3-06-2020 Today we will discuss all about Aggregator module

A module which is going to aggregate.

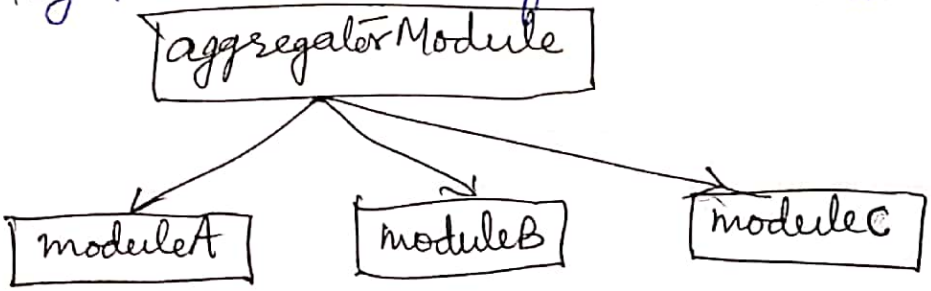
A module which is going to provide several different module's functionality related to some work is known as aggregate module.

Suppose an example of travel agent's :—

'transitive' keyword is very essentially used in making aggregator module.

In other words,— Sometimes a group of modules can be reused by multiple other modules. Then it is not recommended to read each module individually. We can group those common modules into a single module, and we can read that module directly. This module which aggregates functionality of several modules into a single module is called Aggregator module. If any module reads aggregator module then automatically all its modules are by default available to that module.

Aggregator module won't provide any functionality by its own, just it gathers and bundles together a bunch of other modules.



```
module aggregatorModule
{
    requires transitive moduleA;
    requires transitive moduleB;
    requires transitive moduleC;
}
```

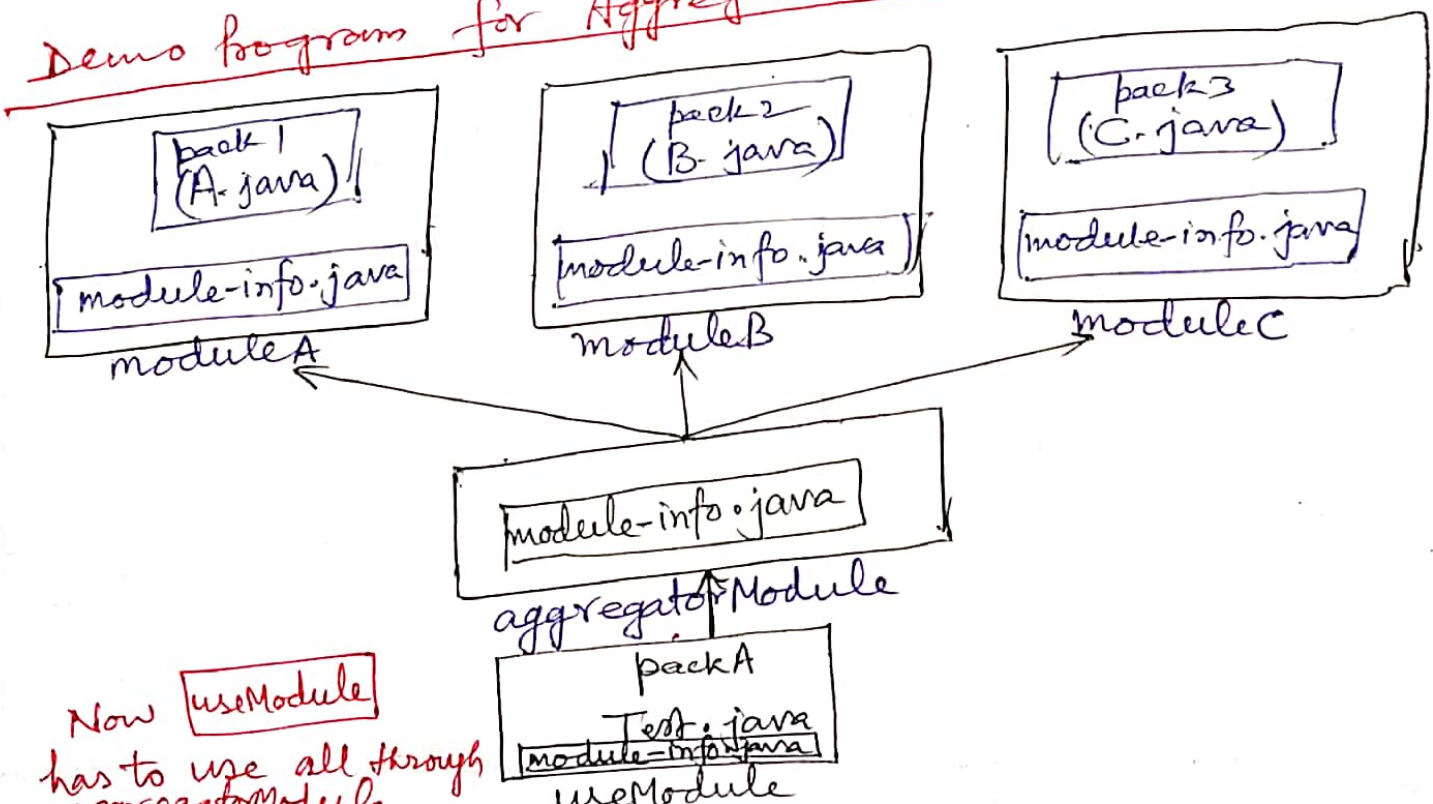
Aggregator module not requires to contain a single java class. Just it "requires transitive" of all common modules.

If any module reads aggregatorModule, automatically all three modules by default available to that module also.

```
module useModule
{
    requires aggregatorModule;
}
```

Now, useModule can use functionality of all three modules moduleA, moduleB and moduleC.

Demo Program for Aggregator Module:-



Now useModule has to use all through aggregatorModule.

In src folder :-

- aggregatorModule
- moduleA
- moduleB
- moduleC
- useModule

5 modules

moduleA

module-info.java

```
module moduleA
{
 exports pack1;
}
```

pack1 contains

A.java

```
package pack1;
public class A
{
public void m1()
 {
System.out.println("moduleA method");
 }
}
```

moduleB

module-info.java

```
module moduleB
 {
 exports pack2;
 }
```

pack2 contains

B.java

```
package pack2;
public class B
 {
public void m1()
 {
System.out.println("moduleB method");
 }
}
```

moduleC

module-info.java

```
module moduleC
 {
 exports pack3;
 }
```

pack3 contains

C.java

```
package pack3;
 public class C {
public void m1() {
System.out.println("moduleC method");
 } }
```

aggregatorModule

module-info.java

```
module aggregatorModule
{
    requires transitive moduleA;
    requires transitive moduleB;
    requires transitive moduleC;
}
```

useModule

module-info.java

```
module useModule {
        requires aggregatorModule;
}
```

packA contains

Test.java

```
package packA;
import pack1.A;
import pack2.B;
import pack3.C;
public class Test{
public static void main(String [] args){
System.out.println("Aggregator Module Demo Program.");
A a = new A();   a.m1();
B b = new B();   b.m1();
C c = new C();   c.m1();
  }
}
```

Let us compile and Run the above

folder src :-

Compile:-

```
javac --module-source-path src -d out -m moduleA,
      moduleB, moduleC, aggregatorModule, useModule
```

Run:-

```
java --module-path out -m useModule/packA.Test
```

Output:-

```
Aggregator Module Demo Program
moduleA method
moduleB method
moduleC method
```