Report.

HOCHSCHULE
HAMM-LIPPSTADT

M U H A M M A D
A L I

20.06.2024

THE HANGMAN'S CHALLENGE: A THEMED C ADVENTURE

Hangman is a word guessing game where players have to uncover a hidden word by guessing eachn letter at a time. The game involves a theme selection, such as Pirates, Crime Drama or Comics which then have corresponding word guessing levels. The maximum number of incorrect guesses is six letters, before the "hangman" is fully drawn or depicted through the special live-action themed levels.

The game includes features like:

- A Main Menu system
- Winning Conditions
- Losing Conditions
- Gameplay Loop (responsible for handling the core mechanics of guessing letters, updating the display, and checking win/lose conditions)
- Incorrect guess counters

Hangman combines elements of word puzzle solving with strategic guessing, making it both entertaining and challenging for players of all ages.

https://github.com/MAliSohail/Hangman

1. Schematics

Operational Mechanism Overview:

- **Initializatialization**
  - SDL Initialization: Initializes SDL for video (SDL_INIT_VIDEO) and optionally audio (SDL_INIT_AUDIO).
  - SDL_ttf Initialization: Initializes SDL_ttf for handling TrueType fonts.
  - SDL_image Initialization: Initializes SDL_image for loading PNG images.

- **Main Menu**
  - Display Main Menu: Shows the main menu screen with options to start the game or quit.
  - User Input Handling: Waits for user input (keyboard or mouse) to proceed.

- **Theme Selection**
  - Display Theme Selection Menu: Shows a menu to select the game theme (Pirates, Crime Drama, Comics) using SDL_image for thematic backgrounds.
  - User Input Handling: Allows the user to choose a theme using keyboard shortcuts (1 for Pirates, 2 for Crime Drama, 3 for Comics).

- **Game Loop:**
  - Word Selection: Randomly selects a word and its corresponding hint from premade lists depending on the chosen theme.
  - Guess Handling: Accepts user input (single letter guesses) and checks it against the chosen word for correct or incorrect guesses.
  - Game State Updates: Tracks and updates the number of incorrect guesses (leading to drawing of hangman parts or images) and correct guesses before the correct word is guessed.
  - Game Over Check: Determines game over conditions (win or lose) based on correct guesses and incorrect attempts.

- **End Screens:**
  - Win Screen: Displays a congratulatory screen with the guessed word upon successful completion.
  - Lose Screen: Displays a game over screen with the correct word when the hangman figure is fully drawn.

Initialization and Main Menu Functions:

- **main(int argc, char argv[])**
  - Purpose: Entry point of the program.
  - Functionality: Initializes SDL, SDL_ttf, and SDL_image libraries. Creates a window and renderer for graphics. Loads necessary fonts and images. Enters an infinite loop to display the main menu until the correct input is eneterd, handles theme selection, and starts the game.

- **showMainMenu(SDL_Renderer renderer, TTF_Font font)**
  - Purpose: Displays the main menu and waits for user input to start the game.
  - Functionality: Loads a background image for the menu. Listens for events (like key presses or window close events) to start the game when the Enter key is pressed.

- **showThemeMenu(SDL_Window window, SDL_Renderer renderer, TTF_Font* font)**
  - Purpose: Displays the theme selection menu and returns the selected theme.
  - Functionality: Loads a background image for the theme selection menu. Records for number key presses (1, 2, 3) corresponding to different themes (Pirates, Crime Drama, Comics). Returns the selected theme enum value.

Game Setup :

- **getRandomWordAndHint(Theme)**
  - Purpose: Retrieves a random word and its hint based on the selected theme.
  - Functionality: Uses the rand() function with srand() initialized by time(NULL) to generate a random index within the array of WordHintPair structs corresponding to the selected theme. Returns a pointer to the randomly selected WordHintPair.

Gameplay Functions:

- **drawWrongGuessImage(SDL_Renderer renderer, int wrongGuesses, Theme theme)**
    - Purpose: Draws the hangman or appropriate image based on incorrect guesses and based on the theme.
    - Functionality: Uses SDL functions to draw different parts of the hangman (for Pirates) or dynamically load subsequent other images (Crime Drama and Comics).

- **drawWord(SDL_Renderer renderer, const char word, const char guessedLetters, int numGuessedLetters)**
    - Purpose: Draws the guessed letters on the word spaces or underscores for unguessed letters.
    - Functionality: Iterates through each character in the word string. Checks if the character has been guessed (iterates through the word array). Draws the letter if guessed, otherwise draws an underscore.

- **drawText(SDL_Renderer renderer, TTF_Font font, const char text, int x, int y, SDL_Color color)**
    - Purpose: Draws text on the screen using a specified font, position, and color.
    - Functionality: Renders the text onto a surface using SDL_ttf functions, then converts the surface to a texture and renders it onto the screen at the specified coordinates.

    - Purpose: Displays the losing screen when the player loses.
    - Functionality: Loads a background image for the losing screen. Displays the correct word that the player failed to guess. Waits for the player to press Enter to restart the game.

Game Over Screens:

- showCongratulationsScreen(SDL_Renderer renderer, TTF_Font font, const char word)
    - Purpose: Displays the congratulations screen when the player wins.
    - Functionality: Loads a background image for the winning screen with a message congratulating the player. Waits for the player to press Enter to restart the game.

- showLosingScreen(SDL_Renderer renderer, TTF_Font font, const char word)
  - Purpose: Displays the losing screen when the player loses.
  - Functionality: Loads a background image for the losing screen with the correct word. Waits for the player to press Enter to restart the game.

3. Details Concerning Libraries Utilized

SDL (Simple DirectMedia Layer):

- Functionality: Provides low-level access to audio, keyboard, mouse, and graphics hardware.
- Usage: Used for window management (SDL_CreateWindow, SDL_DestroyWindow), rendering graphics (SDL_CreateRenderer, SDL_RenderCopy), and handling events (SDL_PollEvent).\

SDL_ttf (SDL TrueType Fonts):

- Functionality: Adds support for rendering TrueType fonts in SDL applications.
- Usage: Used to render text on the screen (TTF_OpenFont, TTF_RenderText_Solid, TTF_CloseFont).

SDL_image:

- Functionality: Provides functions for loading and handling image files (PNG in this case) in SDL applications.
- Usage: Used to load thematic background images and other graphical assets (IMG_Load, SDL_CreateTextureFromSurface).

4. Additional Information

- Gameplay Mechanics:

Traditional Hangman rules are followed where with each single letter input for the word that is to be guessed, the game will either fill in the word until it’s correct or the hangman figure is drawn until the tries are finished.

- Resource Handling:

External resources such as PNG images for theme backgrounds and a TrueType font file for text rendering are utilized. These files are necessary to be accessible and correctly referenced to ensure the game is functionable.

- Platform Considerations:

SDL-based applications are highly portable across different platforms (Windows, Linux, macOS). The game was programmed to ensure that it is functioning and is compatible with Windows through testing and debugging.

- Tools Used

Acquired help from LLM model softwares for debugging and problems concerning SDL.