Assignment 5 – Hangman

Overview

- Implement the game of Hangman.
- You may use the design you created in the Web Design course or a new one.
- You may work in pairs (teams of 2).

Background

Game Controller

You are given a file, “GameController.js”, that manages the state of the game. It contains three public methods.

- The method *newGame* accepts a word to play a game with. Note that the word must be at least 4 characters long.
- The method *processLetter* accepts a letter, fills in blanks in the game word, decides the number of guesses remaining, and determines if the state of the game (in progress or win or lose)
- The method *report* returns an object containing the state of the game, including the number of guesses remaining and which letters of the word have been guessed correctly.

Extensive documentation on the GameController is available at:
http://assignment0.com/hangman/docs/gamecontroller/index.html

Vocabularies

You are given the file, “vocabularies.json”, that contains four categories and multiple words in each category. Use these words to allow the user to choose a word at random from a category.

Design Features

The following features are required:

- New Game Button
  - When clicked, it must be disabled until the current game is over.

- Category Panel
  - This panel allows the user to choose a word category.
  - It should be visible only when a new game is started.
  - It should contain a “OK” button – once the user selects a category and presses the “OK” button, the panel should be hidden.

- Game Word Display
  - It should start out as all blanks – one per letter in the word to guess – and be filled in as the game progresses.

- Letter Guess Component
  - All letters should be available at the beginning of the game.
  - Once a letter has been guessed, it should no longer be available.
  - It must be clear to the user which letters have been used and which are still available to guess.
  - The letter guess components must be disabled when a game is won or lost.

- Hangman Graphic
  - It should start out as just the gallows, and then change with each incorrect guess.
  - When a game is lost, the Hangman figure should be fully drawn.
  - When a game is won, the figure should be incomplete.

- Guesses Remaining Counter
  - The counter should start at the number of guesses allowed when a new game is started and decrease by 1 with each **incorrect** guess.
  - When a game is lost, the counter should be 0.
  - When a game is won, it should be greater than 0.

- Win/Loss Indicator
  - When a game has ended, there should be a clear indication of whether it is a win or a loss.

Game Play

When the “New Game” button is pressed

- Display the Category Panel.

When the “OK” button is pressed (on the Category panel)

- Get the selected category and choose a word at random from that category.
- Hide the Category Panel.
- Call the *newGame* function to start a new game with the chosen word.
- Disable the New Game Button. (A new game cannot be started until the current game is finished.)
- Place the correct number of blanks in the Game Word Display, one blank per letter.
- Ensure that all letters in the Letter Guess Component are enabled.
- Display the opening Hangman Graphic – the one with just the gallows.
- Ensure that the Guesses Remaining Counter is set to the number of guesses allowed.

When the user makes a guess

- Disable that letter in the Letter Guess Component so that it cannot be guessed again.
- Call the *processLetter* function with the letter.
- Call the *report* function to get the updated game state.
- If the letter is in the word:
  - Reveal all occurrences of the letter in the Game Word Display.
- If the letter is NOT in the word:
  - Update the Hangman Graphic
  - Decrement the Guesses Remaining Counter.
- If the game has been won or lost:
  - Disable all letter inputs.
  - Enable the New Game Button.
  - Update the Win/Loss Indicator to either a win or a loss.