# ជំពូកទី ៧ : Packages និង Interfaces

## ១. អ្វីទៅ Package ?

Package មួយបំរើអោយគោលបំណងពីរ ។

- package មួយផ្តល់នូវមធ្យោបាយមួយសំរាប់តាងអោយឈ្មោះបណ្តុំនៃ class ។

- package រួមជាមួយ mechanism សំរាប់ត្រួតពិនិត្យការចូលប្រើនៃភាសា Java និងបញ្ជាក់អោយទីតាំងមួយ ។

By Mr. Chi Kuong

## ១.១ និយមន័យ package

ដើម្បីបង្កើត package មួយ ចូរប្រើពាក្យ package នៅផ្នែកខាងលើនៃឯកសារដើមរបស់ Java ។ ពេលនោះ class ដែលបានប្រកាសនៅក្នុង file ជារបស់ package ដែលបានកំនត់ ។ ទំរង់ទូទៅនៃឃ្លា package ៖

```
package pkg ;
```

ក្នុងនេះ *pkg* ជាឈ្មោះរបស់ package ។

ឧទាហរណ៏ ៖ package Project1;

## ១.២ ឧទាហរណ៏អំពី package

ចូរសង្កេតលើឧទាហរណ៏អំពី package នេះ។ វាបង្ហើតនូវការគ្រប់គ្រងទិន្នន័យសៀវភៅមួយដែលដាក់នៅក្នុង package មួយឈ្មោះ BookPack។ យើងដាក់ឈ្មោះឯកសារនេះជា BookDemo.java ហើយដាក់ក្នុង directory មួយឈ្មោះ BookPack។ បន្ទាប់មកយើងត្រូវធ្វើការបំលែងឯកសាររួចហើយចូរសាកល្បងប្រតិបត្តិការនូវ class ដោយប្រើពាក្យបញ្ជាដូចខាងក្រោមនេះលើ command-line :

```
>java BookPack.BookDemo ↵
```

By Mr. Chi Kuong

# ២. Packages និងការចូលប្រើ member

គេអាចចូលប្រើ members នៃ class ដែលបានកំណត់ដោយពាក្យ private, protected, public ឬ (default) ហើយនិង package ដែលមាន classes ទាំងនោះកំពុងស្ថិតនៅ។ ហេតុនេះ ភាពអាចចូលប្រើធាតុមួយត្រូវបានកំណត់ដោយកំរិតចូលប្រើរបស់វានៅក្នុង class មួយ ហើយនិង package នោះ។

| សិទ្ធិប្រើបាន | Private Member | Default Member | Protected Member | Public Member |
|---|---|---|---|---|
| • កើតប្រើបានក្នុង class ដូចគ្នា | បាន | បាន | បាន | បាន |
| • កើតប្រើបានក្នុង package ដូចគ្នា ចំពោះ subclass | មិនបាន | បាន | បាន | បាន |
| • កើតប្រើបានក្នុង package ដូចគ្នា ចំពោះ non-subclass | មិនបាន | បាន | បាន | បាន |
| • កើតប្រើបានក្នុង package ផ្សេងគ្នា ចំពោះ subclass | មិនបាន | មិនបាន | បាន | បាន |
| • កើតប្រើបានក្នុង package ផ្សេងគ្នា ចំពោះ non-subclass | មិនបាន | មិនបាន | មិនបាន | បាន |

## ២.១ ឧទាហរណ៍ពីការចូលទៅប្រើ package

នៅក្នុងឧទាហរណ៏ package ដែលបានបង្ហាញមុននេះ Book និង BookDemo ត្រូវដាក់ក្នុង package តែ១ ហេតុនេះ វាពុំមានបញ្ហាជាមួយនឹង BookDemo ដែលប្រើ Book ព្រោះក៏រិត default ផ្តល់អោយគ្រប់ members ទាំងអស់នៃ package ដូចគ្នាអាចចូលប្រើបាន ។ ទោះជាយ៉ាងណាក៏ដោយ បើសិនជា Book ដាក់ក្នុង package មួយ ហើយ BookDemo ដាក់ក្នុង package មួយទៀតនោះវាមានសភាពខុសគ្នា ។ នៅក្នុងករណីនេះ ការចូលប្រើ Book នឹងត្រូវបានបដិសេដជាពុំខាន ។

By Mr. Chi Kuong

# ២.២ ការពន្យល់អំពី members មានលក្ខណៈ proctected

protected member មួយអាចអោយ subclass ទាំងអស់ប្រើបាន ប៉ុន្តែវានៅតែមានការរាំងពីការចូលទៅប្រើចំពោះ code ដែលនៅខាងក្រៅ package របស់វា។

ឧទាហរណ៏ :

P P

## ៣. ការនាំយក class ពីក្នុង package មកប្រើ

ឃ្លា import ជាវិធីមួយសំរាប់ប្រើ class នៅក្នុង package ហើយអាចនាំ member មួយឬច្រើនក្នុង package មកប្រើ ។

ទំរង់ទូទៅនៃឃ្លា import មានដូចខាងក្រោមនេះ ៖

```
import pkg.classname;
```

ក្នុងនេះ ***pkg*** ជាឈ្មោះរបស់ package ដែលរួមបញ្ចូល ទាំង path របស់វាដោយមានការបញ្ជាក់យ៉ាងពេញលេញ ហើយនិង ***classname*** ជាឈ្មោះរបស់ class ដែលត្រូវយកមកប្រើ ។

By Mr. Chi Kuong

បើសិនជាយើងចង់នាំ class ទាំងអស់ដែលមាននៅក្នុង package យើងត្រូវប្រើសញ្ញា * ជំនួសអោយឈ្មោះ class ។

នេះជាឧទាហរណ៏នៃទំរង់ទាំងពីរយ៉ាង :

```
import  MyPack.MyClass;
```

ឬ

```
import  MyPack.*;
```

ឧទាហរណ៏ :

By Mr. Chi Kuong

## ៤. Class library របស់ Java ត្រូវបានផ្ទុកនៅក្នុង packages

Java មាន class គំរូជាច្រើនដែលអាចអោយកម្មវិធីប្រើ ។ class library នេះសំដៅទៅលើ Java API (Application Programming Interface) ។ Java API ត្រូវដាក់នៅក្នុង packages ទាំងអស់ ។ នៅខាងលើនៃ លំដាប់ថ្នាក់ package គឺ java ។ លំដាប់ចុះពី java គឺមាន subpackages ជាច្រើន ដូចជា :

| Subpackage | អធិប្បាយ |
|---|---|
| java.lang | ផ្ទុក class ទូទៅមួយចំនួនធំ |
| java.io | ផ្ទុក class ប្រភេទ I/O |
| java.net | ផ្ទុក class សំរាប់ជំនួយបណ្តាញ |
| java.applet | ផ្ទុក class សំរាប់បង្កើត applets |
| java.awt | ផ្ទុក class សំរាប់ជំនួយ Abstract Window Toolkit |

By Mr. Chi Kuong

## ៥. អំពី interfaces

នៅក្នុងកម្មវិធីសំនេរបែបវត្ថុ ជួនកាលវាមានសារៈប្រយោជន៍ដើម្បីកំណត់នូវអ្វីដែល class មួយត្រូវធ្វើ តែមិនកំណត់នូវរបៀបដែលត្រូវធ្វើនោះទេ ។ យើងបានឃើញឧទាហរណ៍បែបនេះ រួចហើយ គឺ abstract method ។ abstract method មួយកំណត់នូវសញ្ញាណ method មួយ ប៉ុន្តែពុំមានការប្រើឡើយ ។ ហេតុនេះ abstract method ផ្តល់នូវ interface ទៅអោយ method ដោយគ្មានការប្រើ ។ ភាសា Java យើងអាចបំបែក interface នៃ class មួយពីការប្រើរបស់វាដោយប្រើពាក្យ interface ។

By Mr. Chi Kuong

ទំរង់ទូទៅនៃ interface មួយ ៖

```
access interface name {
   return-type  method-name1(para-list);
   return-type  method-name2(para-list);
   type final-varname1 = value;
   type final-varname2 = value;
     //…
   return-type method-nameN(para-list);
   type final-varnameN = value;
}
```

ក្នុងនេះ *access* អាចជា public ឬមិនប្រើ ។

ឧទាហរណ៍ ៖

By Mr. Chi Kuong

## ៥.១ ការប្រើ interface

ដើម្បីប្រើ interface មួយគេត្រូវបញ្ចូលឃ្លា implements ទៅក្នុងការកំណត់ class មួយ រួចហើយបង្កើត methods កំណត់ដោយ interface នោះ ។ ទំរង់ទូទៅនៃ class មួយ ដែលបញ្ចូលឃ្លា implements មានដូចខាងក្រោមនេះ :

```
access class classname extends superclass
                    implements interface {
   // class-body
}
```

ក្នុងនេះ *access* អាចជា public ឬមិនប្រើ ។ តាមពិតពាក្យ extends អាចប្រើក៏បាន មិនប្រើក៏បាន ។ ដើម្បីប្រើ interface លើសពីមួយ នោះ interfaces ទាំងអស់ត្រូវខ័ណ្ឌដាច់ពីគ្នាដោយសញ្ញាក្បៀស ។ methods ដែលប្រើតាម interface មួយត្រូវតែប្រកាសជា public ។ ប្រភេទទិន្នន័យនៃ method ដែលកំពុងប្រើ ត្រូវតែស៊ីគ្នានឹងប្រភេទទិន្នន័យដែលបានកំណត់នៅក្នុង interface ។

ឧទាហរណ៍ :

## ៥.២ ការប្រើ interface references

យើងអាចបង្កើតអញ្ញាត interface reference មួយ។ អញ្ញាតប្រភេទនេះអាចបញ្ជាក់អោយ object ណាមួយដែលប្រើ interface របស់វា។ កាលណាយើងហៅ method មួយមកប្រើលើ object មួយតាមរយៈ interface reference នោះវាជា method ប្រើដោយ object ដែលត្រូវប្រតិបត្តិការ។ ដំណើរការនេះស្រដៀងទៅនឹងការប្រើ superclass reference ដើម្បីចូលប្រើ object នៃ subclass មួយ។

ឧទាហរណ៍ :

By Mr. Chi Kuong

## ៥.៣ អញ្ញាតនៅក្នុង interfaces

ដើម្បីកំនត់ចំនួនថេរដែលប្រើរួមគ្នា យើងត្រូវបង្កើត interface មួយ ដែលមានតែចំនួនថេរទាំងនេះដោយគ្មាន method ណាមួយឡើយ ។ ឯកសារនិមួយៗដែលត្រូវការចូល ប្រើចំនួនថេរបានយ៉ាងងាយនោះ គេត្រូវការប្រើ interface ។

ឧទាហរណ៍ :

```
interface IConst {
   int MIN = 0;
   int MAX = 10;
   String ERRORMSG = "Boundary Error";
}
```

By Mr. Chi Kuong

# ៥.៤ Interfaces អាចពង្រីកបន្តបាន

interface មួយអាចទទួលលក្ខណៈពី interface មួយទៀតបានដោយប្រើពាក្យ extends ។ ទំរង់នៃការប្រើមានលក្ខណៈដូចនៅក្នុង class ដែរ ។ កាលណា class មួយប្រើ interface មួយដែលទទួលលក្ខណៈពី interface មួយទៀត នោះវាត្រូវតែប្រើ methods ទាំងអស់ដែលបានកំនត់នៅក្នុង interface បានទទួលលក្ខណៈបន្តគ្នា ។

ឧទាហរណ៏ :

# សំនួរ និង លំហាត់

១ - អ្វីទៅហៅថា package?

២ - តើ CLASSPATH ជាអ្វី?

៣ - ចូរពន្យល់ពីការកំរិតចូលប្រើ members នៃ class ដែលមានលក្ខណៈ private, (default), protected និង public ។

៤ - ចូរពន្យល់ពីការប្រើពាក្យ import និង package ក្នុងភាសា Java ។

By Mr. Chi Kuong

៥ - តើ interface គឺជាអ្វី? ហើយគេអាចសំគាល់វាបាន ដោយសារអ្វី?

៦ - តើពាក្យ implements ប្រើសំរាប់ធ្វើអ្វី?

៧ - តើ interface មួយអាចទទួលលក្ខណៈពី interface មួយផ្សេងទៀតបានដែរឬទេ?

៨ - តើអញ្ញាតប្រកាសក្នុង interface មួយត្រូវមានលក្ខណៈ បែបណា?

៩ - តើ method ប្រកាសក្នុង interface មួយត្រូវមាន លក្ខណៈបែបណា?

By Mr. Chi Kuong