

Java Programming

Arthur Hoskey, Ph.D.
Farmingdale State College
Computer Systems Department

- Javadoc

Today's Lecture

- Suppose we buy a new device and it comes with a user's manual.
- The user's manual will likely describe the following:
 - The general purpose of the device.
 - What each control on the device does.
 - Other details about how to use the device.
- This user's manual will not go into detail about the inner workings of the device.
- It will not be describing the details of what wires are connected on the inside or how the inner workings are pieced together.
- There may be another manual specifically designed for engineers that might describe the inner workings.

Javadoc

- A user manual describes how to operate a device.
- For example, a TV user's manual describes how to operate a TV.
- It does NOT describe the inner workings of a TV.
- For example, it will not give details about the inner workings of a light-emitting diode (LED) screen.
- A user manual will describe:
 - How to turn on the TV.
 - The purpose of each button on the TV.
 - The purpose of each connector port.
 - Etc...

TV User's Manual

Page 1 – Turning on/off
Page 2 – Screen settings
Page 3 – Side panel connectors
Etc..

User Manual

- A Javadoc comment describes methods, classes, and package from the outside (similar to a user manual).
- For example, there may be a Javadoc comment for a method of a class. This Javadoc comment will describe what the method does, its parameters, return type, and some other data about the method.
- It does NOT describe the inner workings of the code in that method. It does not describe loops or variable declarations. This details are irrelevant for user's of the method.

Employee Class Javadoc

Describe what the CalculateSalary method does and how to use it.
Etc...

Javadoc Comments

- You should use normal comments to describe code that is inside of a method (inner workings of a method).
- The comment below would be placed inside a method (it is NOT a Javadoc comment).

```
// This loop adds up the total salary of all the highest-paid
// employees (salaries less than a 60000 are ignored).
int totalSalary=0;
for (int i=0; i<100; i++)
    if (emp[i].getSalary() >= 60000) {
        totalSalary = totalSalary + emp[i].getSalary();
    }
}
```

Normal Java Comments

- Javadoc syntax:

`/**` ← **Start Javadoc comment**

Here is a Javadoc comment.

`*/` ← **End Javadoc comment**

Javadoc

Javadoc Comment Usage

- Every class should have a Javadoc comment above it.
- Every method should have a Javadoc comment above it.


Javadoc Comment Usage

Java Annotations

- Java annotations are metadata about the program.
- Annotations are used to:
 - Give information to the compiler.
 - Information for software tools that are used to generate code or output files related to the program (for example, information to generate HTML Javadoc documents).
 - Runtime processing information.
- Java annotation start with an @.

@Override

**The Override annotation indicates that
a method is overriding another method**



```
public toString() {  
    // toString code goes here...  
}
```


Java Annotations

Javadoc Class Comments and Annotations

- Javadoc has many different annotations you can use in your comments.
- For example:

```
/**  
 * This class contains methods to perform mathematical  
 * calculations  
 *  
 * @author Rose Diaz  
 */  
public class Calculator {  
    // Calculator class code goes here  
}
```

Specify the author of the class
using the @author annotation



Javadoc Class Comments and Annotations

Javadoc Method Comments with Annotations

- Parameters and return value get annotations.
- If the method throws an exception that should have an annotation.

```
/**  
 * Finds the quotient of the given numbers and returns the result.  
 *  
 * @param num Numerator for the division.  
 * @param den Denominator for the division.  
 * @return The quotient of the passed in values.  
 * @throws ArithmeticException Throws exception if the denominator is 0.  
 */  
public double divide(double num, double den) throws ArithmeticException {  
    if (den == 0)  
        throw new ArithmeticException();  
  
    return num/den;  
}
```

Description of what method does ←

Parameter and return value annotations ←

Exception annotation ←

Javadoc Method Comments with Annotations

Generating Javadoc HTML Documents

- To actually generate the HTML files containing your Javadoc comments you need to run the Javadoc tool.
- To generate the Javadoc go to the menu item **Run | Generate Javadoc**.

Generate Javadoc HTML Documents

View Javadoc HTML Documents

- Go to Window|IDE Tools|Javadoc Documentation.
- This will open a tab named Javadoc in the bottom window inside NetBeans.
- If you put the cursor on a class or a method in your code (by left clicking, not hovering) it will show the Javadoc document for that item in the Javadoc tab.
- You can also open File Explorer and navigate to the target/site/apidocs/<your package> directory within your project. The Javadoc .html files that were created when you ran the Javadoc tool will be there.
- You can open the Files tab in NetBeans and get the full name of an HTML file (right-click file and choose Properties). Navigate to the target/site/apidocs/<your package> directory within your project. Paste the full filename of an HTML file into a browser.

Generate Javadoc HTML Documents

- End of Slides

End of Slides