Javadoc

Intro

- A JDK tool to generate web pages based on comments in Java source code files.

- Produces HTML pages describing most classes, interfaces, constructors, methods, and fields
    - not anonymous inner classes
    - not private fields (can be overridden by command line option)

- Generates API documentation or implementation documentation for a set of source files.

Javadoc Comments

- Comment separators
  - /** begins a Javadoc comment block
  - */ ends a Javadoc comment block
- Comment blocks appear before (above) class, interface, constructor, method, field in Java source code file
- Comments begin with a brief, summary sentence of the field, method, or class.

Javadoc Comment Example

```
/**
 * A Fork object. A fork can alternate between two states: being picked up
 * and being put down. If it is currently picked up, it must be put down
 * before it can be picked up again. Similarly, if a fork is put down, it
 * must be picked up before it can be put down again.
 *
 * @author david
 */
public class Fork {
        /** id for this fork */
        private int id;

/** flag to indicate if this fork can be picked up */
private boolean availableFlag = true;

/** part of an exception message */
private static final String pickUpMsg = " is not available";

/** part of an exception message */
private static final String putDownMsg = " is not being held";

/**
 * Creates a Fork with a specified id.
 *
 * @param id the specified id
 */
public Fork(int id) {
        this.id = id;
}
```

Package hw2

Class Fork

java.lang.Object
    hw2.Fork

```
public class Fork
extends java.lang.Object
```

A Fork object. A fork can alternate between two states: being picked up and being put down. If it is currently picked up, it must be put down before it can be picked up again. Similarly, if a fork is put down, it must be picked up before it can be put down again.

Constructor Summary

Constructors

Constructor	Description
`Fork(int id)`	Creates a Fork with a specified id.

Method Summary

All Methods | **Static Methods** | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method	Description
`boolean`	`isAvailable()`	Tests whether this Fork is available.

Javadoc Tags

- Tags (ex @author) identify particular attributes of the item being documented

- Tags are optional

- Javadoc may print a warning message if you omit some tags.

Common Tags

- @author (classes and interfaces only)
- @version (classes and interfaces only)
- @param (methods and constructors)
- @return (methods only)
- @exception (aka @throws)
- @see
- @since
- @deprecated

Directory Structure

- bin/ - scripts, maybe input files
- build/ - Compiler output (.class files)
- docs/ - Javadoc output
- lib/ - Third party .jar files
- src/ - Java source files

Command Line

With a hw2 package
$ javadoc -sourcepath src -d docs hw2

Without a hw2 package but with a hw2 subdirectory.
$ javadoc -sourcepath src/hw2 -d docs

Ant script (1)

```
<!--

    build.xml
    A sample Ant script for CS364

    David Apostal
-->
<project name="CS364" default="compile" basedir=".">
<!-- define properties to be used later -->
<property name="src.dir" location="src" />
<property name="build.dir" location="build" />
<property name="lib.dir" location="lib" />
<property name="docs.dir" location="docs" />

<path id="project.classpath">
   <fileset dir="${lib.dir}" >
      <include name="*.jar" />
   </fileset>
</path>
```

Ant script (2)

```
<!-- delete existing directories (and contents) -->
<target name="clean" description="delete build artifacts">
    <delete dir="${build.dir}"/>
    <delete dir="${docs.dir}" />
</target>

<!-- create output directory -->
<target name="init">
    <mkdir dir="${build.dir}"/>
    <mkdir dir="${docs.dir}" />
</target>
```

Ant script (3)

```
<!-- build the source code -->
<target name="compile"
        description="compile project source code"
        depends="clean, init">
    <javac srcdir="${src.dir}" destdir="${build.dir}" >
        <classpath refid="project.classpath" />
    </javac>
</target>
```

Ant script (4)

```
<target name="gendocs"
        description="generate documentation"
        depends="compile">
    <javadoc sourcepath="${src.dir}" destdir="${docs.dir}" >
        <classpath refid="project.classpath" />
    </javadoc>
  </target>
</project>
```

Final Task

- Copy activemq-all-5.15.8.jar to your lib/ folder.

- Needed for HW3