CISC 370: Javadoc

Sara Sprenkle

June 13, 2006

Javadoc

- Tool for generating API documentation in HTML format from comments in source code
  - API documentation: describes a class and how to use the class
- Online Java classes API documentation was generated using javadoc on the standard Java library source code tree

javadoc Comments

- javadoc creates documentation from javadoc comments
- A javadoc comment looks like

```
/**
    javadoc comment
*/
```

  - Note the two asterisks for the opening

Items to Document

- javadoc supports documentation of the following items:
  - Packages (we haven't looked at these yet)
  - Classes and interfaces
  - Methods
  - Fields
- And, you *should* be documenting each of these items!
- Don't worry about interfaces; more about them later

Comment Location

- A javadoc comment comes immediately before (above) the feature it describes.

```
/**
 * class description
 */
public class Hello
{
     public static void main(String[] args)
     {
          System.out.println(“Hello”);
     }
}
```

Comment Content

- Free-form text
  - contains any text you wish
    - Can use HTML tags to make text more readable
  - provides a summary of the purpose/function of the feature it describes.
  - first sentence should be a concise summary of the feature
    - How would you describe it in one sentence?
- Optional set of tags, which start with '@' symbol
  - Each documentable item has a set of valid tags

Class Comments

- Valid tags
  - @author <name>
  - @version <version text>
  - @since <version that this first appeared in>
  - @deprecated <indicates this is deprecated; text describes a suggestion as an alternative>
  - @see <link>

```
/**
 * Hello is an illustrative class for CISC370.
 * @author Sara Sprenkle
 * @version 1.0
 */
public class Hello
```

Method Comments

- Valid tags
  - @param <variableName> <description>
  - @return <description>
  - @throws <description of an exception this method may throw>

```
/**
   Raises the salary of an employee.
   @param byPercent the percentage to raise the
    salary by.  For example, 10 = raise by 10%
   @return the amount the salary was raised by
*/
public double raiseSalary(double byPercent)
{
   ...
```

Field/Variable Comments

- Typically, only public or protected fields are commented
  - Javadocs are meant to describe the interface and use of your classes
  - Can use javadoc comments for private fields for internal use

```
/**
 * The Hearts playing-card suite
 */
public static final int HEARTS = 1;
```

Package Comments

- To document a package, create a file named `overview.html` and place it in the same directory as the package's source files
- All text between the `<body>` and `</body>` HTML tags is extracted as the package comments.

Generating javadoc with command-line utility

- `javadoc` executable should be in same location as `javac` and `java`
- Many options and customizations available
  - -d: destination directory
  - -private: include private methods and fields
    - Default is …
  - See all options with `man javadoc` or in online documentation

Generating javadoc with command-line utility

- `javadoc -d <doc_dir> *.java`
  - Generate javadoc for all .java files in current directory; save results in directory `doc_dir`
- `javadoc -d <doc_dir> package_name`
  - Generate javadoc for package `package_name`; save in directory `doc_dir`

Generating javadoc in Eclipse

- Select `Project -> Generate Javadoc`
- May be easier to customize Javadocs using an IDE

Javadoc Summary

- The documentation generated by javadoc is very simple to use and is frequently provided with Java classes.
- It is a good idea to insert javadoc comments in the places discussed previously.
  - helps others (and yourself) comprehend your code and its intended function.
- You should be doing this level of commenting anyway – so do it in javadoc format!