

# CS-259

## Data Structures with Java

Javadoc



Instructor: **Joel Castellanos**

e-mail: [joel@unm.edu](mailto:joel@unm.edu)

web: <http://cs.unm.edu/~joel/>

Course website:

<http://cs.unm.edu/~joel/cs259/>

Field Summary	
<code>private int</code>	<code>numberOfInstances</code> Represents the number of instances of this class.

  

Constructor Summary	
<code>private java.lang.Date()</code>	Nonarg constructor will simply pass a new Date object to the Date constructor.
<code>private java.lang.Date(int year, int month, int day)</code>	Constructor used to create this object.

  

Method Summary	
<code>private java.lang.Date</code>	<code>getDate()</code> Returns the Date Time this object was created.
<code>private void</code>	<code>wait(long millis, int nanos)</code> Sole entry point to the class and application.

  

Methods inherited from class java.lang.Object	
<code>clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait</code>	

1

## Javadoc

- Javadoc is a documentation generator from Sun Microsystems for generating API documentation in HTML format from Java source code.
- The "doc comments" format used by Javadoc is the de facto industry standard for documenting Java classes.
- Some IDEs, such as Netbeans and Eclipse will automatically generate Javadoc HTML.
- Many file editors will assist the user in producing Javadoc source and will use the Javadoc info as internal references for the programmer.

## Major Javadoc Structures

- Package Description
- List of Classes
  - Class Name
  - Class Inheritance Hierarchy
  - Summary and Usage Description
  - Field Summary
  - Constructor Summary
  - Method Summary
  - Field Detail
  - Constructor Detail
  - Method Detail

-3-

## JavaDoc Comments

- Javadoc html files are generated from comments imbedded within source files.
- Javadoc generation ignores comments unless they start with /\*\*

```
//Hi, I am ignored by Javadoc generation
/* Me too.
 *
 */
/** BUT NOT ME. I get compiled into HTML!
 *     Ain't I special?
 */
```

-4-

## Javadoc uses HTML tags

<code>&lt;br&gt;</code>	Break, new line
<code>&lt;p&gt;</code>	New paragraph
<code>&lt;i&gt;text&lt;/i&gt;</code>	Italic
<code>&lt;b&gt;text&lt;/b&gt;</code>	Bold
<code>&lt;tt&gt;text&lt;/tt&gt;</code>	Monospace font, often used for class, field or method names.
<code>&lt;ul&gt;</code>	
<code>&lt;li&gt;list item 1&lt;/li&gt;</code>	
<code>&lt;li&gt;list item 2&lt;/li&gt;</code>	
<code>&lt;/ul&gt;</code>	
<code>&lt;ol&gt;&lt;li&gt;item 1&lt;/li&gt;&lt;li&gt;item 2&lt;/li&gt;&lt;/ol&gt;</code>	

- list item 1
- list item 2

-5-

## JAutodoc Eclipse Plugin

- <http://jautodoc.sourceforge.net/update/>
- JAutodoc is an Eclipse Plugin for automatically adding Javadoc placeholders to source code.

-6-

## Lab 11: Javadoc

---

- Add Javadoc to the code turned in for the third milestone of the Breakout project.
- If you have not turned in the third milestone, then use whatever the latest version you have of the Breakout project.
- For full points on this project, the code you document does not need to work correctly. It does, however, need to compile. Document the parts that work, and if some parts do not work, then document what is wrong.
- Each class, all methods (both public and private) and all class fields must be documented in the Javadoc form.

-7-

## Lab 11: Javadoc: Grading Rubric

---

Due Friday, November 6 at midnight.

**6 points:** each class has Javadoc describing what the class does and how it is used.

**7 points:** each method (public and private) has Javadoc describing what the method does, what inputs (if any) are used and what they mean, what (if anything) the method returns, and an overview of the algorithm used in the method.

**7 points:** each class field (public and private) has Javadoc describing what the field is used for, what units (if any) are the field's values, and what is the range of values of the field.

-8-