MIPS Hangman

Colin Chenoweth

Overview

- Hangman
  - Enter word/phrase to be guessed
  - Guess letters or the entire word/phrase
  - Keeps track of letters guessed

Motivation/Reasoning

- Learning MIPS
- No GUI
- Individual Project

Data

```
intro_msg:   .asciiz          "Guessers Look Away"
word_prompt:      .asciiz          "Enter word or phrase to be guessed (in all lowercase):"
guess_prompt:     .asciiz          "Enter a letter or the full word/phrase to guess (in lowercase):"
already_guessed_msg:   .asciiz          "Already guessed this letter, try again."
yes_in_word_msg:          .asciiz          "Correct, that letter is in the word."
not_in_word_msg:          .asciiz          "Sorry, incorrect guess."
lives_msg:    .asciiz          "Lives remaining:  "
game_lost_msg:    .asciiz          "You Lost :("
game_won_msg:   .asciiz          "You Won!! :)"
word_msg:  .asciiz          "The word was:  "
unguessed_letters:        .asciiz "abcdefghijklmnopqrstuvwxyz"
guessed_word: .space 256
word: .space 256
word_length: .word 0
current_letter: .space 256
letter_valid: .word 0                # 0 = not guessed, not in word, 1 = not guessed, in word, 2 = already guessed
lives: .word 6
```

General Code Structure

main

guess

- game_board
- check_guess_length
- check_guess_phrase
- check_guess_letter

game_over

Helper Methods

main: get_word_length, add_space

guess

- game_board: display_guessed_word, check_equal
- check_guess_length
- check_guess_phrase: wrong_guess
- check_guess_letter: check_already_guessed (unguessed, already_guessed), check_in_word (fill_in_letter), wrong_guess, yes_in_word_outputs

game_over: game_won, game_lost, display_word

new_line, return

Complications

- Debugging
- li vs lb vs lw vs la
- beq

Demo