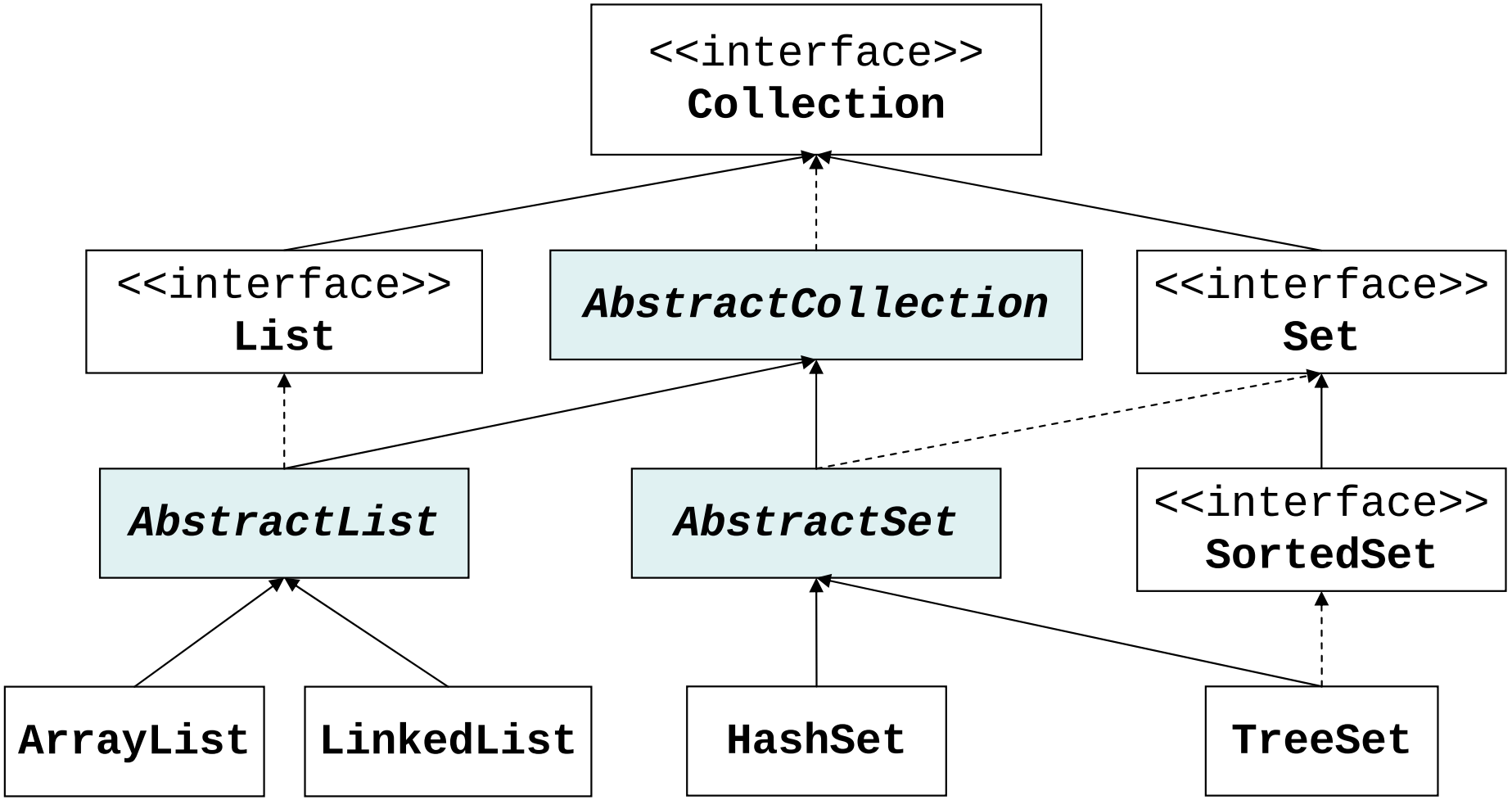
จาวา : Collections Framework

สมชาย ประสิทธิ์จูตระกูล

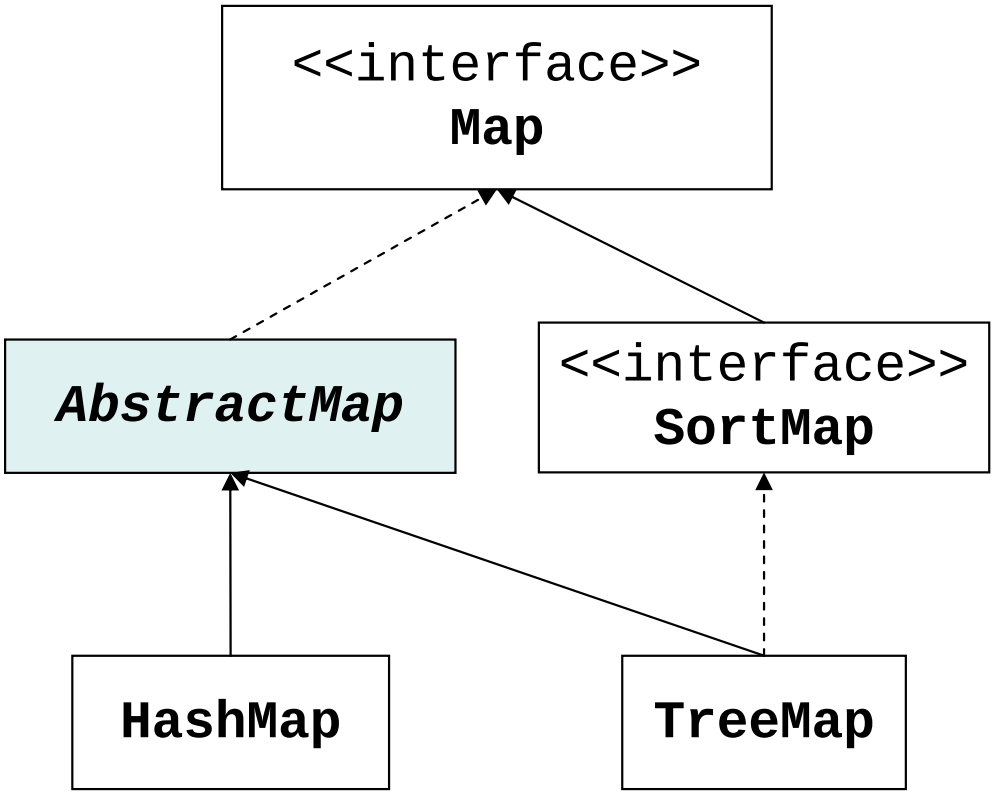
Collections Framework (บางส่วน)



Set เป็น Collection ที่ไม่เก็บตัวซ้ำ

List เป็น Collection ที่เก็บข้อมูลแบบมีอันดับ

Collections Framework



Map เป็นที่เก็บข้อมูลแบบ (key, value)

key ไม่ซ้ำกันใน map

Collection Interface

```
package java.util;

public interface Collection {
    int size();
    boolean isEmpty();
    void clear();
    boolean add(Object element);
    boolean remove(Object element);
    boolean contains(Object element);
    ...
}
```

List Interface

```
package java.util;

public interface List extends Collection {
    Object get(int index);
    Object set(int index, Object element);
    void add(int index, Object element);
    Object remove(int index);
    int indexOf(Object o);
    int lastIndexOf(Object o);
    ...
}
```

List เป็น Collection ที่เก็บข้อมูลแบบมีอันดับ

ตัวอย่างการใช้ List

```
public static void main(String[] args) {
  List c = new ArrayList();
  String[] keywords =
   { "forward", "backward", "left", "right",
     "penup", "pendown", "repeat" };
  for (int i = 0; i < keywords.length; i++) {
    c.add(keywords[i]); c.add(keywords[i]);
  }
  System.out.println(c);
  System.out.println(c.contains("pendown"));
  for (int i = 0; i < c.size(); i++)
    System.out.println(c.get(i));
  c.remove("backward");
  System.out.println(c.contains("backward"));
  c.clear();
  System.out.println(c.contains("repeat"));
}
```

ตัวอย่างการใช้ Set

```
public static void main(String[] args) {
  Set c = new TreeSet();
  String[] keywords =
   { "forward", "backward", "left", "right",
     "penup", "pendown", "repeat" };
  for (int i = 0; i < keywords.length; i++) {
    c.add(keywords[i]); c.add(keywords[i]);
  }
  System.out.println(c);
  System.out.println(c.contains("pendown"));
  System.out.println(c.contains("left"));
  c.remove("backward");
  System.out.println(c.contains("backward"));
  c.clear();
  System.out.println(c.contains("repeat"));
}
```

ตัวอย่างการใช้ SortedSet

```
public static void main(String[] args) throws Exception {
  BufferedReader fi = new BufferedReader(
                              new FileReader("th.txt"));
  String line;
  SortedSet words = new TreeSet();
  BreakIterator boundary =
      BreakIterator.getWordInstance(new Locale("th"));
  while ((line = fi.readLine()) != null) {
    boundary.setText(line);
    int start = boundary.first();
    int end = boundary.next();

    while (end != BreakIterator.DONE) {
      words.add(line.substring(start, end));
      start = end;
      end = boundary.next();
    }
  }
  fi.close();
  System.out.println(words);
}
```

Collection ไม่เก็บ primitive data

- collection เก็บแต่ออปเจกต์
- ถ้าต้องการเก็บ int, double, ... ต้องใช้ Wrapper classes
  - Integer เก็บหนึ่ง int เป็น field ภายใน
  - Double เก็บหนึ่ง double เป็น field ภายใน
  - อื่น ๆ (Float, Short, Byte, Character, Boolean, Long)

```
List list = new ArrayList();
 for (int i = 0; i < 10; i++) {
   int n = (int)(1000 * Math.random());
   list.add(new Integer(n));
 }
 for (int i = 0; i < list.size(); i++) {
   Integer x = (Integer) list.get(i);
   System.out.println( x.intValue() );
 }
```

Map Interface

```
package java.util;

public interface Map {
    int size();
    boolean isEmpty();
    void clear();
    Object get(Object key);
    Object put(Object key, Object value);
    boolean containsKey(Object key);
    boolean containsValue(Object value);
    ...
}
```

Map เป็นที่เก็บข้อมูลแบบ (key, value)

key ไม่ซ้ำกันใน map

Map ไม่ได้ extends Collection แต่อยู่ใน Collections Framework

```
public static void main(String [] args) throws Exception {
  BufferedReader fi = new BufferedReader(new FileReader("th.txt"));
  String line;
  BreakIterator boundary =
      BreakIterator.getWordInstance(new Locale("th"));

  SortedMap words = new TreeMap();

  while ((line = fi.readLine()) != null) {
    boundary.setText(line);
    int start = boundary.first();
    int end = boundary.next();

    while (end != BreakIterator.DONE) {
      String word = line.substring(start, end);

      Object c = words.get(word);
      int cnt = c == null ? 1 : (((Integer) c).intValue() + 1);
      words.put(word, new Integer(cnt));

      start = end;
      end = boundary.next();
    }
  }
  fi.close();
  System.out.println(words);
}
```

Iterator

- ต้องการประมวลผลข้อมูลทั้งหลายที่เก็บใน ArrayList, LinkedList, TreeSet, HashSet, ... หรืออะไรก็ได้ที่เป็น Collection
- ใช้ Iterator เป็นตัวแจกแจงข้อมูลใน collection
- Iterator เป็น interface ทุก ๆ collection มีเมท็อด iterator() ซึ่งคืน iterator object

```
void process(Collection c) {
  Iterator itr = c.iterator();
  while (itr.hasNext()) {
    Object obj = itr.next();
    ...
  }
}
```

```
interface Iterator {
  boolean hasNext();
  Object next();
  void remove();
}
```

Example

```
public static void main(String[] args) {
  String[] s = { "chula.ac.th", "sanook.com",
                 "ith.ith", "hello.co.th" };
  Collection c = new ArrayList();
  for (int i = 0; i < s.length; i++) c.add(s[i]);
  System.out.println(getOnlyTH(c));
}
static Collection getOnlyTH(Collection names) {
  Collection c = new TreeSet();
  Iterator itr = names.iterator();
  while (itr.hasNext()) {
    String name = (String) itr.next();
    if (name.endsWith(".th")) c.add(name);
  }
  return c;
}
```

เมท็อดนี้ทำอะไร ?

Classes

- Set
  - HashSet : เร็วมาก, ใช้ equals, ไม่เป็น SortedSet
  - TreeSet : เร็ว, ใช้ equals compareTo, เป็น SortedSet
- Map
  - HashMap : เร็วมาก, ใช้ equals, ไม่เป็น SortedMap
  - TreeMap : เร็ว, ใช้ equals compareTo, เป็น SortedMap

List	get(idx), set(idx)	itr.remove()
ArrayList	เร็ว	ช้า
LinkedList	ช้า	เร็ว

Lab 6.1 : TreeSet with Collator

- ไปที่คลาส com.somchai.lab6.UniqueWords
- สังเกตผลลัพธ์ที่ได้จากการ run UniqueWords
- ไปที่บรรทัดที่สร้าง `TreeSet( )`
- ให้เปลี่ยนเป็น
  `TreeSet(Collator.getInstance(new Locale("th")));`
- ลอง run UniqueWords ใหม่ แล้วสังเกตผลลัพธ์
- อ่าน API spec.
  - constructor ของ TreeSet( )
  - คลาส Collator

Lab 6.2 : getRange

- ไปที่คลาส com.somchai.lab6.Rational
- เพิ่มเมท็อด getRange

```
public static Collection getRange(
       Collection rNumbers, Rational min, Rational max) {

}
```

- rNumbers คือ collection ของ Rational objects
- สิ่งที่ต้องการ : ให้สร้างและคืน collection ใหม่ที่เก็บ Rational objects ที่มีค่าอยู่ในช่วง min และ max ที่กำหนดให้ (รวมค่าที่เท่ากับ min และ max ด้วย)

ทดสอบ : Run JUnit ของ lab6