Lecture - Javadoc

Introduction to javadoc

Unlike a language such as C++, Java places both the interface and the implementation of a class in the same file.

However, Java has a program called `javadoc` that automatically extracts the interface from a class definition and produces documentation.

This information is presented in HTML format, and can be viewed with a Web browser.

If a class is correctly commented, a programmer need only refer to this *API (Application Programming Interface)* documentation in order to use the class.

`javadoc` can obtain documentation for anything from a single class to an entire package

> This is similar to python's Docstrings. However, Java's provides greater control over the formatting of the documentation, and requires compiling.

Commenting classes for javadoc

The javadoc program extract class headings, the headings for some comments and the headings for all public methods, instance variables and static variables.

In the normal default mode, no method bodies or private items are extracted.

To extract a comment, the following must be true:

- The comment must come *immediately before* a public class or method definition, or some other public item.
- The comment must be a block comment, and the opening `/*` must contain and extra `*` (`/**` ... `*/`)

Note: Extra options can be set in order to extract line comment (//) and private items.

In addition to any general information, the comment preceding a public method definition should include descriptions of parameters, any value returned, and any exceptions that might be thrown.

```
/**
  General comments about the method ...
  @param aParameter    Descrition of aParameter
  @return     What is returned
  ...
*/
```

@ Tags

@ tags should be placed in the following order

- @param Parameter_Name Parameter_Description
- @return Description_Of_Value_Returned
- @throws Exception_Type Explanation
- @deprecated
- @see Package_Name.Class_Name
- @author Author
- @version Version_Information

If there are multiple parameters, each should have its own @param on a separate line, and each should be listed according to its left-to-right order on the parameter list

If there are multiple authors, each should have its own @author on a separate line.

Running javadoc

To run `javadoc` on a package, give the following command:

```
javadoc -d Documentation_Directory Package_Name
```

The HTML documents produced will be placed in the in the `Documentation_Directory`.

If the `-d` and `Documentation_Directory` are omitted, `javadoc` will create suitable directories for the documentation.

To run `javadoc` on a single class, give the following command from the directory containing the class file:

```
javadoc ClassName.java
```

To run `javadoc` on all the classes in a directory, give the following command instead:

```
javadoc *.java
```

Options for Javadoc

The following are some of the many command line options for javadoc.

`-link Link_To_Other_Docs`

Provides a link to another set of documentation. This is usually used with either a path name to a local verion of th java documentation, or the URL of the Oracle web site with the official java documentation.

`-d Documentation_Directory`

Specifies a directory to hold the documentation generated. It may be a relative or absolute path name.

`-classpath List_Of_Directories`

Overrides the CLASSPATH environment variable and makes List_Of_Directories the CLASSPATH for the execution of this call of javadoc. It does not permanently change CLASSPATH.

`-author`

Include author information (from @author tags) in the documentation.

`-version`

Includes version information (from @version tags) in the documentation.

`-private`

Includes private members as well as public members in the documentation.

JavaDoc: Bill

This code slide does not have a description.