CS 635 Advanced Object-Oriented Design & Programming
Spring Semester, 2009
Assignment 1 Comments
Feb 5, 2009

Formatting

```
//One can also improperly indent comments to make it hard to see code
        if (foo > bar + 12) { //This is a very tricky piece of code so I will
add a long comment which will wrap around
            foo += sin(bar) / tan(cat) - 13;//this is tricky too so here we
go again with a comment
            while (bar < 10) {//Of course comments are good but
when you do this it is very hard to see the code.
```

Duh

```
Node newNode = new Node(value); //create new node

//main function
public static void main(String[ ] args) {
```

Extract Method to replace comment

```
if (this.head == null && this.tail == null ) { //case 1 - empty list
     this.head = newNode;
     this.tail = newNode;
} else {
     etc.
```

```
if (isEmpty() ) {
     this.head = newNode;
     this.tail = newNode;
} else {
     etc.
```

```
public boolean isEmpty() {
     return (this.head == null && this.tail == null;
}
```


Read Extract method pp 110 -116. Page 110 is basically this example.

Fowler - Extract Method

"I look at a method that is too long or look at code that needs a comment to understand its purpose.

I then turn that fragment of code into its own method."

Comment?

```
public class CustomLinkedList extends DoubleLinkedList {

    /**
     * constructor
     */
    CustomLinkedList() {
        super();
    }
```

Names - Structure

class node { etc }

class dbll { etc }

public void AddNode(String value) {

Java has a naming convention

Follow it

Why is it important?

Names

public void addToList(String value) { etc }

public String getKthElement(int k) { etc }

Java Names

What is the name of the method in Java to

Add an element to a collection?

Retrieve an element from a collection?

Why is it important to use those names?

Java Interfaces

What is the point of Java Interfaces?

What is Wrong with this?

```
public class LinkedList {
    etc
}
```

Exceptions

What exception is thrown in get() method in Java's

- Vector
- ArrayList
- LinkedList
- etc.

Why is it important to do the same?

Long Names

```
public void insertStringInLexicographicalOrderInList(String value) {
     etc.
 }

public void itTurnsOutThatNamesCanBeTooLongUseAShorterNameWhenItExists()



public class OrderedLinkedList extends LinkedList {

    public boolean add(String value) {
    etc.
```

Why return a boolean?

```
public class OrderedLinkedList extends LinkedList {

    public boolean add(String value) {
    etc.
```

Operations?

```
public class Node {
    private String value;
    private Node next, previous;

    public Node() {
        value = next = previous =  null;
    }
```

```
public String getValue() { etc}
public Node getNext() { etc}
public Node getNext() { etc}
public setValue(String aValue) {etc}
public setNext(Node aNode) {etc}
public setPrevious(Node aNode) {etc}
```

Issues?

```
public class A {
        public int x;
        public int y;
        public int z;
}
```

From First lecture

A verses B

```
public class A {
        public int x;
        public int y;
        public int z;
}
```

```
public class B {
        private int x;
        private int y;
        private int z;

        public int getX() { return x;}
        public int getY() { return y;}
        public int getZ() { return z;}
        public void setX(int value) {x = value;}
        public void setY(int value) {y = value;}
        public void setZ(int value) {z = value;}
}
```

From First lecture

Heuristics

Keep related data and behavior in one place

A class should capture one and only one key abstraction

From First lecture

Heuristics

Beware of classes that have many accessor methods defined in their public interface

Do not create god classes/objects in your system

Beware of classes that have too much noncommunicating behavior

From First lecture

What is the Abstraction?

```
public class LinkedList {
    private Node head;
    private Node tail;

    public void printStringsWithVowels {
        current = head;
        while (current != tail) {
            System.out.println(current.value);
            etc.
}

public void printOddLengthStringsReverseOrder {
        etc
}
etc.
```

Information Hiding

```
public class LinkedList {
    private Node head;
    private Node tail;

    public Node get(int index) {
        etc.
    }
```

Temporary Field

```
public class LinkedList {
    private Node head;
    private Node tail;
    private Node current; //when inserting a node

    public Node add(String value) {
        current = head;
        while (current != tail) {
            etc.
}
```

Helper Methods

```
public class LinkedList {
    private Node head, tail;
    private boolean flag;

    public void checkValue(String value, LinkedList list) {
        Node temp = new Node(value);
        int k = 1;
        flag = false'
        while (flag == false) (
            Node tempList = list.getElement(k);
            String cpmstr = tempList.value();
            String invokingstr =temp.value();
            if ((invokingstr).compareTo((cpmstr) > 0 ) {
            add(k, temp.value());
                flag = true;
            }
            else { k++}
}
```

Helper Method

```
public class LinkedList {
    private Node head, tail;

    private boolean startWithVowel(String value) {
        look at first character to see if it is a vowel
    }
```

System.out.println()

```
//In linked list class
public void getKthElement(int k ) throws DoubleLinkedException {
    some code to find the k'th element

    System.out.println(theKthElement);
}

//In node class
public void setValue() {
    System.out.println("Please enter a string:");
    code to read the response
}
```

Without System.out.println

```
public class LinkedList {
    private Node head;
    private Node tail;

    public ArrayList<String> stringsWithVowels {
            Add the strings to the array list
            Once the caller has the array list they can do many things
                with the strings, including print them out
    }

    public ArrayList<String> oddLengthReverseOrder {
            Same here
    }

    public Object[] toArray() {
        Even better. This method is standard part of the Java collection
        classes. Dump the two methods above and use this one.
    }
```

System.out

Good for debugging

When you don't have a good debugger

Good for Unix/Linux commands

Java is too big to use In this case

Why waste time with Menu systems?

```
public void Menu() {
    while(true) {
        System.out.println( "Please select your choice.");
        System.out.println( "1. display all node in the list");
        System.out.println( "2. Display strings starting with vowels");
        System.out.println( "3. Display odd length strings back to front");
        etc
```