CSC 454/554 Homework #6

Assigned: October 21, 2006
Due: November 4 2006

This homework is business as usual, plus a little bit extra.

OO Design Problem: A console stack application

Let's revisit a programming 101 classic. Implement a console application in Java or C++ that supports the following stack commands:

- `Push <string>` - add a string to the top of the stack
- `Pop` – remove the top string from the stack
- `Clear` – remove all strings on the stack
- `Undo` – undo effects of the previous command
- `Exit` – shutdown the program

Three steps:

1. Show me your UML and notes on a design for this problem before reading the pertinent patterns chapter.
2. Now retool your design using the **Command pattern** and complete a new, improve UML diagram, complete with your notes. Alas, the Command pattern is not discussed in DPE. I have a handout for you, and you're welcome to surf the net for other sources.
3. I want to **Change** step three:
    - Let's all use Java. If you squeak loud enough, I may relent.
    - Also, let's get this program actually up and running.
    - I know we've been doing mostly snippets to this point, but now I would like your code to sparkle and please use Javadoc conventions when commenting your code.

Both `NetBeans` and `Jgrasp` each have the capability to generate web pages from your `Javadoc`-decorated code. `Javadoc` is easy… look:

- en.wikipedia.org/wiki/Javadoc - Wikipedia's description
- java.sun.com/j2se/javadoc/writingdoccomments - the official Sun description (forget it)
- http://java.sun.com/j2se/1.5.0/docs/api/ - the entire Java 5.0 API uses `Javadoc`… here are the web pages to prove it

You know you've done it right if your documentation is complete (all classes, methods and fields) and it looks like the Sun API web site.

My program is named `Sally`. You can call your program what you like, just not `Sally`, I suppose.

Here's a sample session. User entry is in **bold**. The return value printed after each command is the top string on the stack. Go:

```
**> Sally <**
My Favorite Little Program

Commands are: push, pop, clear, undo, exit
The top of the stack is shown after each command

   <empty>
Sally: push Bill
   Bill
Sally: pop
   <empty>
Sally: push George
   George
Sally: push David
   David
Sally: push Stephen
   Stephen
Sally: undo
   David
Sally: pop
   George
Sally: undo
   David
Sally: clear
   <empty>
Sally: undo
   Sorry, I can't undo the clear command
   <empty>
Sally: exit
   yum
```

How many commands can you undo in a row? One? Some? All of them? In your implementation, you can support only one undo, but describe in your notes how your program would change in order to support undo-ing a fixed number of commands, or an unlimited number.

Homework #6 will be worth 8 points… 1 for your original notes, 2 for your pattern-based UML, and 5 for your working, shiny code.
Good luck guys.
code… yow, bill