Introduction

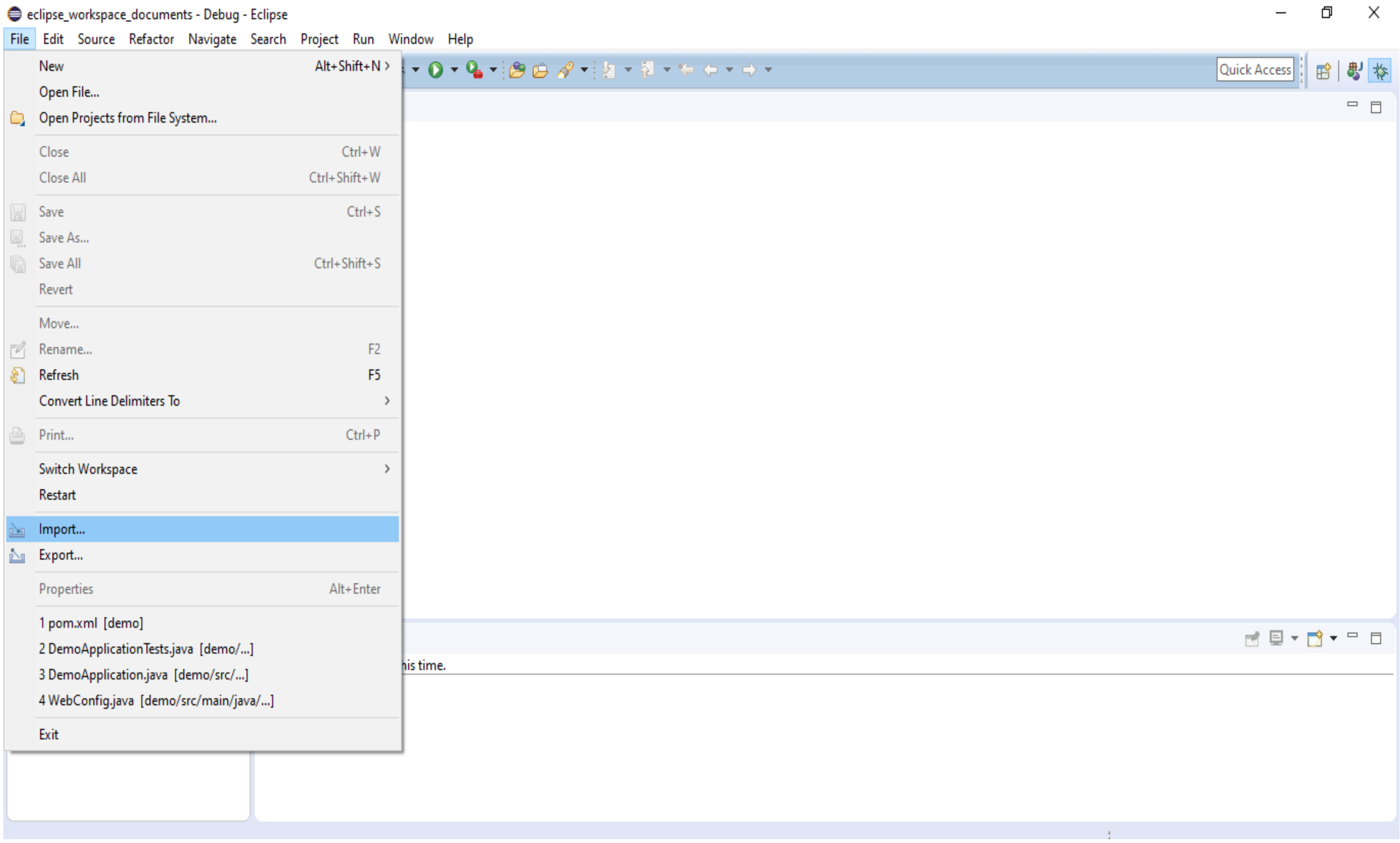
For this project, I used Java to create a line game (see game details below). This was a coding assessment where I was provided with a web client (HTML/CSS, JavaScript) to render the game and manage user interactions (the client is dumb, and knows nothing about the game). My role was to use Java and HTTP API to implement the server-side game logic, maintain the game state, and process requests from and send responses to the web client.

This program uses Spring Boot, specifically Spring Framework 2.5.0 with the dependency Starter Web.

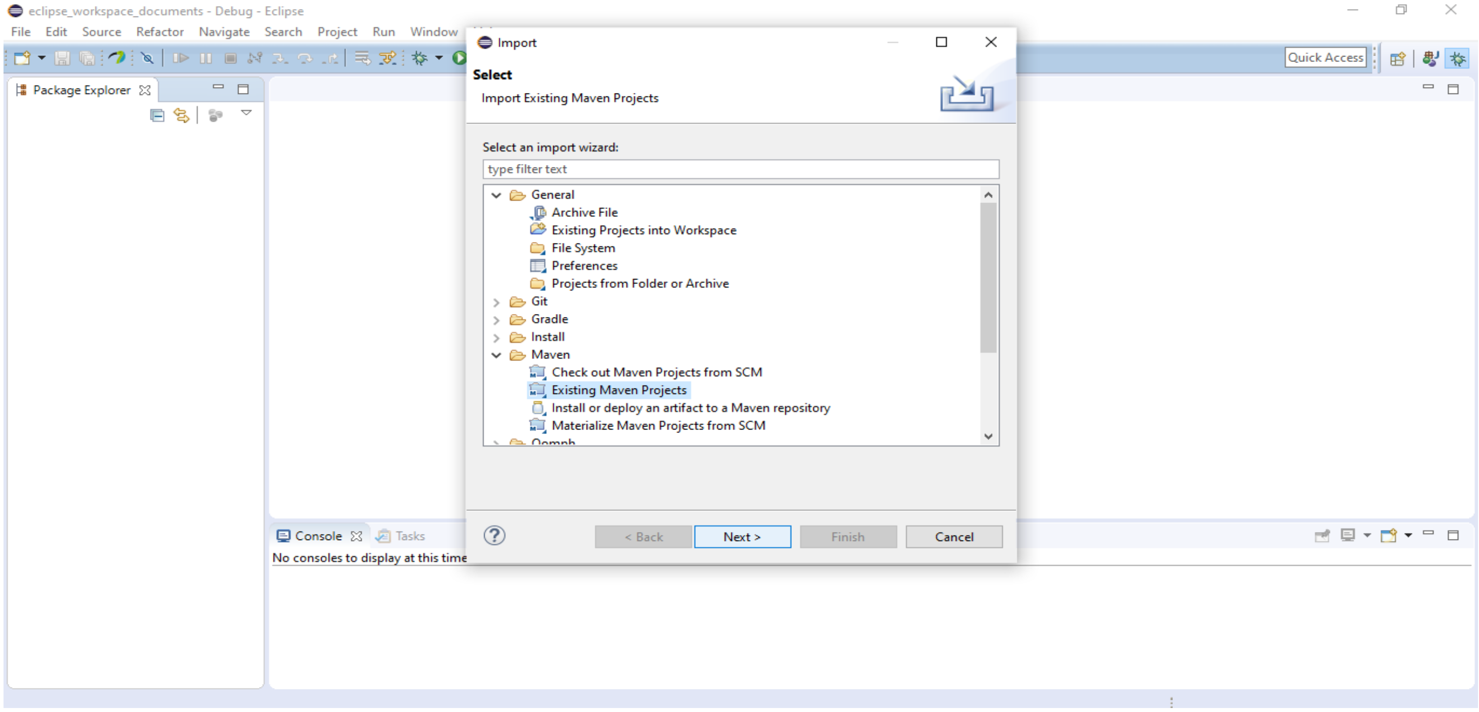
Running the Application

For this project I did my work using eclipse. To start off:

1.) Download the project -> Place it in a good spot (ex: Desktop) -> Unzip project
2.) Open eclipse -> Click file in the top left corner on the toolbar -> Import

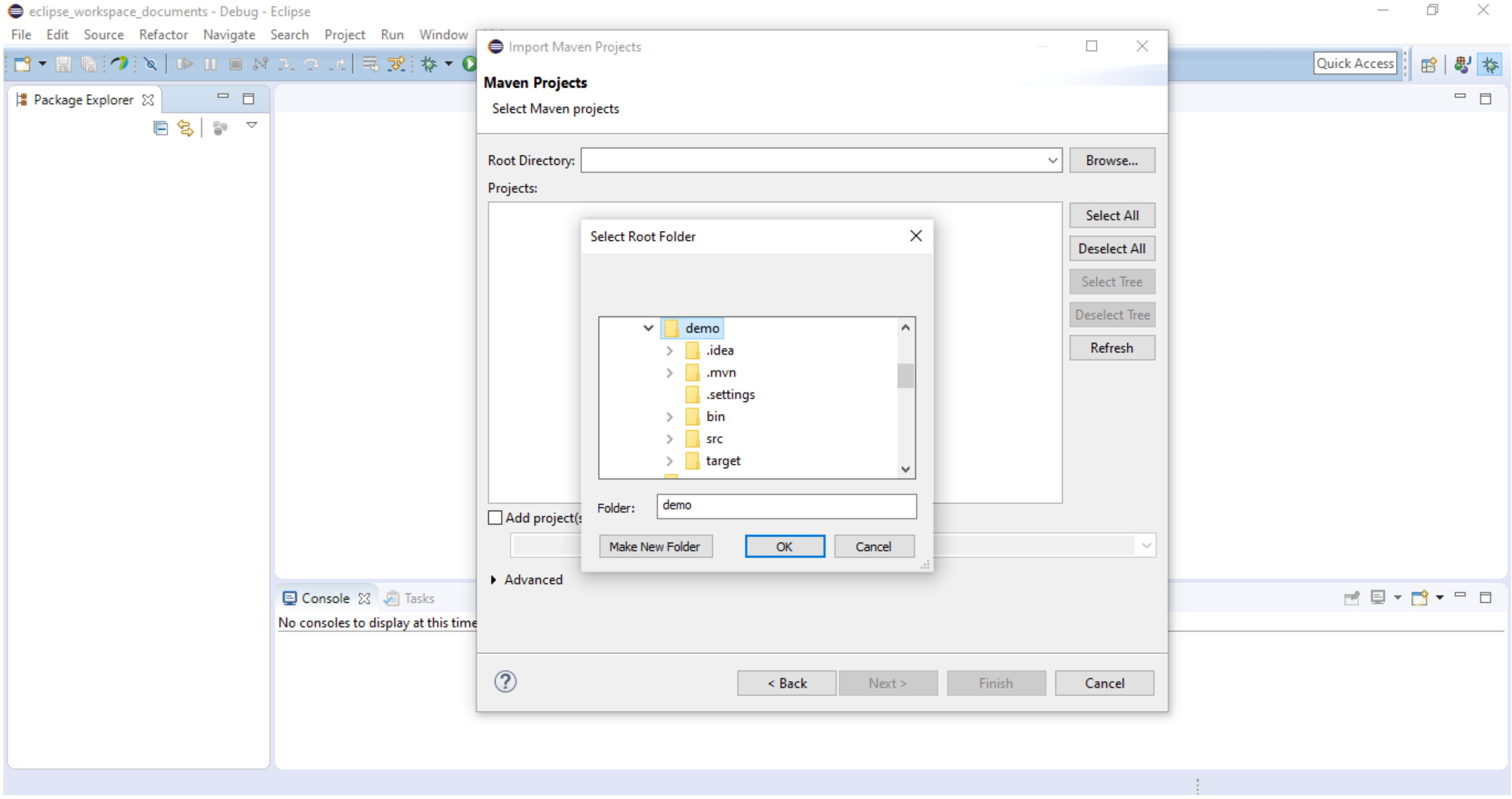

eclipse_workspace_documents - Debug - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
New Alt+Shift+N >
Open File...
Open Projects from File System...
Close Ctrl+W
Close All Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To >
Print... Ctrl+P
Switch Workspace >
Restart
Import...
Export...
Properties Alt+Enter
1 pom.xml [demo]
2 DemoApplicationTests.java [demo/...]
3 DemoApplication.java [demo/src/...]
4 WebConfig.java [demo/src/main/java/...]
Exit
his time.

3.) In the pop-up window -> Drop down maven folder -> Choose existing maven project


eclipse_workspace_documents - Debug - Eclipse
File Edit Source Refactor Navigate Search Project Run Window
Quick Access
Package Explorer
Import
Select
Import Existing Maven Projects
Select an import wizard:
type filter text
General
Archive File
Existing Projects into Workspace
File System
Preferences
Projects from Folder or Archive
Git
Gradle
Install
Maven
Check out Maven Projects from SCM
Existing Maven Projects
Install or deploy an artifact to a Maven repository
Materialize Maven Projects from SCM
Oomph
< Back
Next >
Finish
Cancel
Console
Tasks

No consoles to display at this time

4.) In the root directory box -> Click the browse button -> Select the unzipped project folder wherever it is located for you (ex: Desktop/Demo)

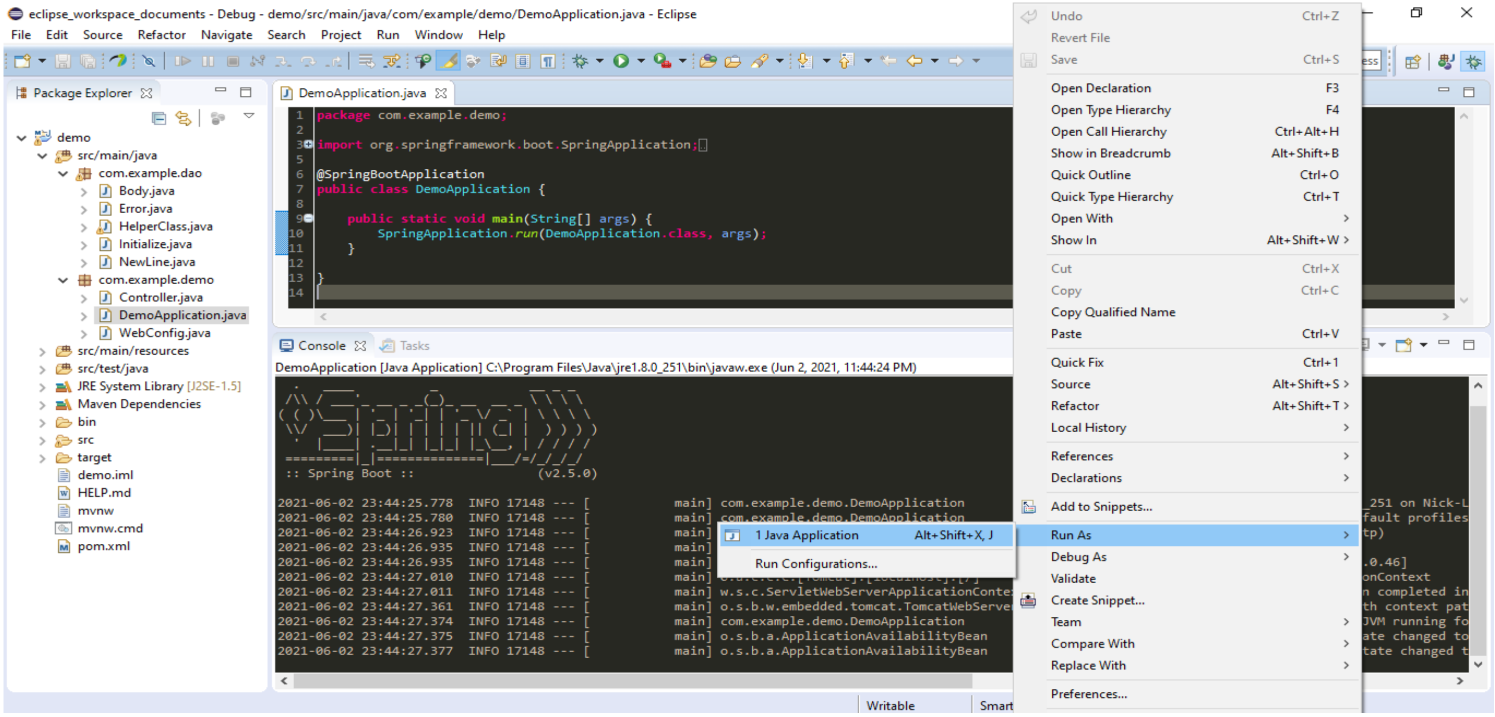


Folder: demo

No consoles to display at this time

5.) On package explorer side tab -> Drop down demo -> Drop down src/main/java -> Drop down com.example.demo

6.) Open DemoApplication.java -> Right Click in the text area -> Run As -> Java Application



```
package com.example.demo;

import org.springframework.boot.SpringApplication;

@SpringBootApplication
public class DemoApplication {

    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }

}
```

DemoApplication [Java Application] C:\Program Files\Java\jre1.8.0_251\bin\javaw.exe (Jun 2, 2021, 11:44:24 PM)

```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.5.0)

2021-06-02 23:44:25.778  INFO 17148 --- [           main] com.example.demo.DemoApplication
2021-06-02 23:44:25.780  INFO 17148 --- [           main] com.example.demo.DemoApplication
2021-06-02 23:44:26.923  INFO 17148 --- [           main]
2021-06-02 23:44:26.935  INFO 17148 --- [           main]
2021-06-02 23:44:26.935  INFO 17148 --- [           main]
2021-06-02 23:44:27.010  INFO 17148 --- [           main]
2021-06-02 23:44:27.011  INFO 17148 --- [           main] w.s.c.ServletWebServerApplicationConte
2021-06-02 23:44:27.361  INFO 17148 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServe
2021-06-02 23:44:27.374  INFO 17148 --- [           main] com.example.demo.DemoApplication
2021-06-02 23:44:27.375  INFO 17148 --- [           main] o.s.b.a.ApplicationAvailabilityBean
2021-06-02 23:44:27.377  INFO 17148 --- [           main] o.s.b.a.ApplicationAvailabilityBean
```

7.) Locate the index.html file (technical-assessment/client/index.html) -> right click the file -> Open with -> Select your browser of choice (ex: Chrome)

8.) Now play the game. See the Game Rules and example game below on how to play the game. You can hit refresh to start the game or to reset and start a new game

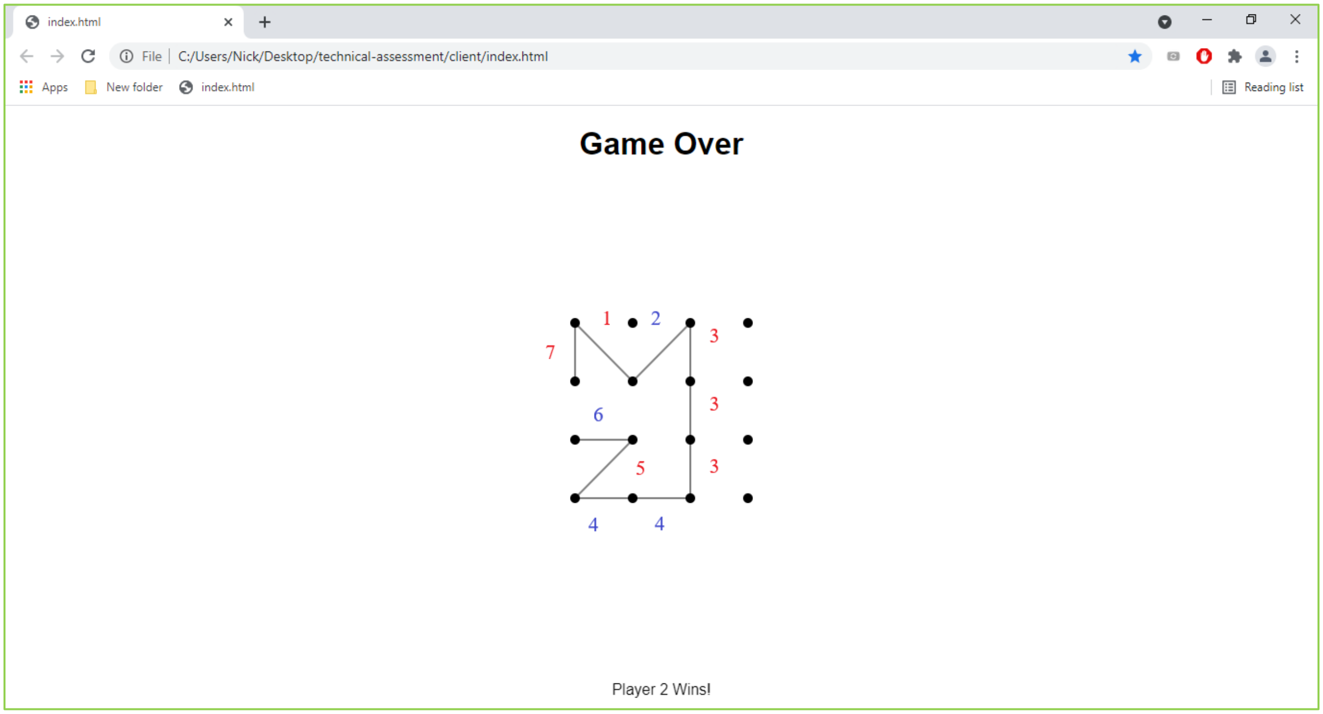
Game Rules

- The game is played on a 4x4 grid of 16 nodes.
- Players take turns drawing octilinear lines connecting nodes.
- Each line must begin at the start or end of the existing path, so that all lines form a continuous path.
- The first line may begin on any node.
- A line may connect any number of nodes.
- Lines may not intersect.
- No node can be visited twice.

- The game ends when no valid lines can be drawn.
- The player who draws the last line is the loser.

Example Game

Each move is numbered. Lines that connect more than two nodes have each segment numbered. Player 1 made the odd numbered moves and Player 2 made the even numbered moves. Player 1 made the first move (1) and was forced to make the last move (7). Thus, Player 2 won.


index.html
File
C:/Users/Nick/Desktop/technical-assessment/client/index.html
Apps
New folder
index.html
Reading list
1
2
7
3
6
3
5
3
4
4

Game Over

Player 2 Wins!

Attribution

- **Game designer**: Sid Sackson
- Spring Framework 2.5.0 with the dependency Starter Web
- **External code**: I used some code from the following page, which I then modified for my specific cases. It was used as a helper method, finding the orientation of a line and to see if two lines pass through one another, for the doesIntersect method in my HelperClass.java file. For more detailed attribution, see my comments in that file.
  - https://www.geeksforgeeks.org/check-if-two-given-line-segments-intersect/