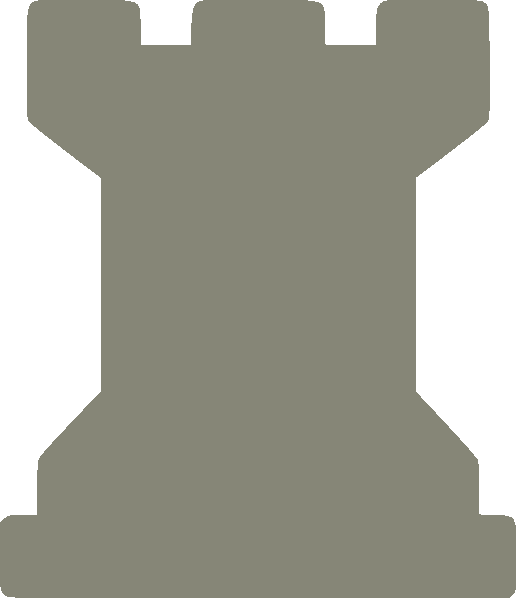
****

**Word Tower**

by Harrison Lee, Simon Tiu, Carter Chung

E177, Prof. Frenklach

May 10th, 2013

# *Abstract*

*Word Tower is an educational game that features different game modes, built on object-oriented programming in the MATLAB environment. The program successfully provides an intuitive user interface for a fun learning experience.*

Table of Contents

*Abstract* 1

Introduction 2

Game Description 2

I. Purpose 2

II. Normal Mode 2

III. Build Mode 3

IV. MatL Mode 3

V. Song Mode 3

VI. Player 3

Technical Description 5

I. UML Diagram 5

II. GamePanel Class 5

III. Player Class 5

IV. WordGame classes 6

Manual of Operation 7

Summary/Conclusion 9

Contributions 9

# Introduction

Word Tower is an educational game aimed at improving typing ability. The goal of our project is *to design a game that allows users to develop accurate typing ability in a fun way*. Rather than force users to go through meaningless keyboard exercises or copy dull paragraphs, Word Tower utilizes a pragmatic word bank and incentivizes fast, accurate timing. Once players become more adept at typing, they have the opportunity to test out their skills in the more advanced *Matlab* and *Song* modes. Players may also challenge their friends to see who can get the highest score. Through playing Word Tower, one can simultaneously have fun and improve typing ability.

# Game Description

## I. Purpose

The primary purpose of this program is to increase the player’s typing ability. The program features four different game modes: Normal, Build, MatL, and Song modes.

## II. Normal Mode

In Normal mode, vocabulary words will pop up under “Current Word”, and you must match the given word by pressing the submit button or hitting the “enter/return” key. If the submission is incorrect or you run out of time, a word will be added to the vertically ascending Word Tower on the leftmost panel. After submission, the timer will reset to full and a new word will appear. When the Word Tower reaches 5 words, the game ends. Thus, in this mode, the player is trying to prevent the Word Tower from building.

The scoring system in Normal is straightforward. Each time the player successfully completes a word, he or she scores one point for every letter. A small bonus is also applied for completely the word quickly. In other words, as the time left to complete the word decreases, so does the time bonus decrease. As the game continues, the difficult of the displayed words will increase.

**Word Tower Facts!**

The Word Tower is **bad** in **Normal Mode**, but **good** in **Build Mode**.

## III. Build Mode

Build mode is almost exactly like Normal mode in regard to game play. The major difference between Build and Normal is that time is tracked on a continual, cumulative basis. In Build mode, if you submit the incorrect word, the time bar will continue to decrease and no words will be added to the Word Tower. If you submit the correct word, that word will be added to your Word Tower and the player will receive a small increment of time. Any incorrect submissions will clear your Word Tower. Thus, the construction of the Word Tower is good in build mode! Upon reaching the 6th correct submission consecutively, the player receives a score bonus and the timer resets back to full. The game ends when the time bar runs out.

## IV. MatL Mode

In MatL mode, definitions of built-in functions of MATLAB are displayed, and the player must answer by submitting the correct name of the function described. The purpose of this game is to test the player’s knowledge of common matlab commands. Gameplay is the same as in Normal mode, except upon hitting “Go!”, a menu will pop up that allows the player to choose the category of built-in MATLAB functions.

Scoring in MatL is simple. The player receives five points for every correct submission. Since there are limited definitions in each category, it is not rare that a player will complete the entire word list, thereby achieving the maximum score.

## V. Song Mode

In Song mode, lyrics for the given song are displayed on screen along with the chorus of the given song. The player must type in the song lyrics along with the song (case-sensitive) before the song ends. In this mode, the time bar allows the players to easily visualize how much time is left. The player’s score percentage is based on how many words are submitted correctly. For an additional challenge, players may select “Chipmunk Mode” which plays each song a 1.5x frequency.

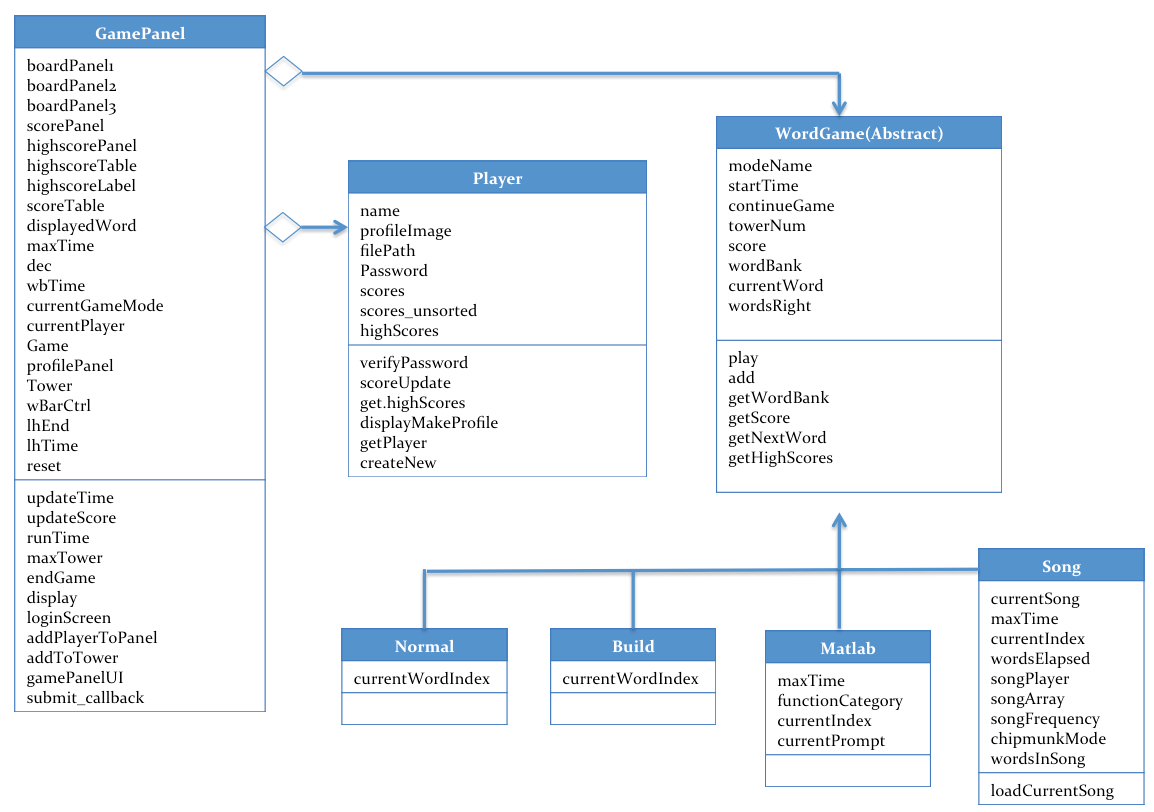
## VI. Player

Word Tower has the ability to track player progress and high scores. In order to play the game, players must create a profile. To do this, players simply specify a login, password, and upload a profile pictures. Once created, this profile will continually track the scores for every player. If a player managed to achieve a personal best, the player score table will automatically update. If a player manages to achieve a top 5 score for a particular game more, the game score table with automatically update with the player’s name a score.

# 

# Technical Description

## I. UML Diagram



## II. GamePanel Class

**GamePanel** is the class that is primarily responsible for the GUI. The consoles through which the user interacts with the program are created via the loginScreen and gamePanelUI methods. GamePanel interacts with the Player class to receive data on personal high scores, overall high scores, and other player specifics (e.g. name, image, etc.). Game Panel also interacts with subclasses of WordGame in order to derive the specific parameters of that particular game mode. Each game has a different scoring system, vocabulary/song list, and a maximum time for the timer. Throughout gameplay, GamePanel calls on methods of the Word Game classes to calculate score and to retrieve the next word to be displayed on the console. GamePanel also utilizes different game logic and mechanics in for each game mode. For example, if interacting with a Normal or MatL game, GamePanel will notify ‘EndGame’ when 5 wrong words are submitted. However, if interacting with a Build or Song game, GamePanel will notify ‘EndGame’ when the timer runs out.

## III. Player Class

**Player** is the class that handles user profiles, including score data for each game mode as well as login information and a profile image. Player information is stored in a Matlab file (.mat) and retrieved when logging into the GamePanel. As the user plays the various game modes, scores are updated and tracked automatically.

## IV. WordGame classes

**WordGame** is an abstract class that manages the various game modes. This class necessitates that each game mode child class stores as properties important parameters of the game, such as the name of the game mode, the score, and the words to be typed in the game. The class also necessitates that certain methods be implemented, such as to load the word bank, begin gameplay, draw from the word bank, calculate the score, and manage high scores. WordGame also has events to broadcast that the game has ended, or that the time bar has expired.

Normal, Build, MatL, and Song classes all inherit from WordGame. While each subclass shares these inherited general properties and methods, implementation of these properties and methods varies across each subclass, and additional methods and properties exist in subclasses according to desired gameplay. In each game mode, word bank data is stored in a .csv file, and loaded and organized using fopen and textscan, so the game can access the data directly by cell array indexing. This data is stored as a property of the class. Words are then drawn from wordbanks by generating a pseudorandom integer (randi) within the size of the word bank to determine the index.

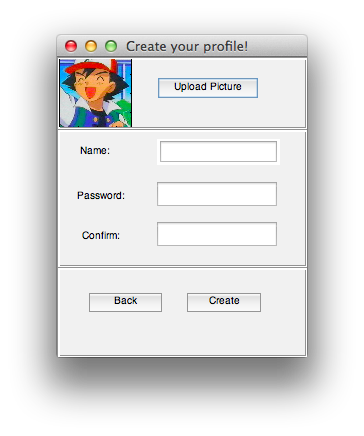
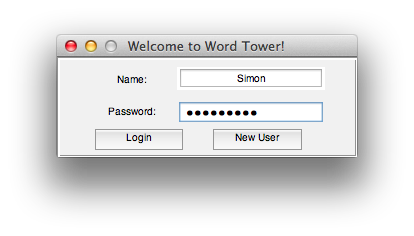
In **Normal** mode, words are stored and indexed according to word length. The index of the current word is also stored, so the word bank entry at that index can be cleared when the word is typed correctly. Each time five words are correctly submitted, the method to get the next word of the Normal object searches for a word that is one character longer. This continues until the word is 15 characters long, and then a word of 15+ characters is returned.

**Build** mode uses the same algorithm as Normal to find the next word. The key differences lie in scoring, and in the GamePanel implementation.

**MatL** class functions similarly to Normal class. The word bank .csv file has function name in the first column and function description in the second column. MatL mode stores additional properties, including the current function description (as the prompt and expected response will be different), and the index of the current prompt, so that after each prompt is used in the game, the name and description are removed from the word bank (the indices in the word bank are set to null) and there are no repetitions. When all words are cleared, the game ends.

In **Song** class, lyrics are again saved as .csv files, with each line of the song in a separate cell. Commas are removed as they are used as delimiters. Music to each song is formatted as .wav, and wavread is used to return the sampled data and the sample rate. The function audioplayer then takes the data and frequency to create an object to play the song when gameplay commences. Chipmunk mode is implemented by increasing the frequency by a factor of 1.5.

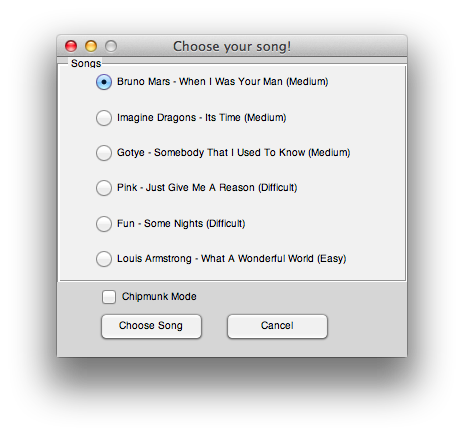
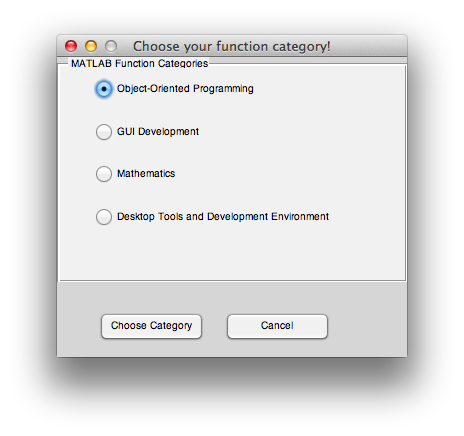
# Manual of Operation



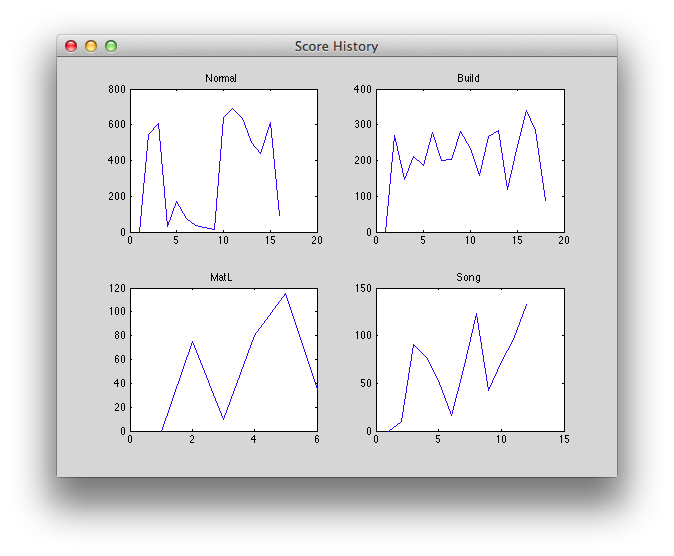
It is crucial that the Matlab path has the root of the project. To run the game, simply type “login” into the command line, and create a “New User”. “Upload Picture” and choose an image from the “images” folder. If you would like to use your own image, first upload it into the “images” folder before attempting to create the new user. Select a name and password, then select “Create”. This will return you to the login screen, where you can enter your newly created username/password and login to the main console.



To begin playing, first choose a mode, then select “Go!”. At any point during the game, press “Reset” to end the game and clear the current game. If the game if finished, select either “Reset” then “Go!” to play a new game, or simply select “Go!”

If you play MatL or Song modes, a figure will pop up allowing you to choose MATLAB function categories or different songs, respectively.



Your personal player profile is located on the right, showing your picture, username, and personal high scores for each game mode. To track your overall progress, select “Score History”, and a graphic displaying your scores over time will be plotted. Below the personal player profile, overall game High Scores are shown in a table for each mode

At the end, you may “Log Out” so that another player may sign in.

# Summary/Conclusion

In completing this project, we successfully created a simple, intuitive, and fun program that may add value to our daily lives. In making our project, we learned how to design class structures from scratch and create interesting MATLAB GUIs. Along the way, we also learned the valuable skill of how to search for and interpret solutions on the Internet. Ultimately, this project allowed us to delve into deeper coding concepts and make an interesting and useful final product.

# Contributions

Each member was primarily responsible for...

Simon Tiu: classes – WordGame, Normal, Build, Player. Contribution - 33%

Carter Chung: classes - MatL and Song. Contribution - 33%

Harrison Lee: classes - GamePanel. Contribution - 33%

Michael Frenklach: classes - E177. Contribution - 100%

While these are our major contributions, we worked together to help each other with refining the GUI, debug, explain concepts, and to restructure our project. Teamwork pays off!