ACCESS CONTROLS

By
M. BABY ANUSHA,
ASST.PROF IN CSE DEPT.,
RGUKT,NUZVID

INTRODUCTION

- Access control is a mechanism, an attribute of encapsulation which restricts the access of certain members of a class to specific parts of a program. Access to members of a class can be controlled using the *access modifiers.*
- There are four access modifiers in Java. They are:
  - public
  - protected
  - default
  - private

CONTINUED…

- If the member (variable or method) is not marked as either *public* or *protected* or *private*, the access modifier for that member will be *default*. We can apply access modifiers to classes also.
- Among the four access modifiers, *private* is the most restrictive access modifier and *public* is the least restrictive access modifier.
- Syntax for declaring a access modifier is shown below:

access-modifier data-type variable-name;

CONTINUED…

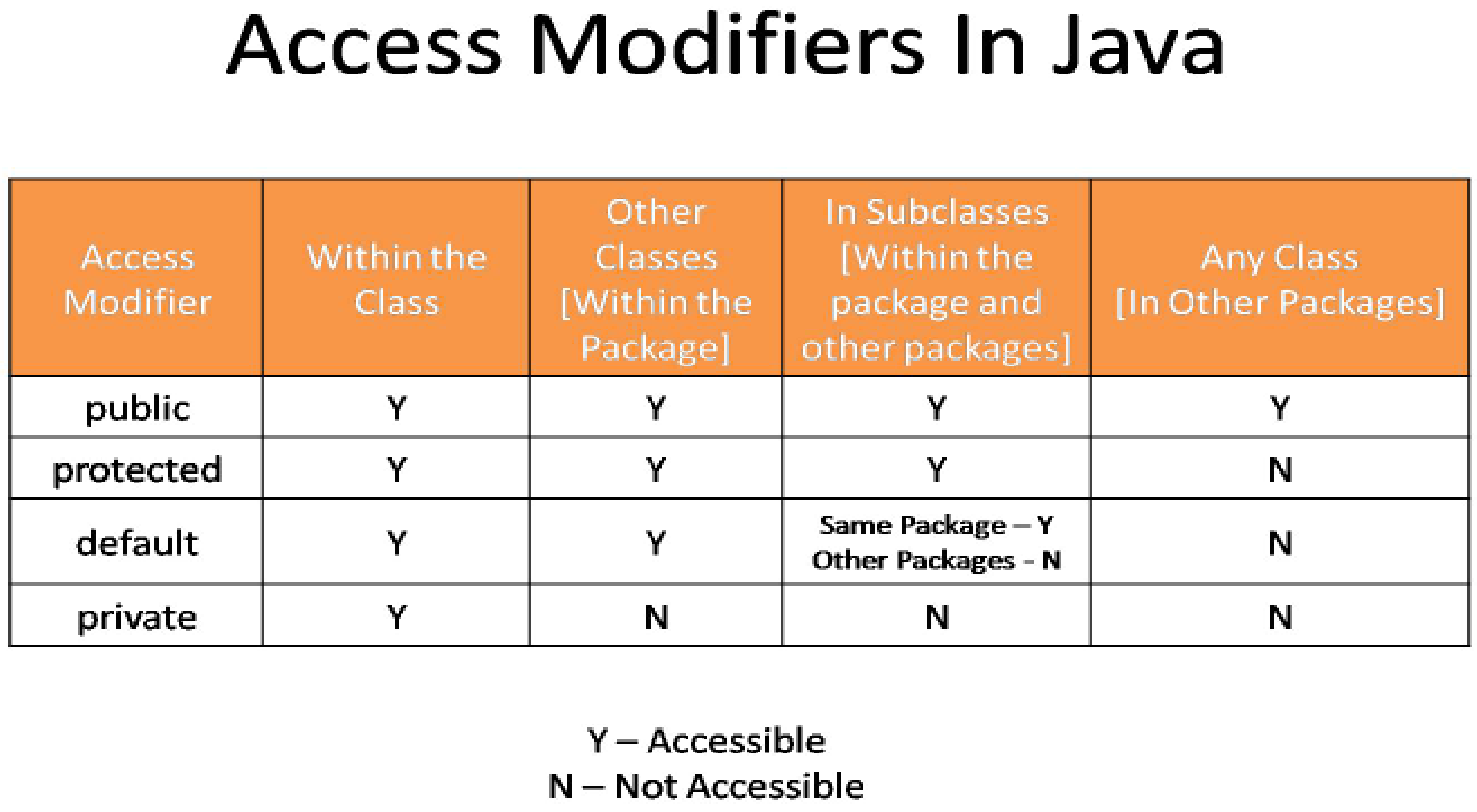
- Example for declaring a *private* integer variable is shown below:

  private int side;

- In a similar way we can apply access modifiers to methods or classes although *private* classes are less common.

CONTINUED…

- Accessibility restrictions of the four access modifiers is as shown below:



Access Modifiers In Java

Access Modifier	Within the Class	Other Classes [Within the Package]	In Subclasses [Within the package and other packages]	Any Class [In Other Packages]
public	Y	Y	Y	Y
protected	Y	Y	Y	N
default	Y	Y	Same Package – Y Other Packages - N	N
private	Y	N	N	N

Y – Accessible
N – Not Accessible

WHAT IS THE USE OF ACCESS MODIFIERS?

- As I had said above, access modifiers are used to restrict the access of members of a class, in particular data members (fields).
- Let me explain this through Java code. Let's consider an employee class as shown below:

```
class Employee
{
int  empid;
String  empname;
float  salary;
//Methods which operate on above data
   members
}
```

CONTINUED…

- By looking at the above code we can say that the access modifier for all the three data members is *default*.
- As members with *default* (also known as **Package Private** or **no modifier**) access modifier are accessible throughout the package (in other classes), a programmer might, by mistake, try to make an employee's salary negative as shown below:

```
Employee e1 = new Employee();
e1.salary = -1000.00;
```

CONTINUED…

- Although above code is syntactically correct, it is logically incorrect.
- To prevent such things to happen, in general, all the data members are declared *private* and are accessible only through *public* methods.
- So, we can modify our *Employee* class as shown below:

SAMPLE PROGRAM :

```
Class Employee
{
   private int empid;
   private String emp;
   private float salary;
   public void setSalary(float sal)
   {
      if(sal < 0)
      {
         System.out.println("Salary cannot be negative");
      }
      else
      {
         salary = sal;
      }
     }
//Other methods
}
```

CONTINUED…

- By looking at the above code, we can say that one can access the *salary* field only through *setSalary()* method.
- Now, we can set the salary of an employee as shown below:

Employee e1 = new Employee();

e1.setSalary(-1000.00); //Gives error as salary cannot be negative

e1.setSalary(2000.00); //salary of e1 will be assigned

2000.00ary of   an employee as shown below:

DEFAULT ACCESS MODIFIER

- When we do not mention any access modifier, it is called default access modifier.
- The scope of this modifier is limited to the package only. This means that if we have a class with the default access modifier in a package, only those classes that are in this package can access this class.
- No other class outside this package can access this class. Similarly, if we have a default method or data member in a class, it would not be visible in the class of another package.
- Lets see an example to understand this:

DEFAULT ACCESS MODIFIER EXAMPLE IN JAVA

- To understand this example, you must have the knowledge of packages in java.
- In this example we have two classes, Test class is trying to access the default method of Addition class, since class Test belongs to a different package, this program would throw compilation error, because the scope of default modifier is limited to the same package in which it is declared.

DEFAULT ACCESS MODIFIER EXAMPLE IN JAVA

Addition.java

```
package abcpackage;
 public class Addition
{
/* Since we didn't mention any access modifier here,
   it would be considered as default. */
   int addTwoNumbers(int a, int b)
   {
      return a+b;
   }
}
```

DEFAULT ACCESS MODIFIER EXAMPLE IN JAVA

```
package xyzpackage;

 /* We are importing the abcpackage but still we will get
   error because the  class we are trying to use has
   default access  modifier. */


import abcpackage.*;
 public class Test
{
    public static void main(String args[])
    {
    Addition obj = new Addition(); /* It will throw error
   because we are trying to access the default method in
   another package */
   obj.addTwoNumbers(10, 21);
    }
}
```

DEFAULT ACCESS MODIFIER EXAMPLE IN JAVA

Output:

```
Exception in thread "main" java.lang.Error: 
Unresolved compilation problem: The method 
addTwoNumbers(int, int) from the type 
Addition is not visible at xyzpackage.Test.
main(Test.java:12)
```

PRIVATE ACCESS MODIFIER

- The scope of private modifier is limited to the class only.
- Private Data members and methods are only accessible within the class
- Class and Interface cannot be declared as private
- If a class has private constructor then you cannot create the object of that class from outside of the class.
- Let’s see an example to understand this:

PRIVATE ACCESS MODIFIER EXAMPLE IN JAVA

- This example throws compilation error because we are trying to access the private data member and method of class ABC in the class Example.
- The private data member and method are only accessible within the class.

PRIVATE ACCESS MODIFIER EXAMPLE IN JAVA

```
class ABC
{
   private double num = 100;
   private int square(int a)
   {
      return a*a;
    }
 }
 public class Example{
   public static void main(String args[])
   {
   ABC obj = new ABC();
   System.out.println(obj.num); System.out.println(obj.
  square(10));
 } }
```

Output:

Compile - time error

PROTECTED ACCESS MODIFIER

- Protected data member and method are only accessible by the classes of the same package and the subclasses present in any package.
- You can also say that the protected access modifier is similar to default access modifier with one exception that it has visibility in sub classes.
- Classes cannot be declared protected. This access modifier is generally used in a parent child relationship.

PROTECTED ACCESS MODIFIER EXAMPLE IN JAVA

- In this example the class Test which is present in another package is able to call the addTwoNumbers() method, which is declared protected.
- This is because the Test class extends class Addition and the protected modifier allows the access of protected members in subclasses (in any packages).

PROTECTED ACCESS MODIFIER EXAMPLE IN JAVA

Addition.java

```
package abcpackage;

public class Addition
 {
   protected int addTwoNumbers(int a, int b)
   {
      return a+b;
   }
}
```

PROTECTED ACCESS MODIFIER EXAMPLE IN JAVA

Test.java

```
package xyzpackage;
 import abcpackage.*;
class Test extends Addition
{
   public static void main(String args[])
   {
   Test obj = new Test(); System.out.println(obj.
   addTwoNumbers(11,22)); }
}
```

Output:

```
33
```

PUBLIC ACCESS MODIFIER

- The members, methods and classes that are declared public can be accessed from anywhere. This modifier doesn’t put any restriction on the access.

public access modifier example in java

- Lets take the same example that we have seen above but this time the method addTwoNumbers() has public modifier and class Test is able to access this method without even extending the Addition class.
- This is because public modifier has visibility everywhere.

PUBLIC ACCESS MODIFIER EXAMPLE IN JAVA

Addition.java

```
package abcpackage;
 public class Addition
 {
   public int addTwoNumbers(int a, int b)
   {
      return a+b;
   }
 }
```

PUBLIC ACCESS MODIFIER EXAMPLE IN JAVA

Test.java

```
package xyzpackage;
 import abcpackage.*;
class Test
{
 public static void main(String args[])
{
 Addition obj = new Addition(); System.out.
   println(obj.addTwoNumbers(100,1));
}
}
```

Output:

```
101
```

Thank you!
shutterstock.com · 567687052