Private access: The private access modifier is accessible only within class.

|  |
| --- |
| //program for private access  class A  {  private int data=30;  private void msg()  {  System.out.println("this is private method");  }  }  public class simple  {  public static void main(String args[])  {  A obj=new A();  System.out.println(obj.data);  obj.msg();  }  } |

Output

|  |
| --- |
| E:\javaprgs\accessmodify>javac simple.java  simple.java:15: data has private access in A  System.out.println(obj.data);  ^  simple.java:16: msg() has private access in A  obj.msg();  ^  2 errors |

Role of Private Constructor

|  |
| --- |
| If you make any class constructor private, you cannot create the instance of that class from outside the class. For example: |

|  |
| --- |
| /program for private access constructor  class A1  {  private A1()  {  System.out.println("welcome");  }  void msg()  {  System.out.println("this is default method");  }  }  public class privateconstructor  {  public static void main(String args[])  {  A1 obj=new A1();  }  } |

Output

|  |
| --- |
| E:\javaprgs\accessmodify>javac privateconstructor.java  privateconstructor.java:18: A1() has private access in A1  A1 obj=new A1();  ^  privateconstructor.java:19: cannot find symbol  symbol : variable data  location: class A1  System.out.println(obj.data);  ^  2 errors |

default access modifier

|  |
| --- |
| If you don't use any modifier, it is treated as **default** bydefault. The default modifier is accessible only within package. |

Example of default access modifier

|  |
| --- |
| In this example, two packages pack and mypack are created. We are accessing the A class from outside its package, since A class is not public, so it cannot be accessed from outside the package. |

Program name: A.java package name: pack

|  |
| --- |
| package pack;  class A  {  void msg()  {  System.out.println("welcome");  }  } |

Program name: B.java package name : mypack

|  |
| --- |
| package mypack;  import pack.\*;  class B  {  public static void main(String args[])  {  A obj=new A();  obj.msg();  }  } |

output

|  |
| --- |
| E:\javaprgs\accessmodify>javac -cp . mypack\B.java  mypack\B.java:7: pack.A is not public in pack; cannot be accessed from outside p  ackage  A obj=new A();  ^  mypack\B.java:7: pack.A is not public in pack; cannot be accessed from outside p  ackage  A obj=new A();  ^  2 errors |

In the above example, the scope of class A and its method msg() is default so it cannot be accessed from outside the package.

### protected access modifier

The **protected access modifier** is accessible within package and outside the package but through inheritance only.

The protected access modifier can be applied on the data member, method and constructor. It can't be applied on the class.

### Example of protected access modifier

In this example, we have created the two packages pack and mypack. The A class of pack package is public, so can be accessed from outside the package. But msg method of this package is declared as protected, so it can be accessed from outside the class only through inheritance.

Program name : A.java package name:pack

|  |
| --- |
| package pack;  public class A  {  protected void msg()  {  System.out.println("welcome");  }  } |

Program name : B.java

|  |
| --- |
| package mypack;  import pack.\*;  class B extends A  {  public static void main(String args[])  {  B obj=new B();  obj.msg();  }  } |

Output

|  |
| --- |
| E:\javaprgs\accessmodify\mypack>cd..  E:\javaprgs\accessmodify>javac -cp . mypack\B.java  E:\javaprgs\accessmodify>java -cp . mypack.B  welcome |

Suppose in the above program if class B is not extending and trying to call protected method

|  |
| --- |
| package mypack;  import pack.\*;  class B  {  public static void main(String args[])  {  A obj=new A();  obj.msg();  }  } |

f

|  |
| --- |
| E:\javaprgs\accessmodify>javac -cp . mypack\B.java  mypack\B.java:8: msg() has protected access in pack.A  obj.msg();  ^  1 error |



