
CSE 142

Iterators

5/11/2003

(c) 2001-3, University of Washington

M-1

Outline for Today

- Quick Review
 - ArrayList collections; add, size, get, etc. methods
 - Iteration and while loops
- Today
 - Iterating through collections
 - Iterator objects

5/11/2003

(c) 2001-3, University of Washington

M-2

Using Collections

- We can create ArrayLists, and put things into them.

```
ArrayList names = new ArrayList();
names.add("Bob");
names.add("Sue");
names.add("Jeremiah");
```

- We can pick out elements at particular index positions.

```
String someNames = names.get(0) + " and " + names.get(1);
```

- But how can we do something for all names?

- Print out all names in the list.
- Find the first name, alphabetically.
- Find what the longest name.
- See if a given name is in the list.

5/11/2003

(c) 2001-3, University of Washington

M-3

Iterating The Old-Fashioned Way

- Using what we already know about ArrayLists, we can iterate as follows:

```
for (int rnum = 0; rnum < myList.size(); rnum++) {
    ElementType oneItem = myList.get(rnum);
    // process this item
    ...
}
```

5/11/2003

(c) 2001-3, University of Washington

M-4

Iterating More Elegantly

- **Abstractly, what we really want is to be able to write:**
For all elements in the list,
Do something.
- To get "all elements in the list", we can use an *iterator*
- An iterator is an object that stands as a intermediary between us and the data structure
 - It knows how many items there are in the structure
 - It keeps track of which ones it has passed out
- **Strategy:**
 - Ask the array list for its iterator object.
 - Ask the iterator object for each element, in turn, as part of a while loop.
- **We don't have to know how many elements are in the list!**

5/11/2003

(c) 2001-3, University of Washington

M-5

Iterator Operations

- **Getting an iterator object from an ArrayList (and many other kinds of Java collections):**

```
Iterator iter = names.iterator();
```

- **Here are the methods provided by Iterator:**

```
// Return true if the iteration has more elements.  
public boolean hasNext();
```

```
// Return the next element in the iteration.  
public Object next();
```

5/11/2003

(c) 2001-3, University of Washington

M-6

Using an Iterator, in English

- **General algorithm:**
 - Get the iterator for the collection [names.iterator()].
 - While the iterator has at least one more element [iter.hasNext()],
 - Get the next element [iter.next()].
 - Do something using the element.
 - Then go back to the top.
 - Otherwise, we're done.
- **This is a very common and very important pattern in programming**

5/11/2003

(c) 2001-3, University of Washington

M-7

Using an Iterator, in Java

```
ArrayList names = ...;
```

```
System.out.println("The names are as follows:");
```

```
Iterator iter = names.iterator(); // get the iterator for the collection.
```

```
while ( iter.hasNext() ) { // while there is another element...  
    String name = (String) iter.next(); // get the element (and cast it if needed)  
    System.out.println(name); // do something using the element.  
} // then go back to the top.
```

```
// Otherwise, we're done.
```

5/11/2003

(c) 2001-3, University of Washington

M-8

Relationships Between Objects

5/11/2003

(c) 2001-3, University of Washington

M-9

Another Example: Finding the Longest Name

- Suppose we want to find the longest name in a list. How would we do it?
 - Recall: "Bob".length() == 3
- What's the algorithm in English?
- What's the Java code?

5/11/2003

(c) 2001-3, University of Washington

M-10

Solution

5/11/2003

(c) 2001-3, University of Washington

M-11

Iterators vs Indexed Access

- We can also process an ArrayList using get(index)

```
for (int k = 0; k < names.size(); k++) {  
    process names.get(k);  
}
```

- Tradeoffs

- Iterators are more general – work on all collections, even if the collection doesn't support indexed access (i.e., using get(k) to access elements directly)
- Iterators only support traversal of a collection from beginning to end – if we want to go backwards or in some other order, we need indexed access (and a container that supports it)
- General rule: use iterators (the more general solution) normally; use other traversals when iterators don't do what you need

5/11/2003

(c) 2001-3, University of Washington

M-12