David Hazall-Farrell

CSL-131

1/22/2024

Lab #1

Step 15a, 15b:

* 1. **How does Java terminate statements or instructions that are to be executed by the computer? (That is, what character is found at the end of each line with a Java statement?)**

**Java terminates statements and instruction whenever there is a semi-colin ( ; ).**

* 1. **How does Java identify comments?**

**Java identifies comments when there is a backslash followed by an asterisk (/\*). You can end comments by doing the same, but reversing the order (\*/). Each comment in between these symbols must have an asterisk at the beginning of the statement.**

**Step 17:**

1. **Write a few sentences that describe how the editor indicates that there is an error.**

**The editor tells me that there is an syntax error in the thread “main.” It says that I need to insert a semi-colin to complete the statements. At the end it tells me what specific line is having the error.**

Exception in thread "main" java.lang.Error: Unresolved compilation problem:

Syntax error, insert ";" to complete BlockStatements

at HelloWorld.main(HelloWorld.java:16)

**Step 19:**

1. **The system pops up with red text telling me that I have a missing semi-colin. It points out to me that this error is on line 16 by putting a red X next to the line and by outputting the message, “**at HelloWorld.main(HelloWorld.java:16).”

**Once I re-ran the program, it output the correct information.**

**Step 20:**

1. **The editor tells me that I have a compilation problem, and that I need to close my dialogue with double quotes. It also points out that the problem is on line 16.**

Exception in thread "main" java.lang.Error: Unresolved compilation problem:

String literal is not properly closed by a double-quote

at HelloWorld.main(HelloWorld.java:16)

Step 21:

1. Overall, I found the editor to be helpful throughout my work. Though it was barely necessary now, it’s going to come in handy when we’re dealing much lengthier and more complicated code.

/\*

\*(Modified output from "Hello World", "I am in CS 131!") to

\*

\*The message "Name: David Hazall-Farrell"

\* "Email: hazalldr22@bonaventure.edu"

\* "Lab Date: 1/22/2024"

\*

\* @author David Hazall-Farrell

\* @version January 22, 2024

\*/

**public** **class** HelloWorld

{

**public** **static** **void** main(String [] args)

{

System.***out***.println("Name: David Hazall-Farrell);

System.***out***.println("Email: hazalldr22@bonaventure.edu");

System.***out***.println("Lab Date: 1/22/2024");

}

}

**import** javax.swing.\*;

/\*\*

\* Displays the string "Hello, World!" by using a JOptionPane MessageDialog.

\* Starting with a simple example, if you just want to show a JOptionPane dialog with a simple text message, all you need is one line of Java source code like:

JOptionPane.showMessageDialog(frame, "A basic JOptionPane message dialog");

I removed "Hello, World!" and replaced it with my name, the date and email.

I then put all of the dialogue on different lines with the use of the escape sequence, '\n'

\*

\* **@author** David Hazall-Farrell

\* **@version** January 22nd, 2024

\*/

**public** **class** DisplayMessage

{

**public** **static** **void** main(String [] args)

{

// declare and create a JFrame object

JFrame myWindow;

myWindow = **new** JFrame();

// set size of the window and make it visible

myWindow.setSize(400,300);

myWindow.setLocation(500, 400);

myWindow.setTitle("My Second Java Program");

myWindow.setVisible(**true**);

// use a message dialog box to display a message

JOptionPane.*showMessageDialog*(myWindow, " Name: David Hazall-Farrell \n Email: hazalldr22@bonaventure.edu \n Lab Date: 1/22/2024");

}

}