HTML EDITOR

RELEASE 2

**Team E**

**ANDRE, SCOTT, BRANDON, ZANE, CHRIS**

Introduction

High Level Design

Total UML Structure

**Introduction**

The assignment that our team designed was a HTML Editor.

**High Level Design**

### C#

We decided to use C# instead of Java because 3 members of the team had significant C# experience and the other two were willing to learn. On top of this, C# offers unique features, such as Language-Integrated Query (LINQ), which provides an intuitive way to operate on collections, and lambda functions. We felt these would speed our development.

### WPF

C# provides two frameworks for GUI applications: Winforms and Windows Presentation Foundation (WPF). We decided on WPF due to its superior support for themes, databinding, styling, and extra features like pop up completion windows.

### Representing files as collections of strings

Early on, it was decided to interpret files as collections of strings with each string being one line. This would allow us to easily implement line-centric features, such as wrapping, in the UI without actually having to maintain a separate version of the text.

### Using IEnumerable<T> where possible

From the start, we have striven to use the enumerator interface in method parameters instead of types that implement that interface. The benefits are twofold:

1) It conforms more closely to LINQ methods, making their adoption easier, and

2) It allows the implementation of “on demand” or generator algorithms. For example, the parser can retrieve lines from the text editor one at a time as it needs them, reducing memory footprint and increasing performance

### AvalonEdit

A week into the project, it became clear that, in order to implement advanced features like syntax highlighting, we would need to spend a lot of time developing our editor control. This felt wrong, since there are several pre-existing solutions for WPF-based text editing controls, such as AvalonEdit, a control by ICSharpCode, used in their popular SharpDevelop IDE.

The decision was made to eliminate all non essential functionality from our editor control, integrate AvalonEdit, and offer the ability to switch between the two editors. Thus, we could meet project requirements while still providing the advanced features we want.

**Current State of Implementation**

The current build of the HTML Editor is operational, and completes all of the functional requirements listed for Release 1. However, there are still some areas that need to be addressed for Release 2. Currently, auto-deindentation does not work with inserting html structures (tables, lists) from the UI. The window prompts for inserting these constructs also does not sanitize user input, meaning they could input values besides integers into the given fields. While there are no currently known issues with HTMLParser, further testing is needed to ensure there are no errors.

**Total UML Structure**

**Sequence Diagrams**

**Implementation**

**Conclusion**

**Pattern Usage Tables**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name:** MVC | | | **GoF pattern:** MVC |
| **Participants** | | | |
| **Class** | **Role in pattern** | **Participant's contribution in the context of the application** | |
| Buffer | ConcreteClass | Holds the HTML text | |
| Window | AbstractClass | Handles interaction from the user | |
| ICodeEditor | Interface | Interprets input and communicates to the Buffer | |
| **Deviations from the standard pattern:**  None | | | |
| **Requirements being covered:**  Entire project | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Name: AvlonEdit** | | | **GoF pattern: Adapter** |
| **Participants** | | | |
| **Class** | **Role in pattern** | **Participant's contribution in the context of the application** | |
| AvlonEdit | ConcreteClass | Adapts AvlonEdit to the interface | |
| ICodeEditor | Interface | Common functionality required by the project | |
| TextEditor | Concrete | AvlonEdit control that is being adapted | |
| **Deviations from the standard pattern:**  None | | | |
| **Requirements being covered:**  None | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Name:** Editor Selection | | | **GoF pattern:** Strategy |
| **Participants** | | | |
| **Class** | **Role in pattern** | **Participant's contribution in the context of the application** | |
| ICodeEditor | Interface | Provides a common way of representing Editors | |
| PlainEditor | ConcreteClass | Basic editor implements project requirements | |
| AvlonEditor | ConcreteClass | Editor that provides advanced editing functionality not required | |
| **Deviations from the standard pattern:**  None | | | |
| **Requirements being covered:**  None | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Name:** UI Interaction | | | **GoF pattern:** Command |
| **Participants** | | | |
| **Class** | **Role in pattern** | **Participant's contribution in the context of the application** | |
| HTMLEditorWindow | ConcreteClass | Contains commands to be executed by any caller | |
| Buffer | Concrete | Contains commands for saving and loading the buffer | |
| ICodeEditor | Interface | Inserting text into Editor | |
| **Deviations from the standard pattern:**  None | | | |
| **Requirements being covered:**  Open an HTML file  Inserting constructs  Cut and Paste  Saving  Undo/Redo | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Name:** Counting Iterator | | | **GoF pattern:** Decorator |
| **Participants** | | | |
| **Class** | **Role in pattern** | **Participant's contribution in the context of the application** | |
| CountingEnumerator | Concrete | Wraps an iterator and provides a way to obtain the number of enumerations | |
| Parser | ConcretClass | Uses pattern to get line number for error | |
|  |  |  | |
| **Deviations from the standard pattern:**  None | | | |
| **Requirements being covered:**  Parsing | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Name:** Font Change | | | **GoF pattern:** Observer |
| **Participants** | | | |
| **Class** | **Role in pattern** | **Participant's contribution in the context of the application** | |
| Control | AbstractClass | It raises the event of font change and communicates with any listeners. | |
| PlainEditor | ConcreteClass | Responds to the font change by adjusting tab size | |
|  |  |  | |
| **Deviations from the standard pattern:**  Event was caught in derived class instead of unrelated class | | | |
| **Requirements being covered:**  None   * Word wrapping * Indenting | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Name:** IEnumerable<T> | | | **GoF pattern:** Iterator |
| **Participants** | | | |
| **Class** | **Role in pattern** | **Participant's contribution in the context of the application** | |
| List | ConcreteClass | Generic collection that could be iterated over | |
| ListIterator | ConcreteClass | Iterator | |
| CodeEditor | ConcreteClass | Consumes the Iterator | |
| **Deviations from the standard pattern:**  Often used in LINQ (Language-Integrated Query) | | | |
| **Requirements being covered:**  Every requirement excluding cut and paste | | | |

**CRC Cards**

|  |  |
| --- | --- |
| **Class:** Avalon Editor |  |
| **Responsibilities:** An implementation of ICodeEditor that helps to take care of special editor tasks, including automatic indentation and syntax highlighting |  |
| **Collaborators** |  |
| **Uses:**  ICodeEditor  HTMLIndentationStrategy  HTMLObject | **Used by:**  Editor |
| **Author:** |  |

|  |  |
| --- | --- |
| **Class:** HTMLParser |  |
| **Responsibilities:** The overall responsibility of this class is to Read through a text file and then parse the file into a HTML tree. The class takes in a string of HTML, loops through and parses out each individual HTML element. It also makes sure the text file is well formed HTML. |  |
| **Collaborators** |  |
| **Uses:**  HTMLElement  HTMLObject | **Used by:**  Editor |
| **Author:** Brandon McAlees, Scott Ketelaar, Chris Jones |  |

|  |  |
| --- | --- |
| **Class:** PlainEditor |  |
| **Responsibilities:**  Loading text  Saving text  Handling keyboard events  Editing text |  |
| **Collaborators** |  |
| **Uses:** | **Used by:**  Buffer |
| **Author:** Scott |  |

|  |  |
| --- | --- |
| **Class:** HtmlEditorWindow (Controller) |  |
| **Responsibilities:** Takes commands from the View and edits the current Buffer |  |
| **Collaborators** |  |
| **Uses:**  Buffer  HtmlParser | **Used by:** |
| **Author:** Whole team |  |

|  |  |
| --- | --- |
| **Class:** Buffer |  |
| **Responsibilities:**  Loading the Editor  Saving the Editor  Gets the type of editor  Checking if editor is dirty |  |
| **Collaborators** |  |
| **Uses:**  PlainEditor | **Used by:**  HtmlEditorWindow |
| **Author:** Chris, Andre, Scott |  |