In this programming assignment, you will apply two of the creational design patterns (Factory Method and Singleton) to a problem in which those patterns will facilitate reusability of code in other similar programs.

The images at right illustrate a straightforward interface with a tool strip right below the title bar, containing three split buttons containing different icons, tooltips, and menu items. The first split button is demonstrated in the images. When the mouse is placed over the split button, its split nature is highlighted and its tooltip appears. When the menu is expanded by clicking on the right portion of the split button (the “arrow”), the menu contents are displayed, with the top item associated with pressing the left portion of the split button, and all other menu items listed below the separator. When the primary (i.e., top) menu item is selected, either via the split button’ left side or the selection of the actual menu item, the rich textbox is made visible on the form, with the appropriate contents loaded inside.

While this application is simple enough to implement in Visual Basic by means of the Designer, your assignment is to implement it without the designer. This means that all of the widgets (the form, the tool strip, the split buttons, the menu items, and the rich textbox) will be developed wholly in code, with the notion that the classes that you create will later be available in other similar development efforts.

The Singleton Pattern will be needed to assure that the form will only possess one rich textbox at any moment in time, as well as only one tool strip. Since the tool strip will only be instantiated when the form itself is opened, its Singleton code should be straightforward. Instead of implementing a single rich textbox in the same way, however, the single instance of that widget will only be instantiated when one of the split buttons (or corresponding menu items) activates it. From that point on, however, your code needs to use Singleton code to ensure that selecting additional menu items or pressing additional split buttons will not instantiate additional tool strips.

The Factory Method Pattern will be used to generate the split buttons and their associated menu items. Remember that the first menu item’s click event handler should be activated whenever the left part of the split button is clicked, and that all other menu items should be listed after a separator.

For this particular application, don’t overdo the implementation, since it’s the application of patterns that we’re concerned with here. Default text should be loaded when the user tries to leave feedback, as well as when the user tries to read, print, or edit past comments. Maintaining an actual database for entered feedback is unnecessary, as is any actual printing of feedback (i.e., when printing is requested, a simple informative dialog box or label will suffice).

This assignment is due on your drop-box by 9 AM on Thursday, February 19, 2009.