Tiwaloluwa Ojo

100700622

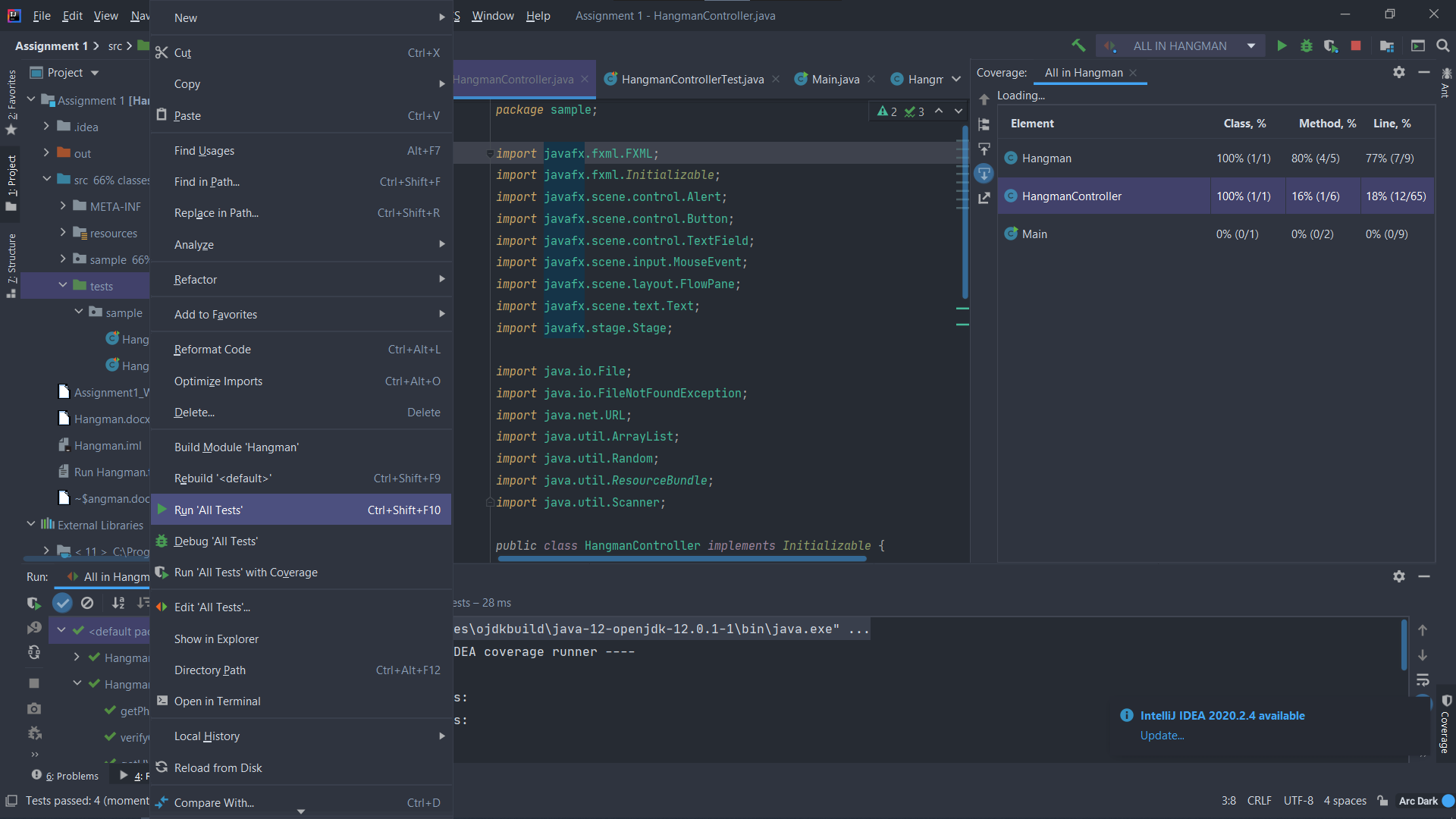
Title: Hangman

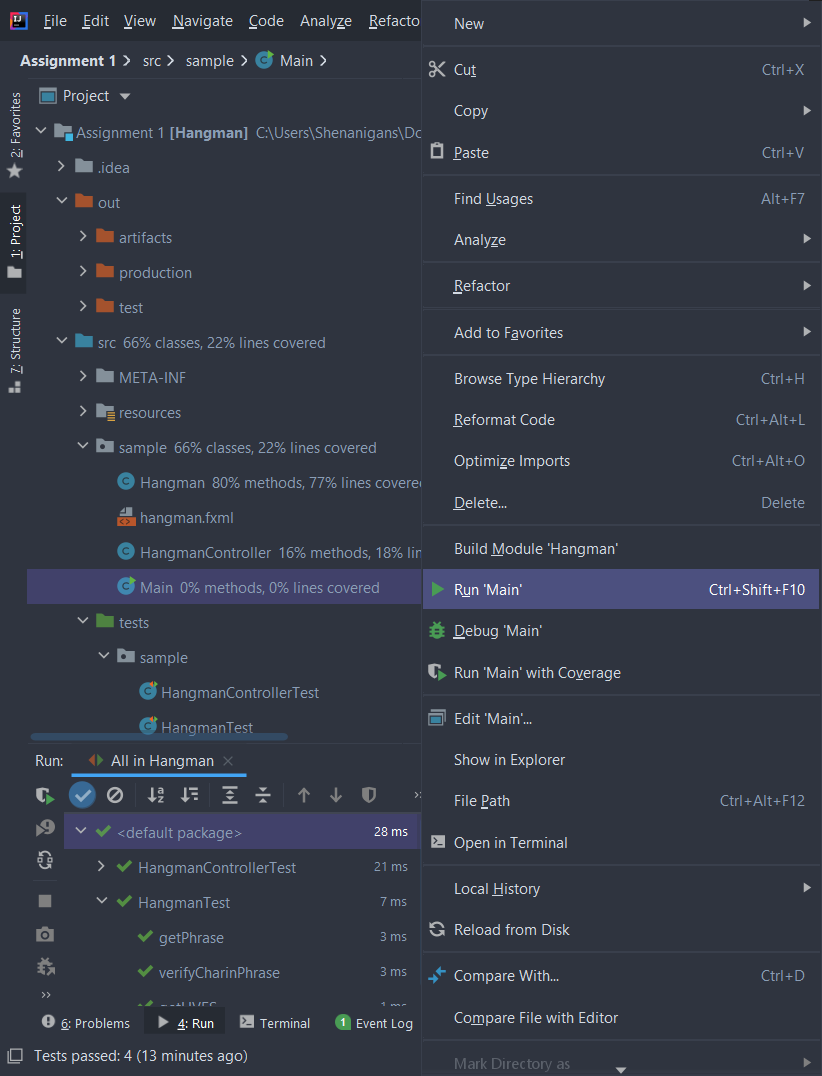
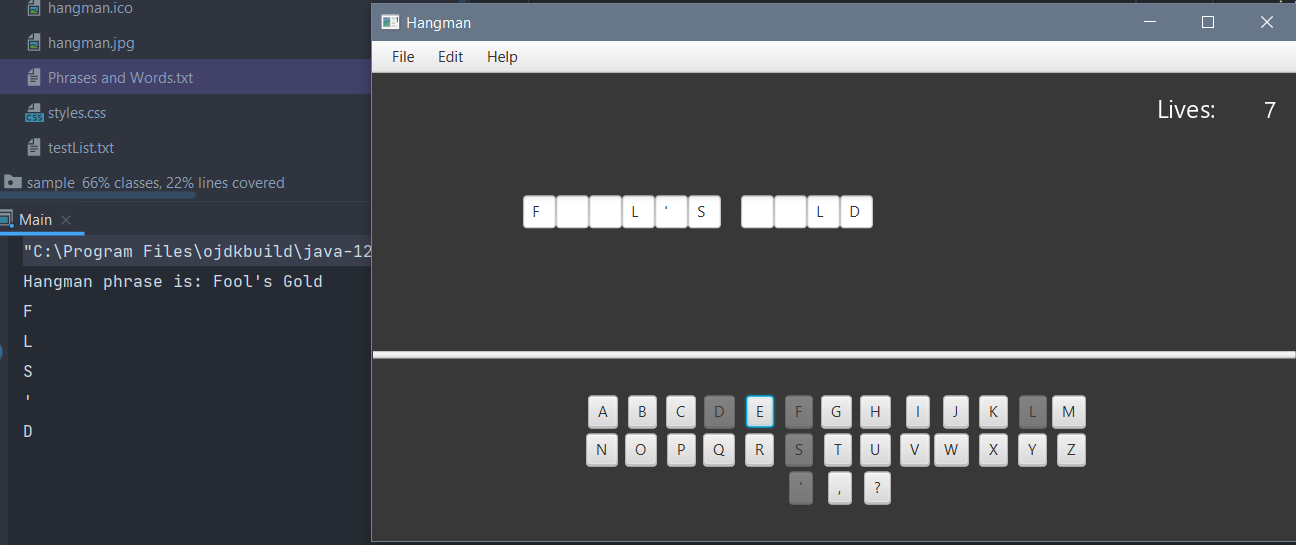
Tools: Java, JavaFX, Intellij, JUnit

**Required**: [Installing](https://openjfx.io/openjfx-docs/#install-javafx) and setting up [JavaFX](https://gluonhq.com/products/javafx/) on Intellij: <https://openjfx.io/openjfx-docs/>

\*Note\*: Upon following the installation instructions, you may need to replace the paths I have setup in my project structure. You may also run the application by navigating to the [Getting Stated with JavaFX](https://openjfx.io/openjfx-docs/#install-javafx) page or running from within a Intellij.

1. Software Process chosen
   1. I chose the Agile Model process due to easy modifiability. Due to JavaFX being a new framework I will be working with for a first time, utilizing an agile process allows me to implement newer features and create optimizations to existing code without having to develop from scratch. It also allows me to publish a working prototype into production (could not be done due to a build error with the JDK) and simultaneously continue developing.
2. Hangman
   1. Description:
      1. Hangman is a small game at which a player tries to determine a phrase or word by analyzing the characters used and slots remaining. If the word/phrase isn’t found before the player runs out of life, the player will loose.
   2. Functionalities
      1. Hangman has an intuitive UI consisting of the slots the character will be entered and the buttons to cause the event. It follows an MVC style architecture of which the [*Hangman.java*](src/sample/Hangman.java)*,* [*Hangman.fxml*](src/sample/hangman.fxml)*, and* [*HangmanController.java*](src/sample/HangmanController.java) are designed to be respectfully. It uses JavaFX as a framework to develop as an application and tests using Junit.
   3. Difficulties
      1. A major difficulty I came across while writing my tests was trying to get a larger coverage. This was rendered unsuccessful as most of my code was bonded to the JavaFX components and methods. I would require another library to test this aspect of my code. However, I was able to write some tests and run coverage on them by splitting my code into testable methods.
      2. I also wasn’t able to build and executable for this project as Oracle removed this functionality JDK.
   4. Tests: Right click on tests to:
      1. Run all tests or Run all tests with coverage
      2. Click highlighted icon to generate coverage report in browser(Can also be accessed from “../out/test/Hangman/sample”)



* 1. To run and play Hangman right click and select “**Run Main”**
     1. 
     2. 
     3. 