INTRODUCTION TO LITERATE PROGRAMMING - R MARKDOWN

CLASS #3 | GEOG 215

Introduction to Spatial Data Science

Spring 2020

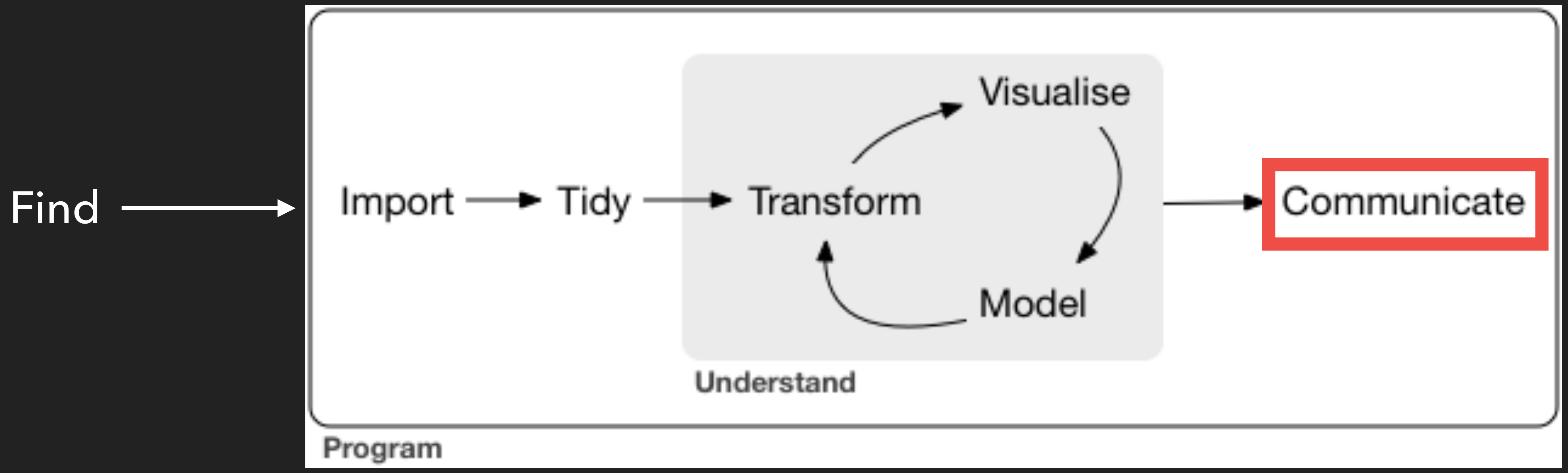
TODAY'S CLASS

- Communicating Data Science
  - Data science is for Humans, Not computers
  - Intro to Literate Programing and (R) Markdown
- DIY RMarkdown
- Start Lab 1

REMINDERS

- Register and complete the Poll everywhere survey
  - https://pollev.com/goelvarun553/
- Sign up for class discussion on Piazza
  - http://piazza.com/unc/spring2020/geog215/home

DATA SCIENCE PROCESS



Source: R for Data Science

Getting the analysis right is only one part of the chain

WHO IS YOUR AUDIENCE

- People, not Computers!
- Future You!
- Your Instructor, TA, Boss
- Smart people who do not know R
- Everyone else: The internet, future employers, others ?

#DEAR FUTURE SELF,
#
# YOU'RE LOOKING AT THIS FILE BECAUSE
# THE PARSE FUNCTION FINALLY BROKE.
#
# IT'S NOT FIXABLE. YOU HAVE TO REWRITE IT.
# SINCERELY, PAST SELF
DEAR PAST SELF, IT'S KINDA CREEPY HOW YOU DO THAT.
#ALSO, IT'S PROBABLY AT LEAST
# 2013. DID YOU EVER TAKE
#THAT TRIP TO ICELAND?
STOP JUDGING ME!

MOTIVATING REAL LIFE EXAMPLE

- As a budding researcher and a data scientist, You are invited for an interview for a Job at the United Nations Population Division
    - Your Task: Analyze population trends over time and show whether there is any relationship between health and Wealth, and send the results to the interviewers.

MOTIVATING EXAMPLE

YOUR INTERVIEWERS

- Smart population experts
- Old School - believe in communicating through Microsoft word
- Never took an R class (believe in excel as Truth)
- Interested in knowing your analytical thinking process - Not just the results, but the process through which you came through your results

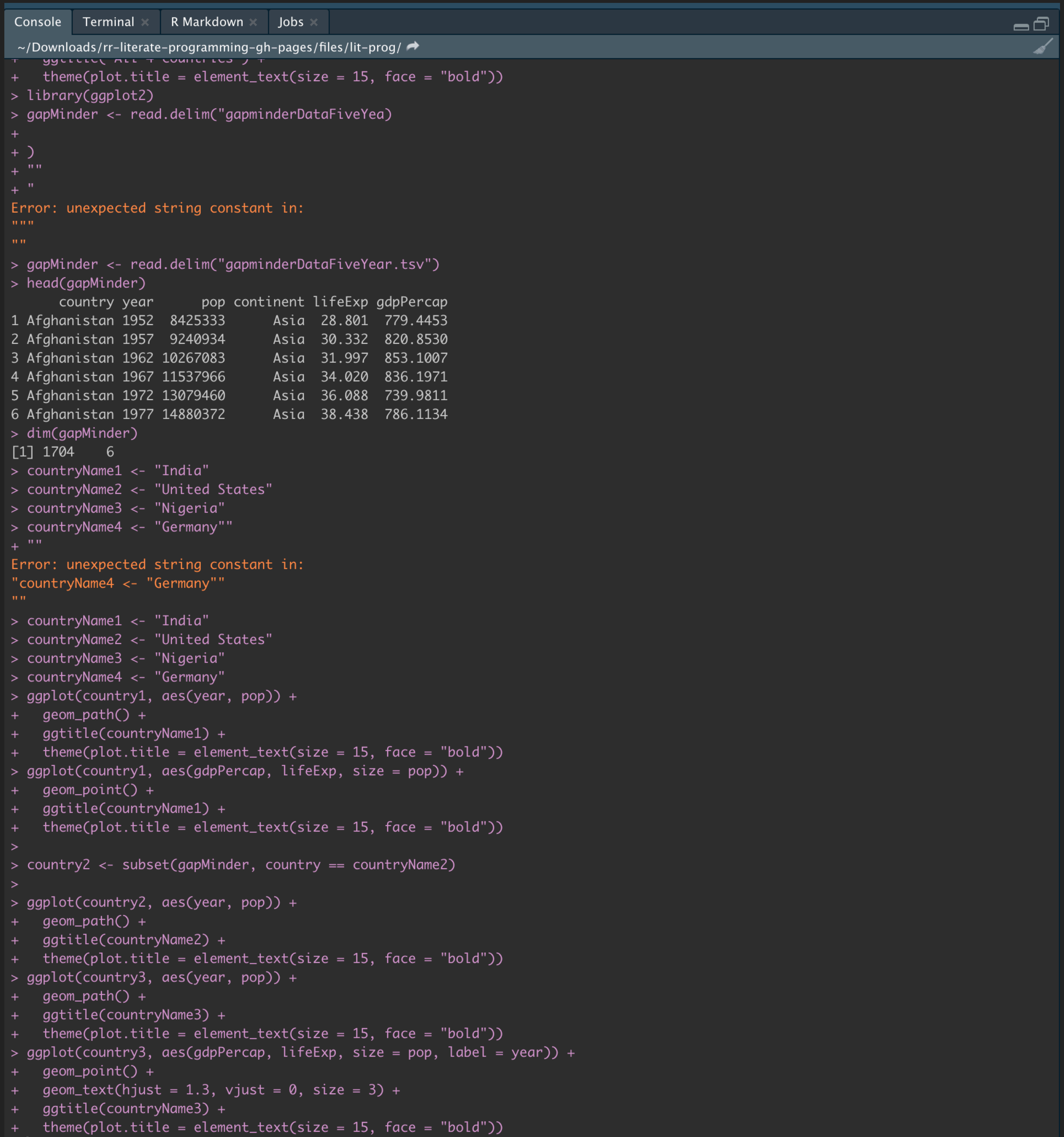
YOU

- R expert
- Taken a few sociology and statistics classes
- Tech Savvy - You have a personal website, and believe in making knowledge easily accessible

GROUP BRAIN STORM (5 MINS)

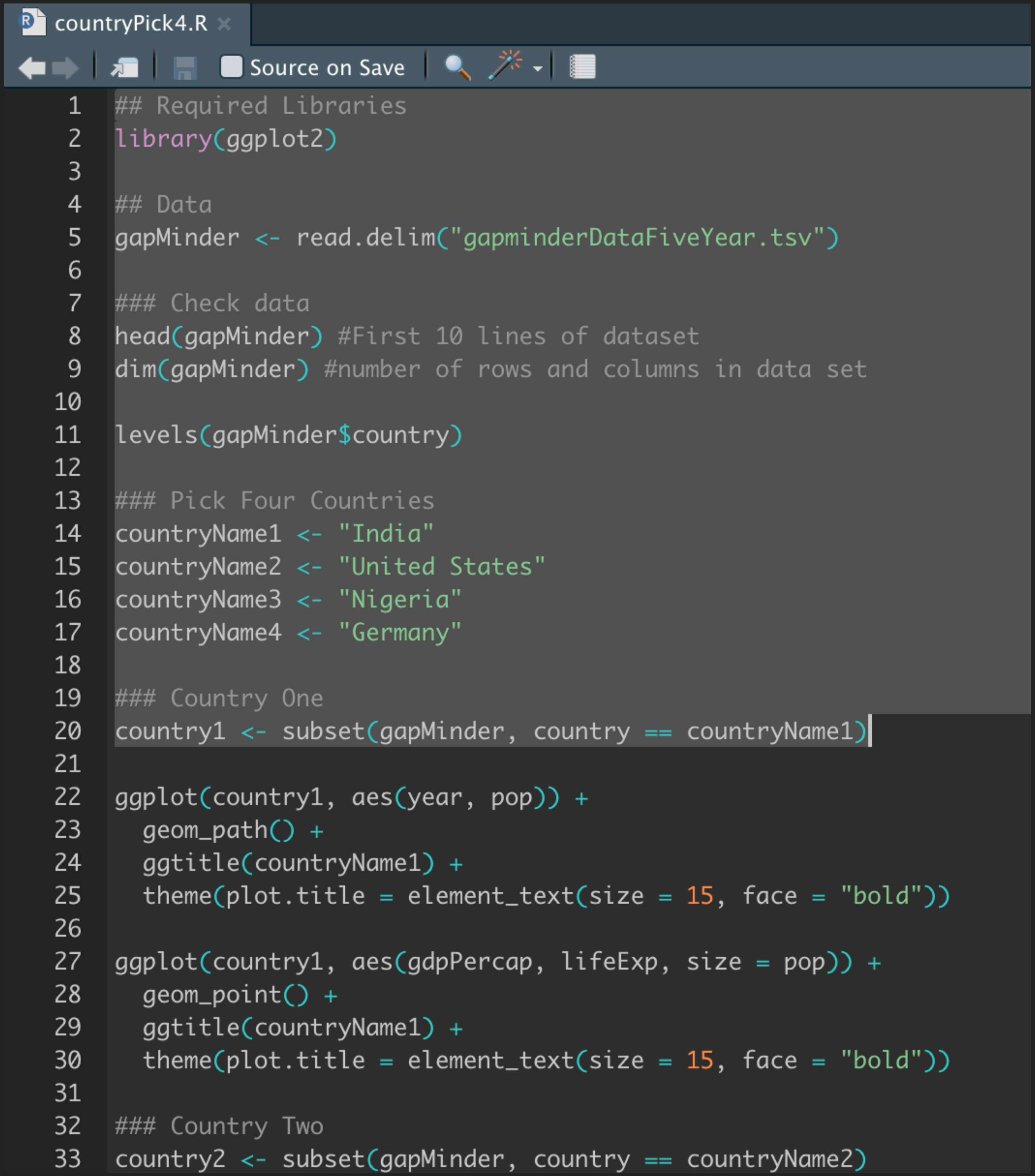
- What all information do you need to analyze the data?
  - Hint: go to https://www.gapminder.org/data/
  - What variables? Mention 1 health and 1 wealth variable
- How will you communicate your results?
  - What kind of visuals or tables would you have?
  - How will you transmit your results to them?

SCENARIO 1


Console
Terminal
R Markdown
Jobs
~/Downloads/rr-literate-programming-gh-pages/files/lit-prog/

```
+   ggtitle("All 4 countries") +
+   theme(plot.title = element_text(size = 15, face = "bold"))
> library(ggplot2)
> gapMinder <- read.delim("gapminderDataFiveYea)
+ 
+ )
+ ""
+ "
Error: unexpected string constant in:
""" 
""
> gapMinder <- read.delim("gapminderDataFiveYear.tsv")
> head(gapMinder)
      country year      pop continent lifeExp gdpPercap
1 Afghanistan 1952  8425333      Asia  28.801  779.4453
2 Afghanistan 1957  9240934      Asia  30.332  820.8530
3 Afghanistan 1962 10267083      Asia  31.997  853.1007
4 Afghanistan 1967 11537966      Asia  34.020  836.1971
5 Afghanistan 1972 13079460      Asia  36.088  739.9811
6 Afghanistan 1977 14880372      Asia  38.438  786.1134
> dim(gapMinder)
[1] 1704    6
> countryName1 <- "India"
> countryName2 <- "United States"
> countryName3 <- "Nigeria"
> countryName4 <- "Germany""
+ ""
Error: unexpected string constant in:
"countryName4 <- "Germany""
""
> countryName1 <- "India"
> countryName2 <- "United States"
> countryName3 <- "Nigeria"
> countryName4 <- "Germany"
> ggplot(country1, aes(year, pop)) +
+   geom_path() +
+   ggtitle(countryName1) +
+   theme(plot.title = element_text(size = 15, face = "bold"))
> ggplot(country1, aes(gdpPercap, lifeExp, size = pop)) +
+   geom_point() +
+   ggtitle(countryName1) +
+   theme(plot.title = element_text(size = 15, face = "bold"))
> 
> country2 <- subset(gapMinder, country == countryName2)
> 
> ggplot(country2, aes(year, pop)) +
+   geom_path() +
+   ggtitle(countryName2) +
+   theme(plot.title = element_text(size = 15, face = "bold"))
> ggplot(country3, aes(year, pop)) +
+   geom_path() +
+   ggtitle(countryName3) +
+   theme(plot.title = element_text(size = 15, face = "bold"))
> ggplot(country3, aes(gdpPercap, lifeExp, size = pop, label = year)) +
+   geom_point() +
+   geom_text(hjust = 1.3, vjust = 0, size = 3) +
+   ggtitle(countryName3) +
+   theme(plot.title = element_text(size = 15, face = "bold"))
```

SCENARIO 2:



```
## Required Libraries
library(ggplot2)

## Data
gapMinder <- read.delim("gapminderDataFiveYear.tsv")

### Check data
head(gapMinder) #First 10 lines of dataset
dim(gapMinder) #number of rows and columns in data set

levels(gapMinder$country)

### Pick Four Countries
countryName1 <- "India"
countryName2 <- "United States"
countryName3 <- "Nigeria"
countryName4 <- "Germany"

### Country One
country1 <- subset(gapMinder, country == countryName1)

ggplot(country1, aes(year, pop)) +
  geom_path() +
  ggtitle(countryName1) +
  theme(plot.title = element_text(size = 15, face = "bold"))

ggplot(country1, aes(gdpPercap, lifeExp, size = pop)) +
  geom_point() +
  ggtitle(countryName1) +
  theme(plot.title = element_text(size = 15, face = "bold"))

### Country Two
country2 <- subset(gapMinder, country == countryName2)
```

SCENARIO 3:

- A document that can contain both *Prose* and *Code in a human readable form*

DEMONSTRATION !

LITERATE PROGRAMMING

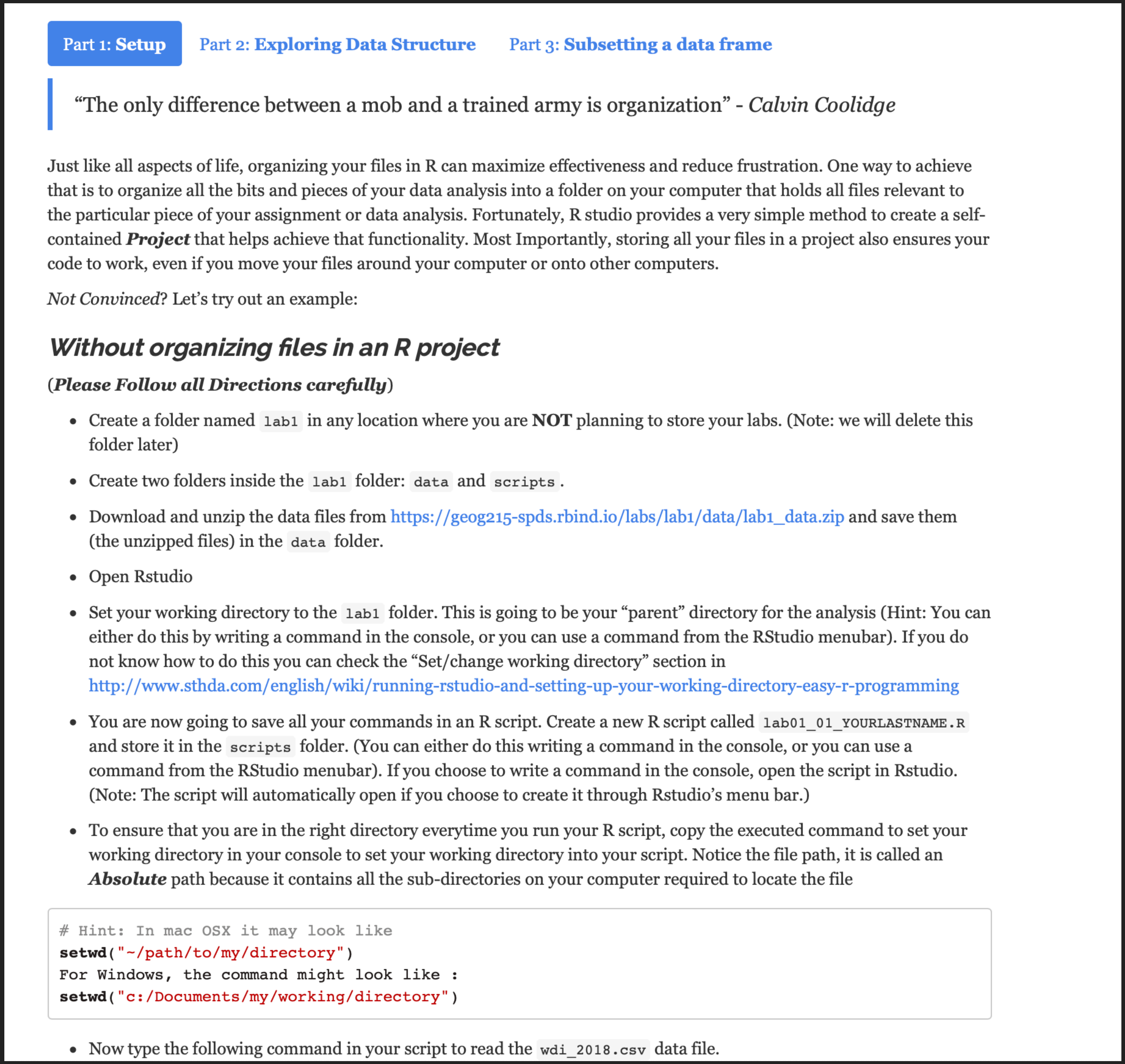
- “Creating computer programs as works of literature” - Donald Knuth
- Tightly integrated prose and computer code
    - Organize your work concisely
    - make work more pleasant for yourself? (less tedious, less manual, less)
    - reduce friction for collaboration
    - reduce friction for communication
    - make your work navigable, interpretable, and repeatable by others

(R)MARKDOWN

- Mix ideas, code and create documents seamlessly
- Easy to learn and use
- Focus is on **content,** not coding and debugging
- Easy to publish and read on web
    - Remember that cool friend with a cool website??
    - And many other formats (word, pdf)
- Enables Reproducibility ! –> **Week 6**

EXAMPLE

YOUR LAB



Part 1: **Setup** Part 2: **Exploring Data Structure** Part 3: **Subsetting a data frame**

> "The only difference between a mob and a trained army is organization" - *Calvin Coolidge*

Just like all aspects of life, organizing your files in R can maximize effectiveness and reduce frustration. One way to achieve that is to organize all the bits and pieces of your data analysis into a folder on your computer that holds all files relevant to the particular piece of your assignment or data analysis. Fortunately, R studio provides a very simple method to create a self-contained ***Project*** that helps achieve that functionality. Most Importantly, storing all your files in a project also ensures your code to work, even if you move your files around your computer or onto other computers.

Not Convinced? Let's try out an example:

Without organizing files in an R project

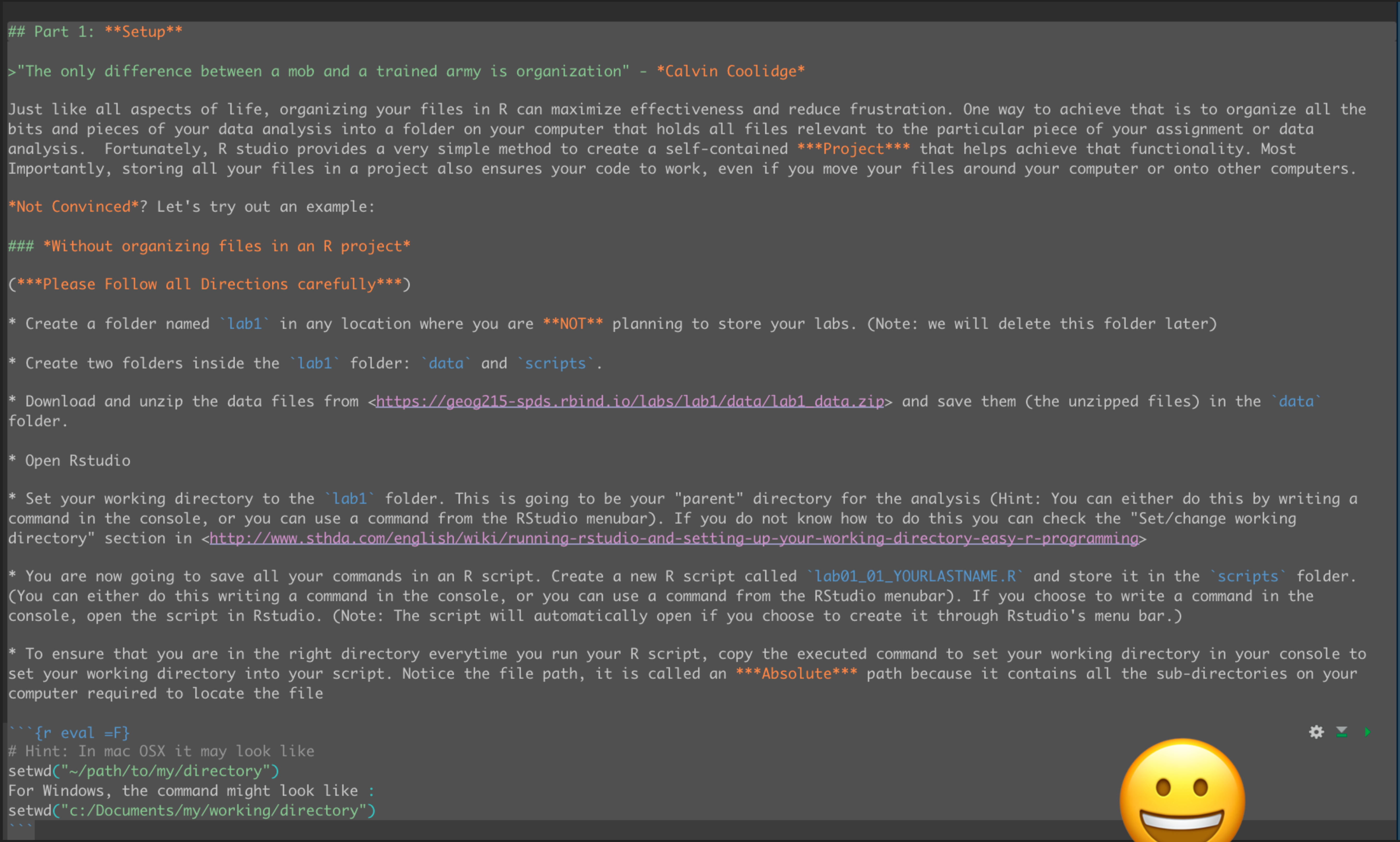
(***Please Follow all Directions carefully***)

- Create a folder named `lab1` in any location where you are **NOT** planning to store your labs. (Note: we will delete this folder later)
- Create two folders inside the `lab1` folder: `data` and `scripts`.
- Download and unzip the data files from https://geog215-spds.rbind.io/labs/lab1/data/lab1_data.zip and save them (the unzipped files) in the `data` folder.
- Open Rstudio
- Set your working directory to the `lab1` folder. This is going to be your "parent" directory for the analysis (Hint: You can either do this by writing a command in the console, or you can use a command from the RStudio menubar). If you do not know how to do this you can check the "Set/change working directory" section in http://www.sthda.com/english/wiki/running-rstudio-and-setting-up-your-working-directory-easy-r-programming
- You are now going to save all your commands in an R script. Create a new R script called `lab01_01_YOURLASTNAME.R` and store it in the `scripts` folder. (You can either do this writing a command in the console, or you can use a command from the RStudio menubar). If you choose to write a command in the console, open the script in Rstudio. (Note: The script will automatically open if you choose to create it through Rstudio's menu bar.)
- To ensure that you are in the right directory everytime you run your R script, copy the executed command to set your working directory in your console to set your working directory into your script. Notice the file path, it is called an ***Absolute*** path because it contains all the sub-directories on your computer required to locate the file

```
# Hint: In mac OSX it may look like
setwd("~/path/to/my/directory")
For Windows, the command might look like :
setwd("c:/Documents/my/working/directory")
```

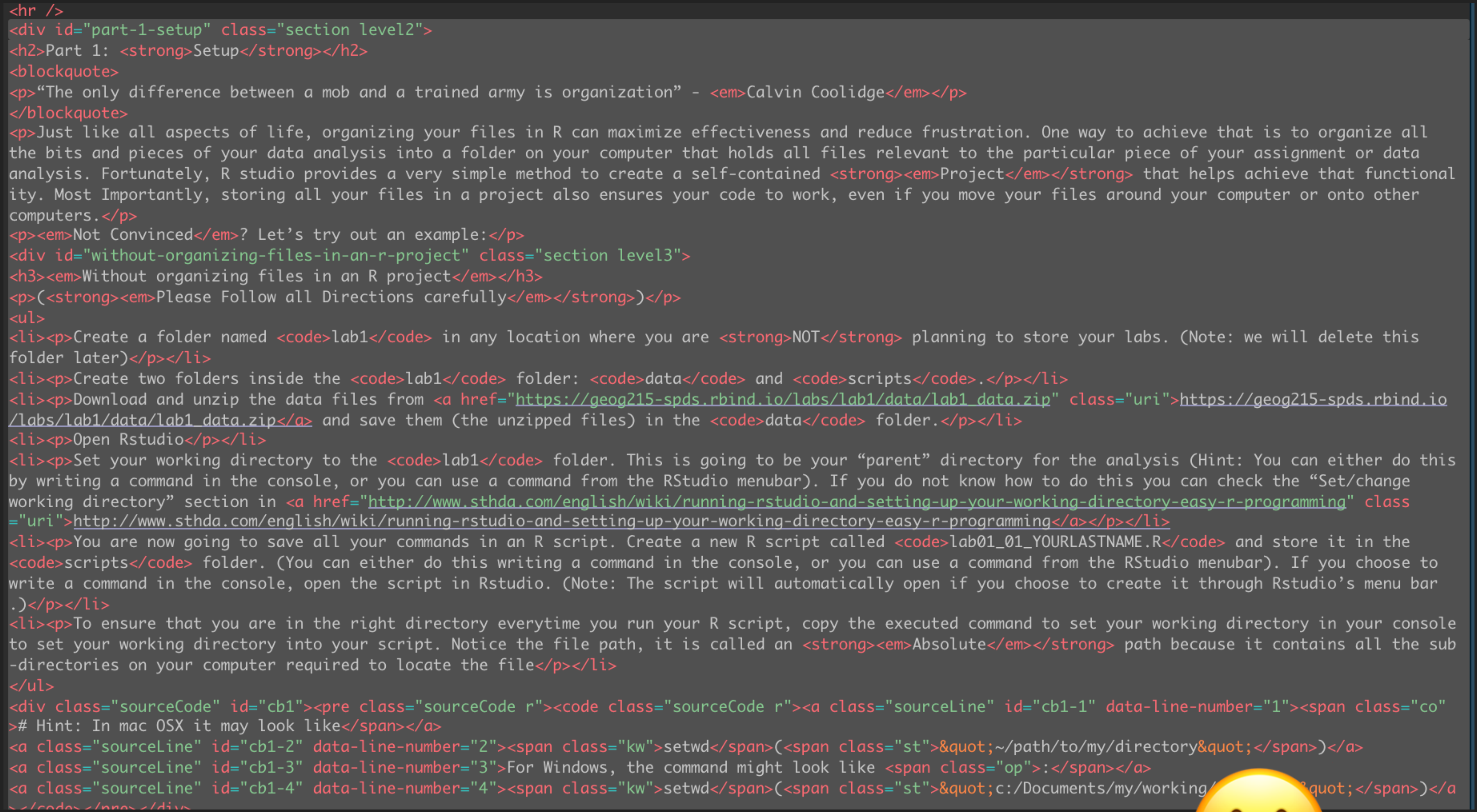
- Now type the following command in your script to read the `wdi_2018.csv` data file.

MARKDOWN CONTENT



~~~
## Part 1: **Setup**

>"The only difference between a mob and a trained army is organization" - *Calvin Coolidge*

Just like all aspects of life, organizing your files in R can maximize effectiveness and reduce frustration. One way to achieve that is to organize all the
bits and pieces of your data analysis into a folder on your computer that holds all files relevant to the particular piece of your assignment or data
analysis.  Fortunately, R studio provides a very simple method to create a self-contained ***Project*** that helps achieve that functionality. Most
Importantly, storing all your files in a project also ensures your code to work, even if you move your files around your computer or onto other computers.

*Not Convinced*? Let's try out an example:

### *Without organizing files in an R project*

(***Please Follow all Directions carefully***)

* Create a folder named `lab1` in any location where you are **NOT** planning to store your labs. (Note: we will delete this folder later)

* Create two folders inside the `lab1` folder: `data` and `scripts`.

* Download and unzip the data files from <https://geog215-spds.rbind.io/labs/lab1/data/lab1_data.zip> and save them (the unzipped files) in the `data`
folder.

* Open Rstudio

* Set your working directory to the `lab1` folder. This is going to be your "parent" directory for the analysis (Hint: You can either do this by writing a
command in the console, or you can use a command from the RStudio menubar). If you do not know how to do this you can check the "Set/change working
directory" section in <http://www.sthda.com/english/wiki/running-rstudio-and-setting-up-your-working-directory-easy-r-programming>

* You are now going to save all your commands in an R script. Create a new R script called `lab01_01_YOURLASTNAME.R` and store it in the `scripts` folder.
(You can either do this writing a command in the console, or you can use a command from the RStudio menubar). If you choose to write a command in the
console, open the script in Rstudio. (Note: The script will automatically open if you choose to create it through Rstudio's menu bar.)

* To ensure that you are in the right directory everytime you run your R script, copy the executed command to set your working directory in your console to
set your working directory into your script. Notice the file path, it is called an ***Absolute*** path because it contains all the sub-directories on your
computer required to locate the file

```{r eval =F}
# Hint: In mac OSX it may look like
setwd("~/path/to/my/directory")
For Windows, the command might look like :
setwd("c:/Documents/my/working/directory")
```
~~~

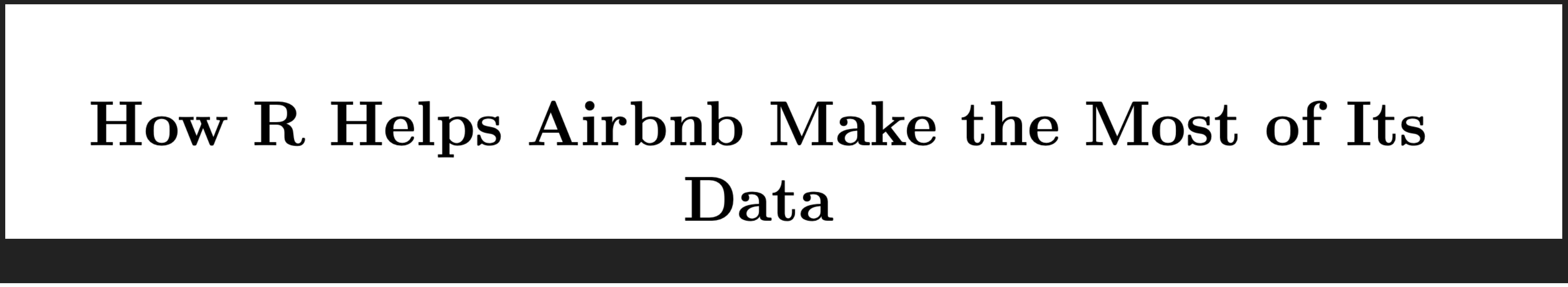
RENDERED HTML



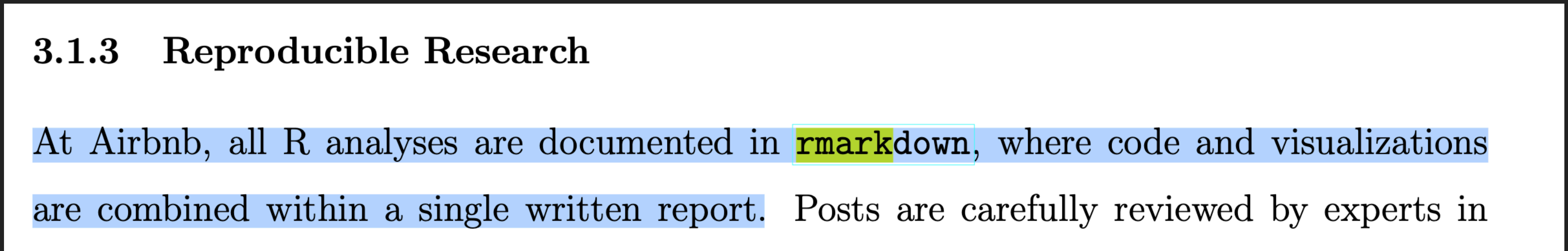
```
<hr />
<div id="part-1-setup" class="section level2">
<h2>Part 1: <strong>Setup</strong></h2>
<blockquote>
<p>“The only difference between a mob and a trained army is organization” - <em>Calvin Coolidge</em></p>
</blockquote>
<p>Just like all aspects of life, organizing your files in R can maximize effectiveness and reduce frustration. One way to achieve that is to organize all
the bits and pieces of your data analysis into a folder on your computer that holds all files relevant to the particular piece of your assignment or data
analysis. Fortunately, R studio provides a very simple method to create a self-contained <strong><em>Project</em></strong> that helps achieve that functional
ity. Most Importantly, storing all your files in a project also ensures your code to work, even if you move your files around your computer or onto other
computers.</p>
<p><em>Not Convinced</em>? Let’s try out an example:</p>
<div id="without-organizing-files-in-an-r-project" class="section level3">
<h3><em>Without organizing files in an R project</em></h3>
<p>(<strong><em>Please Follow all Directions carefully</em></strong>)</p>
<ul>
<li><p>Create a folder named <code>lab1</code> in any location where you are <strong>NOT</strong> planning to store your labs. (Note: we will delete this
folder later)</p></li>
<li><p>Create two folders inside the <code>lab1</code> folder: <code>data</code> and <code>scripts</code>.</p></li>
<li><p>Download and unzip the data files from <a href="https://geog215-spds.rbind.io/labs/lab1/data/lab1_data.zip" class="uri">https://geog215-spds.rbind.io
/labs/lab1/data/lab1_data.zip</a> and save them (the unzipped files) in the <code>data</code> folder.</p></li>
<li><p>Open Rstudio</p></li>
<li><p>Set your working directory to the <code>lab1</code> folder. This is going to be your “parent” directory for the analysis (Hint: You can either do this
by writing a command in the console, or you can use a command from the RStudio menubar). If you do not know how to do this you can check the “Set/change
working directory” section in <a href="http://www.sthda.com/english/wiki/running-rstudio-and-setting-up-your-working-directory-easy-r-programming" class
="uri">http://www.sthda.com/english/wiki/running-rstudio-and-setting-up-your-working-directory-easy-r-programming</a></p></li>
<li><p>You are now going to save all your commands in an R script. Create a new R script called <code>lab01_01_YOURLASTNAME.R</code> and store it in the
<code>scripts</code> folder. (You can either do this writing a command in the console, or you can use a command from the RStudio menubar). If you choose to
write a command in the console, open the script in Rstudio. (Note: The script will automatically open if you choose to create it through Rstudio’s menu bar
.)</p></li>
<li><p>To ensure that you are in the right directory everytime you run your R script, copy the executed command to set your working directory in your console
to set your working directory into your script. Notice the file path, it is called an <strong><em>Absolute</em></strong> path because it contains all the sub
-directories on your computer required to locate the file</p></li>
</ul>
<div class="sourceCode" id="cb1"><pre class="sourceCode r"><code class="sourceCode r"><a class="sourceLine" id="cb1-1" data-line-number="1"><span class="co"
># Hint: In mac OSX it may look like</span></a>
<a class="sourceLine" id="cb1-2" data-line-number="2"><span class="kw">setwd</span>(<span class="st">&quot;~/path/to/my/directory&quot;</span>)</a>
<a class="sourceLine" id="cb1-3" data-line-number="3">For Windows, the command might look like <span class="op">:</span></a>
<a class="sourceLine" id="cb1-4" data-line-number="4"><span class="kw">setwd</span>(<span class="st">&quot;c:/Documents/my/working/        quot;</span>)</a
></code></pre></div>
```

🙁

R MARDOWN AT AIRBNB



How R Helps Airbnb Make the Most of Its Data

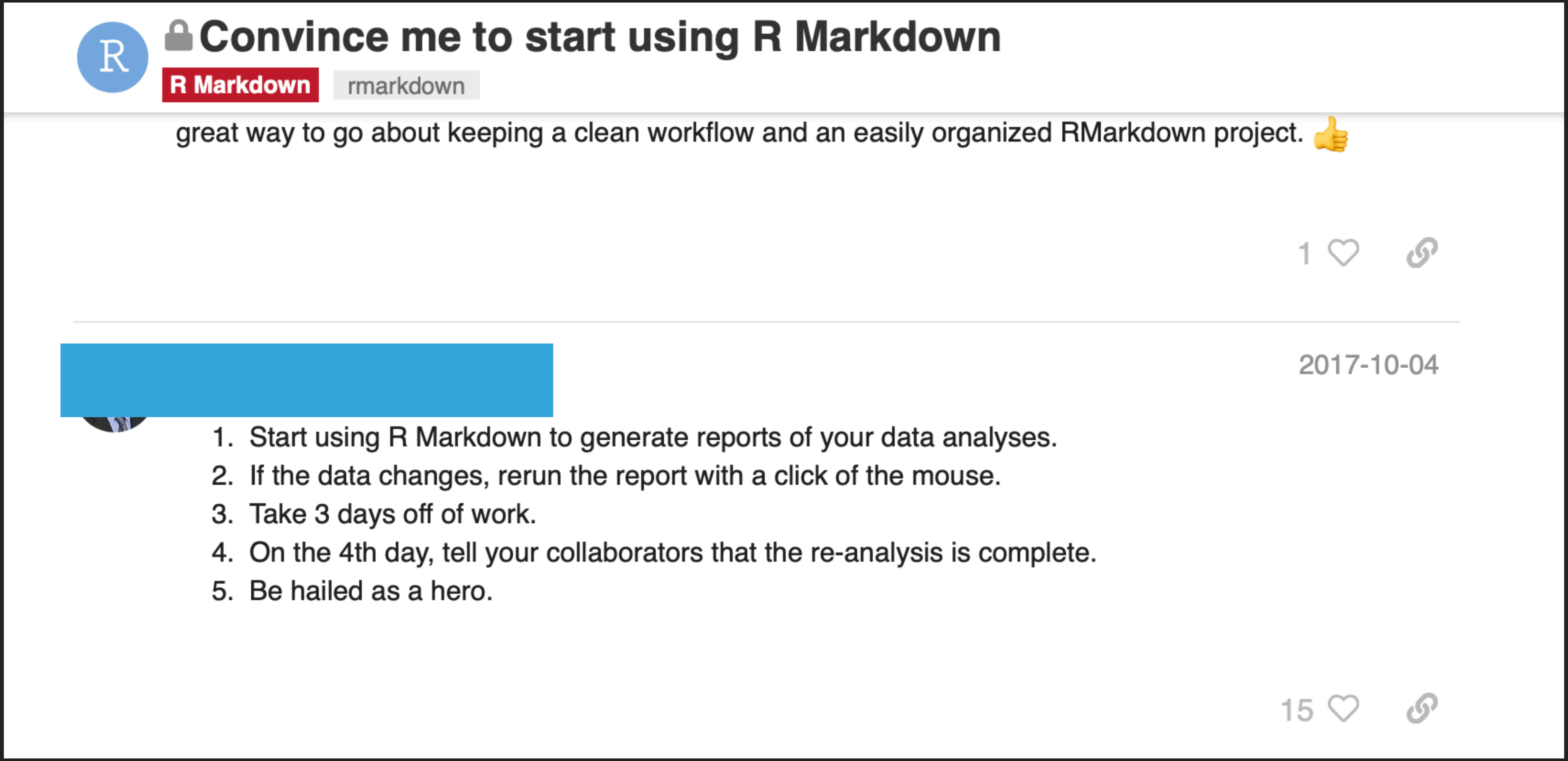


3.1.3 Reproducible Research

At Airbnb, all R analyses are documented in `rmarkdown`, where code and visualizations are combined within a single written report. Posts are carefully reviewed by experts in

Source: https://peerj.com/preprints/3182.pdf

R MARDOWN



R

🔒 **Convince me to start using R Markdown**

R Markdown rmarkdown

great way to go about keeping a clean workflow and an easily organized RMarkdown project. 👍

1 ♡

[redacted] 2017-10-04

1. Start using R Markdown to generate reports of your data analyses.
2. If the data changes, rerun the report with a click of the mouse.
3. Take 3 days off of work.
4. On the 4th day, tell your collaborators that the re-analysis is complete.
5. Be hailed as a hero.

15 ♡

DIY (R) MARKDOWN

- Download in-class exercise files from Website-> Lecture-> Jan 15
- Open Rstudio and :
  - install.packages(“rmarkdown")
- Take a look at the CountryPick2.R script and run it step by step to see what it does.
- Open CountryPick2.Rmd and fill in the empty R chunks in the Rmarkdown file

BEFORE NEXT CLASS

- Finish (Due Next Monday Jan 20 - 11:59 pm)
- Join Piazza (Participate)
- Practice, Practice, Practice
- Read Week 3 readings on Tidyverse
    - I will post readings till week 5 tonight

QUESTIONS ?