|  |  |
| --- | --- |
| **AP Computer Science GridWorld Case Study (GWCS)** | |
| **GridWorldLab08** | **Javadocs and the GridWorld Case Study** |

|  |
| --- |
| **Lab Objectives** |
| Study the class interaction of the Grid World program with Javadocs.  Observe the available classes of the GWCS.  Investigate the attributes, constructors and methods of each GWCS.  Learn how to make the Javadoc comments help to create the Javadoc web pages.  Make a Javadoc tool in JCreator.  Create your own set of web page documentation with Javadoc. | |

|  |
| --- |
| **Lab Prerequisites** |
| **Completed ExpoJava, Chapter 13 and completed GridWorldLab07**  Understand classes, attributes, constructors and methods. | |

|  |  |
| --- | --- |
| **Lab Sequence of Steps** | |
| **#** | **Actions** | | **Comments** |
| **01a** | **Check the GridWorldLab08 folders**  Do NOT load **JCreator** yet!  You will see the folders displayed by **Figure 01**.  **Figure 01** | | This assignment will be done in 4 parts.  The first part will not use **JCreator** at all. |

|  |  |  |
| --- | --- | --- |
| **01b** | **Check the GridWorldLab08-Part1 files**  Double-click the **GridWorldLab08-Part1** folder.  Then double-click **index.html** file.    **Figure 02** | This folder shows a set of different files. There is not a single file with a **.java** ending anywhere in sight.  This folder contains a special set of files that manages a web-paced reference of GridWorld Case Study information.  With these files you will be able to display practical information about class, attributes, constructors and methods.  The files are also organized in a manner that makes it easy to find requested class and/or method information.  The **.html** endings are normal for web pages. HTML means Hyper Text Markup Language. It does not matter if that name makes sense. Think of HTML as another programming language, which is designed to display output about web pages.  The **index.html** file is a special file that is commonly used to start viewing a web site. |
| **01c** | **Activate the GWCS Javadoc Reference, Approach 1**  **Figure 03** shows the result of clicking on **Index.html**.  **Figure 03** | The left-top window shows all the available packages. A package contains multiple classes. Access to the classes inside a package is done with the **import** statement.  The bottom-left window displays all the available classes of the entire program or the selected package.  The major window on the right side shows the details of the information you wish to know.  Keep in mind that this is not organized to teach computer science concepts. The Javadocs, both for the Java language in general, and specifically for the GridWorld Case Study is a reference.  The comments that have been placed inside the GridWorld file will appear here as well. This documentation is an excellent tool to give you an overview of the available choices in this program.  Note that ***Grid*** is *italicized* unlike everything else under **All Classes**. This is because ***Grid***is an **interface**, not a **class**. |

|  |  |  |
| --- | --- | --- |
| **01d** | **Activate the GWCS Javadoc Reference, Approach 1**  Click on the **Actor** class next to the *horizontal arrow* in **Figure 04**.  This brings up the constructor and method summary of **Figure 04**.  Now click on **getDirection** underneath the *vertical arrow*.  **Figure 04** | **Figure 04** shows how Javadocs are traditionally used. Clicking on the **Actor** class brings up the information about *attributes*, *constructors* and *methods* in the right window.  You can quickly see the name of the available methods. They will be listed in *alphabetical* order. You can also see whether they are **void** or **return** methods as well as what parameters are required.  Additional information can be displayed about the individual members of a class by clicking on the member identifier. |
| **01e** | **Activate the GWCS Javadoc Reference, Approach 1**  **Figure 05** shows additional details about the **getDirection** method.  **Figure 05** | Clicking on the method identifier brings up the detailed information. You get to the same location by scrolling down. After the method summary is finished the web page continues with the **Method Details** section. |

|  |  |  |
| --- | --- | --- |
| **01f** | **Activate the GWCS Javadoc Reference, Approach 2**  Right now close the web browser window.  When you return to the **GridWorldLab08-Part1** folder,  double-click the **index-all.html** file shown in **Figure 06**.  *Do not confuse index-all.html with index.html!*  **Figure 06**    You will see **Figure 07**. This is a second reference approach  **Figure 07** | Approach 1 to check the GridWorld reference is the most common approach.  If you want to know information about a particular class, or the methods of that class, go to **index.html**, click on the class, and look up the information.  What if you want information on a method, but you do not know what class it is in?  What is you need to do something in GridWorld and you do not even know if a method that does that is exists, much less what class it is in?  In this case, you may want to go to  **index-all.html**.  This brings up an alphabetical selection of all methods and all attributes from all the GridWorld classes.  There is a brief description for the method/attribute and also a direct link to that class it is from.  When you look at this list, you can right away see that there are several **act** methods from different classes. |

|  |  |  |
| --- | --- | --- |
| **01g** | **Activate the GWCS Javadoc Reference, Approach 2**  Let us say you want information on the **move** method, but do not know what class it is in.  Scroll down until you find the **move** method as shown in **Figure 08**.  From this you should see a brief description of the **move** method.  You should also be able to see that **move** is a method from the **Bug** class.  **Figure 08**    Click the **move** method to see more detailed information as shown in **Figure 09**.  **Figure 09** | Some methods will have additional information in the **Details** section.  Some will not.  The difference between approach 1 and approach 2 is only evident in the beginning.  After you get to the **Summary** or **Details** level, the display is the same. The only difference exists at the initial level when you first start to search for information. |

|  |  |  |
| --- | --- | --- |
| **02a** | **You Can Create Your Own Javadoc Web Page Documentation**  Close your browser.  Navigate to the folder shown in **Figure 10** located at:  **C:\Program Files\Java\jdk1.7.0\_03\bin**  This is the folder of executable Java programs.  We are interested in the **Javadoc.exe** program file.  **Figure 10** | NOTE:  At the time this document was created  I was using *Java Version 7 update 3*.  It is possible you may have a different version of Java on your computer. If you do the name of the *jdk* fold may be a little different, but should still be very similar.  It is possible to create the Javadoc web pages with the command prompt. This requires dealing with DOS commands, path manipulation and other complexities.  In a world of windows and mouse clicking, the command prompt option is not very attractive.  Furthermore, many school district disable access to the command prompt.  It is possible to handle Javadoc web pages with JCreator. This requires that a special **Javadoc** tool is created. |
| There are 4 significant files in the window displayed in **Figure 10** above:   |  |  | | --- | --- | | **Filename** | **Description** | | **appletviewer.exe** | This file is called by **JCreator** anytime you execute an *applet*.  You will see “Applet Viewer” on the execution window. | | **java.exe** | This file is called by **JCreator** anytime you exeute an *application*. | | **javac.exe** | This is the *Java Compiler*.  This file is called by **JCreator** anytime you compile any Java file.  It does not matter if the file is an *application* or an *applet*. | | **javadoc.exe** | This file creates **.html** documentation files from **.java** files. |     Do you remember when you first learned to install **JCreator**? We made a big deal about the fact that you MUST install the **JDK** BEFORE you install **JCreator**. The reason for this is JCreator is little more than a text editor.  It does not have a compiler. It does not have the ability to execute applications or applets. When you click the build/compile icon JCreator accesses the **javac.exe** file in the **bin** directory. This is only possible if **JCreator** knows where the **bin** directory is. This is the whole reason why the **JDK** MUST be installed BEFORE **JCreator**. | |

|  |  |  |
| --- | --- | --- |
| **02b** | **Creating a Tool for Javadoc in JCreator**  Load **JCreator**. Click **Configure** and **Options**.  Select **Tools** in the left window as shown in **Figure 11**.  **Figure 11**    Your screen should now resemble **Figure 12**. Click the **New** button.  **Figure 12**    Your screen should resemble **Figure 13**. Select **Program**.  **Figure 13** | JCreator gives the option to create special shortcuts, called tools. The tools take on a name and once the tool is configured, it can be selected for future use.  We are interested in running a program, as a consequence of selecting this special tool.  Specifically, we want to run the **Javadoc.exe** program so that we can create our own documentation. |
| **02c** | **Creating a Tool for Javadoc in JCreator**  Navigate to the  **C:\Program Files\Java\jdk1.7.0\_03\bin**  folder as shown in **Figure 14**.  Double-click **Javadoc.exe**.  **Figure 14** | You might have a slightly different folder name if you have a different version of the JDK.  You are in the business part of the Java programs. It is in this folder that programs are compiled and executed.  Files that end with **.exe** are executable files. Note the clever ending. The cleverly named file **Javadoc.exe** will be used for the web page creation. In this step JCreator learns where the program is located. |
| **02d** | **Creating a Tool for Javadoc in JCreator**  You should now see **Javadoc** now shows up in 2 different locations underneath 2 different **Tools** menus as shown in **Figure 15**.  You need to click the **Javadoc** on the left side of the screen indicated by the arrow below.  **Figure 15** |  |

|  |  |  |
| --- | --- | --- |
| **02e** | **Creating a Tool for Javadoc in JCreator**  Your screen should now look like **Figure 16**.  **Figure 16**    Make the following changes in the window so it matched **Figure 17**.  Check **Capture output** and **Show command line**.  Enter **$[JavaFiles]** for **Arguments**.  Enter **$[FileDir]** for **Initial directory**.  **Figure 17**    When your window matches **Figure 17** exactly, click **OK**. | It is not necessary to rember these steps. This process is not the same as creating a project, which will happen quite frequently. Once all the steps are finished to create the **Javadoc** tool, you will have access to use the new tool anytime.  Suggestion: Highlight the code to the left, copy it, and paste it in the *Arguments* and *Initial Directory* fields. This will guarentee that the information is entered correctly.  You **MUST** be exact with this!  I strongly suggest you highlight the **$[JavaFiles]** and **$[FileDir]** in this document and paste them into the **Arguments** and **Initial directory** entries in JCreator. |

|  |  |  |
| --- | --- | --- |
| **02f** | **Creating a Tool for Javadoc in JCreator**  Click on **Tools** and select **javadoc** as shown in **Figure 18.**  **Figure 18**    Right after you select **javadoc** you should see something in your  **General Output** window that should be similar to **Figure 19.**  **Figure 19**    At the bottom it shows an error. There is nothing wrong with the tool.  **Javadoc** does not have anything to work with. We need a project. | You now have a newly created **javadoc** tool.  You are almost ready to get serious. The tool is created once and can now be used many times. This is similar to the jar library. You created the library once and then it can be used many times.  You can ignore all of the *Do not this* and *Do not that* statements. Only the last line is significant here. |
| **02g** | **Create a GridWorldLab08-Part2 Project**  **Figure 20** |  |

|  |  |  |
| --- | --- | --- |
| **02h** | **Create a GridWorldLab08-Part2 Project**  After you click **OK** in **Figure 20** your screen should match **Figure 21**.  There is no need to attach any library file. Just click **Finish** twice.  **Figure 21** | After many GridWorld labs, you are now used to attaching the **GridWorldJarFile**.  While Lab **GridWorldLab08-Part2** requires creating a project, it is not actually a *GridWorld*project.  There will be no **Bug**s moving or turning. There will be no **Actor**s doing *flip-flops*.  This means no **GridWorldJarFile** is needed.  You can go ahead and click **Finish** twiceon this screen. |
| **02i** | **Create the Javadoc Web Pages**  Your screen should match **Figure 22**.  **Figure 22** | We can see that this project involves 3 files:   * **Aardvark.java**, * **Mango.java** and * **Widget.java**   These are the only files in the **GridWorldLab08-Part2** folder when you begin the lab.  Each of these file has one of each of the following:   * **data attribute** * **constructor** * **return method** * **void method**   Remember, these files have nothing to do with GridWorld. |
| **02j** | **Create the Javadoc Web Pages**  Click **Tools** and select the **javadoc** tool as you did before.  Now it should work.  Your **General Output** window should be similar to **Figure 23** and not have any errors.  **Figure 23** | This project involves three classes used to demonmstrate **Javadoc** web pages.  This project cannot *execute* properly, but that does not matter. You do need to place the files in the project.  The **General Output** window demonstrates how the **Javadoc.exe** program translates the 3 **.java** files into **.html** web page files.  We can also see that several other **.html** files are created. |
| **02k** | **View the Newly Created Javadoc Web Pages**  Minimize **JCreator** and navigate to the **GridWorldLab08-Part2** folder.  Double-click **index.html**.  **Figure 24** | Note how all the **.html files** ended up in the **GridWorldLab08-Part2** folder where you created the project. You can now double-click on **index.html** to activate the web pages.  This folder use to only have 3 **.java** files in it. As you can see in **Figure 24**, **Javadoc** created several other files. |

|  |  |  |
| --- | --- | --- |
| **02L** | **View the Newly Created Javadoc Web Pages**  You are now at **Figure 25**.  **Figure 25**    Compare this to the Aardvark class in **JCreator** (**Figure 26**).  **Figure 26** | If you look at **Figures 25** and **26** you can see that every method and attribute in the **Aardvark.java** file is shown in the **Aardvark.html** documentation file.  What we do not have in the documentation file are any descriptions of the attributes, methods, or class.  You may remember that in **Part1** of this lab assignment we saw descriptions of the **GridWorld** methods.  The descriptions are supposed to come from the comments in the **.java** file but for some reason they are not showing up.  We need to do something different.  You know about the *single line* comment symbol (//) and *multi-line* comment symbols (/\* and \*/) in Java.  These do not work with **Javadoc**.  To make this work, we need something new. We need a third kind of comment… |

|  |  |  |
| --- | --- | --- |
| **03a** | **Create a GraidWorldLab08-Part3 Project**  Close your web browser and go back to **JCreator**.  Click on **File** and select **Close Workspace** as shown in **Figure 27**.  You do not need to *Save the workspace modifications*.  You do *want to close all document windows*.  **Figure 27**    Create a new **GridWorldLab08-Part3** project:  **Figure 28** |  |

|  |  |  |
| --- | --- | --- |
| **03b** | **Create the Javadoc Web Pages**  Just like you did in **Part2**, click **Tools** and then select **javadoc** and observe the **General Output** window.  You should see the creation of the **.html** files as you did before.  **Figure 29** | As before, the **General Output** window demonstrates how the **Javadoc.exe** program translates the java files into **.html** web page files. |
| **03c** | **View the Newly Created Javadoc Web Pages**  Minimize **JCreator** and navigate to this folder shown in **Figure 30**.:  Double-click **index.html**.  **Figure 30** | Note how all the **.html files** ended up in the **GridWorldLab08-Part3** folder where you created the project. You can now double-click on **index.html** to activate the web pages. |

|  |  |  |
| --- | --- | --- |
| **03d** | **View the Newly Created Javadoc Web Pages**  You are now at **Figure 31**.  **Figure 31**    Compare this to the Aardvark class in JCreator (**Figure 32**).  **Figure 32** | Now look at **Figures 31** and **32**.  The attributes, methods, and class all have descriptions. If you look at the **Aardvark.java** file you should notice these descriptions come from the comments which are before every attrbite, method, and the class itself.  Why did this not work in **Part2**?  The difference is **Part3** is using special *Javadoc Comments*.  These look very similar to the *multi-line comments* except instead of a comment starting with *slash-star* and ending with *star-slash*, it is now *slash-double star* and  *double star-slash*.  **Javadoc** looks for these special comments and makes them part of the **.html** files.  **NOTE: private or *helper* methods will not show up in the html files created by Javadoc**. |

|  |  |
| --- | --- |
| **04** | **Lab Assignment**  For **80 points** create some original class. The class must have at least two attributes, two constructors  and three additional methods.  Each of the methods must have *Javadoc* style comments. Create *Javadoc* web pages of your class.  The methods in your class must have a complete heading, but the method bodies do not require any functional program statements.  For **100 points** create three original classes with *Javadoc* comments.  It is not important that the three classes interact in some way.  For **110 points** create three original classes with *Javadoc* comments.  The three classes need to implement *multi-level* inheritance.  (Meaning the first class *extends* the second class, and the second class *extends* the third.)  You need to place your files in the **GridWorldLab08-Part4** folder. The **GridWorldLab08-Part4** will be the location folder you specify when you set up the project. This will cause the project to be called **GridWorldLab08-Part4** as well.  When you activate the *Javadoc* tool, the html files will also be created in the **GridWorldLab08-Part4** folder.  When you are done, browse to the **GridWorldLab08-Part4** folder, execute **index.html** and show your teacher. |