Lab2

Indentation

Filename

Class

Class header template

/\*\*

[Class description. The first sentence should be a meaningful summary of the class since it

will be displayed as the class summary on the Javadoc package page.]

[Other notes, including guaranteed invariants, usage instructions and/or examples, reminders

about desired improvements, etc.]

@author <A HREF="mailto:[Email address]">[Name]</A>

@version $Revision: 1.1.1.1 $ $Date: 2014/08/09 15:15:25 $

@see [String]

@see [URL]

@see [Class name#method name]

\*\*/

In addition to the class header, each class should contain the following string constant. This constant identifies the version of the class using PVCS keywords. This allows easy determination of the version level of classes deployed in a Jar file or as stand-alone classes.

/\*\* Embedded PVCS revision string \*\*/

public static final String REVISION = "$Revision: 1.1.1.1 $";

Object

Variables

Global local and constant variables

Use *javadoc* conventions to describe the nature, purpose, constraints, and usage of static variables and instance variables. Every variable should be documented. In addition, constants should be cleary documented, including the rationale for the value of the constant. Use HTML format, with added tags:

* @see *string*
* @see *URL*
* @see *classname#methodname*

Example:

/\*\*

\* The current number of elements.

\* must be non-negative, and less than or equal to capacity.

\*\*/

protected int myCount;

Comments

Ex:Write all /\*\* ... \*\*/ comments using *javadoc* conventions. (Even though not required by javadoc, end each /\*\* comment with \*\*/ to make it easier to read and check.)

Preface each class with a /\*\* ... \*\*/ comment describing the purpose of the class, guaranteed invariants, usage instructions, and/or usage examples. Also include any reminders or disclaimers about required or desired improvements.

## Naming Conventions

packages

lowercase.   
All package names should start with "PITT"

files

The java compiler enforces the convention that file names have the same base name as the public class they define.

classes:

CapitalizedWithInternalWordsAlsoCapitalized

Exception class:

ClassNameEndsWithException.

Interface. When necessary to distinguish from similarly named classes:

InterfaceNameEndsWithIfc.

Class. When necessary to distinguish from similarly named interfaces:

ClassNameEndsWithImpl

constants (finals):

UPPER\_CASE\_WITH\_UNDERSCORES

private or protected:

myVar (i.e. prefix with my)

static private or protected:

ourVar

local variables:

firstWordLowerCaseButInternalWordsCapitalized

methods:

firstWordLowerCaseButInternalWordsCapitalized()

method parameters:

inParam, outParam, or inOutParam. If the method reads the parameter but does not modify its contents, prefix the parameter with in. If the method receives an empty parameter and sets its contents as a return value, prefix the parameter with out. If the method reads the contents of a parameter and modifies them, prefix the parameter with inOut.

factory method for objects of type X:

newX

converter method that returns objects of type X:

toX

method that reports an attribute x of type X:

X getX()

method that changes an attribute x of type X:

void setX(X value)