A Quick Introduction to Javadoc

1. What is Javadoc?

Traditionally, when you write a program you must write a separate document recording how the program works. When the program changes, the separate documentation must change as well. For large programs that are constantly being changed and improved, it is a tedious and error-prone task to keep the documentation consistent with the actual code.

Javadoc is a tool that helps with this problem. Instead of writing and maintaining separate documentation, the programmer writes specially-formatted comments in the Java code itself. Javadoc takes these comments and transforms them into documentation in HTML (web page) format. For a short example of documentation generated by Javadoc, see the external documentation for our sample program. Another good example of a program with Javadoc comments is the SavitchIn class, which is printed in an appendix to the text and also included in the CD which accompanies the text. Here is the output you get when you run Javadoc on SavitchIn. For a larger, real-life example of documentation generated by Javadoc, see the Java 1.3 API documentation.

You should have a Javadoc comment immediately before each class, method and top-level variable (that means a global or instance variable, not a local variable within a method). Your Javadoc comment at the beginning of the "main" class file should describe the entire program.

2. General format of Javadoc Comments

All Javadoc comments have the following format:

```
/**
 * Summary sentence.
 * More general information about the
 * program, class, method or variable which
 * follows the comment, using as many lines
 * as necessary.
 *
 * zero or more tags to specify more specific kinds
 * of information, such as parameters and return
 * values for a method
 */
```

You start out with the normal beginning of comment delimiter (/*) followed by another (*). All following lines start with an asterisk lined up under the first asterisk in the first line. The last line contains just the normal end of comment delimiter (*/). The first sentence is a "summary sentence". This should be a short description of the program, class, method or variable that can stand on its own. Even though it's refered to as a "summary sentence", it is often a phrase rather than a complete sentence, as in the example below. It can extend over several lines if needed; just don't include any blank lines. (It ends with the first period, so you must avoid using abbreviations such as "e.g." in this sentence!) Following the summary sentence, you may include more sentences to give more details; again, don't include any blank lines. Following this general description, there should be a blank line, then a sequence of

"tags". Different tags will be appropriate for different situations. Here is an example of a Javadoc comment without tags which describes the variable declared immediately below it:

```
/**
 * The number of students in the class.  This variable must not be
 * negative or greater than 200.
 */
public int numStudents;
```

Note: You still can and should have comments within methods, describing local variables and the computing going on inside the methods. There is, however, no Javadoc format for these comments. Use // or /*..*/, whichever you prefer.

3. Tags

The general form of a tag is:

* @name comment

where the name of the tag specifies what kind of information we're giving and the comment gives the information. For example, the author tag is used to tell us who wrote a class or program, as in

* @author William Shakespeare

Each tag should start on a new line. The tag comments can extend into multiple lines if necessary, but there should be no blank lines in between tags.

3.1. Tags for Classes

In a Javadoc comment before a class definition, you should have an author tag, specifying who wrote the program.

@author your full name, as it will appear in a class list (no nicknames, please), plus your lab section

For example:

@author Jane Smith, lab X

If you worked with a partner, make an author tag for each one of you, on separate lines.

3.2 Tags for Methods

For each parameter to a method, include a param tag:

@param <name of parameter> short description of parameter

For example,

* @param size number of elements in the array

If a method has several parameters, the param tags should appear in the same order as the parameters are declared. A method with no parameters will have no param tags.

After the param tag(s), if any, include a return tag unless your return type is void (no return value):

@return short description of return value

For example:

* @return true if the value was found in the array

If your method throws an exception, you should also include an exception tag:

@exception <name of exception> description of circumstances under which the exception is thrown

For example:

```
    * @exception NumberFormatException raised if the user's input is not in valid integer
format
```

If your method throws more than one exception, they should appear in alphabetical order by exception name. Exception tags should follow param and return tags.

3.3 Tags for Variables

A Javadoc comment before a variable declaration needs no tags. See Section 2 for an example.

4. Running Javadoc

Javadoc is a DOS program without a fancy user interface. The use of a batch file, however, makes it quite simple to run.