In-Class Exercise 3: Drawing a Moving Line

Instructor: Mackale Joyner

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

Goals for this exercise

- Get familiar with JavaScript intervals
- Animate a moving object
- Think through use cases

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 Exercise 3 Repo

First, you'll need to accept the exercise 3 repository by visiting GitHub ex3. Check out your github classroom exercise 3 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 ex3 repo. You should see a ex3 directory created in your working directory with the source code for this exercise.

2 Exercise

For this exercise, you'll use the MVC design pattern to draw a line that moves across the canvas. You'll get experience using the JavaScript `setInterval` and `clearInterval` functions for line animation. The model will determine where to initially draw the line and the view determines how fast the line should move across the canvas. For a moving line, the view will update the line position every 0.2 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 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 has an argument `velX` that represents the line's horizontal velocity. The line does not have a vertical velocity. The `MovingLine` class has 2 `TODO`s. You'll need to complete the following:

1. Initialize the line with the start location. The starting line location is `(0,0)` to `(200,100)`.
2. Modify the `update` method to change the line position given the velocity.

2.2 Controller

There are 3 `TODO`s in the `LineDrawController` class. The controller will interact with the model to support the following behavior:

1. When the user clicks the `Draw Line` button, the line should be drawn in it's original location.
2. Line position updates occur when the user clicks the `Move Line` button. The line's velocity controls how fast the line moves across the canvas. The `Move Line` button should only move a line that has been drawn.
3. When the user clicks the `Reset` button, the line's position is reset to the original location. It does not move until the user clicks the draw button and then the move line button.

2.3 View

There are 4 `TODO`s in `view.js`. They are the following:

1. Modify the `GET` request for the `/line` endpoint. A non-moving line should be drawn on the canvas when the user clicks on the `Line Draw` button.
2. Update the `setUpdateFreq` function to request a line update every 0.2 seconds after the user clicks the `Move Line` button. You'll need to decide what to do if a user clicks on the buttons out of order or hits a button multiple times.
3. Change the `GET` request for the `/update` function to update the line position.
4. Modify the `GET` request for the `/reset` endpoint. The line should no longer appear on the canvas when the user clicks the `Reset` button.

3 Submitting Exercise

In the README file, add your name and the heroku app name `[netid]-ex3-move-line`. 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...".