.println("Name: "+name);
43        System.out.println("Matric Number: "+matricNo);
44        System.out.println("Grade: "+grade);
45      }
46  }
47
```

public method
- determineGrade

public Modifier: Example

Client class

```
1   package student;
2   public class publicModifier {
3     public static void main(String[] args)
4     {
5       Student PGStudent = new Student("Basir",67890,90);
6       System.out.println("Name: "+PGStudent.name);
7       System.out.println("Matric Number: "+PGStudent.matricNo);
8       System.out.println("Grade: "+PGStudent.grade);
9       System.out.println("Grade: "+PGStudent.determineGrade(90));
10    }
11  }
12
```

access public instance variable - `grade`

invoke public method - `determineGrade`

Output

```
run:
Name: Basir
Matric Number: 67890
Grade: PASS
Grade: PASS
BUILD SUCCESSFUL (total time: 2 seconds)
```

`public` Modifier

- As shown in from previous slides, **`public`** instance variable and method can be accessed in its own class and other classes.

- Variable(s) and method(s) that have been declared as **`public`** have unlimited access control but could violate the object encapsulation principle.

public Modifier: Encapsulation issue

Consider the following class and output:

Client class

```
1   package student;
2
3   public class encapsulationTest {
4     public static void main(String[] args)
5     {
6         Student PGStudent = new Student("Basir",67890,90);
7       System.out.println("Name: "+PGStudent.name);
8       System.out.println("Matric Number: "+PGStudent.matricNo);
9       System.out.println("Grade: "+PGStudent.grade);
10      System.out.println("Grade: "+PGStudent.determineGrade(30));
11    }
12
13  }
14
```

Output

```
run:
Name: Basir
Matric Number: 67890
Grade: PASS
Grade: FAIL
BUILD SUCCESSFUL (total time: 2 seconds)
```

`public` Modifier: Encapsulation Issue

- From previous slide, the `grade` variable has been changed from `PASS` to `FAIL` due to the change of `mark` value in `line 10`.
- Code in `line 10` is valid since the method `determineGrade` is declared as **`public`** and `grade` variable is not encapsulated in `Student` object.
- In the example, the `main()` method can directly change the `Student` class members.
- Thus, `determineGrade method` should be encapsulated in a class by making it **`private`**.

Accessor and Mutator

- If instance variable is `private`, a class usually provides services to access and modify data values.
- An accessor method returns the current value of a variable.
- A mutator method changes the value of a variable.
- The names of accessor and mutator methods take the form getX and setX, respectively, where X is the name of the variable.
- They are sometimes called “getters” and “setters”.

Accessor

- Client program cannot directly access **`private`** instance variables, so classes provide **`public`** accessor methods for access purpose.
- Syntax:

```
public returnType getInstanceVariable(  )
   {
     return instanceVariable;
   }
```

where

`returnType` is the same data type as the instanceVariable data type.

Accessor Method: Example

`Student` class

```
1   package student;
2   public class Student {
3       String name;
4       int matricNo;
5       private String grade;
6       int noOfStudent = 0;
7
```

private instance variable - `grade`

```
35      public String getName()
36      {
37        return name;
38      }
39
40      public int getMatricNo()
41      {
42        return matricNo;
43      }
44      public String getGrade()
45      {
46        return grade;
47      }
48      public int getNoOfStudent()
49      {
50        return noOfStudent;
51      }
```

public accessor method to access private instance variable `grade`

Accessor Method: Example

Client class

```
1   package student;
2   public class accessorMethod {
3       public static void main(String[] args)
4       {
5           Student PGStudent = new Student("Ali",12345,85);
6           System.out.println("Name: "+PGStudent.name);
7           System.out.println("Matric Number: "+PGStudent.matricNo);
8           System.out.println("Grade: "+PGStudent.getGrade());
9       }
10  }
```

invoke accessor method

Output

```
run:
Name: Ali
Matric Number: 12345
Grade: PASS
BUILD SUCCESSFUL (total time: 2 seconds)
```

Mutator Method

- Mutator method allows client program to change the values of instance variables.
- Syntax:

```
public returnType setInstanceVariable(dataType newValue)
    {
      // assign newValue to instance variable
    }
```

where

returnType is **void**.

Mutator Method: Example

`Student` class

```
1   package student;
2   public class Student {
3      String name;
4      private int matricNo;
5      private String grade;
6      int noOfStudent = 0;
7
```

private instance variable - `matricNo`

```
26      public void setName(String newName)
27      {
28           name = newName;
29      }
30      public void setMatricNo(int newMatricNo)
31      {
32           matricNo = newMatricNo;
33      }
34
35      public String getName()
36      {
37        return name;
38      }
```

public mutator method to access private instance variable `grade`

Mutator Method: Example

Client class

```
1   package student;
2   public class mutatorMethod {
3       public static void main(String[] args)
4       {
5           Student PGStudent = new Student("Ali",12345,85);
6           System.out.println("Name: "+PGStudent.name);
7           PGStudent.setMatricNo(67890);
8           System.out.println("Matric Number: "+PGStudent.getMatricNo());
9           System.out.println("Grade: "+PGStudent.getGrade());
10      }
11
12  }
```

invoke mutator method

Output

```
run:
Name: Ali
Matric Number: 67890
Grade: PASS
BUILD SUCCESSFUL (total time: 2 seconds)
```

Summary

- There are two common accessibility modifiers used in a program:
  - `public` modifier – The data and method is visible to any class in any package.
  - `private` modifier - The data and method can be accessed only by the declaring class.
- `public` modifier enables class members to be accessible everywhere and resulted in its members not encapsulated in the class.
- If instance variable is `private`, a class usually provides services to access (accessor) and modify data values (mutator).
- An accessor method returns the current value of a variable while a mutator method changes the value of a variable.