Class classname{

Specific requirement / B. logic

}

Interface interface name{

Common requirement

}

A **java package** is a group of similar types of classes, interfaces and sub-packages.

Package in java can be categorized in two form, built-in package and user-defined package.

There are many built-in packages such as java, lang, awt, javax, swing, net, io, util, sql etc.

## Advantage of Java Package

1) Java package is used to categorize the classes and interfaces so that they can be easily maintained.

2) Java package provides access protection.

3) Java package removes naming collision.

The **package keyword** is used to create a package in java.

1. //save as Simple.java
2. **package** mypack;
3. **public** **class** Simple{
4. **public** **static** **void** main(String args[]){
5. System.out.println("Welcome to package");
6. }
7. }

javac -d . Simple.java

|  |
| --- |
| **To Compile:** javac -d . Simple.java |
| **To Run:** java mypack.Simple |

------------------

There are two types of modifiers in Java: **access modifiers** and **non-access modifiers**.

The access modifiers in Java specifies the accessibility or scope of a field, method, constructor, or class. We can change the access level of fields, constructors, methods, and class by applying the access modifier on it.

There are four types of Java access modifiers:

1. **Private**: The access level of a private modifier is only within the class. It cannot be accessed from outside the class.
2. **Default**: The access level of a default modifier is only within the package. It cannot be accessed from outside the package. If you do not specify any access level, it will be the default.
3. **Protected**: The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.
4. **Public**: The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.

Java.util.\*;

Understanding Java Access Modifiers

Let's understand the access modifiers in Java by a simple table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Access Modifier** | **within class** | **within package** | **outside package by subclass only** | **outside package** |
| **Private** | Y | N | N | N |
| **Default** | Y | Y | N | N |
| **Protected** | Y | Y | Y | N |
| **Public** | Y | Y | Y | Y |

### Private

The private access modifier is accessible only within the class.

**Simple example of private access modifier**

In this example, we have created two classes A and Simple. A class contains private data member and private method. We are accessing these private members from outside the class, so there is a compile-time error.

**class** A{

**private** **int** data=40;

**private** **void** msg(){

System.out.println("Hello java");

}

}

**public** **class** Simple{

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

   A obj=**new** A();

   System.out.println(obj.data);//Compile Time Error

   obj.msg();//Compile Time Error

   }

}

class Data {

// private variable

private String name;

}

public class Main {

public static void main(String[] main){

// create an object of Data

Data d = new Data();

// access private variable and field from another class

d.name = "Programiz";

}

}

class Data {

private String name;

// getter method

public String getName() {

return this.name;

}

// setter method

public void setName(String name) {

this.name= name;

}

}

public class Main {

public static void main(String[] main){

Data d = new Data();

// access the private variable using the getter and setter

d.setName("Programiz");

System.out.println(d.getName());

}

}

### …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:

**class** A{

**private** A(){}//private constructor

**void** msg(){System.out.println("Hello java");}

}

**public** **class** Simple{

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

   A obj=**new** A();//Compile Time Error

 }

}

#### Note: A class cannot be private or protected except nested class.

### 2) Default

If you don't use any modifier, it is treated as **default** by default. The default modifier is accessible only within package. It cannot be accessed from outside the package. It provides more accessibility than private. But, it is more restrictive than protected, and public.

**Example of default access modifier**

In this example, we have created two packages pack and mypack. 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.

1. //save by A.java
2. **package** pack;
3. **class** A{
4. **void** msg(){
5. System.out.println("Hello");}
6. }
7. //save by B.java
8. **package** mypack;
9. **import** pack.\*;
10. **class** B{
11. **public** **static** **void** main(String args[]){
12. A obj = **new** A();//Compile Time Error
13. obj.msg();//Compile Time Error
14. }
15. }

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

### 3) Protected

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.

It provides more accessibility than the default modifer.

**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.

class Animal {

// protected method

protected void display() {

System.out.println("I am an animal");

}

}

class Dog extends Animal {

public static void main(String[] args) {

// create an object of Dog class

Dog dog = new Dog();

// access protected method

dog.display();

}

}

//save by Jyothi.java

**package** pack;

**public** **class** Jyothi{

**protected** **void** msg(){System.out.println("Hello");}

}

//save by Bhavani .java

**package** mypack;

**import** pack.\*;

**class** Bhavani **extends** Jyothi{

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

   Bhavani  obj = **new** Bhavani  ();

   obj.msg();

  }

}

### 4) Public

The **public access modifier** is accessible everywhere. It has the widest scope among all other modifiers.

**Example of public access modifier**

1. //save by A.java
3. **package** pack;
4. **public** **class** A{
5. **public** **void** msg(){System.out.println("Hello");}
6. }
7. //save by B.java
9. **package** mypack;
10. **import** pack.\*;
12. **class** B{
13. **public** **static** **void** main(String args[]){
14. A obj = **new** A();
15. obj.msg();
16. }
17. }