|  |  |  |
| --- | --- | --- |
| Group Member | Student URN | Percentage of Mark |
| Patrick Dobinson | 6420583 | 25 |
| Ryan Croft | 6421218 | 25 |
| Yram Duadze | 6428487 | 25 |
| Alex Drew | 6417705 | 25 |

**MATLAB Mavericks Report**

Hangman

HANGMAN - Based on the retro word game which features the hangman animation. The game has three rounds namely sports, capital cities and countries.

The aim of the game is to allow a player to guess a word one letter at a time. It indicates the number of letters in the word as well as which letters the player has currently guessed properly.

Each player has 10 lives at the beginning of the game and after correctly guessing a word in a round, the game automatically proceeds to the next round.

The game posts a generic tweet to (Hangman.MATLAB @ECO2017) at the end of each game with the contents of the message depending on the player winning or losing.

\*PLEASE NOTE: However, twitter restricts the number of tweets sent out within a given period from external applications. As such, if the game is played multiple times within a short period, only the first two tweets are sent and error messages appear for the twit or twit1 functions after successive games have ended. This however does not affect the game itself and it should still run normally.

**INSTRUCTIONS ON HOW TO RUN THE GAME.**

* Download all files in the repository into a folder ideally titled "Hangman" and add the folder to the path on matlab either by using the addpath command or copying and pasting the folder into the main MATLAB folder ensure all functions correspond to the names they have been saved with. i.e The ‘wordgame’ function saved as 'wordgame.m' etc
* To initialise the game, on the command window type ‘wordgame’. The game should now display a welcome message and should prompt you to type a letter.
* After guessing a letter incorrectly, the hangman animation is displayed in a new window. The animation is displayed gradually with every life lost until the full image is shown which indicates the game is over and you have lost.

\*PLEASE NOTE: The animation takes about a minute to initialise so please be patient. This depends on the processing capabilities of the computer the game is being played on.

* Follow the instructions provided and enjoy

The number of lives you have is displayed and would reduce if the player guesses the incorrect letter. The player is also notified if a letter is used more than once in a round and prompted to choose another. After correctly guessing the word, the game proceeds to the next stage.

If you lose the game by running out of lives, the game notifies you, posts a tweet and terminates. If you guess all words correctly through all three rounds then a congratulatory message is shown indicating the number of lives you have left at the end of the game as well as a tweet being sent out.

**Motivations**

When our group first got together, one of the first things we agreed on was that we wanted to create a game. We quickly decided that the game would be more along the lines of a retro game such as battleships or mine sweeper. After doing a bit of research we set our goals on Hangman. We chose this game as we saw it challenging to code, but simple enough respective to our MATLAB abilities. In addition, as technology continues to revolutionise traditional games, we wanted to bring Hangman into the 21st century, with no need for pen and paper! This 21st century vision is furthered by our social media aspect. With twitter, we can create a hangman community, facilitating interaction between players.

**Improvements for the Future**

1. When we added the ‘*twit*’ function (M,2003), Twitter did not allow repetitive tweeting from the same account. i.e it came up with an error. Hence the twit function does not work if the game is played more than once in a short-period of time.
2. Furthermore, we would like to tag the player’s twitter username in the tweet.
3. Improve the GUI for the animation
4. Add more levels for different categories to increase the length and difficulty.

**Challenges**

1. Initial challenges we faced included the creation of the animation as working with a gui was something completely new to us. On several occasions, we tried creating a code for the gui based on similar games we had seen on the matlab file exchange. However, these did not achieve the desired results. We subsequently discovered the ‘imshow’ command on matlab documentation and upon editing our code with the command in mind discovered it was the perfect tool in realising our goal.
2. We initially tried to create the entire game in one script/function discovered it was cumbersome. We kept getting several error messages so decided to split the code by creating hangman.m. We subsequently created a function titled ‘*wordgame*’ which called the hangman function, hence making the code more simplistic.
3. When we ran the game initally, and had someone play it, they inputted a capital letter which led to an error. This was completely unexpected and could be described as an oversight. To overcome this, we used the lowercase command ‘lower’ to convert all characters without the need to edit the entire list of words.
4. Originally our tweet just stated that someone had played the game as we kept receiving error messages in our attempt to tailor the tweets to either a player winning or losing. Eventually, we decided to preload a message within the twit function. However, the message was such that it could only be either winning or losing. We had difficulties adding multiple messages and to overcome this we cloned the twit function rename it as twit1 which is specific to losing hence tweets a message if you lose and vice versa for twit.
5. An Initial oversight was the number of words in the list and only noticed when the the same word came up every consistently. We expanded word bank to account for this.

**Results**

It works! We have a game that is simple and straightforward, no knowledge of MATLAB needed to use the game, making it more accessible to everyone. It clearly shows you how many letters the word is, if your letter was wrong or right (letter appears instead of the asterisk) and how many lives you have left. If you win, you get a congratulatory message and if not you are notified as well and can simply enter the command ‘wordgame’ to play again, with new lives etc. In addition, a tweet is successfully sent on the twitter account when a player plays the game.

**Communication and Distribution of Tasks**

WhatsApp

We organised meetings to discuss the code and any ideas we had for the project, as this platform was quick and convenient to use.

GitHub

This platform was not used as much for communicating as we decided to create the code as a group and in person where we felt we could workshop our ideas better.

In Person

When we came together, we tried to allocate tasks to match each person’s set of skills so no time is wasted and no work is overlapped. However, we all ended up playing a role in writing sections of the code or given suggestions to that effect with the goal of creating a project which appeared to be coherent.

**Bibliography and References**

Inspiration for portions of the code were taken from sources listed below as well as the Matlab community and file exchange platforms which were equally pivotal in our ability to achieve the results we desired.

M, M. (2003). *Hangman*. Mathworks.

Ruthramoorthy, N. (2016). *Twit*. Mathworks.

Mi SL and Walter Roberson – Matlab community – Accessed useful information with regards to the use of ‘IMSHOW’