University of Utah
School of Computing

CS 3500/5010 **Handout 5** **September 30, 2010**

Problem Set Four

In this problem set, you will be creating a Visual Studio solution that solves the problem described below. Your solution must be called `ps4`, and must consist of two projects: a Windows Form project called `SpreadsheetGUI` and a Coded UI Test project called `SpreadsheetGUITests`. Use C# as your implementation language.

We will retrieve your solution for grading by running the Linux command

```
svn --username cs3500 --password ######### checkout
       svn://lenny.eng.utah.edu/home/XXXXXXXX/cs3500/ps4/trunk ps4
```

(put it all on one line) where `#########` is your grading password and `XXXXXXXX` is your CADE login name. We will run the command (which will give us the most recently committed version of your solution) sometime on the morning of October 6. (If you're still working on the morning of October 6, be sure that you don't commit a broken solution.)

You would be wise to verify that the command above works with your repository. If it doesn't work for you, it won't work for us!

Problem

You are to implement a GUI front end for the spreadsheet class that you implemented last week. I am not going to specify the exact appearance and behavior of your GUI, but it is required to exhibit the following characteristics:

- It should have a grid of at least 26 columns and 99 rows. Each cell in the grid should have a content (string, double, or formula) and a value (string, double, or error) as was specified in PS 4.
- There should always be one selected cell, which should be highlighted in some fashion.
- There should be a way to change the selected cell.
- Independent of the grid, there should be non-editable displays showing the address and value of the selected cell, and an editable display showing the contents of the selected cell.

- There should be a way to edit the contents of the selected cell. When the selected cell is successfully edited, the values of other cells in the spreadsheet should be updated as necessary. When the selected cell is unsuccessfully edited, an error message should be displayed.
- There should be a File menu that allows the user to save the spreadsheet to a file, read a spreadsheet from a file, create an empty spreadsheet, and quit the application. Standard file dialogs should be used for saving and reading. If any of these operations would result in the loss of unsaved data, a warning dialog should be displayed.
- There should be a Help menu that displays information on how to use the spreadsheet.

You are also to implement a coded UI test suite that automatically exercises your GUI interface, with the goal of achieving 100% coverage of the control code that links your GUI to the implementation from PS 3.

How to Get Started

First, create your GUI project, which should be a Windows Form application.

Next, you need to add the Spreadsheet.dll from last week's project to this week's project. Right click on your new project in the Solution Explorer. Choose "Add" and then "Existing Item." In the file dialog that appears, set "Objects of type" to "Executable Files." Navigate to last week's spreadsheet project, drill down to the bin\Debug directory, and choose the Spreadsheet.dll file.

The DLL file is now part of your project. However, you need to create a reference to it so the compiler will pay attention. In the Solution Explorer, right click on "References" and choose "Add Reference." Navigate to the DLL file and click "OK".

If you would rather use the Spreadsheet.dll file that I created, you are welcome to do so. It is not ready yet, but when it is I will send out e-mail telling you where to find it.

I have implemented a new Windows Form control called SpreadsheetPanel that you will probably find useful. You need to install it in your project so that it will be available from the Toolbox.

First, you'll need to open the SpreadsheetPanel project from the repository. Open

```
svn://lenny.eng.utah.edu/home/cs3500/examples/SpreadsheetPanel
```

After you have done this, return to your new project. Right click on your new project in the Solution Explorer. Choose "Add" and then "Existing Item." In the file dialog that appears, set "Objects of type" to "Visual C# Files." Navigate to the SpreadsheetPanel project and choose both SpreadsheetPanel.cs and SpreadsheetPanel.Designer.cs. Click "Add" and they will be added to your project.

Rebuild your project and you'll be good to go. You can access the public classes and methods from last week's project (don't forget that they're in namespace SS), and you'll be be able to use my SpreadsheetPanel control.

A Note About Spreadsheet Panel

See the Form1 class in the SpreadsheetPanel project for an example of how to write code that reacts to a SelectionChanged event.

A Note About Testing

When you create your coded UI tests, you won't be able to make assertions about the contents of the spreadsheet grid. Instead, you'll need to make assertions about the remaining controls. This will be sufficient, however, to get good coverage.

Grading

Your grade will be based on the quality and completeness of your GUI, the quality and correctness of your controller code, and the quality, correctness, and completeness of your coded UI tests.