មជ្ឈមណ្ឌលកូរ៉ែ សហ្វវែរ អេច អ ឌី

Korea Software HRD Center

ការស្វែងយល់ពី Collection ក្នុង Java

ណែនាំដោយ : Dr. Kim Tae Kyung

Webcash

KOICA
Korea International Cooperation Agency

មាតិកា

១. ការស្វែងយល់អំពី Java Collection

- Collection ក្នុង Java គឺជា Framework មួយដែលផ្តល់ architecture សំរាប់រក្សាទុក និង រៀបចំបណ្តុំនៃ objects។

- គ្រប់ប្រតិបត្តិការណ៍ទាំងអស់ដែលធ្វើការជាមួយ data ដូចជា searching, sorting, insertion, manipulation, deletion, etc ទាំងអស់នេះអាចធ្វើដោយ Java Collections ។

១. ការស្វែងយល់អំពី Java Collection(ត)

- Java Collection គឺមានលក្ខណៈជា single unit objects(a group) ។ Java Collection Framework ផ្តល់នូវ Interfaces ជាច្រើន ដូចជា Set List Map។ល។ និង Classes ជាច្រើន ដូចជា ArrayList, LinkedList, HashMap ។ល។

- **java.util** package រួមមាន classes និង Interfaces ទាំងអស់សំរាប់ Collection Framework

១. ការស្វែងយល់អំពី Java Collection(ត)

- គុណសម្បត្តិនៃ collection
  - អាច add Append, insert, remove object
  - Auto resize
  - Serializable (convert ជា byte សម្រាប់ write ចូល file)
- គុណវិបត្តិ
  - អាច store បានតែ object (មិនអាច store Primitive value បានទេ)។

១.១. ការស្វែងយល់អំពី Method របស់ Collection

No.	Method	Description
1	public boolean add(Object element)	បញ្ចូល element ទៅក្នុង collection
2	public boolean addAll(Collection c)	បញ្ចូល elements របស់ collection ជាក់លាក់ណាមួយ ចូលទៅកាន់ element នីមួយរបស់ collection ដែលបានហៅ
3	public boolean remove(Object element)	លុប element ណាមួយពី collection.
4	public boolean removeAll(Collection c)	លុប elements ទាំងអស់របស់ invoking collection ទៅតាម elements របស់ collection ជាក់លាក់ក្នុង parameter
5	public boolean retainAll(Collection c)	លុប elements ទាំងអស់របស់ invoking collection លើកលែងតែ elements របស់ collection ជាក់លាក់ក្នុង parameter
6	public int size()	ផ្តល់ចំនួនសរុបនៃ elements ក្នុង collection.
7	public void clear()	លុប element ទាំងអស់ពី collection

១.១. ការស្វែងយល់អំពី Method របស់ Collection(ត)

No.	Method	Description
8	public boolean contains(Object element)	ស្វែងរក element ណាមួយ
9	public Iterator iterator()	ផ្តល់ iterator មកវិញ
10	public boolean isEmpty()	ពិនិត្យមើល តើ collection ទទេ ឬអត់
11	public boolean equals(Object element)	ធ្វៀងផ្ទាត់រវាង collection ពីរ
12	public int hashCode()	ផ្តល់នូវ hashcode number សម្រាប់ collection.

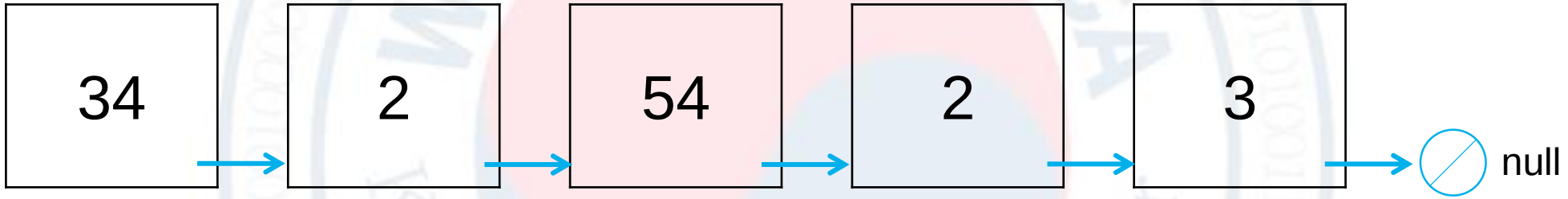
២. ការស្វែងយល់យល់អំពី List

អ្វីទៅជា List?

- List គឺជា Interface ដែល extends ពី Collection ។
- វាបានធ្វើការ declares behavior នៃ collection ដែល store sequence នៃ elements។ វាត្រូវបាន class មួយចំនួន implement ដើម្បីបង្កើតជា class មានលក្ខណៈជា collection។
- ចំនុចសំខាន់ List អនុញ្ញាត្តិឲ្យយើងគឺ add duplicated element។
- Element របស់វាអាច insert រឺ access បានតាមរយៈ index។

២. ការស្វែងយល់យល់អំពី List(ត)

- នៅក្នុង List អាចផ្ទុកទិន្នន័យ duplicated បាន។



- Class ដែលធ្វើការ implement ពី interface List មានដូចជា:

២.១. សិក្ខាអំពី ArrayList

- ArrayList class ធ្វើការ extend ចេញពី AbstractList ហើយធ្វើការ implement ចេញពី List Interface ។ ArrayList គឺជា Dynamic arrays មានន័យថា វាអាចពង្រីកពង្រួមប្រវែង តាមតំរូវការ ។
  - ArrayList class អាចមាន elements ស្ទួន
  - ArrayList class អនុញ្ញាតឱ្យ random access ព្រោះវាធ្វើការតាមរយៈ index
  - ArrayList class មានដំណើរការយឺតចំពោះការ manipulation ព្រោះវាមានការផ្លាស់ប្តូរបើសិន element ណាមួយមានការលុបពី ArrayList

២.១. សិក្ខាអំពី ArrayList(ត)

- តើ ArrayList ប្រើនៅពេលណា?
  - ជាធម្មតា Array មានលក្ខណៈ Fix Length, បន្ទាប់ពី Array ត្រូវបានបង្កើតគឺធាតុរបស់វាមិនអាចកើនឬក៏ធ្លាក់ចុះទេ ប៉ុន្តែ ArrayList យើងអាចបង្កើតវាជាមួយ initial size នៅពេលដែលទំហំរបស់វាលើស Collection របស់វានិងកើនឡើងដោយស្វ័យប្រវត្តិហើយនៅពេល Collection របស់វាត្រូវបាន removed នោះទំហំ Collection នឹងធ្លាក់ចុះ។

២.១. សិក្ខាអំពី ArrayList(ត)

- ArrayList class មាន ៣ Constructors ដូចខាងក្រោម :
  - ArrayList( ): Constructor ដំបូងនៃ ArrayList បង្កើតដោយ arrayList ទទេ (empty) គឺមានទំហំ 10។
  - ArrayList(int capacity): constructor នៃ ArrayList មាន capacity ដំបូងជាក់លាក់។ Capacity គឺជាទំហំមូលដ្ឋាន array ដែលប្រើសំរាប់រក្សាទុក (elements) ។
  - ArrayList(Collection c): តាមរយៈការបង្កើត constructor នៃ ArrayList ការផ្តល់តំលៃ elements តាមរយៈ collection ។

២.១. សិក្ខាអំពី ArrayList(ត)

- **ឧទាហរណ៍**: ArrayList Class

```
public static void main(String[] args) {
   ArrayList<String> arrayList = new ArrayList<String>();// creating arraylist
   arrayList.add("Sok");// adding object in arraylist
   arrayList.add("Sao");
   arrayList.add("Sara");
   arrayList.add(1,"Difference");

   //Display Size and data
   System.out.println("Size: " + arrayList.size() + "\nData: " + arrayList + "\n");

   // getting Iterator from arraylist to traverse elements
   Iterator<String> itr = arrayList.iterator();
   while (itr.hasNext())
      System.out.println(itr.next());
}
```

```
Size: 4
Data: [Sok, Difference, Sao, Sara]

Sok
Difference
Sao
Sara
```

២.២. សិក្សាអំពី LinkedList

តើអ្វីទៅ LinkedList?

- LinkedList គឺជាប្រភេទ collection ដែលមានភាពពេញនិយមសម្រាប់រក្សាទុក Data ជាលក្ខណៈ Array។ ធ្វើការ extend ពី AbstractSequentialList ហើយ implement ចេញពី List Interface ។

- គុណសម្បត្តិចម្បងរបស់ LinkedList គឺមិនចំបាច់កំណត់នូវទំហំជាក់លាក់នៃ List របស់យើង និងអាចធ្វើការផ្លាស់ប្តូរទំហំបាននៅកំឡុងពេលដែលប្រើប្រាស់។

២.២. សិក្ខាអំពី LinkedList(ត)

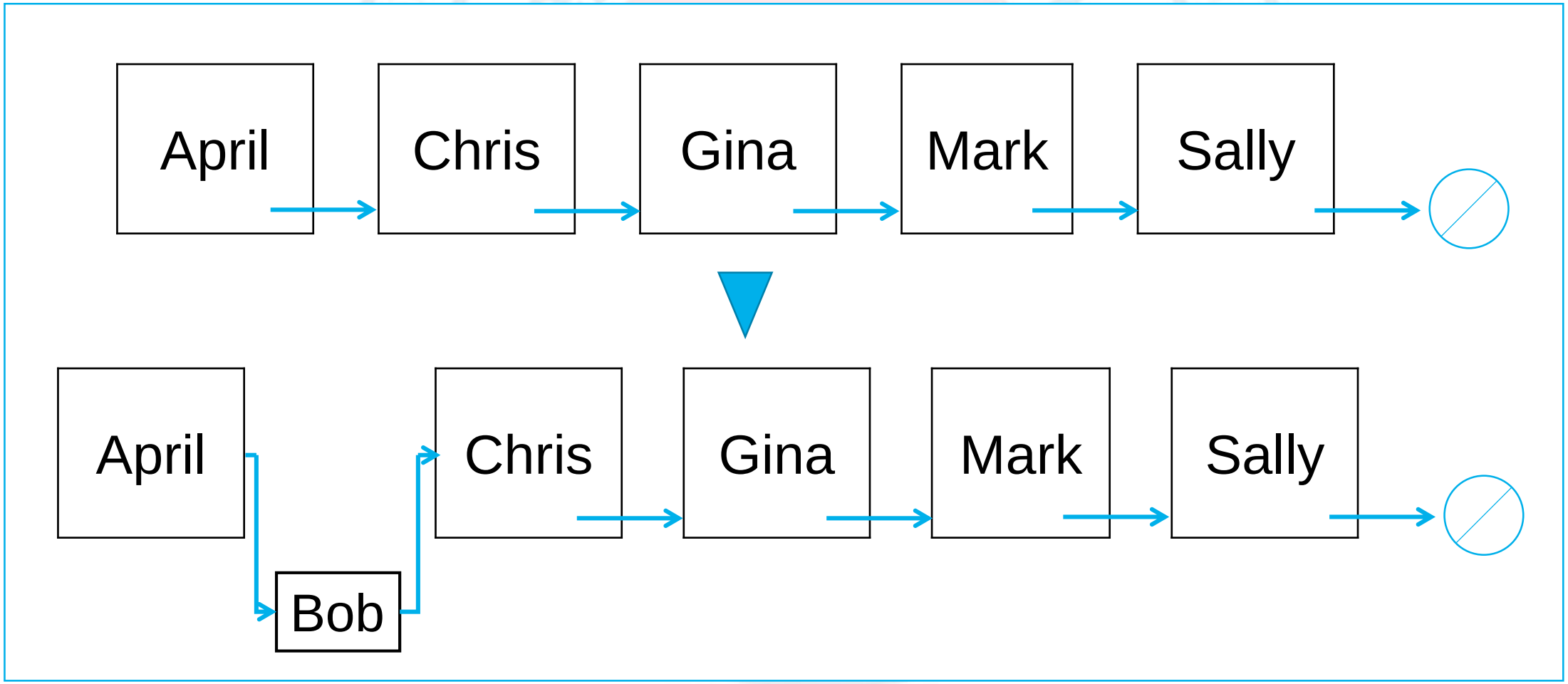
- លក្ខណៈពិសេសរបស់ Linked List:
  - LinkedList Class ក្នុង Java ប្រើប្រាស់ doubly linked list សម្រាប់ store ធាតុរបស់វា។
  - LinkedList Class ក្នុង Java អាចផ្ទុកនូវ ធាតុជាន់គ្នាបាន។
  - ការបញ្ជូលទិន្នន័យមានលក្ខណៈជា ordered
  - យើងមិនចំបាច់កំណត់ធាតុជាក់លាក់សម្រាប់ store ធាតុ ដូច Array ឡើយ។
  - LinkedList class គឺមានលក្ខណៈ non synchronized។
  - LinkedList class មានដំណើរការលឿនចំពោះការ manipulation ព្រោះវាមិនមានការផ្លាស់ប្តូរ element។

២.២. សិក្ខាអំពី LinkedList(ត)

- LinkedList Class មាន Constructors ២:
  - LinkedList( ): បង្កើត Linked list ទទេមួយ គ្មាន parameter ។
  - LinkedList(Collection c): បង្កើត Linked list មួយ ដែលមាន parameter គឺជាតំលៃនៃធាតុនៅក្នុង collection ។

២.២. សិក្ខាអំពី LinkedList(ត)

- ដំណើរការនៃការបញ្ចូលធាតុចូលទៅក្នុង LinkedList



២.២. សិក្ខាអំពី LinkedList(ត)

Example នៃ LinkedList

```
public class LinkedListDemo {
   public static void main(String[] args) {
   LinkedList<String> linkedList=new LinkedList<String>();
   linkedList.add("Phnom Penh");
   linkedList.add("Battambang");
   linkedList.add("Siem Reap");

   Iterator<String> itr=linkedList.iterator();
   while(itr.hasNext())
      System.out.println(itr.next());

   linkedList.clear();
   System.out.println("\n" + linkedList);
   }
}
```

Output:

```
Phnom Penh
Battambang
Siem Reap

[]
```

២.៣. ភាពខុសគ្នារវាង ArrayList និង LinkedList

- ភាពខុសគ្នារវាង ArrayList និង LinkedList

ArrayList	LinkedList
ArrayList ប្រើប្រាស់ dynamic array	LinkedList ប្រើប្រាស់ doubly linked list
ការធ្វើ manipulation ជាមួយ ArrayList គឺយឺត	ការធ្វើ manipulation ជាមួយ LinkedList គឺលឿន
ចំនុចល្អរបស់ ArrayList គឺសម្រាប់ storing និង accessing data.	ចំនុចល្អរបស់ LinkedList គឺសម្រាប់ manipulating data.

៣. ការស្វែងយល់យល់អំពី Map (ត)

- អ្វីទៅជា Map?
  - Map គឺជា object ដែលមាន key ភ្ជាប់ទៅកាន់ value របស់វា ហើយ Value អាស្រ័យទៅនឹង Key របស់វា (Key-Value pair)
  - Map មិនអនុញ្ញាតឲ្យមាន key ស្ទួនទេ
  - រាល់ key នីមួយៗរបស់វាអាចមាន value តែមួយគត់
  - Map ជា Collection មួយជំនួសឲ្យ Dictionary Class មួយដែលគេលែងប្រើ
  - HashMap ជា Class ដែលគេនិយមប្រើនៅចំពោះ Key, Value Pair

៣. ការស្វែងយល់យល់អំពី Map (ត)

- Map មាន Method ដូចជា ៖

Method	Description
Object put(Object key, Object value)	Stores key និង value ក្នុង Map
Object get(Object key)	Returns value តាម key ជាក់លាក់ដែលមាននៅក្នុង Map ប្រសិនបើមិនមានវា return null value
boolean containsKey(Object key)	Returns true ប្រសិនបើមាន Key នៅក្នុង Map.
boolean containsValue(Object value)	Returns true ប្រសិនបើមាន Value មួយយ៉ាងតិចនៅក្នុង Map.
void clear()	Removes all entries ដែលមាននៅក្នុង Map.

៣. ការស្វែងយល់យល់អំពី Map (ត)