Spike: 13
Title: Unit Testing

Author: Michael Williams, 7668481

Goals / deliverables:

- Code
- To improve software through unit testing.
- To write unit tests to verify the correct functionality of the code.
- To learn how to write unit tests.
- To learn how to implement unit testing in visual studio.

Technologies, Tools, and Resources used:

- Visual Studio IDE
- Various web tutorials

Tasks undertaken:

- Research how to implement unit testing in Visual Studio
- Implement chosen method.
- Write unit tests.
- Run unit tests.

What we found out:

We found out that once you know how, implementing unit testing into an existing project in visual studio is very straightforward. There are only things which need to be changed. The first is adding a build dependency between the test project and the existing project. The second is adding an additional location to look for library files. The third is adding an additional location to look for header files. Once all of these have been added, you just need to write the tests.

Through the research we conducted, we found out that unit testing should predominantly cover the details external to the user. By avoiding testing inside classes, this allows code to be changed much more easily without the possibility of breaking tests. Furthermore, to test the inside of classes is not possible without either breaking encapsulation or by writing a large amount of overhead code (mainly getter functions) in order to allow private variables to be used.

Our testing was limited to the functionality that someone using the class can use. We did coverage testing on all public methods where possible for Spikes 8 (Command Pattern), 9 (Composite Pattern) and 10 (Component Pattern). Overall we wrote approximately 10 tests for these spikes.