**Access Modifiers in Java**

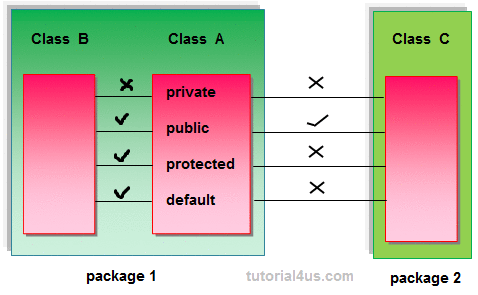
**Access modifiers**are those which are applied before data members or methods of a class. These are used to where to access and where not to access the data members or methods. In Java programming these are classified into four types:

* Private
* Default (not a keyword)
* Protected
* Public

**Note:**Default is not a keyword (like public, private, protected are keyword)

If we are not using private, protected and public keywords, then JVM is by default taking as default access modifiers.

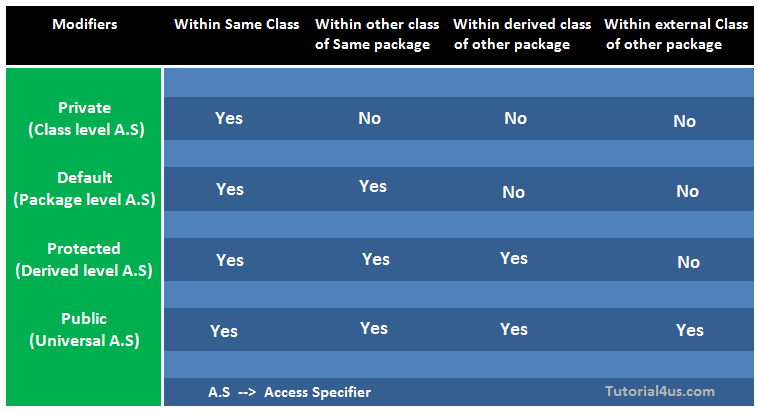
Access modifiers are always used for, how to reuse the features within the package and access the package between class to class, interface to interface and interface to a class. Access modifiers provide features accessing and controlling mechanism among the classes and interfaces.



**Note:**Protected members of the class are accessible within the same class and another class of same package and also accessible in inherited class of another package.

**Rules for access modifiers:**

The following diagram gives rules for Access modifiers.



**private:**Private members of class in not accessible anywhere in program these are only accessible within the class. Private are also called class level access modifiers.

**Example**

**class** Hello

{

**private** **int** a=20;

**private** **void** show()

{

System.**out**.println("Hello java");

}

}

**public** **class** Demo

{

**public** **static** **void** main(String args[])

{

Hello obj=**new** Hello();

System.**out**.println(obj.a); //Compile Time Error, you can't access private data

obj.show(); //Compile Time Error, you can't access private methods

}

}

**public:**Public members of any class are accessible anywhere in the program in the same class and outside of class, within the same package and outside of the package. Public are also called universal access modifiers.

**Example**

**class** Hello

{

**public** **int** a=20;

**public** **void** show()

{

System.**out**.println("Hello java");

}

}

**public** **class** Demo

{

**public** **static** **void** main(String args[])

{

Hello obj=**new** Hello();

System.**out**.println(obj.a);

obj.show();

}

}

**Output**

20

Hello Java

**protected:**Protected members of the class are accessible within the same class and another class of the same package and also accessible in inherited class of another package. Protected are also called derived level access modifiers.

In below the example we have created two packages pack1 and pack2. In pack1, class A is public so we can access this class outside of pack1 but method show is declared as a protected so it is only accessible outside of package pack1 only through inheritance.

**Example**

// save A.java

**package** pack1;

**public** **class** A

{

**protected** **void** show()

{

System.**out**.println("Hello Java");

}

}

//save B.java

**package** pack2;

**import** pack1.\*;

**class** B **extends** A

{

**public** **static** **void** main(String args[]){

B obj = **new** B();

obj.show();

}

}

**Output**

Hello Java

**default:**Default members of the class are accessible only within the same class and another class of the same package. The default are also called package level access modifiers.

**Example**

//save by A.java

**package** pack;

**class** A

{

**void** show()

{

System.**out**.println("Hello Java");

}

}

//save by B.java

**package** pack2;

**import** pack1.\*;

**class** B

{

**public** **static** **void** main(String args[])

{

A obj = **new** A(); //Compile Time Error, can't access outside the package

obj.show(); //Compile Time Error, can't access outside the package

}

}

**Output**

Hello Java

**Note:** private access modifier is also known as native access modifier, default access modifier is also known as package access modifier, protected access modifier is also known as an inherited access modifier, public access modifier is also known as universal access modifier.