Fpt University

FPT POLYTECHNIC

LẬP TRÌNH JAVA

Bài 1: Packages và Interfaces

www.poly.edu.vn

Điểm danh

Nội dung bài học

- Packages
- Phạm vi truy cập
- Sử dụng package
- Các package thư viện
- Class Abstract
- Interfaces

Packages

- Package được tạo bởi sự kết hợp của nhiều class hay interface.
- Trong package có thể chứa các package khác.
- Package thường chứa các class, interface hay sub-package có liên quan với nhau.
- Có các cơ chế kiểm soát truy cập khác nhau trong package.

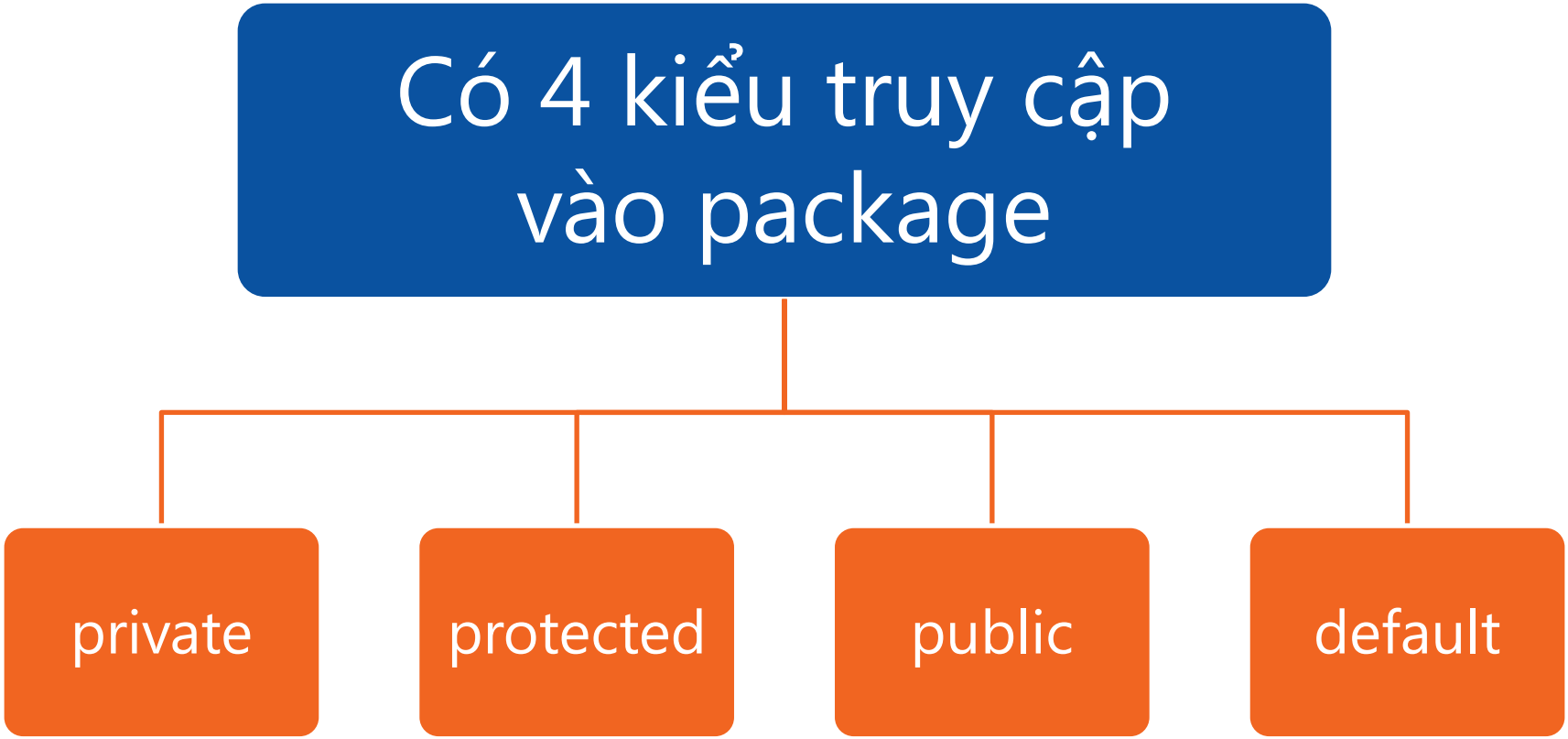
Packages

```
package mypackage;

public class example{
private int count=0;

    public example(int c){
        count = c;
    }
  ...
}
```

Truy cập các thành phần trong package

- Các class mà dự định sẽ được sử dụng bên ngoài package sẽ được khai báo là public.
- Các package khác nhau có thể có các class trùng tên với nhau.
- Nếu các package khác nhau mà có các class có tên trùng nhau thì khi sử dụng bắt buộc phải import đầy đủ tên package và tên class.

Truy cập các thành phần trong package



Truy cập các thành phần trong package

Từ khóa	Trong cùng class	Trong cùng package	Trong sub-package	Package khác
private	Có	Không	Không	Không
default	Có	Có	Không	Không
protected	Có	Có	Có	Không
public	Có	Có	Có	Có

Truy cập các thành phần trong package

Như vậy các member (là các class trong package) được khai báo là:

private: Chỉ có thể được truy cập bởi chính class đó.

default: Được truy cập bởi các class cùng package.

protected: Được truy cập bởi các class cùng trong package và các class là sub-class của class này.

public: Được truy cập bởi tất cả các class ở cùng package hay khác package.

Sử dụng packages

- Cú pháp:
  - import tên_package.tên_class
- Ví dụ:
  - import mypack.MyClass;
  - import mypack.*;
- Ký hiệu *: là import tất cả các class trong package mypack.

Sử dụng packages

```
import java.util.Scanner;
import java.io.File;
public class example{
    public void input(){
        Scanner input = new Scanner(System.in);
         String fileName = input.next();
         File f = new File(fileName);
            . . .
     }
}
```

Các package thư viện

Tên Package	Mô tả
java.lang	Chứa các class như Integer, String, System… và được tự động import vào mỗi chương trình Java.
java.util	Các các Java collections như List, Set, Map …
java.io	Chứa các class liên quan đến việc nhập, xuất dữ liệu như File, Reader, Writer…
java.awt và java.swing	Chứa các class liên quan đến việc trình bày giao diện đồ họa và xử lý sự kiện.
. . .	

Class abstract

```
abstract class A
{
   abstract void callme();

   void callmetoo(){
         System.out.println(“A concrete method.");
   }
}
class B extends A
{
   void callme(){
       System.out.println("B is implementation of
                                callme");
   }
}
```

Class abstract

```
class TestAstract
{
  public static void main(String a[]){
      B obj = new B();// hoặc: A obj = new B()
      obj.callme();
      obj.callmetoo();
  }
}
```

Interfaces

- Trong interface chỉ có các method **abstract** và các biến **final**
- Khi một class **thực thi** một interface, nó phải viết lại (**override**) tất cả các method trong interface.
- Interface là public hoặc default
- Interface có thể được kế thừa
- Một interface có thể được thực hiện bởi **nhiều** class, và một class có thể thực thi **nhiều** interface. Đó chính là cách để dùng "kế thừa từ nhiều class".

Interfaces

```
package mypackage;
interface myinterface{
    void mymethod1();
    void mymethod2();
}
```

Interfaces

```
package mypackage;
class myclass implements myinterface{

    public void mymethod1(){ //phải là public
        System.out.println(“Override my method 1");
    }

    public void mymethod2(){
        System.out.println(“Override my method 2");
    }

    void mymethod3(){//không là method trong interface
        System.out.println("My method 3");
    }
}
```

Interfaces

```
public static void main (String a[]){

            myclass obj1 = new myclass();
            obj1.mymethod1();
            obj1.mymethod2();
            obj1.mymethod3();

            myinterface obj2 = new myclass();
            obj2.mymethod1();
            obj2.mymethod2();
            obj2.mymethod3();// error;
      }
}
```

Interfaces

```
interface A{
	void meth1();
	void meth2();
}
interface B extends A{
	void meth3();
}
class MyClass implements B{
	public void meth1(){
		System.out.println("Implements method 1");
	}

	public void meth2(){
		System.out.println("Implements method 2");
	}
	public void meth3(){
		System.out.println("Implements method 3");
	}
}
```

Tổng kết bài học

- Package
- Phạm vi truy cập
- Sử dụng package
- Các package thư viện
- Class Abstract
- Interface