Write detailed comparison of Memento and Command Pattern Approaches

Memento

As I learned from the previous assignment, Memento is a behavioural design pattern that focuses on preserving snapshots/memory captures of activities handled by the program. It will be always one step behind the program as it is saving states the program was in instead of the programs current state. In my assignment 3 , I had a Memento class. My Memento class consisted of :

An originator list: this kept track of the current state of my canvas (I.e my undo list)

A caretaker list: this kept memento snapshots of the previous state of my canvas (I.e my redo list)

A history list: this kept track of all the changes made to my canvas, purely for printing purposes and also because most programs I use have a history list or show every change made to the program

An undo method: this would revert my canvas back to its previous state upon being called

A redo method: this would revert my canvas back to its original form before the undo method was called

A clear redo list method: this would clear my redo list everytime a new change was made to the canvas.

If I were to add more shapes to my canvas or if the canvas was bigger and I wanted to undo more complex states such as when a user wants to swap the z-index of two shapes . It would not be possible with my code, as I only have saved the snapshots of created shapes and not their specific location,. I would need to redesign my program for that to be implemented. I believe I did not correctly implement the Memento design pattern , though I think for a project like this, My Memento design is not particularly scalable with complex designs. It makes life easier when you don't have to many functions to undo/redo and you don't care too much for what commands you are undoing so long as it results in the format you are looking for. My design for Memento only takes into account the shapes that are being created and whether they will be undone/redone is up to the user. The size of the canvas nor the amount of shapes added to my canvas does not affect my Memento design.It took me quite awhile to understand how to implement memento into my program so I used arraylists as my memento snapshots were just my svg strings.

Command

In my Command Patter design for my assignment 4, I have gained a better understanding of Command Patterns since I received one-on-one detailed guidance from a kind demonstrator who thoroughly explained what the Command Pattern was and how to implement it into my code. In my assignment 4 code I had a Command class, and Invoker class and individual classes for each of my instructions/commands I wanted to design.

My Invoker class consisted of:

An undo stack list of type command: this would keep track of the commands the user would make, so they can undo them if they so wish

A redo stack list of type command: this would keep track of the commands the user would undo , so they can redo them if they so wish

An Undo method: which will undo the most recent command the user initiated upon being invoked

A Redo method: which will redo the most recent undone command the user initiated upon being invoked

My Command class consisted of a blueprint for my individual designed commands to follow. The base of each of my commands was that they had to execute and their execution had to be undone aswell.

If I were to add more commands I would like to undo or redo , increase the canvas size, add more shapes, it would be simple, tedious but simple. Since the design of My Command Patter allows me to just copy and paste a simple blueprint that will work for every single one of my patterns, all I would need to adjust is what the command is going to need to know and how my command is actually going to function. The tricky part is that I will need to design and implement multiple classes for each of these commands which as I have mentioned before it will be extremely tedious. My Command pattern design is more scalable than my Memento Pattern designs, as it will allow me to perfom complex commands such as undoing/redoing z-list swaps if I were to implement that into my code.

In conclusion I believe for me, since I gained a better understanding of Command Pattern design through demonstrator guidance , and my code implementation of Command Patter worked better than my Memento one, Command works better for my design than Memento does.