In-Class Exercise 4: Moving Line: Strategy Design Pattern

Instructor: Mackale Joyner

Course Website: https://www.clear.rice.edu/comp504

Goals for this exercise

- Get familiar with Strategy design pattern
- Determine when to use the Singleton design pattern

Important tips and links

NOTE: The instructions below are written for Mac OS and Linux computers, but should be easily adaptable to Windows with minor changes. For example, you may need to use \ instead of / in some commands.

1 Checkout Github Classroom In-Class Exercise 4 Repo

First, you'll need to accept the exercise 4 repository by visiting GitHub ex4. Check out your github classroom exercise 4 starter code from the remote github repo in IntelliJ from either the "Check out from Version Control" option on the welcome screen or the "Checkout from Version Control" option accessible under the top menu's VCS tab. Use the plus button in the pop-up window to add your GitHub ex4 repo. You should see a ex4 directory created in your working directory with the source code for this exercise.

2 Exercise 4

For this exercise, you will use the Strategy and Singleton design patterns to draw a line that can move across the canvas. Recall that the Strategy design pattern separates the context from the context behavior. As a result, the context strategies can be swapped out at anytime. In this exercise, the model will determine where to initially draw the moving line and how fast the line should move across the canvas. For a moving line, the view will update the line position every .2 seconds. The controller will process REST method requests from the view and call the appropriate function in the model to update the line position. The controller may also signal the view to switch strategies. The next sections detail what you'll need to implement in the model, view, and controller.

2.1 Model

You will need to modify the `MovingLine` class in the model. The `MovingLine` constructor calls the `init` method to initialize the line location, velocity, and strategy. The moving line will have a randomly chosen velocity. The line will begin moving once it is drawn on the canvas. As a result, there's no need to have separate `draw line` and `move line` buttons as we did in exercise 3. The initial strategy is the horizontal strategy. The horizontal strategy moves the line horizontally to the right. There is no collision detection with the right wall of the canvas. As a result, the line will keep moving right until the line moves off the canvas. The `MovingLine` class has the following `TODO`s:

- Update the line location given the velocity x and y components using the strategy design pattern

- Switch strategies when the user clicks on the `switch strategy` button. The horizontal strategy switches to the vertical strategy. The vertical strategy moves the line vertically down. The line keeps this strategy until the `switch strategy` button is clicked again. The vertical strategy switches to the composite strategy. The composite strategy has the horizontal and vertical strategies as children. Recall, the only job the composite strategy has is to call each child's update operation. Again, the line keeps the composite strategy until the `switch strategy` button is clicked again. The composite strategy switches back to the horizontal strategy. You'll implement the `updateState` method for each concrete strategy. Be sure and use the Singleton design pattern in the model where appropriate.

2.2 Controller

You'll need to implement the endpoints in the controller. The controller has the following endpoints:

- `line` initially draws a line
- `update` updates the line using the line strategy
- `switch` switches strategies
- `reset` clears the canvas

2.3 View

The view needs to ensure that the line starts moving immediately after it's drawn on the canvas. The view will update the line position every .2 seconds. You should now be able to start up a local server, draw a moving line, and switch strategies every time the user clicks the `switch strategy` button. Complete the `TODO`s in `view.js`.

3 Demonstrating and submitting the exercise

You'll need to host your app on heroku. The app name should be [netid]-ex4-move-line. Place the app name and your name in the README file.

Please don't forget to commit and push your work to your github classroom repository. To perform a git commit, select VCS on the menu and click on "Commit...". Add a commit message and click on the `Commit` button. To push the local changes, select VCS on the menu, highlight the Git option and select "Push...".