Welcome to Hatman in R!

This tutorial will guide you through how to play this "Hatman" game. "Hatman" is very similar to the popular game "Hangman", except in this version, no one gets hurt :)

This game relies on R and performs best when R is run through RStudio. R can be downloaded here: https://www.r-project.org/ RStudio can be downloaded here: https://www.rstudio.com/products/rstudio/download/

You may use this code for any projects you like but please contact me (chris.j.blackford@gmail.com or github.com/Christopher-Blackford) if you do so I know it has been useful!

Getting started:

To open the game, both the "**Hatman_1of2**" (R project file) and "**Hatman_2of2**" (R file) files need to be loaded. There are multiple ways to do this – I suggest first opening (i.e., clicking on) the **Hatman_1of2** file and then opening (i.e., clicking on) the **Hatman_2of2** file.

Note (for more advanced R users): The R project file is only used to make sure your working directory is properly set. For more advanced R users, you can change your working directory with the setwd() function or, in RStudio, you can set the working directory to always be wherever the R code file is located under Tools>Global options. If you do either of these steps, you don't need to use the Hatman_1of2 file at all.

Gameplay:

Setup: In Hatman, you are trying to guess what letters make up a mystery word by only knowing the number of letters in that word. If you make too many wrong guesses a Hatman appears, indicating you have lost the game. R provides the mystery word and you have some flexibility in deciding what types of words R will choose (explained further below).

Hint: The game board will appear in the plotting window, so make this window fairly large so you can see the game board clearly.

1. To start, you will need to load (and install if they aren't already) a couple of R packages. The "tm", "pdftools", and "stringr" packages are required for text parsing, and the "rgeos" and "sp" packages are required to visualize the Hatman board. To do this, run all the code up to the "rm(list=ls())" line.

You can run the code by hitting the "Run" button at the top of the Source window or you can hit Ctrl+Enter on Windows or Command+Enter on Mac. You should be able to see the code executing in the console window and the game board appear in the plotting window as you are running the code. Now the game is setup and ready to go.

2. Run the "rm(list=ls())" line. This is the line that will be used to clear R memory in between Hatman games.

3. The "Word_list_to_use" line defines what word list that computer will use to generate words for Hatman. By default the line reads:

Word_list_to_use <- "DEFAULT"

which means the computer will choose a word from a large word list based on an online dictionary. But you don't have to use this default word list! You can "upload" a pdf to the program and then R will choose from the words in that pdf. To do this, you need to place the pdf in the folder directory "./Hatman/word_files/pdf", then write the name of that pdf EXACTLY into this line (don't include the .pdf extension into the name).

For example, if you put a pdf titled "Poetry" into the pdf folder, the line needs to be changed to:

Word_list_to_use <- "Poetry"

Note: If you have a specific word list in mind that you would prefer to play with (instead of parsing from a pdf), you can do this to! Create a word list in a text file and place it in the "./word_files/word_lists" directory. Words must be in all caps and each occupy their own line.

The title of the text document needs to be "X_WordList" where "X" can be any character string and is also what you define as "Word_list_to_use" in the "Hatman2of2.R" code (i.e. Word_list_to_use <- "X"). You can look at the "DEFAULT_WordList" file to see what I mean.

4. Run the "source("./code/Selecting_Word.R")" line.

If you choose to upload your own pdf, R will begin parsing your pdf document into a word list. Depending on the size of the pdf, this make take several minutes, be patient! R will tell you how long it took to create the word list at the end of the process. At the end of this process, R saves the word list so if you play multiple times with the same pdf, you won't have to spend time on this again.

If you choose to use the default word list, R will immediately load in that word list instead.

5. “Run the source("./code/Hatman_Game_Board.R") and source("./code/Guessing_Function.R") lines. You should see the game board now in the plotting window. You can drag the window to change the size if it’s too small for your liking.

6. To guess a letter, type in the source window or console:

guess(“your_letter”)

(where your_letter is “a”, “b”, “c”, etc.). You just rerun this line for different letters until you win or lose.

At the end of the game, you can rerun the code from the line: “rm(list=ls())” which will clear your memory and allow you to play again.

Note: I hid a “reveal” function in the Guessing_Function.R code that allows the user to reveal the word if you end up losing but want to still see the word. I wanted to reward people who either read documentation or go into the sub-code.

To use the reveal function, simply run the code:

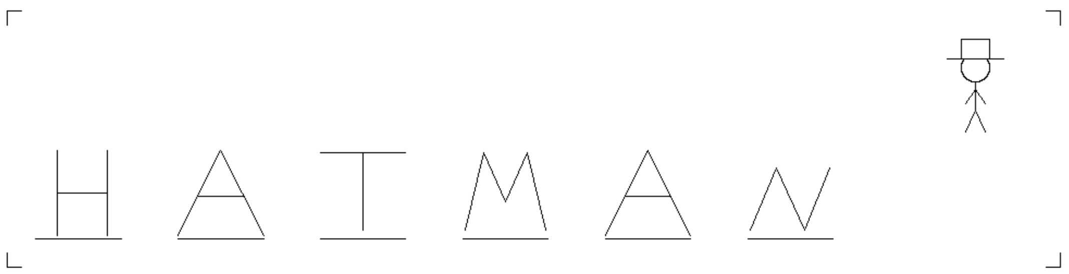
reveal()

Some caveats

1. This game only works for English words since it doesn’t handle non-English characters or symbols well (e.g. ê, ö). It’d be a really neat expansion of the game if someone was able to have it recognize non-English characters.

2. The pdf parsing isn’t perfect. In testing, I was able to get around 96% accuracy (i.e. the code identified a “word” that wasn’t actually a real word 4% of the time) with the dictionary word list but there will be some cases where non-words will slip through the cracks.

3. Words in the margin of the user supplied pdf document are likely to be removed entirely.



Have fun!