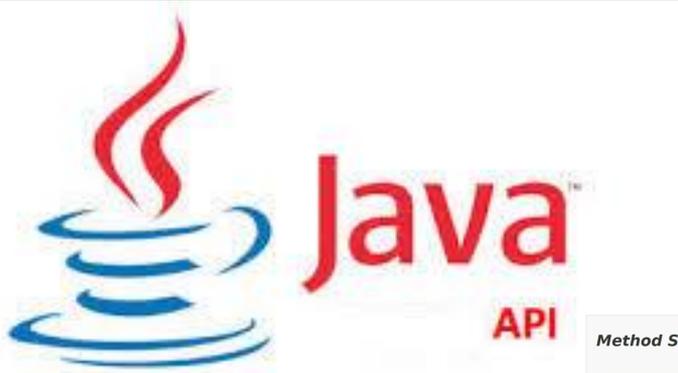
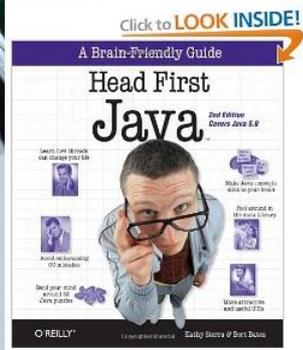


Course Overview, Java Library, API, javadoc



Method Summary

All Methods	Static Methods	Concrete Methods
Modifier and Type	Method and Description	
static Object	get(Object array, int index)	Returns the value of the indexed component in the specified array object.
static boolean	getBoolean(Object array, int index)	Returns the value of the indexed component in the specified array object, as a boolean.
static byte	getBytes(Object array, int index)	Returns the value of the indexed component in the specified array object, as a byte.
static char	getChar(Object array, int index)	Returns the value of the indexed component in the specified array object, as a char.
static double	getDouble(Object array, int index)	Returns the value of the indexed component in the specified array object, as a double.

Outline

- **Course Overview**
 - Website
 - Syllabus
 - Assignments
 - Exams
- **Java Library**
- **API**
- **javadoc**

Course Website and Other Important Info

Michele Van Dyne

Museum 204B

Office Hours: M 2:00-3:00, WF 1:00-2:00 or by
appt.

mvandyne@mtech.edu

<http://katie.mtech.edu/classes/csci136>

The Java Library

- The Java Library
 - Many handy classes and methods
 - Importing a package
 - Only the most important are automatically available without excessive typing:
 - ✦ Things like `String`, `System.out`, etc.
 - URL:
<https://docs.oracle.com/javase/9/docs/api/>

Java Packages

- A collection of classes under one namespace
 - Avoids problems if multiple classes have same name
- Common stuff in java.lang package

```
// Two ways to declare a String  
String s = "hello world!";  
java.lang.String s2 = "hello world!";
```

- The String class lives in a package called java.lang
- Qualifying it is optional for this package
- We will be using other java packages this semester
 - For today I want you to pay attention to the API format

API: Example – Some Methods for Array

Method Summary

All Methods	Static Methods	Concrete Methods
Modifier and Type	Method and Description	
static Object	get(Object array, int index) Returns the value of the indexed component in the specified array object.	
static boolean	getBoolean(Object array, int index) Returns the value of the indexed component in the specified array object, as a boolean.	
static byte	getBytes(Object array, int index) Returns the value of the indexed component in the specified array object, as a byte.	
static char	getChar(Object array, int index) Returns the value of the indexed component in the specified array object, as a char.	
static double	getDouble(Object array, int index) Returns the value of the indexed component in the specified array object, as a double.	

javadoc

- General
- Tags
- Generating Documentation
- Running javadoc



javadoc: General

- **Software Documentation**
 - Ubiquitous problems
 - Writing API documentation for a system is one of the most unpleasant jobs a developer will ever face
 - ✦ The kind of job that could drive you to despair
 - No documentation = no code
 - “Informal” documentation isn’t standard
 - As software evolves, “informal” and/or external documentation and code become out of sync
 - Eventually (often very quickly), documentation becomes unusable making code hard to understand and update!

javadoc

- javadoc utility makes writing and maintaining code documentation easier!
 - Ships with JDK
 - Defines a set of specially formatted comments
 - Can be added to document each
 - ✦ package,
 - ✦ class (& interfaces),
 - ✦ method, and
 - ✦ field (variable)
- Used to generate HTML documentation of classes or packages after parsing the comments
- HTML documentation of the API

javadoc Comment Format

- IMMEDIATELY precedes the item it describes
- Consists of:
 - Description of the feature
 - ✦ Copied as is to the documentation page
 - List of tags
 - ✦ Formatted by javadoc in a consistent style

javadoc Comment Format

- Has 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.
```

```
* <SPACE>
```

```
* zero or more tags to specify more specific kinds of  
* information, such as parameters and return values for a  
* method
```

```
*/
```

javadoc Guidelines

- For this class, must be provided for:
 - **Every public class or interface**
 - **Each public method (including constructors)**

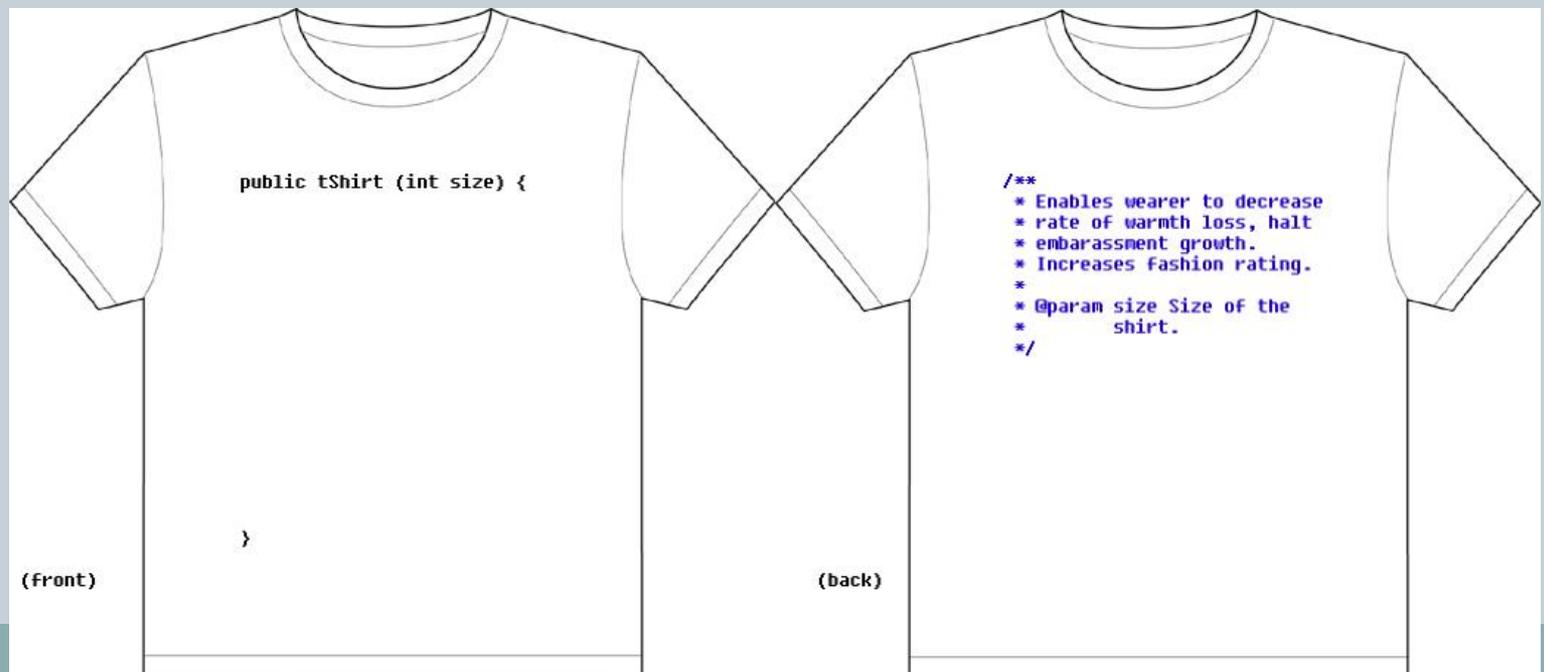
javadoc Tags: @author

- **@author name**
 - Name one of the authors of this class
 - Use multiple tags if there are multiple authors
 - Used in: Class, Interface, Method
 - E.g.

@author Jane Smith, lab X

javadoc Tags: @param

- @param <name of parameter> short description of parameter
 - Describe the named parameter to this method
 - Skip this tag if the method has no parameters
 - Used in Method
 - E.g. @param size number of elements in the array



javadoc Tags: @return

- **@return text**
 - Describe the value returned by this method
 - Skip this tag if the method has no return value
 - Appears after @param tag(s)
 - Used in: Method
 - E.g.

@return String a string representing the object's description

javadoc Example: Method

```
/**
 * Validates a chess move.
 *
 * @author John Doofus
 * @param theFromLoc Location of piece being moved
 * @param theRank Rank of piece being moved
 * @param theToLoc Location of destination square
 * @return true if a valid chess move or false if invalid
 */
boolean isValidMove(int theFromLoc, int theRank, int theToLoc)
{
    ...
}
```

javadoc Reference

- More than you ever wanted to know about javadoc:

<http://docs.oracle.com/javase/1.5.0/docs/tooldocs/windows/javadoc.html>



Required javadoc for this Course

- Class or Interface:

- ◦ @author



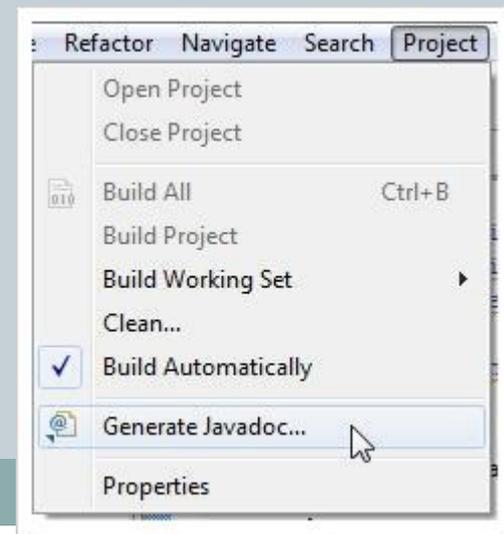
Required javadoc for this Course

- Method/Constructor
 - Description of method
 -
 - @param
 - @return



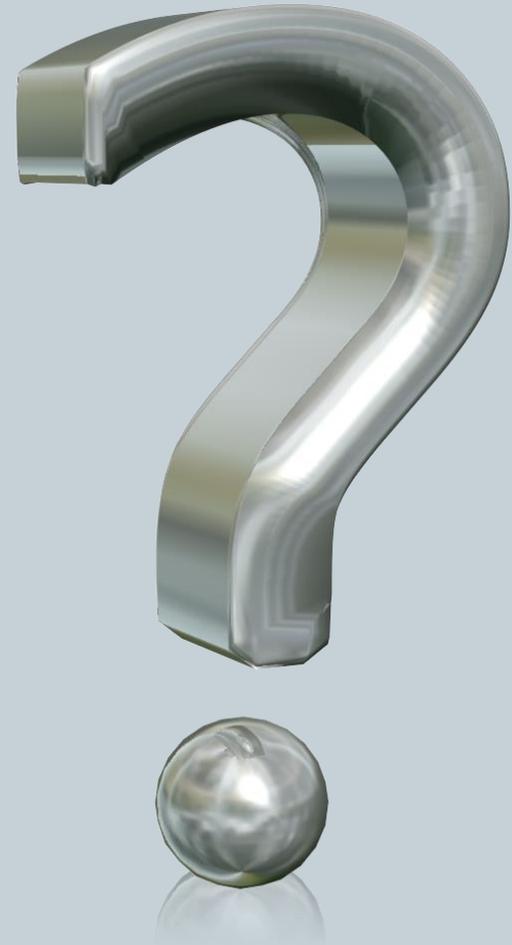
Running javadoc in Eclipse

- Go to Projects menu at top
- Choose Generate Javadoc from that menu
 - You can accept all the defaults and javadoc will run as you've seen
 - You may need to “configure” it
 - ✦ Click in the configure box, click Browse
 - ✦ On my machine I navigate to:
 - C:\Program Files\Java\jdk1.5.0_22\bin\javadoc.exe
 - This is where the javadoc.exe program “lives”
 - If you have not generated documentation for a project yet, it will ask if you want to create a “doc” directory
 - ✦ Yes, you do
 - Can go to doc directory (now listed in the projects)
 - ✦ Open index.html to see main javadoc document



Summary

- **Course Overview**
 - Website
 - Syllabus
 - Assignments
 - Exams
- **Java Library**
- **API**
- **javadoc**



Your Turn

- Download the TryJavadoc.java program from the website and add appropriate javadoc comments
- Open Moodle, go to CSCI 136, Section 01, open the dropbox for today
- Drag and drop your TryJavadoc.java program file to the Moodle dropbox
- You get: 1 extra credit point if you turn in something, 2 extra credit points if you turn in something that has correct javadoc documentation.
- You only need to turn in the code, not the documentation.