SNS COLLEGE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTION)
COIMBATORE – 35
DEPARTMENT OF COMPUTER SIENCE AND ENGINEERING

UNIT 3

Java protected keyword

A Java protected keyword is an access modifier. It can be assigned to variables, methods, constructors and inner classes.

Points to remember

- The protected access modifier is accessible within the package. However, it can also accessible outside the package but through inheritance only.
- We can't assign protected to outer class and interface.
- If you make any constructor protected, you cannot create the instance of that class from outside the package.
- If you are overriding any method, overridden method (i.e., declared in the subclass) must not be more restrictive.
- According to the previous point, if you assign protected to any method or variable, that method or variable can be overridden to sub-class using public or protected access modifier only.

Examples of protected keyword

Example 1

Let's see an example to determine whether the protected variable is accessible or not outside the package.

```
1.      //save by A.java
2.      package com.java;
3.
4.      public class A {
5.
6.       protected String msg="Try to access the protected variable outside the package";
7.
```

```
8.          }
9.          //save by ProtectedExample1.java
10.         package com.javatpoint;
11.         import com.java.A;
12.
13.         public class ProtectedExample1 {
14.         public static void main(String[] args) {
15.             A a=new A();
16.            System.out.println(a.msg);
17.
18.         }
19.         }
```

Output:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        The field A.msg is not visible
```

Example 2

Let's see an example to determine whether the protected variable is accessible or not outside the class and within the package.

```
1.          class A {
2.
3.           protected String msg="Try to access the protected variable outside the class within th
e package";
4.
5.          }
6.
7.          public class ProtectedExample2 {
8.          public static void main(String[] args) {
9.              A a=new A();
10.            System.out.println(a.msg);
11.
12.         }
13.         }
```

Output:

```
Try to access the protected variable outside the class within the package
```

Example 3

Let's see an example to determine whether the protected method is accessible or not outside the package.

```
1.  //save by A.java
2.  package com.java;
3.
4.  public class A {
5.
6.  protected void msg()
7.   {
8.      System.out.println("Try to access the protected method outside the package ");
9.   }
10.
11. }
12. //save by ProtectedExample3.java
13. package com.javatpoint;
14. import com.java.A;
15.
16. public class ProtectedExample3 {
17. public static void main(String[] args) {
18.    A a=new A();
19.   a.msg();
20.
21. }
22. }
```

Output:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        The method msg() from the type A is not visible
```

Example 4

Let's see an example to determine whether the protected method is accessible or not outside the package using inheritance.

```
1.  //save by A.java
2.  package com.java;
```

```
3.
4.          public class A {
5.
6.          protected void msg()
7.           {
8.               System.out.println("Try to access the protected method outside the package using in
heritance");
9.           }
10.
11.         }
12.         //save by ProtectedExample4.java
13.         package com.javatpoint;
14.         import com.java.A;
15.
16.         public class ProtectedExample4 extends A {
17.         public static void main(String[] args) {
18.            ProtectedExample4 a=new ProtectedExample4();
19.           a.msg();
20.
21.         }
22.         }
```

Output:

```
Try to access the protected method outside the package using inheritance
```

Example 5

Let's see an example to determine whether we assign protected to the outer class.

```
1.          protected class ProtectedExample5 {
2.          void display()
3.          {
4.             System.out.println("Try to access outer protected class");
5.          }
6.          public static void main(String[] args) {
7.             ProtectedExample5 p=new ProtectedExample5();
8.             p.display();
9.
```

```
10.         }
11.         }
```

Output:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
```

Example 6

Let's see an example to determine whether we create the instance of protected constructor from outside the class.

```
1.          //save by A.java
2.
3.          package com.java;
4.
5.          public class A
6.          {
7.           String msg;
8.          protected A(String msg)
9.           {
10.             this.msg=msg;
11.          }
12.         public void display()
13.         {
14.             System.out.println(msg);
15.         }
16.
17.         }
18.         //save by ProtectedExample6.java
19.         package com.javatpoint;
20.         import com.java.A;
21.
22.         public class ProtectedExample6 {
23.         public static void main(String[] args) {
24.             A a=new A("Try to create the instance of protected constructor outside the package"
    );
25.             a.display();
26.
```

```
27.
28.         }
29.         }
```

Output:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        The constructor A(String) is not visibles
```

Example 7

Let's see an example to determine whether the protected method is overridden to sub-class using protected access modifier.

```
1.          //save by A.java
2.
3.          class A
4.          {
5.             protected void msg()
6.             {
7.                System.out.println("Try it");
8.             }
9.          }
10.         //save by ProtectedExample7.java
11.         class ProtectedExample7 extends A {
12.            protected void msg()
13.         {
14.            System.out.println("Try to access the overridden method");
15.         }
16.         public static void main(String[] args) {
17.            ProtectedExample7 p=new ProtectedExample7();
18.            p.msg();
19.
20.         }
21.         }
```

Output:

```
Try to access the overridden method
```

Example 8

Let's see an example to determine whether the protected method is overridden to sub-class using private access modifier.

```
1.  class A
2.  {
3.     protected void msg()
4.     {
5.        System.out.println("Try it");
6.     }
7.  }
8.
9.  class ProtectedExample8 extends A {
10.    private void msg()
11. {
12.    System.out.println("Try to access the overridden method");
13. }
14.    public static void main(String[] args) {
15.    ProtectedExample8 p=new ProtectedExample8();
16.    p.msg();
17.
18. }
19. }
```

Output:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        Cannot reduce the visibility of the inherited method from A
```

Example 9

Let's see an example to determine whether the protected method is overridden to sub-class using default access modifier.

```
1.  class A
2.  {
3.     protected void msg()
4.     {
5.        System.out.println("Try it");
```

```
6.          }
7.      }
8.
9.      class ProtectedExample8 extends A {
10.         void msg()
11.     {
12.         System.out.println("Try to access the overridden method");
13.     }
14.     public static void main(String[] args) {
15.         ProtectedExample9 p=new ProtectedExample9();
16.         p.msg();
17.
18.     }
19.     }
```

Output:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        Cannot reduce the visibility of the inherited method from A
```

Example 10

Let's see an example to determine whether the protected method is overridden to sub-class using public access modifier.

```
1.      class A
2.      {
3.          protected void msg()
4.          {
5.              System.out.println("Try it");
6.          }
7.      }
8.
9.      class ProtectedExample10 extends A {
10.         public void msg()
11.     {
12.         System.out.println("Try to access the overridden method");
13.     }
14.     public static void main(String[] args) {
15.         ProtectedExample10 p=new ProtectedExample10();
```

```
16.             p.msg();
17.
18.         }
19.         }
```

Output:

```
Try to access the overridden method
```