AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

19CSE204
Object Oriented Paradigm
2-0-3-3

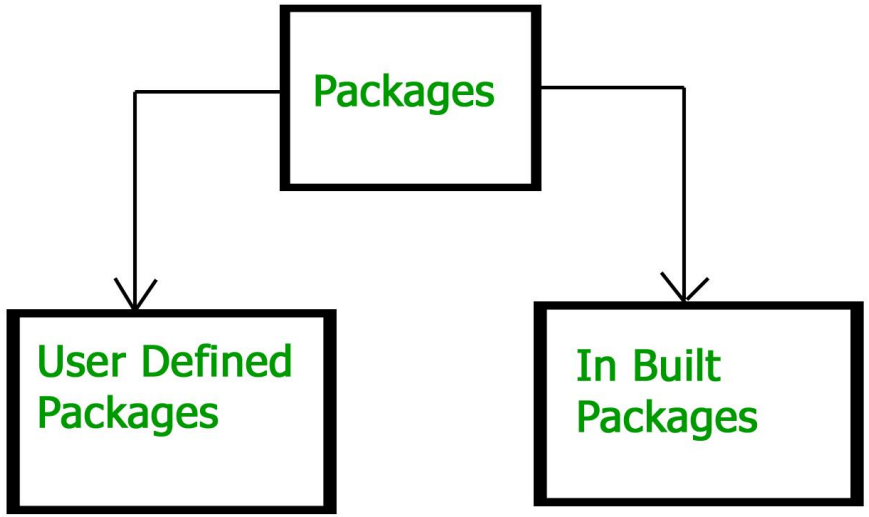
Packages in Java

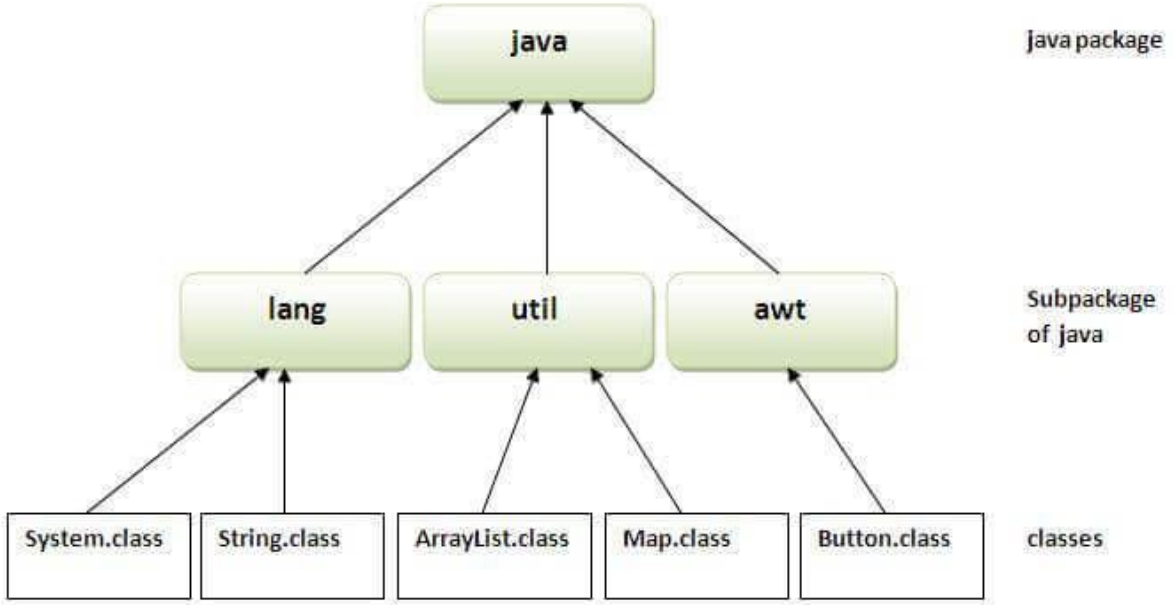
What are packages

- Java provides a **mechanism for partitioning the class name space into more manageable chunks** called **packages.**
- ***Packages* are containers for classes** that are used to keep the class name space compartmentalized.
  - For example, a package allows you to create a class named **List**, which you can store in your own package without concern that it will collide with some other class named **List** stored elsewhere.
- **Packages are stored in a hierarchical manner** and are **explicitly imported** into new class definitions.
- Package is both a **naming and a visibility control mechanism**.
  - You can define classes inside a package that are not accessible by code outside that package.
  - You can also define class members that are only exposed to other members of the same package

Advantage of Java Package

- Java package is used to categorize the classes and interfaces so that they can be easily maintained.
- Java package provides access protection.
- Java package removes naming collision.





Defining a Package

- To create a package ,include a **package** command as the first statement in a Java source file.
- The **package** statement defines a name space in which classes are stored. If you omit the **package** statement, the class names are put into the default package, which has no name

```
package MyPackage;
```

- This statement creates a package of name *MyPackage.Java* uses file system directories to store packages. The .class files for any classes you declare to be part of *MyPackage* must be stored in a directory called *MyPackage*.
- You can create a hierarchy of packages. To do so, simply separate each package name from the one above it by use of a period,

```
package pkg1[.pkg2[.pkg3]];
```

```
package java.awt.image;
```

- needs to be stored in java/awt/image, java\awt\image, or java:awt:image on your UNIX, Windows, or Macintosh file system, respectively

Access Protection

- Java addresses four categories of visibility for class members
  - Subclasses in the same package
  - Non-subclasses in the same package
  - Subclasses in different packages
  - Classes that are neither in the same package nor subclasses
- The three access specifiers, **private**, **public**, and **protected**, provide a variety of ways to produce the many levels of access required by these categories

    - Anything declared **public** can be accessed from anywhere.
    - Anything declared **private** cannot be seen outside of its class.
    - When a member does not have an explicit access specification, it is visible to subclasses as well as to other classes in the same package. This is the **default access**.
    - If you want to allow an element to be seen outside your current package, but only to classes that subclass your class directly, then declare that element **protected**.

	Private	No modifier	Protected	Public
Same class	Yes	Yes	Yes	Yes
Same package subclass	No	Yes	Yes	Yes
Same package non-subclass	No	Yes	Yes	Yes
Different package subclass	No	No	Yes	Yes
Different package non-subclass	No	No	No	Yes

Table 9-1. *Class Member Access*

Import Packages

- The library is divided into **packages and classes**. Meaning you can **either import a single class** (along with its methods and attributes), **or a whole package** that contain all the classes that belong to the specified package.
- To import java package into a class, we need to **use java import keyword** which is used to access package and its classes into the java program.
- **Use import to access built-in and user-defined packages** into your java source file so that your class can refer to a class that is in another package by directly using its name.
- There are **3 different ways** to refer to any class that is present in a different package:
  1. **without import the package**
  2. **import package with specified class**
  3. **import package with all classes**

Without import package

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

```java
//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();
    }
}
```

If you use **fully qualified name to import any class** into your program, then **only that particular class** of the package will be **accessible i**n your program, other classes in the same package will not be accessible.
For this approach, there **is no need to use the import statement**.

Import Packages

- To use a class or a package from the library, you need to use the import keyword

The import key word is used to make the classes of another package accessible to the current package.

import *pkg1*[.*pkg2*].(*classname*|*);

Here, *pkg1* is the name of a top-level package, and *pkg2* is the name of a subordinate package inside the outer package separated by a dot (.).

```
import packagename.Class;   // Import a single class
import packagename.*;   // Import the whole package
```

Import Specific Class of a package

- Package can have many classes but sometimes **we want to access only specific class in our program** in that case, Java allows us to specify class name along with package name. If we use **import packagename.classname** statement
-

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

```java
//save by Test.java
package mypack;
import pack.Demo;
class Test {
    public static void main(String args[]) {
        Demo obj = new Demo();
        obj.msg();
    }
}
```

Import all classes of the package

- If we use **packagename.*** statement, then all the classes and interfaces of this package will be accessible but the classes and interface inside the subpackages will not be available for use.

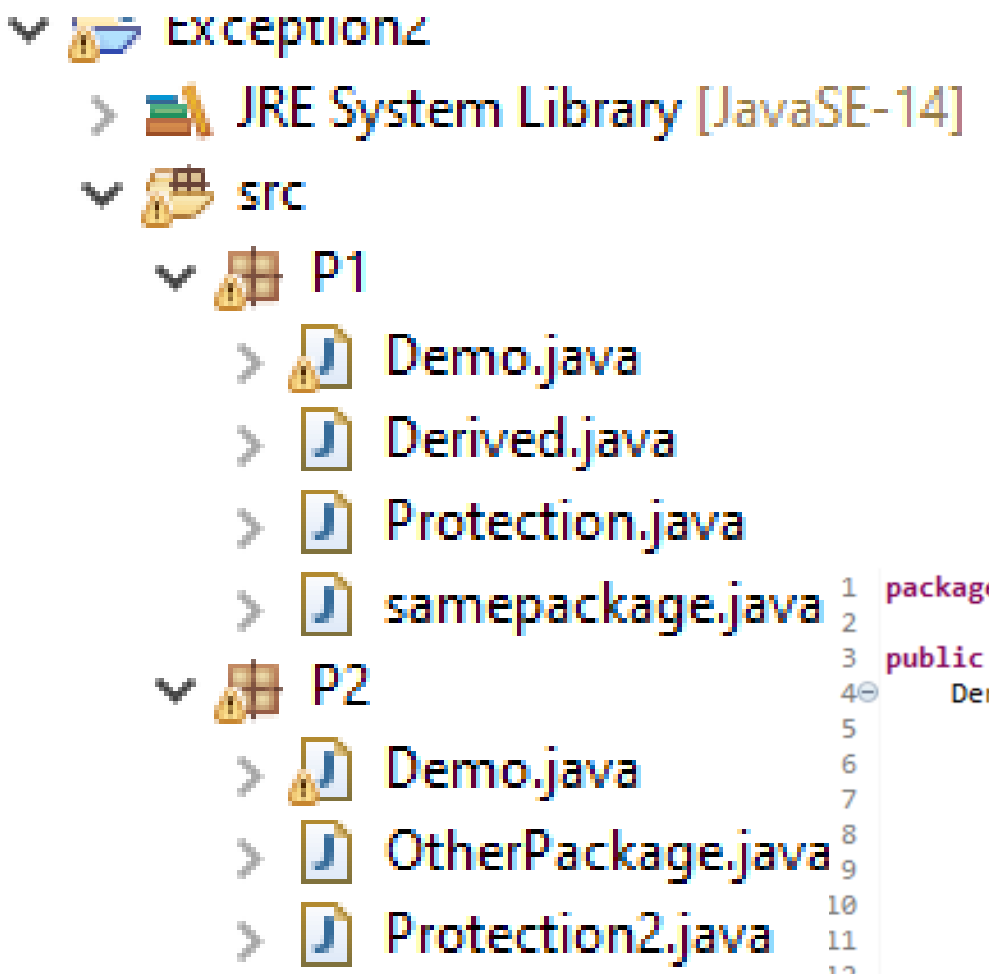
```java
//save by First.java
package learnjava;
public class First{
    public void msg() {
        System.out.println("Hello");
    }
}
```

```java
//save by Second.java
package Java;
import learnjava.*;
class Second {
    public static void main(String args[]) {
        First obj = new First();
        obj.msg();
    }
}
```

	Private	No modifier	Protected	Public
Same class	Yes	Yes	Yes	Yes
Same package subclass	No	Yes	Yes	Yes
Same package non-subclass	No	Yes	Yes	Yes
Different package subclass	No	No	Yes	Yes
Different package non-subclass	No	No	No	Yes

Table 9-1. *Class Member Access*

Example project



Protection.java

```java
package P1;

public class Protection {
    int n = 1;
    private int n_pri = 2;
    protected int n_pro = 3;
    public int n_pub = 4;
    int n_def=5;
    public Protection() {
    System.out.println("base constructor");
    System.out.println("n = " + n);
    System.out.println("n_pri = " + n_pri);
    System.out.println("n_pro = " + n_pro);
    System.out.println("n_pub = " + n_pub);
    System.out.println("n_def = " + n_def);

}
}
```

Derived.java

```java
package P1;

public class Derived extends Protection {
    Derived() {
        System.out.println("derived constructor");
        System.out.println("n = " + n);
        // Below stmt gives error . Private members not accessible in subclasses of same package
        // System.out.println("n_pri = " + n_pri);
        //Public and Protected members of super class accessible in subclass
        System.out.println("n_pri not accessible in subclasses of same package ");
        System.out.println("n_pro = " + n_pro);
        System.out.println("n_pub = " + n_pub);
        System.out.println("n_def = " + n_def);
        }
        }

```

samepackage.java

```java
package P1;

public class samepackage {

    samepackage() {
        Protection p = new Protection();
        System.out.println("same package constructor");
        System.out.println("n = " + p.n);
        // The below statement will give error as private members not accessible
        //same package non subclass
        // System.out.println("n_pri = " + p.n_pri);
        System.out.println("n_pri not accessible in same package non subclass" );
        System.out.println("n_pro = " + p.n_pro);
        System.out.println("n_pub = " + p.n_pub);
        System.out.println("n_def = " + p.n_def);
        }

}
```

Demo.java (Driver class in P1)

```java
package P1;


public class Demo {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Protection ob1 = new Protection();
        Derived ob2 = new Derived();
        samepackage ob3 = new samepackage();
    }

}
```

OUTPUT: DEMO P1

```
base constructor
n = 1
n_pri = 2
n_pro = 3
n_pub = 4
n_def = 5
base constructor
n = 1
n_pri = 2
n_pro = 3
n_pub = 4
n_def = 5
derived constructor
n = 1
n_pri not accessible in subclasses of same package
n_pro = 3
n_pub = 4
n_def = 5
```

```
base constructor
n = 1
n_pri = 2
n_pro = 3
n_pub = 4
n_def = 5
same package constructor
n = 1
n_pri not accessible in same package non
subclass
n_pro = 3
n_pub = 4
n_def = 5
```

Protection2.java

```java
package P2;
import P1.*;
public class Protection2 extends Protection{

    Protection2() {
        System.out.println("derived other package constructor");
        //Below statement will give error as default not accessible in subclass of diff package
        // System.out.println("n_pri = " + n_pri);
        System.out.println("n_pro = " + n_pro);
        System.out.println("n_pub = " + n_pub);
        //Below statement will give error as default not accessible in subclass of diff package
        //System.out.println("n_def = " + n_def);
        }

}
```

OR

Protection2.java

```java
package P2;

public class Protection2 extends P1.Protection{

    Protection2() {
        System.out.println("derived other package constructor");
        //Below statement will give error as default not accessible in subclass of diff package
        // System.out.println("n_pri = " + n_pri);
        System.out.println("n_pro = " + n_pro);
        System.out.println("n_pub = " + n_pub);
        //Below statement will give error as default not accessible in subclass of diff package
        //System.out.println("n_def = " + n_def);
        }

}
```

OtherPackage.java

```java
package P2;

public class OtherPackage {

    OtherPackage() {
        P1.Protection p = new P1.Protection();
        System.out.println("other package constructor");
        // class or package only
        // System.out.println("n_pri = " + p.n_pri);
        // class, subclass or package only
        // System.out.println("n_pri = " + p.n_pri);
        System.out.println("n_pri not accessible from non subclass diff package");
        System.out.println("n_pro not accessible from non subclass diff package");
        //System.out.println("n_def = " + p.n_def);
        System.out.println("n_def not accessible from non subclass diff package");
        System.out.println("n_pub = " + p.n_pub);
        }

}
```

OR

OtherPackage.java

```java
package P2;
import P1.*;
public class OtherPackage {

    OtherPackage() {
        P1.Protection p = new Protection();
        System.out.println("other package constructor");
        // class or package only
        // System.out.println("n_pri = " + p.n_pri);
        // class, subclass or package only
        // System.out.println("n_pri = " + p.n_pri);
        System.out.println("n_pri not accessible from non subclass diff package");
        System.out.println("n_pro not accessible from non subclass diff package");
        //System.out.println("n_def = " + p.n_def);
        System.out.println("n_def not accessible from non subclass diff package");
        System.out.println("n_pub = " + p.n_pub);
        }

}
```

Demo.java (Driver class in P2)

```java
package P2;

public class Demo {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Protection2 ob1 = new Protection2();
        OtherPackage ob2 = new OtherPackage();
    }

}
```

OUTPUT: DEMO P2

```
base constructor
n = 1
n_pri = 2
n_pro = 3
n_pub = 4
n_def = 5
derived other package constructor
n_pro = 3
n_pub = 4
base constructor
n = 1
n_pri = 2
n_pro = 3
n_pub = 4
n_def = 5
```

```
other package constructor
n_pri not accessible from non subclass diff
package
n_pro not accessible from non subclass diff
package
n_def not accessible from non subclass diff
package
n_pub = 4
```

The two classes defined in **p2** cover the other two conditions which are affected by access control. The first class, **Protection2**, is a subclass of **p1.Protection**. This grants access to all of **p1.Protection**'s variables except for **n_pri** (because it is **private**) and **n**, the variable declared with the default protection. Remember, the default only allows access from within the class or the package, not extra-package subclasses. Finally, the class **OtherPackage** has access to only one variable, **n_pub**, which was declared **public.**

Namah Shivaya!