CHAPTER 8

ENCAPSULATION

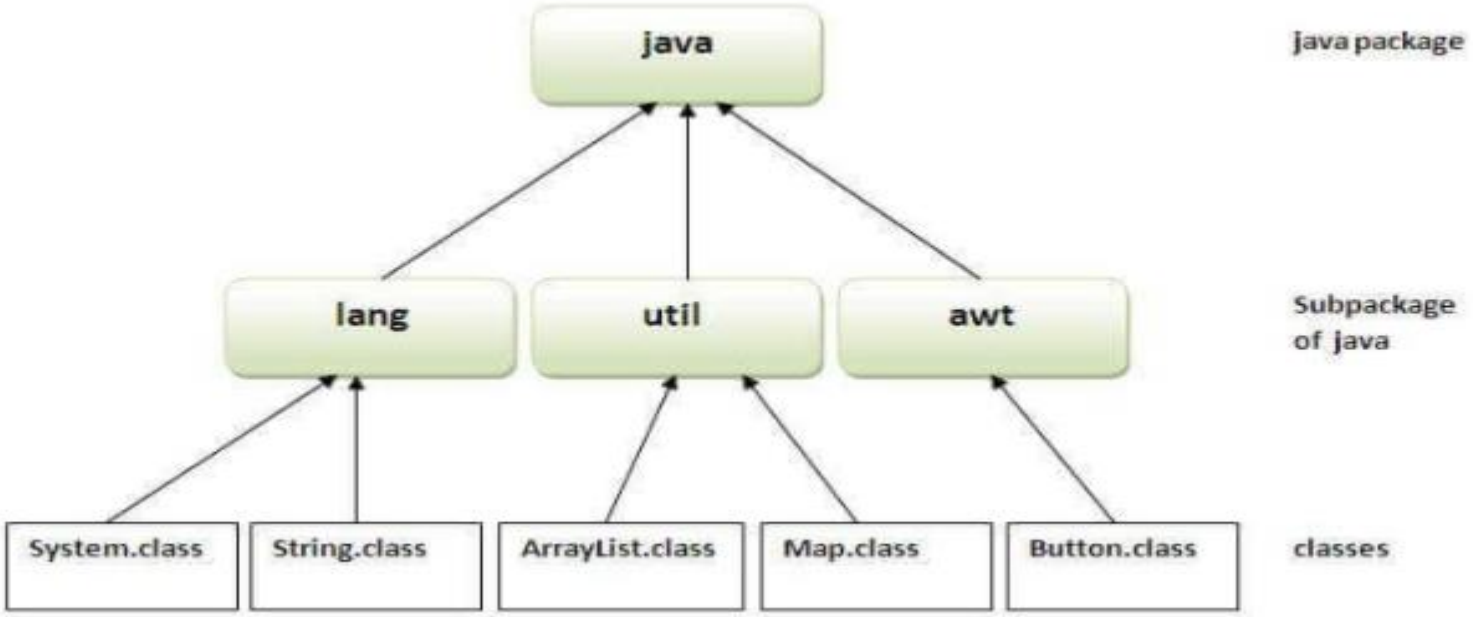
Contents

- Package
- Access Modifiers
- Encapsulation

Package

- 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.**

Example


java
java package
lang
util
awt
Subpackage
of java
System.class
String.class
ArrayList.class
Map.class
Button.class
classes

Example of User defined package

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

```
//save as Simple.java
package mypack;
public class Simple{
 public static void main(String args[]){
    System.out.println("Welcome to package");
   }
}
```

How to compile java package

```
C:\java>javac -d . Simple.java

C:\java>java mypack.Simple
Welcome to package
```

To Compile:

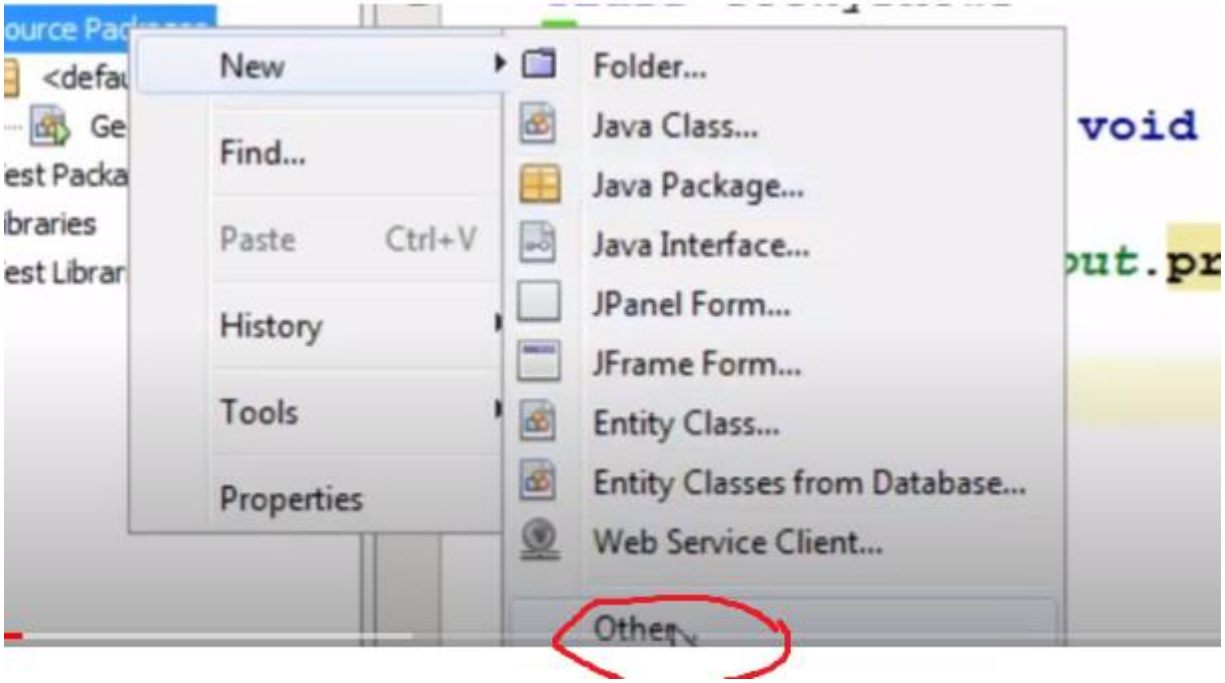
javac -d . Simple.java

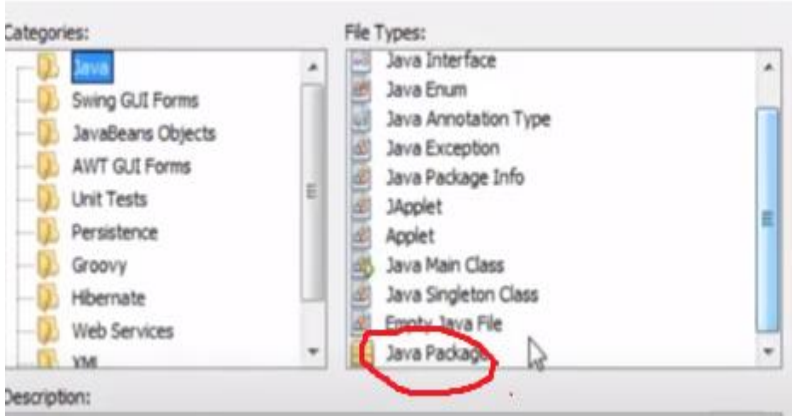
To Run:

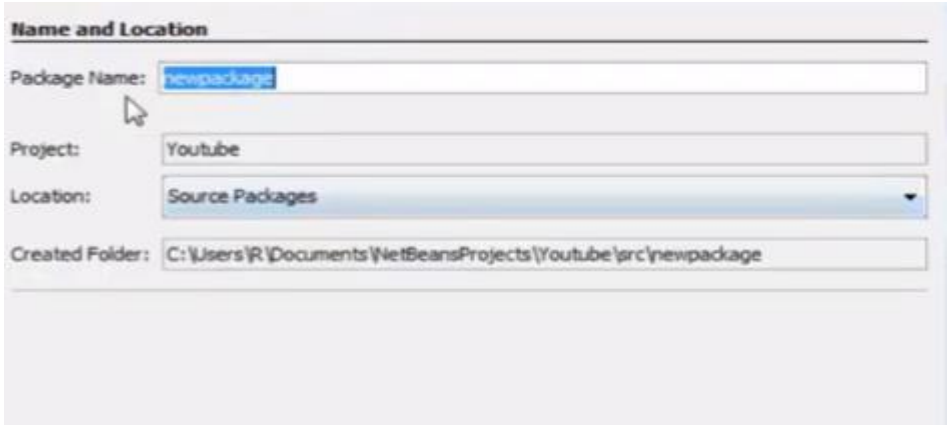
java mypack.Simple

Output

```
Output:Welcome to package
```


New
Find...
Paste Ctrl+V
History
Tools
Properties
Folder...
Java Class...
Java Package...
Java Interface...
JPanel Form...
JFrame Form...
Entity Class...
Entity Classes from Database...
Web Service Client...
Other...


Categories:
File Types:
Java
Swing GUI Forms
JavaBeans Objects
AWT GUI Forms
Unit Tests
Persistence
Groovy
Hibernate
Web Services
Java Interface
Java Enum
Java Annotation Type
Java Exception
Java Package Info
JApplet
Applet
Java Main Class
Java Singleton Class
Java Package
Description:


Name and Location
Package Name: newpackage
Project: Youtube
Location: Source Packages
Created Folder: C:\Users\R\Documents\NetBeansProjects\Youtube\src\newpackage

How to access package from another package?

- There are three ways to access the package.
  1) import package.*;
  2) import package.classname;
  3) fully qualified name.

1) Using packagename.*

- If you use package.*

  then all the classes and interfaces of this package will be accessible.

- The import keyword is used to make the classes and interface of another package accessible to the current package.

Example of package that import the packagename.*

```
//save by A.java
package pack;
public class A{
  public void msg(){System.out.println("Hello");}
}
```

```
//save by B.java
package mypack;
import pack.*;

class B{
  public static void main(String args[]){
   A obj = new A();
   obj.msg();
  }
}
```

```
Output:Hello
```

2) Using packagename.classname

- If you import package.classname then only declared class of this package will be accessible.

Example of package by import package.classname

```
//save by A.java

package pack;
public class A{
  public void msg(){System.out.println("Hello");}
}
```

```
//save by B.java
package mypack;
import pack.A;

class B{
  public static void main(String args[]){
   A obj = new A();
   obj.msg();
  }
}
```

```
Output:Hello
```

3) Using fully qualified name

- If you use fully qualified name then only declared class of this package will be accessible.
- Now there is no need to import.
- But you need to use fully qualified name every time when you are accessing the class or interface.

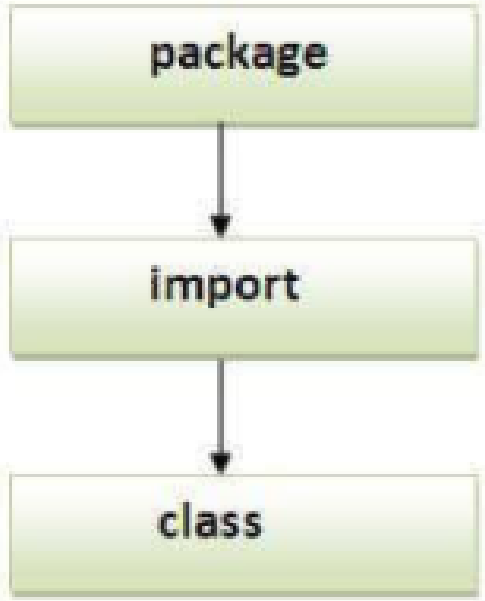
Example of package by import fully qualified name

```
//save by A.java
package pack;
public class A{
  public void msg(){System.out.println("Hello");}
}
```

```
//save by B.java
package mypack;
class B{
  public static void main(String args[]){
   pack.A obj = new pack.A();//using fully qualified name
   obj.msg();
  }
}
```

```
Output:Hello
```

Note: Sequence of the program must be package then import then class.



3.Encapsulation in JAVA

- **Encapsulation in Java** is a *process of wrapping code and data together into* ***a single unit***, for example, **a capsule which** is mixed of several **medicines**.
- We can create a fully encapsulated class in Java by making all the data members of the class private. Now we can **use setter and getter methods to set and get the data** in it.

Advantage of Encapsulation in Java

- It is a way to achieve **data hiding** in Java because other class will not be able to access the private data members.

Example of data encapsulation

File: Student.java

```
//A Java class which is a fully encapsulated class.
//It has a private data member and getter and setter methods.
package com.javatpoint;
public class Student{
//private data member
private String name;
//getter method for name
public String getName(){
return name;
}
//setter method for name
public void setName(String name){
this.name=name
}
}
```

Example of data encapsulation

File: Test.java

```
//A Java class to test the encapsulated class.
package com.javatpoint;
class Test{
public static void main(String[] args){
//creating instance of the encapsulated class
Student s=new Student();
//setting value in the name member
s.setName("vijay");
//getting value of the name member
System.out.println(s.getName());
}
}
```