

CSE373 Optional Section

Java Collections

11/12/2013

Luyi Lu

Today's Topic

- Java Collection Interface
- Generics Collections Usage
 - Wrapper Class
- CSE 142/143 Collections Review
 - List & Set
 - Stack & Queue
 - Map

Java Collection Interface

- Interface is just a type
- Learn how to look up Java Documentation of the standard library
 - What are the basic operations?
 - <http://docs.oracle.com/javase/7/docs/api/java/util/Collection.html>
- Iterable<E> and Comparable<E>
 - To use for loop and define the rule how object stored is compared
 - Not used in HW5, but good to know

Collections Declaration with Generic Types

- How?
 - `List<Vertex> v = new ArrayList<Vertex>();`
 - `List<Edge> e = new ArrayList<Edge>();`
 - Can we construct `List ArrayList<int>`?
 - **NO!** In Java, anything that is used as generics has to be convertible to `Object`
 - Solution: Use Wrapper Class for primitive types. E.g. `ArrayList<Integer>`
 - Every primitive type has a corresponding wrapper class:
 - **Integer** for `int`,
 - **Double** for `double`,
 - **Character** for `char`,
 - **Boolean** for `boolean`,
 - And so on
 - BTW.. `String` is not a primitive type and `String` objects are immutable
- (Continue on next slide)

Collections Declaration with Generic Types

Autoboxing and Unboxing

```
List<Integer> numbers1 = new ArrayList<Integer>();  
    numbers.add(18);  
    numbers.add(34);
```

- Java will automatically "box" the ints for us (i.e., wrap them up in Integer objects)

```
int product = numbers.get(0) * numbers.get(1);
```

- Java automatically "unboxes" the values for you, unwrapping the Integer objects and giving you the ints that are contained inside.

Lists & Sets

- Difference?
 - Sets don't allow duplicates
 - Client can control order over lists, no index (How to remove?)
 - HashSet doesn't keep order
 - TreeSet keeps things in sorted order
- APIs
 - List: <http://docs.oracle.com/javase/7/docs/api/java/util/List.html>
 - Set: <http://docs.oracle.com/javase/7/docs/api/java/util/Set.html>
- Different Types and their Tradeoffs
 - List
 - ArrayList
 - LinkedList
 - Set
 - HashSet(fast)
 - TreeSet
- Declaration: `LinkedList<E> l = LinkedList<E>();` is not good style
should be `List<E> l = LinkedList<E>();`

You can find all of them
in JAVA API!

Stack & Queue

- Discussed in CSE373
- APIs
 - Stack Class:
<http://docs.oracle.com/javase/7/docs/api/java/util/Stack.html>
 - `Stack<E> s = new Stack<E>();`
 - Queue Interface:
<http://docs.oracle.com/javase/7/docs/api/java/util/Queue.html>
 - `Queue<E> q = new LinkedList<E>();`

Map

- Dictionary that stores key/value pairs
 - One to one relation
- Map<K,V> can have two types. You can have V as Lists or other data structures
 - E.g. Map<String, Set<String>>
- API:
<http://docs.oracle.com/javase/7/docs/api/java/util/Map.html>
- TreeMap and HashMap

Useful Info

ArrayList vs. Linked List

<http://stackoverflow.com/questions/322715/when-to-use-linkedlist-over-arraylist>

HashSet vs. TreeSet

<http://stackoverflow.com/questions/1463284/hashset-vs-treeset>

Some material of this slide is credited to Stuart Reges' CSE143 Notes