

Generics

Monday, September 26, 2022 1:11 PM

```
public class Node<S>{  
    int first;  
    Node next;  
}
```

```
public class MutableList<T>{  
    Node<T> start; // front of the list  
  
    public void addFirst(intT newItem) {  
        newNode = new Node<T>(newItem, this.start);  
        this.start = newNode;  
    }
```

This particular <T> is optional. Because Java knows that newNode is of type Node<T> when we declared it, we can simply write
newNode = new Node<>(newItem, this.start)

Use this like:

```
MutableList<String> stringList = new MutableList<>();  
MutableList<Integer> intList = new MutableList<>();  
...
```

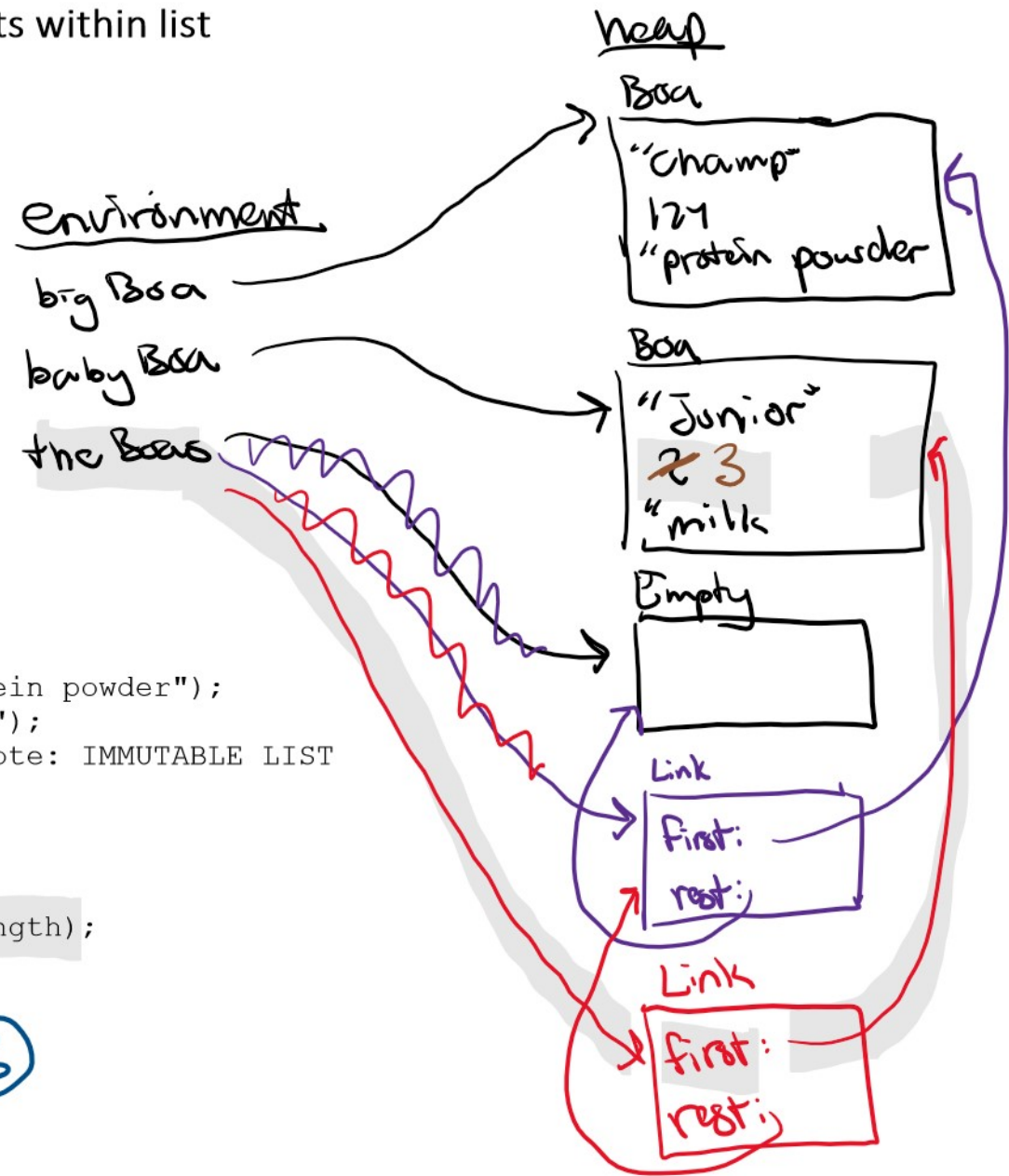
Does list-immutability extend to the contents within list elements?

Monday, September 26, 2022 1:12 PM

```
public class Link implements IList<T> {  
    T first;  
    IList<T> rest;  
    ...  
  
    public Link addFirst(T newBoa) {  
        return new Link(newBoa, this);  
    }  
  
    public T getFirst() {  
        return this.first;  
    }  
}
```

```
public void example() {  
    Boa bigBoa = new Boa("Champ", 124, "protein powder");  
    Boa babyBoa = new Boa("Junior", 2, "milk");  
    IList<Boa> theBoas = new Empty<>(); // Note: IMMUTABLE LIST  
    theBoas = theBoas.addFirst(bigBoa);  
    theBoas = theBoas.addFirst(babyBoa);  
    babyBoa().growBy(1);  
  
    System.out.println(theBoas.getFirst().length);  
}
```

2

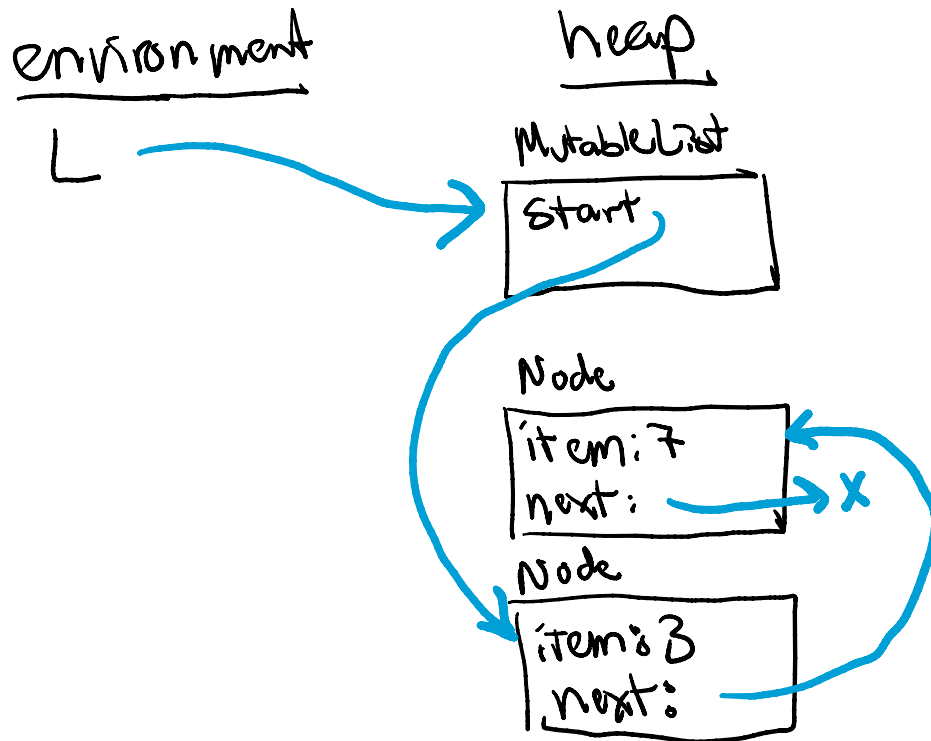


Memory Diagrams with Addresses Explicit

Monday, September 26, 2022 1:13 PM

```
// the list [3, 7]
MutableList<Integer> L = new MutableList<>();
L.addFirst(7);
L.addFirst(3);
```

With arrows:



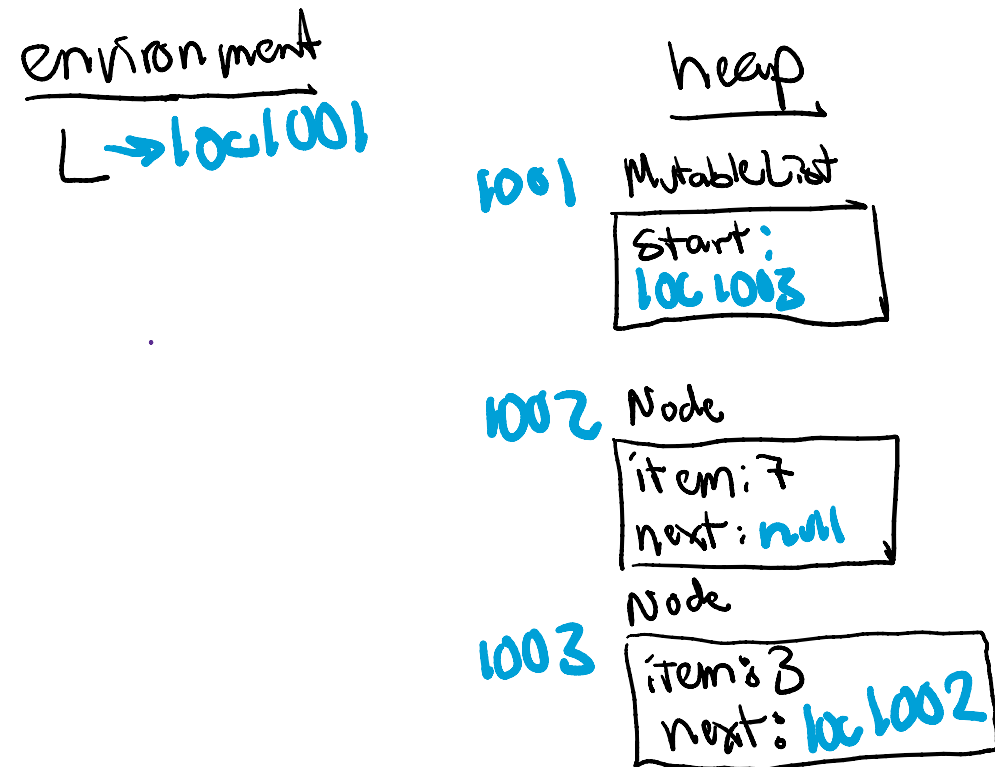
For reference:

```
public class MutableList<T> {
    Node<T> start; // front of the list

    public MutableList() { this.start = null; }

    public void addFirst(T newItem) {
        newNode = new Node<>(newItem, this.start);
        this.start = newNode;
    }
}
```

Same diagram, but drawn by assigning numbers to arrow destinations instead



Rules for drawing memory diagrams with addresses

Monday, October 3, 2022 6:10 PM

- start from a random 4-digit number (ex. 1001)
- number the objects in the order they got created
- replace each arrow w/ its number

For reference:

```
public class MutableList<T> {  
    Node<T> start; // front of the list  
  
    public MutableList() { this.start = null; }  
  
    public void addFirst(T newItem) {  
        newNode = new Node<>(newItem, this.start);  
        this.start = newNode;  
    }  
}
```

(Order of object creation)

```
public void Example2() {  
    ① MutableList<Integer> L = new MutableList<>();  
    L.addFirst(6);  
    ③ Boa teenBoa = new Boa("Scout", 10, "chips");  
    L.addFirst(3);  
}
```

environment

L → loc1001

teenBoa → loc1003

