1.4 Importing/exporting data

Objectives:

- Understand the concept of a working directory
- Be able to read in data
- Be able to export data

Contents:

- Working directory
- Reading data
- Reading/writing data

Working directory

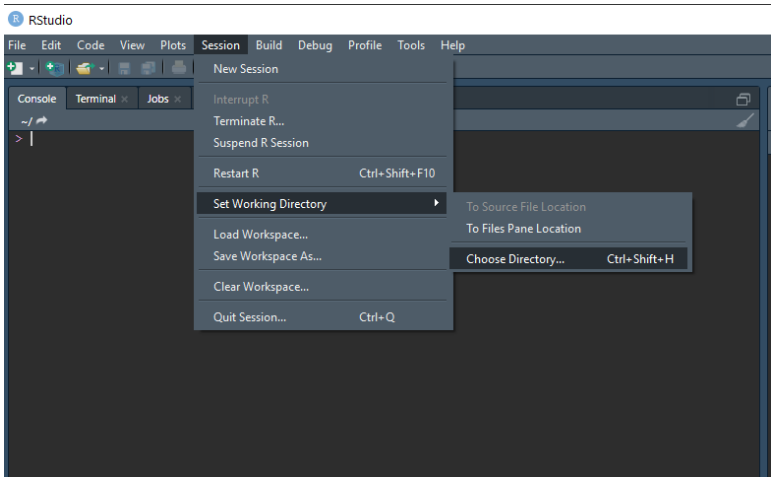
A working directory is the folder R will look in for files. We strongly recommend that you set a specific working directory before you begin any project. It will make managing your output much easier in the future.

When you first set up RStudio you can set your default working directory as follows:

> 💡 *Tools –> Global options –> click on "Browse" to select the default working directory you want.*

Each time you close R/RStudio, you will be asked whether you want to save the data from your R session. If you decide to save, the data/working directory will be available in future R sessions.

An easy way to do this in RStudio is shown below:



If you know where you want to create the directory, you can write out the path to the `setwd()` function

```
setwd("/path/to/my/directory")
```

And the `getwd()` function will return the current working directory

```
> getwd()
[1] "C:/Users/Ruairi"
```


Exercise 1:
Set your current working directory to the 1.4 Importing data folder

Note: as you become more proficient R users, it is advised that you move towards R projects, rather than setting working directories. For more information on projects, see this excellent tutorial.

Reading data

A simple file type you might need to read in is a .txt. For this we can use the `read.table()` function. Let's call `help` on `read.table()`

Of this long list of arguments to `read.table()` we mainly need to worry about `file`, `sep`, `header` and `stringsAsFactors`. We will detail these for `read.table()` but many of the arguments are similar for other reading functions.

`file` allows us to tell R where the file is located. If this file is in the working directory then we can just type the file name. If not you will need to specify the full path.

`sep` argument tells R how the columns are separated in the file (again, for a comma−separated file (CSV), use sep = ","}, for a tab−delimited file, use sep = "\t".

`header` argument is a logical value (`TRUE` or `FALSE`) telling R whether or not the first row in the data is the name of the data columns.

`stringsAsFactors` is a logical value indicating whether or not to convert strings to factors.

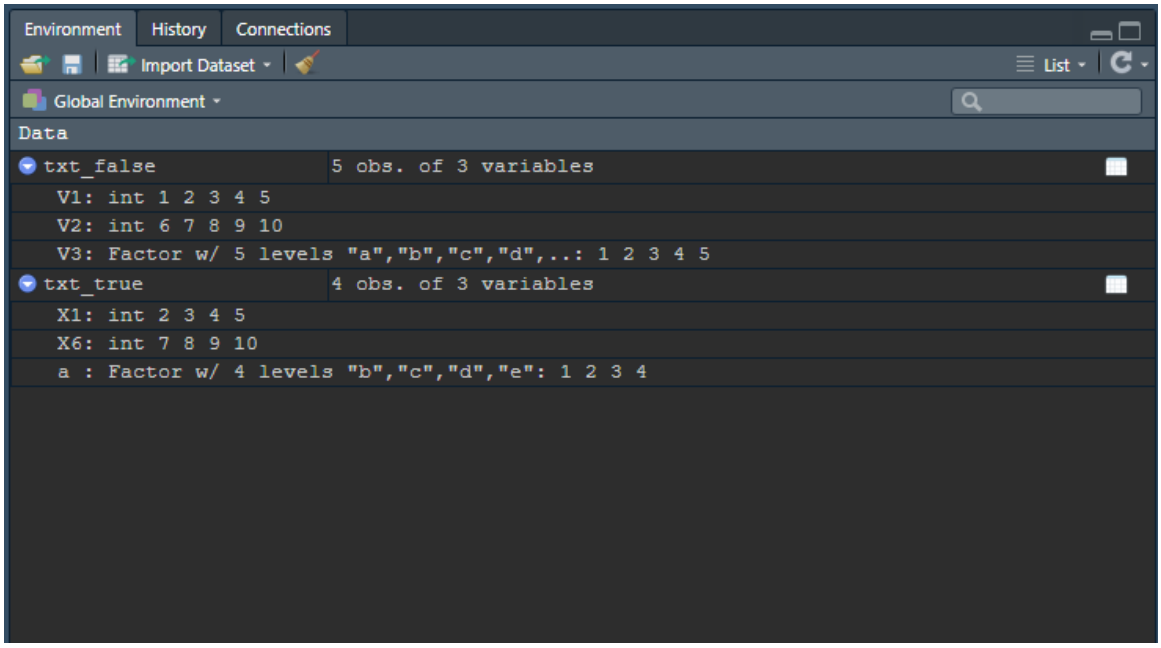
Exercise 2:

Use the help function to read in the text file in folder

Let's explain the functions a bit

```
txt_false=read.table('text_file.txt', header = FALSE)
txt_true=read.table('text_file.txt', header =TRUE)
```

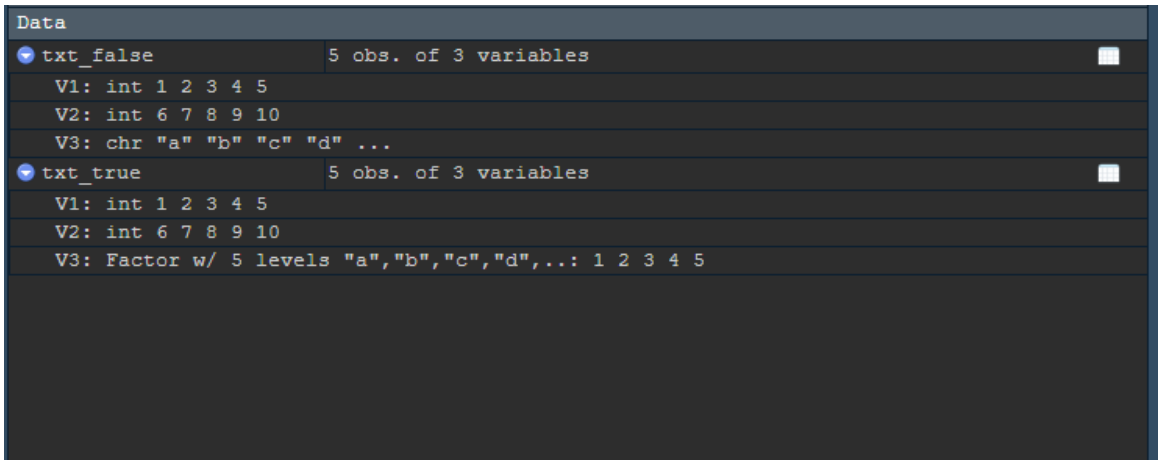
Be very careful with this function argument, it drops a row to treat it as a header.



Let's test out `stringsAsFactors`

```
txt_false=read.table('text_file.txt', stringsAsFactors = FALSE)
txt_true=read.table('text_file.txt', stringsAsFactors = TRUE)
```

Note the variable types, V3 is a factor in the second dataframe.



Reading data and writing data

We can read in excel files with two very useful functions, `read.excel()` and `read.csv()` and then write out files with the `write.csv()`
Note: R can not read in a file when it is open in another programme.

```
library(readxl)
# read in each sheet as a seperate dataset
happiness_2015=read_excel("happiness_report.xlsx", sheet = "2015")
...
...
happiness_2019=read_excel("happiness_report.xlsx", sheet = "2019")

#write out csv
write.csv(happiness_2015, 'happy_2015.csv')
...
...
write.csv(happiness_2019, 'happy_2019.csv')

# read in csv
h15 = read.csv('happy_2015.csv')
```

Summary

You have learnt:

- How to use a working directory
- How to read and write data from different files

```
# solution 1
setwd('/path/1.4 Importing data')
```

```
# solution 2
mydata <- read.table(file = 'text_file.txt',
                     sep = '\t',
                     header = FALSE,
                     stringsAsFactors = FALSE)
```

Additional exercises

Note: Before you begin, you should know there are more efficient ways to finish this challenge. They involve writing functions and save copy and pasting. This kind of solution is a little bit more complex, and we will not work on this in this workshop.

1. Read in each of the happiness files
2. Select the columns representing rank and country name from each
3. Add a label column to each file detailing the year
4. Join the files together by rows (hint: check the column names!)
5. Write to a csv file called 'happiness_solved.csv'

Materials used in this course can be cloned directly by clicking here. Alternatively, you can view the repository online:

> *https://github.com/RJODRISCOLL/Introduction-to-R*

For any help and advice, I can be contacted at:

> *R.ODriscoll@leeds.ac.uk*