

# Java Package

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

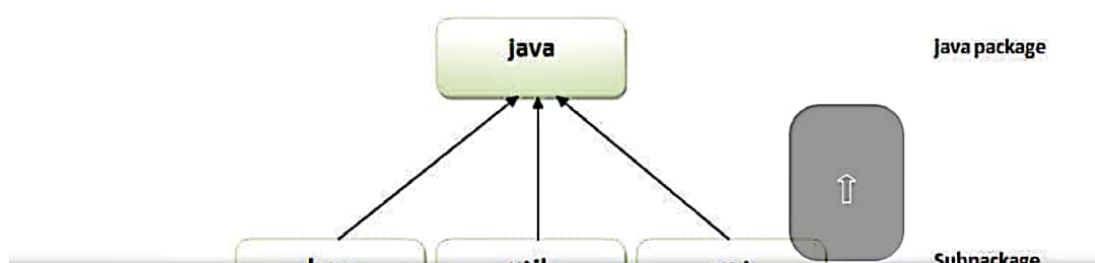
Package in java can be categorized in two form, built-in package and user-defined package.

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

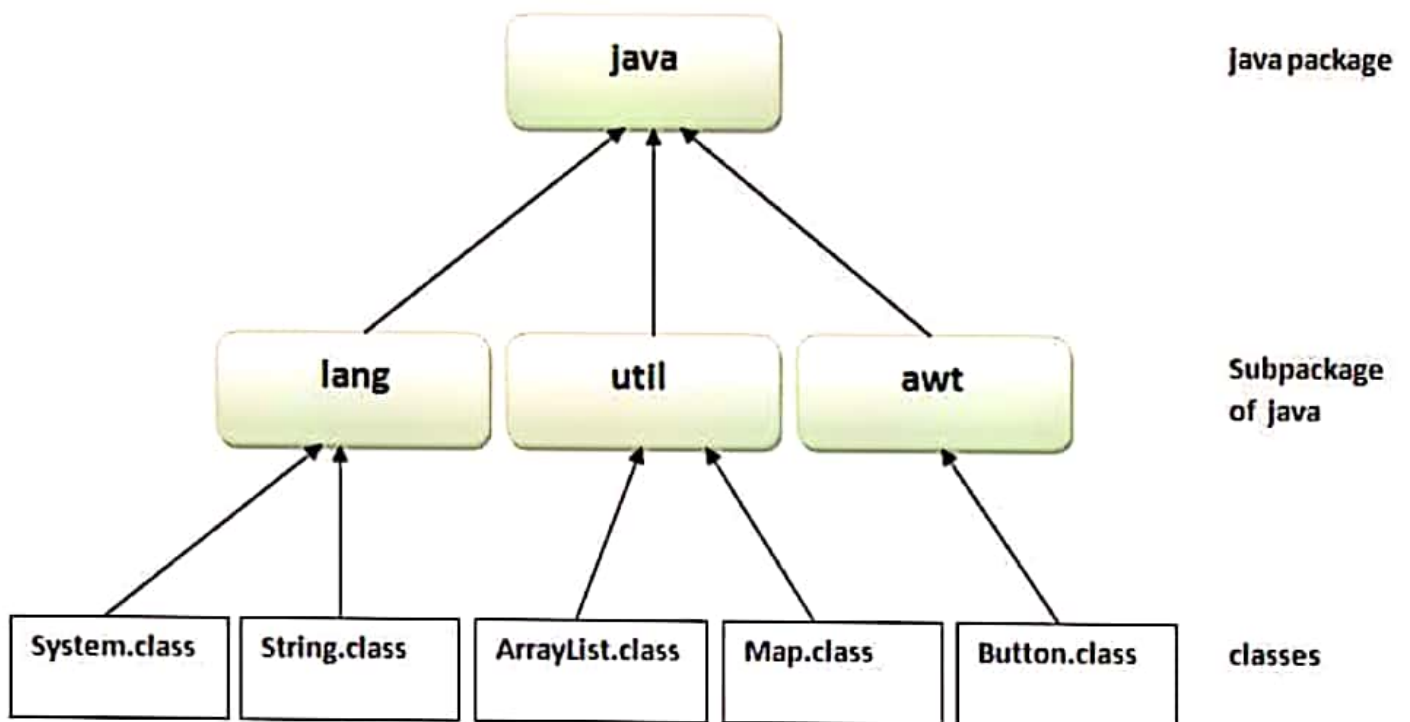
Here, we will have the detailed learning of creating and using user-defined packages.

## 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.



- 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.



---

## Simple example of java package

# How to access package from another package?

There are three ways to access the package from outside 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 but not subpackages.

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{
```



```
//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 package.name.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[]){
```



```
//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.

It is generally used when two packages have same class name e.g. java.util and java.sql packages contain Date class.

#### 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 R{
```





# 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: If you import a package, subpackages will not be imported.**

If you import a package, all the classes and interface of that package will be imported excluding the classes and interfaces of the subpackages. Hence, you need to import the subpackage as well.

---

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

