For this project you will develop a simple editor for HTML files. The project will start in Unit 2 and be continued in Unit 3. Teams will stay the same through both units of work. The emphasis for the project is the application of patterns in appropriate places and justification of those choices. A working program must be created. Since the emphasis of the course is design, spiffy implementations sitting on top of a weak design will not be viewed favorably. This is also not a pattern frenzy, in that, there almost definitely will be areas where it is not appropriate to apply a pattern to the design.

Each release will have a set of requirements defined. The release 2 requirements may necessitate refactoring of your release 1 code base. As new patterns are discussed in class incorporation of them should also be considered.

## Release 1 Requirements

The release 1 functional requirements are:

* The editor shall provide a point-and-click method to specify an HTML file to edit and a method to browse directories. The name of the HTML file can also be specified as a command line argument.
* The editor should be able to have multiple files open for editing. If multiple files are open, the user can select which file is being viewed.
* The editor shall support the insertion of standard HTML constructs. At a minimum, this support should allow the user to select an HTML construct. The editor shall then insert the standard set of matching tags for that construct into the current HTML buffer at the current location for editing. At least the following constructs should be supported: headers, font emphasis (bold, italics), list (numbered, bulleted, dictionary), tables. The editor may provide additional support such as prompting for fields in the construct. It shall allow for specifying the size of a table or list and cause that number of internal item tags to be inserted in the file.
* The editor shall support cut-and-paste editing within a buffer and between buffers.
* The editor shall allow the user to select editor options via menu choices. The editor may optionally provide selection via short-cut or accelerator keys.
* The editor shall provide a method to save changes to the HTML file with the current file name, a modified file name, or newly specified name if it is the first editing session on this file.
* The editor shall provide a check that the buffer is well-formed HTML. The editor shall perform a well-formed check when loading a file and alert the user that the loaded file is not well-formed. The editor may operate with reduced functionality until the user corrects the buffer and a well-formed check passes. The editor shall perform a well-formed check when saving a buffer and alert the user that the buffer being saved is not well-formed. The editor shall allow the user to abort the save or continue.
* The editor shall provide a way to terminate operation. It shall warn the user if any buffers have unsaved changes and allow the user to abort program termination.
* The editor shall support auto-wrap of text at the end of a line. When the cursor reaches the right margin a new line will be opened below. The word currently being typed is moved to this new line and entry continues. The word-wrap feature can be turned on and off.
* The editor shall support auto-indent of HTML constructs. Auto-indent will show the structure of the HTML constructs such as lists and tables. If the cursor is auto-wrapped while information is being entered the cursor will auto-indent the wrapped-text. If the enter key is typed the cursor will auto-indent on the next line. Indentation on a new line will be at the same level as the previous line unless a new structure has been entered such as a list item. Once a line is indented the user can change the indentation spacing by editing the text. This change will be the spacing used for future indentation of the current level. The auto-indentation feature can be turned on and off. The user can specify the number of characters for indentation.
* The editor shall have a command for indenting the current line, the selected text or the entire buffer. This will re-indent the selected item to the current indentation specification.

The project non-functional requirements are:

* The editor shall be written in Java.
* The editor shall be tested for correct functionality on the systems in the SE labs.
* The editor may use code from the standard Java SDK. The use of any other material must have prior instructor approval.
* The working editor shall be delivered as an executable jar file.