



A. Review the build

The downloaded zip file verifies using "ant -f verify.build.xml" and runs using "ant run". However there is no jar file and the far file in the zip file that was downloaded does not work. It shows HTTP ERROR: 500. and org/wattdpot/client/NotAuthroizedException.

B. Review system usage

What happened to the logo? The logo is not there when it runs. There is a big space at the top.

The system does what it is suppose to do. Since there is only 4 usable things on the page, there is no way to mess up.

The system catches every error, especially if you put a future date.

There is no explanation of the program. Just enter a number and submit. The output fits to the screen if you scroll down a little. Because the logo does not show, it looks as if there is just a big space.

C. Review the JavaDocs.

Download the system and generate the javadocs (use "ant -f javadoc.build.xml"). Navigate to the build/javadoc folder and click on index.html to display the JavaDocs in a browser. Read through the JavaDocs and assess the following:

The overview does not provide a high-level description of the purpose of the system. Also it is named wrong. It still says edu.hawaii.wicket. It should have the project name edu.hawaii.greensmart. This could

be done by refactoring the the package and change parts of the build.xml file and jetty.java for the program to work.

Since there is only one package being used, there is no package summary(package.html). Also, it is named wrong edu.hawaii.wicket. Refer to previous paragraph on how to change this. And it does not describe greensmart, it describes "An example Wicket application".

Yes, each class gives a high-level description of the class, but it does not provide sample coode for the class.

The method summaries explain very briefly about them. For example in the CarbonContent class, the errors just say "Explains the error". Maybe say what the error is and an example of what the correct output could be.

Following Java style, everything looks ok.

D. Review the names

One of the most important (if not the most important) form of documentation in any software system is the choice of names for program elements, such as packages, classes, methods, instance variables, and parameters. Due to evolution in requirements and design changes, the name originally chosen for a program element may no longer be appropriate or optimal. An important goal of review is to ensure that the names of program elements are well suited to their function. Due to the refactoring capabilities in modern IDEs such as Eclipse, renaming need not be a burden.

All the class names and source code are chosen well.

E. Review the testing.

The system should provide a useful set of test cases.

There is only one test case. So looking at emma, there is not much coverage for the project.

F. Review the package design

The JavaDoc review is focussed on whether the system design is correctly explained. In this section, you start to look at whether the system design is itself correct.

Because there is only one package being used, not much can be said here. I think if more test cases are gonna be made, put those test cases in a test case package.

G. Review the class design

Examine each class implementation with respect to at least the following issues.

Looking at each class, it seems that every class does what it is suppose to do. When called they would return back what is desired.

Each instance variable looks appropriate for the class.

There is not many methods established as private, however this is methods that are protected.

H. Review the method design

Examine each method implementation with respect to at least the following issues.

It looks as if each method accomplishes one thing.

Each method in the program does not exceed a ridiculous amount. Some look long, but that is because it is calling the same method with different parameters.

There are no side effect methods.

I. Check for common look and feel

Every class looks organized in a way that one person made it.

J. Review the documentation

The home page of the project looks very nice, because of the screenshot. It does summarize what the system does, but not what to do from here.

As a developer, it is easy to look at this and know what to do without reading. But for a first time user to a computer, the wiki page does not have a step by step and the user would be lost.

There is a developer guide, but it does not show much on what to do.

K. Review the Software ICU data

For this step, you must ask the Hackystat project owner to add you as a spectator to their project so you can invoke the Software ICU for their system. Run the Software ICU on the project.

From the looks of things, it seems that everything is being gathered consistently.

Looking at the portfolio, it shows that the project is a little unstable. And looking at the telemetry, it shows that the project was worked on a lot on the first day, but after it was not worked on that much, then it was worked on more and more when the due date was comming.

It looks as if the group shared equal amount of work throughout the week.

L. Review Issue management

Go to the project home page, then click on the Issues tab. Next, search for "All Issues" to retrieve a page containing all issues, both open and closed. Next, select "Grid" view, and select "Owner" for the rows. The result should be a page of issues in a grid layout, where each rows shows all of the issues (both open and closed) for a particular project member.

The issue management page is missing, there is no information put into the page.

M. Review continuous integration

Go to the Hudson server, login, and select the continuous integration job associated with the project.

From the looks of it, it shows that the project was committed everyday. However, it was not worked on that much on the 21st, but before and after that day it was worked on a lot.

In the beginning of the project, there was a fail at #2 and did not recover until 2 hours later. The other fails only lasted for about 5-10 minutes.

Yes there is a continuous integration job. But it shows that it is triggered by a commit.

Yes there is a daily build job. But it shows that it is triggered by a commit. It does not run correctly everyday at 11.

Summary

Overall, the project looks good from a developers perspective. But for a new user, this project does not show guidance. The only changes needed for this project is to explain things and to improve the look. The project does fulfill the requirements but not in detail. Also, the project needs changes to the hudson.