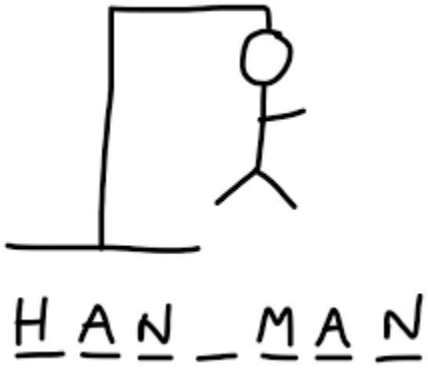
Object-Oriented Hangman



Status summary:

Team: Ainsley Fletcher, Arthur Barbosa, Lindy Zhang

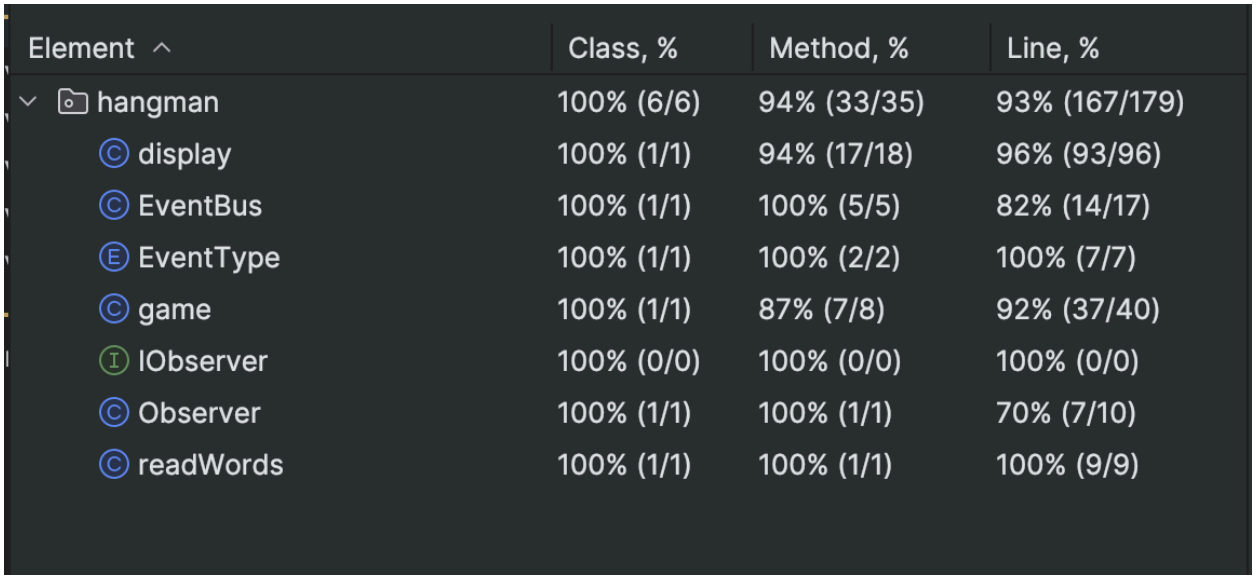
Link to recorded demo:
https://drive.google.com/file/d/1wST9OdbgmNgzORpMDjdOKpEUdxBoSxSb/view?usp=sharing

Work done: We have implemented most of the Hangman game knowledge and for now we are doing it all in the terminal. As a team, the three of us got together and pair-programmed most of it, however, Ainsley focused mostly on the Event Bus implementation, Lindy mostly on the game logic, and Arthur on the terminal display functionality and words/definitions.

Changes/Issues encountered: Testing the main method when using terminal input from the user was challenging. We changed how we'll go about the difficulty in the game, initially, we wanted to do something related to giving the user fewer guesses, but since that goes against the core idea of Hangman we have decided to change the difficulty option to select harder words (i.e fewer vowels and more uncommon letters) from our word bank

Patterns:
-Singleton Pattern: asserts that there's only one event bus.
-Event Bus: Reduces coupling and makes code easy to refactor when we change the User Interface. Currently being used to print alerts for the game.

Test coverage:



Element	Class, %	Method, %	Line, %
hangman	100% (6/6)	94% (33/35)	93% (167/179)
display	100% (1/1)	94% (17/18)	96% (93/96)
EventBus	100% (1/1)	100% (5/5)	82% (14/17)
EventType	100% (1/1)	100% (2/2)	100% (7/7)
game	100% (1/1)	87% (7/8)	92% (37/40)
IObserver	100% (0/0)	100% (0/0)	100% (0/0)
Observer	100% (1/1)	100% (1/1)	70% (7/10)
readWords	100% (1/1)	100% (1/1)	100% (9/9)

Class Diagram:

BDD Scenarios:

Scenario 1:
Feature: Making a correct guess in the Hangman game
Scenario: The player guesses a letter correctly
  Given the secret word is "class"
  And the current guessed word is "_ _ _ _ _"
  When the player guesses the letter "c"
  Then the guessed word should be updated to "c_ _ _ _"
  And the number of incorrect guesses should remain the same
  And a message "Correct guess!" should be displayed
  And the game should not be over

Feature: Making an incorrect guess and losing the game

Scenario 2
Feature: The player makes a final incorrect guess
  Given the secret word is "syntax"
  And the current guessed word is "s _ n _ t a _"
  And the player has already made 5 incorrect guesses
  When the player guesses the letter "o"

Then the guessed word should remain "s _ n _ t a _"
And the number of incorrect guesses should increase to 6
And a message "Incorrect guess!" should be displayed
And the hangman graphic should be completed
And the game should be over
And a message "Game over! The correct word was 'syntax'" should be displayed

Plan for Next Iteration:

Complete all functionalities for the game, including more patterns and making the game more thorough with functionalities such as hints and difficulty. The main plan for the next iteration will be to use Spring Boot and turn the game from a terminal game into a web-based experience.

UML