Kai Chen
Instructor : JimR

CS3330 Final Project Documentation

May 2022

Hello! To showcase the skills I learned this semester in CS3330, I designed a hangman game with a few novelties.

Overview

I, along with many other people, feel the need to take frequent breaks from being productive and relax by watching media and playing games. I created "Hangman 2*", which takes user input to create an experience similar to the hangman games many of us played during childhood. Users can either enter in individual characters or whole strings to input their guesses and they can view their collective game score and win streak after their game, in addition to their highest score.

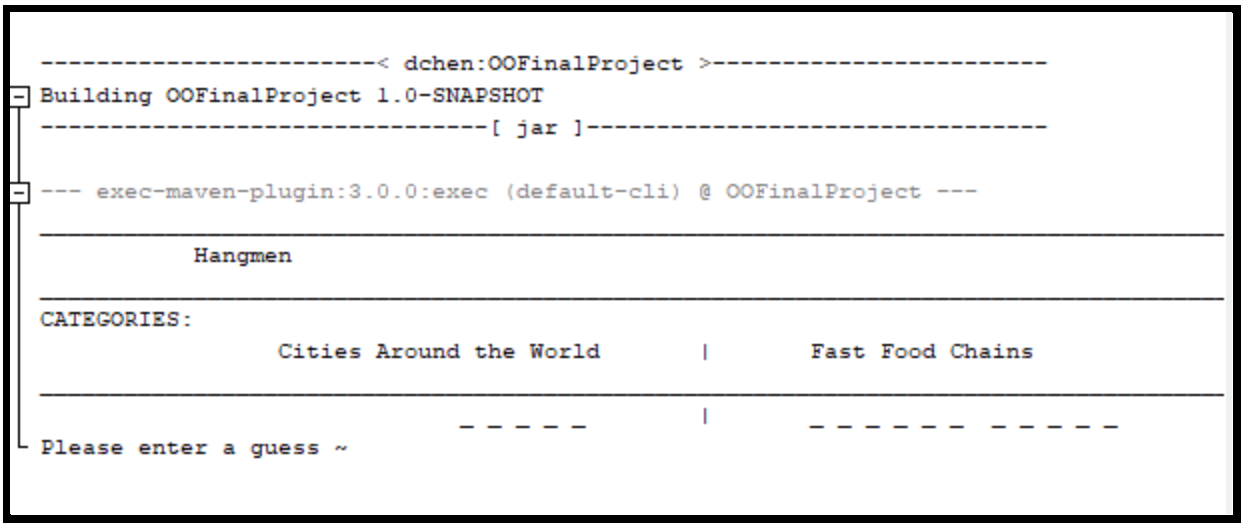
* I couldn't decide if I liked "Hangman 2" or "Hangmen" as a name for this project more, so you'll probably see me flip-flopping between the two names across the project.

Purpose

The purpose of this project was to create a lighthearted game that won't be too much of a distraction when someone wants to take a break. The game is not time-sensitive, meaning users can put it away and come back later if they have other obligations. The novel factor of solving two strings at once makes the game a little easier, as less thought needs to be put into a specific string. Additionally, the game contains many historical figures and vocab terms that the user can refresh themselves on if they're not already familiar with them. Mainly, though, this application was just because I get bored pretty easily and this application will help me entertain myself during those boring times.

Functionality

After booting up the application, the user is greeted with a screen displaying two columns, with each column displaying a category name that corresponds with the column's string to be solved. The two strings are displayed under the categories, but are encoded for the user to solve. The category is determined through a random number generator, and the string is determined through another random number generator within the category. The application uses a scanner to scan the user's guesses, which can either be singular letters or full strings. After scanning user input, the application uses multiple string and character functions to determine input validity. If a character is not legal (not a letter), the application will either throw an exception or go to the next character in the string depending if the user entered a string or a character. Similarly, the application uses a hashmap to store user input to check for duplicates efficiently. If the function detects a duplicate, it will either ignore it or throw an exception, depending on if the user entered a character or string. When the character enters a legal character that matches with any character in a string, the display will update and show the new string with the corresponding characters revealed. If both strings are fully revealed, the game ends. If the user enters a legal character but it is not in either string, they will lose a life, starting from a total of 6 lives. When the user reaches 0 lives, the game ends and the hidden strings are revealed. Once the game ends, the user is prompted to see if they'd like to continue playing.


------------------------< dchen:OOFinalProject >------------------------
Building OOFinalProject 1.0-SNAPSHOT
--------------------------------[jar]---------------------------------
--- exec-maven-plugin:3.0.0:exec (default-cli) @ OOFinalProject ---
Hangmen
CATEGORIES:
Cities Around the World | Fast Food Chains
_ _ _ _ _ | _ _ _ _ _ _ _ _ _ _ _ _
Please enter a guess ~


_ _ _ _ _ | _ _ _ _ _ _ _ _ _ _ _ _
Please enter a guess ~
aeious
You guessed : A
You guessed : E
You guessed : I
Ouch! You now have 5 lives
You guessed : O
Ouch! You now have 4 lives
You guessed : U
Ouch! You now have 3 lives
You guessed : S
Ouch! You now have 2 lives
_ _ A _ A | _ A _ E _ A _ _ E A _
Please enter a guess ~
m
You guessed : M
Ouch! You now have 1 lives
_ _ A _ A | _ A _ E _ A _ _ E A _
Please enter a guess ~
s
You already guessed that letter!
Please enter a guess ~
.
That's not a letter!
Please enter a guess ~
tnm
You guessed : T
Ouch! You now have 0 lives
YOU LOSE

Required Elements

Classes:

1. Hangman.java
2. Game.java
3. ActualGame.java
4. Category.java
5. main.java

Subclasses:

Extend Category.java:

1. Bands.java
2. Characters.java
3. Cities.java
4. FastFood.java
5. Holidays.java
6. Landmarks.java
7. Leaders.java
8. NerdWords.java
9. PotatoChip.java
10. Superheroes.java

Extend Exception:

1. BadInput.java

Abstract Classes:

1. Category.java

Collections:

1. HashSet in Game.java (line 23)

Exception Handling:

1. Try-catch statement in Game.java (line 73-139)