250 Northern Ave, Boston, MA 02210
Phone: 844-448-1212

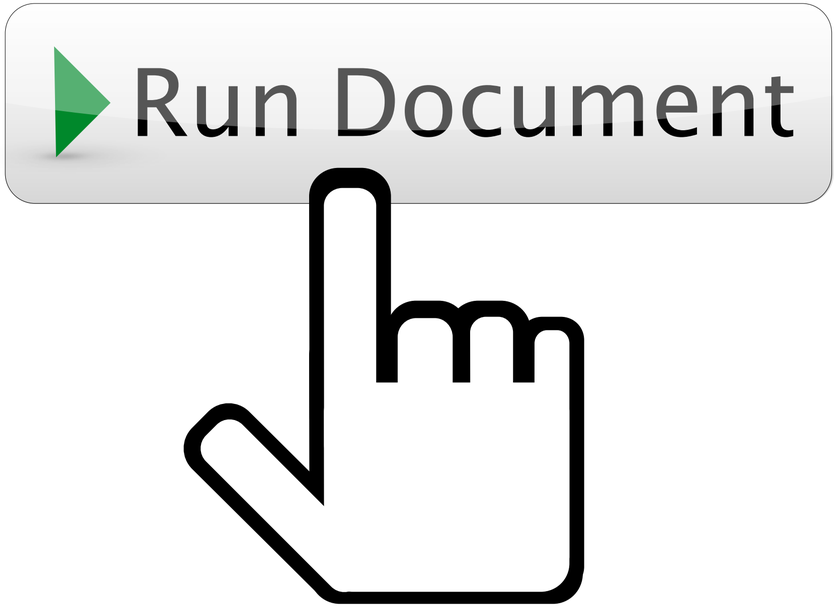
Email: info@rstudio.com
Web: http://www.rstudio.com

250 Northern Ave, Boston, MA 02210
Phone: 844-448-1212

Email: info@rstudio.com
Web: http://www.rstudio.com

Shiny and R Markdown

Interactive apps and reproducible reports from R



Garrett Grolemund

Data Scientist and Master Instructor
June 2015
Email: garrett@rstudio.com

Follow @rstudio

R Markdown Resources

File > New File > R Markdown…
http://rmarkdown.rstudio.com/
http://www.rstudio.com/resources/cheatsheets/
http://rmarkdown.rstudio.com/developer_document_templates.html
http://rmarkdown.rstudio.com/developer_parameterized_reports.html

htmlwidgets Resources

htmlwidgets.org
http://www.htmlwidgets.org/showcase_leaflet.html

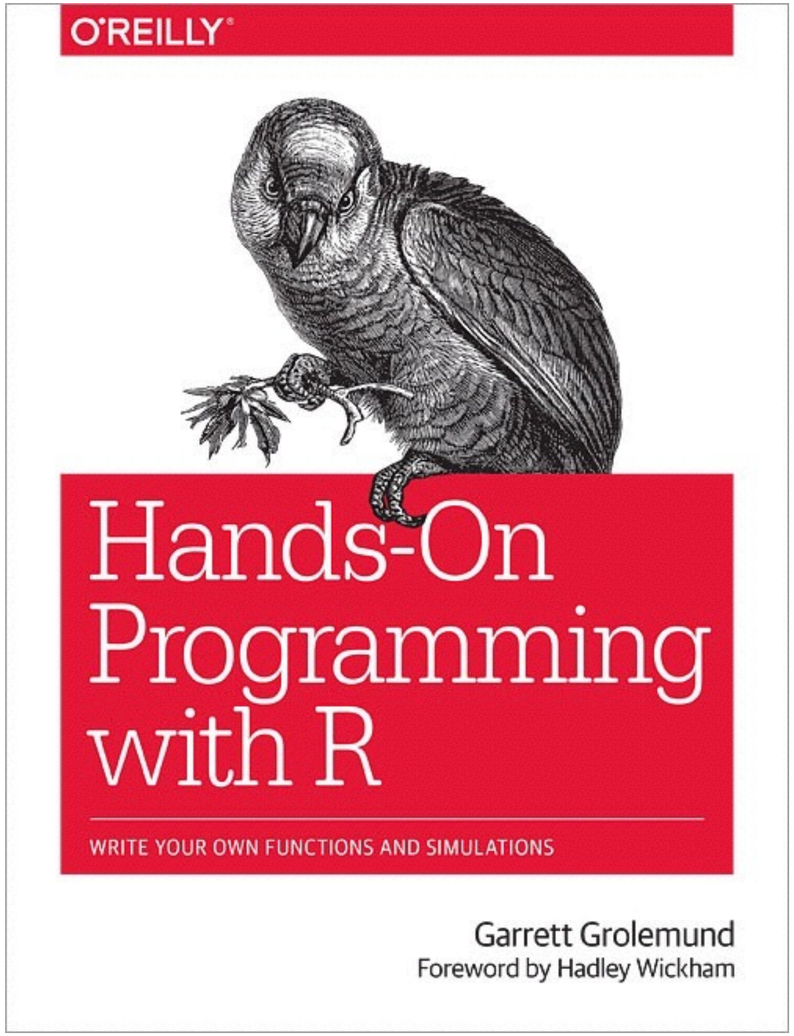
Shiny Resources

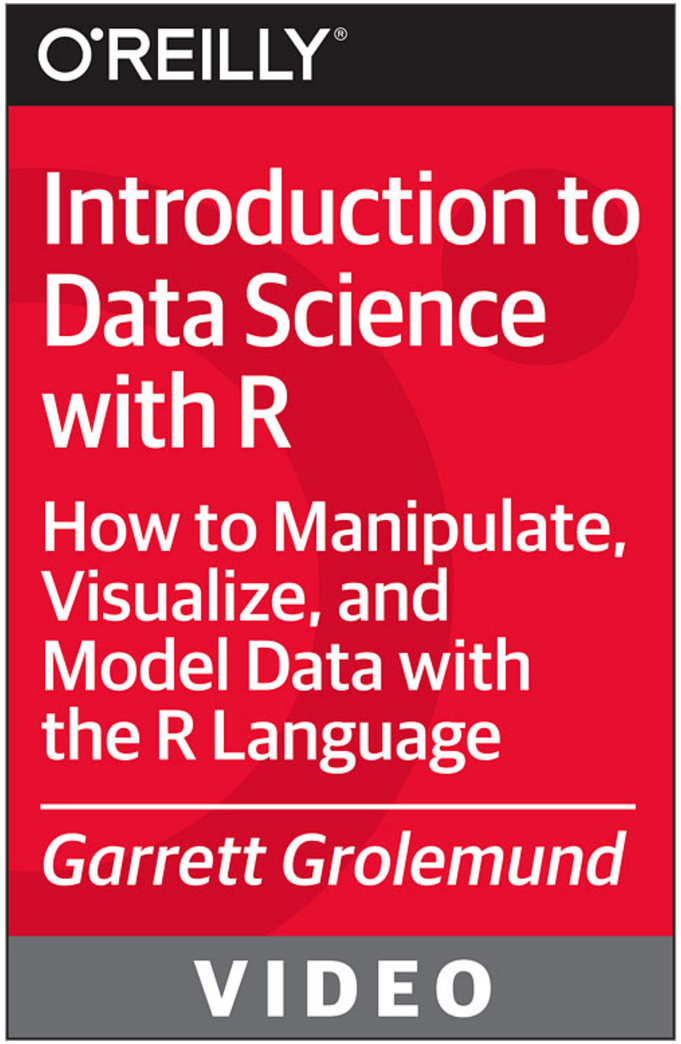
shiny.rstudio.com/articles
shiny.rstudio.com/tutorial
http://www.rstudio.com/resources/cheatsheets/

Shiny Demos

http://www.rstudio.com/products/shiny/shiny-user-showcase/
http://shiny.rstudio.com/gallery/

http://webpopix.org:8080/dashboard/absorption/
http://shiny.rstudio.com/gallery/widget-gallery.html
http://rstudio.github.io/shinydashboard/
https://gallery.shinyapps.io/marketing/
https://gallery.shinyapps.io/LDAelife/
http://shiny.rstudio.com/gallery/superzip-example.html
https://gallery.shinyapps.io/EDsimulation/
http://shiny.rstudio.com/articles/#interactive-plots
http://shiny.rstudio.com/gallery/authentication-and-database.html
https://gallery.shinyapps.io/ga-effect/
http://gallery.shinyapps.io/087-crandash





DataCamp

- Reporting with R Markdown
- Data Visualization in R with ggvis
- Data Manipulation in R with dplyr

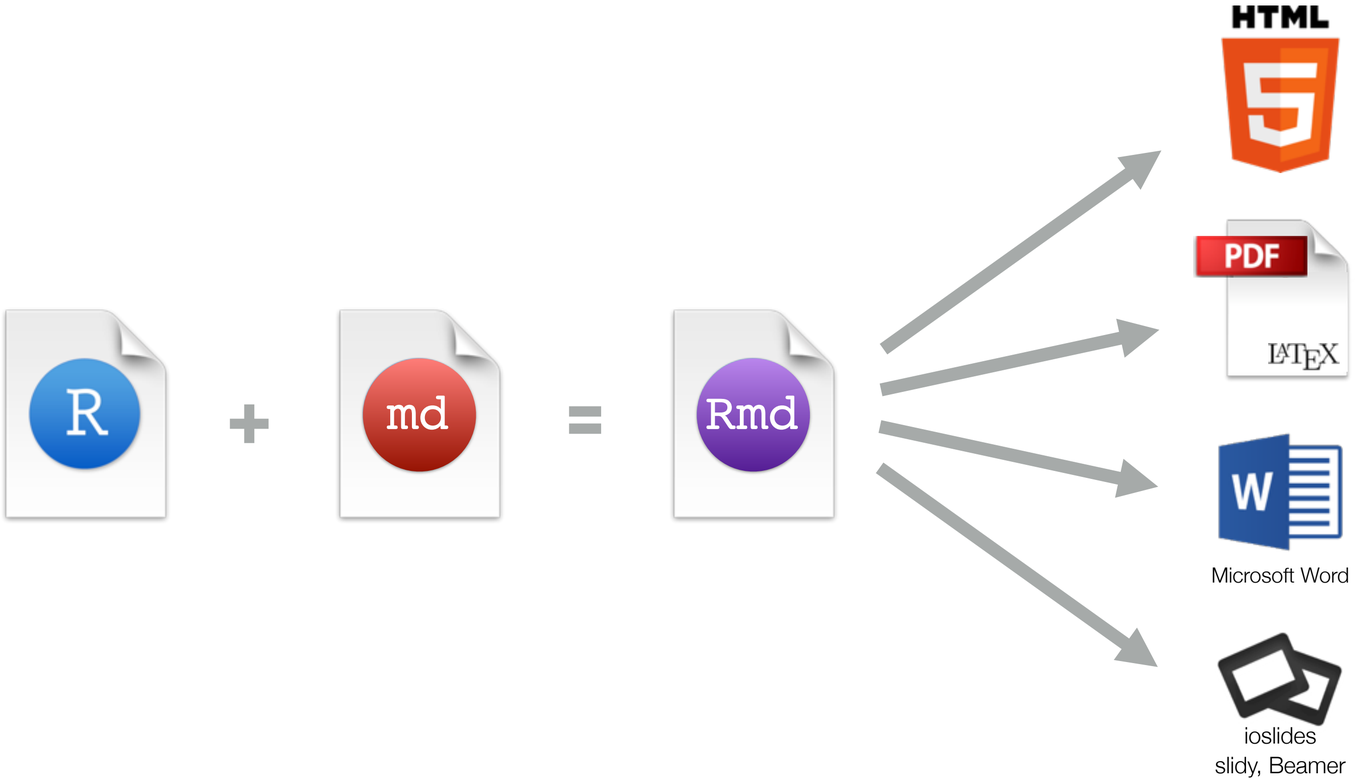
R Markdown

Can we do better?

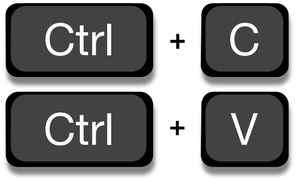
Ctrl + C (Copy)

Ctrl + V (Paste)

R Markdown

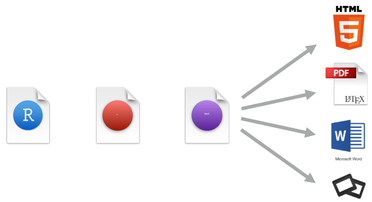


Recap: R Markdown

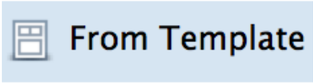


Reproducible

Automatic



Flexible



Reusable

```
params:
```

Parameterizable

Teach yourself R Markdown

The R Markdown Development Center

rmarkdown.rstudio.com

R Markdown — Dynamic D ×

Garrett

rmarkdown.rstudio.com

R Markdown v2 Home Authoring Formats Developer

R Markdown — Dynamic Documents for R

R Markdown is an authoring format that enables easy creation of dynamic documents, presentations, and reports from R. It combines the core syntax of markdown (an easy-to-write plain text format) with embedded R code chunks that are run so their output can be included in the final document. R Markdown documents are fully *reproducible* (they can be automatically regenerated whenever underlying R code or data changes).

This website describes R Markdown v2, a next generation implementation of R Markdown based on knitr and pandoc. This implementation brings many enhancements to R Markdown, including:

- Many available output formats including HTML, PDF, and MS Word.
- Support for creating Beamer, ioslides, and Slidy presentations.
- New markdown syntax including expanded support for tables and bibliographies.
- Hooks for customizing HTML and PDF output (include CSS, headers, and footers).
- Include raw LaTeX within markdown for advanced customization of PDF output.
- Compile HTML, PDF, or MS Word notebooks from R scripts.
- Extensiblity: create custom templates and even entirely new output formats.
- Create interactive R Markdown documents using Shiny.

Note that PDF output (including Beamer slides) requires an installation of TeX.

Quick Tour

Installation

You can install the R Markdown package from CRAN as follows:

```
install.packages("rmarkdown")
```

The R Markdown Reference Guide

www.rstudio.com/resources/cheatsheets/

R Markdown Reference Guide

Learn more about R Markdown at rmarkdown.rstudio.com
Learn more about Interactive Docs at shiny.rstudio.com/articles

Contents:

Syntax	Becomes

Make a code chunk with three back ticks followed by an r in braces. End the chunk with three back ticks:
``{r}
paste("Hello", "World!")
``
	Make a code chunk with three back ticks followed by an r in braces. End the chunk with three back ticks:
	`paste("Hello", "World!")`
	`## [1] "Hello World!"`

Place code inline with a single back ticks. The first back tick must be followed by an R, like this `r paste("Hello", "World!")`.	Place code inline with a single back ticks. The first back tick must be followed by an R, like this Hello World!.

Add chunk options within braces. For example, `echo=FALSE` will prevent source code from being displayed:
``{r eval=TRUE, echo=FALSE}
paste("Hello", "World!")
``
	Add chunk options within braces. For example, `echo=FALSE` will prevent source code from being displayed:
	`## [1] "Hello World!"`

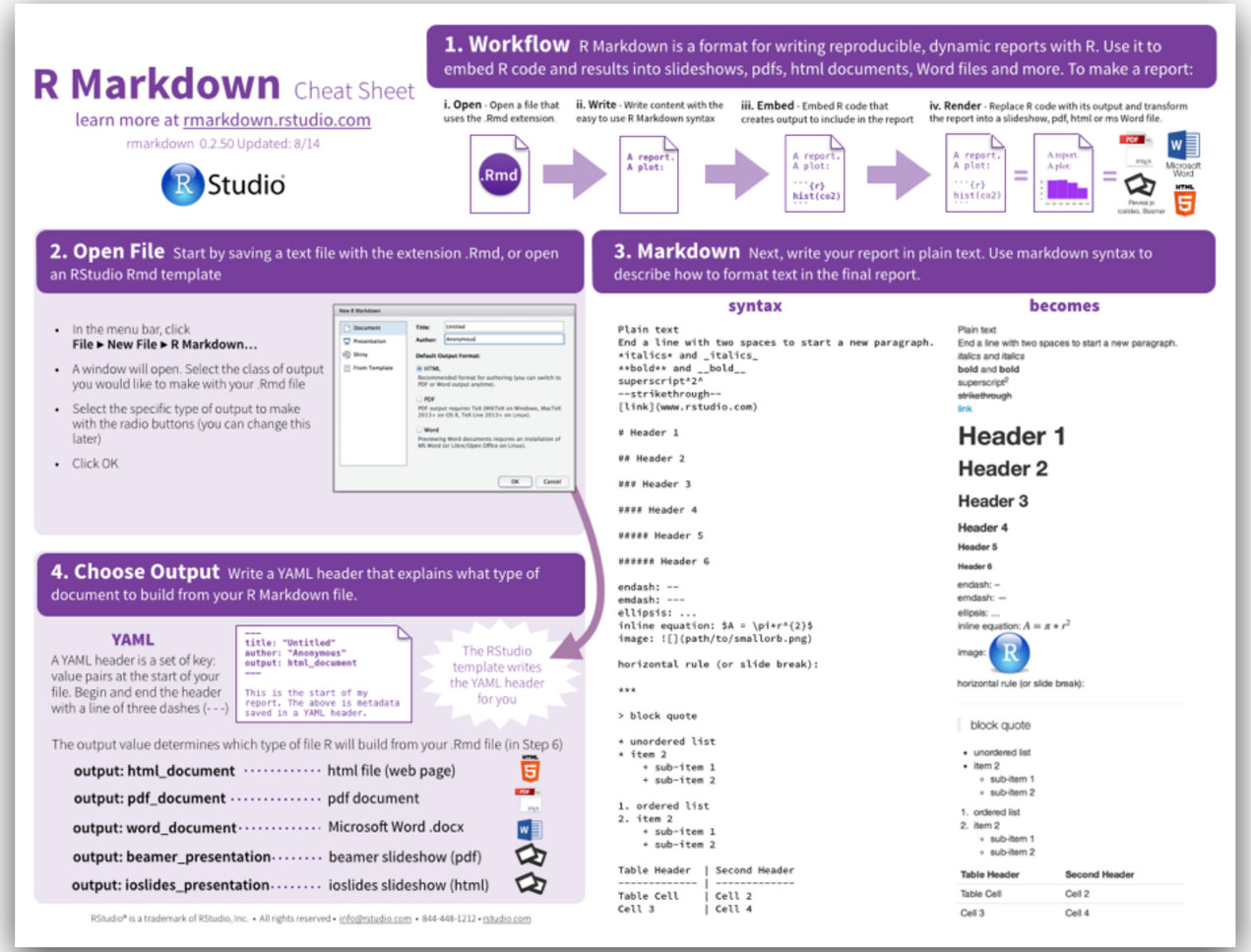
Learn more about chunk options at http://yihui.name/knitr/options

Chunk options

option	default value	description
Code evaluation		
child	NULL	A character vector of filenames. Knitr will knit the files and place them into the main document.
code	NULL	Set to R code. Knitr will replace the code in the chunk with the code in the code option.
engine	'R'	Knitr will evaluate the chunk in the named language, e.g. **engine = 'python'**. Run **names(knitr::knit_engines$get())** to see supported languages.
eval	TRUE	If **FALSE**, knitr will not run the code in the code chunk.
include	TRUE	If **FALSE**, knitr will run the chunk but not include the chunk in the final document.
purl	TRUE	If **FALSE**, knitr will not include the chunk when running **purl()** to extract the source code.
Results		
collapse	FALSE	If **TRUE**, knitr will collapse all the source and output blocks created by the chunk into a single block.
echo	TRUE	If **FALSE**, knitr will not display the code in the code chunk above it's results in the final document.
		If **'hide'**, knitr will not display the code's results in the final document. If **'hold'**, knitr will delay displaying all output

The R Markdown Cheat Sheet

www.rstudio.com/resources/cheatsheets/



R Markdown Cheat Sheet

learn more at rmarkdown.rstudio.com

rmarkdown 0.2.50 Updated: 8/14

RStudio

1. Workflow

R Markdown is a format for writing reproducible, dynamic reports with R. Use it to embed R code and results into slideshows, pdfs, html documents, Word files and more. To make a report:

i. Open - Open a file that uses the .Rmd extension.

ii. Write - Write content with the easy to use R Markdown syntax

iii. Embed - Embed R code that creates output to include in the report

iv. Render - Replace R code with its output and transform the report into a slideshow, pdf, html or ms Word file.

2. Open File

Start by saving a text file with the extension .Rmd, or open an RStudio Rmd template

- In the menu bar, click **File ▸ New File ▸ R Markdown...**
- A window will open. Select the class of output you would like to make with your .Rmd file
- Select the specific type of output to make with the radio buttons (you can change this later)
- Click OK

3. Markdown

Next, write your report in plain text. Use markdown syntax to describe how to format text in the final report.

syntax

```
Plain text
End a line with two spaces to start a new paragraph.
*italics* and _italics_
**bold** and __bold__
superscript^2^
~~strikethrough~~
[link](www.rstudio.com)

# Header 1

## Header 2

### Header 3

#### Header 4

##### Header 5

###### Header 6

endash: --
emdash: ---
ellipsis: ...
inline equation: $A = \pi*r^{2}$
image: ![](path/to/smallorb.png)

horizontal rule (or slide break):

***

> block quote

* unordered list
* item 2
    + sub-item 1
    + sub-item 2

1. ordered list
2. item 2
    + sub-item 1
    + sub-item 2

Table Header  | Second Header
------------- | -------------
Table Cell    | Cell 2
Cell 3        | Cell 4
```

becomes

Plain text
End a line with two spaces to start a new paragraph.
italics and *italics*
bold and **bold**
superscript^2
~~strikethrough~~
link

Header 1
Header 2
Header 3
Header 4
Header 5
Header 6

endash: –
emdash: —
ellipsis: ...
inline equation: $A = \pi * r^2$
image:
horizontal rule (or slide break):

> block quote

- unordered list
- item 2
  - sub-item 1
  - sub-item 2

1. ordered list
2. item 2
   - sub-item 1
   - sub-item 2

Table Header	Second Header
Table Cell	Cell 2
Cell 3	Cell 4

4. Choose Output

Write a YAML header that explains what type of document to build from your R Markdown file.

YAML

A YAML header is a set of key: value pairs at the start of your file. Begin and end the header with a line of three dashes (- - -)

```
---
title: "Untitled"
author: "Anonymous"
output: html_document
---

This is the start of my
report. The above is metadata
saved in a YAML header.
```

The RStudio template writes the YAML header for you

The output value determines which type of file R will build from your .Rmd file (in Step 6)

- **output: html_document** ····· html file (web page)
- **output: pdf_document** ····· pdf document
- **output: word_document** ····· Microsoft Word .docx
- **output: beamer_presentation** ····· beamer slideshow (pdf)
- **output: ioslides_presentation** ····· ioslides slideshow (html)

RStudio® is a trademark of RStudio, Inc. • All rights reserved • info@rstudio.com • 844-448-1212 • rstudio.com

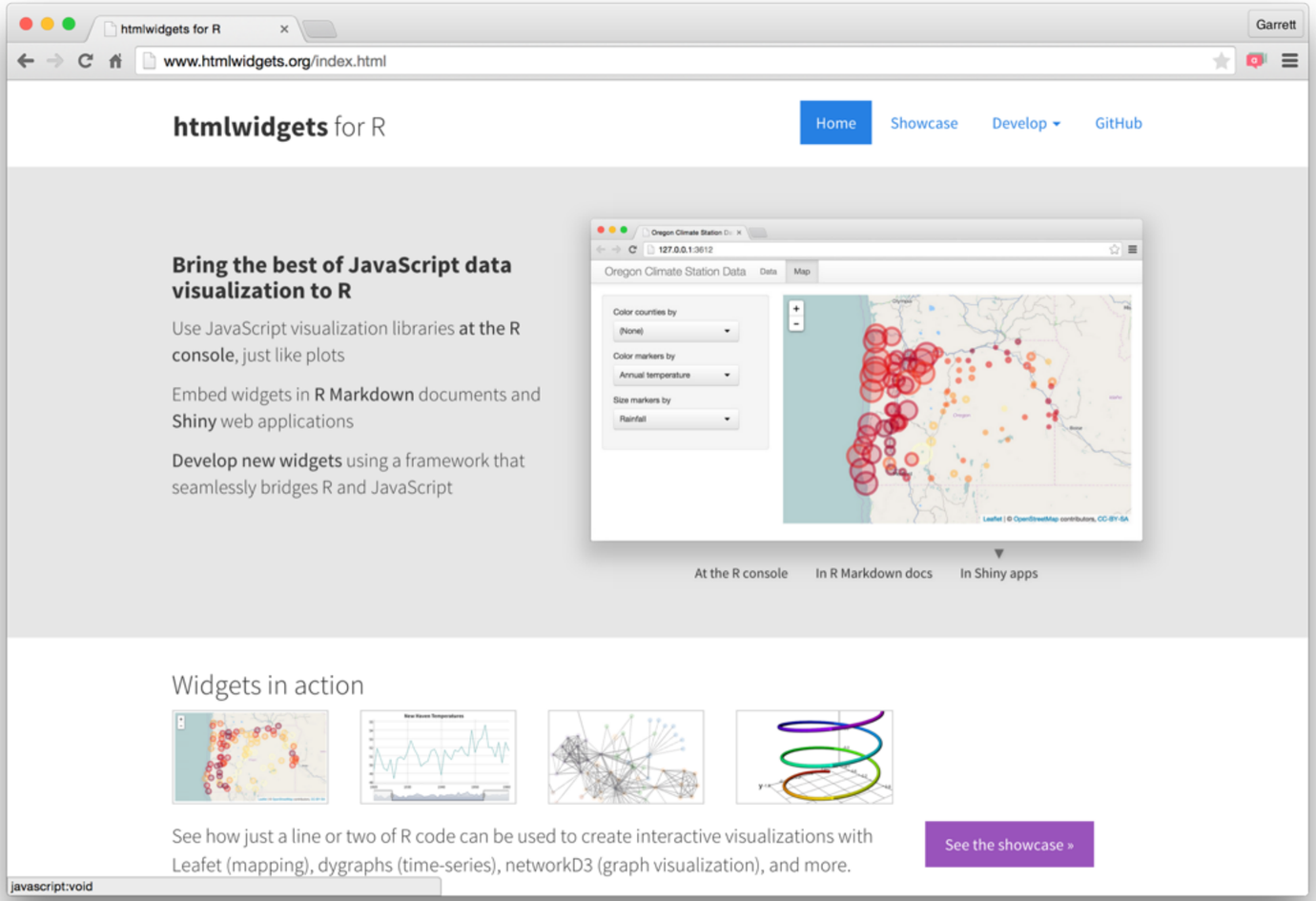
htmlwidgets

htmlwidgets

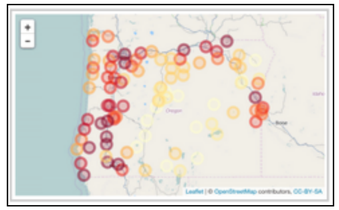
Use htmlwidgets in:

- RStudio viewer pane
- R Markdown files
- Shiny Apps

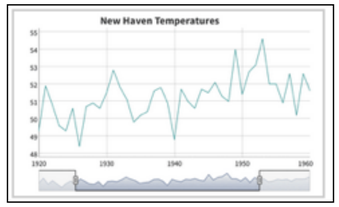
www.htmlwidgets.org



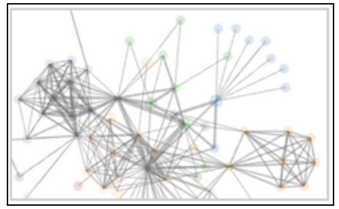
Packages with ready-to-use htmlwidgets:



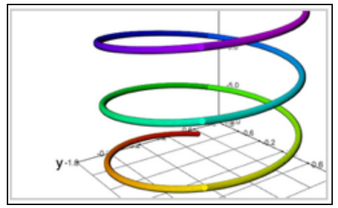
Leaflet - Interactive maps
rstudio.github.io/leaflet/



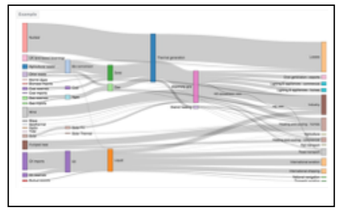
dygraphs - Time series
rstudio.github.io/dygraphs



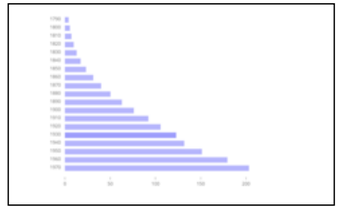
networkD3 - network graphs
christophergandrud.github.io/networkD3/



threejs - 3D charts
github.com/bwlewis/rthreejs

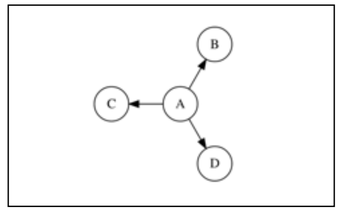


rCharts - Various charts
rCharts.io



MetricsGraphics - d3 charts
hrbrmstr.github.io/metricsgraphics/

DT - Data tables
rstudio.github.io/DT/

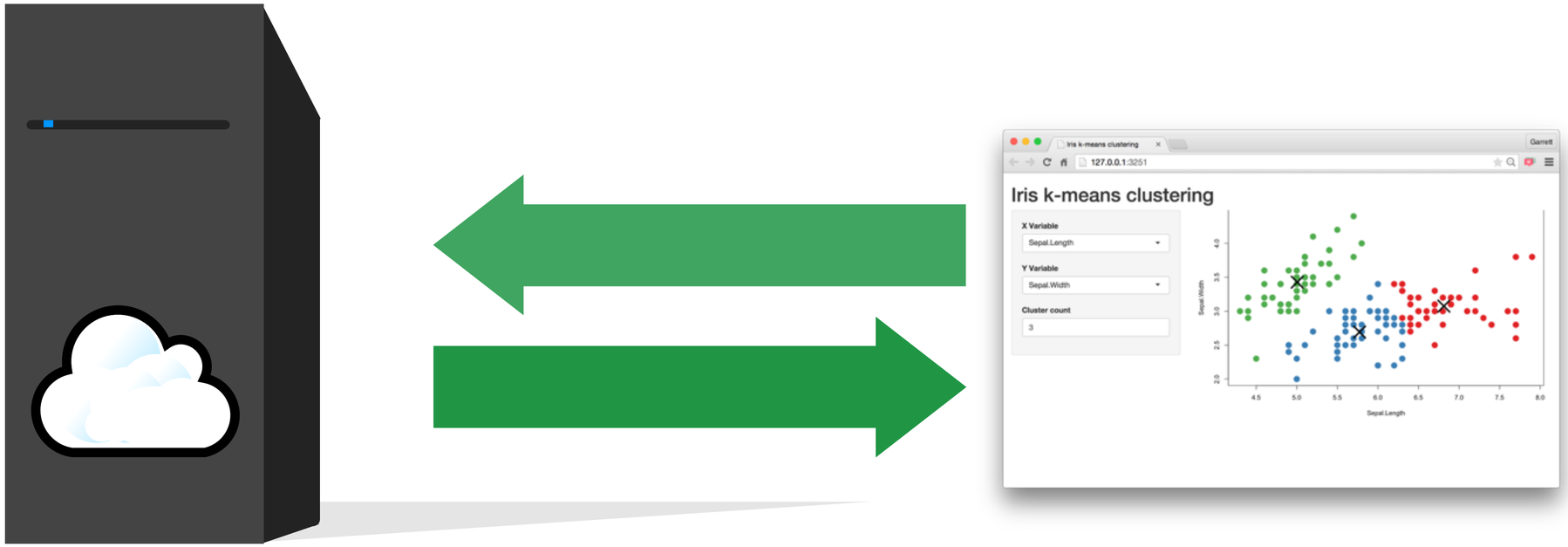


Diagrammer - Diagrams
rich-iannone.github.io/DiagrammeR/

Shiny

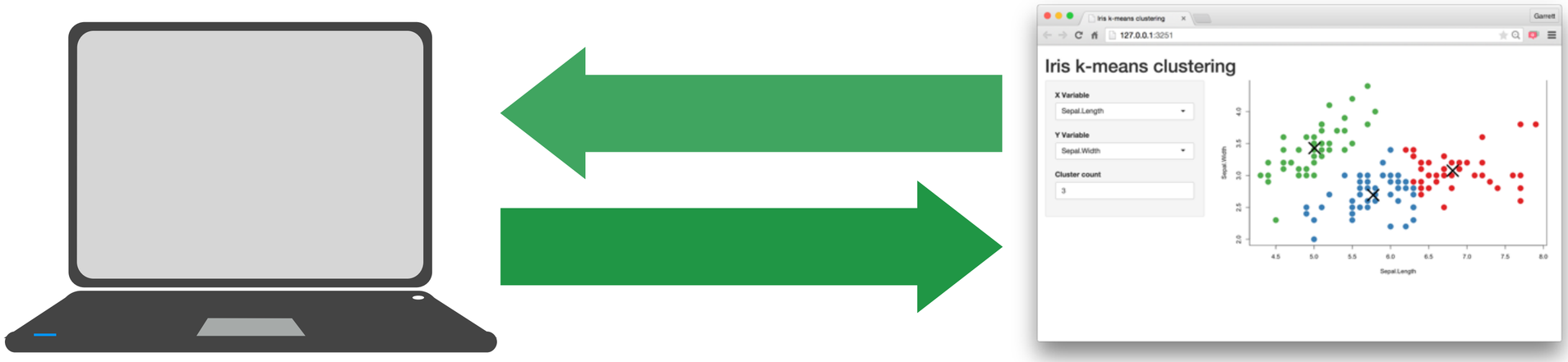
Shiny

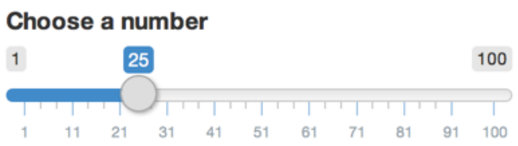
Every Shiny app is maintained by a computer running R

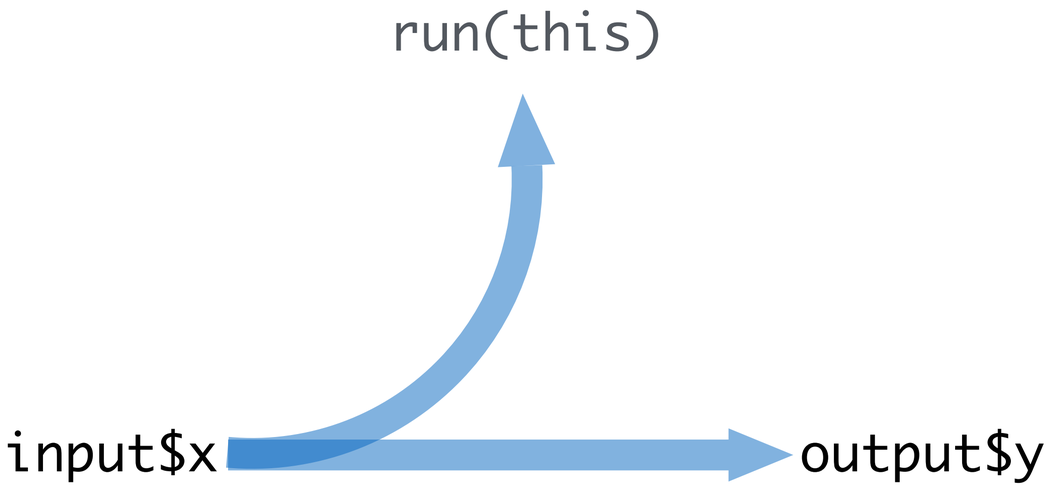


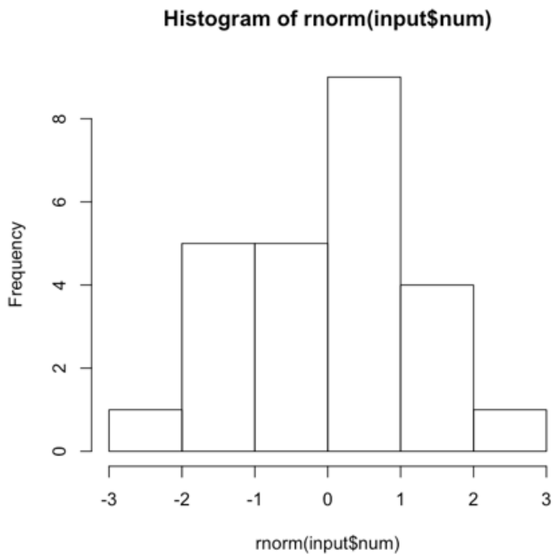
How to make an app

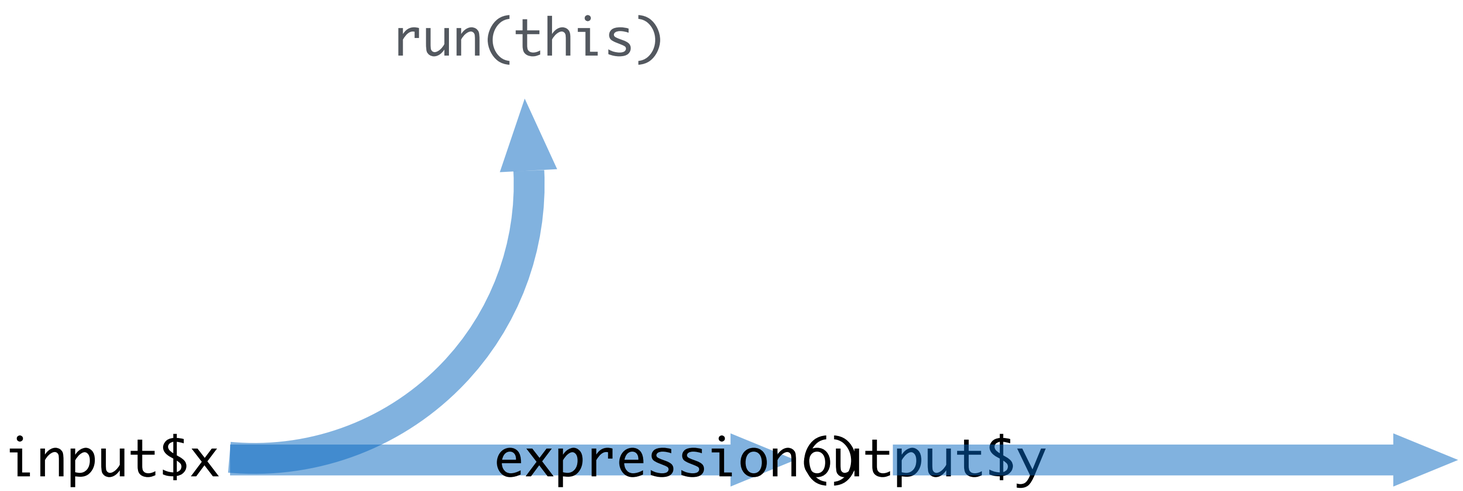
Every Shiny app is maintained by a computer running R

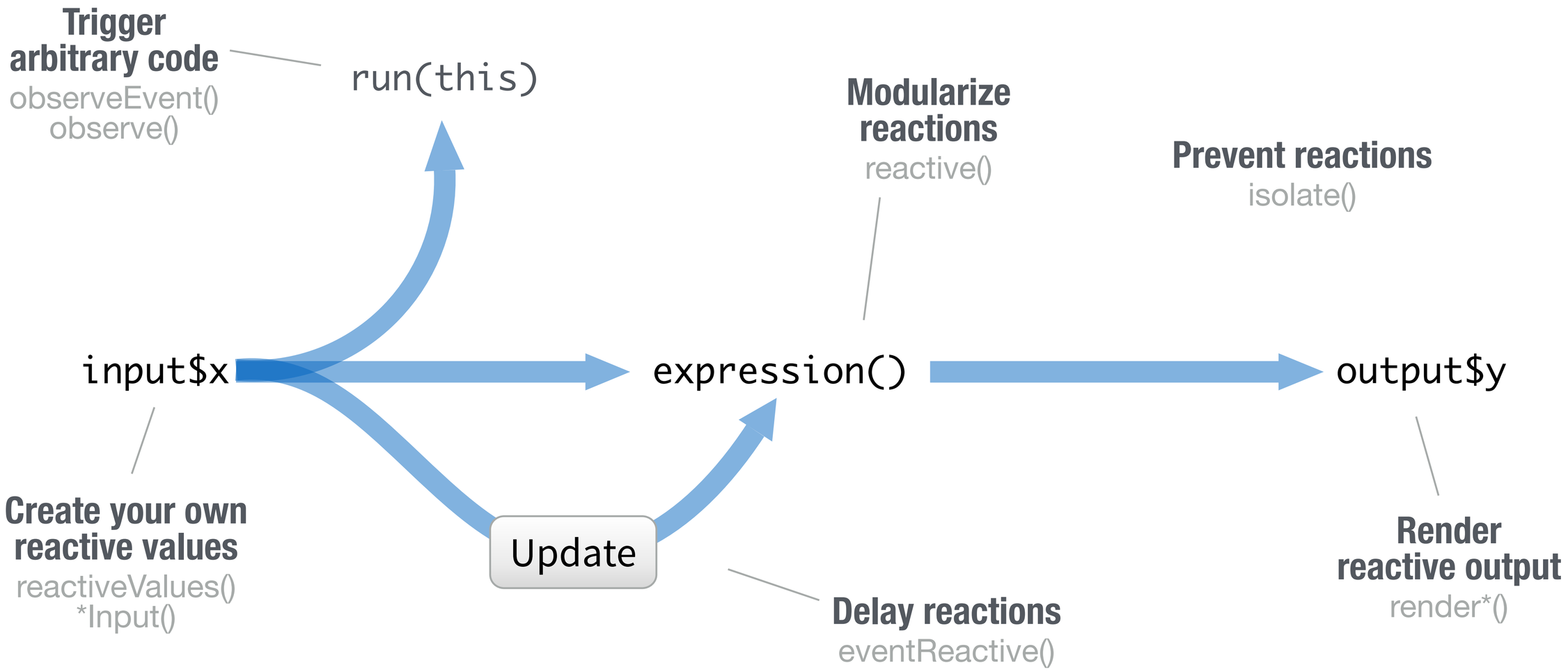










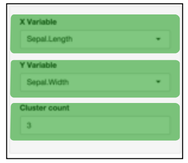


Recap: UI

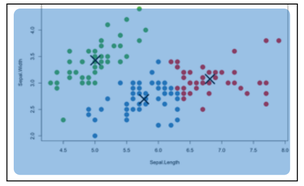
```
library(shiny)
ui <- fluidPage()
server <- function(input, output) {}
shinyApp(ui = ui, server = server)
```

Begin each app with the template

Add elements as arguments to **fluidPage()**



Create reactive inputs with an ***Input()** function



Display reactive results with an ***Output()** function

Recap: Server

Use the server function to assemble inputs into outputs. Follow 3 rules:

`output$hist <-`

1. Save the output that you build to **output$**

```
renderPlot({
  hist(rnorm(input$num))
})
```

2. Build the output with a **render*()** function

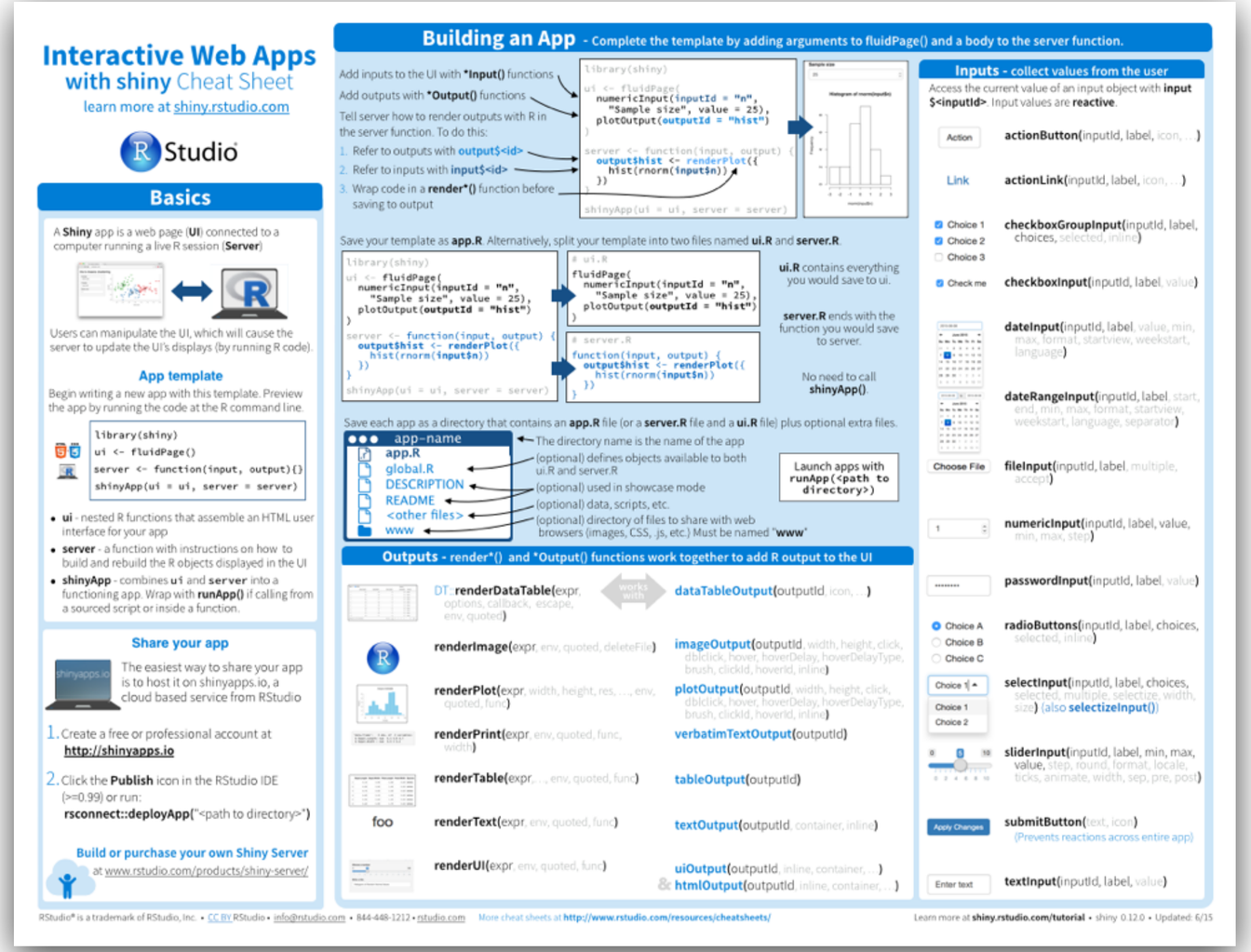
`input$num`

3. Access input values with **input$**

Teach yourself Shiny

The Shiny Cheat Sheet

www.rstudio.com/resources/cheatsheets/



Interactive Web Apps with shiny Cheat Sheet

learn more at shiny.rstudio.com

RStudio

Basics

A **Shiny** app is a web page (**UI**) connected to a computer running a live R session (**Server**)

Users can manipulate the UI, which will cause the server to update the UI's displays (by running R code).

App template

Begin writing a new app with this template. Preview the app by running the code at the R command line.

```
library(shiny)
ui <- fluidPage()
server <- function(input, output){}
shinyApp(ui = ui, server = server)
```

- **ui** - nested R functions that assemble an HTML user interface for your app
- **server** - a function with instructions on how to build and rebuild the R objects displayed in the UI
- **shinyApp** - combines **ui** and **server** into a functioning app. Wrap with **runApp()** if calling from a sourced script or inside a function.

Share your app

The easiest way to share your app is to host it on shinyapps.io, a cloud based service from RStudio

1. Create a free or professional account at **http://shinyapps.io**
2. Click the **Publish** icon in the RStudio IDE (>=0.99) or run: **rsconnect::deployApp("<path to directory>")**

Build or purchase your own Shiny Server at www.rstudio.com/products/shiny-server/

Building an App - Complete the template by adding arguments to fluidPage() and a body to the server function.

Add inputs to the UI with ***Input()** functions

Add outputs with ***Output()** functions

Tell server how to render outputs with R in the server function. To do this:

1. Refer to outputs with **output$<id>**
2. Refer to inputs with **input$<id>**
3. Wrap code in a **render*()** function before saving to output

```
library(shiny)
ui <- fluidPage(
  numericInput(inputId = "n",
    "Sample size", value = 25),
  plotOutput(outputId = "hist")
)
server <- function(input, output) {
  output$hist <- renderPlot({
    hist(rnorm(input$n))
  })
}
shinyApp(ui = ui, server = server)
```

Save your template as **app.R**. Alternatively, split your template into two files named **ui.R** and **server.R**.

```
library(shiny)
ui <- fluidPage(
  numericInput(inputId = "n",
    "Sample size", value = 25),
  plotOutput(outputId = "hist")
)
server <- function(input, output) {
  output$hist <- renderPlot({
    hist(rnorm(input$n))
  })
}
shinyApp(ui = ui, server = server)
```

```
# ui.R
fluidPage(
  numericInput(inputId = "n",
    "Sample size", value = 25),
  plotOutput(outputId = "hist")
)
```

ui.R contains everything you would save to ui.

```
# server.R
function(input, output) {
  output$hist <- renderPlot({
    hist(rnorm(input$n))
  })
}
```

server.R ends with the function you would save to server.

No need to call **shinyApp()**.

Save each app as a directory that contains an **app.R** file (or a **server.R** file and a **ui.R** file) plus optional extra files.

- app-name
  - app.R
  - global.R — (optional) defines objects available to both ui.R and server.R
  - DESCRIPTION — (optional) used in showcase mode
  - README — (optional) data, scripts, etc.
  - <other files> — (optional) directory of files to share with web browsers (images, CSS, .js, etc.) Must be named "**www**"
  - www

The directory name is the name of the app

Launch apps with `runApp(<path to directory>)`

Outputs - render*() and *Output() functions work together to add R output to the UI

	DT::**renderDataTable**(expr, options, callback, escape, env, quoted)	**dataTableOutput**(outputId, icon, ...)
	renderImage(expr, env, quoted, deleteFile)	**imageOutput**(outputId, width, height, click, dblclick, hover, hoverDelay, hoverDelayType, brush, clickId, hoverId, inline)
	renderPlot(expr, width, height, res, ..., env, quoted, func)	**plotOutput**(outputId, width, height, click, dblclick, hover, hoverDelay, hoverDelayType, brush, clickId, hoverId, inline)
	renderPrint(expr, env, quoted, func, width)	**verbatimTextOutput**(outputId)
	renderTable(expr, ..., env, quoted, func)	**tableOutput**(outputId)
foo	**renderText**(expr, env, quoted, func)	**textOutput**(outputId, container, inline)
	renderUI(expr, env, quoted, func)	**uiOutput**(outputId, inline, container, ...) & **htmlOutput**(outputId, inline, container, ...)

Inputs - collect values from the user

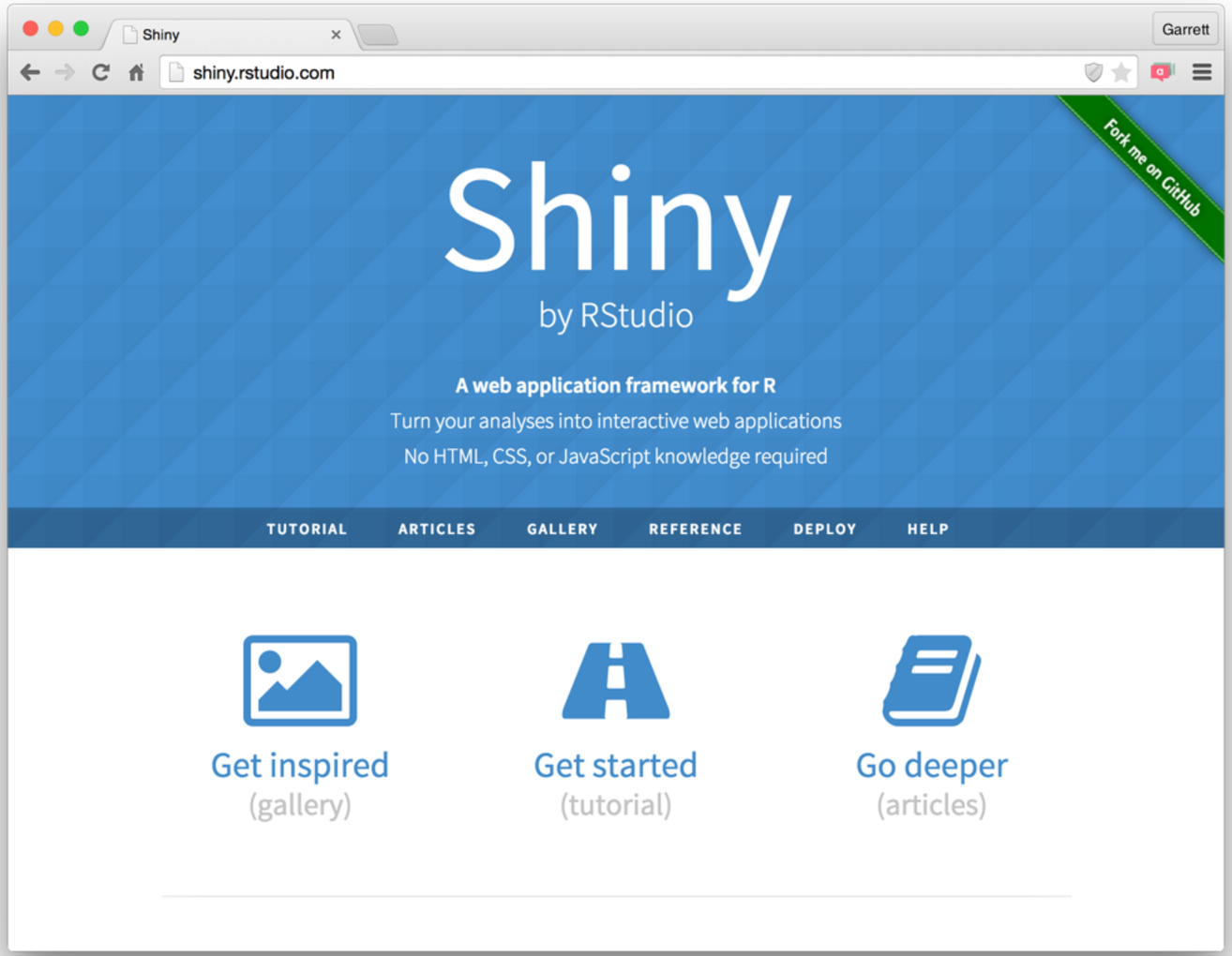
Access the current value of an input object with **input $<inputId>**. Input values are **reactive**.

Action	**actionButton**(inputId, label, icon, ...)
Link	**actionLink**(inputId, label, icon, ...)
Choice 1, Choice 2, Choice 3	**checkboxGroupInput**(inputId, label, choices, selected, inline)
Check me	**checkboxInput**(inputId, label, value)
	dateInput(inputId, label, value, min, max, format, startview, weekstart, language)
	dateRangeInput(inputId, label, start, end, min, max, format, startview, weekstart, language, separator)
Choose File	**fileInput**(inputId, label, multiple, accept)
1	**numericInput**(inputId, label, value, min, max, step)
•••••••	**passwordInput**(inputId, label, value)
Choice A, Choice B, Choice C	**radioButtons**(inputId, label, choices, selected, inline)
Choice 1, Choice 1, Choice 2	**selectInput**(inputId, label, choices, selected, multiple, selectize, width, size) (also **selectizeInput()**)
0, 5, 10	**sliderInput**(inputId, label, min, max, value, step, round, format, locale, ticks, animate, width, sep, pre, post)
Apply Changes	**submitButton**(text, icon) (Prevents reactions across entire app)
Enter text	**textInput**(inputId, label, value)

RStudio® is a trademark of RStudio, Inc. • CC BY RStudio • info@rstudio.com • 844-448-1212 • rstudio.com More cheat sheets at http://www.rstudio.com/resources/cheatsheets/ Learn more at shiny.rstudio.com/tutorial • shiny 0.12.0 • Updated: 6/15

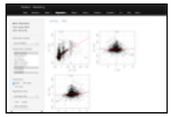
The Shiny Development Center

shiny.rstudio.com



What can an app do?

What can a Shiny app do?



Make R analysis accessible to **non-programmers**

Highly **customizable**, highly **shareable** HTML front end

Read and write to **databases**

Monitor **streaming data**

Require and use **authentication**

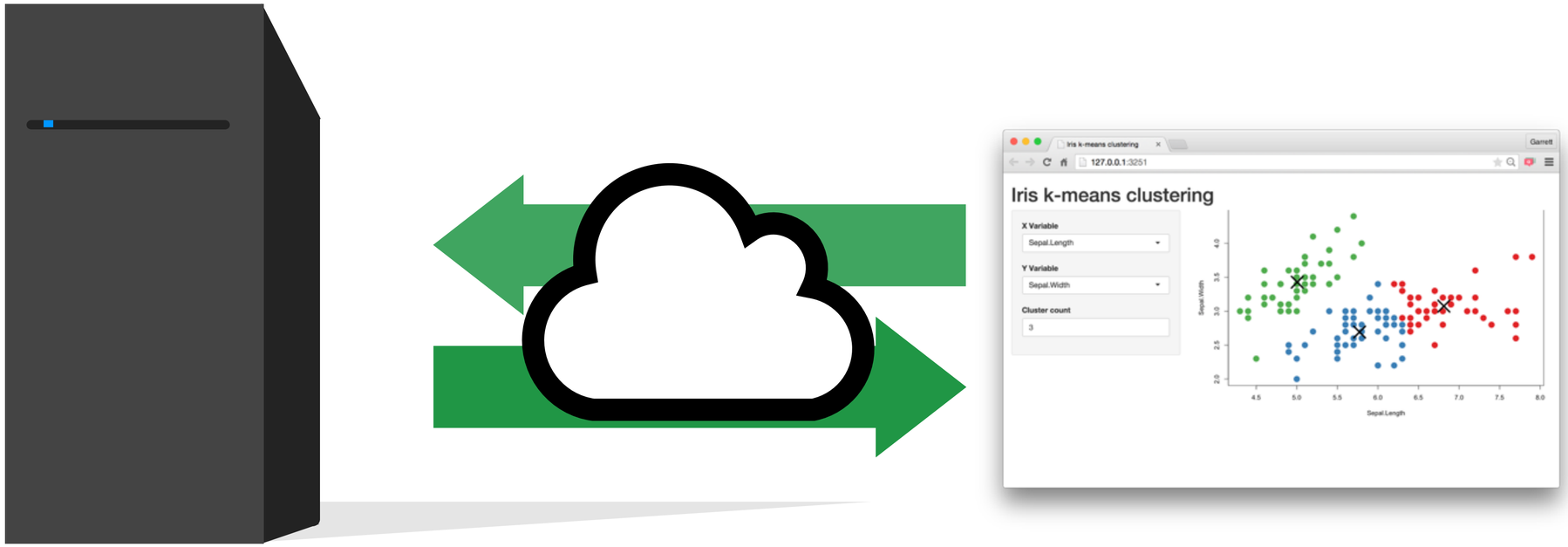
Ideal for Exploratory Data Analysis

Ideal Data Portal / Results Explorer / Simulation API / Dashboard

Share

your app

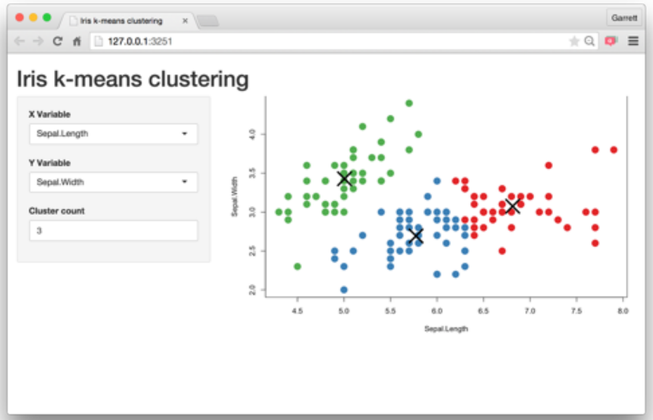
Every Shiny app is maintained by a computer running R



Shinyapps.io

A server maintained by RStudio

- free
- easy to use
- secure
- scalable

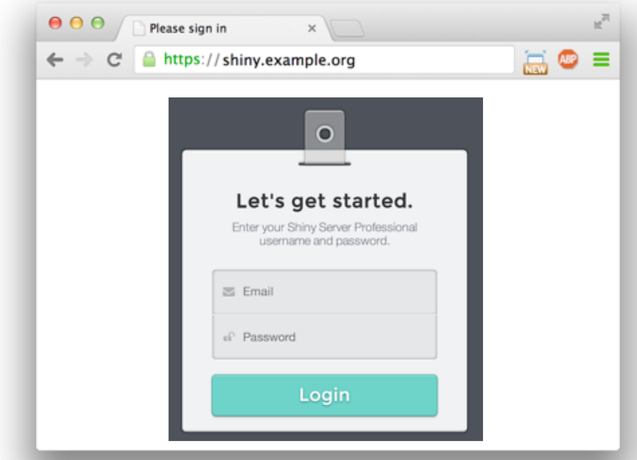


Build your own server

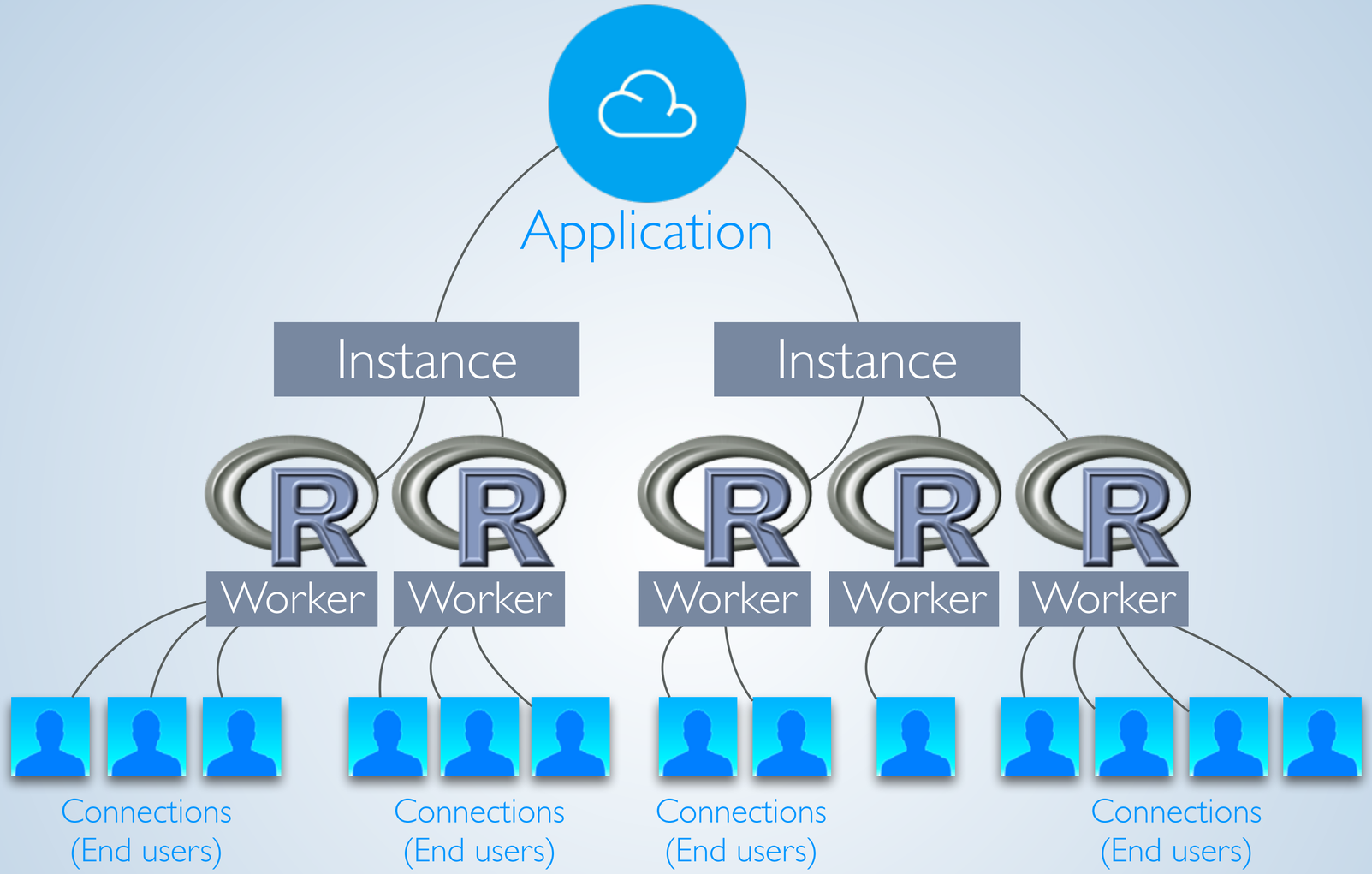
Shiny Server **Pro**

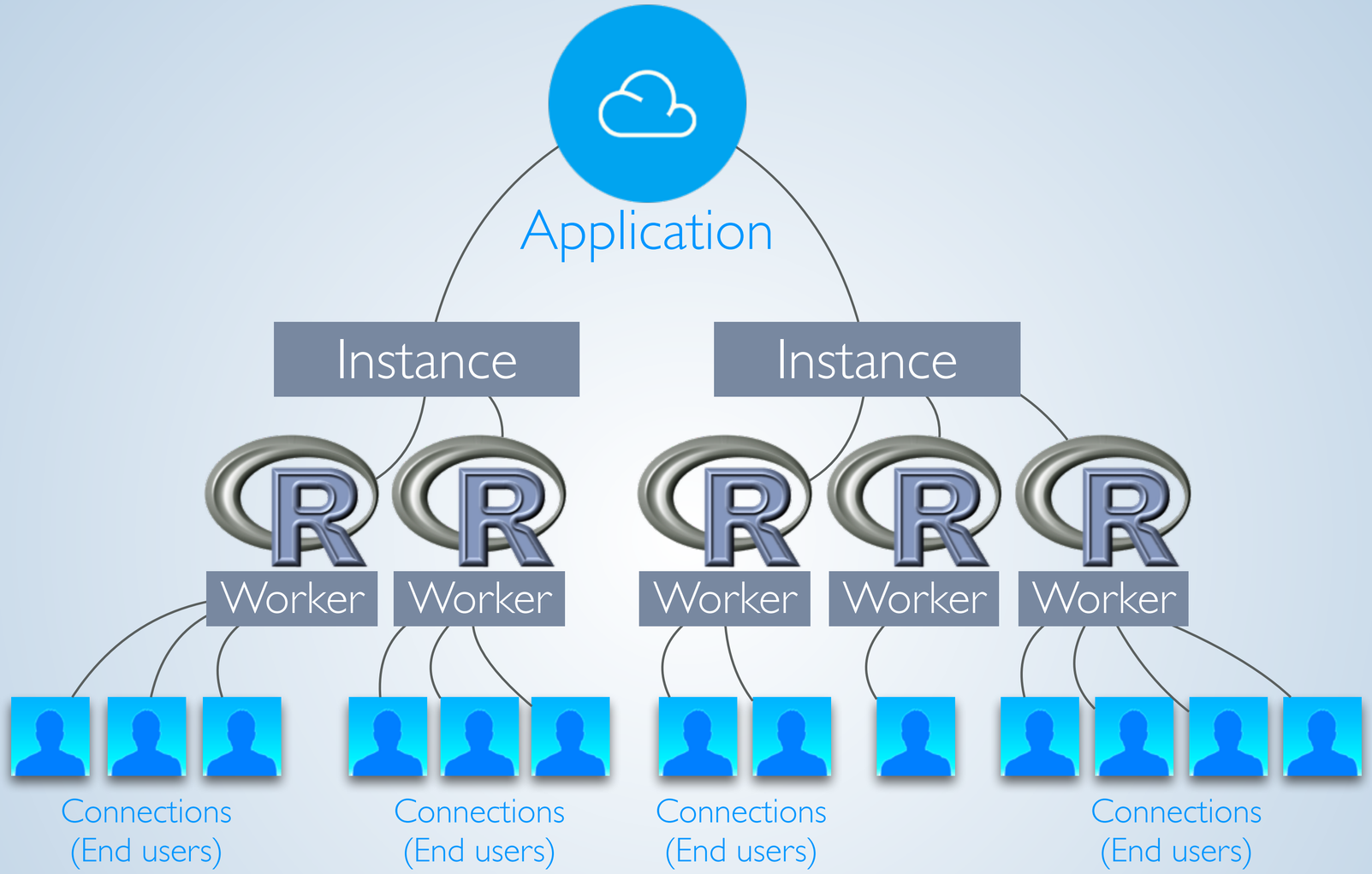
www.rstudio.com/products/shiny/shiny-server/

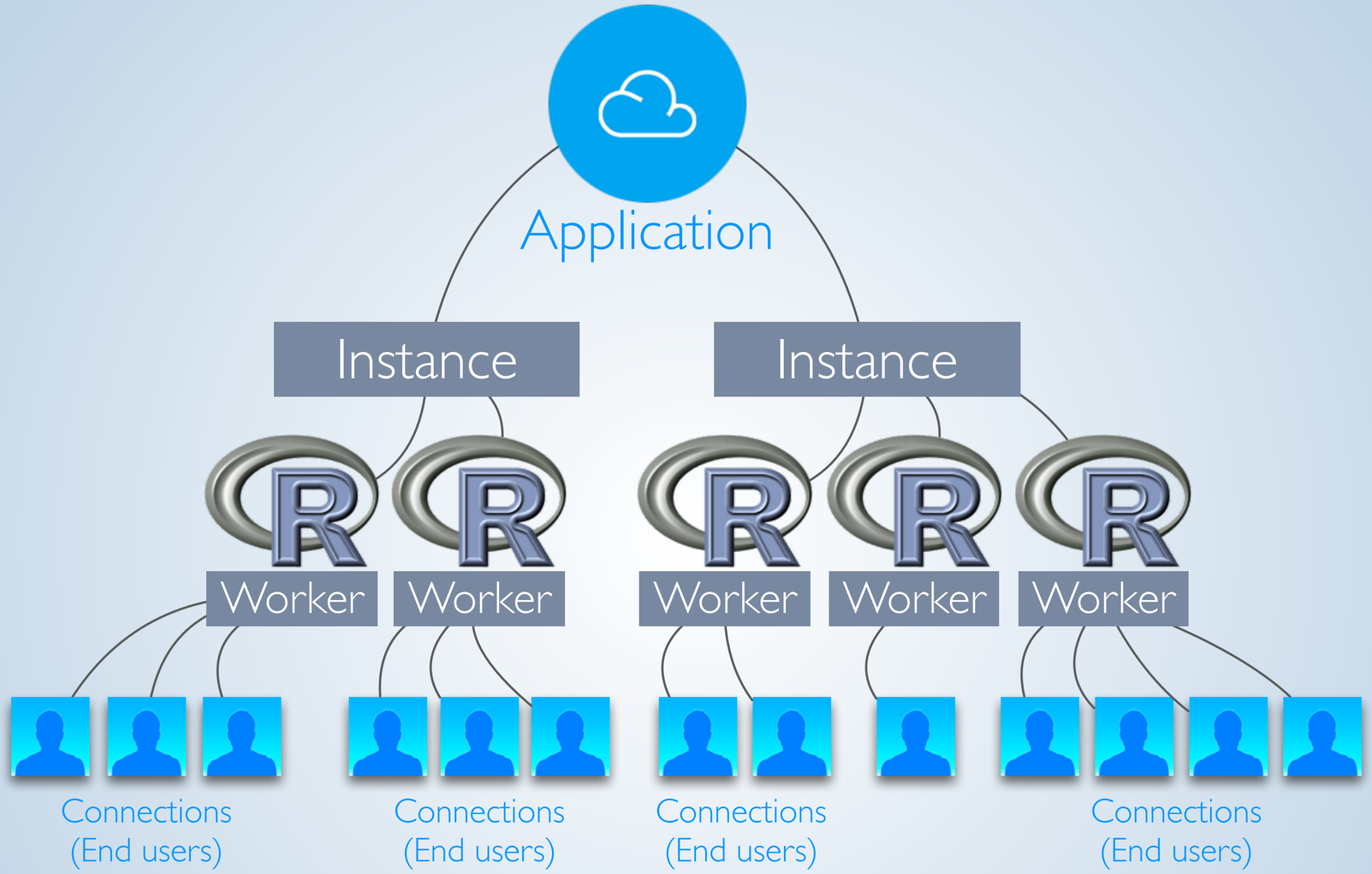
- ✔ **Secure access** - LDAP, GoogleAuth, SSL, and more
- ✔ **Performance** - fine tune at app and server level
- ✔ **Management** - monitor and control resource use
- ✔ **Support** - direct priority support



45 day evaluation free trial


Application
Instance
Instance
Worker
Worker
Worker
Worker
Worker
Connections
(End users)
Connections
(End users)
Connections
(End users)
Connections
(End users)


Application
Instance
Instance
Worker
Worker
Worker
Worker
Worker
Connections
(End users)
Connections
(End users)
Connections
(End users)
Connections
(End users)

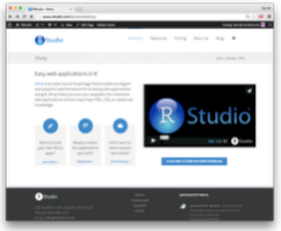

Application
Instance
Instance
Worker
Worker
Worker
Worker
Worker
Connections
(End users)
Connections
(End users)
Connections
(End users)
Connections
(End users)

Useful websites

The Shiny development center:
shiny.rstudio.com

The R Markdown development center:
rmarkdown.rstudio.com

Shiny and R Markdown cheat sheets:
www.rstudio.com/resources/cheatsheets



RStudio products:
www.rstudio.com/products/shiny