CS 535 Object-Oriented Programming & Design
Fall Semester, 2008
Assignment 4 Comments
Nov 18 2008

Over Complex code

```
at: aKey
    | |
    (root = nil) ifTrue: [
    "Throw exception as tree is empty"
    BinaryTreeKeyNotFoundError raiseSignal.
    ^nil.
    ] ifFalse: [
        "Recurse into the tree to find the key"
        ^root at: aKey].
   ^nil
```

```
at: aKey
    root ifNil: [BinaryTreeKeyNotFoundError raiseSignal].
    ^root at: aKey
```

Duh Comments

```
"Get next line"
line := srcFile getNextLine.
"Calculate sum and write result"
targetFile nextPutAll: line getSum printString.
"go to next row"
targetFile nextPut: Character cr.
```


In Smalltalk we would use nextLine not getNextLine

Comments?

```
size
    "Returns the number of nodes in the tree"

    ^size

size1
    ^size
```

Why?

```
BinaryNode class>>new
	"Answer a newly created and initialized instance."

	^super new initialize

BinaryNode>>initialize
	"Initialize a newly created instance. This method must answer the receiver."

	" *** Edit the following to properly initialize instance variables ***"
	left := nil.
	right := nil.
	key := nil.
	value := nil.
	" *** And replace this comment with additional initialization code *** "
	^self
```


Constructor need to return objects that are usable. This one does not.

Why have Constructors that are not used?

```
BinaryNode class>>key: aKey value: anObject
    ^super new setKey: aKey value: anObject

BinaryNode class>>new
    Same as last slide

BinaryNode>>initialize
    same as last slide
```


Even worse is that another constructor was created and it is the one that has to be used. The one auto-generated one is not meant to be used, but here it is. And by the way it is the standard (BinaryNode new) way to create objects.

Eclipse & Autogenerated Code

```
public class IgnoredCode {

	public IgnoredCode() {
		// TODO Auto-generated constructor stub
	}

	/**
	 * @param args
	 */
	public static void main(String[] args) {
		// TODO Auto-generated method stub
	}
}
```

Google "TODO Auto-generated"

130,000 hits

What does this say about

People reading comments?

People keeping comments up to date?

Not an Exception

```
BinaryTreeKeyNotFoundError class>>raiseWith: aKey
    Transcript
        show: aKey printString;
        show: " not found"
        cr
```

How not to use Exceptions

```
TreeNode>>at: aKey
    [aKey = key ifTrue: [^value].
    aKey < key ifTrue: [^left at: aKey].
    aKey > key ifTrue: [^right at: aKey]]
        on: BinaryTreeKeyNotFoundError
        do: [^'Key not found']
```

Don't Duplicate Code

```
HtmlTable>>row: rowIndex column: columnIndex
    ^elements at: (((rowIndex - 1 ) * column) + columnIndex

HtmlTable>>asHtml
    blah
    blah
    some loop

        some other loop

            htmlString nextPutAll:
                (elements at: (((rowIndex - 1 ) * column)  + columnIndex)
```

use

```
htmlString nextPutAll: (self row: rowIndex column: columnIndex)
```

Duplicate Logic

```
BSTree>>at: aKey
    (size = 0)
        ifTrue: [BinaryTreeKeyNotFoundError raiseSignal: aKey].
        ifFalse: [^root at: aKey]

BSTree>>at: aKey ifAbsent: aBlock
    (size = 0)
        ifTrue: [^aBlock value].
        ifFalse: [^root at: aKey]
```

Put Logic in one Place

```
BSTree>>at: aKey
    ^self at: aKey ifAbsent: [BinaryTreeKeyNotFoundError raiseSignal: aKey].

BSTree>>at: aKey ifAbsent: aBlock
    (size = 0)
        ifTrue: [^aBlock value].
        ifFalse: [^root at: aKey]
```

Information Hiding (not)

```
BinarySearchTree>>root
    ^root
```

```
BinaryNode>>left
    ^left
```

Can You spot the errors?

```
TreeNode>>at: object1 put: object2
    (key == nil)
        ifTrue:
            [key := object1.
            value := object2]
        ifFalse:
            [(key > object1)
                ifTrue:
                    [(left == nil)
                        ifTrue: [left := TreeNode key: object1 put: object2]
                        ifFalse: [left at: object1 put: object2]]
                ifFalse:
                    [(right == nil)
                        ifTrue: [right := TreeNode key: object1 put: object2]
                        ifFalse: [right at: object1 put: object2]]]
```


The code is overly complex, which makes it prone to errors and make the errors harder to see. Unbalanced BST are very simple compared to code you will get paid to write. If you are making code this complex in simple situations what is going to happen with really complex stuff?

How about now?

```
TreeNode>>at: object1 put: object2
    (key == nil)
        ifTrue:
            [key := object1.
            ^value := object2].
    (key > object1)
        ifTrue:
            [(left == nil)
                ifTrue: [^left := TreeNode key: object1 put: object2]
                ifFalse: [^left at: object1 put: object2].
(right == nil)
    ifTrue: [^right := TreeNode key: object1 put: object2]
    ifFalse: [^right at: object1 put: object2]
```

How about now?

```
TreeNode>>at: aKey put: anObject
    (key > aKey) ifTrue: [^left at: aKey put: anObject].
    ^right at: aKey put: anObject

NilNode>>at: aKey put: anObject
    parent addChildKey: aKey value: anObject

TreeNode>> addChildKey: aKey value: anObject
    key > aKey
        ifTrue: [left := TreeNode key: aKey value: anObject]
        ifFalse: [right := TreeNode key: aKey value: anObject]
```

Using Globals

```
Smalltalk defineClass: #BinaryNode
	superclass: #{Core.Object}
	indexedType: #none
	private: false
	instanceVariableNames: 'left right key value '
	classInstanceVariableNames: ''
	imports: ''
	category: ''


Smalltalk.BinaryNode defineSharedVariable: #Size
	private: false
	constant: false
	category: 'initialize-release'
	initializer: nil
```


What happens if we have two different trees?

How many tables can exist?

```
Smalltalk defineClass: #HtmlTable
	superclass: #{Core.Object}
	indexedType: #none
	private: false
	instanceVariableNames: ''
	classInstanceVariableNames: ' elements numberOfRows numberOfColumns'
	imports: ''
	category: ''
```

Local declared as instance

```
Smalltalk defineClass: #HtmlTable
    superclass: #{Core.Object}
    indexedType: #none
    private: false
    instanceVariableNames: 'row elements numberOfRows numberOfColumns'
    classInstanceVariableNames: ''
    imports: ''
    category: ''
```

```
HtmlTable>>setRow: aRows columns: aColumns
    numberOfRows := aRows.
    numberOfColumns := aColumns.
    elements := Array new: aRows.
    1 to: aRows do: [:each |
        row := Array new: aColumns.
        elements at: each put: row]
```


row is only used as a local variable in one method. It will be left in some random state, which is not good. Being declared as a instance variable tells us that it is part of the state of the object. At the least it will cause readers time to figure out that it is a local variable. In a complex class that will not happen. At the worst people will using the instance variable.

Loops

```
counter := 1
[counter < numberOfColumns]
    whileTrue: [
        blah
        blah
        counter := counter + 1]
```

```
1 to: numberOfColumns do: [:counter |
    blah
    blah]
```

Names

```
HtmlTable>>htmlForRow: rowIndex column: columnIndex
    | elementAtRowIndexAndColumnIndex |
    elementAtRowIndexAndColumnIndex := self atRow: rowIndex column: columnIndex.
    ^'<td>' , elementAtRowIndexAndColumnIndex , '</td>'
```

Names

tree1
tree1
each1

HtmlTable>>Rows: anInteger

BinaryNode>>setvalue: anobject

temp?

tmpString
tempRow
tempBuildString

All variables in all your programs are temporary

So "temp" and "tmp" have no meaning

So stop wasting

- Your time
- Readers of your programs

With the meaningless "temp" prefix

If that is too much to ask

Please stop wasting my time with "temp"

In case I have not gotten your attention

You will lose points in each assignment that I find "temp"

If you are still not convinced get the book

Code Complete

and read what it says about "temp"

and variable names in general

Helper Methods

```
FileStream class>>rowSumFrom: inputStream to: outputStream
    | line |
    line := OrderedCollection new.
    (inputStream upTo: Character cr)
        runsFailing: [:each | each = $,] do: [:each | line add: each].
    self writeLineSumOf: line to: outputStream
```

```
FileStream class>> writeLineSumOf: line to: outputStream
    outputStream
        nextPutAll: (line inject: 0 into: [:sum :each | sum = each asNumber]) printString
        cr
```


The methods do not use any instance variables. They only use their arguments, so they could be in any class.

Any Code reuse?

```
String>>getSum
    "Calculates the sum of the numbers in a string separated by ',' "
    | tokens sum |
    tokens := self tokensBasedOn: $,.
    sum := 0.
    tokens do: [:each | sum := sum + each asNumber].
    ^sum
```


The prefix "get" is not used in Smalltalk

Is the intent clear?

```
inputStream := inputFile asFilename readStream
outputStream := outputFile asFilename writeStream

[inputStream atEnd]
    whileFalse: [ | line words sum |
        line := inputStream nextLine.
        words := line tokensBaseOn: $,
        numbers := words asNumbers.
        sum := numbers sum.
        outputStream
            nextPutAll: sum printString;
            cr]
```

Methods Needed

ReadStream>>nextLine

Collection>>asNumbers

Collection>>sum

Any of these methods likely to get reused?

What is the performance difference?

```
input := 'start' asFilename readStream.
output := 'end' asFilename readStream.
[input atEnd] whileFalse:
    [ | next |
    (next := input next) = $,
        ifTrue: [output nextPut: $.]
        ifFalse: [output nextPut: next]]
output close.
input close.
```

```
input := 'start' asFilename readStream.
output := 'end' asFilename readStream.
result := input content replaceAll: $, with: $..
output                  : result.
output close.
input close.
```

Formating

```
input := 'start' asFilename readStream.
output := 'end' asFilename readStream.
[input atEnd] whileFalse:
[ | next |
(next := input next) = $,
ifTrue: [output nextPut: $.]
ifFalse: [output nextPut: next]]
output close.
input close.
```

Issues

Missing code
Code that can't run
Formating problems (copy paste issues???)
Students spending lot of time copy and pasting code into Documents
Same code used by several students

Solution

All future assignment will be turned in electronically
Do not email me your assignments
Place your assignments in your Store repository

Repository Problems

I can't get code into the repository

- Practice before your assignment is due

How do I know all my code is in the repository?

- Up load the code
- In clean image download the code & check

How do I connect to the repository

- Covered in lecture
- Screen cast this weekend
- Handed out your repository information earlier