

## Java Packages

It is an encapsulation mechanism to group related classes and interfaces into single module.

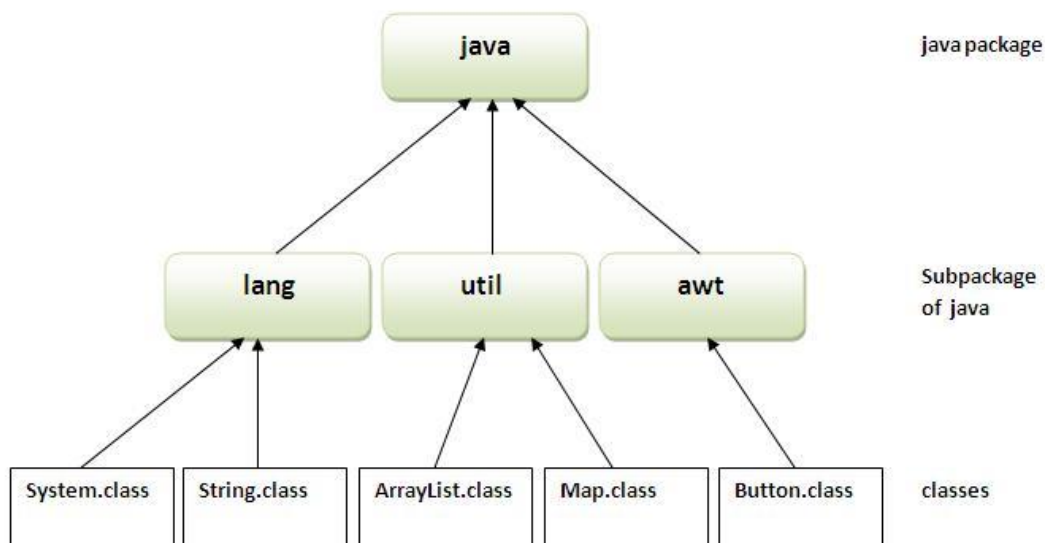
A **java package** is a group of similar types of classes, interfaces and sub-packages.

Package in java can be categorized in two form:

- built-in package like lang,util, awt, javax, swing, net, io, util, sql etc.
- user-defined package

### Advantage of Java Package

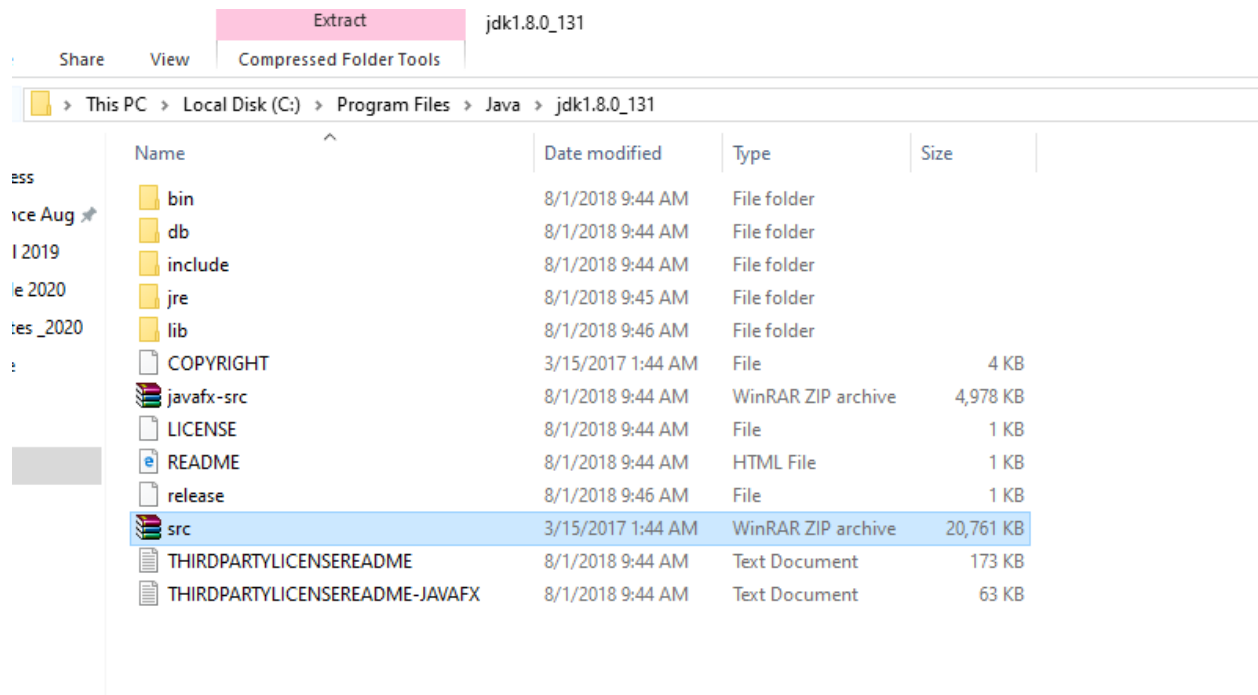
- Java package is used to categorize the classes and interfaces so that they can be easily maintained.
- Java package provides access protection/security to the classes and interfaces. So that outside person cannot access directly.
- It improve modularity of the application.
- Java package removes naming collision.



If you want to see the .class files, they're in lib\rt.jar in the JRE directory (.jar is the same as .zip, so you can open it with anything that can open zip files).

This PC > Local Disk (C:) > Program Files > Java > jre1.8.0_131 > lib					
	Name	Date modified	Type	Size	
ss	calendar	8/1/2018 9:50 AM	Properties Source ...	2 KB	
ce Aug	charsets	8/1/2018 9:50 AM	Executable Jar File	2,966 KB	
2019	classlist	8/1/2018 9:50 AM	File	83 KB	
e 2020	content-types	8/1/2018 9:50 AM	Properties Source ...	6 KB	
es_2020	currency.data	8/1/2018 9:50 AM	DATA File	5 KB	
	deploy	8/1/2018 9:50 AM	Executable Jar File	4,921 KB	
	flavormap	8/1/2018 9:50 AM	Properties Source ...	4 KB	
	fontconfig.bfc	8/1/2018 9:50 AM	BFC File	4 KB	
	fontconfig.properties.src	8/1/2018 9:50 AM	SRC File	11 KB	
	hijrah-config-umalqura	8/1/2018 9:50 AM	Properties Source ...	14 KB	
	javafx	8/1/2018 9:50 AM	Properties Source ...	1 KB	
	javaws	8/1/2018 9:50 AM	Executable Jar File	919 KB	
	jce	8/1/2018 9:50 AM	Executable Jar File	114 KB	
	jfr	8/1/2018 9:50 AM	Executable Jar File	548 KB	
	jfxswt	8/1/2018 9:50 AM	Executable Jar File	34 KB	
	jsse	8/1/2018 9:50 AM	Executable Jar File	570 KB	
	jvm.hprof	8/1/2018 9:50 AM	Text Document	5 KB	
	logging	8/1/2018 9:50 AM	Properties Source ...	3 KB	
	management-agent	8/1/2018 9:50 AM	Executable Jar File	1 KB	
	meta-index	8/1/2018 9:50 AM	File	3 KB	
	net	8/1/2018 9:50 AM	Properties Source ...	5 KB	
	plugin	8/1/2018 9:50 AM	Executable Jar File	1,878 KB	
	psfont.properties.ja	8/1/2018 9:50 AM	JA File	3 KB	
	psfontj2d	8/1/2018 9:50 AM	Properties Source ...	11 KB	
	resources	8/1/2018 9:50 AM	Executable Jar File	3,411 KB	
	rt	8/1/2018 9:50 AM	Executable Jar File	53,231 KB	
	sound	8/1/2018 9:50 AM	Properties Source ...	2 KB	
	tzdb.dat	8/1/2018 9:50 AM	DAT File	104 KB	

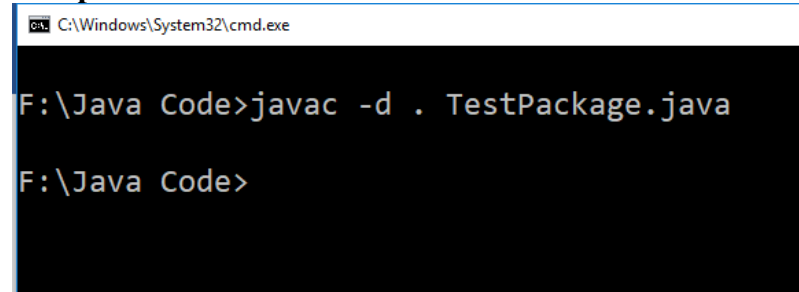
If you want to see the source code, look in src.zip in the JDK directory.



## My first java package

```
package mypackage;
public class TestPackage{
    public static void main(String args[]){
        System.out.println("This is my package");
    }
}
```



## Compilation command :







The -d is a switch that tells the compiler where to put the class file i.e. it represents destination. The . (dot) represents the current working directory/folder.

After compilation of this program, compiler will generate two things:

1. Mypackage folder in F:\Java Code folder
2. TestPackage.class file in mypackage folder.

 > This PC > UPES (F:) > Java Code			
	Name	Date modified	Type
SS	 mypackage	2/21/2019 9:42 AM	File folder

 > This PC > UPES (F:) > Java Code > mypackage			
	Name	Date modified	Type
SS	 TestPackage.class	2/21/2019 9:42 AM	CLASS File
ds			
nts			

**Some Wrong Commands:**

```
F:\Java Code>javac-d.TestPackage.java
'javac-d.TestPackage.java' is not recognized as an internal or
operable program or batch file.

F:\Java Code>javac -d.TestPackage.java
javac: file not found: -d.TestPackage.java
Usage: javac <options> <source files>
use -help for a list of possible options

F:\Java Code>javac -d .TestPackage.java
javac: directory not found: .TestPackage.java
Usage: javac <options> <source files>
use -help for a list of possible options
```

**Execution command :**

```
F:\Java Code>javac -d . TestPackage.java

F:\Java Code>java mypackage.TestPackage
This is my package
```

### User defined location:

```
package mypackage1;
public class TestPackage1{
    public static void main(String args[]){
        System.out.println("This is my package1");
    }
}
```

### WRONG:

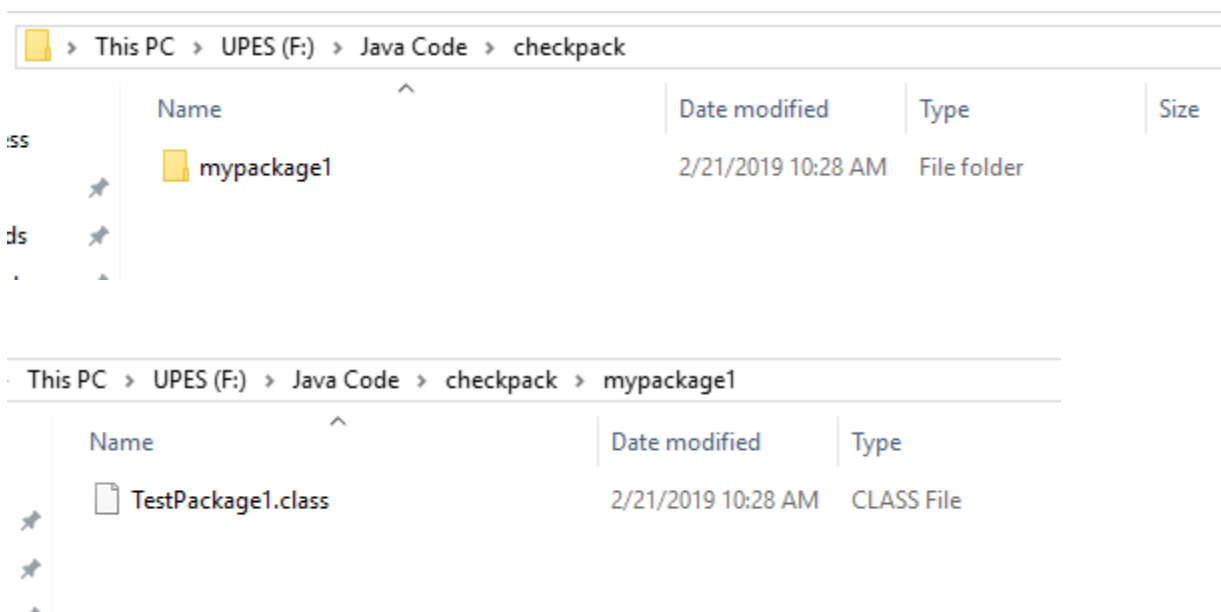
```
F:\Java Code\checkpack>javac -d F:\Java Code\checkpack TestPackage1.java
javac: invalid flag: Code\checkpack
Usage: javac <options> <source files>
use -help for a list of possible options
```

### Right:

```
F:\Java Code>javac -d "F:\Java Code\checkpack" TestPackage1.java
F:\Java Code>
```

After compilation of this program, compiler will generate two things:

1. Mypackage1 folder in F:\Java Code\checkpack folder
2. TestPackage1.class file in mypackage1 folder.



```
F:\Java Code>cd checkpack

F:\Java Code\checkpack>java mypackage1.TestPackage1
This is my package1
```

There are three ways to access the package from outside the package.(from one package to another package)

**1. import package.\*;**

```
1 package upes;
2 public class Ifm{
3     public void ifmData()
4     {
5         System.out.println("Data of IFM Students");
6     }
7 }
```

```
package adminUpes;
import upes.*;

class Admin{
    public static void main(String args[]){
        Ifm i = new Ifm();
        i.ifmData();
    }
}
```

```
F:\Java Code>javac -d . Ifm.java

F:\Java Code>javac -d . Admin.java

F:\Java Code>java adminUpes.Admin
Data of IFM Students
```

**2. import package.classname;**

```

1 package upes;
2 public class Ifm{
3     public void ifmData()
4     {
5         System.out.println("Data of IFM Students");
6     }
7 }

```

---

```

1 package adminUpes;
2 import upes.If;
3
4 class Admin{
5     public static void main(String args[]){
6         If i = new If();
7         i.ifmData();
8     }
9 }

```

```

F:\Java Code>javac -d . Ifm.java

F:\Java Code>javac -d . Admin.java

F:\Java Code>java adminUpes.Admin
Data of IFM Students

```

### 3. fully qualified name.

```

package adminUpes;
//import upes.If;

class Admin{
    public static void main(String args[]){
        upes.If i = new upes.If();
        i.ifmData();
    }
}

```

```
F:\Java Code>javac -d . Ifm.java

F:\Java Code>javac -d . Admin.java

F:\Java Code>java adminUpes.Admin
Data of IFM Students
```

There are only two ways to access the package from outside the package. (class → package)

### 1. **import package.\*;**

```
package Sport;
public class Cricket{
String name="Subham";
    public void run()
    {
        System.out.println(name+ " is a Cricket Player ");
    }
}
```

**import Sport.\*;** //will give compile time error

```
class Hockey{
    public static void main(String args[]){
        Cricket c=new Cricket();
        c.run();
    }
}
```



```
C:\Windows\System32\cmd.exe
F:\Java Code>javac -d . Cricket.java

F:\Java Code>javac Hockey.java
Hockey.java:6: error: cannot access Cricket
    Cricket c=new Cricket();
    ^
bad source file: .\Cricket.java
    file does not contain class Cricket
    Please remove or make sure it appears in
the correct subdirectory of the sourcepath.
1 error
```

## 2. `import package.classname;`

```
import Sport.Cricket;
```

```
class Hockey{
    public static void main(String args[]){
        Cricket c=new Cricket();
        c.run();
    }
}
```

```
F:\Java Code>javac -d . Cricket.java

F:\Java Code>javac Hockey.java

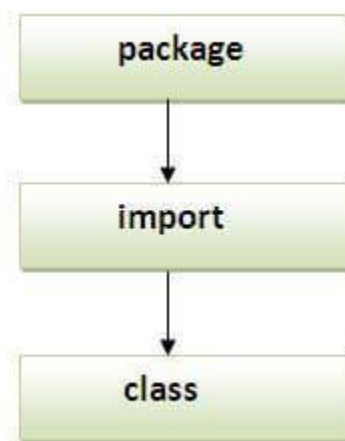
F:\Java Code>java Hockey
Subham is a Cricket Player
```

### 3. fully qualified name.

```
class Hockey{  
    public static void main(String args[]){  
        Sport.Cricket c=new Sport.Cricket();  
        c.run();  
    }  
}
```

```
F:\Java Code>javac -d . Cricket.java  
  
F:\Java Code>javac Hockey.java  
  
F:\Java Code>java Hockey  
Subham is a Cricket Player
```

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.



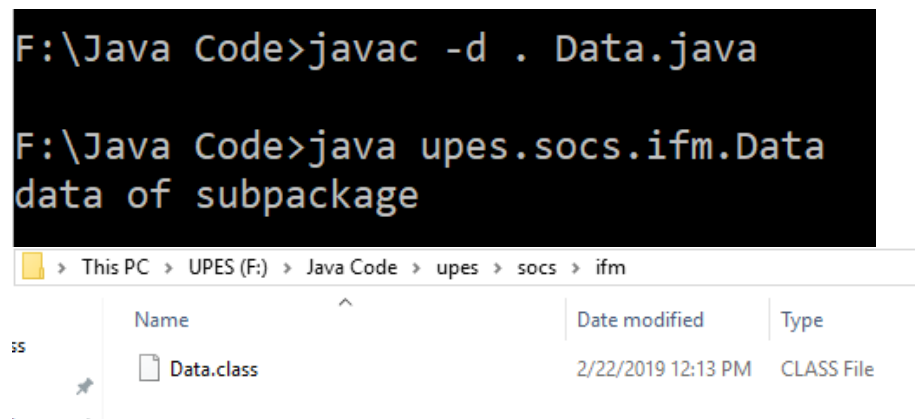
### Subpackage in java

Package inside the package is called the **subpackage**. It should be created to categorize the package further.

Let's take an example, Sun Microsystem has defined a package named java that contains many classes like System, String, Reader, Writer, Socket etc. These classes represent a particular group e.g. Reader and Writer classes are for Input/Output operation, Socket and ServerSocket classes are

for networking etc and so on. So, Sun has subcategorized the java package into subpackages such as lang, net, io etc. and put the Input/Output related classes in io package, Server and ServerSocket classes in net packages and so on.

```
package upes.socs.ifm;
class Data{
    public static void main(String args[]){
        System.out.println("data of subpackage");
    }
}
```



**Note:** In any java program there should be only at most 1 package statement. If we are taking more than one package statement we will get compile time error.

```
package pack1;
package pack2;//error
```

```
class MyPackage1
{
//
--statements--
}
```

compile time error: class ,interface and enum expected.

**Note:** In any java program the first non comment statement should be package statement (if it is available).

```
import java.util.*;
package pack1;//error
class MyPackage2
{

}
```

Compile time error: class ,inetrface and enum expected.

Sturcture of Java source file:

Order	↓	package statement—atmost one
		import statements—any number.
		class/interface/Enum declarations --Any number