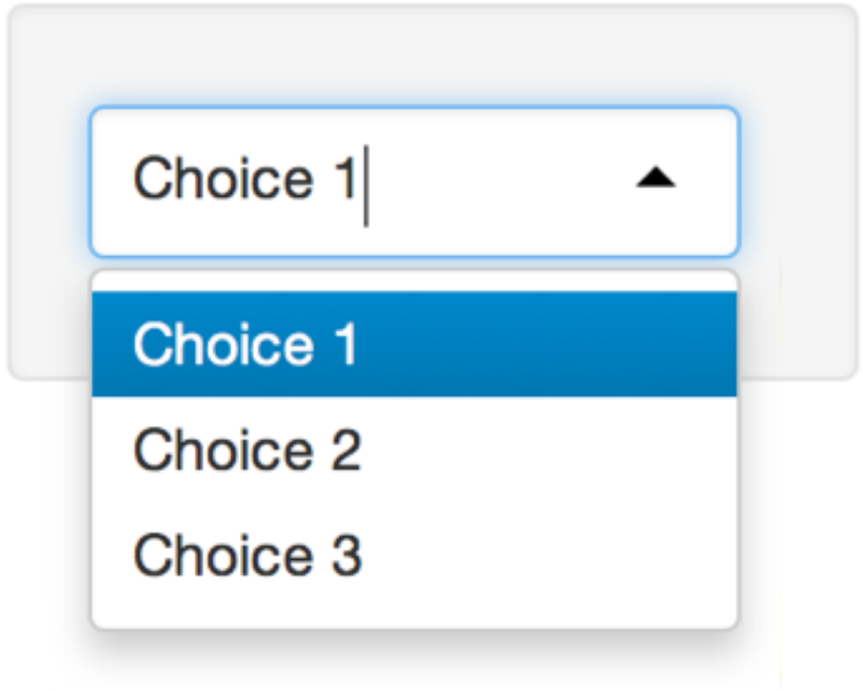
All Training materials are provided "as is" and without warranty and RStudio disclaims any and all express and implied warranties including without limitation the implied warranties of title, fitness for a particular purpose, merchantability and noninfringement.

Studio

Interactive Data Display

with Shiny and R Markdown



Garrett Grolemund

Master Instructor, RStudio

June 2014

HELLO

my name is

Garrett

garrett@rstudio.com

@StatGarrett

Warm up

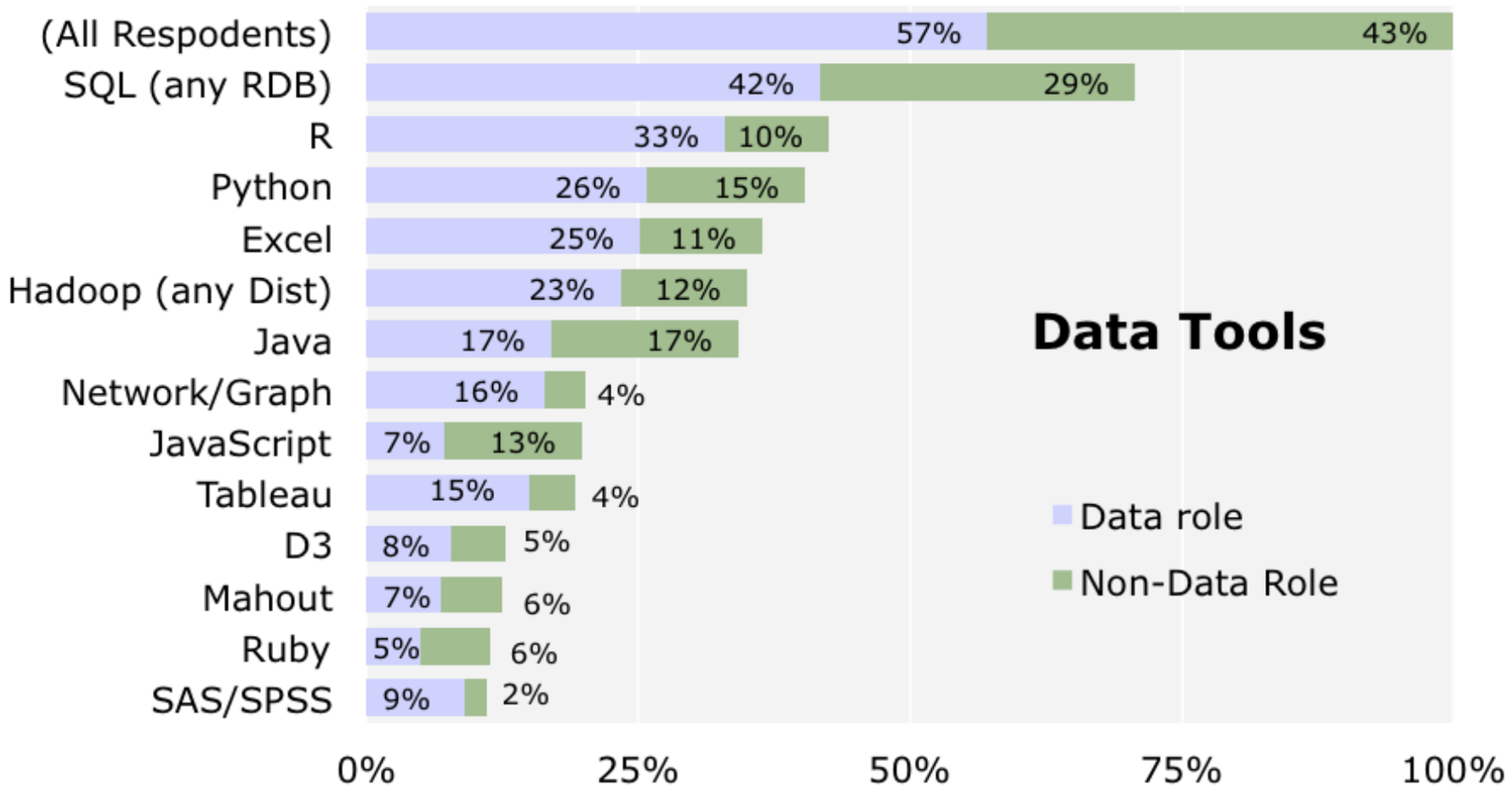
Introduce yourself to your neighbor. Tell them

* your name

* your job

* 2 to 3 interesting things about yourself

2013 Data Science Salary Survey



Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

O'REILLY®

Source: http://r4stats.com/articles/popularity/

RStudio

Go to file/function

Project: (None)

Console ~/

```
> WorldPhones
     N.Amer Europe Asia S.Amer Oceania Africa Mid.Amer
1951  45939  21574 2876   1815    1646     89      555
1956  60423  29990 4708   2568    2366   1411      733
1957  64721  32510 5230   2695    2526   1546      773
1958  68484  35218 6662   2845    2691   1663      836
1959  71799  37598 6856   3000    2868   1769      911
1960  76036  40341 8220   3145    3054   1905     1008
1961  79831  43173 9053   3338    3224   2005     1076
>
>
> summary(iris$Sepal.Width)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.000   2.800   3.000   3.057   3.300   4.400
>
>
> lm(mpg ~ disp + hp, data = mtcars)

Call:
lm(formula = mpg ~ disp + hp, data = mtcars)

Coefficients:
(Intercept)         disp           hp
   30.73590     -0.03035     -0.02484


>
> hist(iris$Sepal.Width, border = "white", col = "darkgrey")
>
```

Environment | History

Import Dataset | Clear | Grid

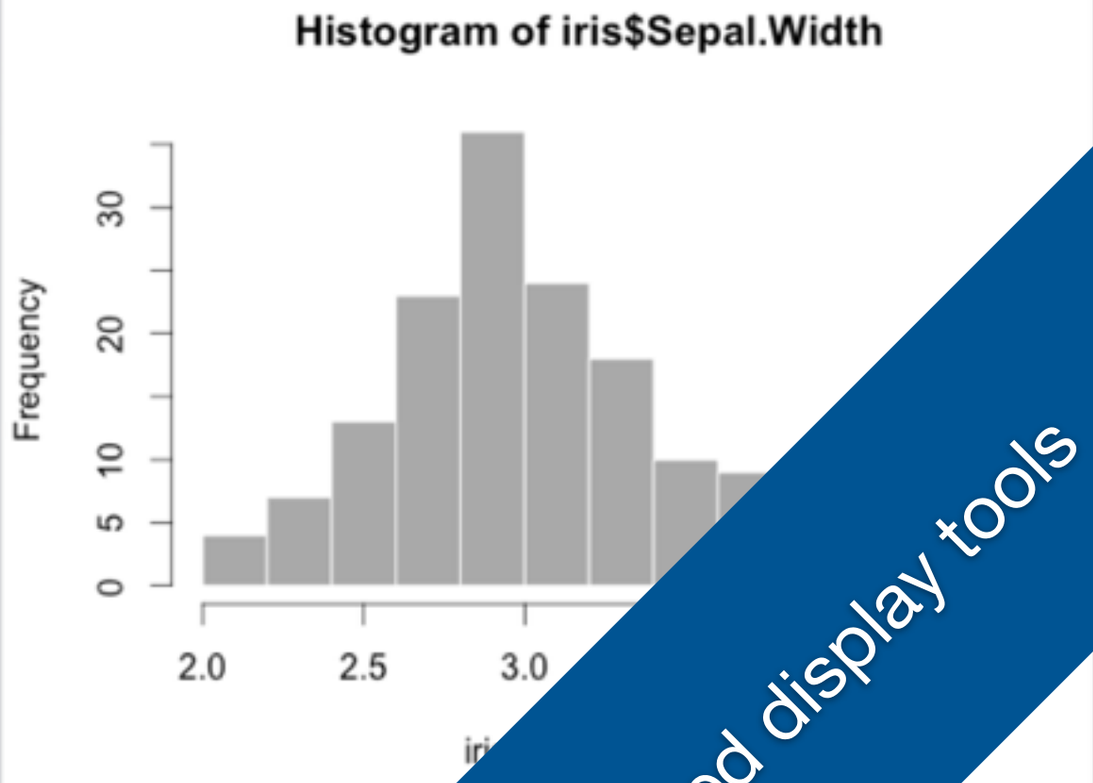
Global Environment

Name	Type	Length	Size	Value

Environment is empty

Files | Plots | Packages | Help | Viewer

Zoom | Export | Clear All



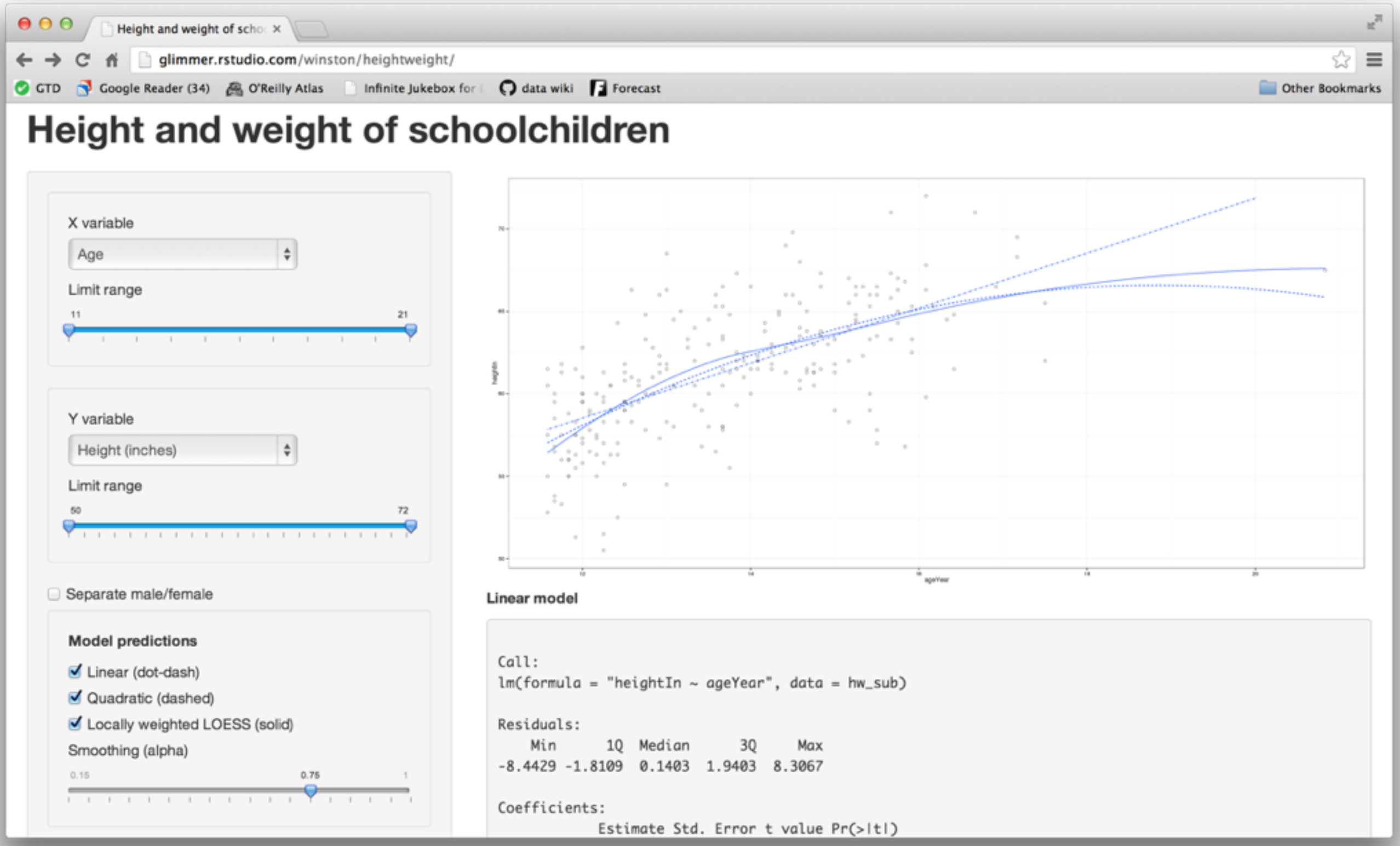
Limited display tools

Shiny

Analytic
Power

+

Display abilities
and interactivity



http://glimmer.rstudio.com/winston/heightweight/

Shiny

An R package that does two things

1. Creates reactive R objects
2. Builds HTML web pages

Studio

R Markdown

Analytic
Power

+

Microsoft Word

Reveal.js
ioslides, Beamer

Report generation

1. Reports in R
2. Interactive Reports
3. Web Apps
4. Sharing

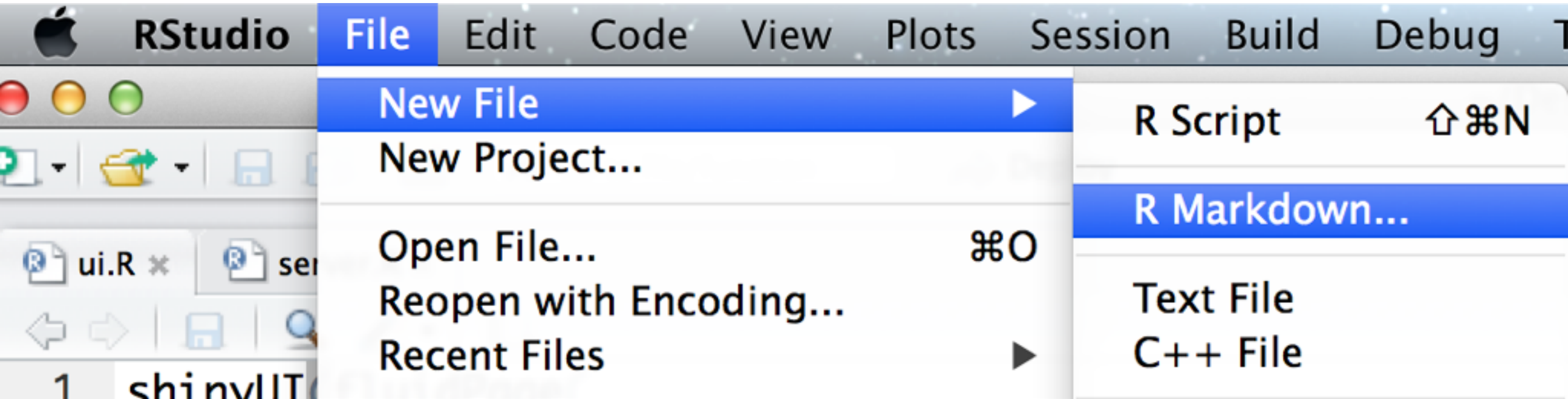
RStudio IDE

text editor

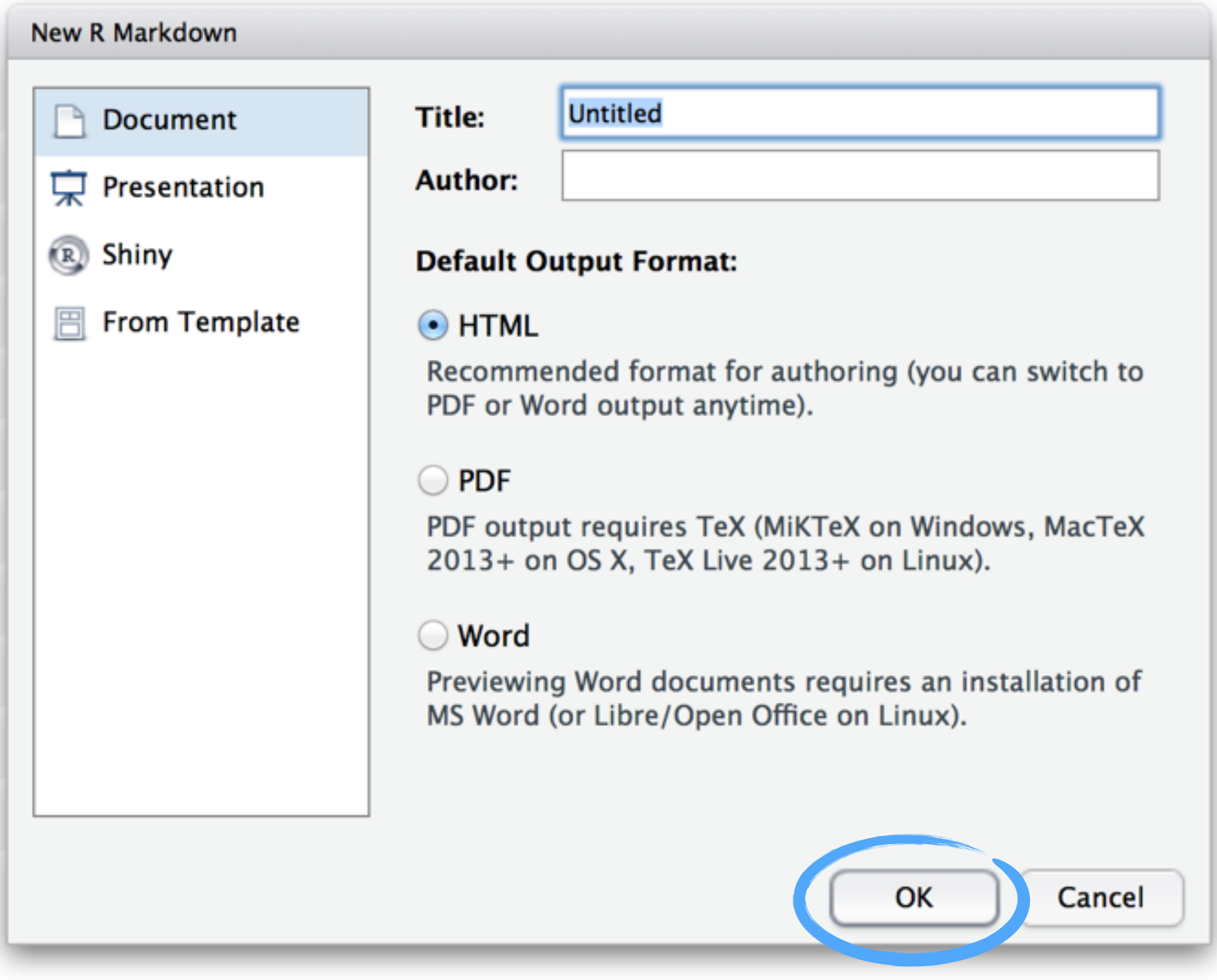
RStudio text editor

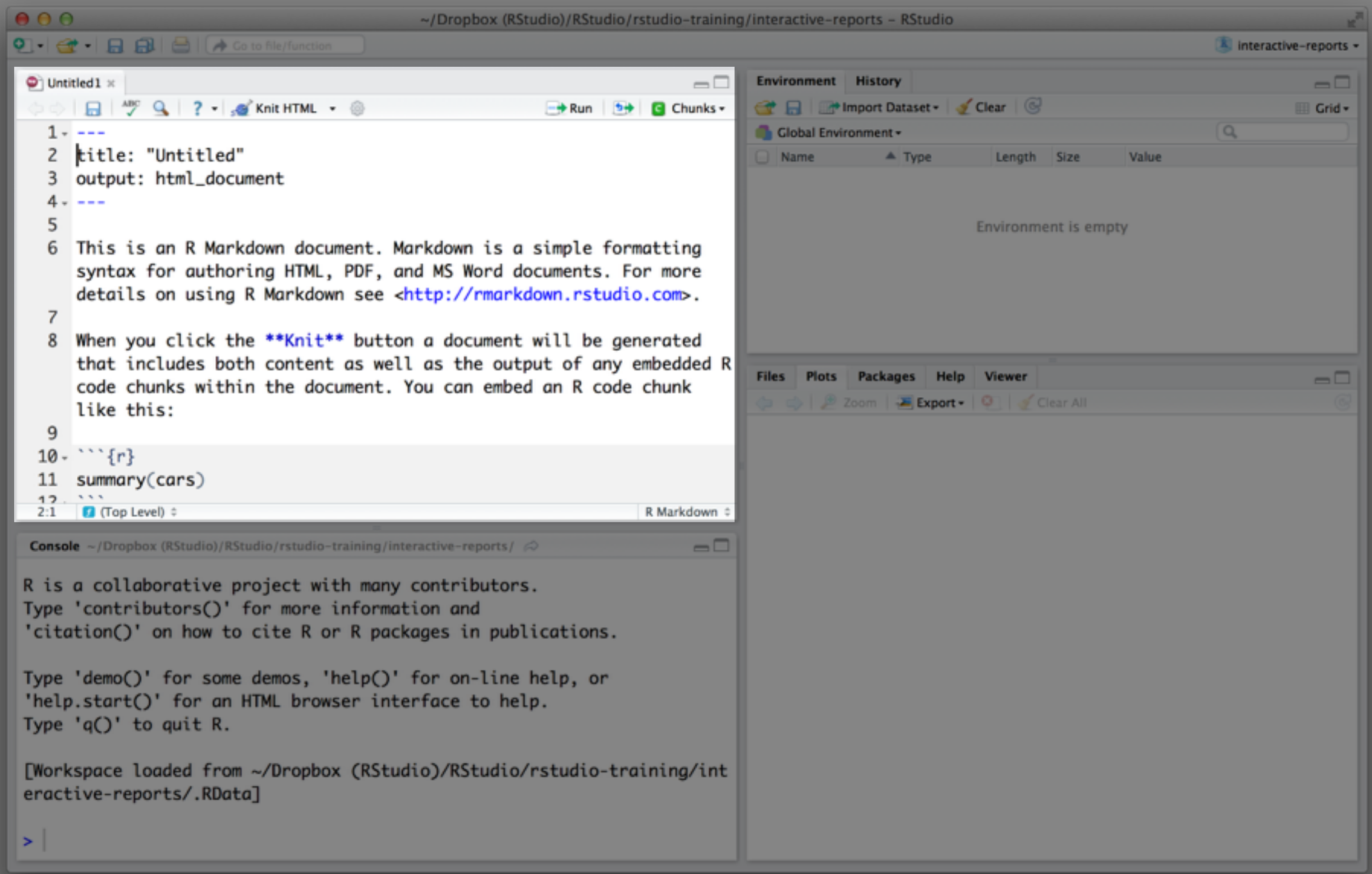
A dedicated pane for composing various types of documents

File > New File > ...



To open a basic R markdown document select "R Markdown" and then click OK in the pop up





Your turn

Do you remember what you said to your neighbor?

Open a .Rmd file. Erase everything in it. Then write down:

* your name

* your job

* 2 to 3 interesting things about yourself

Garrett Grolemund

Master Instructor

I wrote the lubridate R package

I'm the author of _Hands On Programming with R_.

I like to play the ukulele (but I'm not very good at it).

(Pandoc)

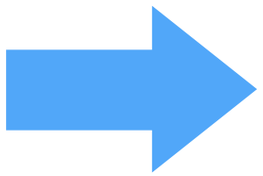
Markdown

Markdown

A simple way to write HTML.

Mostly plain english (like an email)

```
# Say Hello

To my little
friend, markdown.


Markdown is

* _easy_ to use

* __simple__

* fun?
```



Say Hello

To my little friend, markdown.
Markdown is

- *easy* to use
- **simple**
- fun?

Studio

Web sites that use markdown

* **GitHub** www.github.com

* **StackOverflow** www.stackoverflow.com

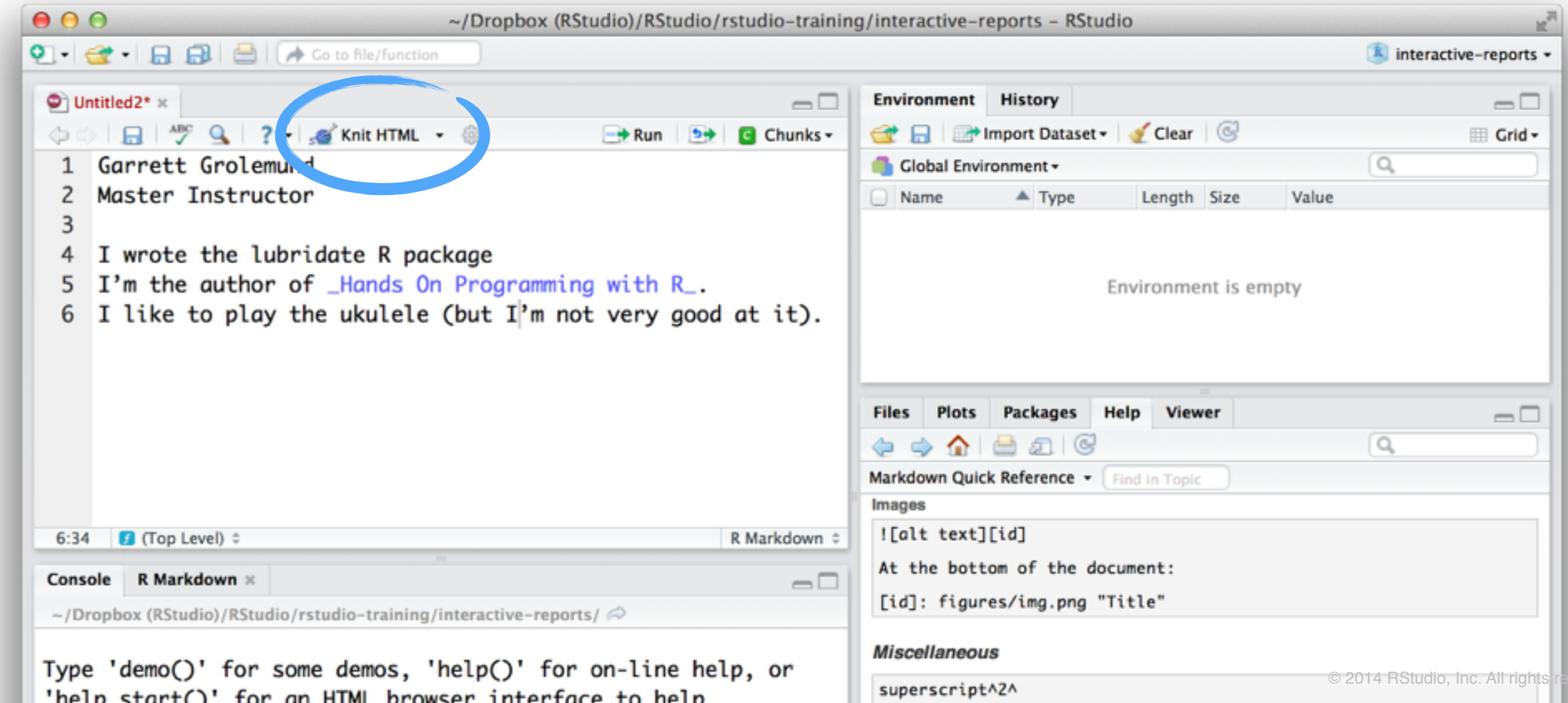
* **Reddit** www.reddit.com

* **Meteor** www.meteor.com

* many more

To compile markdown into HTML, click the "Knit HTML" button.

Note your file must have the extension .md or .Rmd

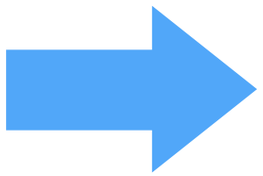


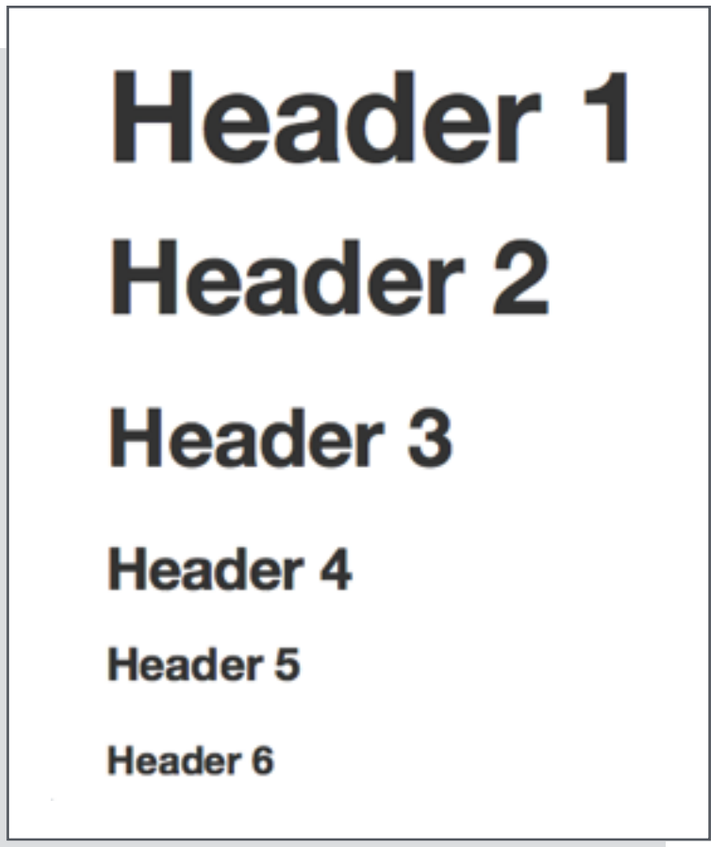
Basics

Use # to create headers.

Multiple #'s create lower level headers.

```
# Header 1

## Header 2

### Header 3

#### Header 4

##### Header 5

###### Header 6
```



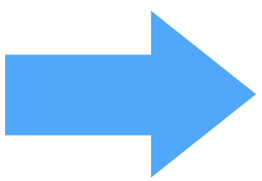


Basics

Text is rendered as plain text.
Surround words with underscores (_) to make italics.
Use two underscores (_ _) to make bold.
Use back tics to make code.

Add two spaces at the end of a line to start a new line

```
Text
_italics_
__bold__
`code`
```



Text

italics

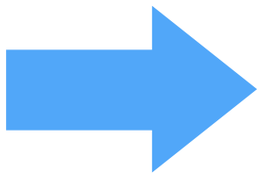
bold

`code`

Basics

Use asterisks to make bullet points.
Use numbers to make numbered lists.

```
Bullets

* bullet 1
* bullet 2

Numbered list

1. item 1
2. item 2
```



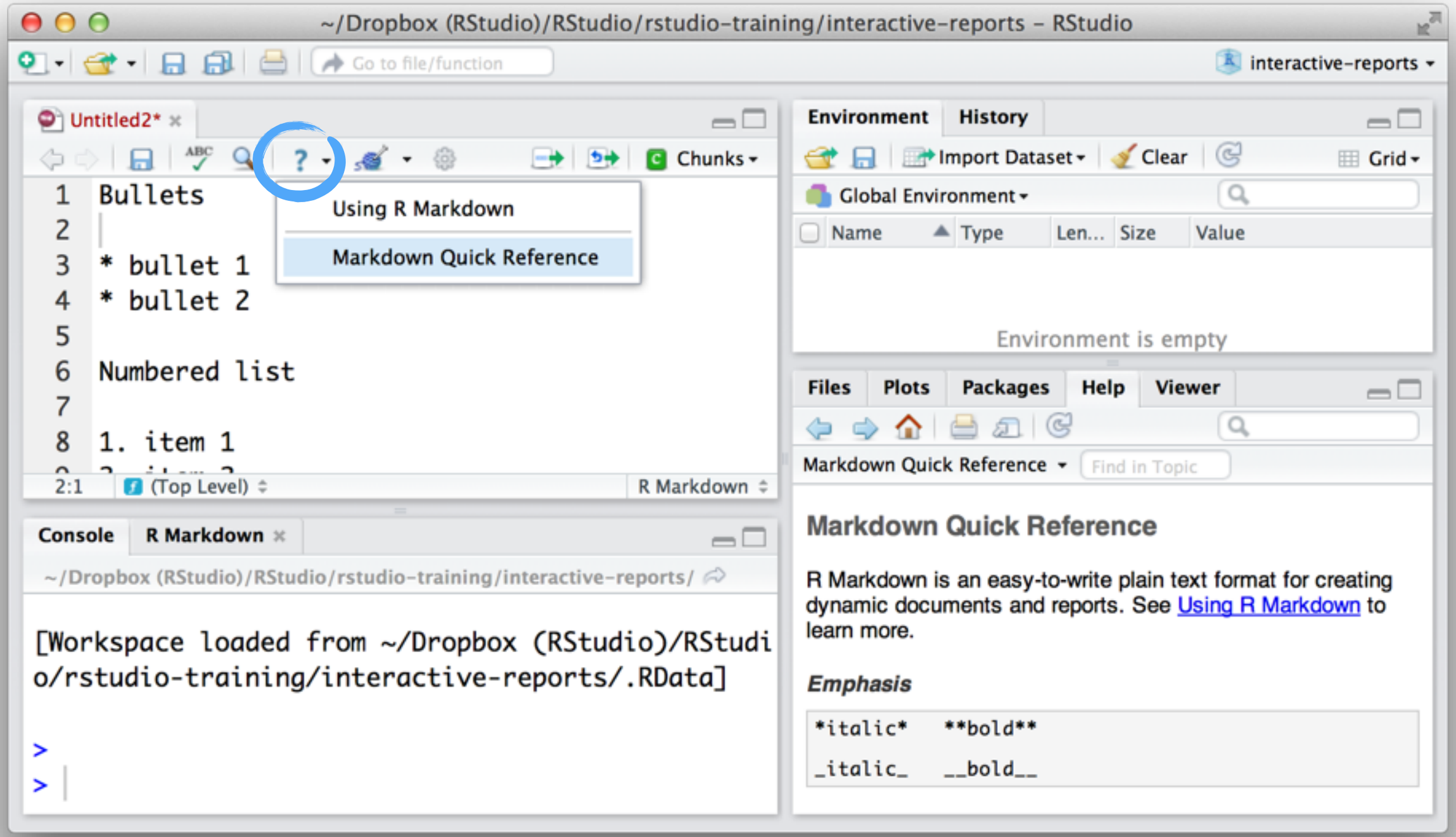
Bullets

- bullet 1
- bullet 2

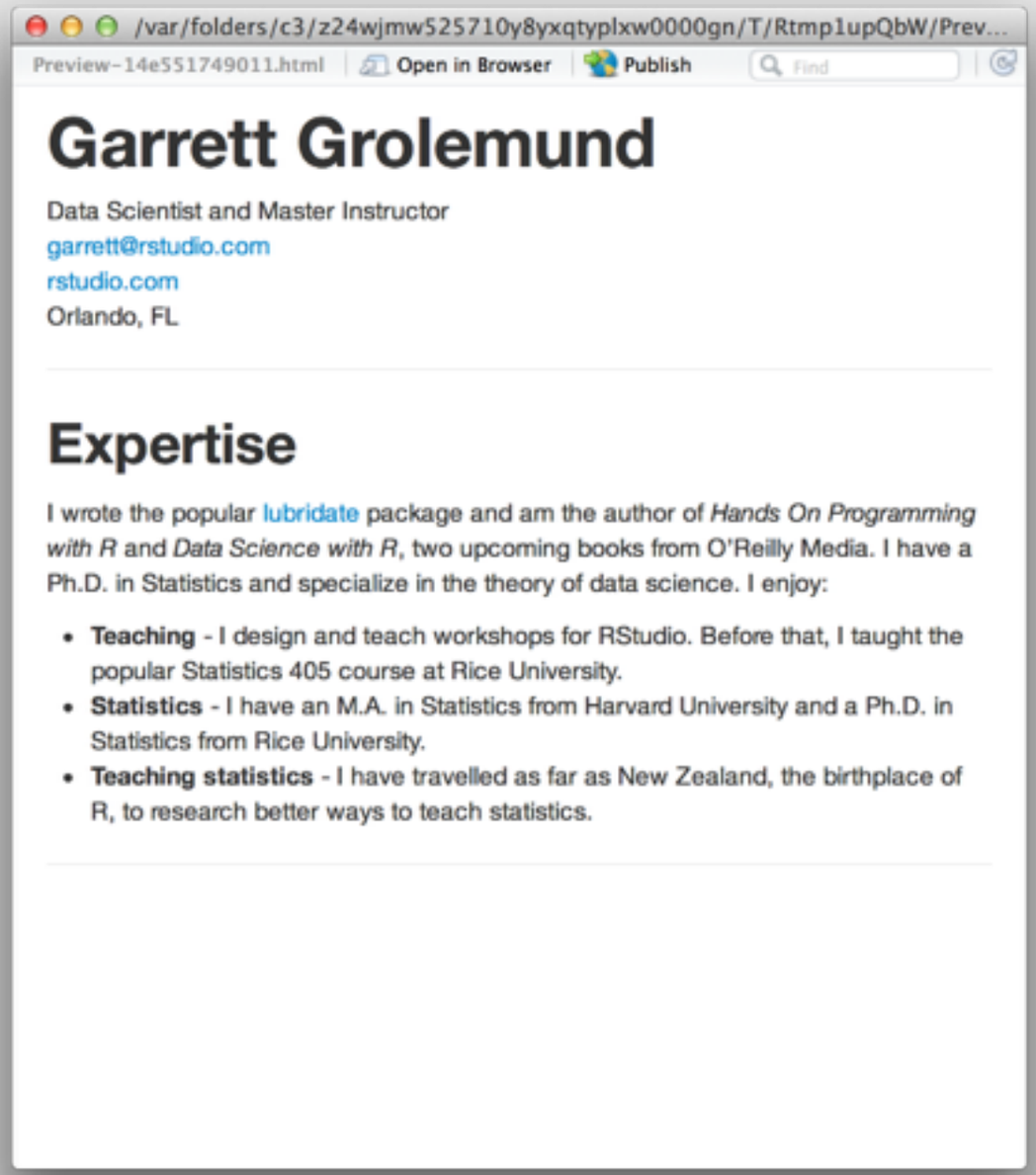
Numbered list

1. item 1
2. item 2

Markdown guide



Your turn



Turn your text into an HTML resume. Use:

1. headings
2. italics and bold
3. a list
4. hyperlinks (challenge)
5. horizontal rules (challenge)

Open the RStudio markdown guide if you need help.

Garrett Grolemund
Data Scientist and Master Instructor
garrett@rstudio.com
[rstudio.com](www.rstudio.com)
Orlando, FL

****

Expertise

I wrote the popular [lubridate](http://www.r-statistics.com/2012/03/do-more-with-dates-and-times-in-r-with-lubridate-1-1-0/) package and am the author of _Hands On Programming with R_ and _Data Science with R_, two upcoming books from O'Reilly Media. I have a Ph.D. in Statistics and specialize in the theory of data science. I enjoy:

* __Teaching__ - I design and teach workshops for RStudio. Before that, I taught the popular Statistics 405 course at Rice University.
* __Statistics__ - I have an M.A. in Statistics from Harvard University and a Ph.D. in Statistics from Rice University.
* __Teaching statistics__ - I have travelled as far as New Zealand, the birthplace of R, to research better ways to teach statistics.

****

```
# Garrett Grolemund
Data Scientist and Master Instructor
garrett@rstudio.com
[rstudio.com](www.rstudio.com)
Orlando, FL

****

## Expertise

I wrote the popular [lubridate](http://www.r-statistics.com/2012/03/do-more-with-dates-
and-times-in-r-with-lubridate-1-1-0/) package and am the author of _Hands On
Programming with R_ and _Data Science with R_, two upcoming books from O'Reilly Media.
I have a Ph.D. in Statistics and specialize in the theory of data science. I enjoy:


* __Teaching__ - I design and teach workshops for RStudio. Before that, I taught the
popular Statistics 405 course at Rice University.
* __Statistics__ - I have an M.A. in Statistics from Harvard University and a Ph.D. in
Statistics from Rice University.
* __Teaching statistics__ - I have travelled as far as New Zealand, the birthplace of
R, to research better ways to teach statistics.


****
```

Header

```
# Garrett Grolemund
Data Scientist and Master Instructor
garrett@rstudio.com
[rstudio.com](www.rstudio.com)
Orlando, FL

****

## Expertise

I wrote the popular [lubridate](http://www.r-statistics.com/2012/03/do-more-with-dates-and-times-in-r-with-lubridate-1-1-0/) package and am the author of _Hands On Programming with R_ and _Data Science with R_, two upcoming books from O'Reilly Media. I have a Ph.D. in Statistics and specialize in the theory of data science. I enjoy:


* __Teaching__ - I design and teach workshops for RStudio. Before that, I taught the popular Statistics 405 course at Rice University.
* __Statistics__ - I have an M.A. in Statistics from Harvard University and a Ph.D. in Statistics from Rice University.
* __Teaching statistics__ - I have travelled as far as New Zealand, the birthplace of R, to research better ways to teach statistics.


****
```

```
# Garrett Grolemund
Data Scientist and Master Instructor
garrett@rstudio.com
[rstudio.com](www.rstudio.com)
Orlando, FL

****

## Expertise

I wrote the popular [lubridate](http://www.r-statistics.com/2012/03/do-more-with-dates-
and-times-in-r-with-lubridate-1-1-0/) package and am the author of _Hands On
Programming with R_ and _Data Science with R_, two upcoming books from O'Reilly Media.
I have a Ph.D. in Statistics and specialize in the theory of data science. I enjoy:

* __Teaching__ - I design and teach workshops for RStudio. Before that, I taught the
popular Statistics 405 course at Rice University.
* __Statistics__ - I have an M.A. in Statistics from Harvard University and a Ph.D. in
Statistics from Rice University.
* __Teaching statistics__ - I have travelled as far as New Zealand, the birthplace of
R, to research better ways to teach statistics.

****
```

bold

Garrett Grolemund
Data Scientist and Master Instructor
garrett@rstudio.com
[rstudio.com](www.rstudio.com)
Orlando, FL

****

Expertise

I wrote the popular [lubridate](http://www.r-statistics.com/2012/03/do-more-with-dates-and-times-in-r-with-lubridate-1-1-0/) package and am the author of _Hands On Programming with R_ and _Data Science with R_, two upcoming books from O'Reilly Media. I have a Ph.D. in Statistics and specialize in the theory of data science. I enjoy:

list

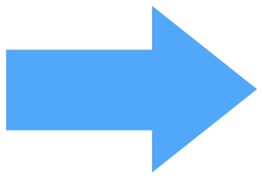
* __Teaching__ - I design and teach workshops for RStudio. Before that, I taught the popular Statistics 405 course at Rice University.
* __Statistics__ - I have an M.A. in Statistics from Harvard University and a Ph.D. in Statistics from Rice University.
* __Teaching statistics__ - I have travelled as far as New Zealand, the birthplace of R, to research better ways to teach statistics.

****

Hyperlinks

Use brackets to denote a link. Place the URL in parentheses.

```
This is a

[link](www.git.com).
```



This is a link.

Garrett Grolemund
Data Scientist and Master Instructor
garrett@rstudio.com
[rstudio.com](www.rstudio.com)
Orlando, FL

****

Expertise

I wrote the popular [lubridate](http://www.r-statistics.com/2012/03/do-more-with-dates-and-times-in-r-with-lubridate-1-1-0/) package and am the author of _Hands On Programming with R_ and _Data Science with R_, two upcoming books from O'Reilly Media. I have a Ph.D. in Statistics and specialize in the theory of data science. I enjoy:

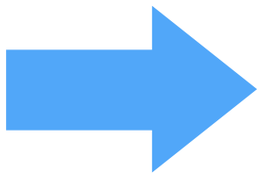
* __Teaching__ - I design and teach workshops for RStudio. Before that, I taught the popular Statistics 405 course at Rice University.
* __Statistics__ - I have an M.A. in Statistics from Harvard University and a Ph.D. in Statistics from Rice University.
* __Teaching statistics__ - I have travelled as far as New Zealand, the birthplace of R, to research better ways to teach statistics.

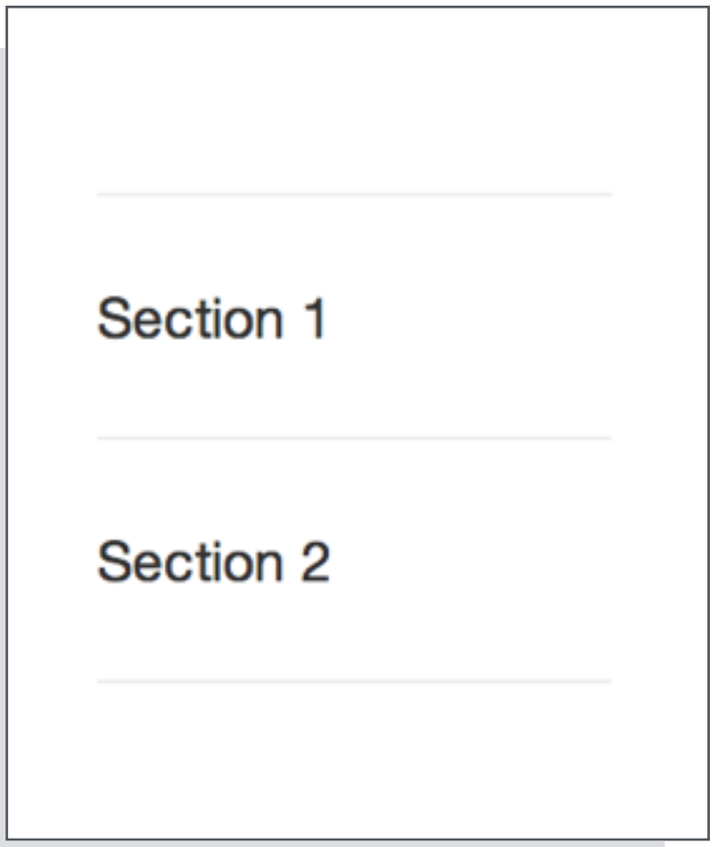
****

Horizontal Rule

Use three or more asterisks to make a horizontal rule (a horizontal dividing line).

```
****

Section 1

****

Section 2

****
```





```
# Garrett Grolemund
Data Scientist and Master Instructor
garrett@rstudio.com
[rstudio.com](www.rstudio.com)
Orlando, FL

****
```

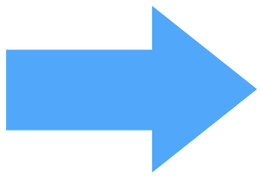
Horizontal Rule

```
## Expertise

I wrote the popular [lubridate](http://www.r-statistics.com/2012/03/do-more-with-dates-and-times-in-r-with-lubridate-1-1-0/) package and am the author of _Hands On Programming with R_ and _Data Science with R_, two upcoming books from O'Reilly Media. I have a Ph.D. in Statistics and specialize in the theory of data science. I enjoy:


* __Teaching__ - I design and teach workshops for RStudio. Before that, I taught the popular Statistics 405 course at Rice University.
* __Statistics__ - I have an M.A. in Statistics from Harvard University and a Ph.D. in Statistics from Rice University.
* __Teaching statistics__ - I have travelled as far as New Zealand, the birthplace of R, to research better ways to teach statistics.


****
```

Equations

Write equations with latex math commands and surround them in $'s.

```
According to
Einstein,
$E=mc^{2}$
```

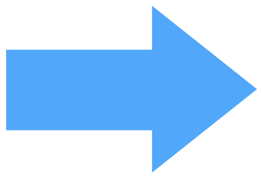


According to Einstein, $E = mc^2$

Equations

Use two $'s to make centered equation blocks.

```
According to
Einstein,


$$E=mc^{2}$$
```



According to
Einstein,

$$E = mc^2$$

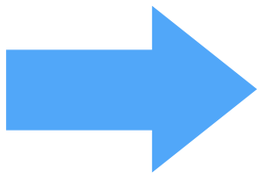
Images

Use a link preceded by an ! to insert an image.

The link text should be

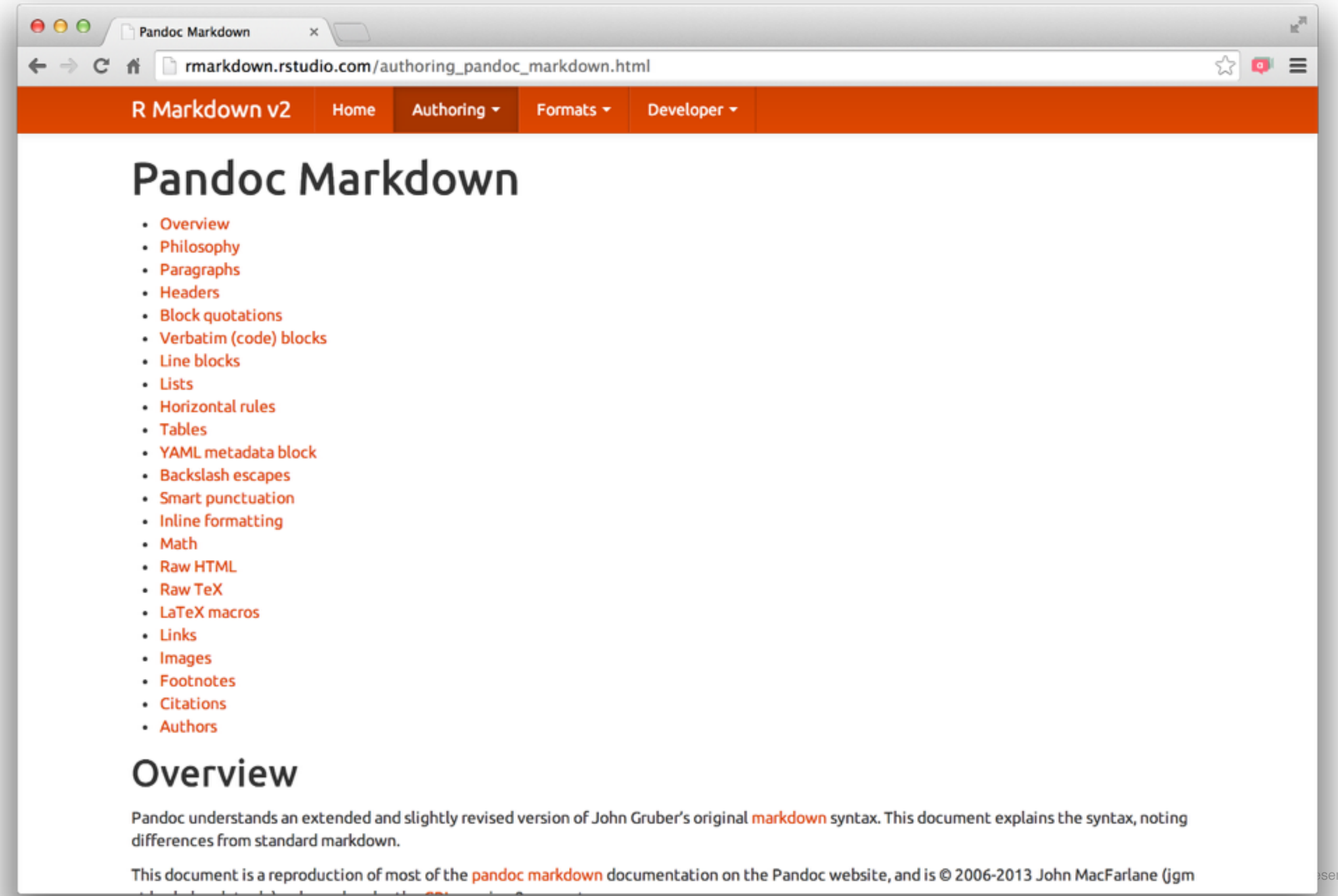
- *a URL (if the image is hosted online)*
- *a file path (if the image is saved as a file)*

```
![](images/1.png)
The RStudio logo.
```



Note: You must save your .Rmd file before the preview will find the image

rmarkdown.rstudio.com



Embed Code

babynames

Popularity of baby names in USA since 1880. From SSA.

1,792,091 records

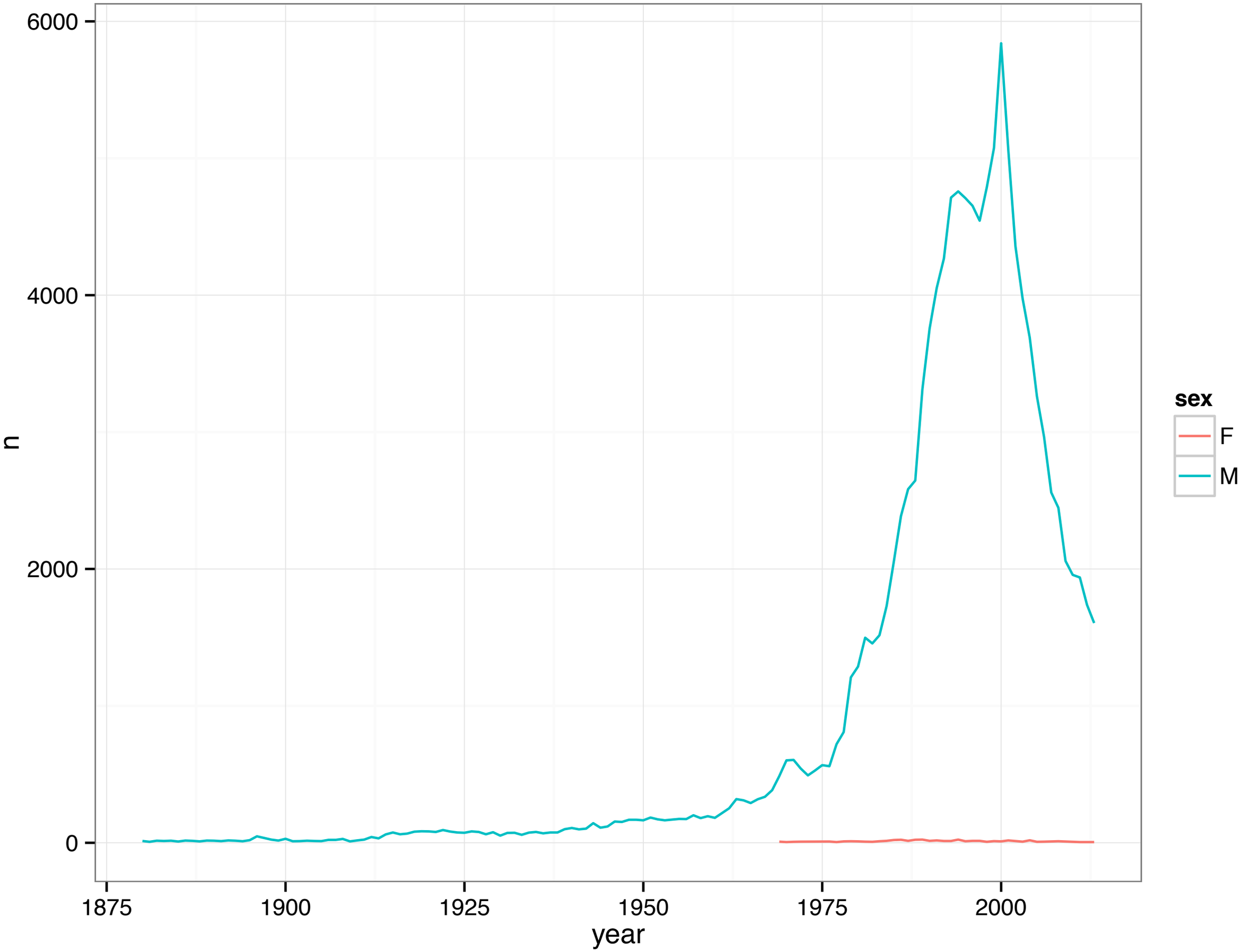
Five variables: year, sex, name, n, and prop.

```
# devtools::install_github("babynames", "hadley")
library(babynames)
```

Warm up

Load the `babynames` package, and graph the history of your name.

```
# devtools::install_github("babynames", "hadley")
library(babynames)
library(ggplot2)

garrett <- subset(babynames, name == "Garrett")
qplot(year, n, data = garrett, geom ="line",
  color = sex) + theme_bw()
```



Garrett Grolemund

Data Scientist and Master Instructor
garrett@rstudio.com
rstudio.com
Orlando, FL

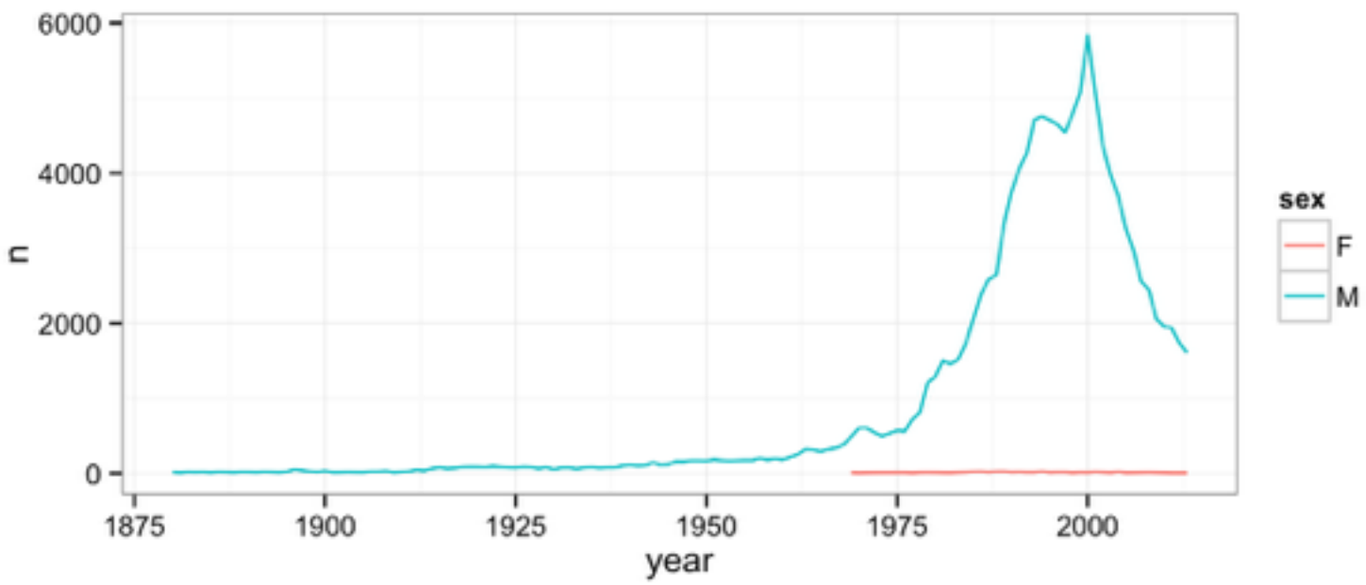
Expertise

I wrote the popular lubridate package and am the author of *Hands On Programming with R* and *Data Science with R*, two upcoming books from O'Reilly Media. I have a Ph.D. in Statistics and specialize in the theory of data science. I enjoy:

- **Teaching** - I design and teach workshops for RStudio. Before that, I taught the popular Statistics 405 course at Rice University.
- **Statistics** - I have an M.A. in Statistics from Harvard University and a Ph.D. in Statistics from Rice University.
- **Teaching statistics** - I have travelled as far as New Zealand, the birthplace of R, to research better ways to teach statistics.

What's in a name?

Here is a graph of the number of people named Garrett.



My name was more popular than ever in 2000. So far, 125065 people have been named Garrett.

Code chunks

Insert a chunk of R code with

````
```{r}
# some code

```
````

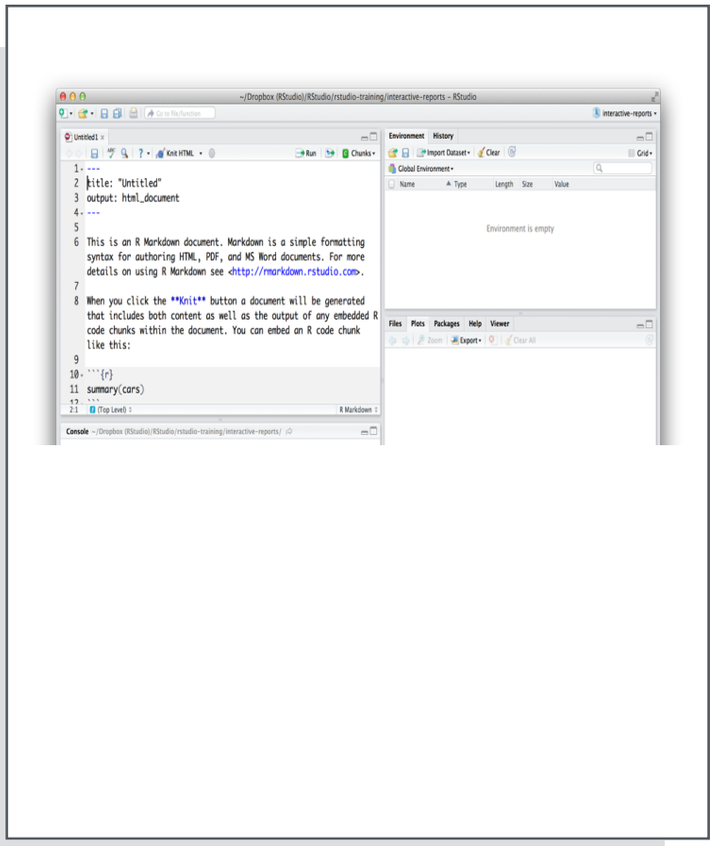
When you compile, R markdown will run the code and include its results. R markdown will also remove the ```` ```{r} ```` and ```` ``` ````.

eval

Change the display by adding arguments in the brackets after `r`.

`eval = FALSE` prevents the code from being run.

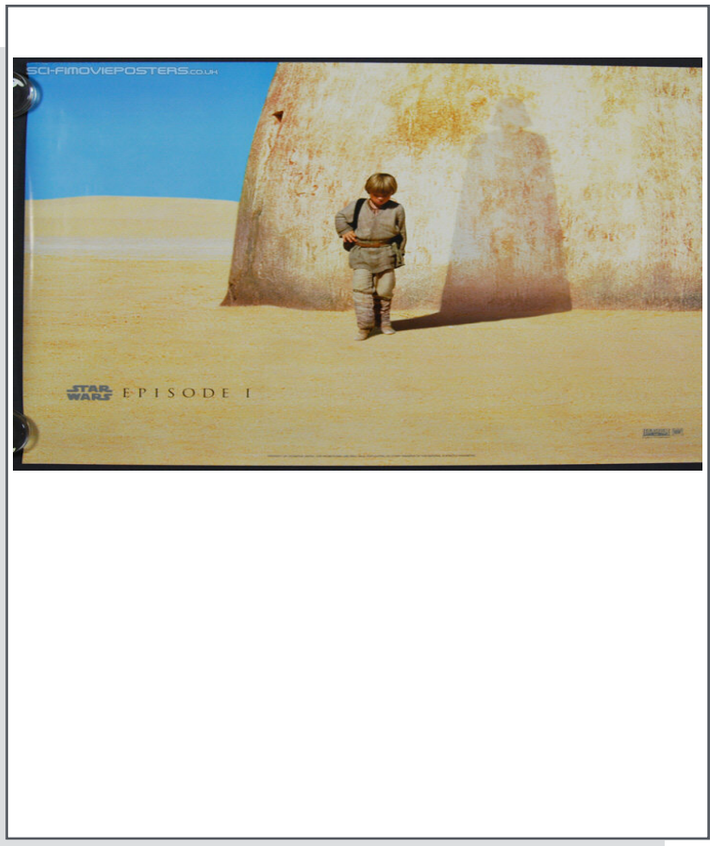
````
Here's some code

```{r eval=FALSE}
dim(iris)
```
````



echo

`echo = FALSE` displays the results, but not the code.

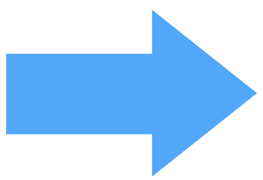
````
Here's some code

```{r echo=FALSE}
dim(iris)
```
````

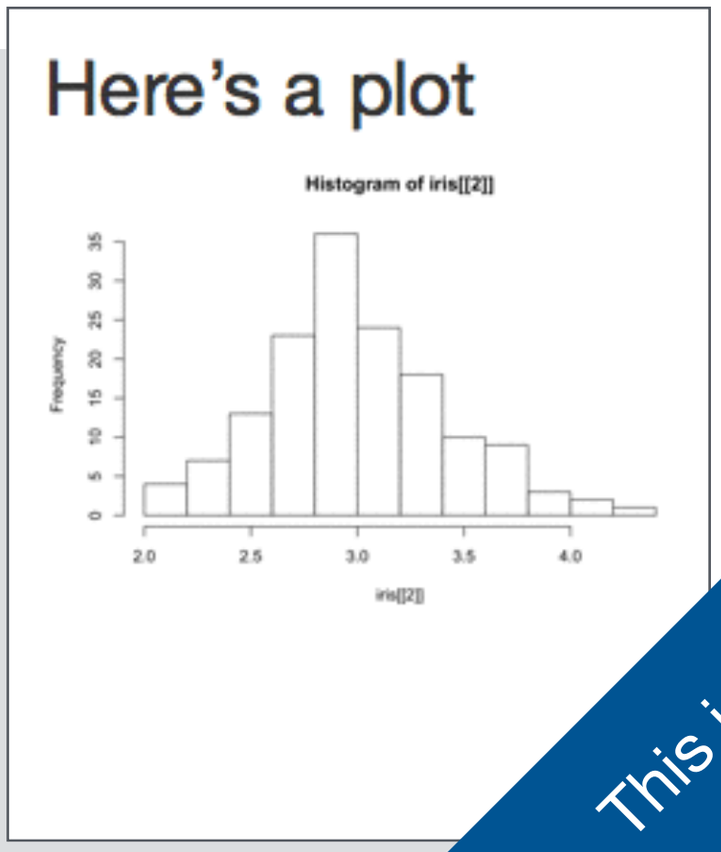


echo

`echo = FALSE` displays the results, but not the code.

````
Here's a plot

```{r echo=FALSE}
hist(iris[[2]])
```
````



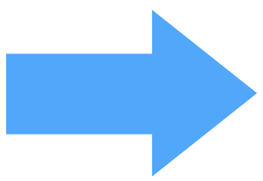


This is very useful for plots

inline code

Place code in a sentence with `` `r #code` ``. R Markdown will replace the code with its results.

```
Two plus two
equals `r 2 + 2`.
```

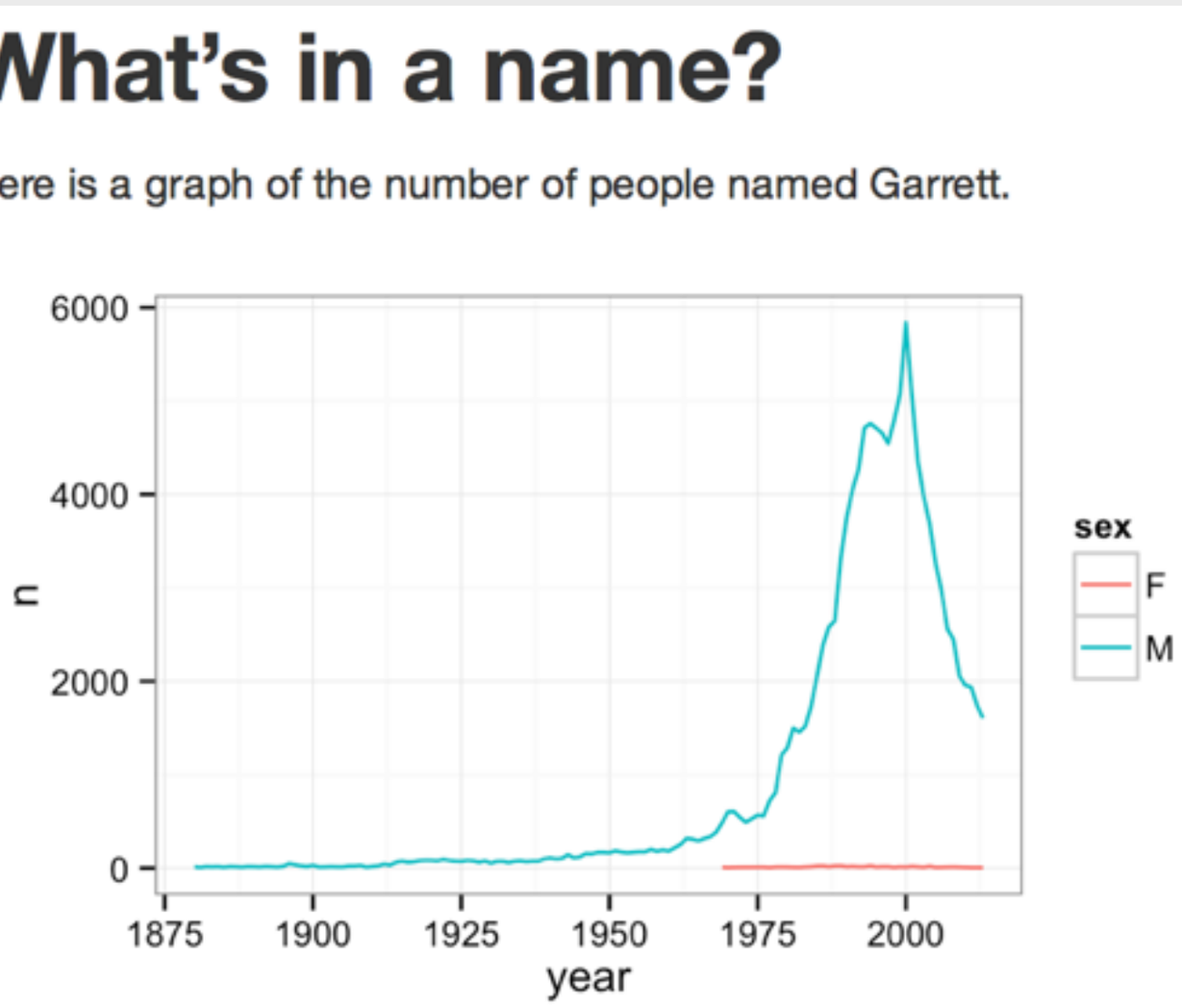


Your turn

Add a third section that describes your name.

What's in a name?

Here is a graph of the number of people named Garrett.



My name was more popular than ever in 2000. So far, 125065 people have been named Garrett.

```
library(babynames)
library(ggplot2)
garrett$year[which.max(garrett$n)]
sum(garrett$n)
```

…

What's in a name?

Here is a graph of the number of people named Garrett.

```{r echo=FALSE, fig.height=3, fig.width=5}
library(babynames)
library(ggplot2)

garrett <- subset(babynames, name == "Garrett")
qplot(year, n, data = garrett, geom ="line", color = sex) +
theme_bw()
```

My name was more popular than ever in `r garrett $year[which.max(garrett$n)]`. So far, `r sum(garrett$n)` people have been named Garrett.

…

What's in a name?

Here is a graph of the number of people named Garrett.

```{r echo=FALSE, fig.height=3, fig.width=5}
library(babynames)
library(ggplot2)

garrett <- subset(babynames, name == "Garrett")
qplot(year, n, data = garrett, geom ="line", color = sex) +
theme_bw()
```

Code chunk embeds plot

My name was more popular than ever in `r garrett$year[which.max(garrett$n)]`. So far, `r sum(garrett$n)` people have been named Garrett.

…

What's in a name?

Here is a graph of the number of people named Garrett.

```{r echo=FALSE, fig.height=3, fig.width=5}
library(babynames)
library(ggplot2)

garrett <- subset(babynames, name == "Garrett")
qplot(year, n, data = garrett, geom ="line", color = sex) +
theme_bw()
```

inline code

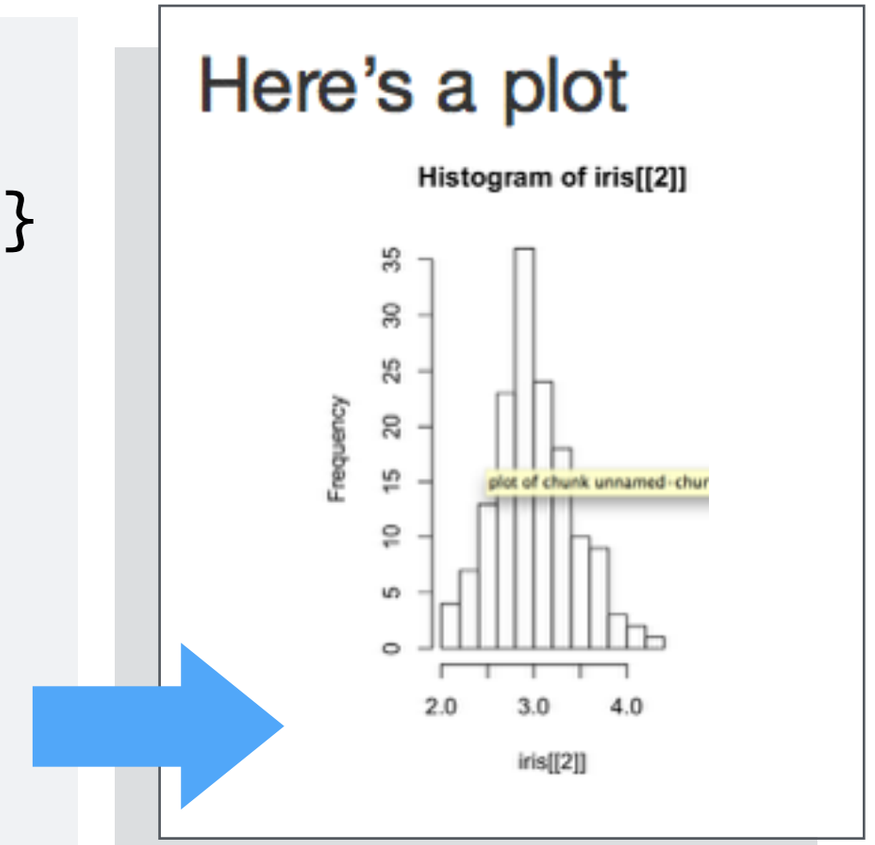
My name was more popular than ever in `r garrett$year[which.max(garrett$n)]`. So far, `r sum(garrett$n)` people have been named Garrett.

fig.width, fig.height

Specify the dimension of plots (in inches) with `fig.width` and `fig.height`.

Separate multiple arguments with commas.

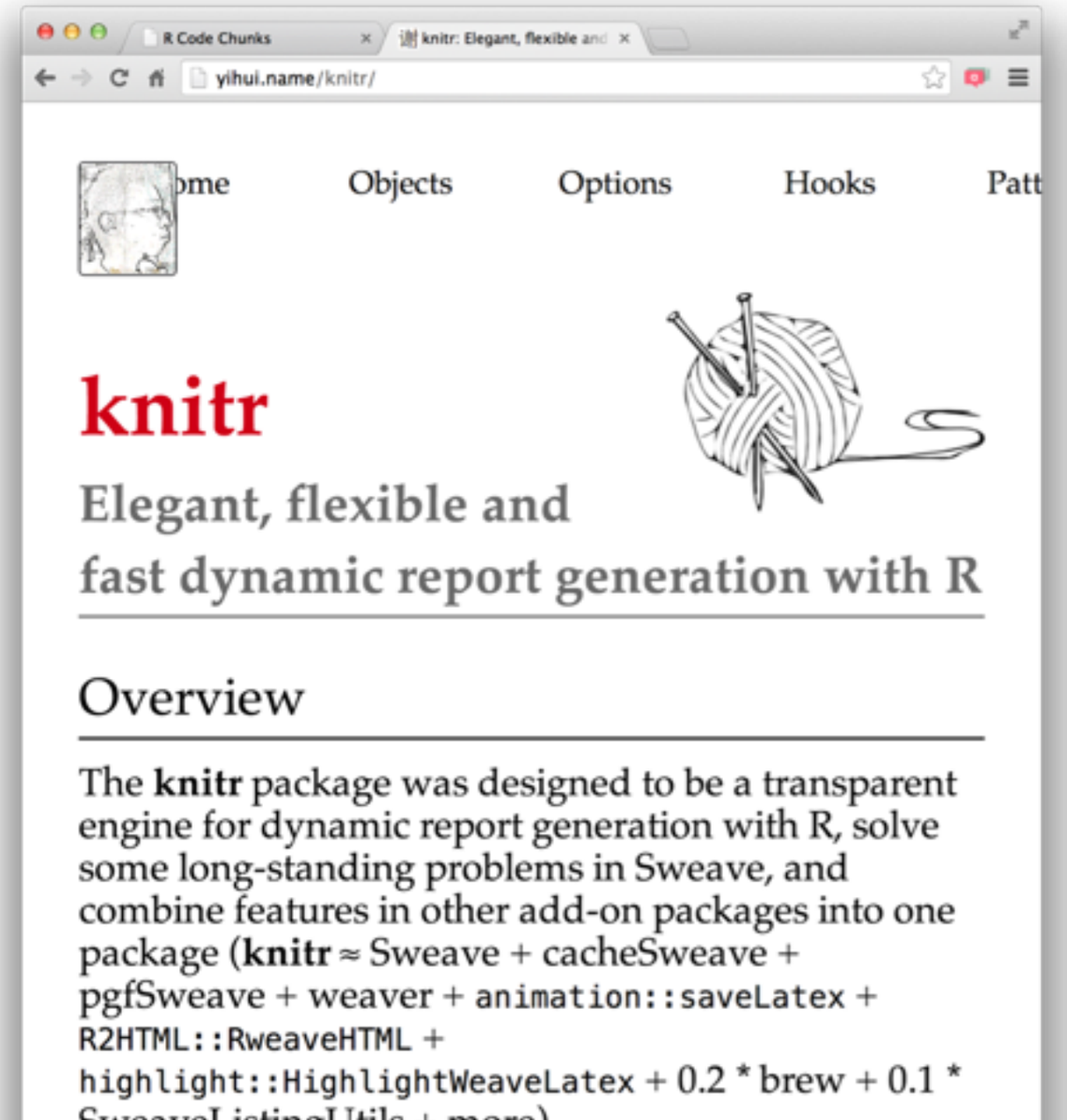
````
Here's a plot

```{r echo=FALSE, fig.width=1, fig.height=2}
hist(iris[[2]])
```
````

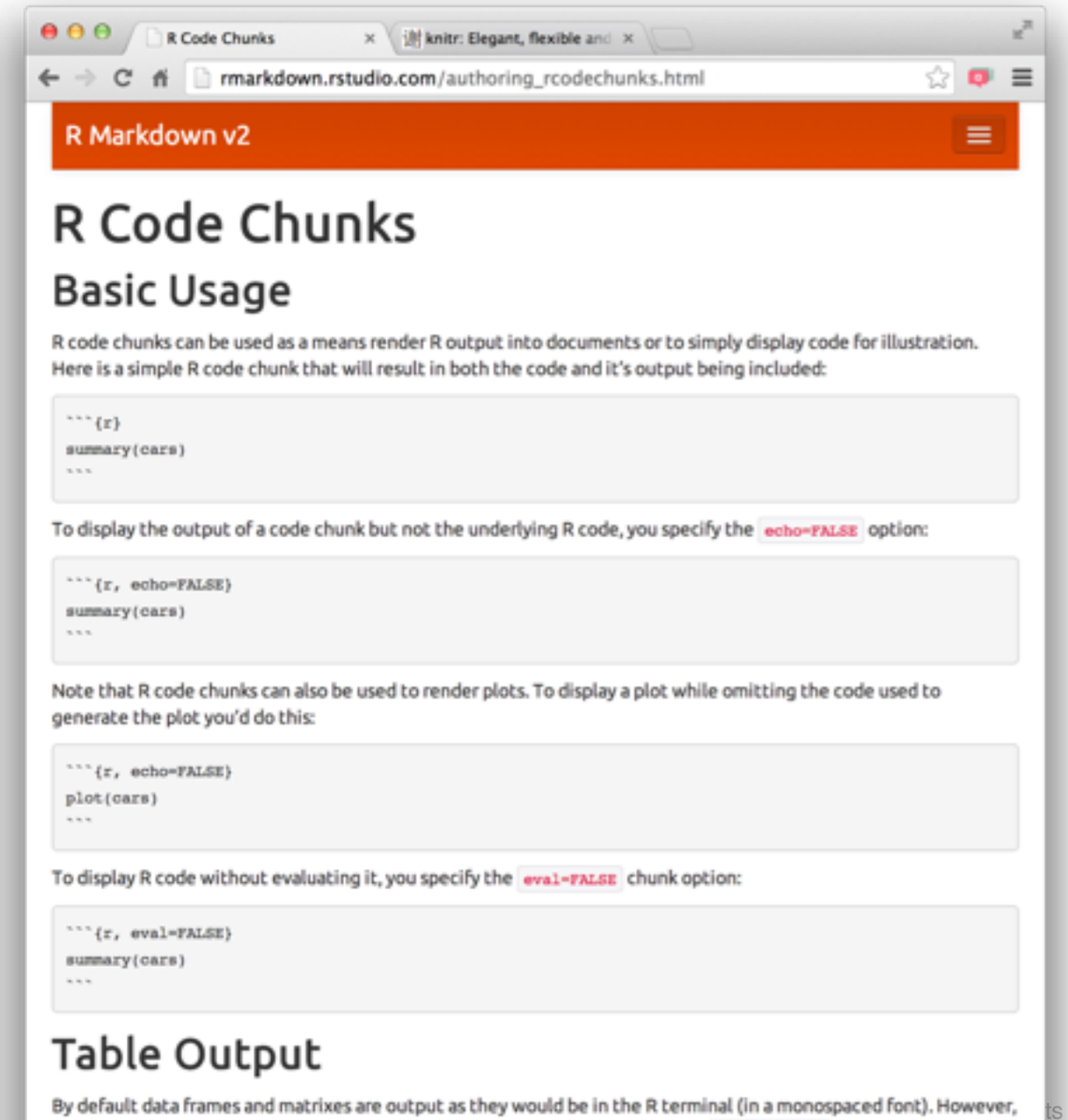


For more on chunk options

yihui.name/knitr/

rmarkdown.rstudio.com





packrat

(Reproducible Research)

packrat

http://rstudio.github.io/packrat/

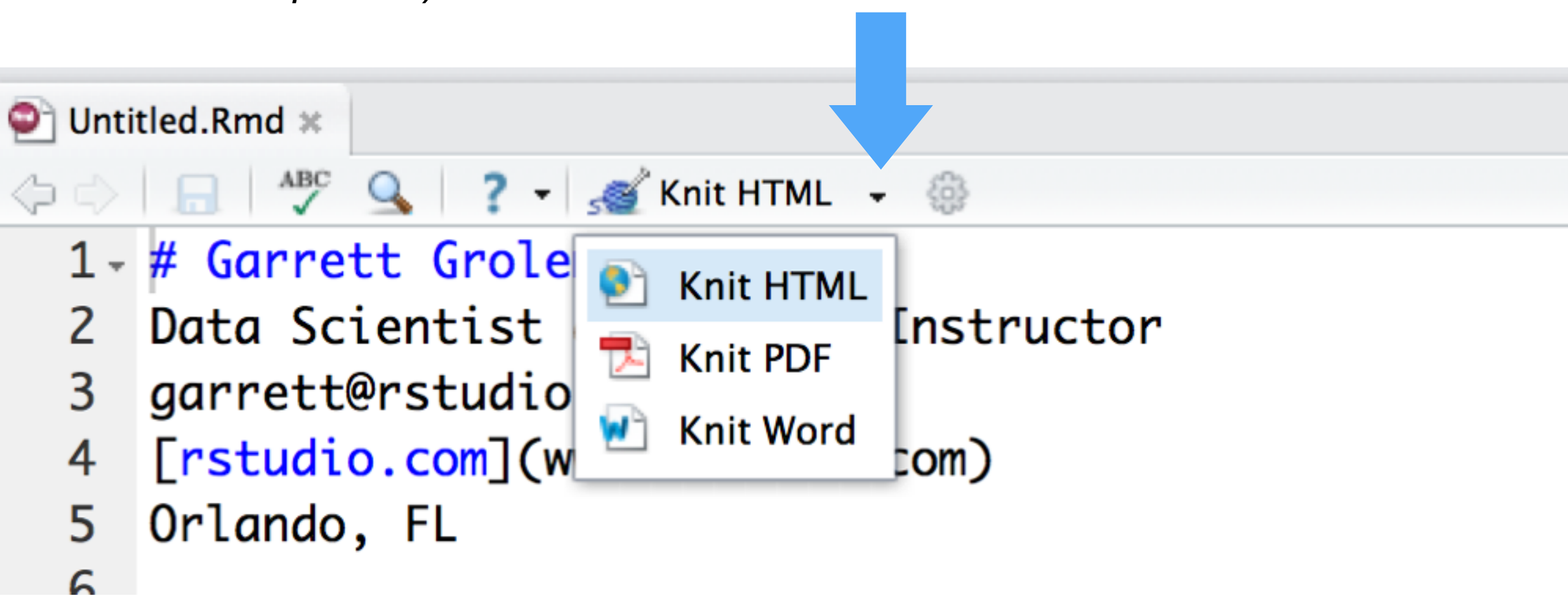
Packrat is a dependency management system for R. Use packrat to make your R projects more:

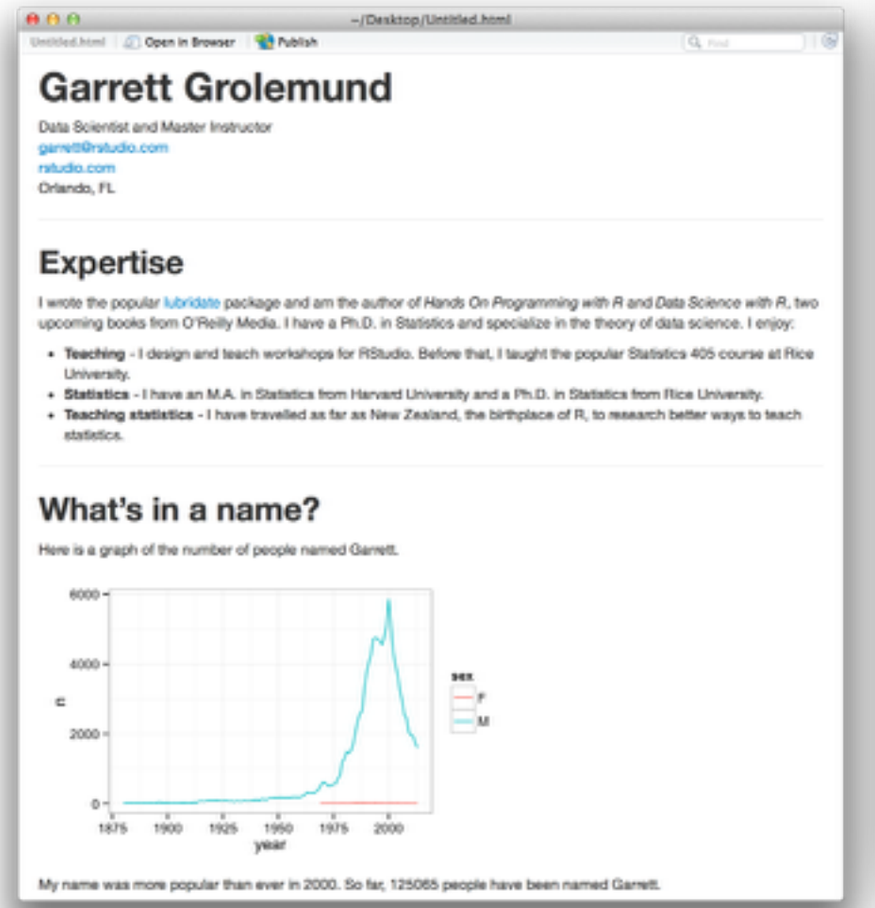
- **Isolated**: Installing a new or updated package for one project won't break your other projects, and vice versa. That's because packrat gives each project its own private package library.
- **Portable**: Easily transport your projects from one computer to another, even across different platforms. Packrat makes it easy to install the packages your project depends on.
- **Reproducible**: Packrat records the exact package versions you depend on, and ensures those exact versions are the ones that get installed wherever you go.

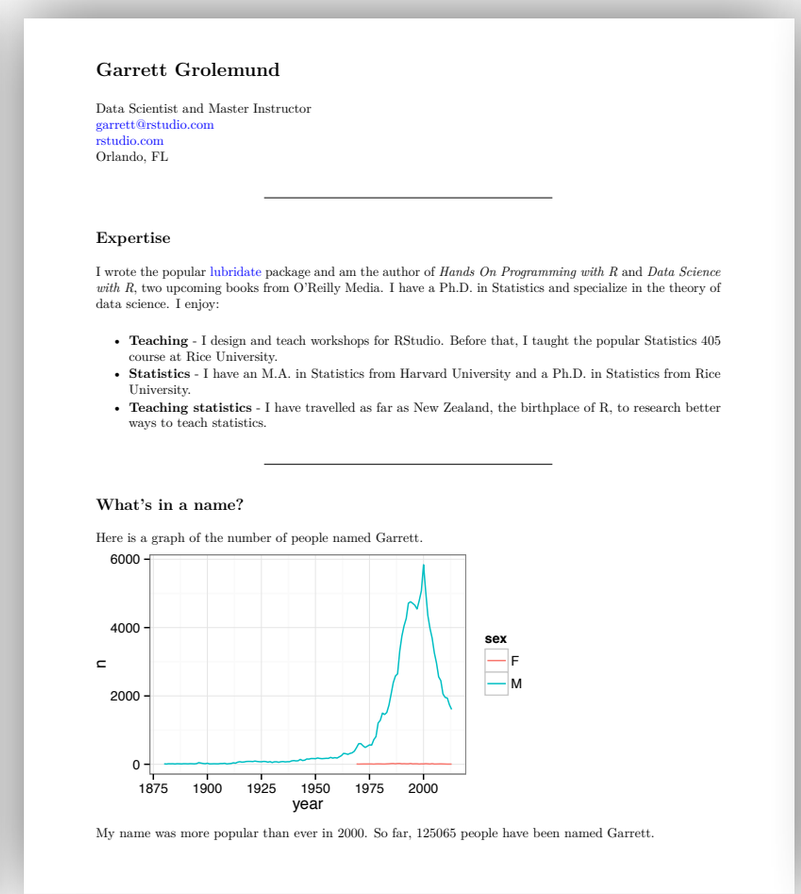
Multiple output formats

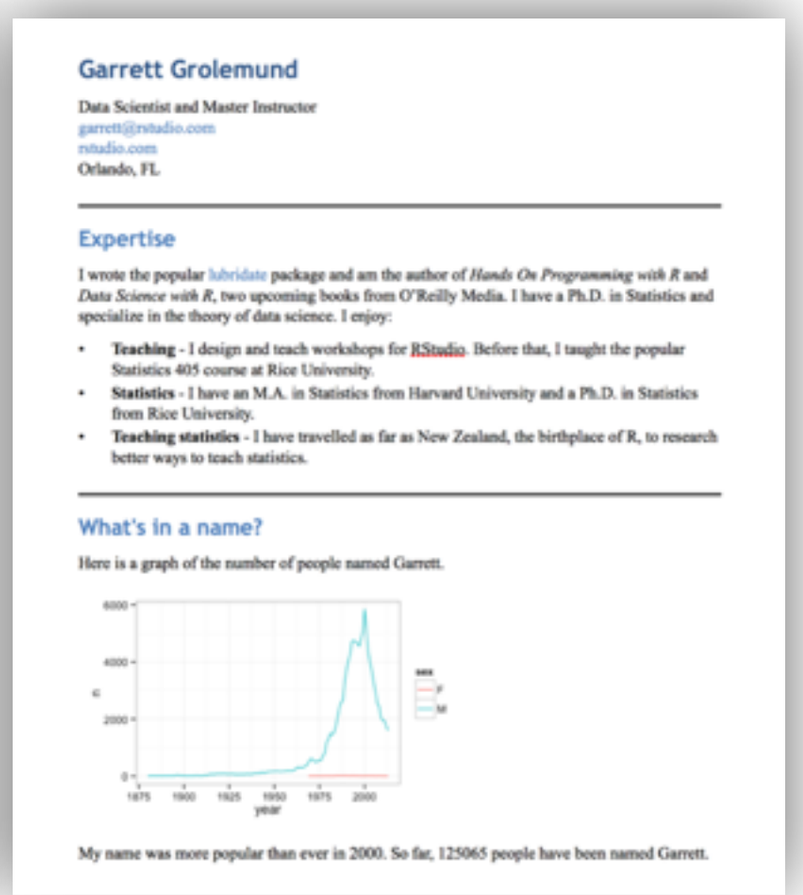
R Markdown can generate three preset file formats from a .Rmd file

- HTML
- PDF *(Must have latex installed on your computer)*
- MS Word *(Must have Word installed on your computer)*









HTML

PDF

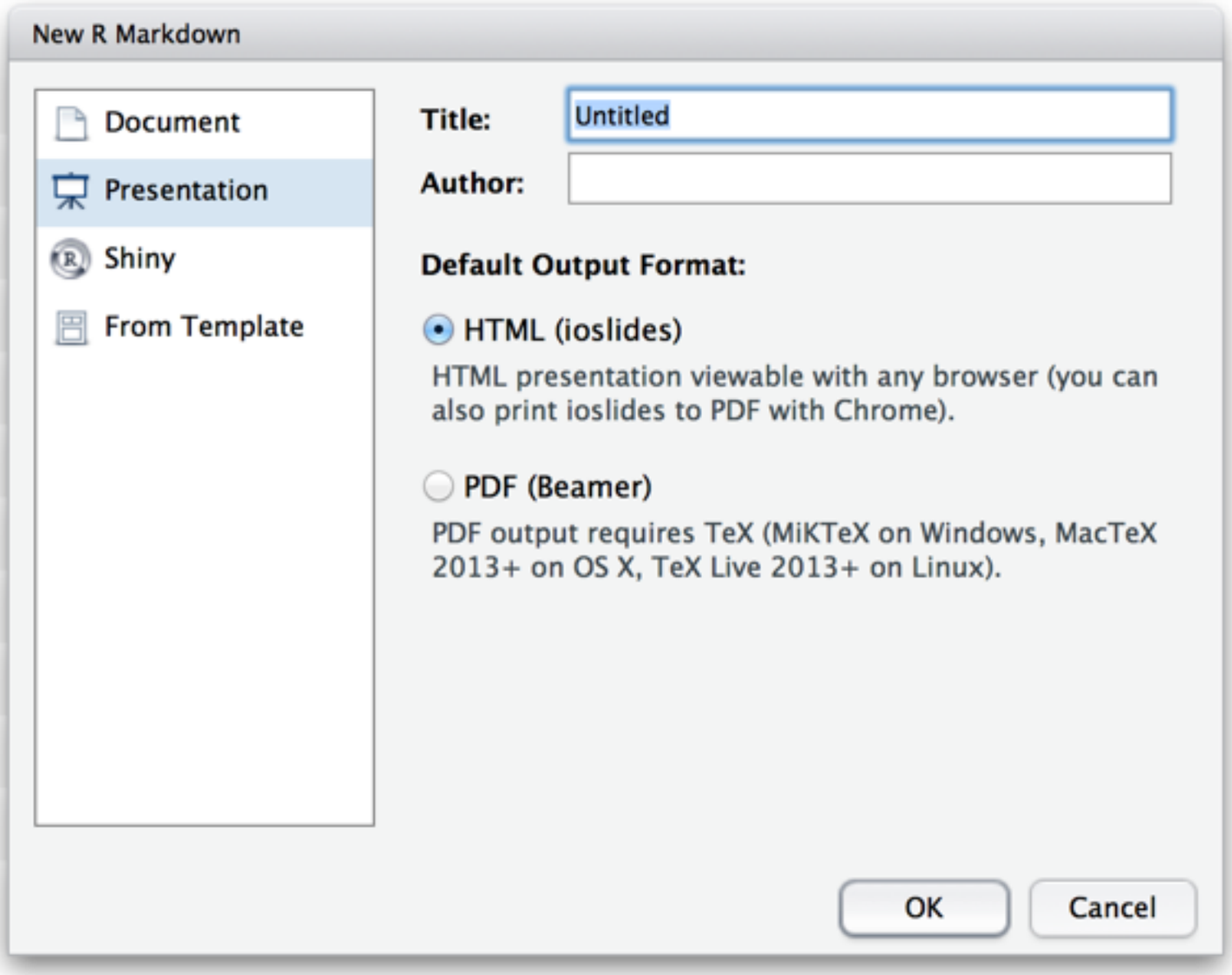
** requires tex*

MS Word

** requires "Word"*

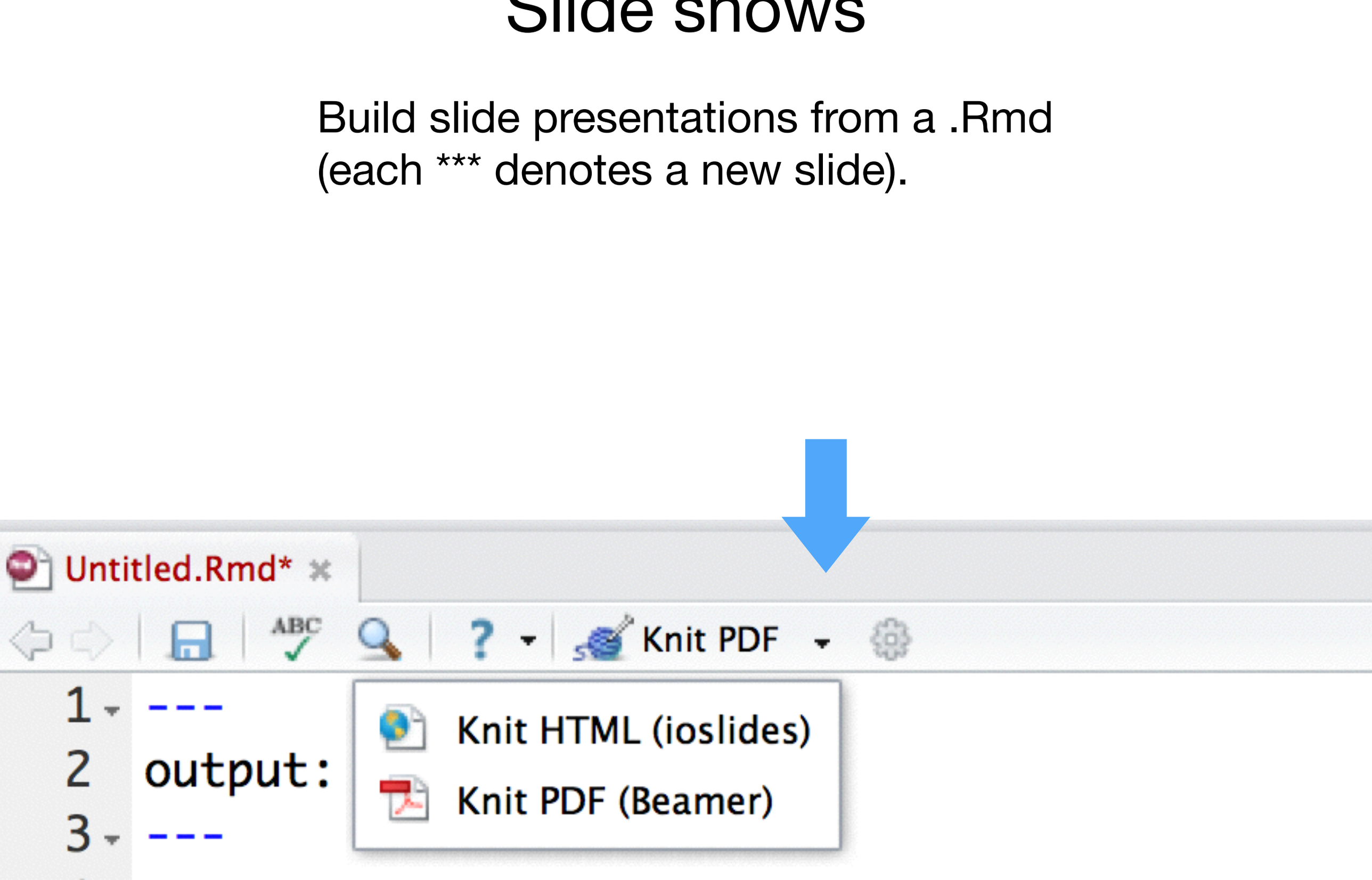
Slide shows

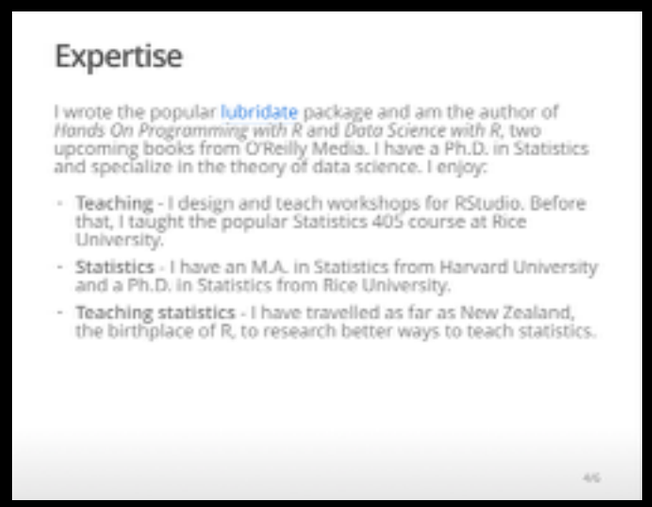
Build slide presentations from a .Rmd (each *** denotes a new slide).

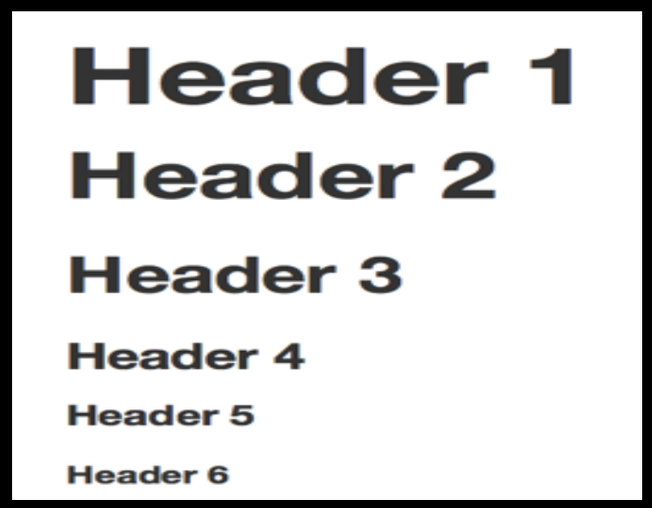


Slide shows

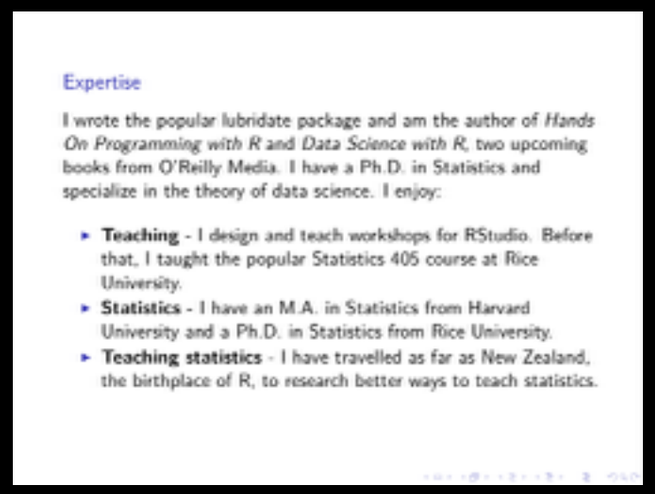
Build slide presentations from a .Rmd (each *** denotes a new slide).

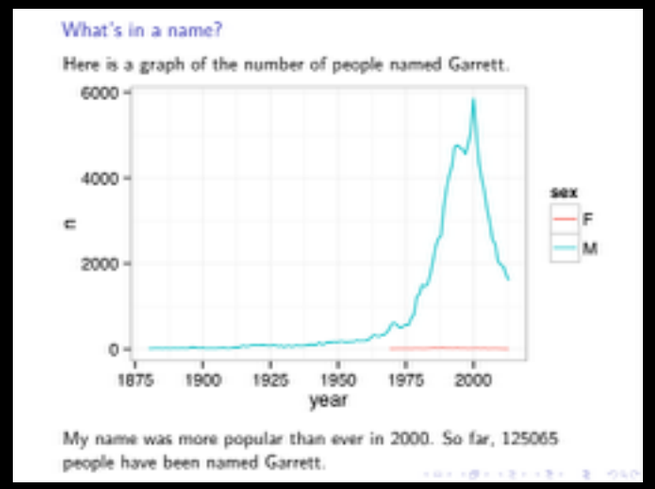






ioslides
(HTML)

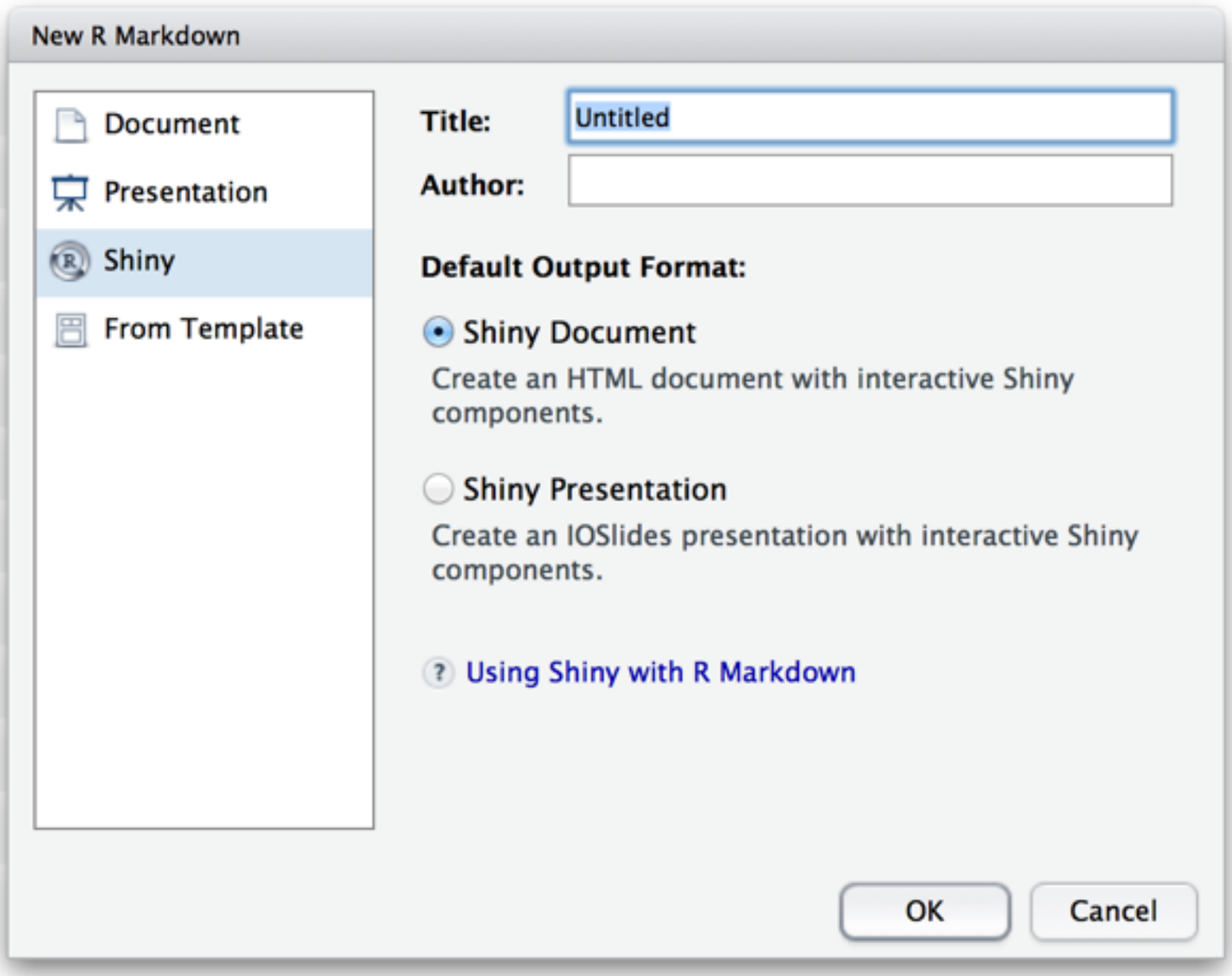




Beamer
(PDF)

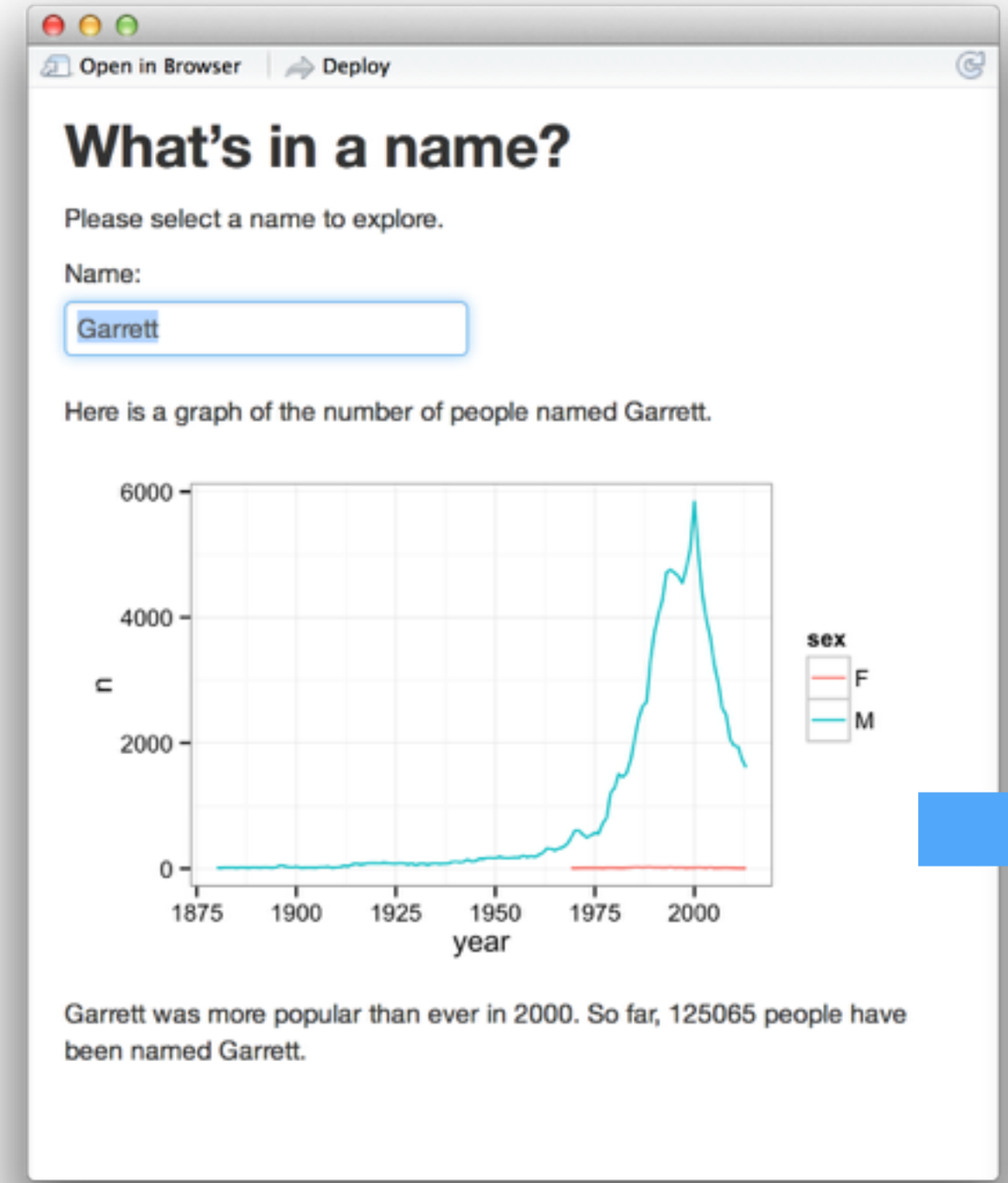
Interactive Documents

Select Shiny to add live Shiny elements to code chunks



Shiny

Goal: make graphs and text *interactive*



Shiny

An R package that does two things

1. Creates reactive R objects
2. Builds HTML web pages

Shiny

An R package that does two things

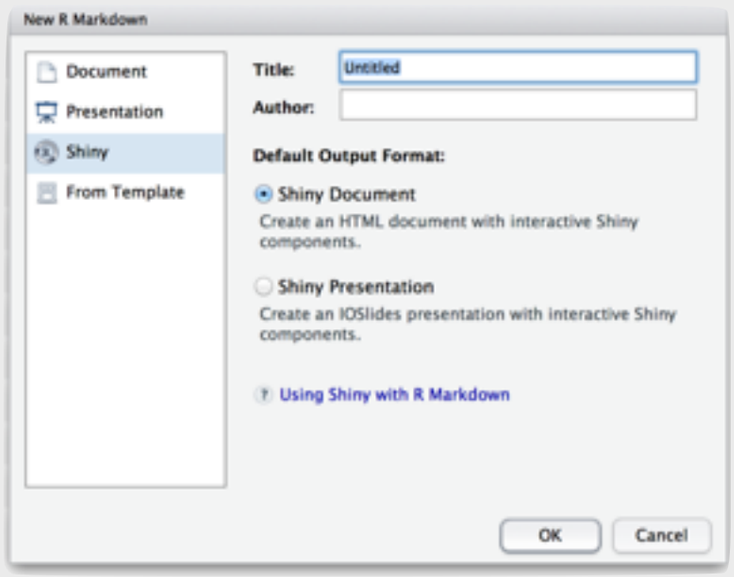
1. **Creates reactive R objects**
2. Builds HTML web pages

Get Ready

Open an interactive document:

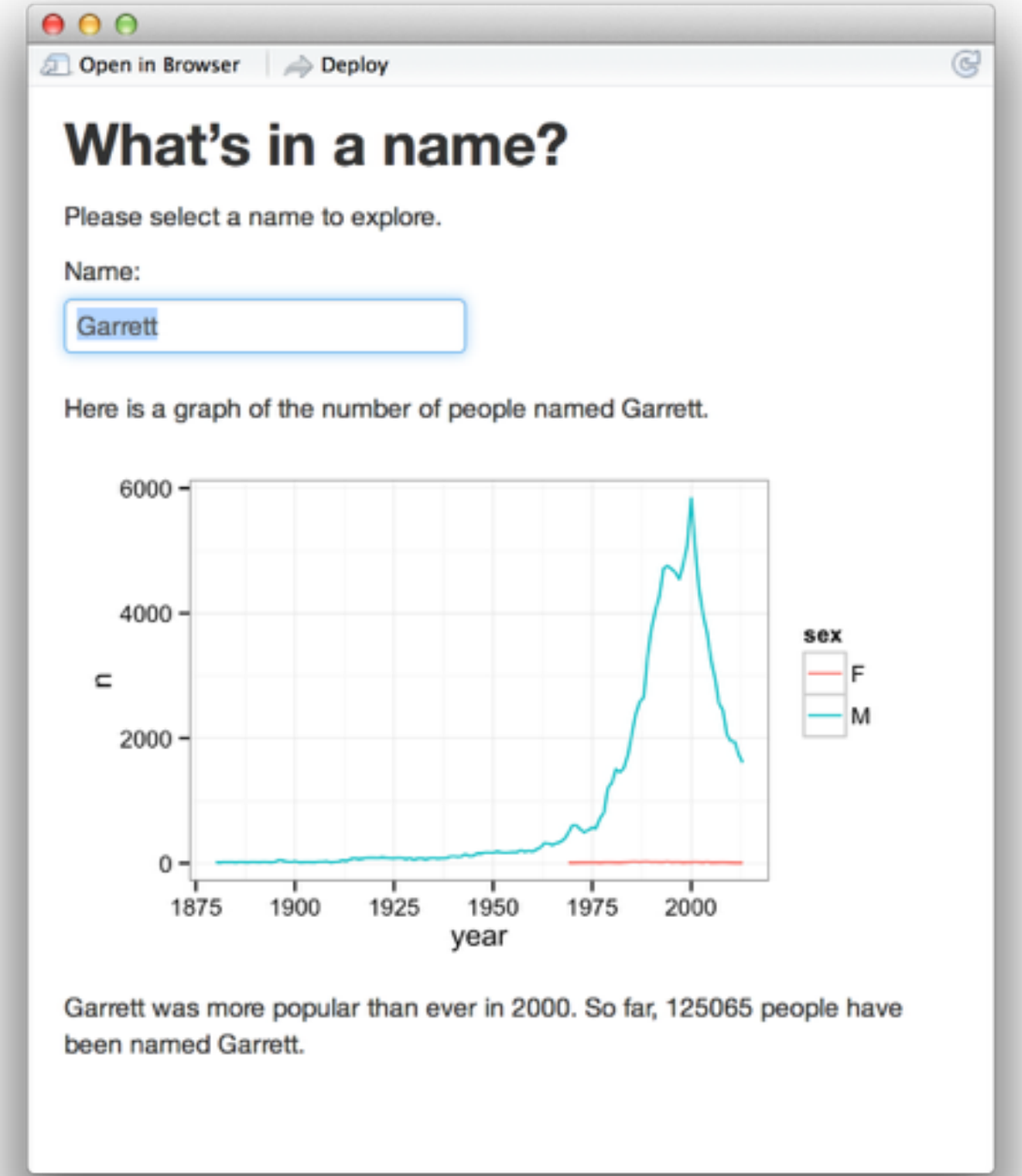
1. File > New File > R Markdown…

2. Select Shiny, then Shiny Document (default)



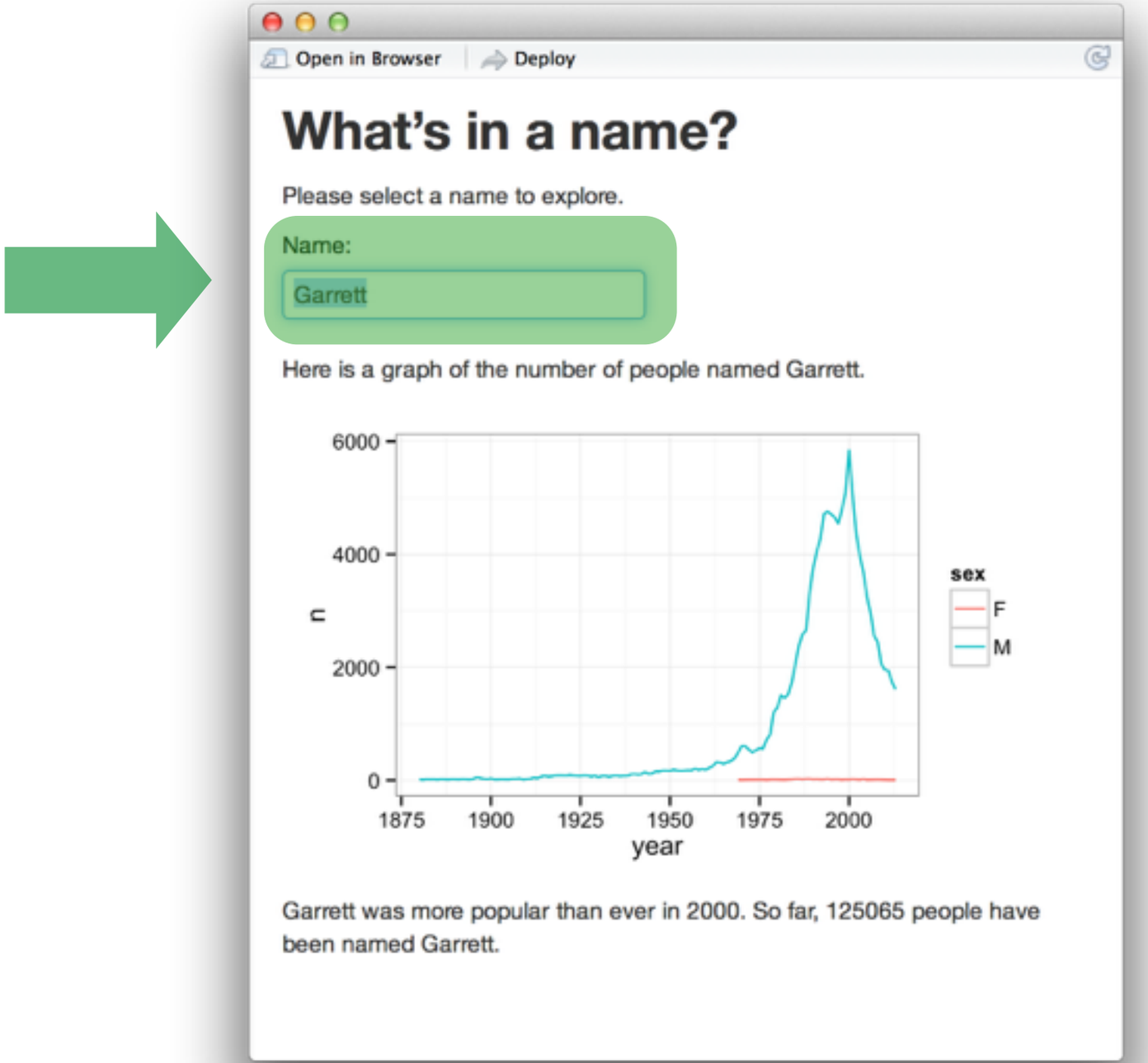
3. Delete everything under the second — — —

Types of reactive R objects



Types of reactive R objects

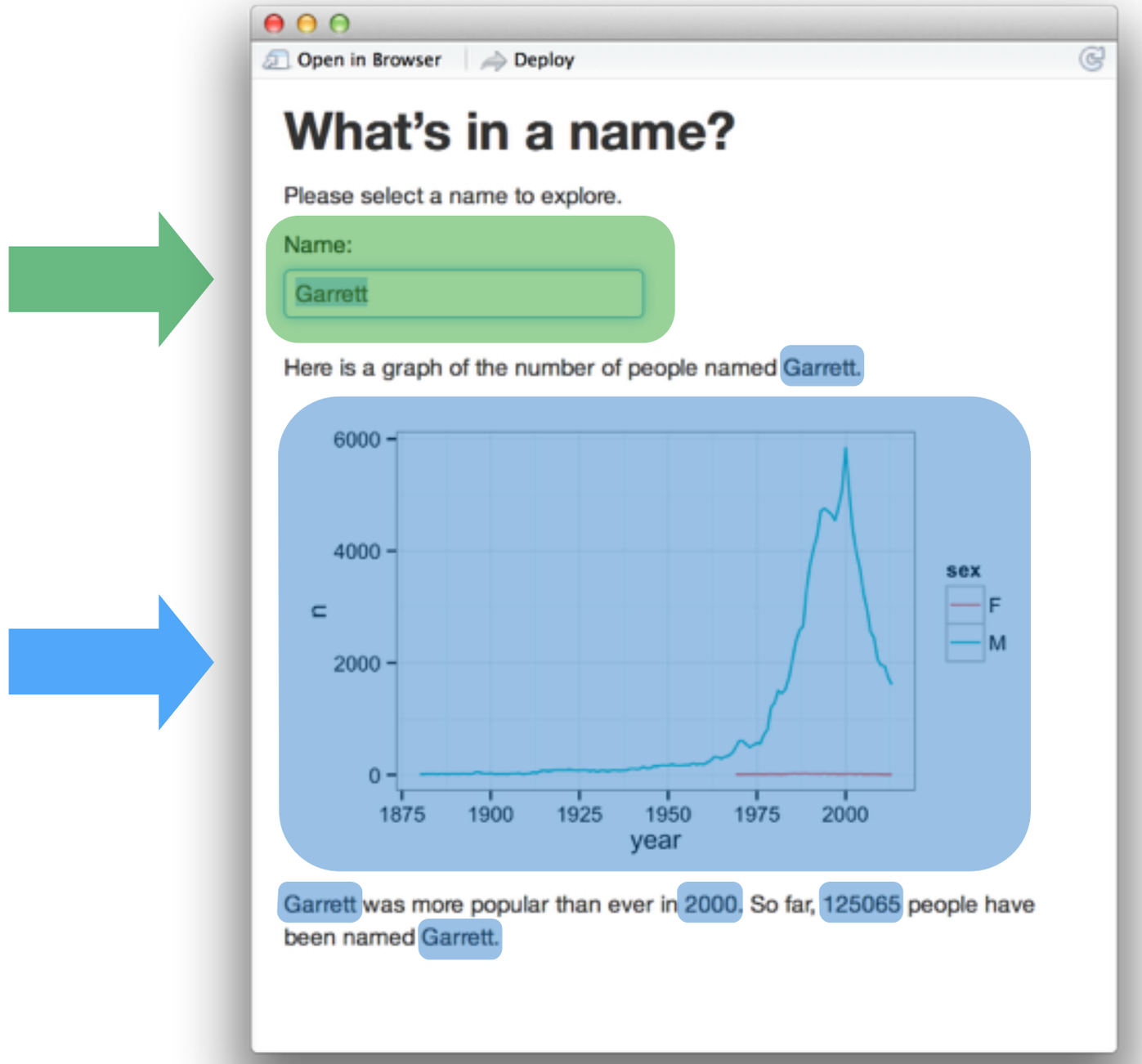
Widgets - let users set a value by typing, clicking, etc.



Types of reactive R objects

Widgets - let users set a value by typing, clicking, etc.

Rendered - respond whenever a widget value changes



Open in Browser | Deploy

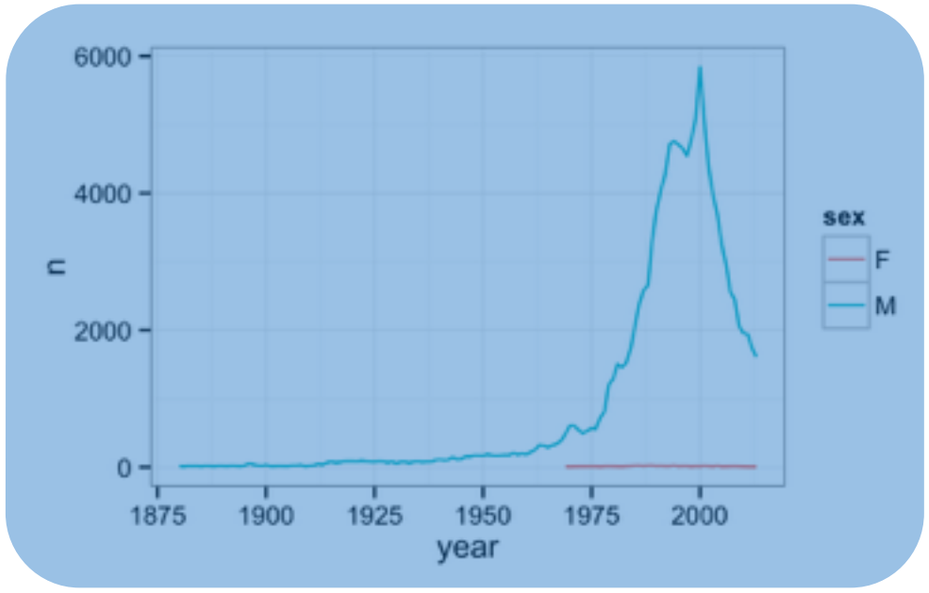
What's in a name?

Please select a name to explore.

Name:

Garrett

Here is a graph of the number of people named Garrett.



Garrett was more popular than ever in 2000. So far, 125065 people have been named Garrett.

Open in Browser | Deploy

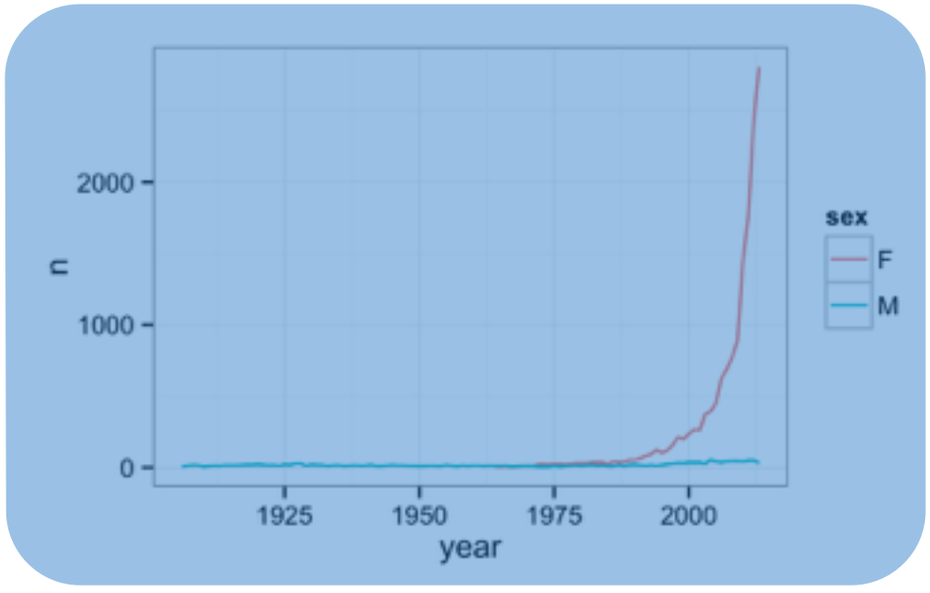
What's in a name?

Please select a name to explore.

Name:

Hadley

Here is a graph of the number of people named Hadley.



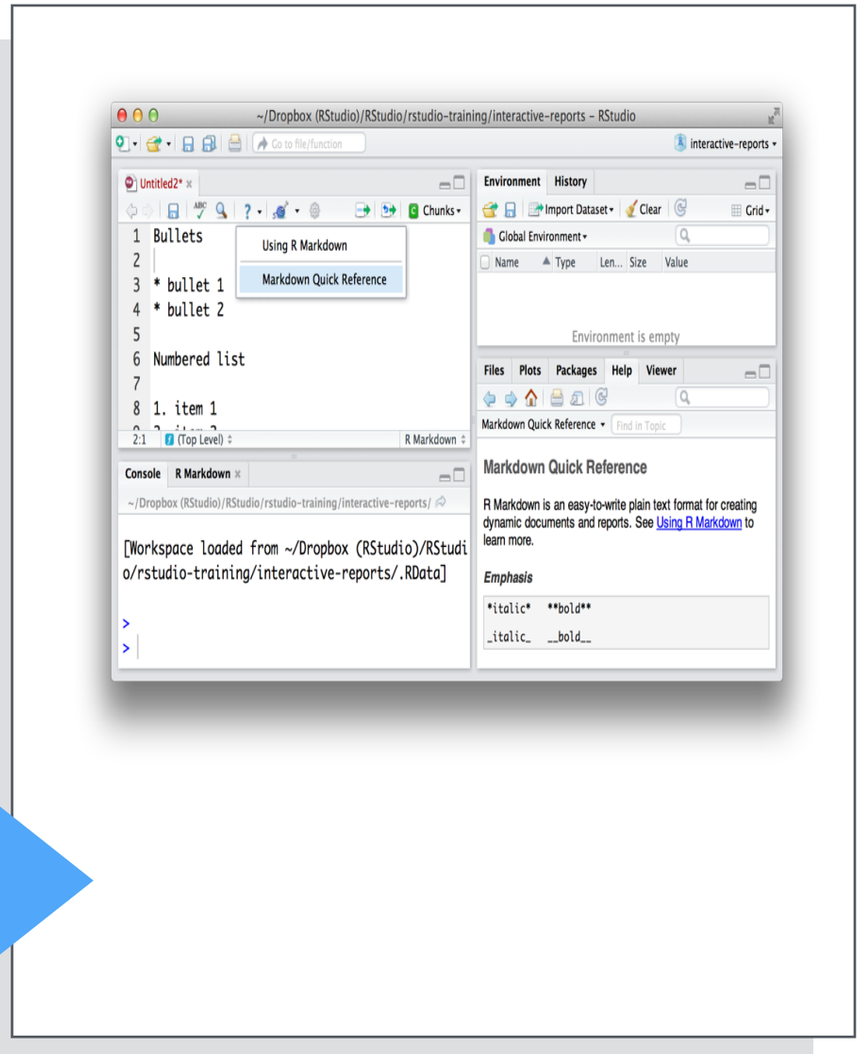
Hadley was more popular than ever in 2013. So far, 16866 people have been named Hadley.

Widgets

Widgets

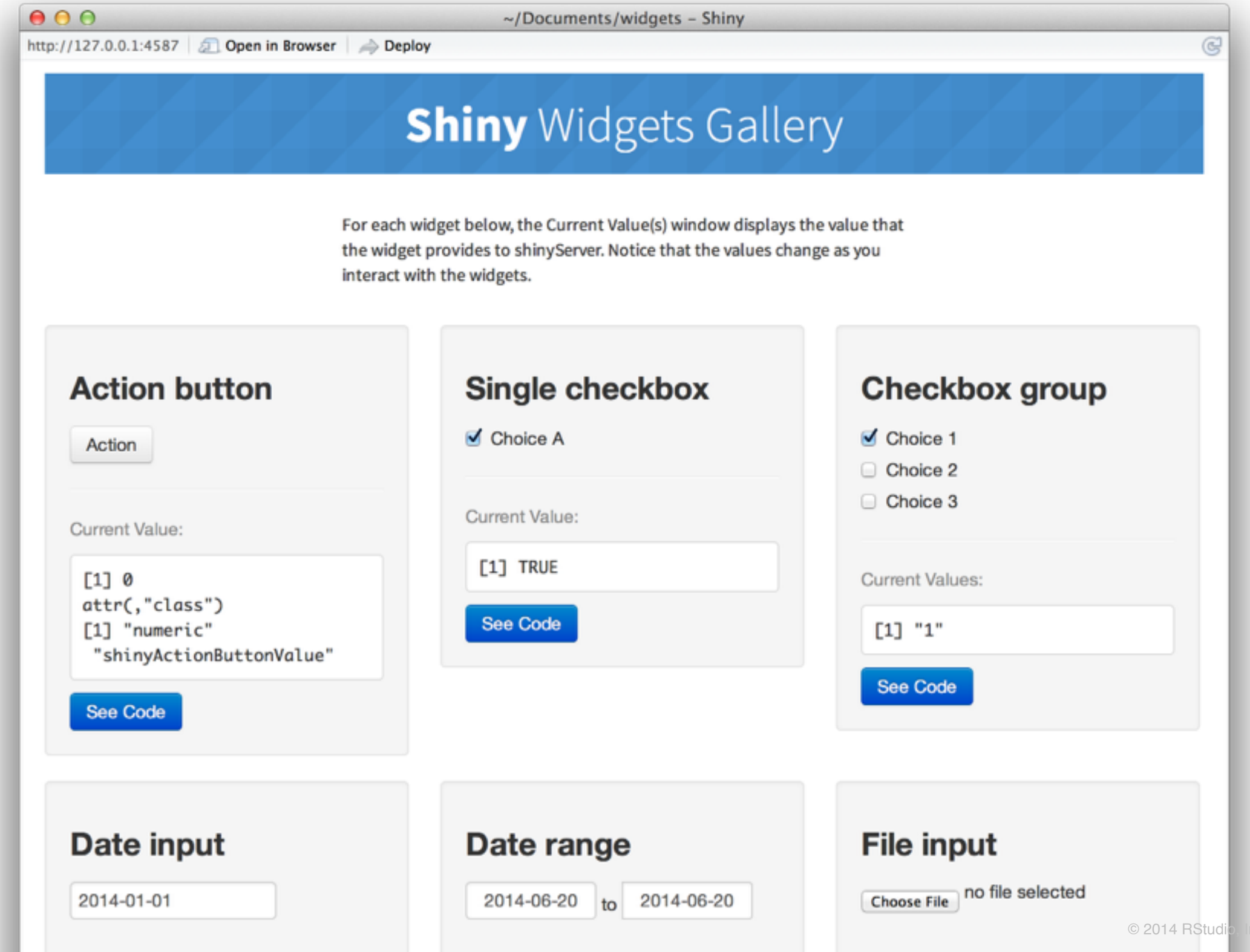
Create a widget with a widget function. Since the function is R code, you'll need to put it in a code chunk.

````
### A select box widget

```{r echo=FALSE}
selectInput(name = "choice",
  label = "Pick one, Any one",
  choices = c("Choice 1",
              "Choice 2",
              "Choice 3"))
```
````



function	widget
`actionButton`	Action button
`checkboxGroupIn`	Group of checkboxes
`checkboxInput`	Single checkbox
`dateInput`	Calendar to aid date selection
`dateRangeInput`	Pair of calendars for selecting date range
`fileInput`	File upload control wizard
`helpText`	Help text to accompany other widgets
`numericInput`	Field to enter numbers
`radioButtons`	Set of radio buttons
`selectInput`	Box with choices to select from
`sliderInput`	Slider bar
`submitButton`	Submit button
`textInput`	Field to enter text

shiny.rstudio.com/gallery/widgets

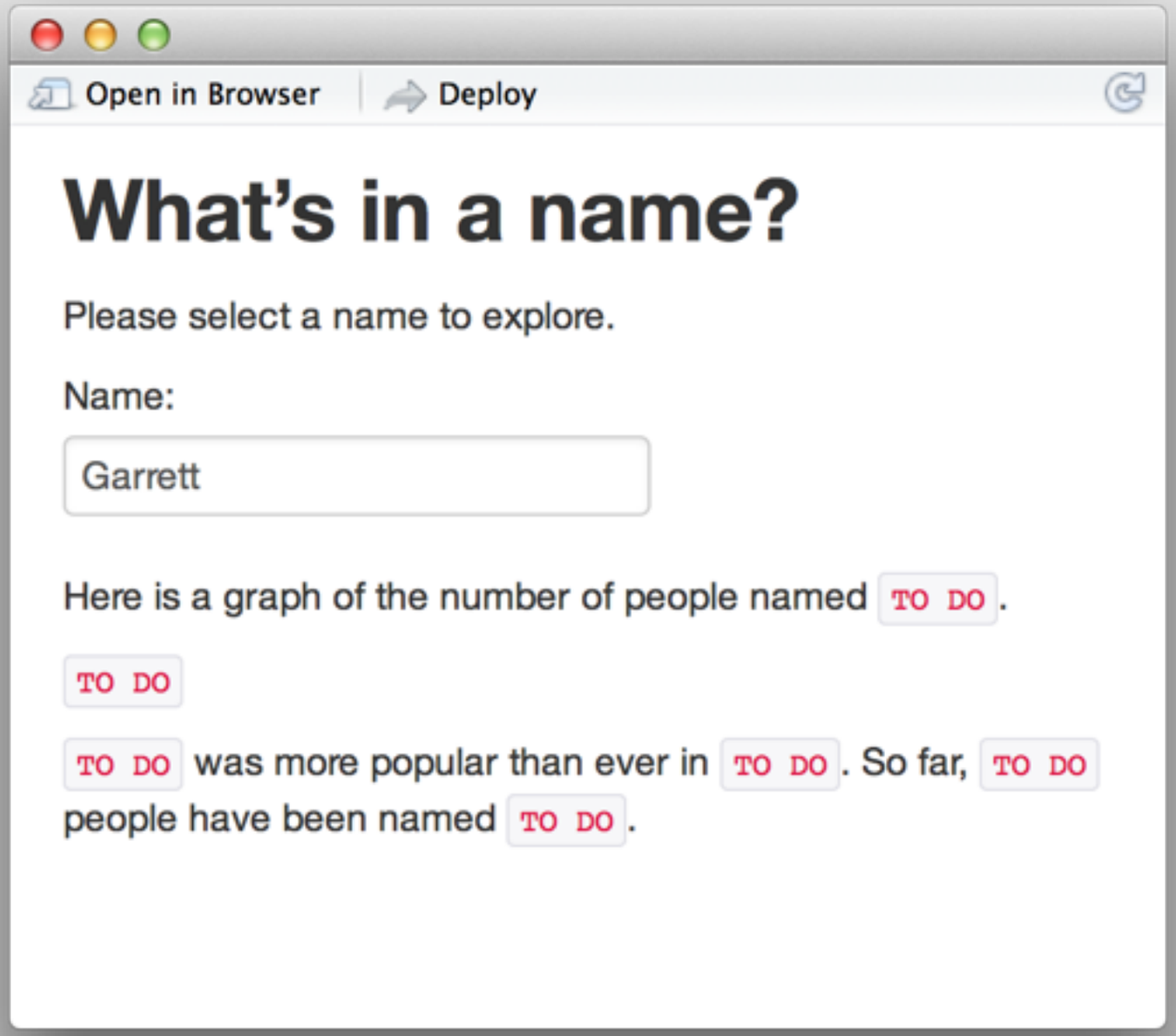


Your turn

Write the text of your document and add a widget like the one below (use your own name).

Look up the widget's help page or gallery example to find its arguments.

I added the place holders with `` `TO DO` ``.



````
---
runtime: shiny
output: html_document
---

## What's in a name?

Please select a name to explore.

```{r echo=FALSE}
textInput("name", "Name:", value = "Garrett")
```


Here is a graph of the number of people named `TO DO`.

`TO DO`

`TO DO` was more popular than ever in `TO DO`. So far, `TO DO` people have been named `TO DO`.
````

Reactivity 101

Your widget saves a value in R that you can call with input$<inputId>.

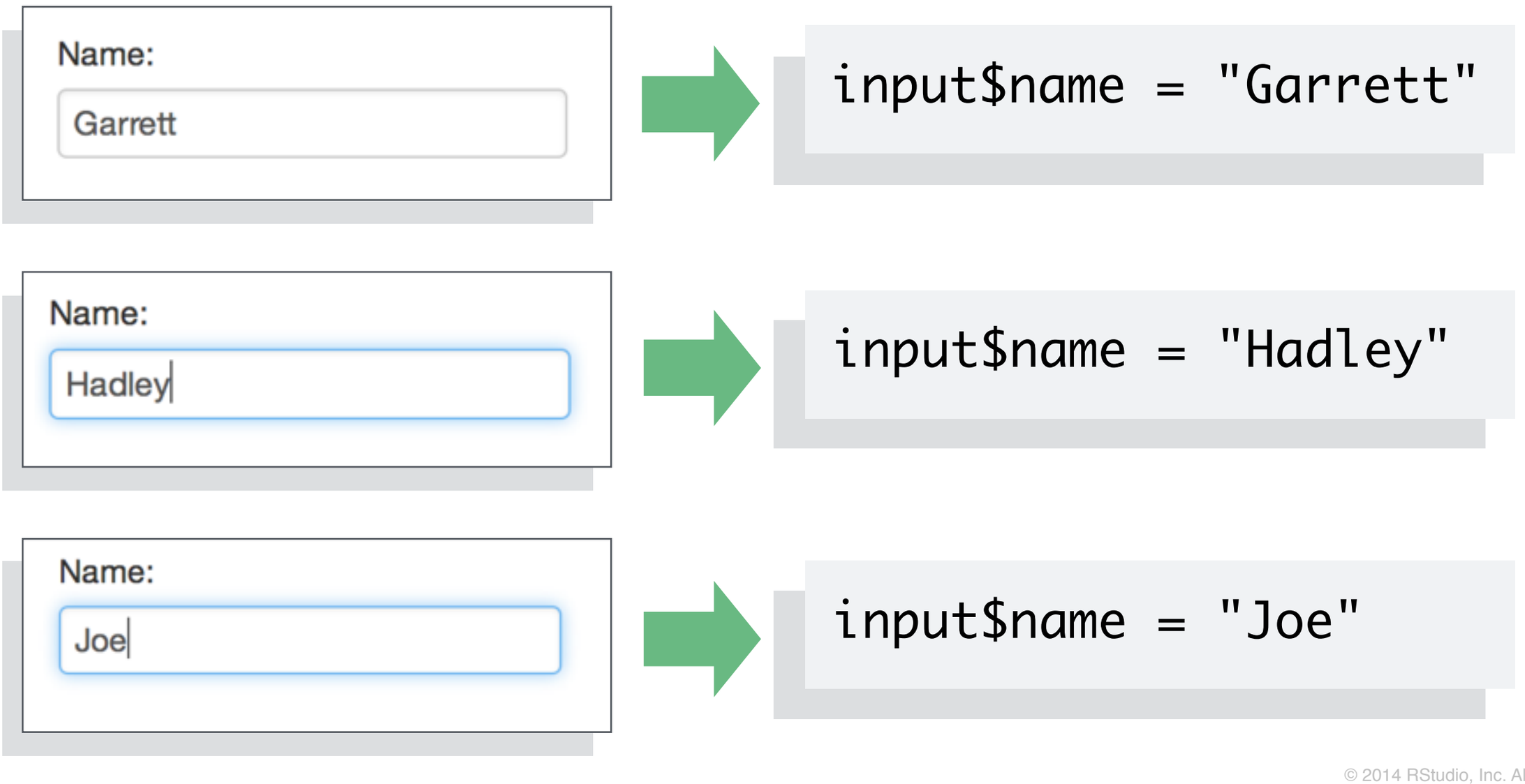
For example, you would call the value of this widget with this call.

```
textInput(
  inputId = "name",
  label = "Name:",
  value = "Garrett")
```

```
input$name
```

Reactivity 101

The widget value changes whenever a user changes the widget. (*Any Shiny output that uses the widget value will also change*).



Reactivity 101

You **CANNOT** call a widget value with a normal R function. (Shiny won't let you).

```
nchar(input$name)
```

Reactivity 101

You **CAN** call a widget value when you wrap an R expression with one of *render**, *reactive*, *isolate*, or *observe*.

```
renderText({nchar(input$name)})
```

```
reactive({nchar(input$name)})
```

```
isolate({nchar(input$name)})
```

```
observe({nchar(input$name)})
```

Reactivity 101

You **CAN** call a widget value when you wrap an R expression with one of *render**, *reactive*, *isolate*, or *observe*.

```
renderText({nchar(input$name)})
```

```
reactive({nchar(input$name)})
```

```
isolate({nchar(input$name)})
```

```
observe({nchar(input$name)})
```

Rendered outputs

render*

A family of functions that make R objects that use widget values and **update** whenever a widget value changes.

```
renderText({
  names <- subset(babynames, name == input$name)
  sum(names$n)
})
```

Use the function that creates the type of object you wish to make.

function	creates
`renderDataTable`	An interactive table (from a data frame, matrix, or other table-like structure)
`renderImage`	An image (saved as a link to a source file)
`renderPlot`	A plot
`renderPrint`	A code block of printed output
`renderTable`	A table (from a data frame, matrix, or other table-like structure)
`renderText`	A character string
`renderUI`	a Shiny UI element

render*

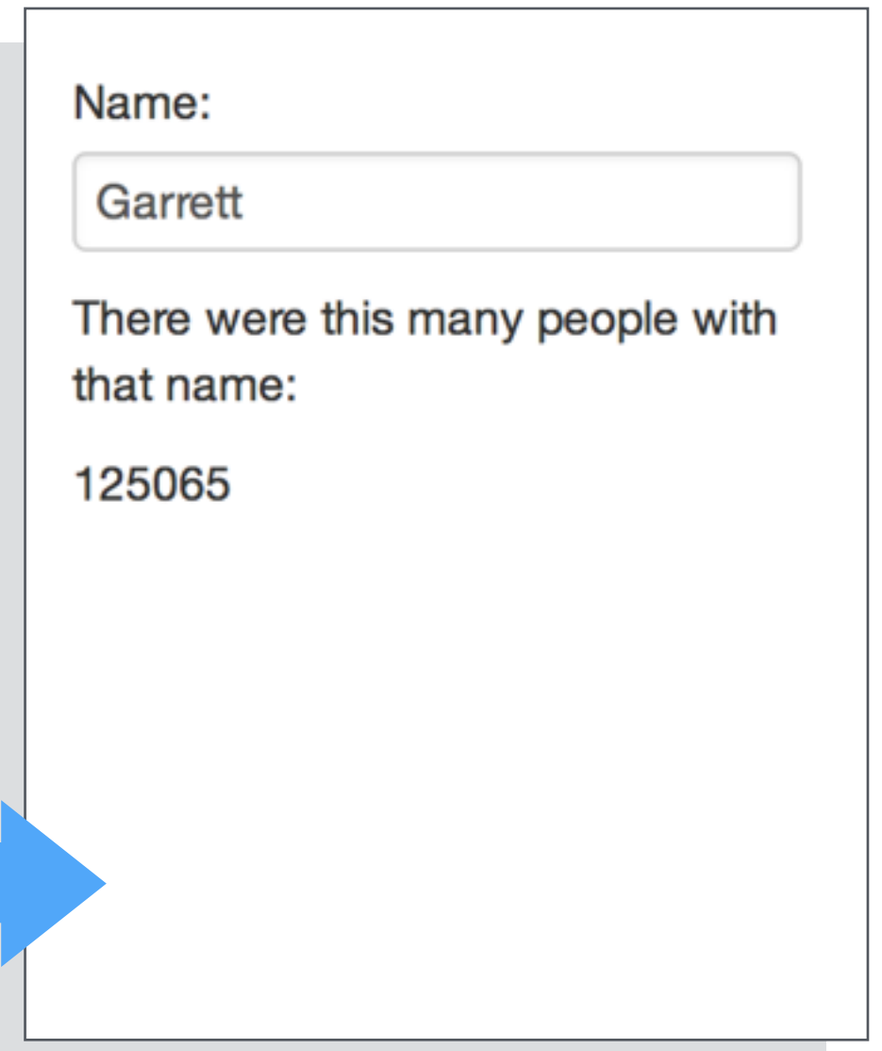
Each render function takes one argument: the code needed to build the final object. *(Use as many lines as you like)*

```
renderText({
  names <- subset(babynames, name == input$name)
  sum(names$n)
})
```

Putting it together

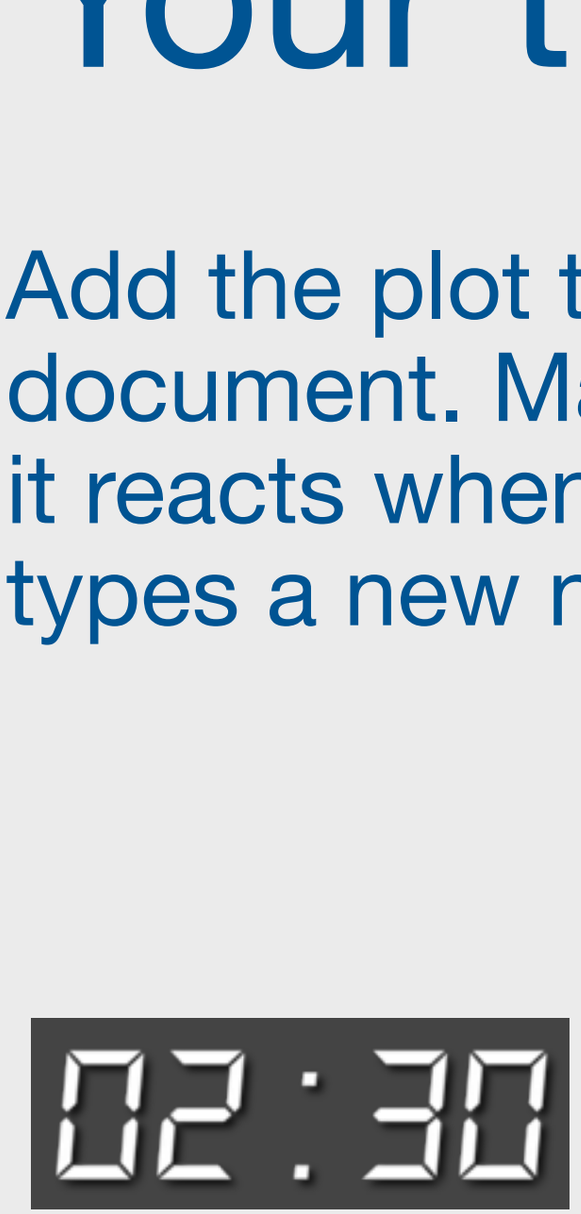
Once you make a widget, you can refer to it in later code blocks.

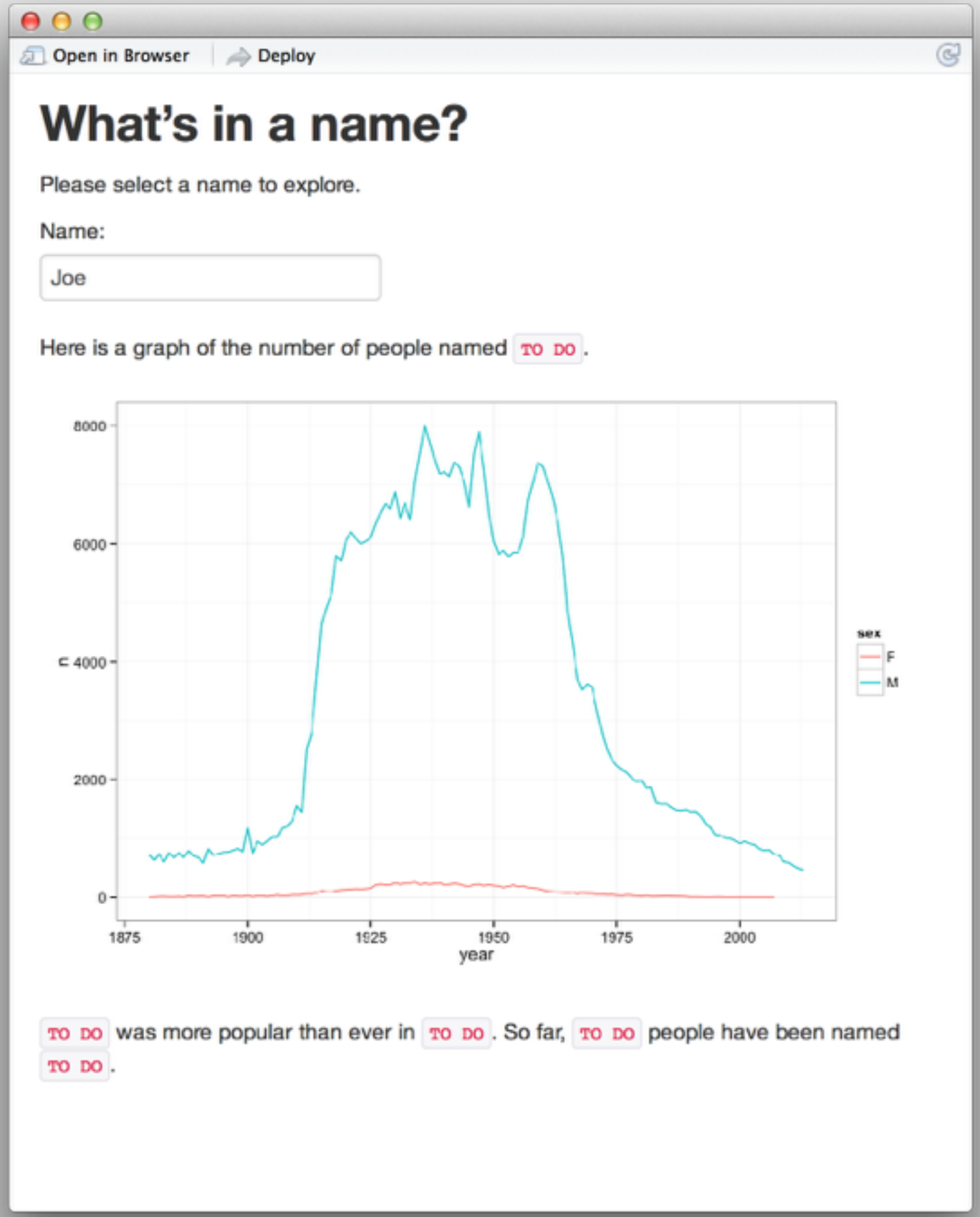
````
```{r echo=FALSE}
textInput("name", "Name:", value =
"Garrett")
```

There were this many people with that name:

```{r echo=FALSE}
library(babynames)
renderText({
  names <- subset(babynames,
    name == input$name)
  sum(names$n)
})
```
````



Your turn

Add the plot to your document. Make sure it reacts when a user types a new name.





````
---
runtime: shiny
output: html_document
---

## What's in a name?

Please select a name to explore.

```{r echo=FALSE}
textInput("name", "Name:", value = "Garrett")
```

Here is a graph of the number of people named `TO DO`.

```{r echo=FALSE, fig.height=3, fig.width=5}
library(babynames)
library(ggplot2)

renderPlot({
  names <- subset(babynames, name == input$name)
  qplot(year, n, data = names, geom ="line", color = sex) +
    theme_bw()
})
```

`TO DO` was more popular than ever in `TO DO`. So far, `TO DO` people have been named `TO DO`.
````

Please select a name to explore.

```{r echo=FALSE}
textInput("name", "Name:", value = "Garrett")
```

Here is a graph of the number of people named `TO DO`.

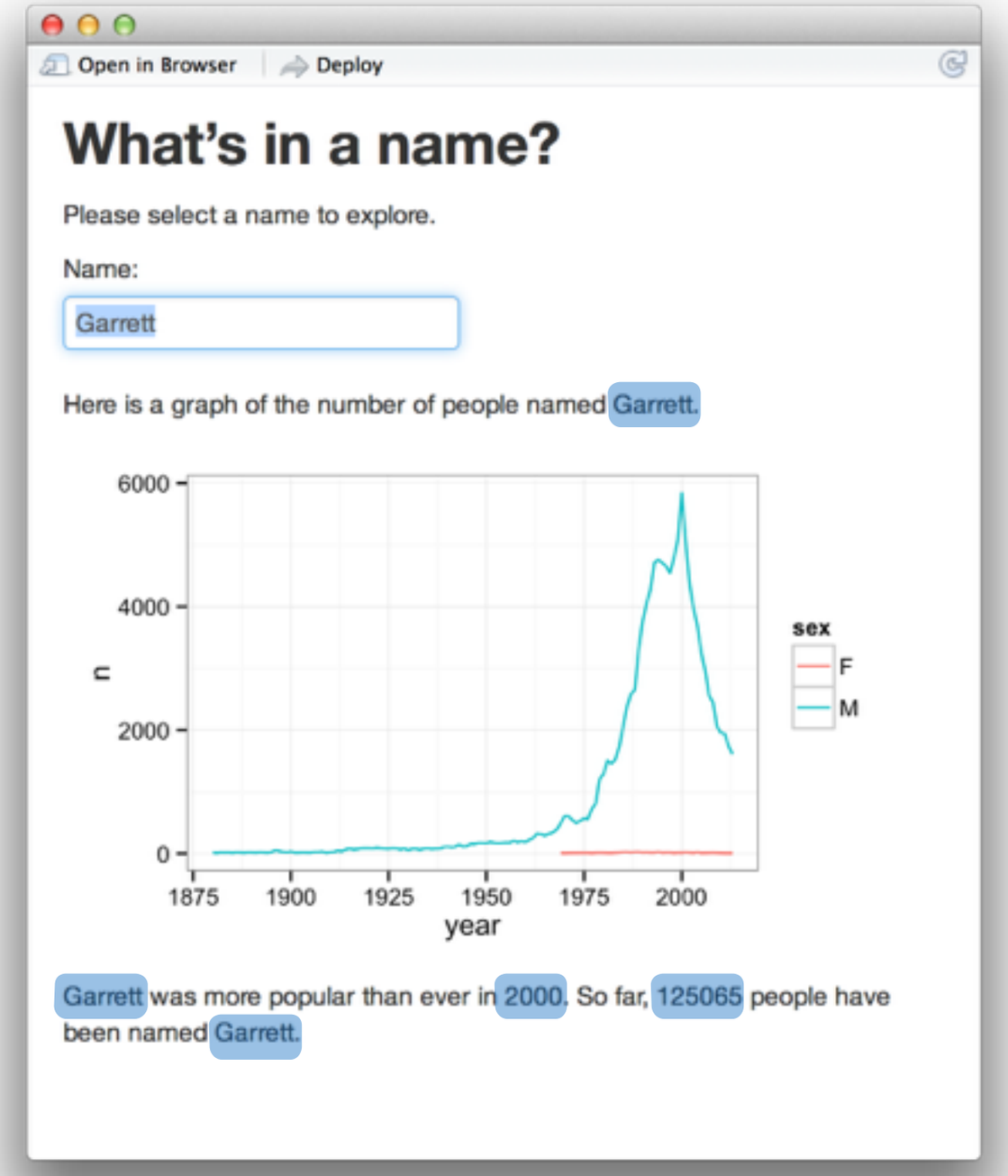
```{r echo=FALSE, fig.height=3, fig.width=5}
library(babynames)
library(ggplot2)

renderPlot({
  names <- subset(babynames, name == input$name)
  qplot(year, n, data = names, geom ="line", color = sex) +
    theme_bw()
})
```

`TO DO` was more popular than ever in `TO DO`. So far, `TO DO`

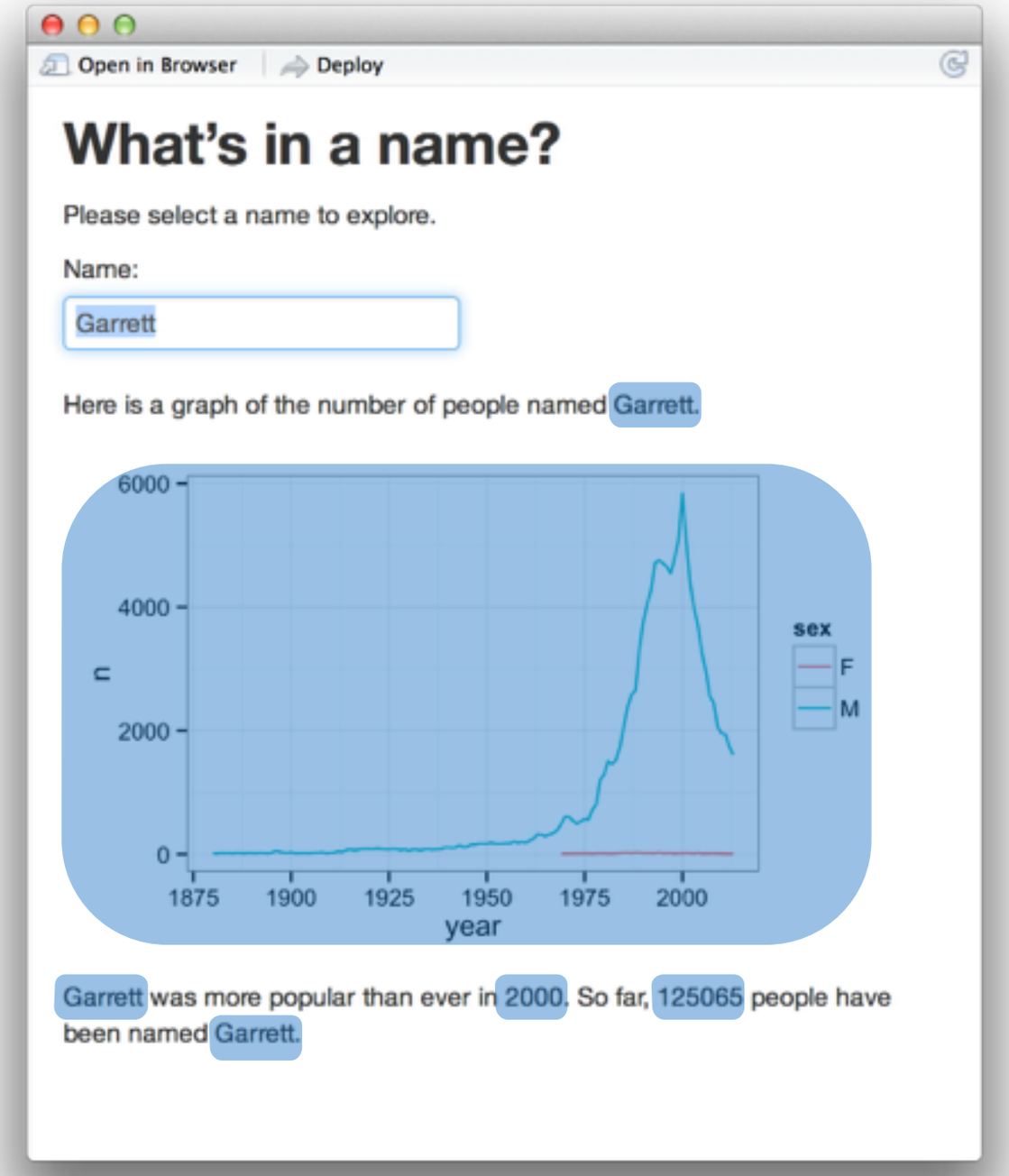
Reactive expressions

We've built the reactive plot.



Now, let's build the inline reactive text.

Each of these depends on input$name in the same way.



One reactive object used in multiple places.

```
names <- subset(babynames, name == input$name)
```

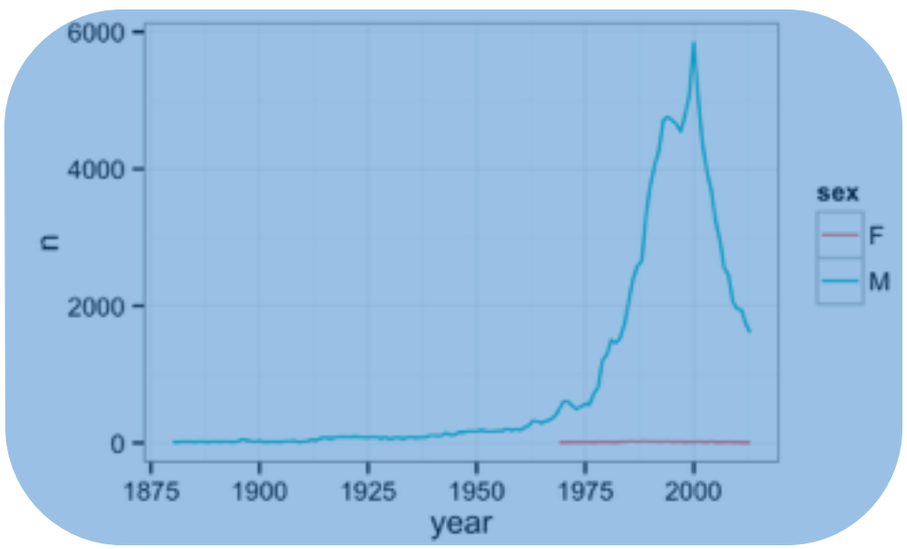
Garrett.
```
names$name[1]
```

Garrett
```
names$name[1]
```

Garrett.
```
names$name[1]
```

2000
```
names$year[which.max(names$n)]
```

125065
```
sum(names$n)
```

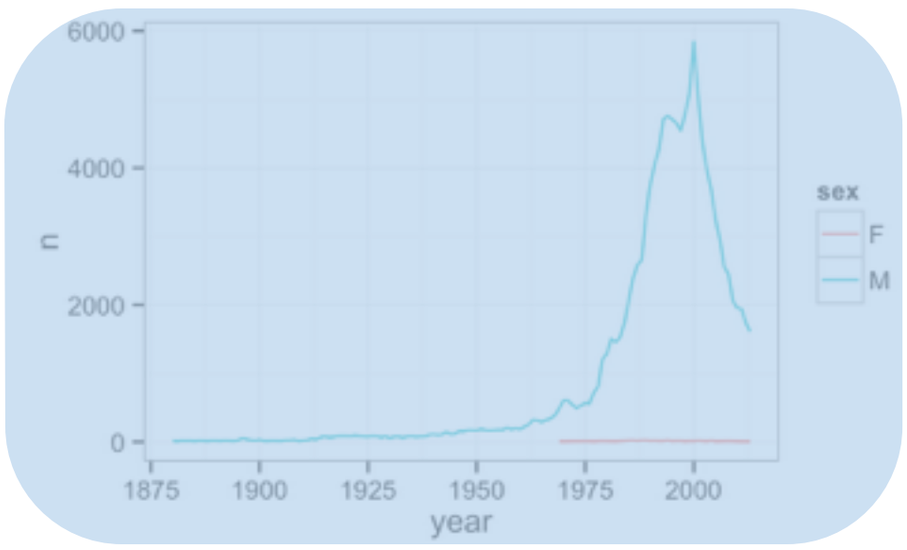



```
qplot(year, n, data = names,
  geom ="line", color = sex) +
  theme_bw()
```

One reactive object used in multiple places.

```
names <- subset(babynames, name == input$name)
```

Garrett.
```
names$name[1]
```
Garrett
```
names$name[1]
```
Garrett.
```
names$name[1]
```
2000.
```
names$year[which.max(names$n)]
```
125065
```
sum(names$n)
```




```
qplot(year, n, data = names,
  geom ="line", color = sex) +
  theme_bw()
```

```
Please select a name to explore.

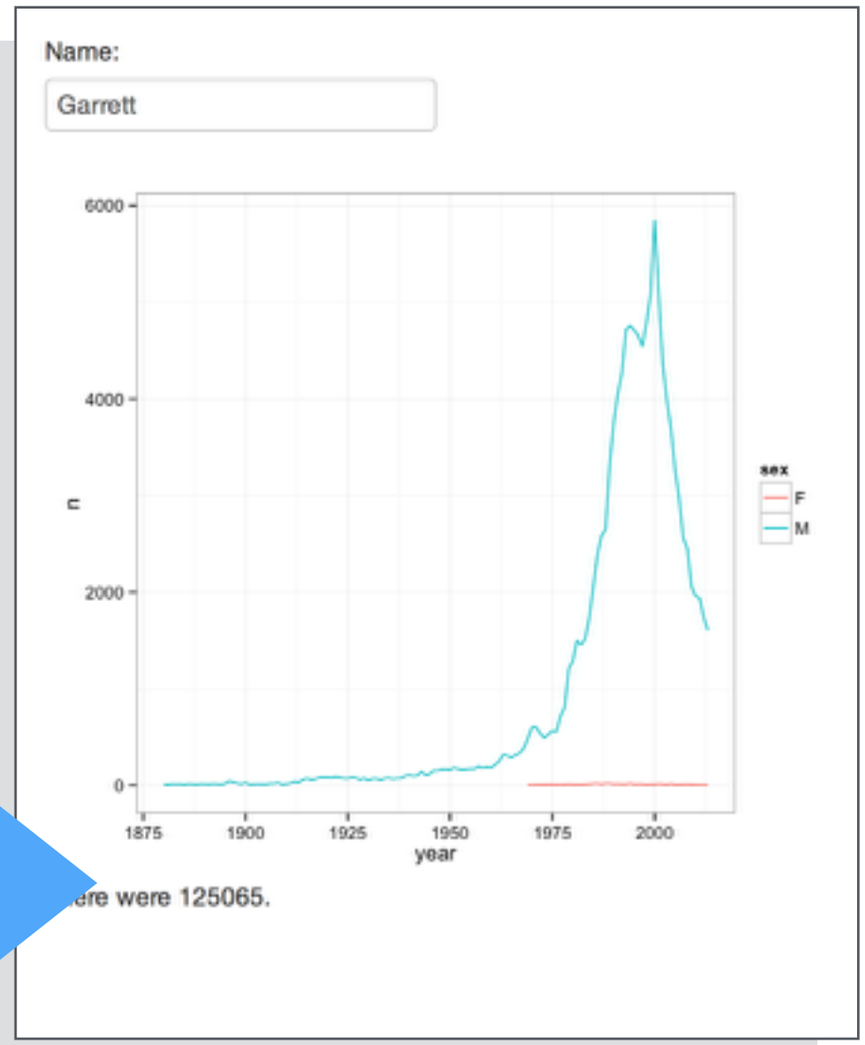
```{r echo=FALSE}
textInput("name", "Name:", value = "Gar
```

Here is a graph of the number of peopl

```{r echo=FALSE, fig.height=3, fig.wi
library(babynames)
library(ggplot2)

renderPlot({
  names <- subset(babynames, name == input$name)
  qplot(year, n, data = names, geom ="line", color = sex) +
    theme_bw()
})
```

`TO DO` was more popular than ever in `TO DO`. S
```

Objects created *inside* a render* function are only available *inside* that render function

Inefficient to repeat this in every render* function

Reactivity 101

You **CAN** call a widget value when you wrap an R expression with one of *render**, *reactive*, *isolate*, or *observe*.

```
renderText({nchar(input$name)})
```

```
reactive({nchar(input$name)})
```

```
isolate({nchar(input$name)})
```

```
observe({nchar(input$name)})
```

Reactive expressions

Take widget values and make a new **reactive** value.

```
names <- reactive({
  subset(babynames, name == input$name)
})
```

The value will change whenever the widget changes

Reactive expressions

You can use a reactive value in multiple render* functions.
Call a reactive values, as if it were a function.

````
```{r echo=FALSE}
textInput("name", "Name:", value = "Garrett")
library(babynames)
names <- reactive({
  subset(babynames, name == input$name)
})
```

```{r echo=FALSE}
library(ggplot2)
renderPlot({
  qplot(year, n, data = names(),
    color = sex) + theme_bw()
})
```

There were `r renderText(sum(names()$n))`.
````



Reactive expressions are reactive

So you must call reactive values from within *render**, *reactive*, *isolate*, or *observe*.

Your turn

Finish your reactive document.

1. Create a reactive expression that returns the data set for the selected name.

2. Replace each `` `TO DO` `` with an inline rendered output (that uses the reactive expression from step 1).

````
---
runtime: shiny
output: html_document
---

## What's in a name?

Please select a name to explore.

```{r echo=FALSE}
library(babynames)
textInput("name", "Name:", value = "Garrett")
names <- reactive(subset(babynames, name == input$name))
```

Here is a graph of the number of people named `r renderText(input$name)`.

```{r echo=FALSE, fig.height=3, fig.width=5}
library(ggplot2)
renderPlot({
  qplot(year, n, data = names(), geom ="line", color = sex) +
    theme_bw()
})
```


`r renderText(input$name)` was more popular than ever in `r renderText(names()
$year[which.max(names()$n)])`. So far, `r renderText(sum(names()$n))` people have been named `r
renderText(input$name)`.
````

isolate & observe

Reactivity 101

You **CAN** call a widget value when you wrap an R expression with one of *render**, *reactive*, *isolate*, or *observe*.

```
renderText({nchar(input$name)})
```

```
reactive({nchar(input$name)})
```

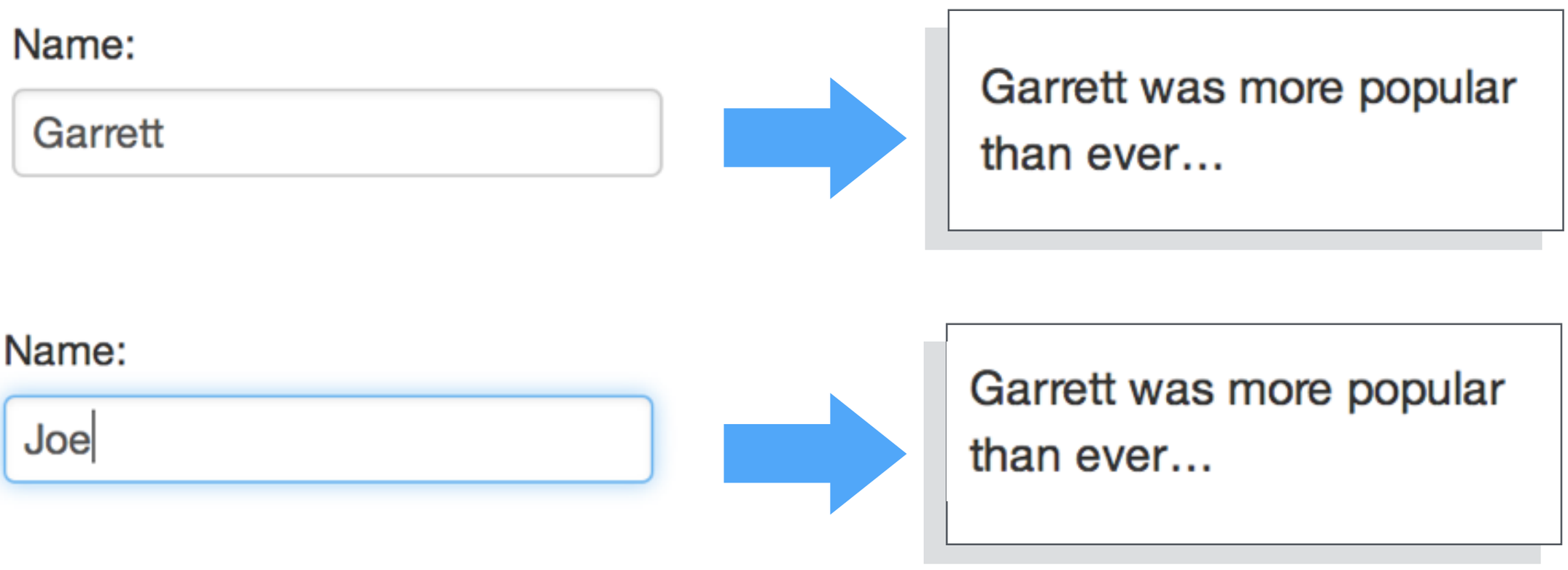
```
isolate({nchar(input$name)})
```

```
observe({nchar(input$name)})
```

isolate

Isolate isolates an expression from a widget. The object will **NOT** react when the isolated widget changes.

```
`r renderText(isolate(input$name))` was more popular than ever…
```

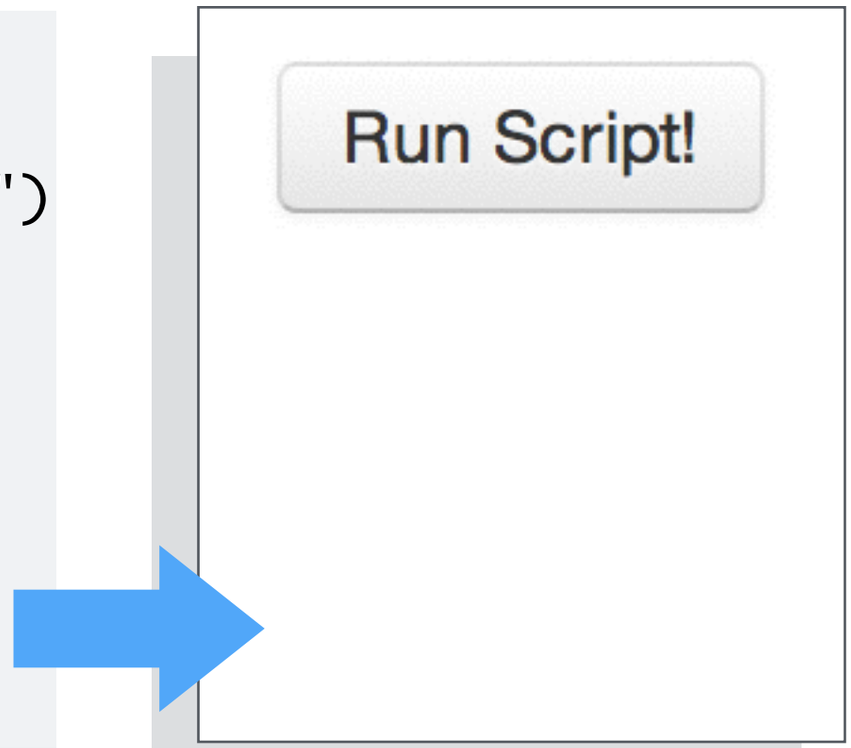


observe

observe will rerun the R expression each time a widget changes, but observe will not return a result.

Use observe for the side effects of the R expression.

```
```{r echo=FALSE}
actionButton("button", "Run Script!")
observe({
  if(input$button != 0)
    source("secret-script.R")
})
```
```

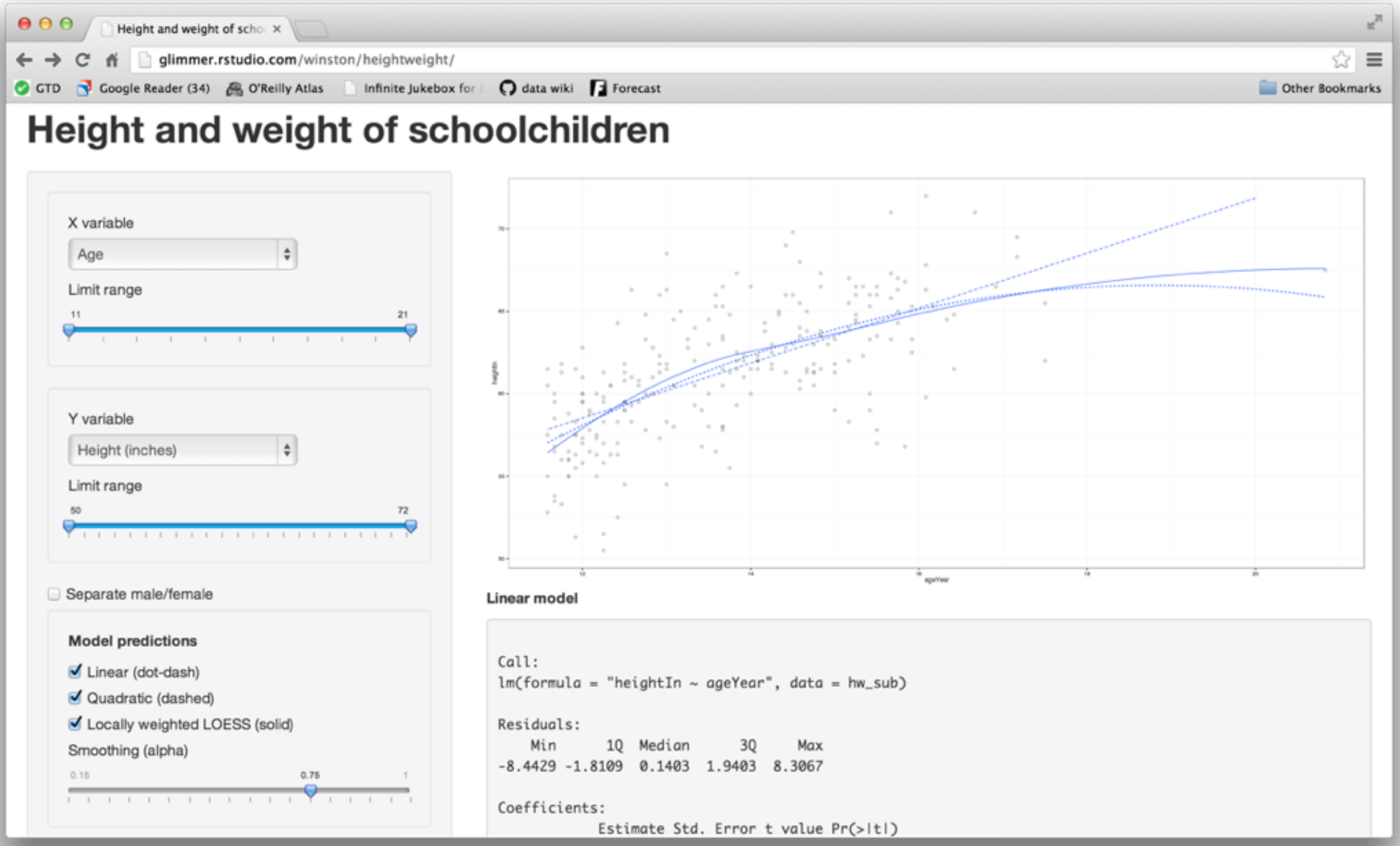


Shiny

An R package that does two things

1. Creates reactive R objects
2. **Builds HTML web pages**

Web Apps



http://glimmer.rstudio.com/winston/heightweight/

Dialect Survey Results

spark-1590165977.us-west-2.elb.amazonaws.com/jkatz/SurveyMaps/

GTD | Google Reader (34) | O'Reilly Atlas | Infinite Jukebox for | data wiki | Forecast | Other Bookmarks

Dialect Survey Results

NC STATE UNIVERSITY

Question:

105. What is your generic ter

City:

--

Starting with the point-referenced data from Bert Vaux's online survey of English dialects, we used a *k*-nearest neighbor smoothing algorithm to estimate the probability of seeing a particular answer—eg, whether a person would say *soda*, *pop*, or *coke*—at every point in the continental US.

The composite map gives a picture of the overall distribution, coloring each cell according to whichever answer is estimated to be most likely at that location. The more clearly one answer dominates, the darker the color. Individual maps show estimated probability of each particular answer at a given location, with larger probabilities shown in red and smaller probabilities shown in blue. At the moment, **only the four most popular answers for each survey question are displayed.**

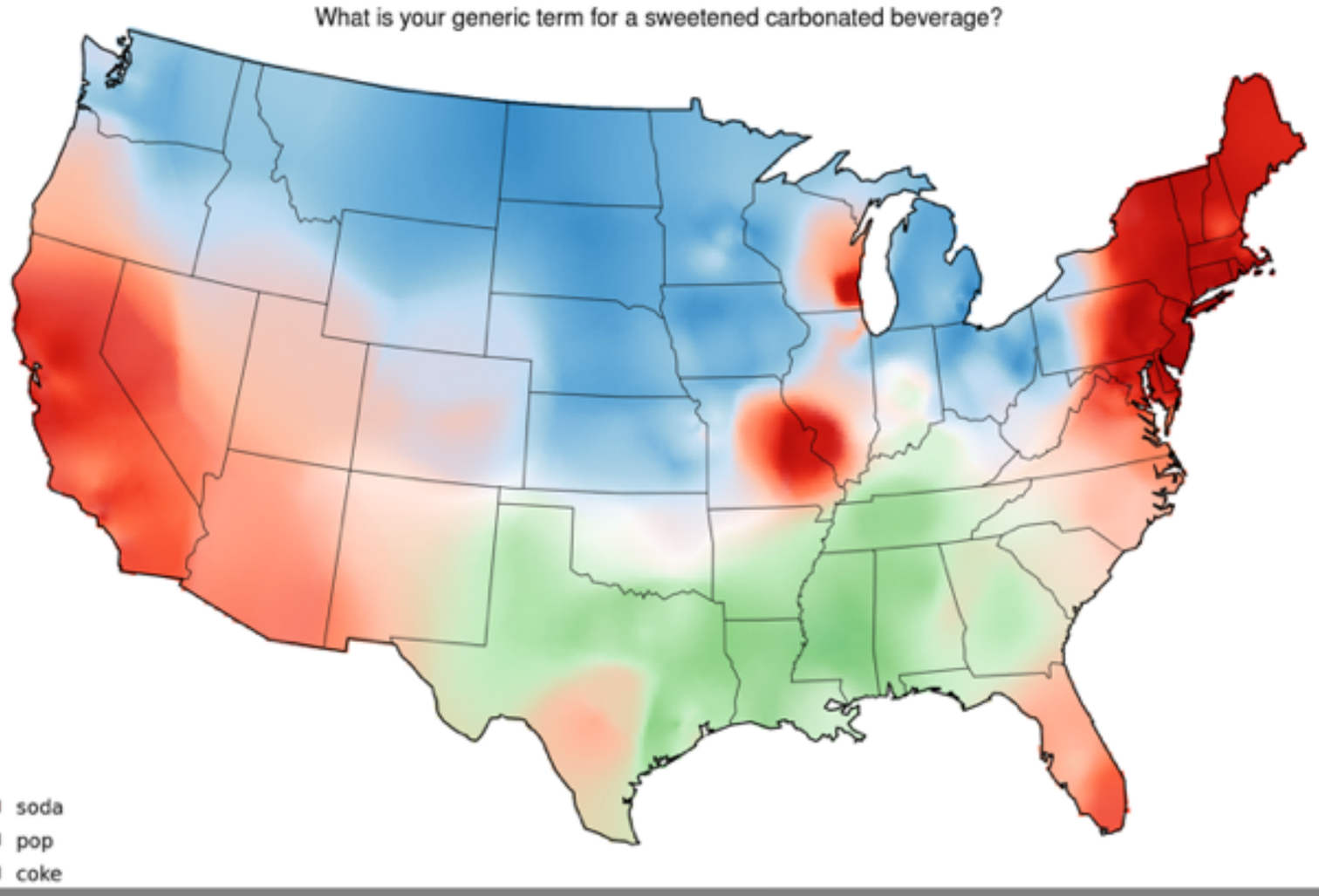
Further details of the model construction can be found in the accompanying poster. See also the aggregate dialect difference map.

All coding was done in R / Shiny.

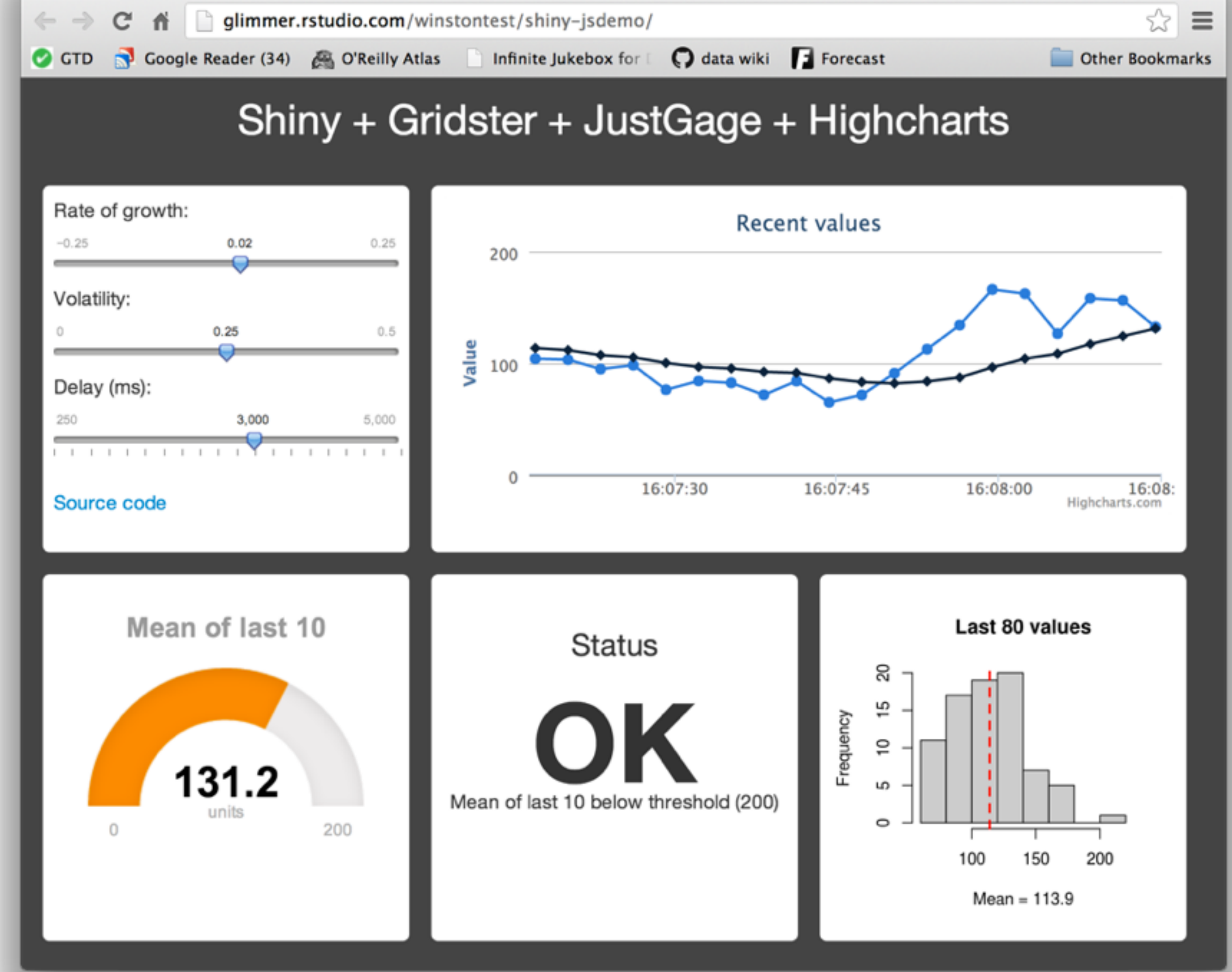
Joshua Katz 2013

Original survey map

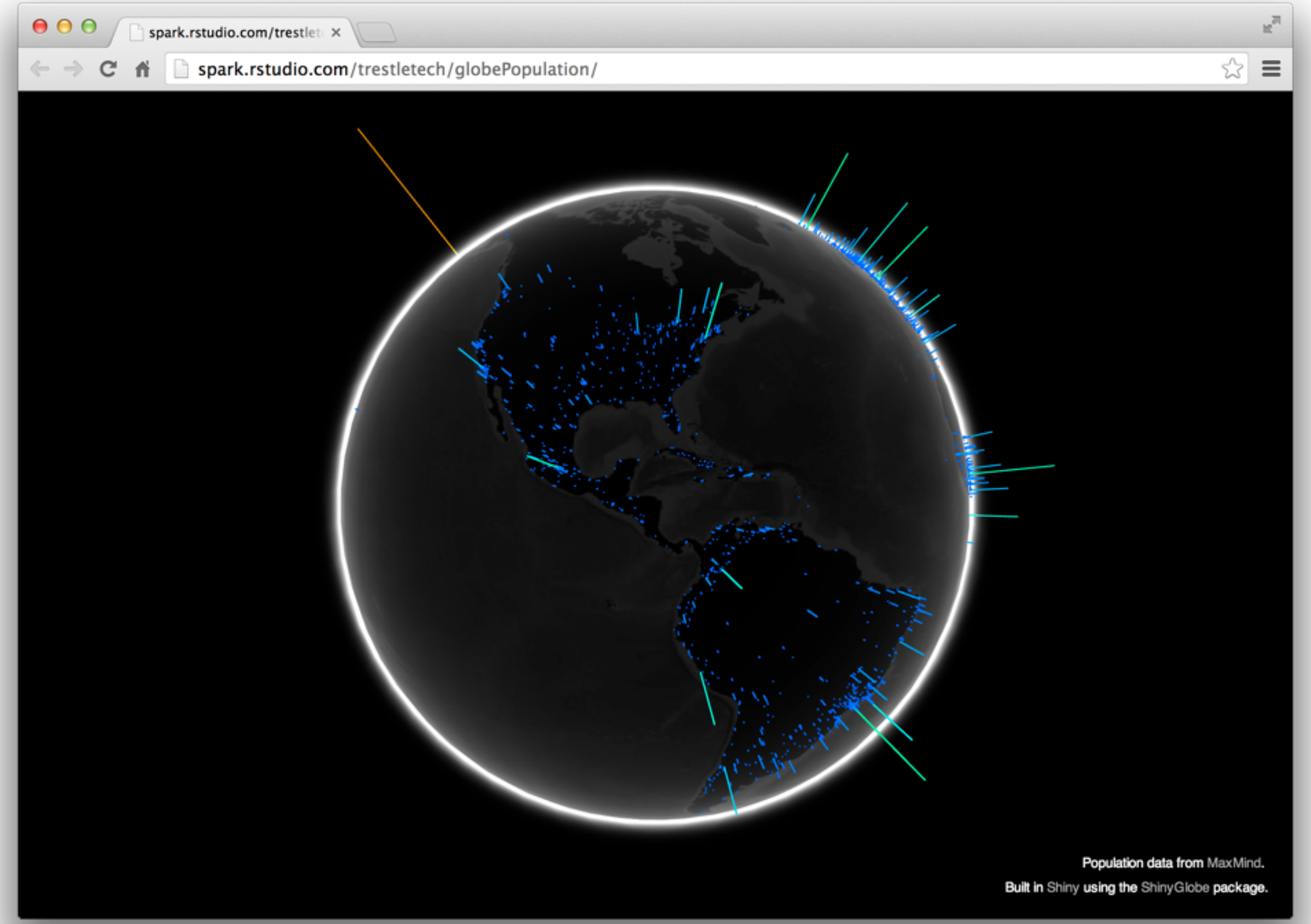
Composite | Individual



http://spark-1590165977.us-west-2.elb.amazonaws.com/jkatz/SurveyMaps/



http://glimmer.rstudio.com/winstontest/shiny-jsdemo/



http://spark.rstudio.com/trestletech/globePopulation/

~/Desktop/shiny1 - Shiny

http://127.0.0.1:3517 | Open in Browser | Deploy

What's in a name?

Name:

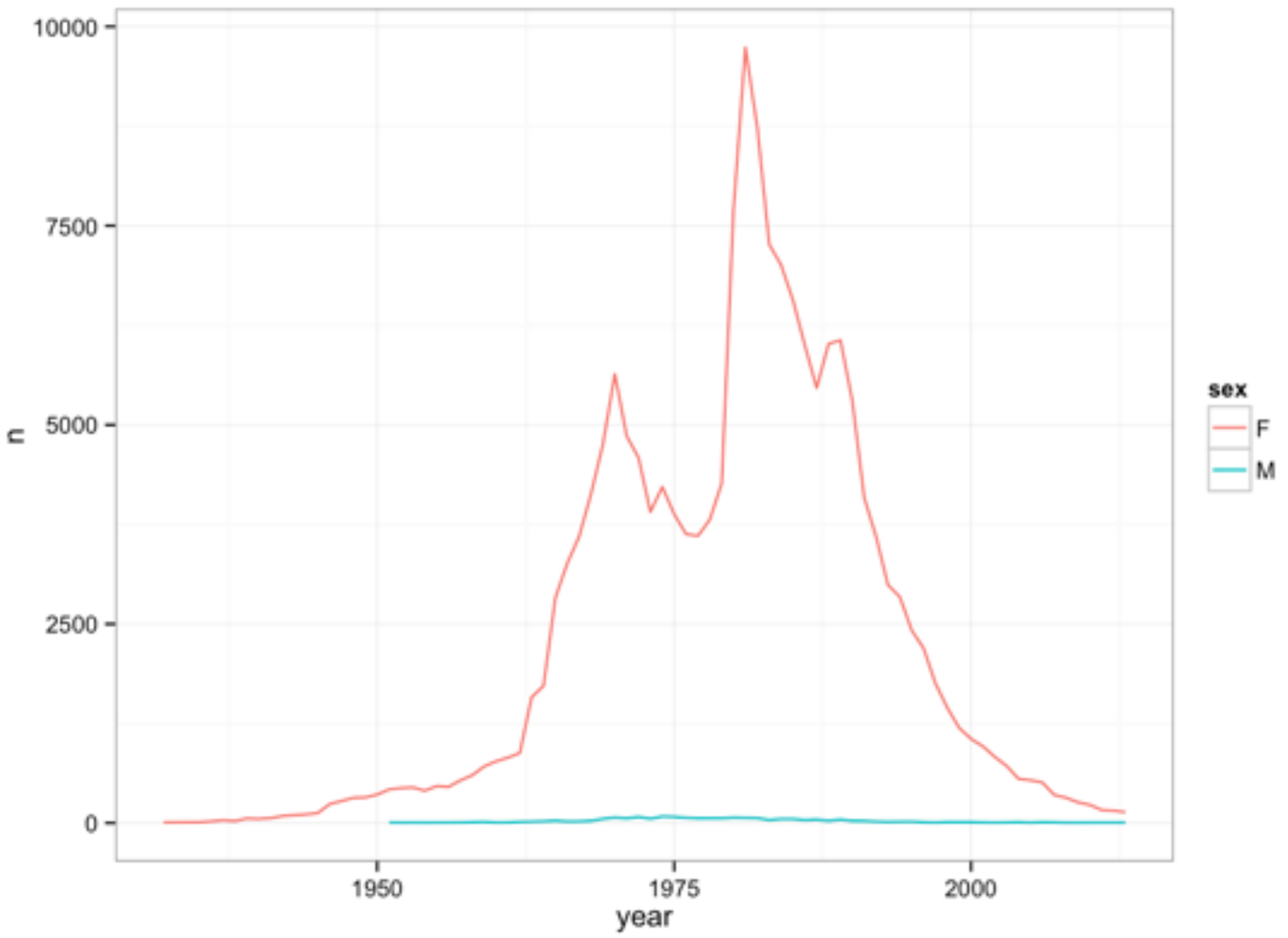
Kristin

180207

people had this name

1981

peak year

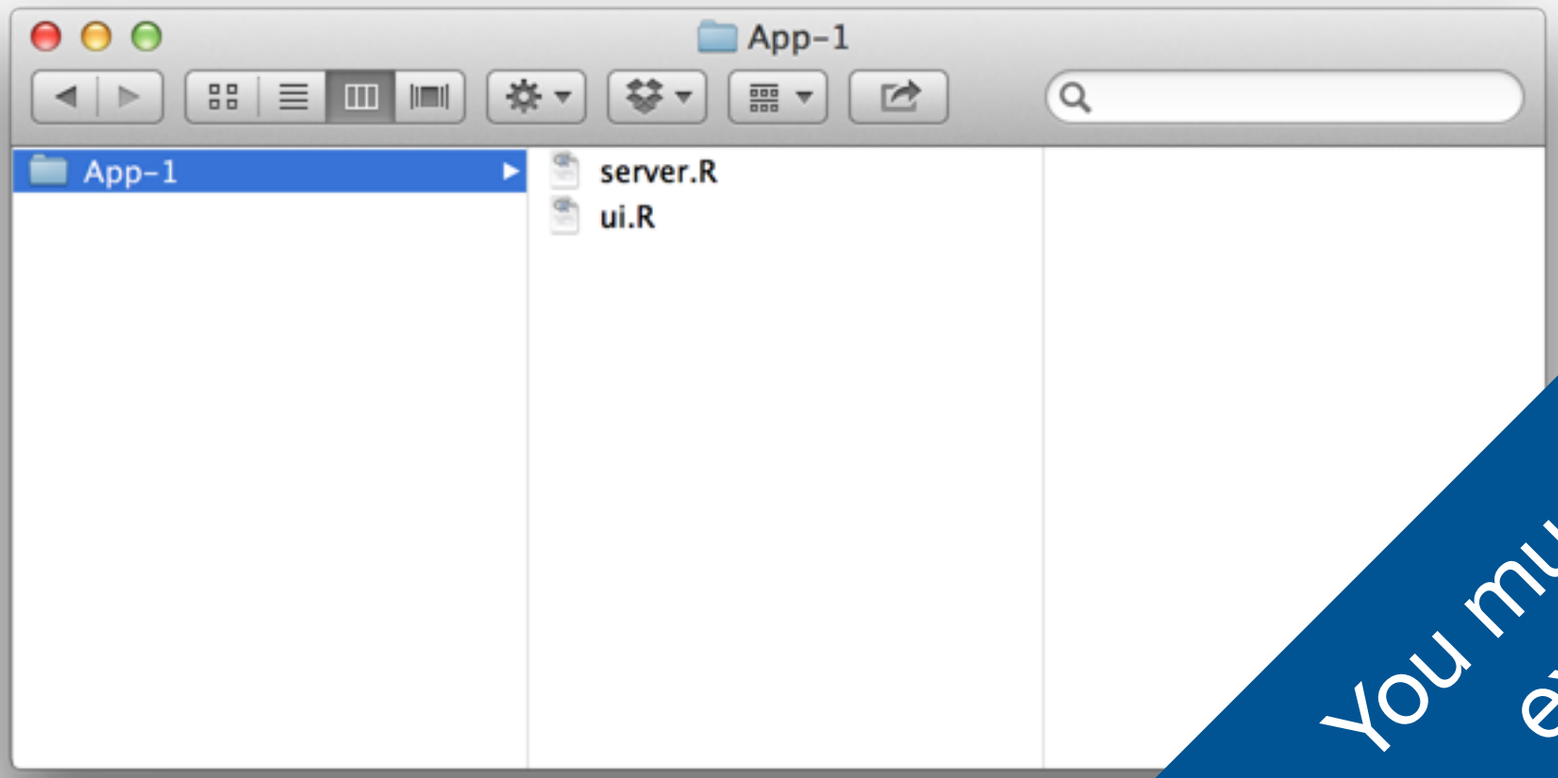


Structure

Structure of a Shiny web app

One directory with two files:

- server.R
- ui.R



You must use these exact names

runApp

You launch the app with `runApp` (your computer will build a local web site that hosts the app).

```
runApp("~/Documents/App-1")
```

File path to app directory.

R will append the file path to the working directory, if path does not begin at the home directory

Warm up

Open a new RStudio project.

New Project > New Directory > Shiny Web App

RStudio will create a directory that has a template server.R and ui.R file in it. (Rstudio will set your working directory to this directory).

Run the template app.

server.R

server.R contains all of the instructions (R code) your app needs to build its rendered outputs.

server.R **always** includes the following code.

```
shinyServer(function(input, output) {


})
```

Place all of the R code from your document between the braces (except for the widget functions).

```
shinyServer(function(input, output) {
  library(babynames)
  names <- reactive(subset(babynames, name == input$name))
  renderText(input$name)

  library(ggplot2)
  renderPlot({
    qplot(year, n, data = names(), geom ="line", color = sex) +
      theme_bw()
  })
  renderText(input$name)
  renderText(names()$year[which.max(names()$n)])
  renderText(sum(names()$n))
  renderText(input$name)
})
```

Move to the top anything that only needs to be run once per server (*e.g., loading packages, data sets, or scripts*)

```
library(babynames)
library(ggplot2)

shinyServer(function(input, output) {
  names <- reactive(subset(babynames, name == input$name))
  renderText(input$name)
  renderPlot({
    qplot(year, n, data = names(), geom ="line", color = sex) +
      theme_bw()
  })
  renderText(input$name)
  renderText(names()$year[which.max(names()$n)])
  renderText(sum(names()$n))
  renderText(input$name)
})
```

Save **every** rendered object to an element of `output`

```
library(babynames)
library(ggplot2)

shinyServer(function(input, output) {
  names <- reactive(subset(babynames, name == input$name))
  output$name <- renderText(input$name)
  output$trend <- renderPlot({
    qplot(year, n, data = names(), geom ="line", color = sex) +
      theme_bw()
  })
  output$peak <- renderText(names()$year[which.max(names()$n)])
  output$total <- renderText(sum(names()$n))
})
```

Studio

Save **every** rendered object to an element of `output`

```
library(babynames)
library(ggplot2)

shinyServer(function(input, output) {
  names <- reactive(subset(babynames, name == input$name))
  output$name <- renderText(input$name)
  output$trend <- renderPlot({
    qplot(year, n, data = names(), geom ="line", color = sex) +
      theme_bw()
  })
  output$peak <- renderText(names()$year[which.max(names()$n)])
  output$total <- renderText(sum(names()$n))
})
```

Warm up 2

Copy the code on the following slide into your server.R script. What rendered objects does it create?

```
library(babynames)
library(ggplot2)

shinyServer(function(input, output) {
  names <- reactive({
    subset(babynames, name == input$name)
  })

  output$trend <- renderPlot({
    qplot(year, n, data = names(), geom = "line", color = sex) +
      theme_bw()
  })

  output$total <- renderText({
    sum(names()$n)
  })

  output$peak <- renderText({
    names()$year[which.max(names()$n)]
  })

})
```

ui.R

ui.R builds the web page that displays your widgets and rendered output. In this file, you

- create the layout
- place widgets and output
- add html elements (optional)

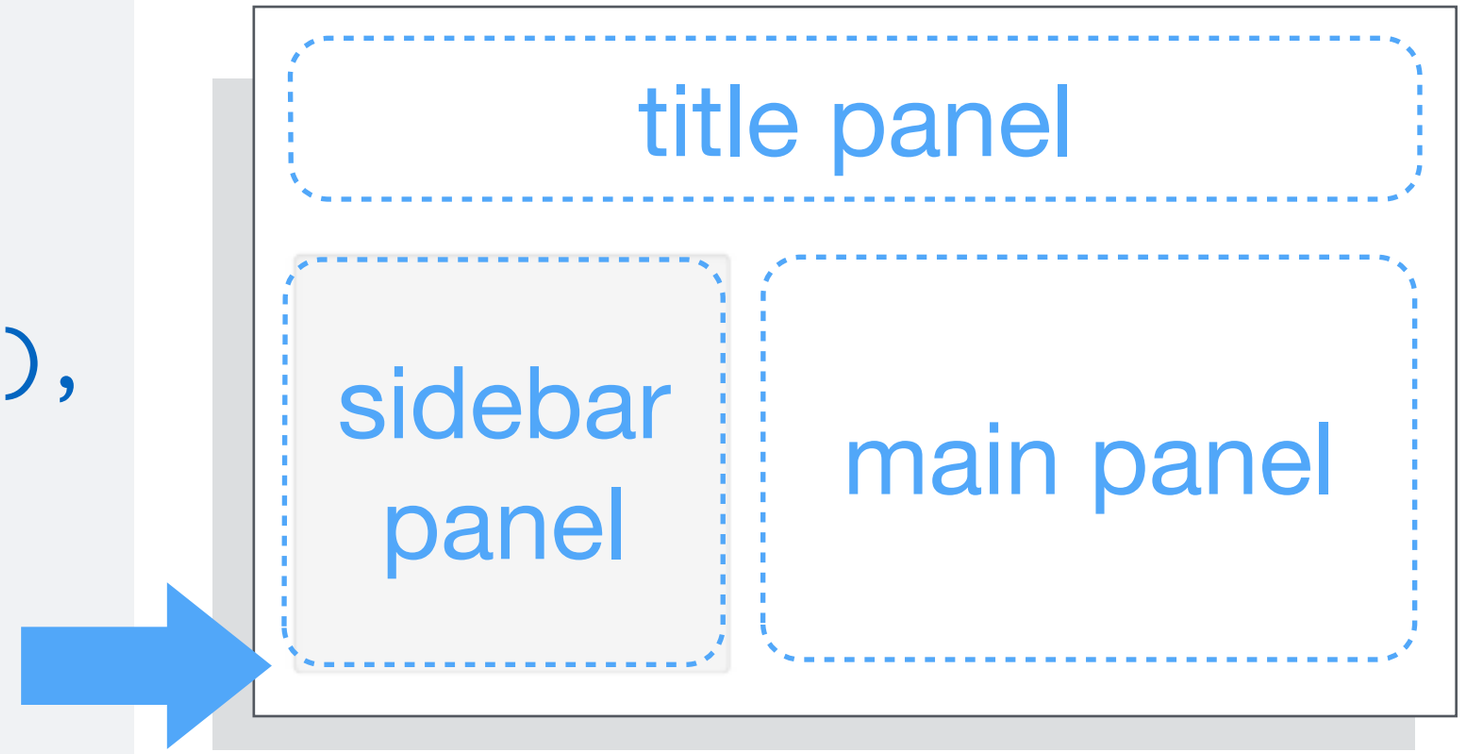
Layout

The shortest viable ui.R file:

```
shinyUI(fluidPage())
```

Place layout functions inside `fluidPage` to layout the app

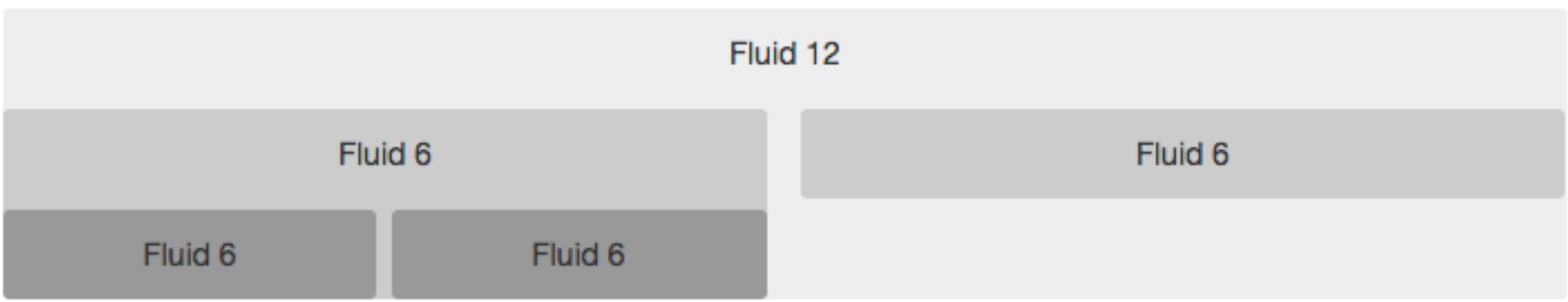
```
shinyUI(fluidPage(
  titlePanel(""),
  sidebarLayout(
    sidebarPanel(),
    mainPanel()
  )
))
```



Application Layout Guide

http://shiny.rstudio.com/articles/layout-guide.html

`sidebarLayout` is an easy, pre-packaged layout, but you can create any layout you like with Shiny's fluid grid system.



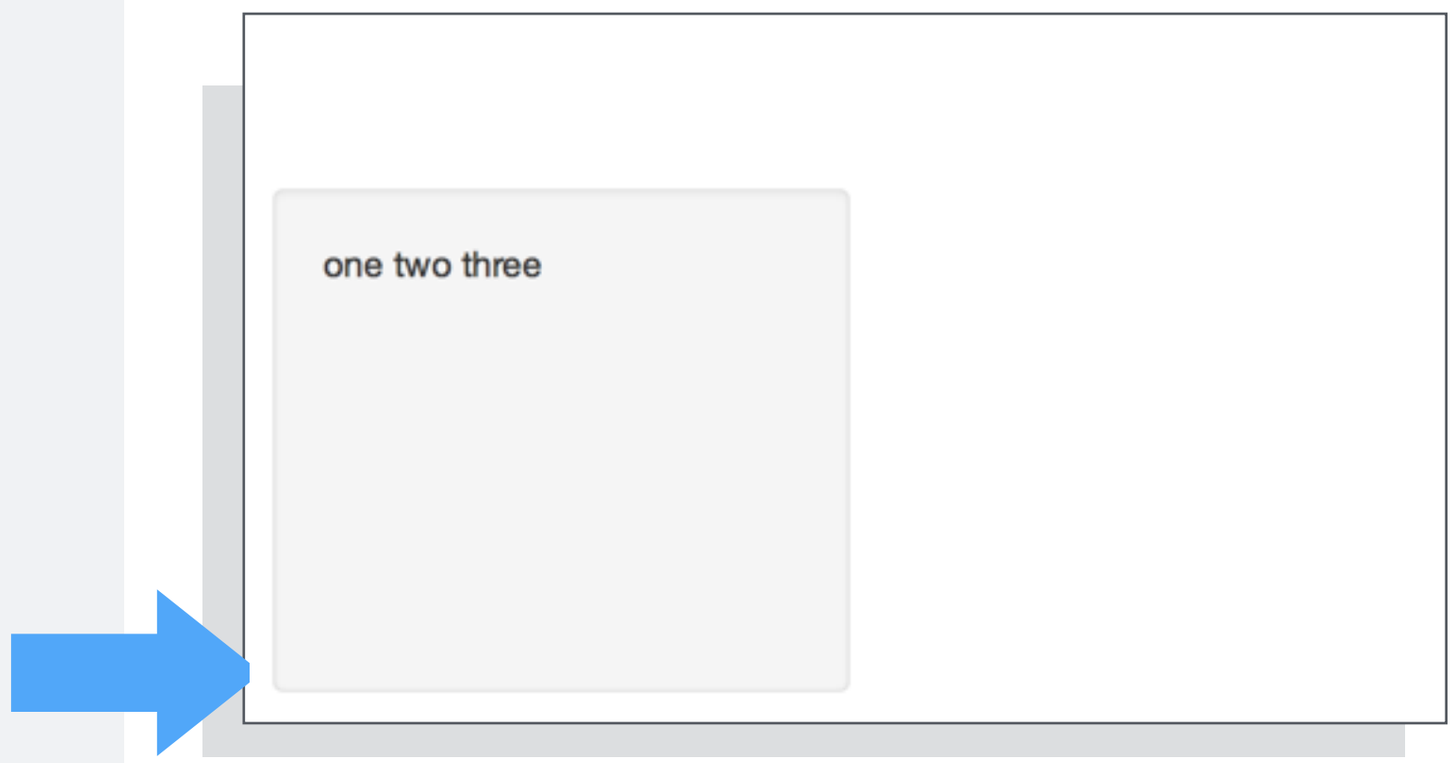
Adding text

Place raw text into any panel you like.

```
shinyUI(fluidPage(
  titlePanel("one"),
  sidebarLayout(
    sidebarPanel("two"),
    mainPanel("three")
  )
))
```

Place raw text into any panel you like.

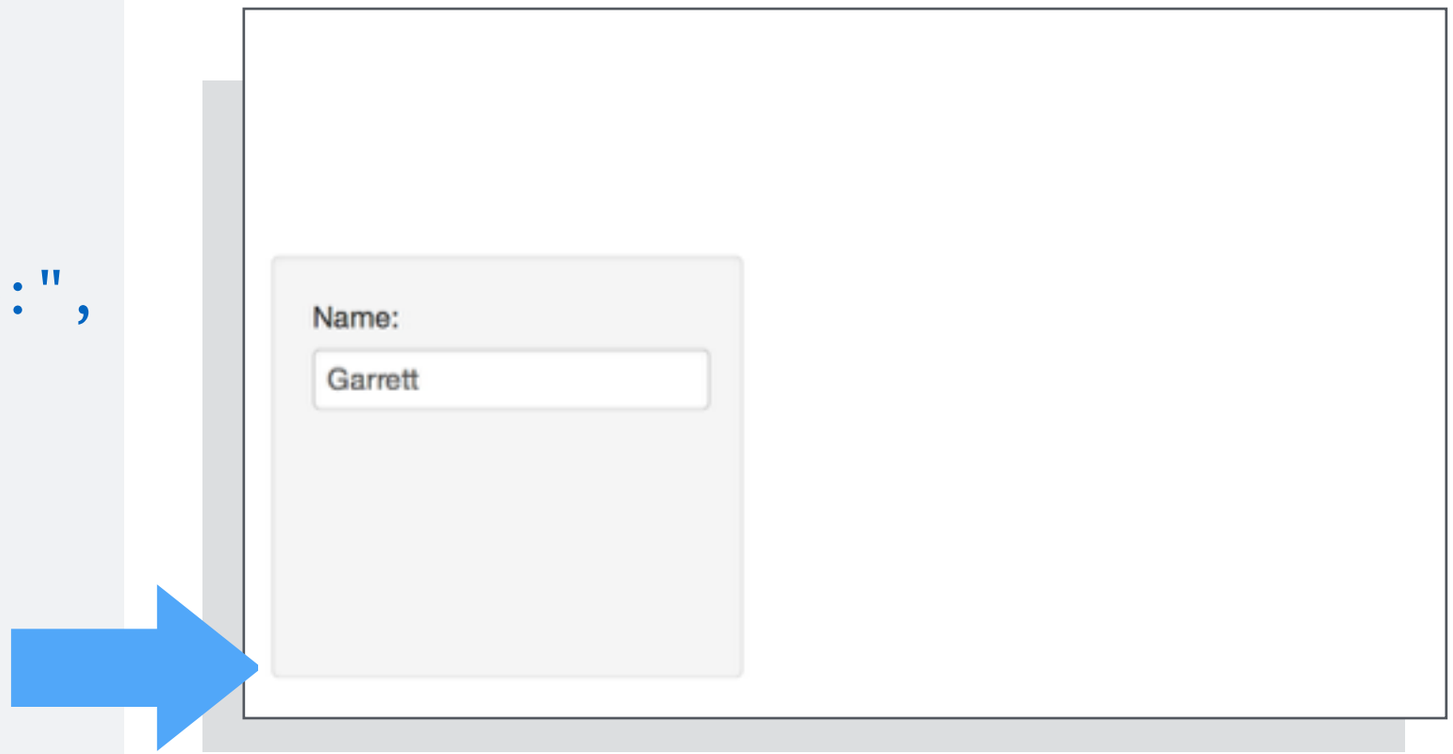
```
shinyUI(fluidPage(
  titlePanel(""),
  sidebarLayout(
    sidebarPanel(
      "one",
      "two",
      "three"),
    mainPanel()
  )
))
```



Adding widgets

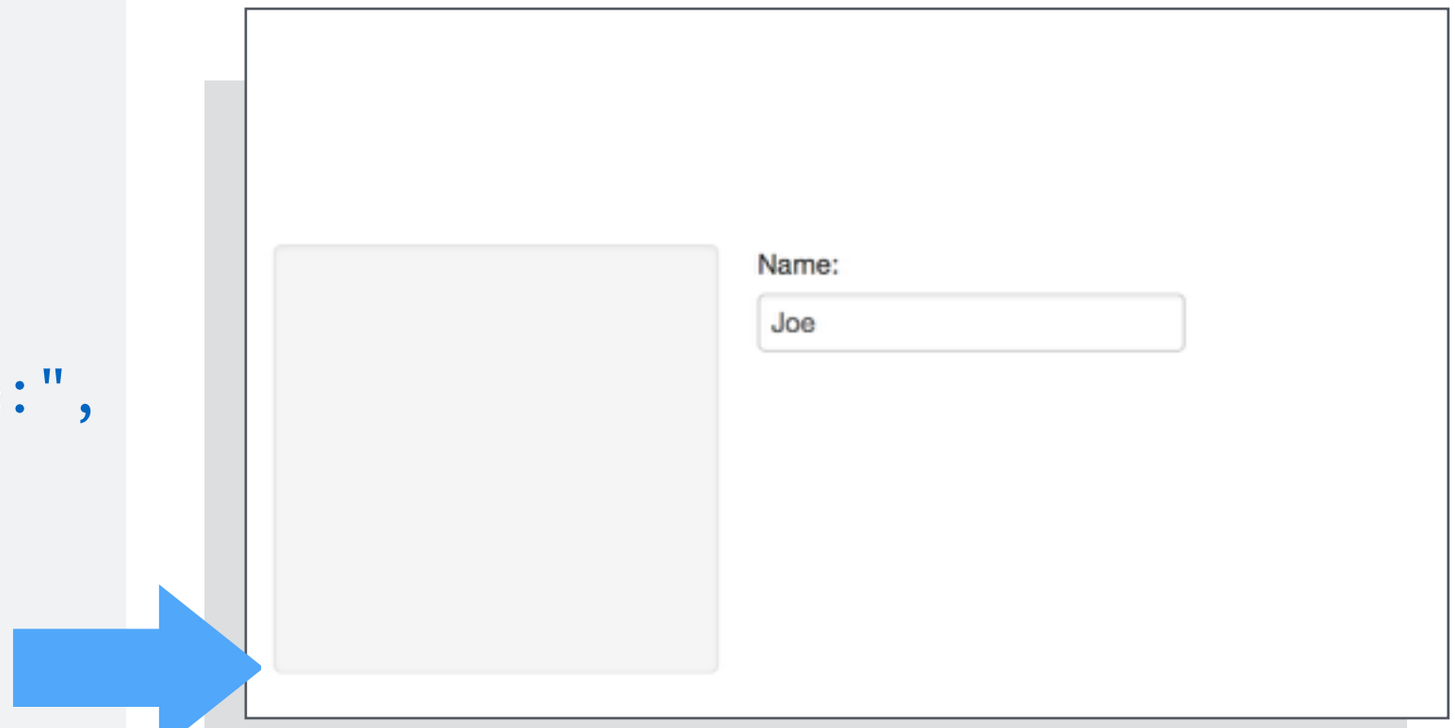
Place your widget functions where you want the widgets to appear. (*titlePanel is not an option*)

```
shinyUI(fluidPage(
  titlePanel(""),
  sidebarLayout(
    sidebarPanel(
      textInput("name","Name:",
        value = "Joe")
    ),
    mainPanel()
  )
))
```



Place your widget functions where you want the widgets to appear. (*titlePanel is not an option*)

```
shinyUI(fluidPage(
  titlePanel(""),
  sidebarLayout(
    sidebarPanel(),
    mainPanel(
      textInput("name","Name:",
        value = "Joe")
    )
  )
))
```



Adding output

ui.R

Use an *Output function to place each element of output in a panel.

```
plotOutput("trend")
```

a *Output function that matches the type of object

name of element (set in server.R)

Function	Inserts
`dataTableOutput`	an interactive table
`htmlOutput`	rawHTML
`imageOutput`	image
`plotOutput`	plot
`tableOutput`	table
`textOutput`	text
`uiOutput`	a Shiny UI element
`verbatimText`	text

server.R

```
counties <- readRDS("data/
counties.rds")
source("helpers.R")

shinyServer(function(input, output) {
 output$table <-renderDataTable({
   counties
 })

 output$map <- renderPlot({
   percent_map(var = counties$asian,
     color = "magenta",
     legend.title = "Percent Asian",
     min = 0, max = 10)
 })
})
```

ui.R

```
shinyUI(fluidPage(
  titlePanel("censusVis"),
  sidebarLayout(
    sidebarPanel(
      helpText("Create demographic
        maps with information from the
        2010 US Census."),
      selectInput("var", label =
       "Choose a variable to display",
        choices = c("Percent White",
          "Percent Black",
          "Percent Hispanic",
          "Percent Asian",
          "Percent Other"),
        selected = "Percent White"),
     sliderInput("range",
       label = "Range of interest:",
       min=0,max=100,value =c(0,100)),
     img(src = "bigorb.png", height =
       50, width = 50),
```

server.R

```
counties <- readRDS("data/
counties.rds")
source("helpers.R")

shinyServer(function(input, output) {
 output$table <-renderDataTable({
   counties
 })

 output$map <- renderPlot({
   percent_map(var = counties$asian,
     color = "magenta",
     legend.title = "Percent Asian",
     min = 0, max = 10)
 })
})
```

```
    choices = c("Percent White",
      "Percent Black",
      "Percent Hispanic",
      "Percent Asian",
      "Percent Other"),
    selected = "Percent White"),
  sliderInput("range",
    label = "Range of interest:",
    min=0,max=100,value =c(0,100)),
   img(src = "bigorb.png", height =
     50, width = 50),
  "This was made with",
    a("Shiny", href =
      "shiny.rstudio.com"), "."
  ),
  mainPanel(
    plotOutput("map"),
    hr(),
    dataTableOutput("table")
  )
 )
))
```

server.R

```
counties <- readRDS("data/
counties.rds")
source("helpers.R")

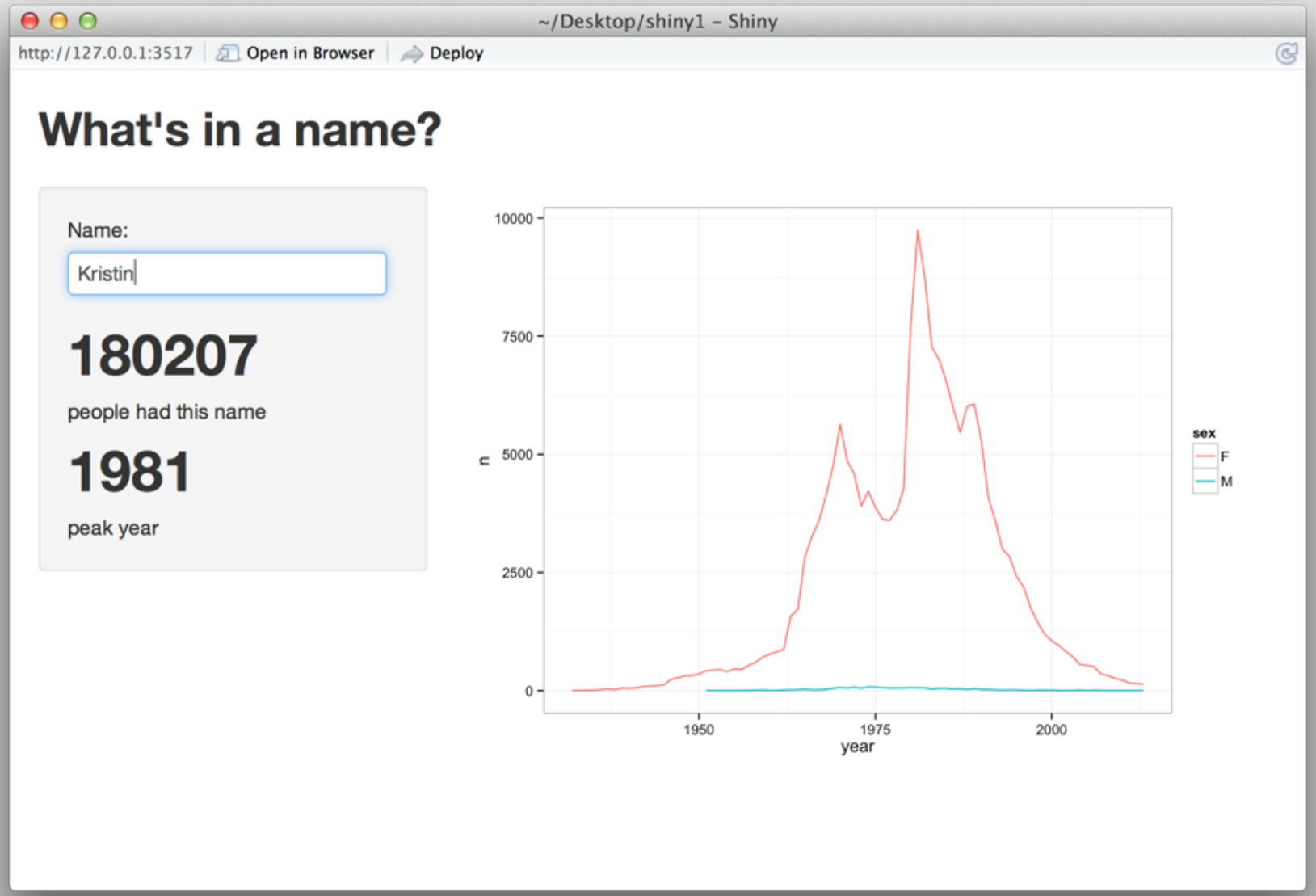
shinyServer(function(input, output) {
 output$table <-renderDataTable({
   counties
 })

 output$map <- renderPlot({
   percent_map(var = counties$asian,
     color = "magenta",
     legend.title = "Percent Asian",
     min = 0, max = 10)
 })
})
```

```
    choices = c("Percent White",
      "Percent Black",
      "Percent Hispanic",
      "Percent Asian",
      "Percent Other"),
    selected = "Percent White"),
  sliderInput("range",
    label = "Range of interest:",
    min=0,max=100,value =c(0,100)),
   img(src = "bigorb.png", height =
     50, width = 50),
  "This was made with",
      a("Shiny", href =
        "shiny.rstudio.com"), "."
    ),
    mainPanel(
      plotOutput("map"),
      hr()
      dataTableOutput("table")
  ))
))
```

Your turn

Modify your ui.R file to recreate the web app.



Tip: You can increase the size of text (or textOutput(…)) by surrounding it with h1(), e.g. h1("Header One size")

```
shinyUI(fluidPage(

  titlePanel("What's in a name?"),

  sidebarLayout(
    sidebarPanel(
      textInput("name", "Name:", value = "Garrett"),
      h1(textOutput("total")),
      "people had this name",
      h1(textOutput("peak")),
      "peak year"
    ),
    mainPanel(
      plotOutput("trend")
    )
  )
))
```

```
shinyUI(fluidPage(

  titlePanel("What's in a name?"),

  sidebarLayout(
    sidebarPanel(
      textInput("name", "Name:", value = "Garrett"),
      h1(textOutput("total")),
      "people had this name",
      h1(textOutput("peak")),
      "peak year"
    ),
    mainPanel(
      plotOutput("trend")
    )
  )
))
```

HTML elements

Shiny HTML tag functions

h1() ~ <h1></h1>
a ~ <a>

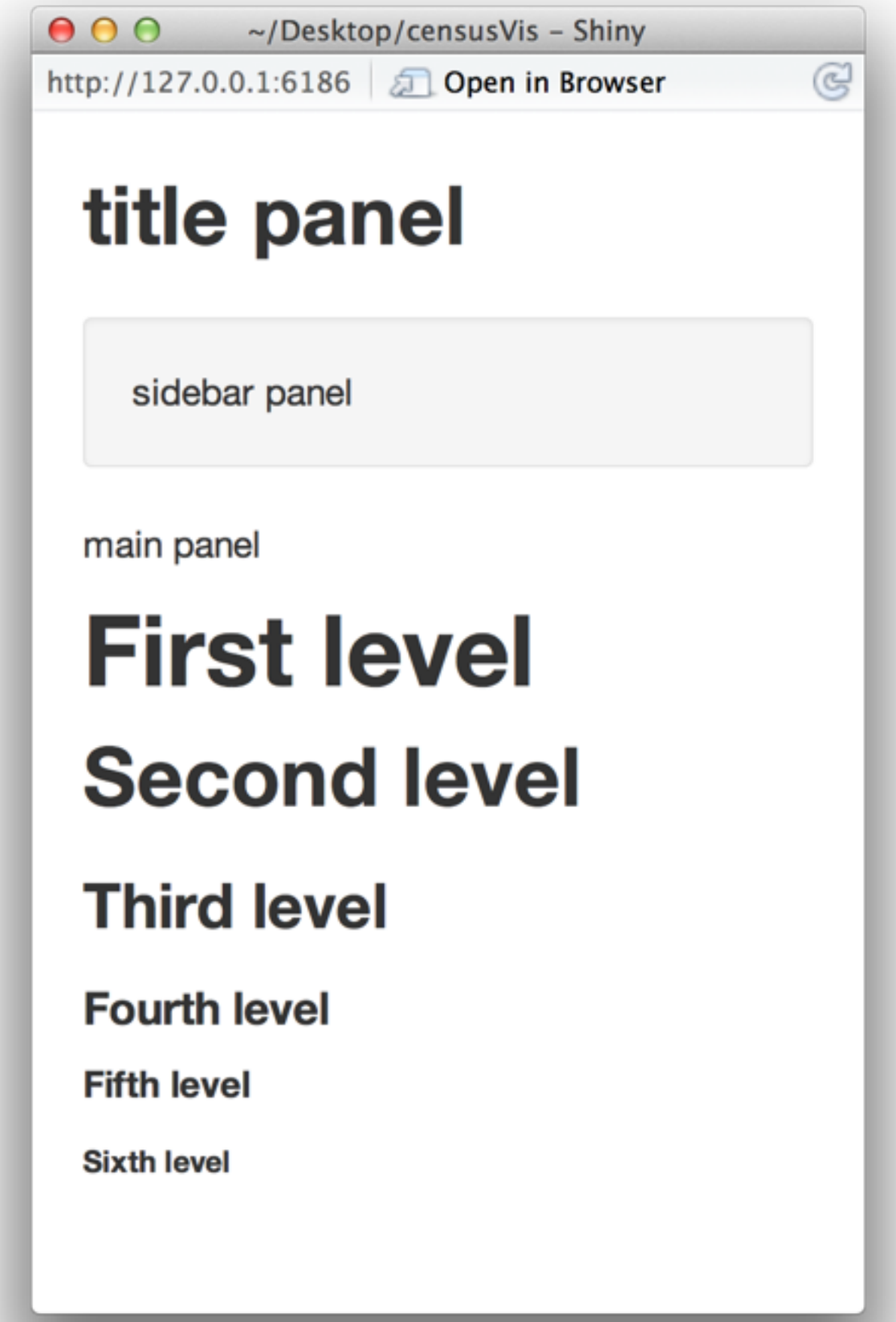
http://shiny.rstudio.com/articles/html-tags.html

Shiny provides R functions that recreate common HTML tags. You can find a list of these functions and how to use them to enhance your app at the link above.

h1 - h6

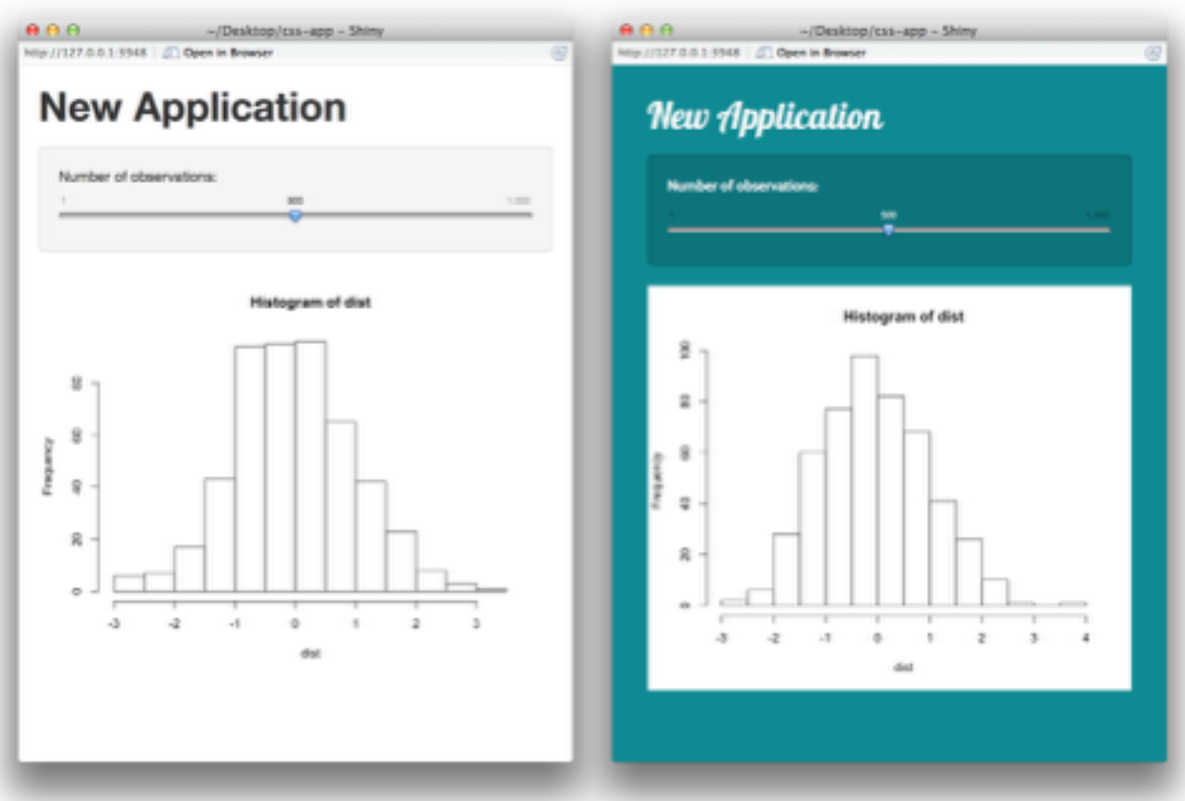
Headers

```
shinyUI(fluidPage(
  titlePanel("title panel"),
  sidebarLayout(
    sidebarPanel("sidebar panel"),
    mainPanel("main panel",
      h1("First level"),
      h2("Second level"),
      h3("Third level"),
      h4("Fourth level"),
      h5("Fifth level"),
      h6("Sixth level"))
  )
))
```



Customize your apps with HTML, CSS, and Javascript

http://shiny.rstudio.com/articles/css.html



You can pair any app with whatever web technologies you wish. The above guide explains how to style your app with CSS.

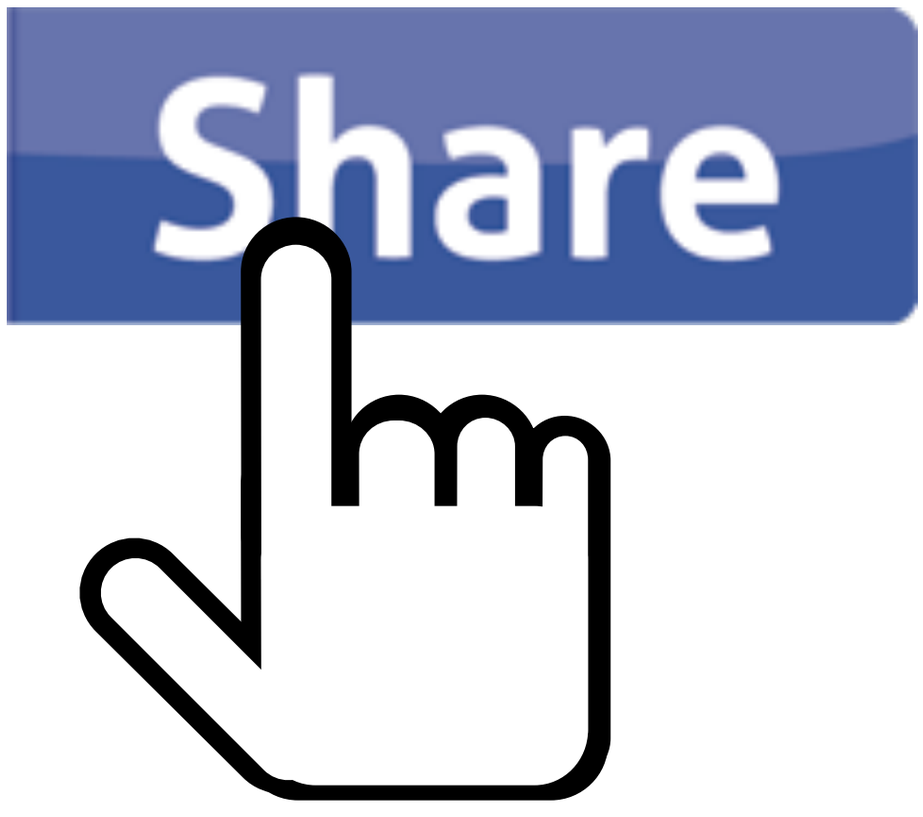
Sharing

These functions can help users run your server.R and ui.R files on their own computer.

```
runApp()

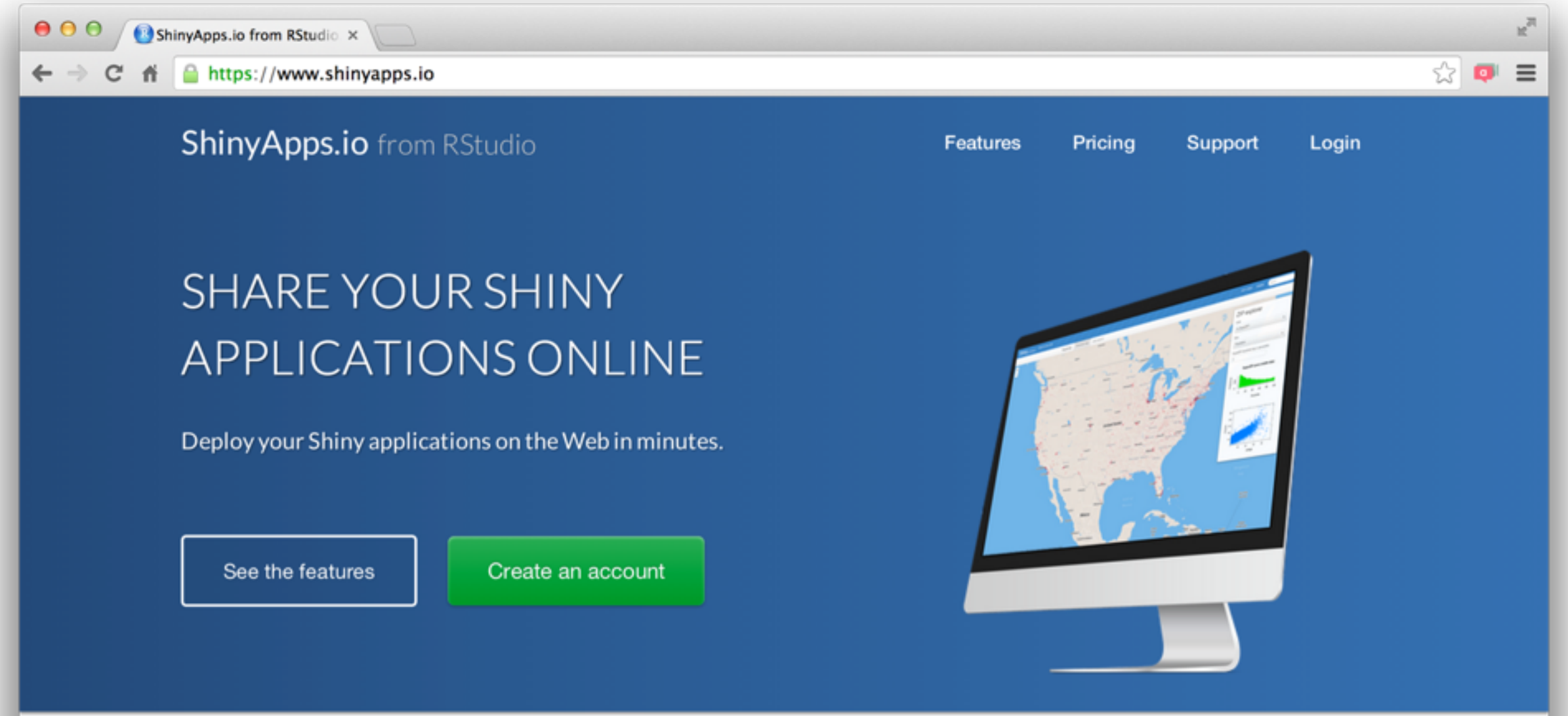
runGitHub()

runGist()
```



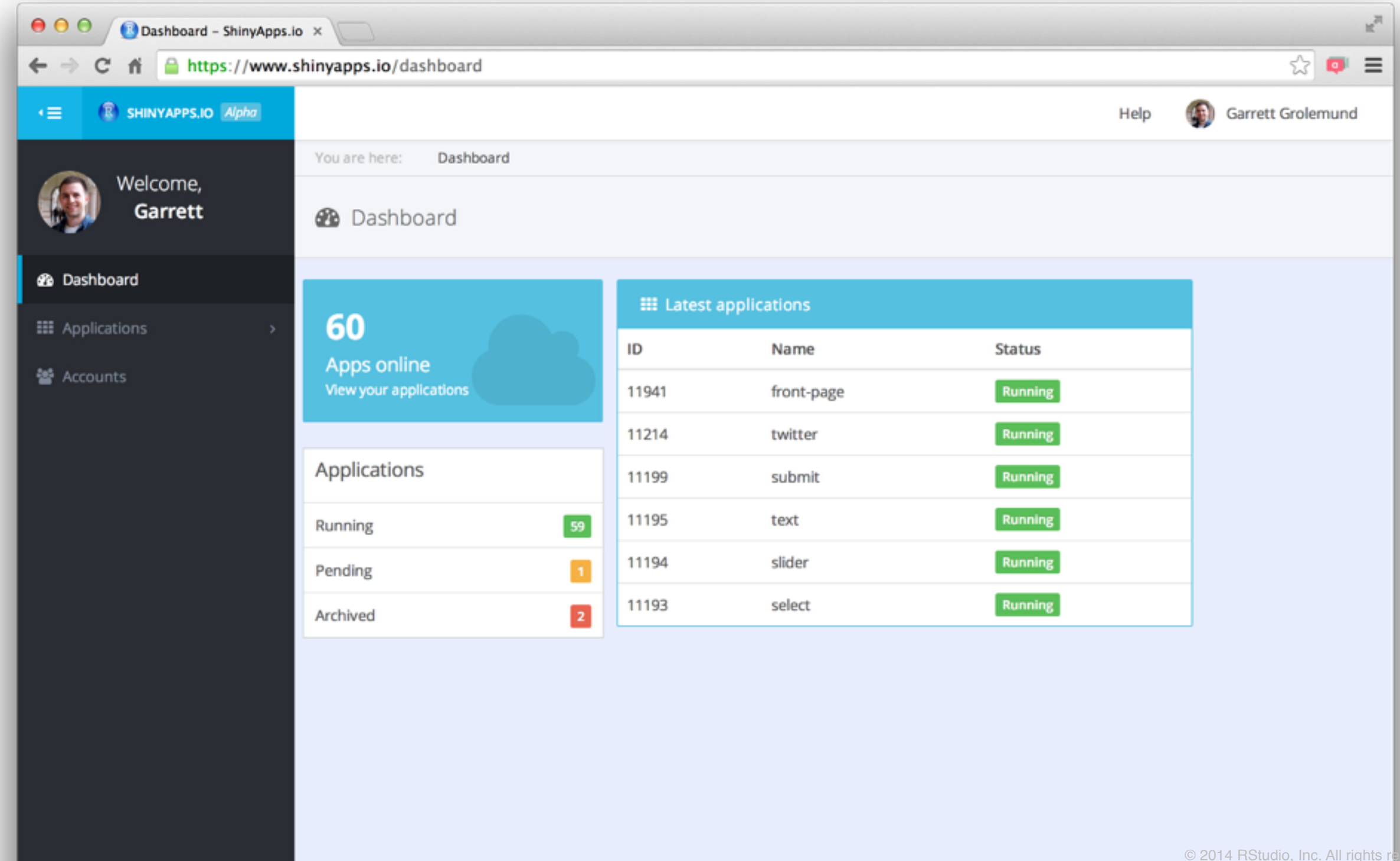
ShinyApps.io

Hassle-free cloud hosting for Shiny



Studio

Hassle-free cloud hosting for Shiny



ShinyApps.io

- Each app is hosted at its own URL on server run by RStudio
- Administrative tools, etc.
- Designed to host Shiny apps

Get Started

Guide at:
http://shiny.rstudio.com/articles/shinyapps.html

1. install `shinyapps` package
2. create account at www.shinyapps.io
3. Run `setAccountInfo()` at command line

`deployApp()` or click button to host app online

Shiny Server (Pro)

Shiny Server

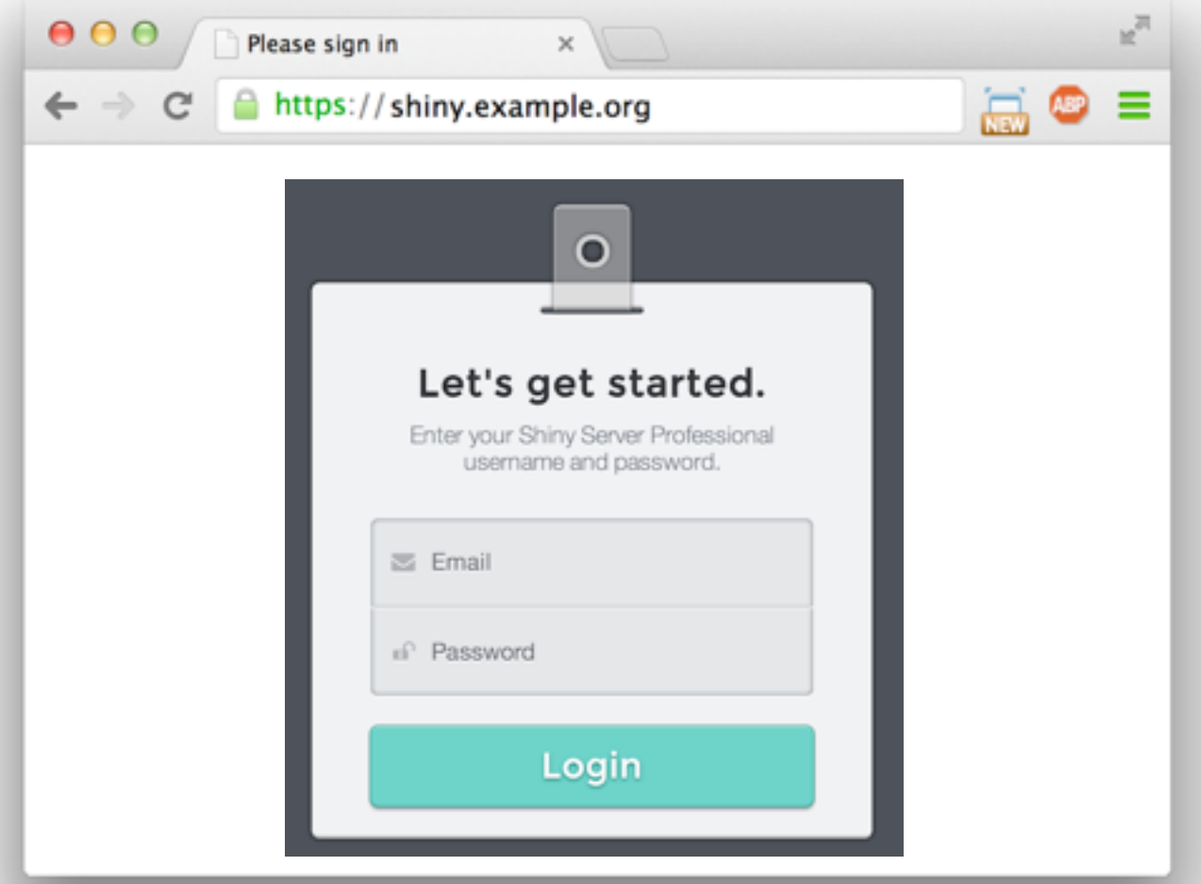
A back end program that builds a web server specifically designed to host Shiny apps

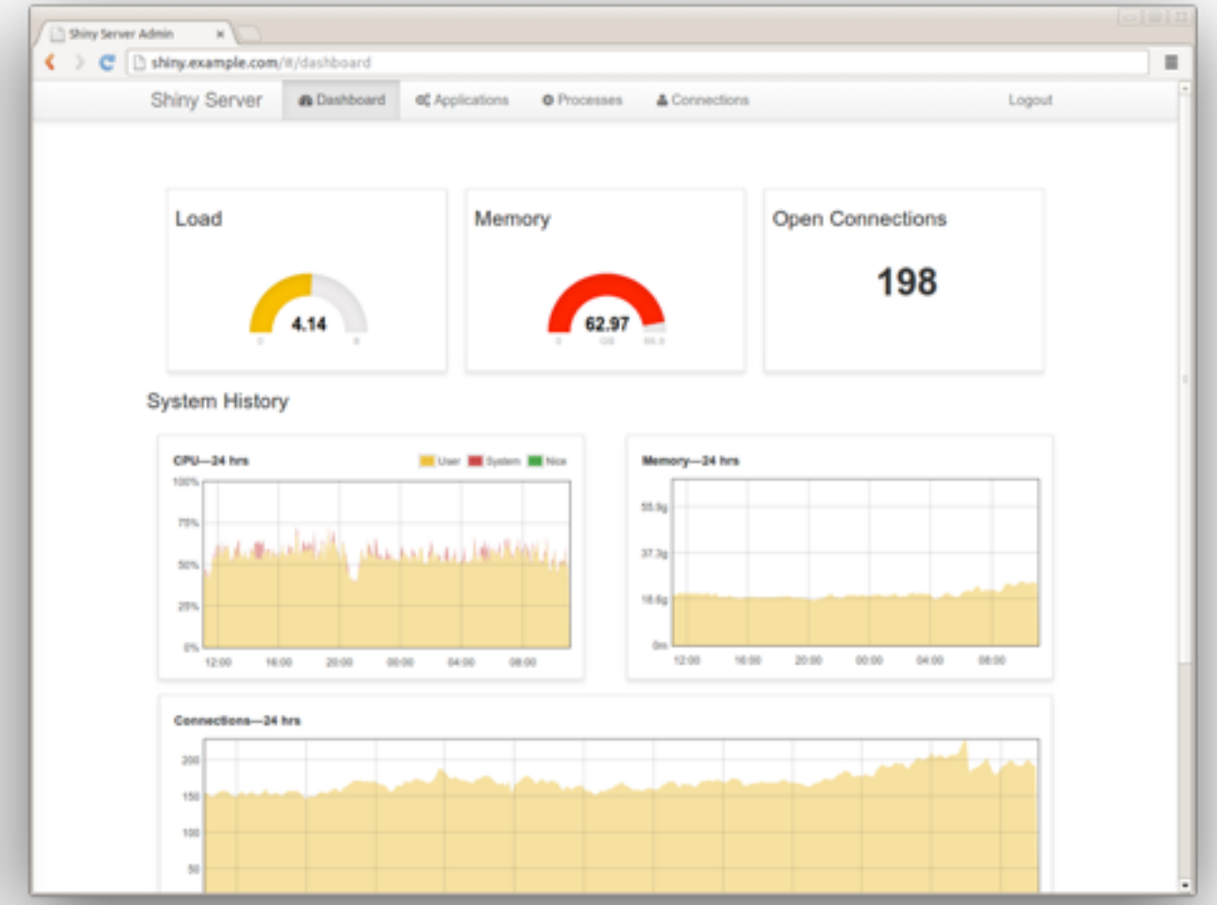
- Each app is hosted at its own URL
- Can deploy to internet, or within a controlled environment
- Starts app when user visits, closes app when user leaves
- Runs on a linux server

http://shiny.rstudio.com/articles/shiny-server.html

Shiny Server Pro

- Password protect apps, SSL
- Administrative tools, priority support





http://rstudio.com/shiny/server

		Open Source Edition	Professional Edition
General	Deploy Shiny applications to the Internet	•	•
	Move computation close to the data	•	•
	Host multiple applications on a single server	•	•
	Deploy Shiny applications behind firewalls	•	•
	Custom page templates	•	•
Security and authentication	Password file authentication		•
	LDAP and Active Directory authentication		•
	Google authentication (OAuth2)		•
	PAM authentication & sessions		•
	Group based authorization		•
	SSL support		•
Tuning and scaling	Scale applications across multiple processes		•
	View and manage active sessions		•
	Allocate resources on a per application basis		•
	Define application concurrency limits		•
Server monitoring	System performance and resource metrics		•
	Per application performance and resource metrics		•
	Application usage metrics		•
	Customizable health check end point		•

* For volume discounts, OEMs, or additional capacity for larger audiences please email us at sales@rstudio.com.

RStudio

http://www.rstudio.com/shiny/server/pro

Shiny Server Pro Edition

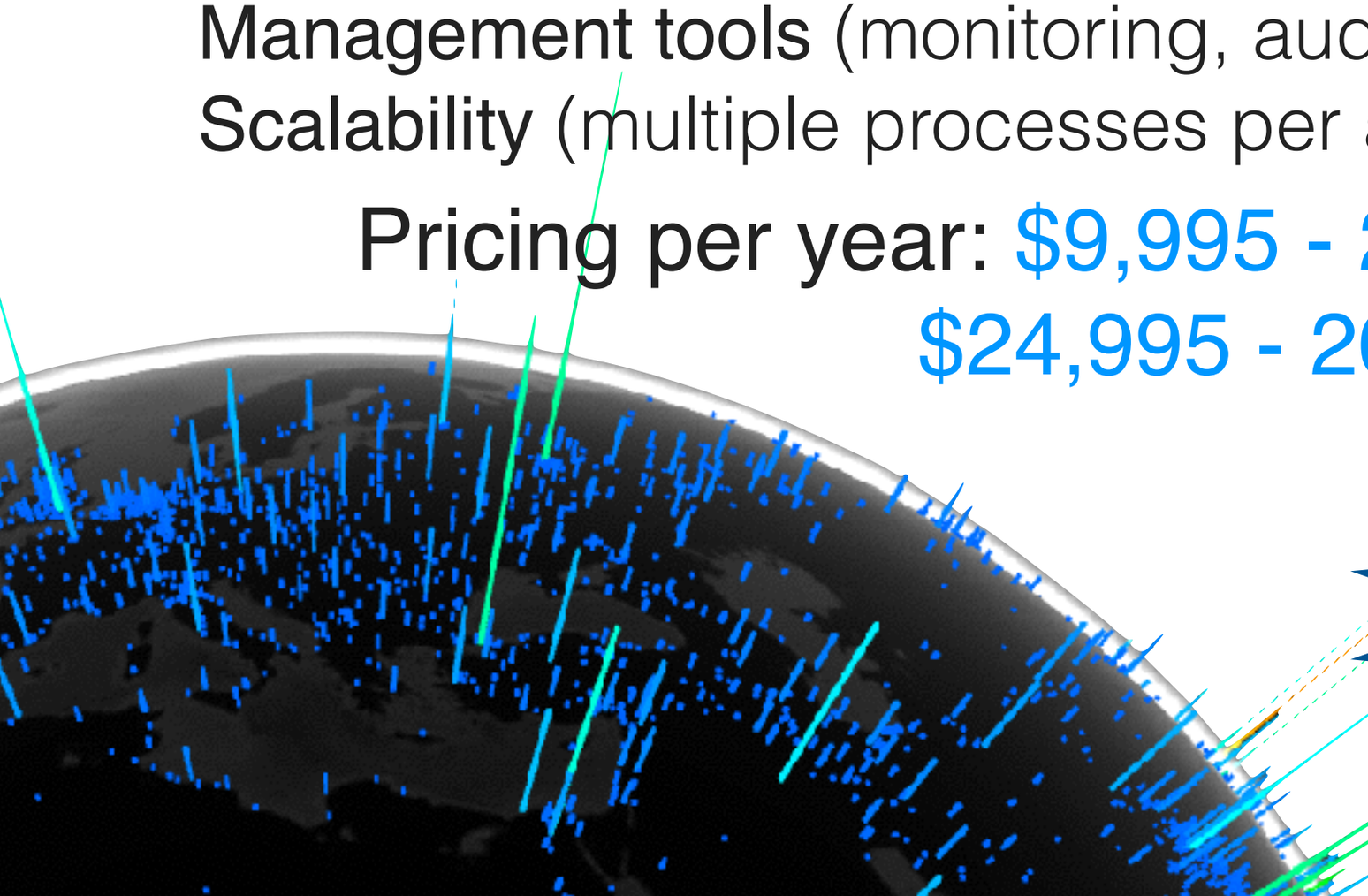
Security tools (authentication, encryption)

Management tools (monitoring, auditing, allocation)

Scalability (multiple processes per application)

Pricing per year: $9,995 - 20 concurrent sessions

$24,995 - 200 concurrent sessions



Academic and Small Business discounts available

How to learn more

Shiny

by RStudio

A web application framework for R

Turn your analyses into interactive web applications

No HTML, CSS, or JavaScript knowledge required

TUTORIAL ARTICLES GALLERY REFERENCE DEPLOY HELP

Get inspired
(gallery)

Get started
(tutorial)

Go deeper
(articles)

http://shiny.rstudio.com

Centralized learning site

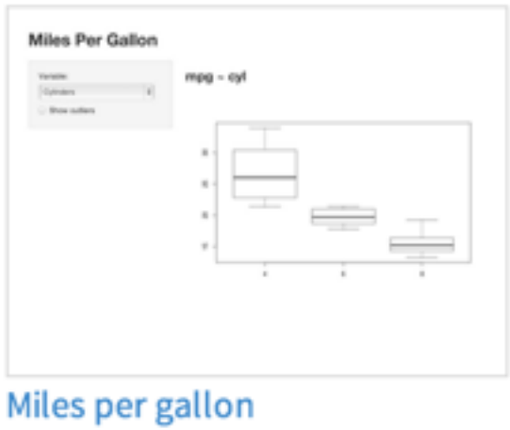
Gallery

Start simple

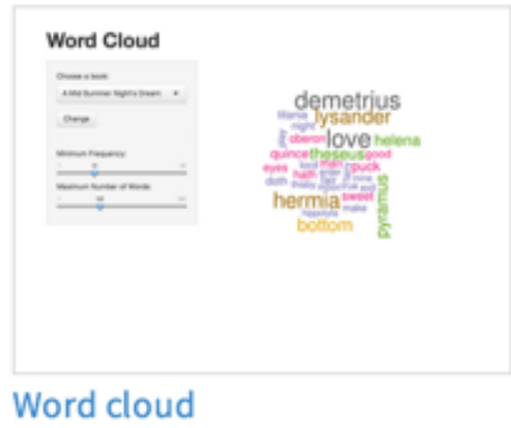
If you're new to Shiny, these simple but complete applications are designed for you to study.

Kmeans example

Telephones by region



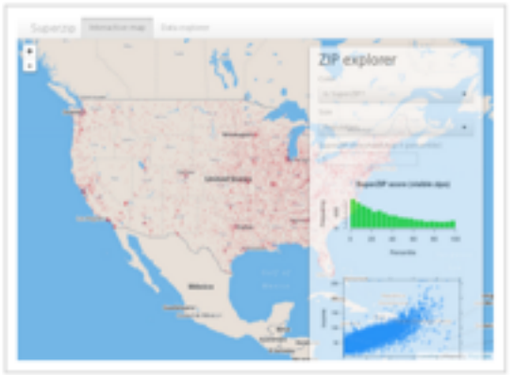
Miles per gallon



Word cloud

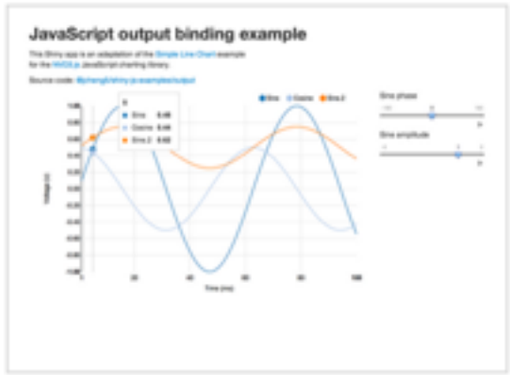
Interactive visualizations

Shiny is designed for fully interactive visualization, using JavaScript libraries like d3, Leaflet, and Google Charts.

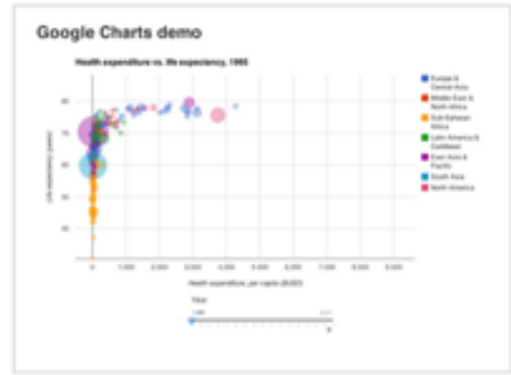


SuperZip example

Movie explorer



NVD3 line chart output



Google Charts

Widgets

Get to know many of the input and output widgets that are available in Shiny with these examples.

Teach yourself Shiny

Who should take the tutorial?

You will get the most out of this tutorial if you already know how to program in R, but not Shiny.

If R is new to you, you may want to check out the learning resources at www.rstudio.com/training before taking this tutorial. If you are not sure whether you are ready for Shiny, try our quiz.

If you use Shiny on a regular basis, you may want to skip this tutorial and visit the articles section of the Development Center. In the articles section, we cover individual Shiny topics at an advanced level.

Get started with Shiny

With this seven lesson tutorial, you will move up from R programmer to Shiny developer. Each lesson takes about 15 minutes and teaches one new Shiny skill. By the end of the lessons, you will know how to build and deploy a Shiny app.

Each lesson includes an exercise. Don't skip the exercises, even if you are tempted to get to the next lesson. The learning occurs in the exercises. How do we know? Because we designed the tutorial and organized the material around the exercises.

Click the Lesson 1 button to get started and say hello to Shiny!

- Lesson 1 - Welcome to Shiny
- Lesson 2 - Layout the user interface
- Lesson 3 - Add control widgets
- Lesson 4 - Display reactive output
- Lesson 5 - Use R scripts and data
- Lesson 6 - Use reactive expressions
- Lesson 7 - Share your apps

Continue to lesson 1

Articles

The basics

If you've been through the tutorial and need a refresher, these articles are a good place to start. They describe the lay of the land.

The basic parts of a Shiny app
How to build a Shiny app
How to launch a Shiny app
How to get help
The Shiny Cheat sheet

Layouts and UI

These articles explain how to control the layout, user-interface, and general appearance of your Shiny apps.

Application layout guide
Display modes
Tabsets
Customize your UI with HTML
Build your entire UI with HTML
Build a dynamic UI that reacts to user input
Shiny HTML Tags Glossary

Widgets

These articles describe Shiny's pre-built widgets and provide ideas on how to use them. (See also Lesson 3 in the tutorial, and the Widgets section in the gallery.)

Using sliders
Help users download data from your app
Using selectize input

Outputs

These articles show you how to create and use different output objects, the parts of your app that display results and react to user input.

Render images in a Shiny app
How to use DataTables in a Shiny App

Deploying apps

These articles describe the different ways to share your Shiny apps with users.

Getting started with ShinyApps.io
Introduction to Shiny Server
Deploying Shiny apps over the web
Sharing apps to run locally

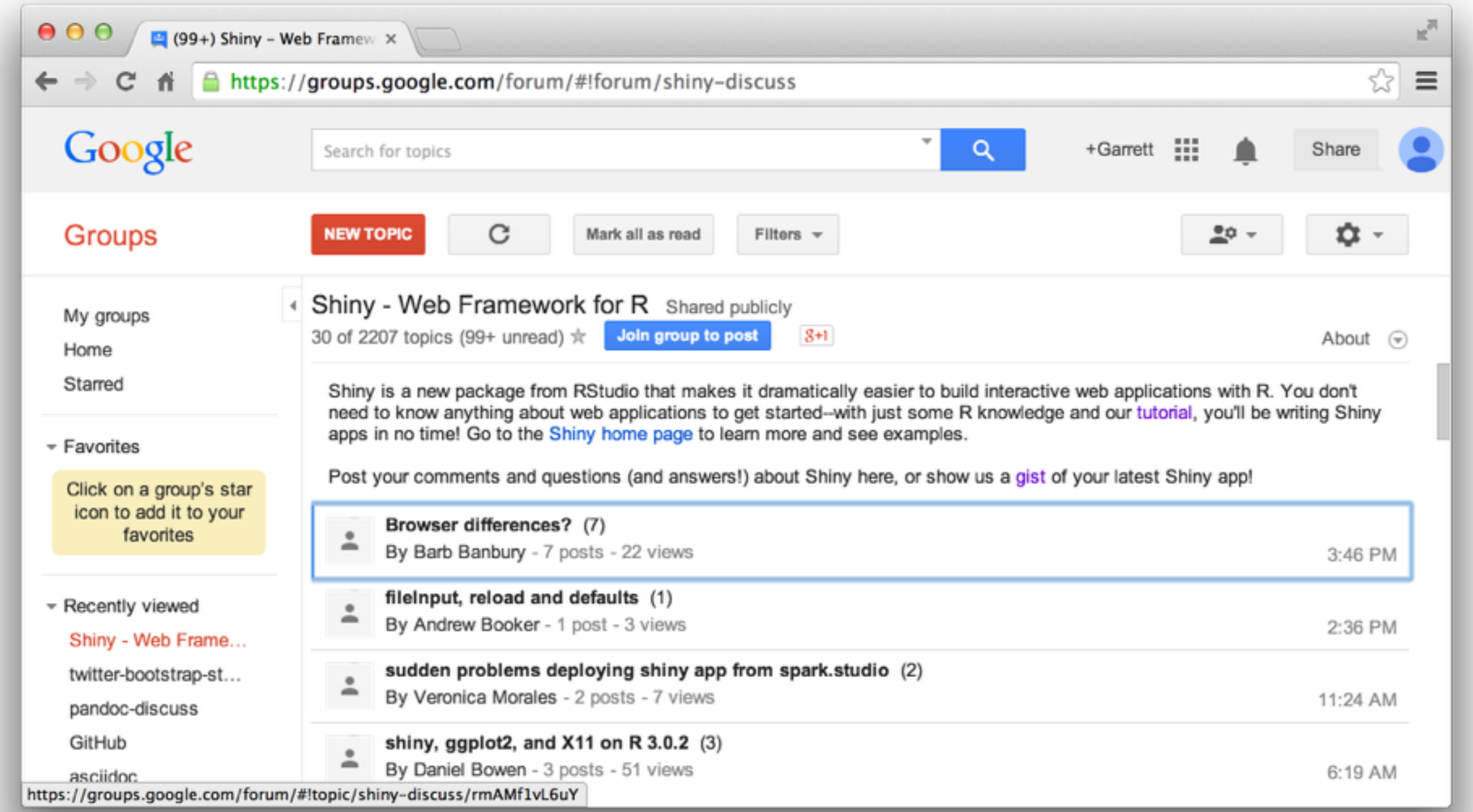
Reactive programming

These articles describe reactivity from a conceptual level. Understanding reactivity will help you build apps that are more efficient, robust, and correct.

Reactivity: An overview
Stop reactions with isolate()
Execution scheduling

Shiny Discussion group

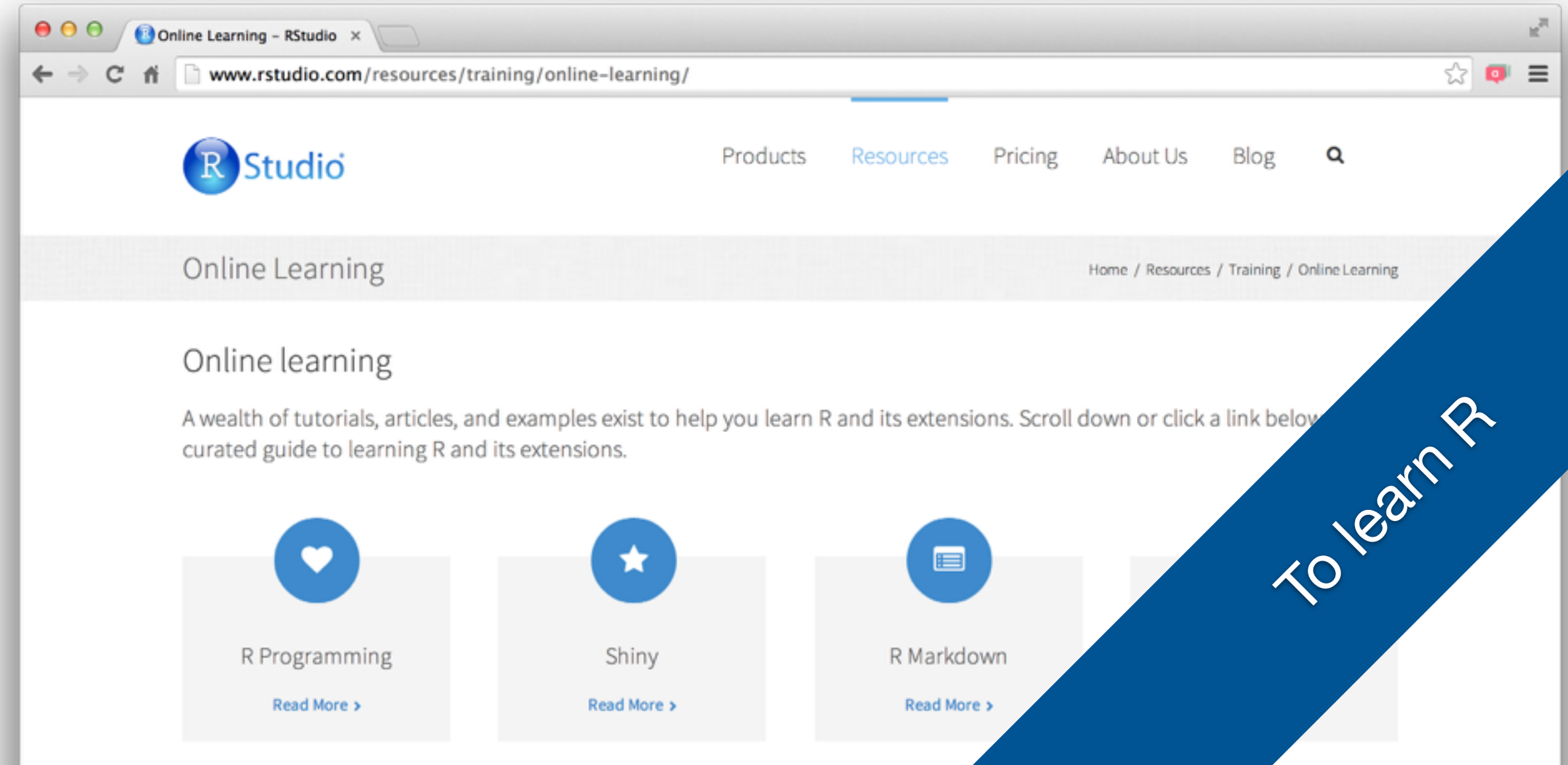
https://groups.google.com/forum/#!forum/shiny-discuss



RStudio training

www.rstudio.com/resources/training/online-learning.html

Links to recommended tutorials, articles, examples to help you learn R and its extensions



Codecademy

http://www.codecademy.com

Free tutorials that teach HTML, CSS, and Javascript with instant feedback. I recommend this for learning the basics of these languages.