

# R Markdown

Garrick Aden-Buie  
@grrrck

Toronto Data Workshop  
2021-02-11

# R Markdown

R's Secret Ingredient

Garrick Aden-Buie  
@grrrck

Toronto Data Workshop  
2021-02-11

# R Markdown

R's Special Sauce

Garrick Aden-Buie  
@grrrck

Toronto Data Workshop  
2021-02-11

# R Markdown

An Incomplete History

Garrick Aden-Buie  
@grrrck

Toronto Data Workshop  
2021-02-11

# R Markdown

Stuff I'm Working On and Want To Show Off

Garrick Aden-Buie  
@grrrck

Toronto Data Workshop  
2021-02-11

# About Me

👋 Hi, I'm **Garrick Aden-Buie**

# About Me

👋 Hi, I'm **Garrick Aden-Buie**

🐦 @grrrck

# About Me

👋 Hi, I'm **Garrick Aden-Buie**

🐦 @grrrck

💻 garrickadenbuie.com



# About Me

👋 Hi, I'm **Garrick Aden-Buie**

🐦 @grrrck

💻 garrickadenbuie.com

® RStudio: gradethis, learnr

# What is R Markdown?

(wrong answers only)

one

two



**Break Free From Plastic** engaged  
14,734 volunteers in 55 countries  
to conduct 575 brand audits.  
These volunteers collected  
346,494 pieces of plastic waste.

**Break Free From Plastic** engaged  
14,734 volunteers in 55 countries  
to conduct 575 brand audits.  
These volunteers collected  
346,494 pieces of plastic waste.

# A brief history of rmarkdown





A brief history of  
*literate programming*



Let us change our traditional attitude to the construction of programs:

**Instead of** imagining that our main task is to **instruct a computer what to do**, let us concentrate rather on **explaining to human beings** what we want a computer to do.



I was coerced like everybody else into adopting the ideas of structured programming, because I couldn't bear to be found guilty of writing **unstructured programs**.

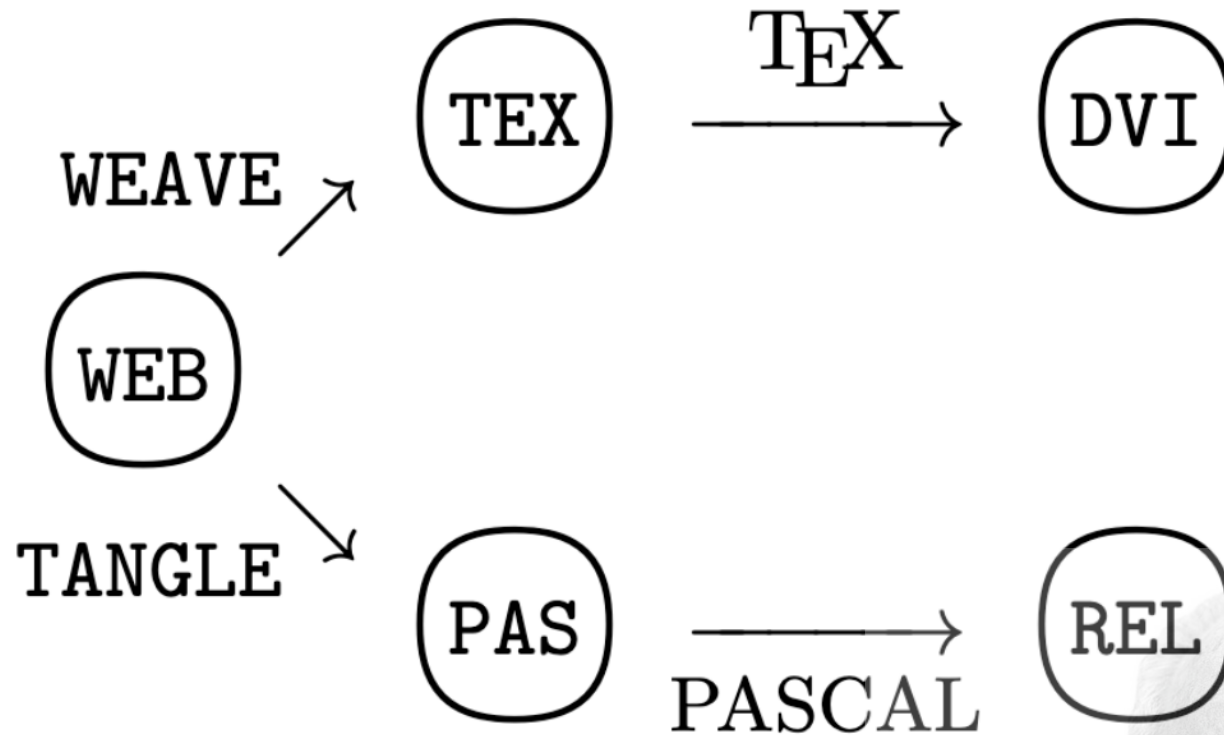
Now I have a chance to get even ... surely nobody wants to admit writing an **illiterate program**.



This language and its associated programs  
have come to be known as **the WEB system**.

I chose the name **WEB** partly because  
it was one of the few three-letter  
words of English that hadn't  
already been applied to computers.





**Figure 1.** Dual usage of a WEB file.



The result of the program will be to produce a list of the first thousand prime numbers...

```
⟨Program to print the first thousand prime numbers 2⟩ ≡  
program print_primes (output);  
const m = 1000;  
  ⟨Other constants of the program 5⟩  
var ⟨Variables of the program 4⟩  
  begin ⟨Print the first m prime numbers 3⟩ ;  
  end.
```



We shall proceed to fill out the rest of the program by making whatever decisions seem easiest at each step.

So let's come up with a list of prime numbers.

⟨Print the first  $m$  prime numbers 3⟩ ≡

⟨Fill table  $p$  with the first  $m$  prime numbers 11⟩

⟨Print table  $p$  8⟩



Now that the appropriate auxiliary variables have been introduced, the process of outputting table  $p$  almost writes itself.

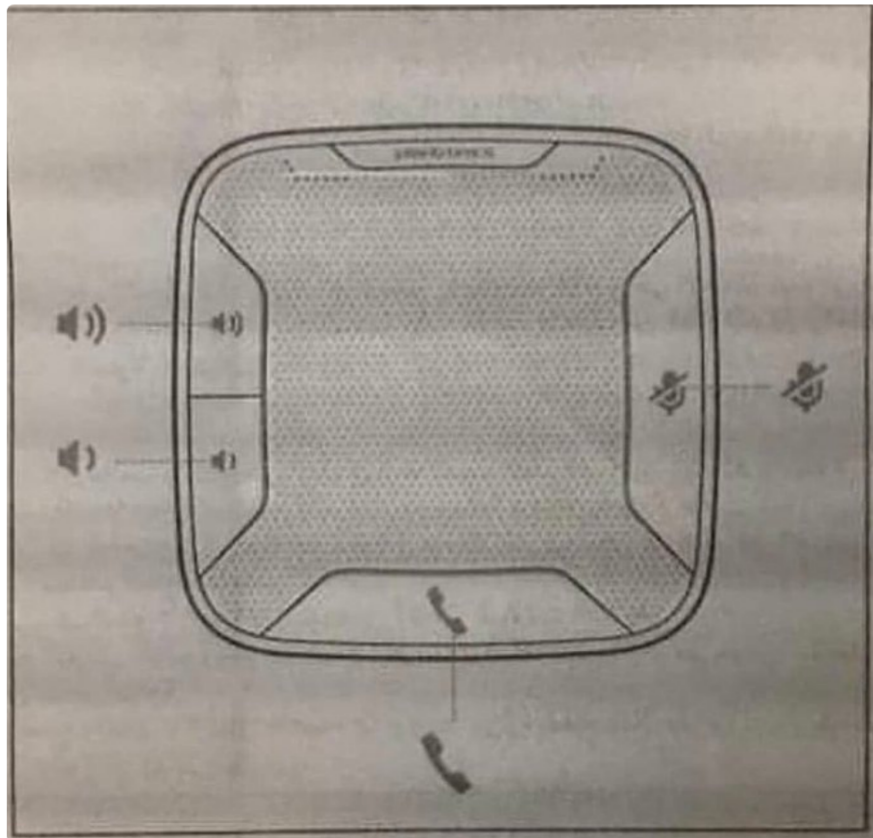
```
⟨Print table  $p$  8⟩ ≡  
begin page_number ← 1; page_offset = 1;  
while page_offset ≤  $m$  do  
  begin ⟨Output a page of answers 9⟩ ;  
    page_number ← page_number + 1;  
    page_offset ← page_offset +  $rr * cc$ ;  
  end;  
end;
```





**Ricardo Ferreira** @riferrei · Feb 6, 2021

Developers commenting their code



**Tolga Mirmirik**

@mirmirik

Always...





A brief history of  
*literate programming*  
in R

sweave



# sweave

```
\documentclass{article}

\usepackage{amsmath}
\usepackage{amscd}
\usepackage[utf8]{inputenc}

\begin{document}
\SweaveOpts{concordance=TRUE}

\title{An Sweave Demo}
\author{Charles J. Geyer}
\maketitle

% . . . .
```

# sweave

This is a demo for using the `\verb@Sweave@` command in R. To get started make a regular `\LaTeX\` file (like this one) but give it the suffix `\verb@.Rnw@` instead of `\verb@.tex@` and then turn it into a `\LaTeX\` file (`\verb@foo.tex@`) with the (unix) command

```
\begin{verbatim}
R CMD Sweave foo.Rnw
\end{verbatim}
```

Well, we can now include R in our document. Here's a simple example

```
<<two>>=
2 + 2
@
```

# sweave

Figure~\ref{fig:one} (p.~\pageref{fig:one})

is produced by the following code

```
<<label=fig1plot,include=FALSE>>=
```

```
plot(x, y)
```

```
abline(out1)
```

```
@
```

```
\begin{figure}
```

```
\begin{center}
```

```
<<label=fig1,fig=TRUE,echo=FALSE>>=
```

```
<<fig1plot>>
```

```
@
```

```
\end{center}
```

```
\caption{Scatter Plot with Regression Line}
```

```
\label{fig:one}
```

```
\end{figure}
```

Note that \verb@x@, \verb@y@, and \verb@out1@ are remembered from the preceding code chunk. We don't have to regenerate them.

All code chunks are part of one R ``session''.

# sweave

Figure~\ref{fig:one} (p.~\pageref{fig:one})

is produced by the following code

```
<<label=fig1plot,include=FALSE>>=
```

```
plot(x, y)
```

```
abline(out1)
```

```
@
```

```
\begin{figure}
```

```
\begin{center}
```

```
<<label=fig1,fig=TRUE,echo=FALSE>>=
```

```
<<fig1plot>>
```

```
@
```

```
\end{center}
```

```
\caption{Scatter Plot with Regression Line}
```

```
\label{fig:one}
```

```
\end{figure}
```

Note that \verb@x@, \verb@y@, and \verb@out1@ are remembered from the preceding code chunk. We don't have to regenerate them.

All code chunks are part of one R ``session''.

# sweave

```
Figure~\ref{fig:one} (p.~\pageref{fig:one})  
is produced by the following code  
<<label=fig1plot,include=FALSE>>  
plot(x, y)  
abline(out1)  
@  
\begin{figure}  
\begin{center}  
<<label=fig1,fig=TRUE,echo=FALSE>>=  
<<fig1plot>>  
@  
\end{center}  
\caption{Scatter Plot with Regression Line}  
\label{fig:one}  
\end{figure}
```

Note that `\verb@x@`, `\verb@y@`, and `\verb@out1@` are remembered from the preceding code chunk. We don't have to regenerate them. All code chunks are part of one R ``session''.

(for once we won't show the code chunk itself, look at `foo.Rnw` if you want to see what the actual code chunk was).

Figure 1 (p. 2) is produced by the following code

```
> plot(x, y)  
> abline(out1)
```

Note that `x`, `y`, and `out1` are remembered from the preceding code chunk. We don't have to regenerate them. All code chunks are part of one R "session".

# sweave

Figure~\ref{fig:one} (p.~\pageref{fig:one})

is produced by the following code

```
<<label=fig1plot,include=FALSE>>=
```

```
plot(x, y)
```

```
abline(out1)
```

```
@
```

```
\begin{figure}
```

```
\begin{center}
```

```
<<label=fig1,fig=TRUE,echo=FALSE>>=
```

```
<<fig1plot>>
```

```
@
```

```
\end{center}
```

```
\caption{Scatter Plot with Regression Line}
```

```
\label{fig:one}
```

```
\end{figure}
```

Note that \verb@x@, \verb@y@, and \verb@out1@ are remembered from the preceding code chunk. We don't have to regenerate them.

All code chunks are part of one R ``session''.



# sweave

Figure~\ref{fig:one} (p.~\pageref{fig:one})

is produced by the following code

```
<<label=fig1plot,include=FALSE>>=
```

```
plot(x, y)
```

```
abline(out1)
```

```
@
```

```
\begin{figure}
```

```
\begin{center}
```

```
<<label=fig1,fig=TRUE,echo=FALSE>>=
```

```
<<fig1plot>>
```

```
@
```

```
\end{center}
```

```
\caption{Scatter Plot with Regression Line}
```

```
\label{fig:one}
```

```
\end{figure}
```

Note that \verb@x@, \verb@y@, and \verb@out1@ are remembered from the preceding code chunk. We don't have to regenerate them.

All code chunks are part of one R ``session''.

# sweave

Figure~\ref{fig:one} (p.~\pageref{fig:one})  
is produced by the following code

```
<<label=fig1plot,include=FALSE>>=  
plot(x, y)  
abline(out1)  
@  
\begin{figure}  
\begin{center}  
<<label=fig1,fig=TRUE,echo=FALSE>>=  
<<fig1plot>>  
@  
\end{center}  
\caption{Scatter Plot with Regression Line}  
\label{fig:one}  
\end{figure}
```

Note that \verb@x@, \verb@y@, and \verb@ou  
the preceding code chunk. We don't have t  
All code chunks are part of one R ``sessio

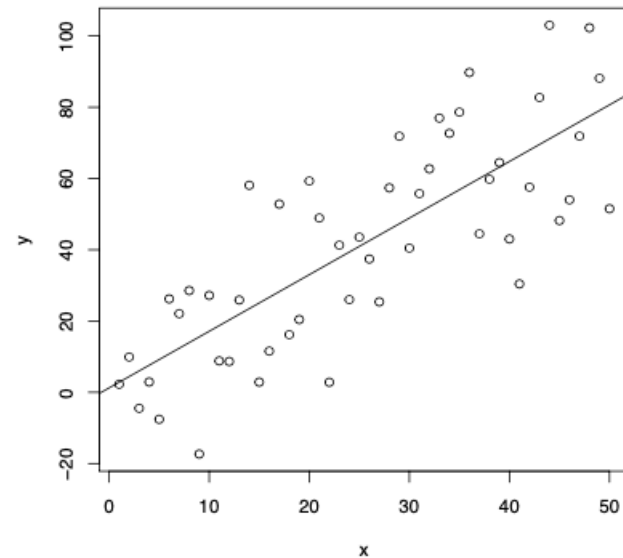


Figure 1: Scatter Plot with Regression Line

# knitr

knitr



# knitr

Yihui Xie - Interview by DataScience.LA at useR 2014



# knitr

1. Write in markdown
2. Cleaner chunk and inline R code syntax
3. Easy figures
4. Still iterate

# knitr

```
Let's write another program that computes prime numbers, called `prime_numbers(  
...  
prime_numbers <- function(m = 1) {  
  <<prime-numbers>>  
}  
...
```

# knitr

```
Let's write another program that computes prime numbers, called `prime_numbers(  
...  
prime_numbers <- function(m = 1) {  
  <<prime-numbers>>  
}  
...
```

Let's write another program that computes prime numbers, called `prime_numbers()`.

```
prime_numbers <- function(m = 1) {  
  <<prime-numbers>>  
}
```



# knitr

Well, we can now include R in our document. Here's a simple example.

```
```${r two}  
2 + 2  
```
```

# knitr

Well, we can now include R in our document. Here's a simple example.

```
```${r two}
2 + 2
```
```

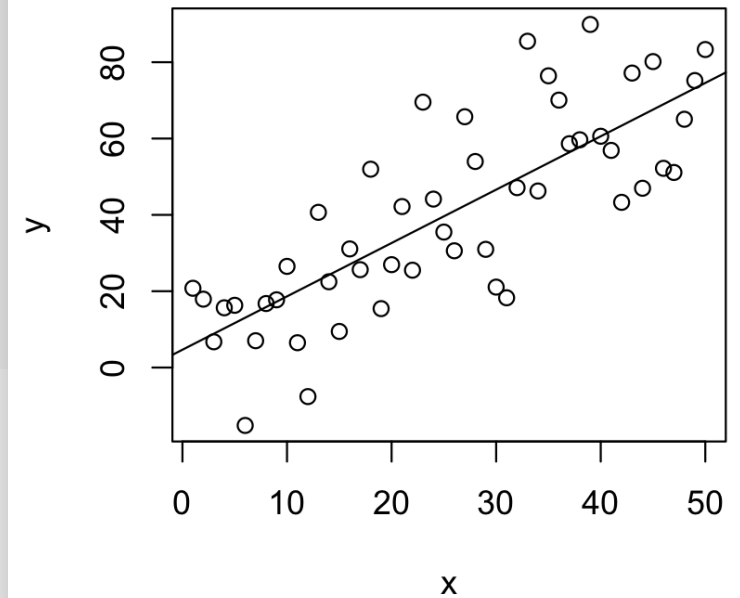
Well, we can now include R in our document. Here's a simple example.

```
2 + 2
```

```
## [1] 4
```

Figure 1 is produced by the following code

```
```{r fig1plot, fig.width = 4, fig.height = 4}
n <- 50
x <- seq(1, n)
y <- 3 + (1.5 * x) + (17.3 * rnorm(n))
fit <- lm(y ~ x)
plot(x, y)
par(mar = rep(0, 4))
abline(fit)
```
```



# knitr

For one point, `x` is `r x[10]`, `y` is `r y[10]` and we predict `y` will be `r predict(fit, list(x = 10))`.

# knitr


```
For one point, `x` is `r x[10]`, `y` is `r y[10]` and we predict  
`y` will be `r predict(fit, list(x = 10))`.
```

For one point, `x` is 10, `y` is 26.5050704 and we predict `y` will be 18.6470099.

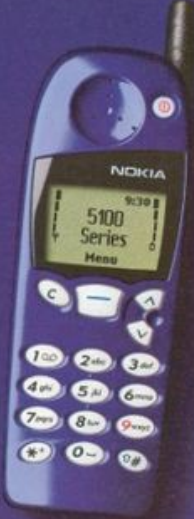
# knitr with pandoc

# knitr with pandoc

Along with useful features like long battery life, the Nokia 5100 Series wireless phone sports Xpress-on™ color covers that snap on and off, putting the power of change in your hands.



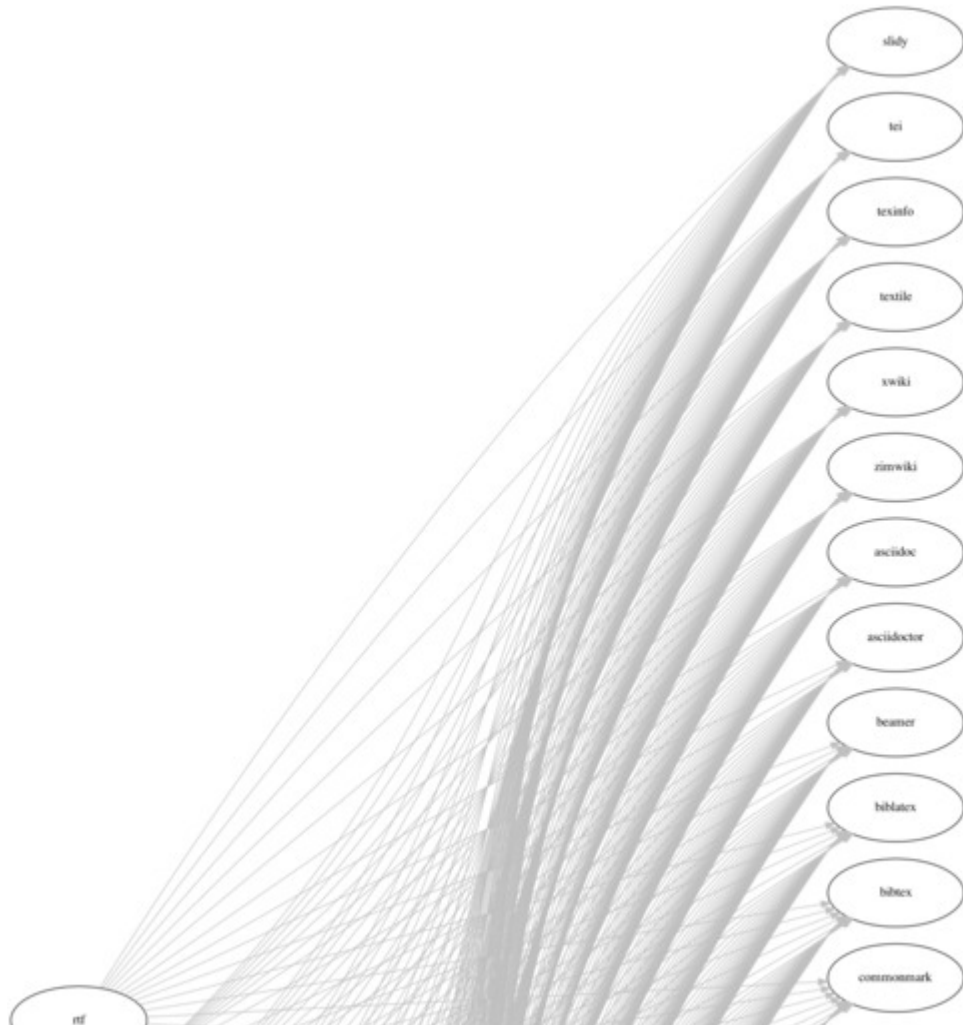
You have the power to change things.  
Well, at least the power to change the color of your phone.




**NOKIA**  
CONNECTING PEOPLE  
[www.NokiaUSA.com](http://www.NokiaUSA.com)

© 2004 Nokia Business Phones, Inc. Nokia, the Connecting People logo, the Nokia 5100 Series and Xpress-on™ are trademarks of Nokia Corporation and/or its affiliates. Nokia is a leader in the 5100 Series.

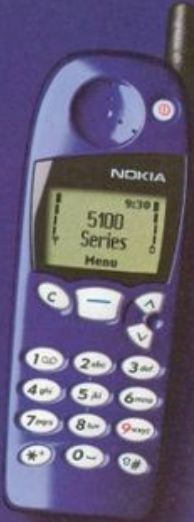
# knitr with pandoc



Along with useful features like long battery life, the Nokia 5100 Series wireless phone sports Xpress-on™ color covers that snap on and off, putting the power of change in your hands.



You have the power to change things.  
Well, at least the power to change the color of your phone.

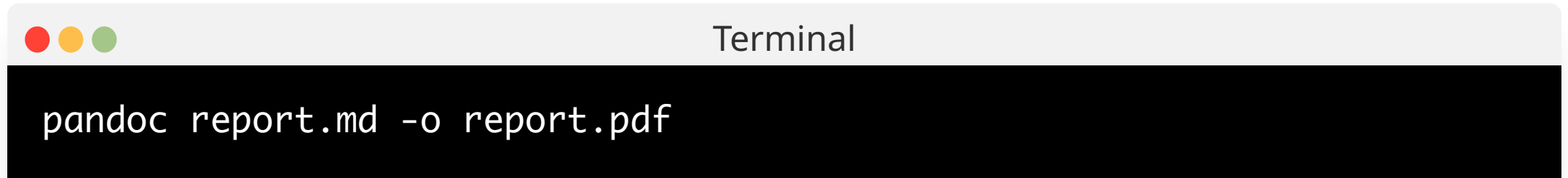


**NOKIA**  
CONNECTING PEOPLE  
[www.NokiaUSA.com](http://www.NokiaUSA.com)

© 2004 Nokia Business Phones, Inc. Nokia, the Connecting People logo, the Nokia 5100 Series and Nokia™ are trademarks of Nokia Corporation and/or its affiliates. Nokia is a leader in the 5100 Series.

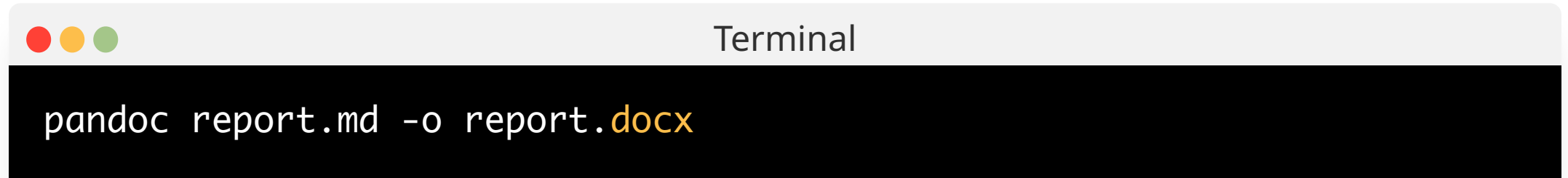


# knitr with pandoc

A terminal window with a light gray title bar containing three colored window control buttons (red, yellow, green) on the left and the word "Terminal" in the center. The main area of the terminal is black with white text showing the command `pandoc report.md -o report.pdf`.

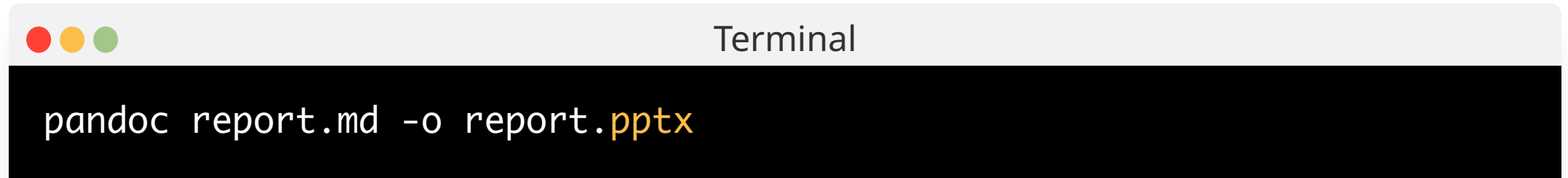
```
Terminal  
pandoc report.md -o report.pdf
```

# knitr with pandoc

A terminal window with a light gray title bar containing three colored window control buttons (red, yellow, green) on the left and the word "Terminal" in the center. The main area of the terminal is black with white text. The text displayed is the command "pandoc report.md -o report.docx", where ".docx" is highlighted in orange.

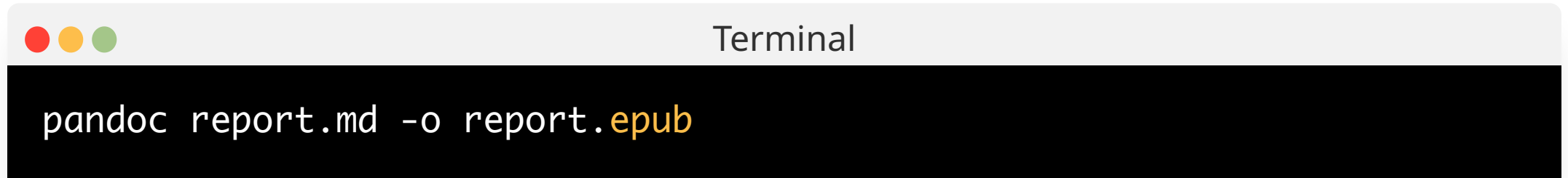
```
pandoc report.md -o report.docx
```

# knitr with pandoc

A terminal window with a light gray title bar containing three colored window control buttons (red, yellow, green) on the left and the word "Terminal" in the center. The main area of the terminal is black with white text. The text shows a command: "pandoc report.md -o report.pptx".

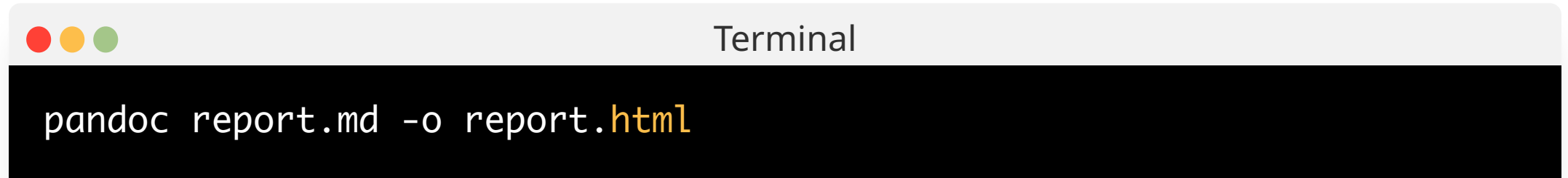
```
Terminal  
pandoc report.md -o report.pptx
```

# knitr with pandoc

A terminal window with a light gray title bar containing three colored window control buttons (red, yellow, green) on the left and the word "Terminal" in the center. The main area of the terminal is black with white text. The text displayed is the command "pandoc report.md -o report.epub", where "epub" is highlighted in orange.

```
pandoc report.md -o report.epub
```

# knitr with pandoc

A terminal window with a light gray title bar containing three colored window control buttons (red, yellow, green) on the left and the word "Terminal" in the center. The main area of the terminal is black with white text. The text shows a command being entered: "pandoc report.md -o report.html".

```
Terminal  
pandoc report.md -o report.html
```

# knitr with pandoc



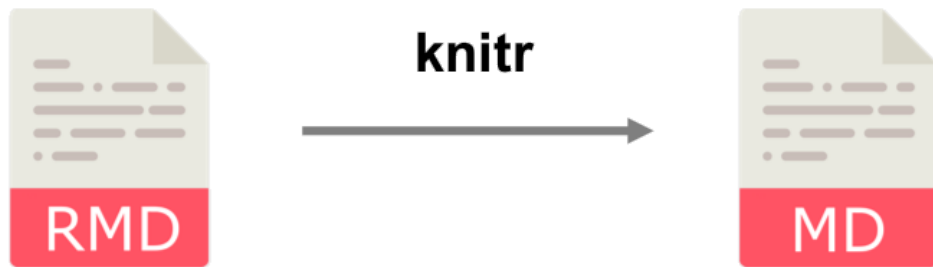
Terminal

```
pandoc report.md -o report.html --no-highlight \  
  --css assets/css/title-slide.css \  
  --css assets/css/toronto-data-workshop.css \  
  --section-divs --standalone --variable math=true
```

# R Markdown

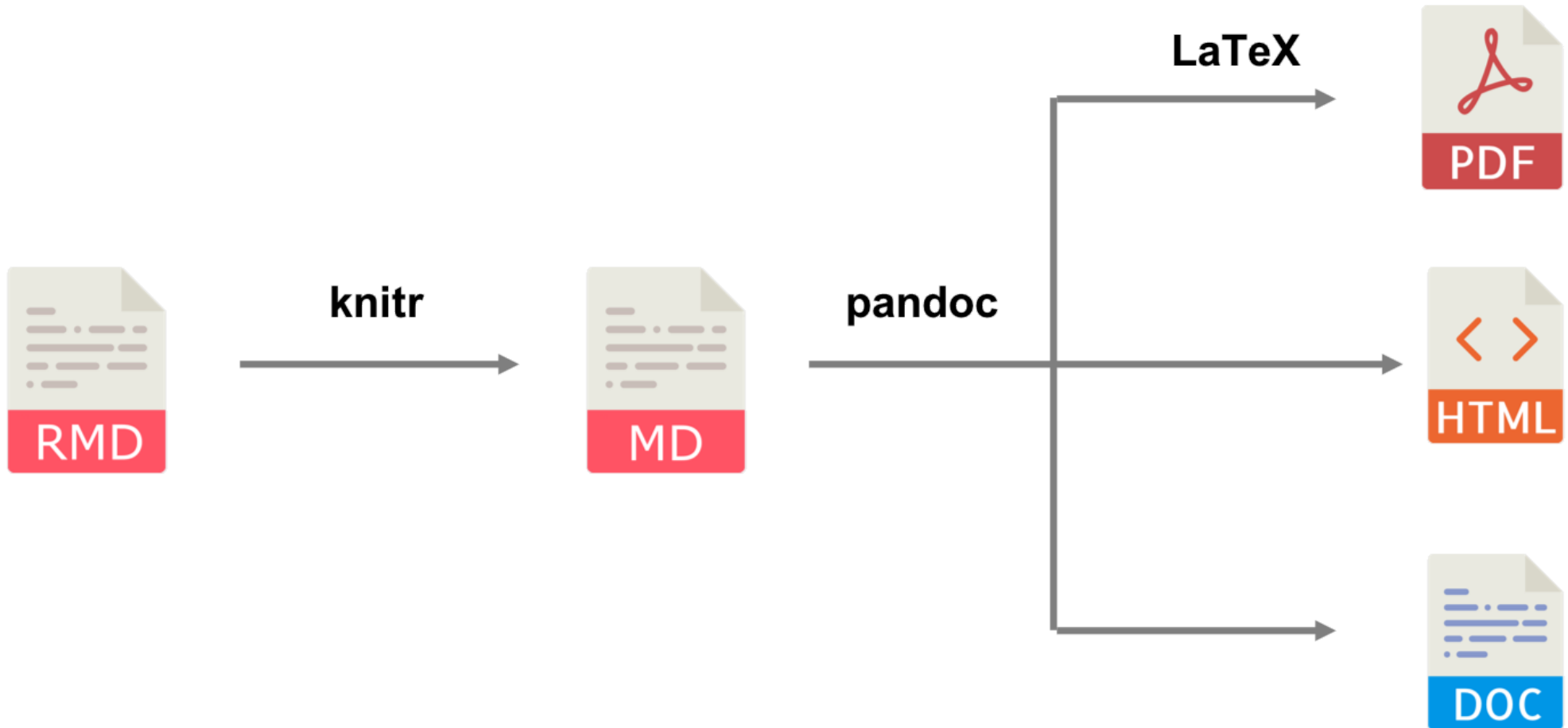


# R Markdown





# R Markdown



# R Markdown



# R Markdown



Steve Jobs is shown on the left side of the slide, standing in front of a blue background with white text. The visible text includes "inger g" and "ed".



An iPhone is shown in the center of the slide, tilted at an angle.

- Works like magic
- No stylus
- Far more accurate
- Ignores unintended touch
- Multi-finger gestures
- Patented !



PDF Reports

Word Documents

PowerPoint Presentations

Interactive Dashboards

Books

Websites



Slides Like These!

# rmarkdown

[rmarkdown.rstudio.com](http://rmarkdown.rstudio.com)

## R Markdown

from  RStudio

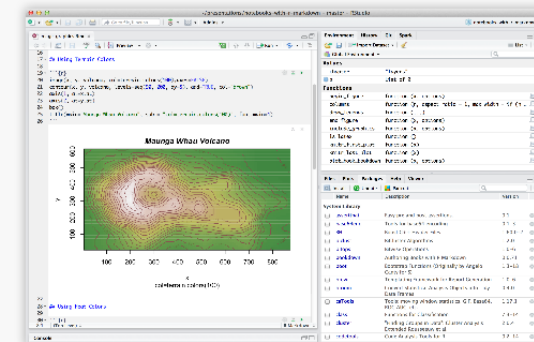
[Get Started](#) [Gallery](#) [Formats](#) [Articles](#) [Book](#) [References](#) 



Analyze. Share. Reproduce.

Your data tells a story. Tell it with R Markdown.  
Turn your analyses into high quality documents,  
reports, presentations and dashboards.

R Markdown documents are fully reproducible.  
Use a productive **notebook interface** to weave  
together narrative text and code to produce  
elegantly formatted output. Use **multiple  
languages** including R, Python, and SQL.



R Markdown supports dozens of static and  
dynamic output formats including **HTML**, **PDF**,  
**MS Word**, **Beamer**, **HTML5 slides**, **Tufte-style**

ероху

# knitr documents can write themselves

```
```{r}
years <- c(2019, 2020)
grand_total <- c(858462, 346494)

```{r plastics}
items <- paste(
  "\n- In", years, "we collected", grand_total, "pieces of plastic."
)
items
```
```



# knitr documents can write themselves

```
```{r}
years <- c(2019, 2020)
grand_total <- c(858462, 346494)

```{r plastics}
items <- paste(
  "\n- In", years, "we collected", grand_total, "pieces of plastic."
)
items
```
```

```
## [1] "\n- In 2019 we collected 858462 pieces of plastic."
## [2] "\n- In 2020 we collected 346494 pieces of plastic."
```

# knitr documents can write themselves

```
```{r}
years <- c(2019, 2020)
grand_total <- c(858462, 346494)

```{r plastics, results = "asis"}
items <- paste(
  "\n- In", years, "we collected", grand_total, "pieces of plastic."
)
cat(items)
```
```

# knitr documents can write themselves

```
```{r}
years <- c(2019, 2020)
grand_total <- c(858462, 346494)

```{r plastics, results = "asis"}
items <- paste(
  "\n- In", years, "we collected", grand_total, "pieces of plastic."
)
cat(items)
```
```

- In 2019 we collected 858462 pieces of plastic.
- In 2020 we collected 346494 pieces of plastic.

# knitr documents can write themselves

```
```{r}
years <- c(2019, 2020)
grand_total <- c(858462, 346494)

```{r plastics, results = "asis"}
items <- paste(
  "\n- In", years, "we collected", grand_total, "pieces of plastic."
)
cat(items)
```
```

- In 2019 we collected 858462 pieces of plastic.
- In 2020 we collected 346494 pieces of plastic.

# Meet glue

```
paste(  
  "\n- In", years, "we collected", grand_total, "pieces of plastic."  
)
```

```
## [1] "\n- In 2019 we collected 858462 pieces of plastic."  
## [2] "\n- In 2020 we collected 346494 pieces of plastic."
```

# Meet glue

```
paste(  
  "\n- In", years, "we collected", grand_total, "pieces of plastic."  
)
```

```
## [1] "\n- In 2019 we collected 858462 pieces of plastic."  
## [2] "\n- In 2020 we collected 346494 pieces of plastic."
```

---

```
library(glue)
```

```
glue("\n- In {years} we collected {grand_total} pieces of plastic.")
```

```
## - In 2019 we collected 858462 pieces of plastic.  
## - In 2020 we collected 346494 pieces of plastic.
```

# epoxy, like superglue

👉 [gadenbuie/epoxy](#)

# epoxy, like superglue

👉 [gadenbuie/epoxy](#)



# epoxy, like superglue

👉 [gadenbuie/epoxy](#)

```
library(epoxy)
```

# epoxy, like superglue

```
glue("\n- In {years} we collected {grand_total} pieces of plastic.")
```

```
## - In 2019 we collected 858462 pieces of plastic.
```

```
## - In 2020 we collected 346494 pieces of plastic.
```

# epoxy, like superglue

```
```${epoxy}
- In {years} we collected {grand_total} pieces of plastic.
```
```

- In 2019 we collected 858462 pieces of plastic.
- In 2020 we collected 346494 pieces of plastic.

# epoxy, like superglue

#tidytuesday

Break Free From Plastics

```
library(dplyr)

# plastics <- tidytuesdayR::tt_load(2021, week = 5)$plastics
plastics <- readr::read_csv(here::here("data", "plastics.csv"))

plastics_grand_summary <-
  plastics %>%
  group_by(country, year, num_events, volunteers) %>%
  summarize(
    grand_total = sum(grand_total, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  arrange(year, desc(grand_total))
```

# epoxy, like superglue

```
plastics_grand_summary
```

```
## # A tibble: 107 × 5
##   country      year num_events volunteers grand_total
##   <chr>      <dbl>     <dbl>     <dbl>     <dbl>
## 1 Taiwan_ Republic of China ... 2019         2     31318     241292
## 2 NIGERIA    2019        14     1648     161140
## 3 EMPTY     2019       145     1416     113910
## 4 Philippines 2019        20     3751     74032
## 5 Indonesia  2019        32     6850     26618
## 6 ECUADOR   2019         1      455     25430
## 7 Vietnam   2019         4      400     21774
## 8 Kenya    2019         5     1560     18988
## 9 Cameroon  2019        10      387     17190
## 10 Switzerland 2019         6      327     15002
## # i 97 more rows
```

# epoxy, like superglue

```
plastics_year_summary <-  
  plastics_grand_summary %>%  
  group_by(year) %>%  
  summarize(  
    countries = n(),  
    across(c(num_events, volunteers, grand_total), sum, na.rm = TRUE)  
  ) %>%  
  mutate(across(-(1:2), format, big.mark = ","))
```

# epoxy, like superglue

```
plastics_year_summary
```

```
## # A tibble: 2 × 5
##   year countries num_events volunteers grand_total
##   <dbl>      <int> <chr>      <chr>      <chr>
## 1  2019         52 483      72,236     858,462
## 2  2020         55 575     14,734     346,494
```

# epoxy, like superglue

```
```{epoxy data = plastics_year_summary}
- **In {year}**, _Break Free From Plastic_ engaged {volunteers} volunteers in
  {countries} countries to conduct {num_events} brand audits.
  These volunteers collected {grand_total} pieces of plastic waste.
```
```



# epoxy, like superglue

- **In 2019**, *Break Free From Plastic* engaged 72,236 volunteers in 52 countries to conduct 483 brand audits. These volunteers collected 858,462 pieces of plastic waste.
- **In 2020**, *Break Free From Plastic* engaged 14,734 volunteers in 55 countries to conduct 575 brand audits. These volunteers collected 346,494 pieces of plastic waste.

# shinyComponents

# R Markdown all the things

👉 [gadenbuie/shinyComponents](https://github.com/gadenbuie/shinyComponents)

# Resources

# Links and Further Reading

- [epoxy](#)
- [shinyComponents](#)
- [R Markdown Cookbook](#)
- [Wrap Vectors in Markdown Formatting • gluedown](#)
- [Yihui Xie - New developments in knitr and R Markdown v2 \(2014\)](#)
- [Yihui Xie - Interview by DataScience.LA at useR 2014](#)