INTRODUCTION TO

AND

Studio®

Ezgi Karaesmen

April 2019 Pelotonia Fundraiser

@ Columbus Collaboratory

ONE GOAL

Hello!

I am Ezgi! (Ez-ghee)

I am a PhD student at OSU, my research interests are Statistical Genetics and Pharmacogenomics.

And I am a Pelotonia Fellow!

Email: Ezgi.Karaesmen@osumc.edu
Twitter: @e_Karaesmen
LinkedIn: www.linkedin.com/in/ezgi-karaesmen

ONE GOAL

PELOTONIA®

ONE GOAL

Who are the R-Ladies?

R-Ladies is a worldwide organization to promote gender diversity in the R community.

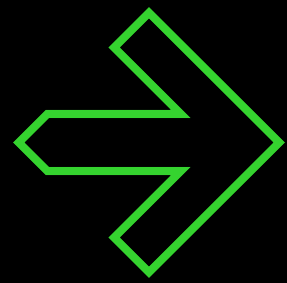
Our mission: To achieve proportionate representation by encouraging, inspiring, and empowering the minorities currently underrepresented in the R community.

As such, leadership and mentoring roles are reserved for women *(including trans women, genderqueer women, and non-binary people who are significantly female-identified)*, and men are welcome to attend meetings as guests.

R-Ladies Columbus chapter

- R-Ladies Columbus is the Columbus chapter of R-Ladies.
- Current Co-Organizers:
  Ezgi (me)
  Katie Sasso (Data Scientist at Columbus Collaboratory)

- Become a member : meetup.com/RLadies-Columbus/
- Follow us on Twitter: @RLadiesColumbus



R and RStudio

Bonnie and Clyde of Statistical Programming

A Computer Programming Language

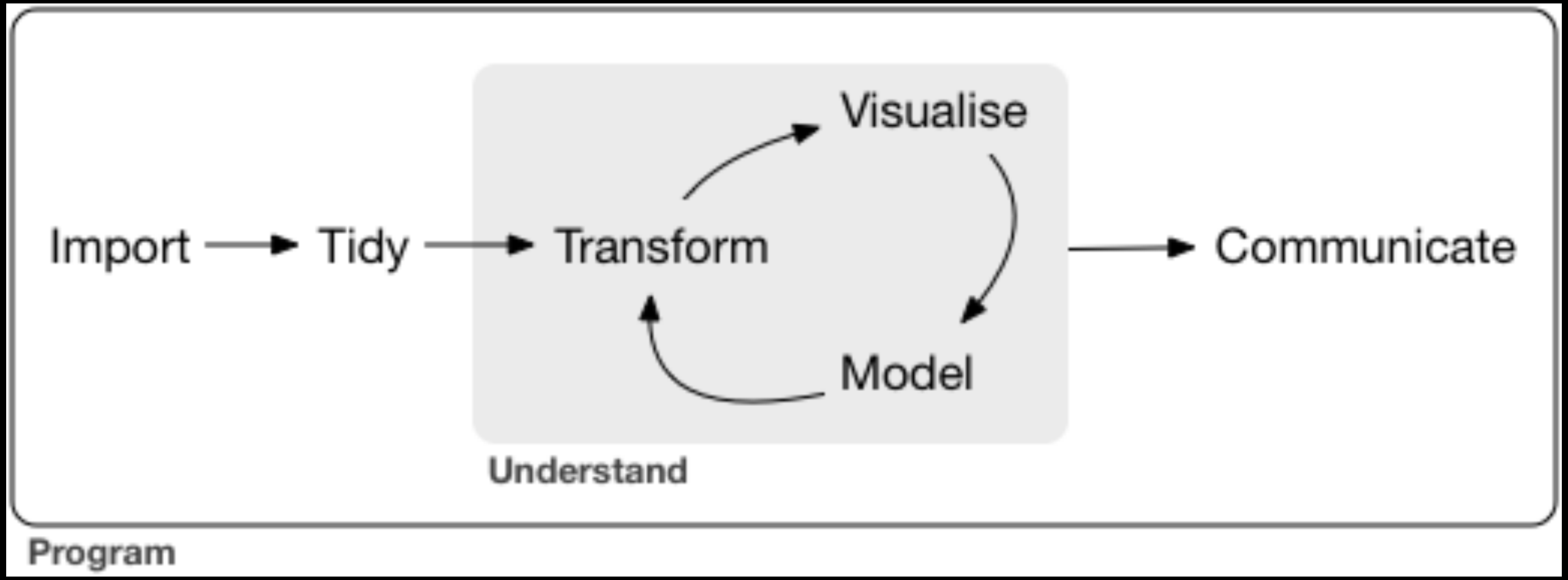
- Free and open source programming language for statistical computing and graphics
- Descends from S programming language
- Created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand
- Does not need RStudio

ONE GOAL

RStudio®: A Sofware Program

- If R is a pretty painting, Rstudio is the frame that enhances R's beauty.
- A free and open-source integrated development environment (IDE) for R
- Built to help you write R code, run R code, and analyze data with R
- Text editor, version control, latex integration, keyboard shortcuts, debugging tools, and much more

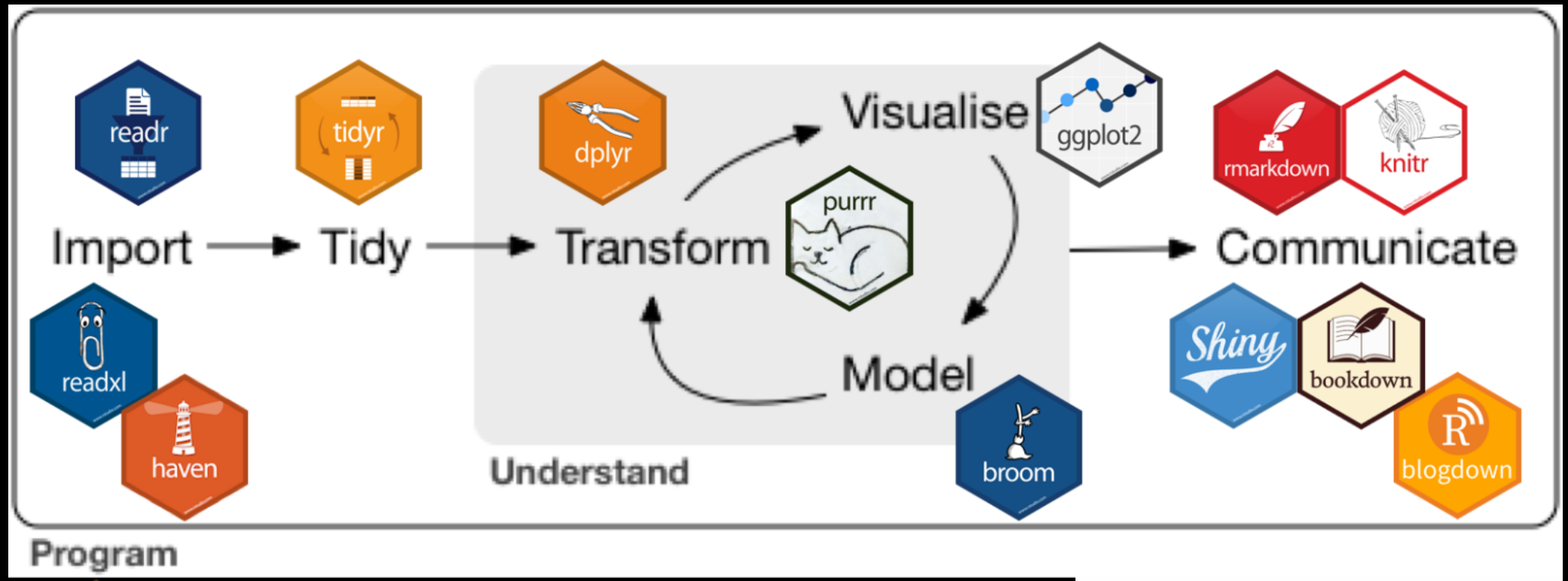
A Data Science Workflow



R for Data Science - *Garrett Grolemund & Hadley Wickham*

ONE GOAL

Why R is so awesome?



R Studio®

ONE GOAL

Why R is so awesome?

R community is the best!

#rstats and #rladies on Twitter

ONE GOAL

There are so many things one can do with R and RStudio

Data Visualization
http://www.r-graph-gallery.com/portfolio/ggplot2-package/

Interactive Apps with Shiny
https://shiny.rstudio.com/gallery/

Reports and Books
https://rmarkdown.rstudio.com/gallery.html

Some very useful links

R for Data Science Book – The Book! : https://r4ds.had.co.nz/

Introduction to Data Science Book: https://leanpub.com/datasciencebook

RStartHere: https://github.com/rstudio/RStartHere

#rstats on Twitter

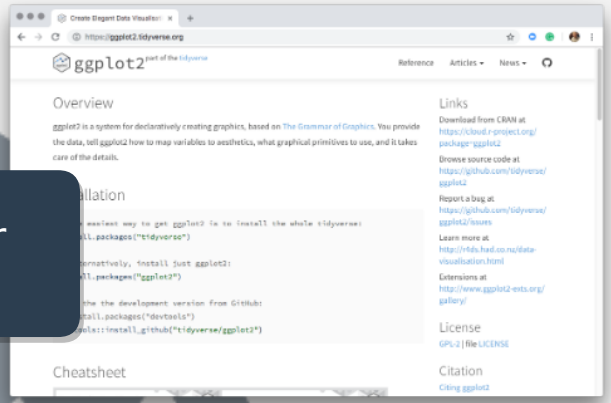
DataCamp https://www.datacamp.com/ 

Learn more about what's going on with DataCamp read:
https://juliasilge.com/blog/datacamp-misconduct/
https://medium.com/@heathernolis/on-datacamp-aafd82f94e60

ONE GOAL

Data Science with R Workflow

The Data Science With R Workflow is available in the book: R For Data Science. If you want to learn R and this workflow ***for business analysis***, take the **R For Business Analysis (DS4B 101-R) course through Business Science University.**



Click the links for Documentation

ggplot2 (CS)

dplyr (CS)
stringr (CS)
lubridate (CS)
forcats (CS)
Base R (CS)

Import → **Tidy** → **Transform** → **Visualize** → **Model** → **Communicate**

readr (CS)
readxl / writexl
odbc / DBI
rvest

tibble (CS)
tidyr (CS)

purrr (CS)
(iteration)

recipes
rsample
parsnip

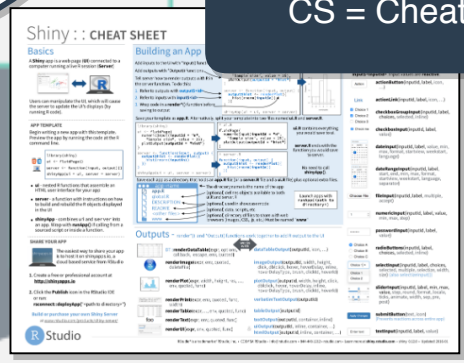
broom
yardstick
dials

RMarkdown (CS)
Shiny (CS)

RStudio IDE (CS)

fs (file system)

CS = Cheat Sheet



Important Resources

- **R For Data Science Book:** http://r4ds.had.co.nz/
- **Rmarkdown Book:** https://bookdown.org/yihui/rmarkdown/
- **Data Visualization Book**: https://rkabacoff.github.io/datavis/
- **More Cheatsheets:** https://www.rstudio.com/resources/cheatsheets/
- **tidyverse packages:** https://www.tidyverse.org/
- **Connecting to databases:** https://db.rstudio.com/
- **RMarkdown website:** https://rmarkdown.rstudio.com/
- **Shiny web applications website:** http://shiny.rstudio.com/
- **Jenny Bryan's purrr tutorial:** https://jennybryan.org/

"Data Science Courses for Business"

Business Science University
university.business-science.io

version: 1.2

Data Science with R

Special Topics

Text Analysis & NLP

- Text Mining with R (Book): tidytext
- NLP:
  - H2O word2vec: Word embeddings
  - text2vec: fast vectorization, topic modeling
  - udpipe: UDPipe C++ lib in R

Machine Learning

- Multi-Threaded/Scalable/Production ML:
  - H2O (CS)
  - Extreme Gradient Boosting: xgboost
  - R + Spark: sparklyr (CS)
  - Sparkling Water (Spark + H2O): rsparkling
- ML (Tidy): parsnip
- ML: caret (CS)

Deep Learning

- R Interface to TensorFlow Homepage:
  - Keras (CS)
  - TF Estimators
  - TensorFlow (Core)

Time Series Analysis

- Time-aware tibbles: tibbletime & tsibble
- Convert between classes: timetk & tsbox
- Time Series Index Summary: timetk
- Generating Future Series: timetk

Forecasting

- ARIMA, ETS, etc: forecast & fable
- Tidy, glance, augment for forecast models: sweep
- Converting forecast prediction to tibble: sweep

Anomaly Detection

- Identify anomalies: anomalize

Network Analysis

- Network Data Transformations (Tidy): tidygraph
- Network Data Transformations: igraph

Network Viz

- Static:
  - ggraph - Graph plotting utilities for ggplot2
- Interactive (JavaScript):
  - networkD3 - D3 Networks in R
  - plotly (CS) - plotly.js (network graphs) in R

Speed & Scale

- Fastest Single-Node Speed: data.table (CS)
- Distributed Cluster (Spark): sparklyr (CS)
- Parallel Processing: furrr

Interoperability

- Java: rJava
- Python: reticulate
- C++: Rcpp

Financial Analysis

- Getting financial data: tidyquant & quantmod
- Quantitative Analysis: tidyquant & xts/TTR
- Portfolio Analysis: tidyquant & PerformanceAnalytics

Financial & Time Viz

- Static:
  - tidyquant - Financial ggplot2 geoms
- Interactive:
  - highcharter - highchart.js in R
  - dygraphs - xts plotting
  - plotly (CS) - plotly.js (financial) in R

Geospatial Analysis

- Geocoding (getting lat/long, bboxes, & sf's):
  - ggmap - Google API (requires key)
  - osmdata - OpenStreet Overpass API
  - tmaptools - OpenStreet Nominatum API
- Simple Features (sf objects): sf (CS) (tidy)
- Spatial Objects (sp objects): sp (non-tidy)

Geospatial Viz

- Static:
  - ggmap - Google API (requires key)
  - osmplotr - Impressive Maps via OSM
  - tmap - Thematic Maps
  - cartography (CS) - Thematic Maps
- Interactive (JavaScript):
  - leaflet (CS) - leaflet.js in R
  - plotly (CS) - plotly.js (maps) in R

Miscellaneous Tools

- **Interactive Plotting:** htmlwidgets for R
- **Building R Packages:** R packages Book
  - Pkg Development Tools: devtools (CS)
  - R Templates: usethis
  - Build Web Doc's: pkgdown
- **Advanced Concepts** (Advanced R Book)
  - rlang & Tidy Evaluation (CS)
- **Making Blogs & Books:**
  - blogdown, bookdown
- **Posting Code (GitHub, Stack Overflow):** reprex

Don’t forget!

- This is like learning new language – it’s not so fun at the beginning but once you become good at it, it’s really useful (and fun, even sexy).
- Prompting errors messages is the way R communicates with you. Although this may be frustrating, trying to understand what R is telling you is the only way to establish a healthy and happy relationship. ♥
- Google is like a the couples therapist for you and R – never underestimate the power of Google 🔍

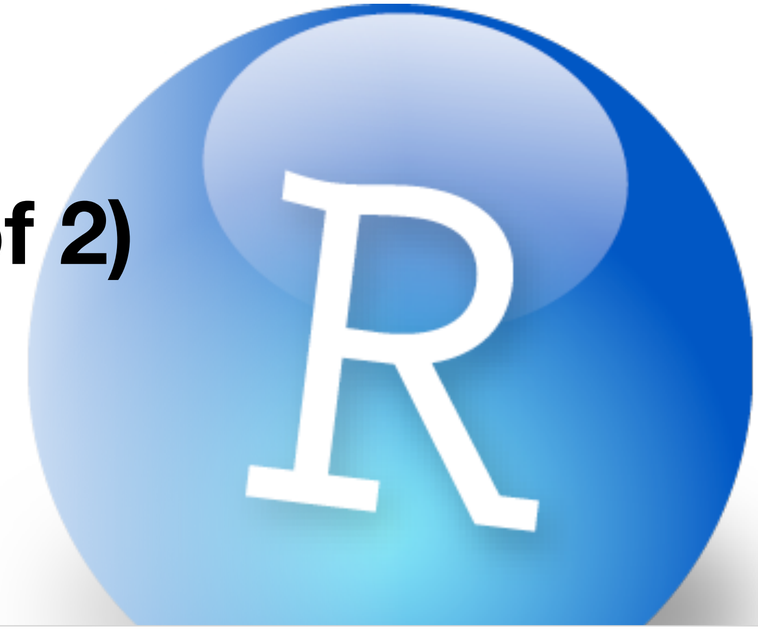
ONE GOAL

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

The R Language (1 of 2)

Understand how R stores and works with data



Garrett Grolemund

Master Instructor, RStudio

August 2014

1. RStudio basics
2. R basics
3. Data structures
4. Data types

Warm ups

Your turn

Which of these are numbers? Which are words? How can you tell?

1 "1" "one"

Your turn

The matrix below is named **M**. What is the value of $\mathbf{M}_{34}$?

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17
18	19	20	21	22	23

Your turn

The matrix below is named **M**. What is the value of $\mathbf{M}_{34}$?

Your turn

The matrix below is named **M**. What is the value of $\mathbf{M}_{34}$?

Your turn

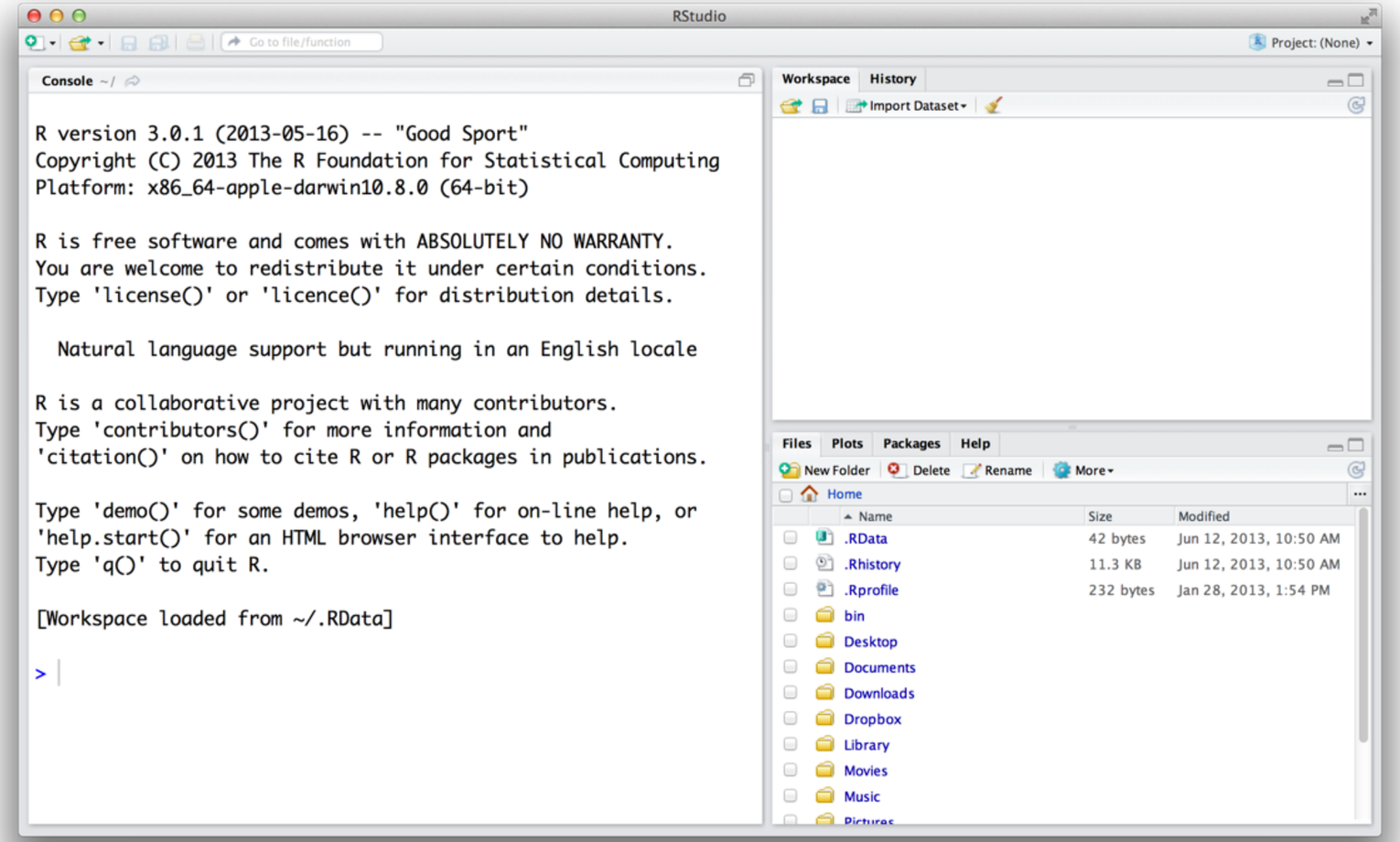
The matrix below is named **M**. What is the value of $\mathbf{M}_{34}$?

RStudio

Getting started

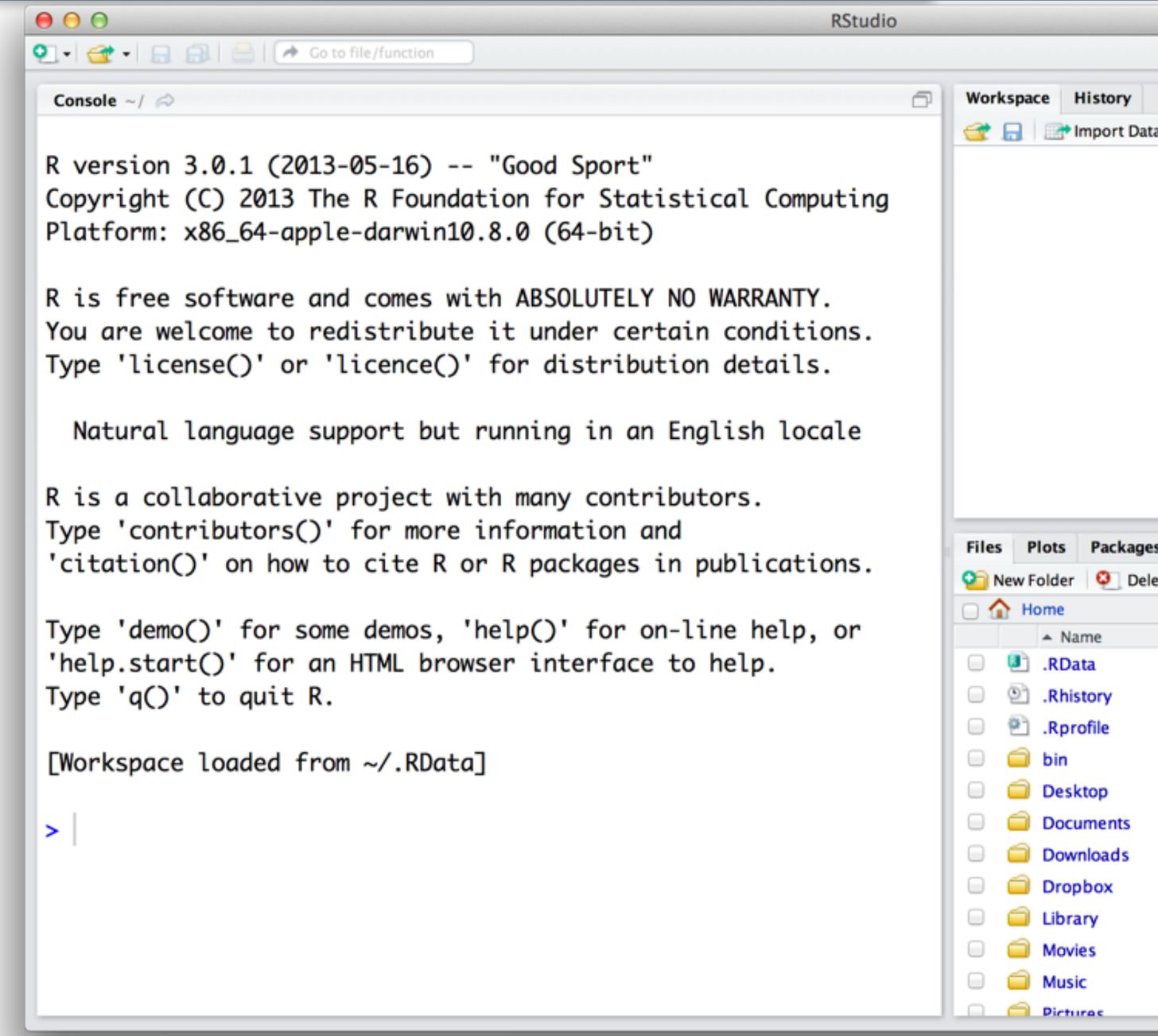
Open RStudio as you would any program. It's icon looks like this

The RStudio window is divided into 3-4 panes. Each keeps track of separate information.



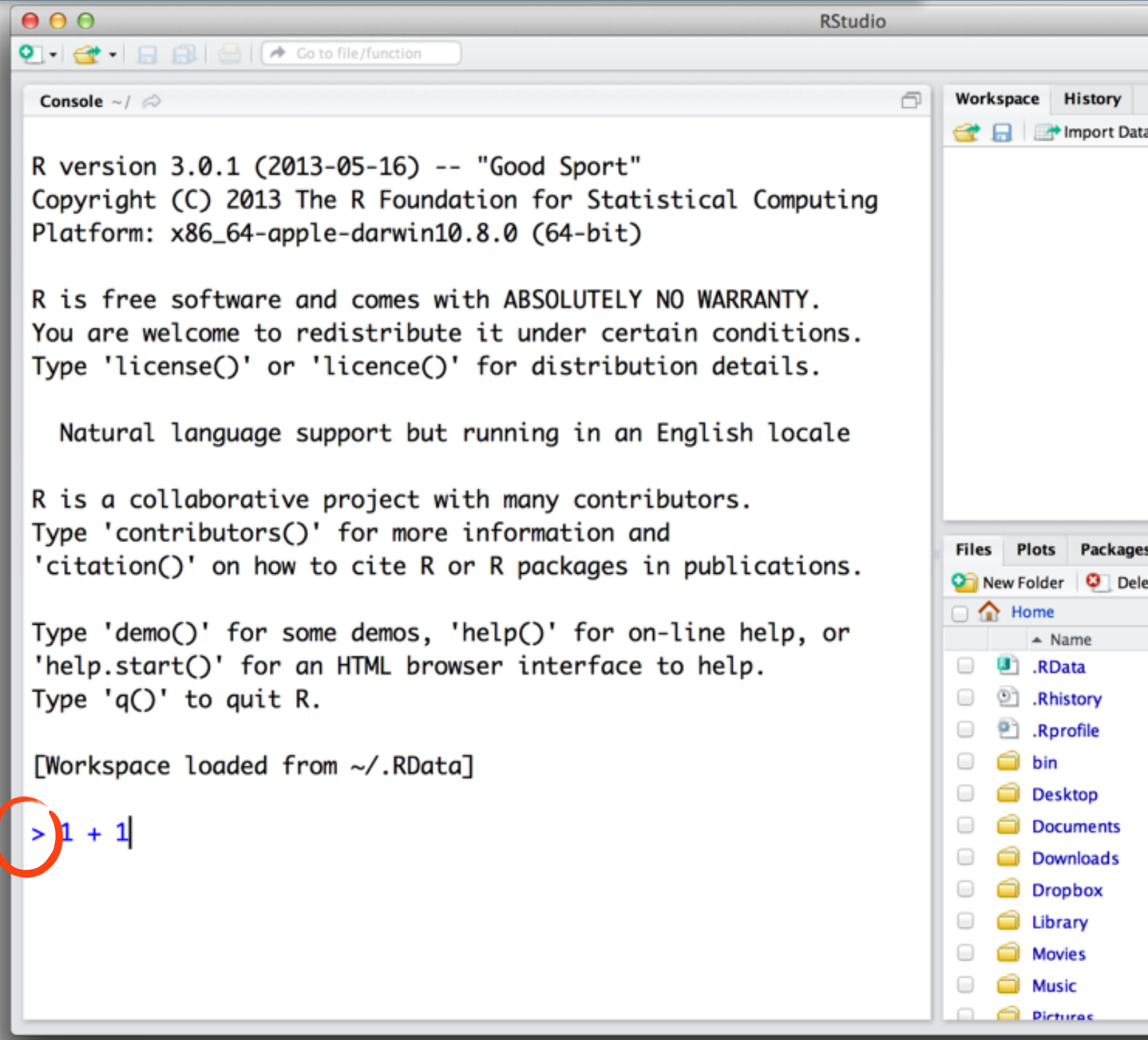
R Console

The console gives you a place to execute commands written in the R computer language



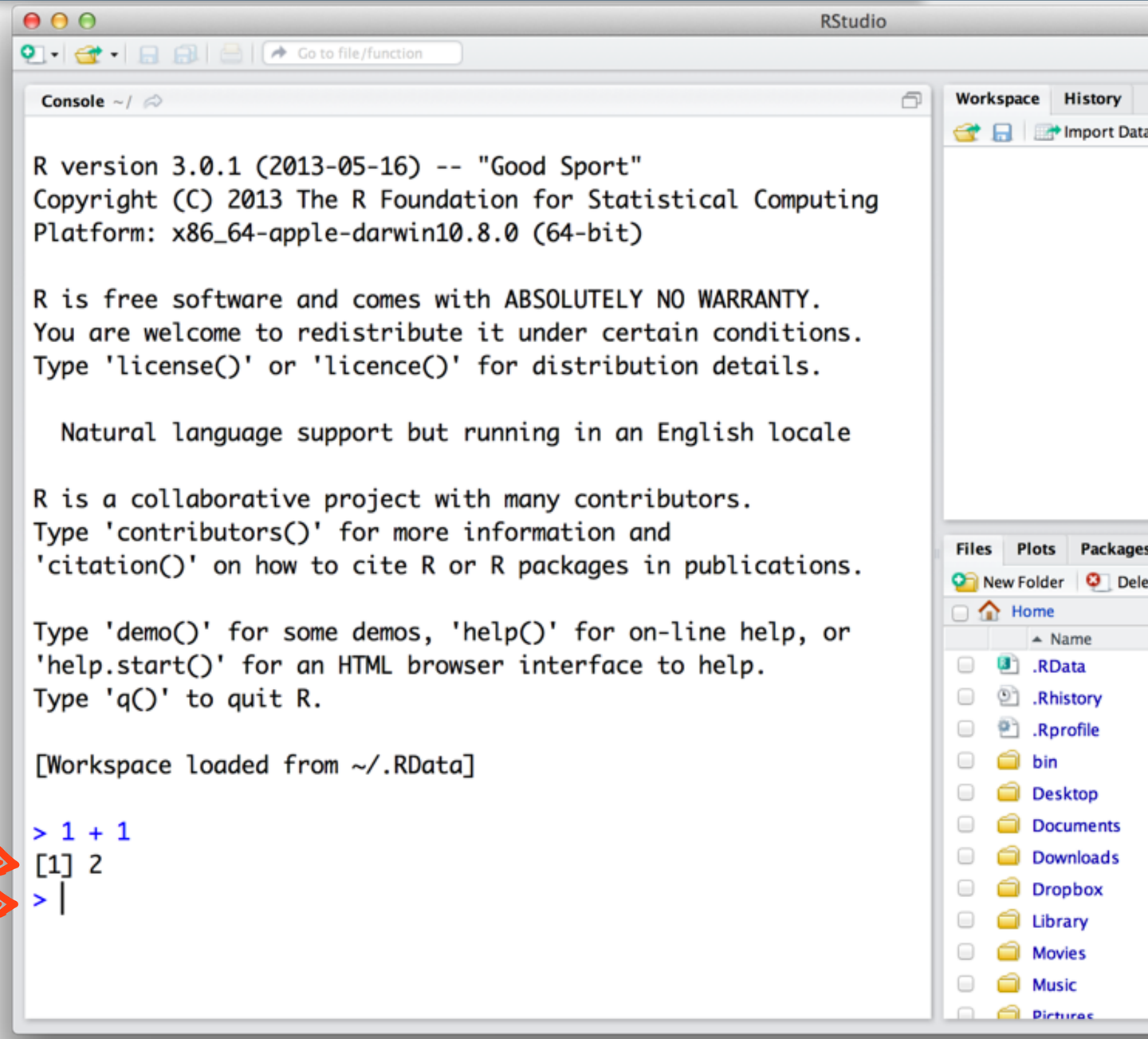
Command prompt

Type commands on the line that begins with a > sign (known as the prompt)



Output

When you hit enter, RStudio will run your command and display any output below it

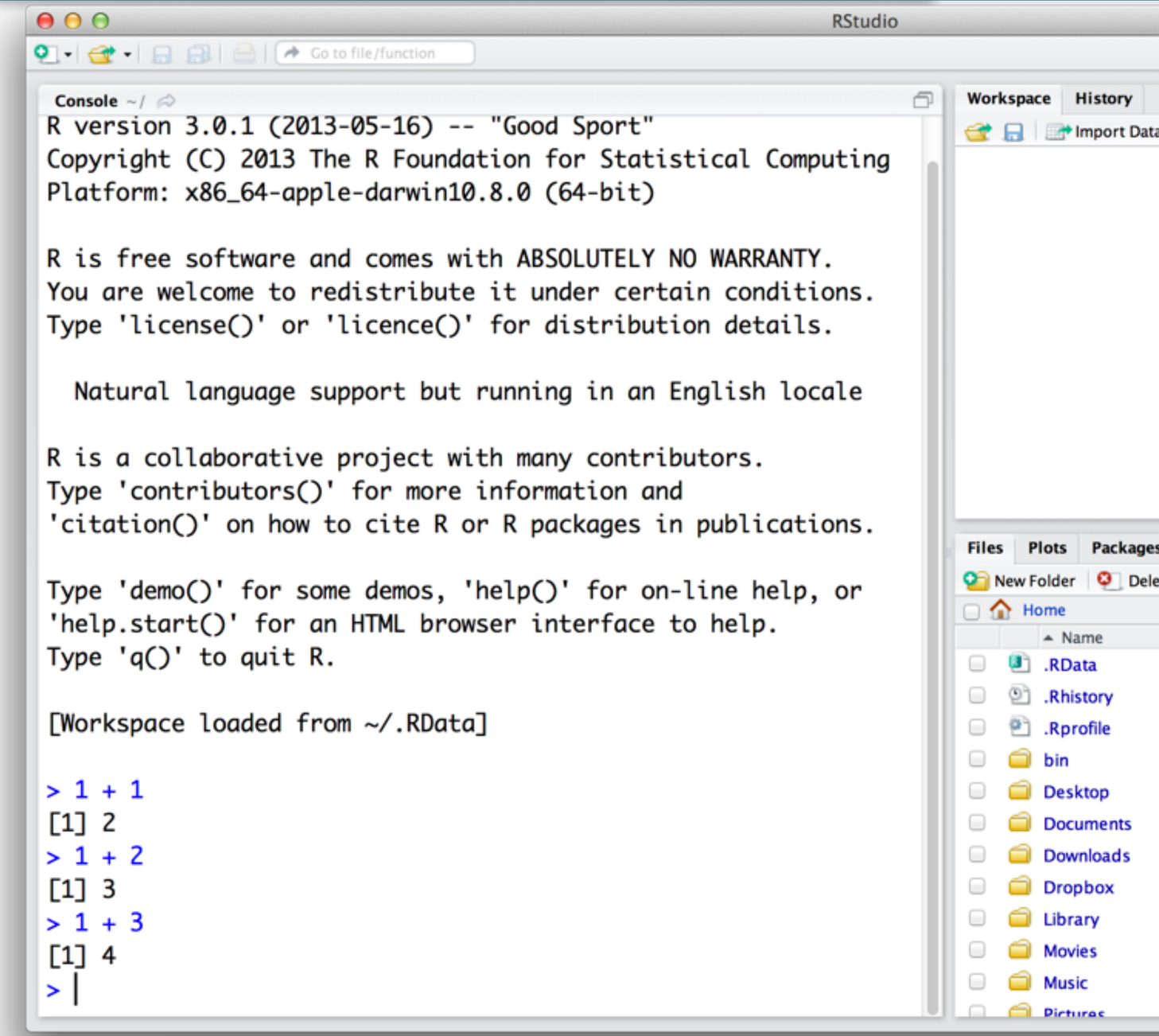


Output

New prompt

History

As you enter commands, you accumulate a history of past commands

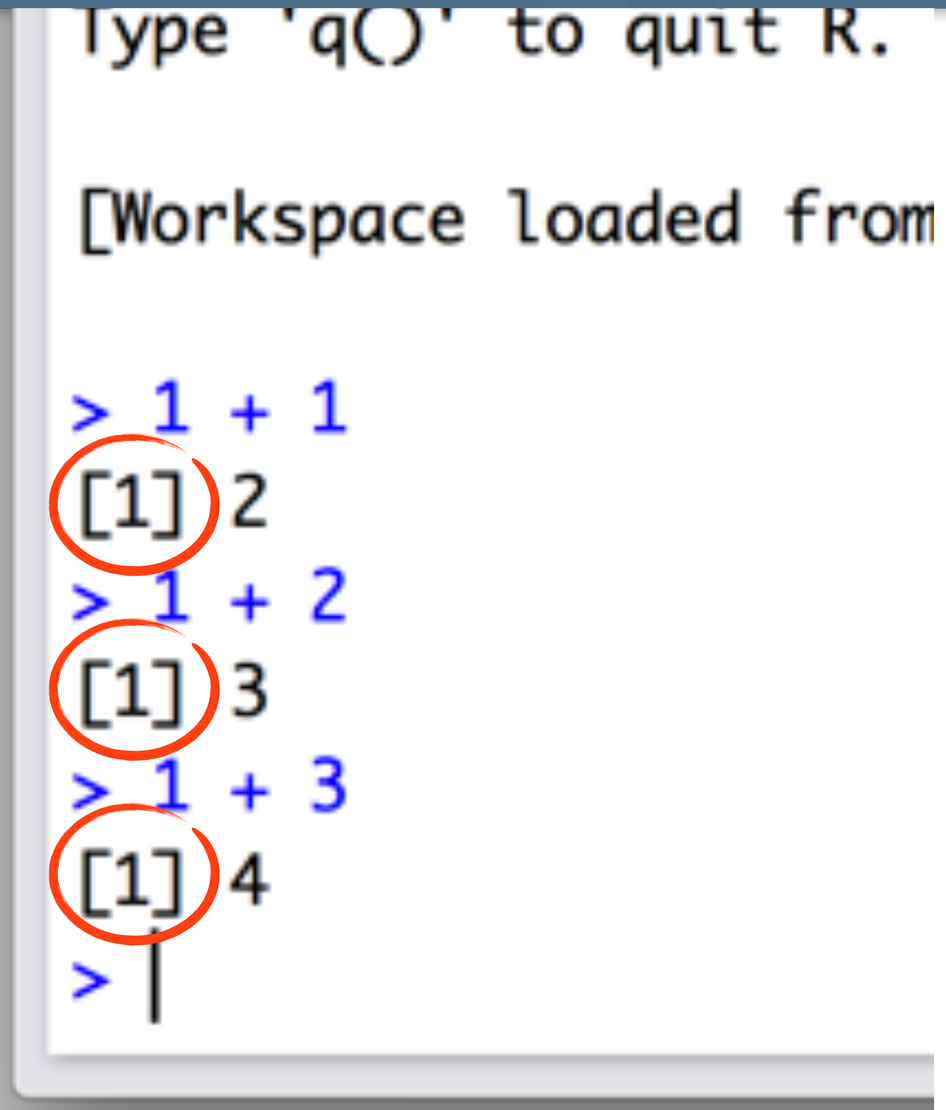


[1]

R displays an index next to the output.

Just ignore this.

Somewhat helpful when R returns more than one value in the output.



R is like a fancy calculator on your computer

```
5 + 5
# 10

4 - 1
# 3

1 * 2
# 2

4 ^ 2
# 16
```

It can do algebra

```
a <- 1
b <- 2

a + b
# 3

A <- 3

a + b - A
# 0
```

And it has functions that let you do more sophisticated manipulations

```
round(3.1415)
# 3
```

```
factorial(3)
# 6
```

3! = 3 x 2 x 1

```
sqrt(9)
# 3
```

square root

Calling functions in R

R has many built-in functions which are structured like this:

```
function_name(arg1 = val1, arg2 = val2, …)
```

- For example `seq()` function generates **seq**uences of numbers.
- To see the details for this function type `?seq` on your console.
- Generally this `seq()` function is structured like this:

```
seq(from, to, by)
```

Your turn!

```
seq(from, to, by)
```

Above is the structure of `seq()` function, generate a sequence of numbers from 1 to 10 increasing by 0.5

You have 30 seconds!

Your turn!

```
seq(from, to, by)
```

Above is the structure of `seq()` function, generate a sequence of numbers from 1 to 10 increasing by 0.5

Answer:

```
seq(from=1, to=10, by=0.5)
```

You can also have functions within functions

For example I can generate a sequence of numbers from 1 to 10, increasing by 1, compute the mean for this sequence and take its square root.

```
sqrt(mean(seq(from=1, to=10, by=1)))
```

Your turn

What do you think this will return?
You have 30 seconds.

Your turn

What do you think this will return?
You have 30 seconds.

```
factorial(round(2.0015) + 1)
```

R always works from the innermost parenthesis to the outermost (just like a calculator).

```
factorial(round(2.0015) + 1)
```



```
factorial(2 + 1)
```



```
factorial(3)
```



```
6
```

R always works from the innermost parenthesis to the outermost (just like a calculator).

```
factorial(round(2.0015) + 1)
```



```
factorial(2 + 1)
```



```
factorial(3)
```



```
6
```

R always works from the innermost parenthesis to the outermost (just like a calculator).

```
factorial(round(2.0015) + 1)
```



```
factorial(2 + 1)
```



```
factorial(3)
```



```
6
```

R always works from the innermost parenthesis to the outermost (just like a calculator).

```
factorial(round(2.0015) + 1)
```



```
factorial(2 + 1)
```



```
factorial(3)
```



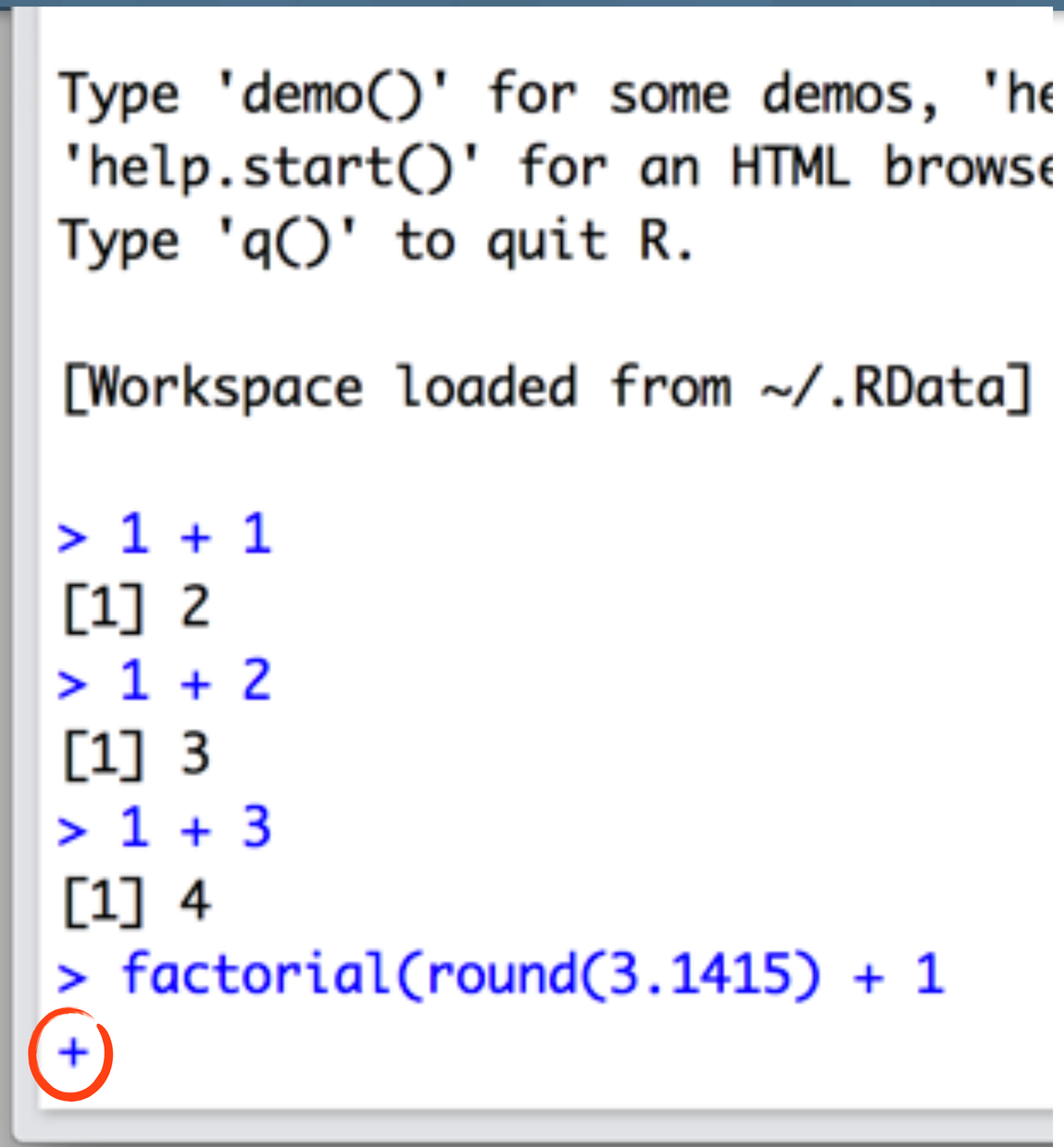
```
6
```

3! = 3 x 2 x 1

+ prompt

If your prompt turns into a "+", R thinks you haven't finished your previous command.

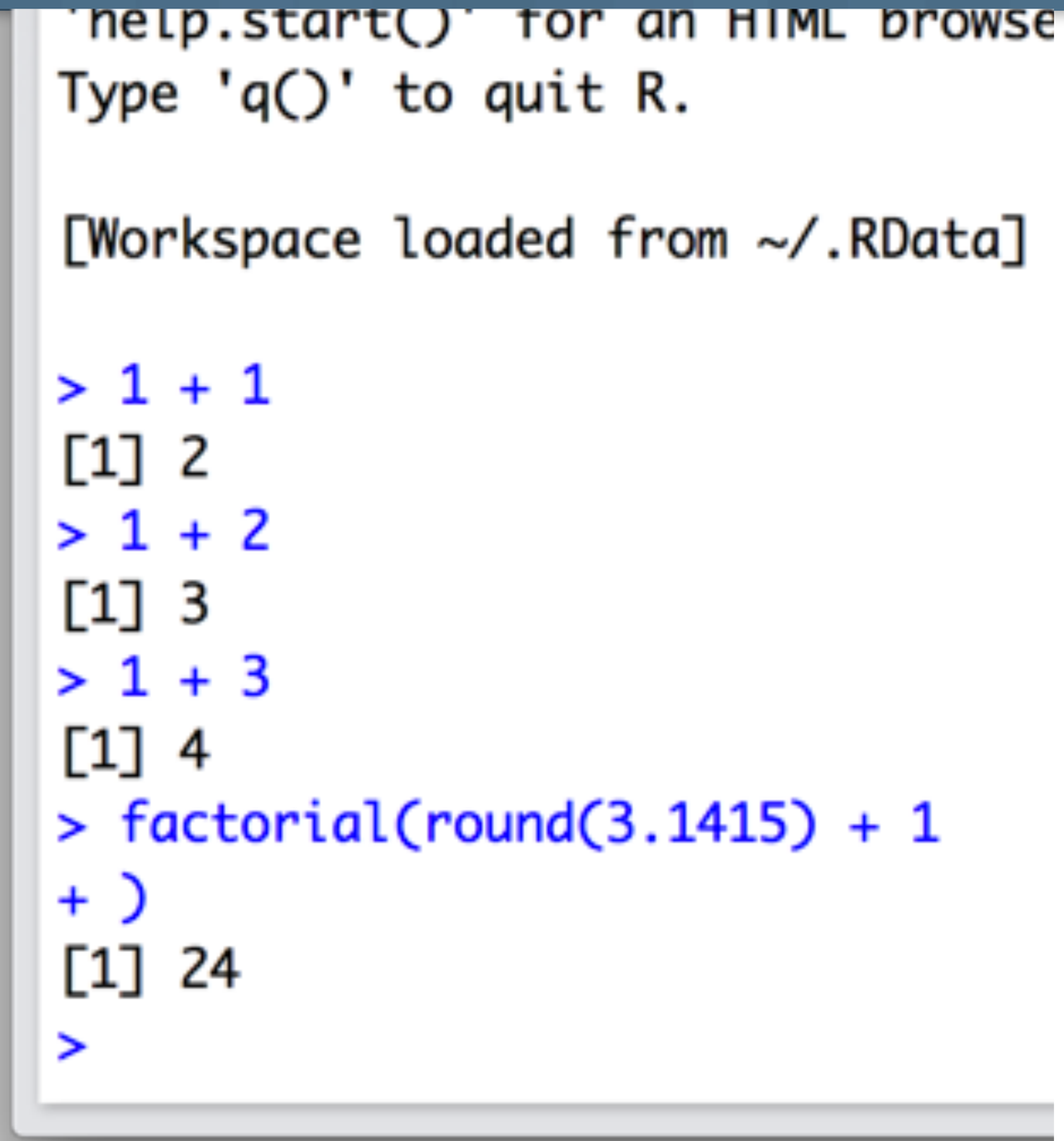
Either finish the command, or press escape.



+ prompt

If your prompt turns into a "+", R thinks you haven't finished your previous command.

Either finish the command, or press escape.



Your turn

Open RStudio and try the following tasks:

1. Pick a number and add 2 to it
2. Multiply the result by 3
3. Subtract 6 from the result of step 2
4. Divide the result of step 3 by 3

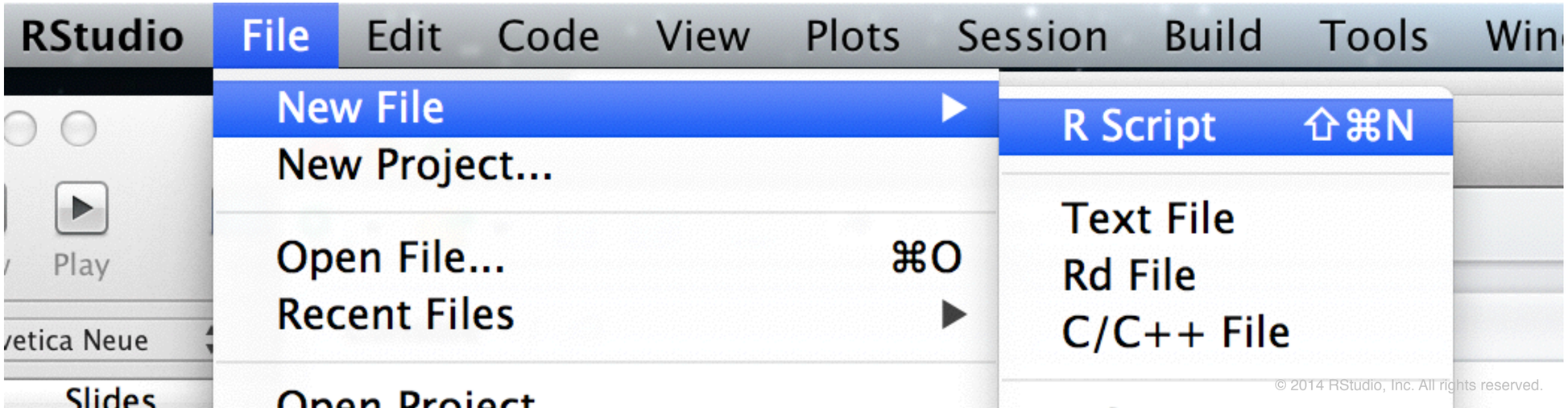
```
10 + 2
# 12

12 * 3
# 36

36 - 6
# 30

30 / 3
# 10
```

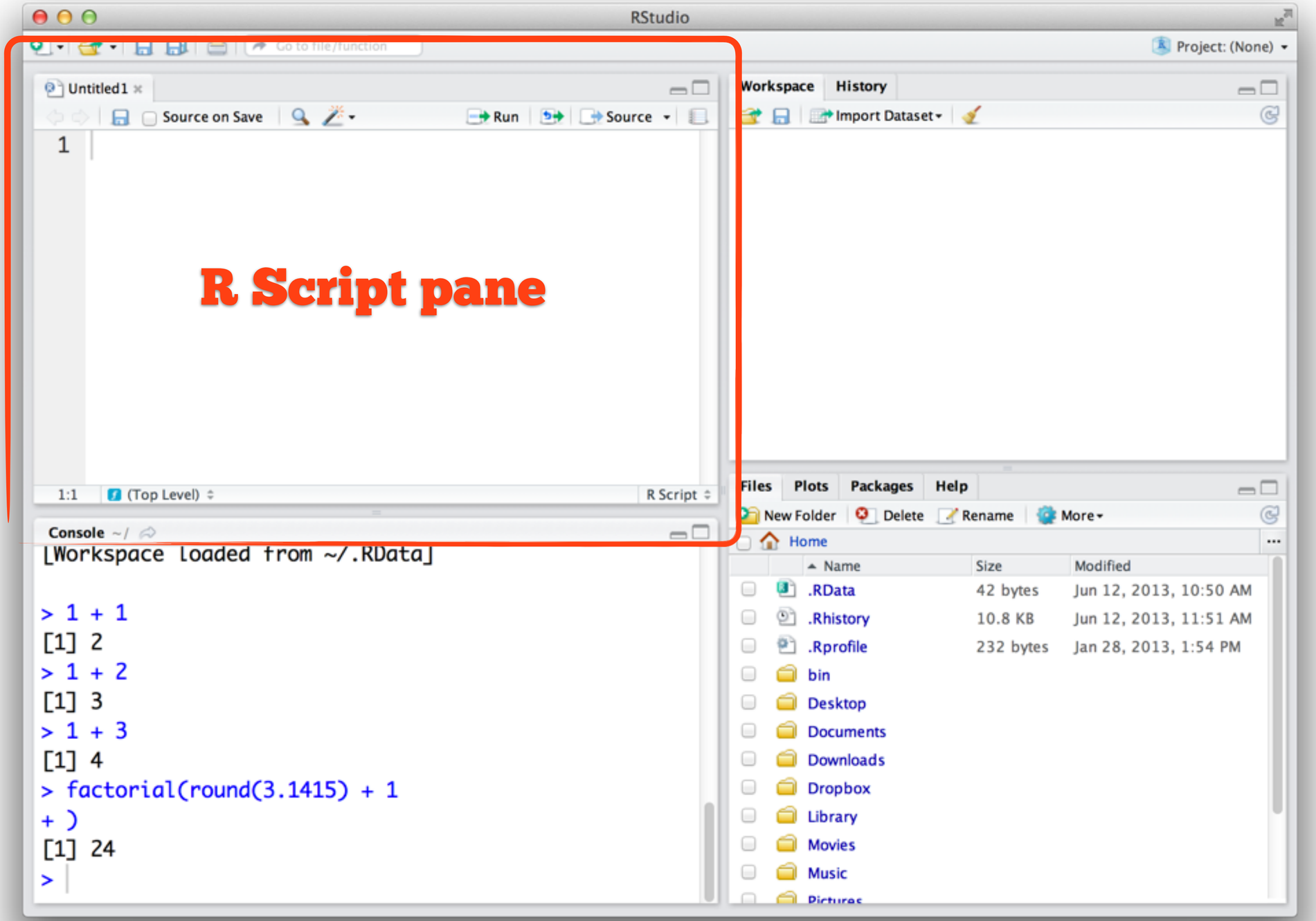
Workflow

R scripts

It is easier to compose your code in an R script than in the command line.

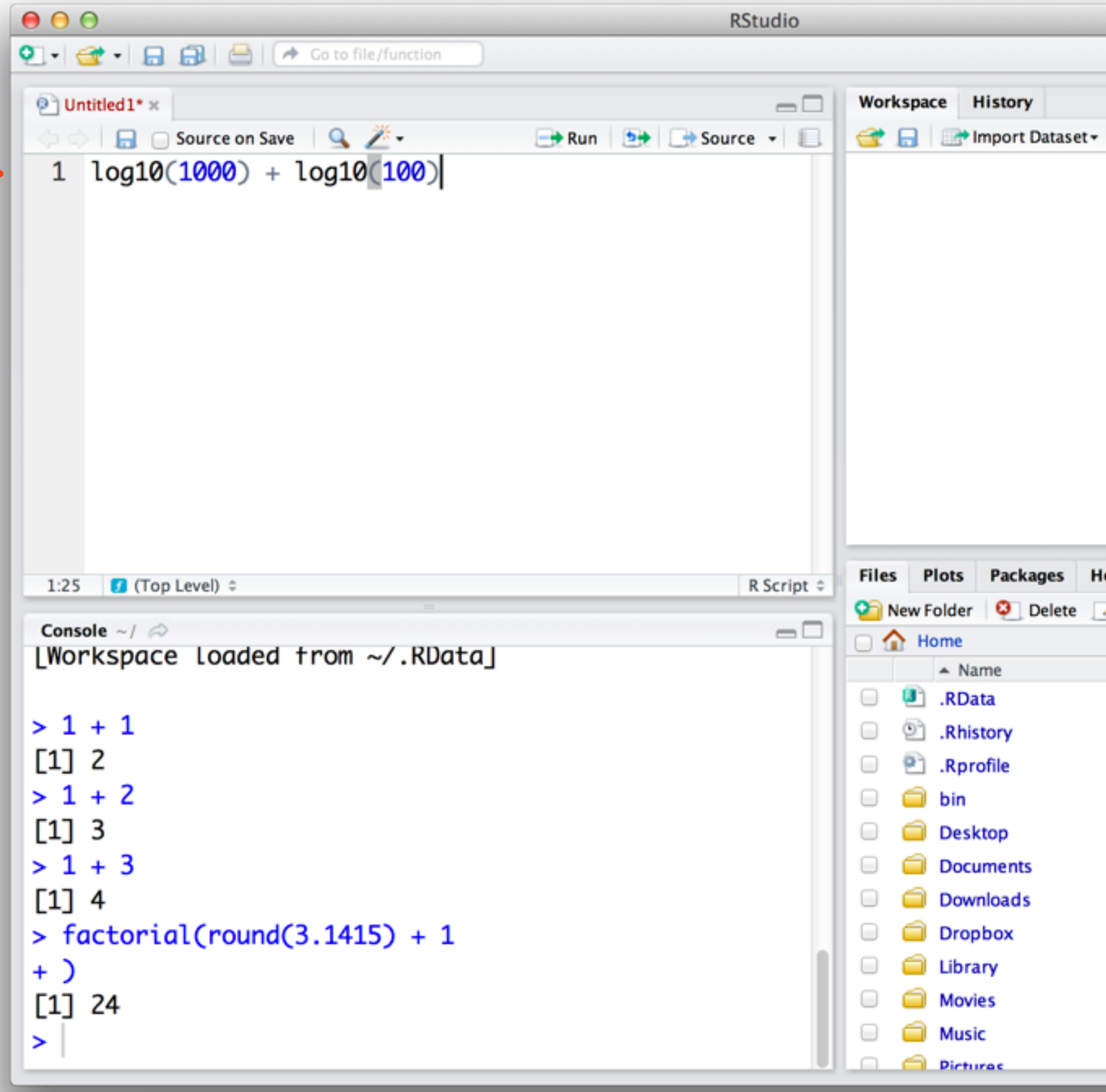
To open an R script, go to File>New File>R Script in the toolbar.





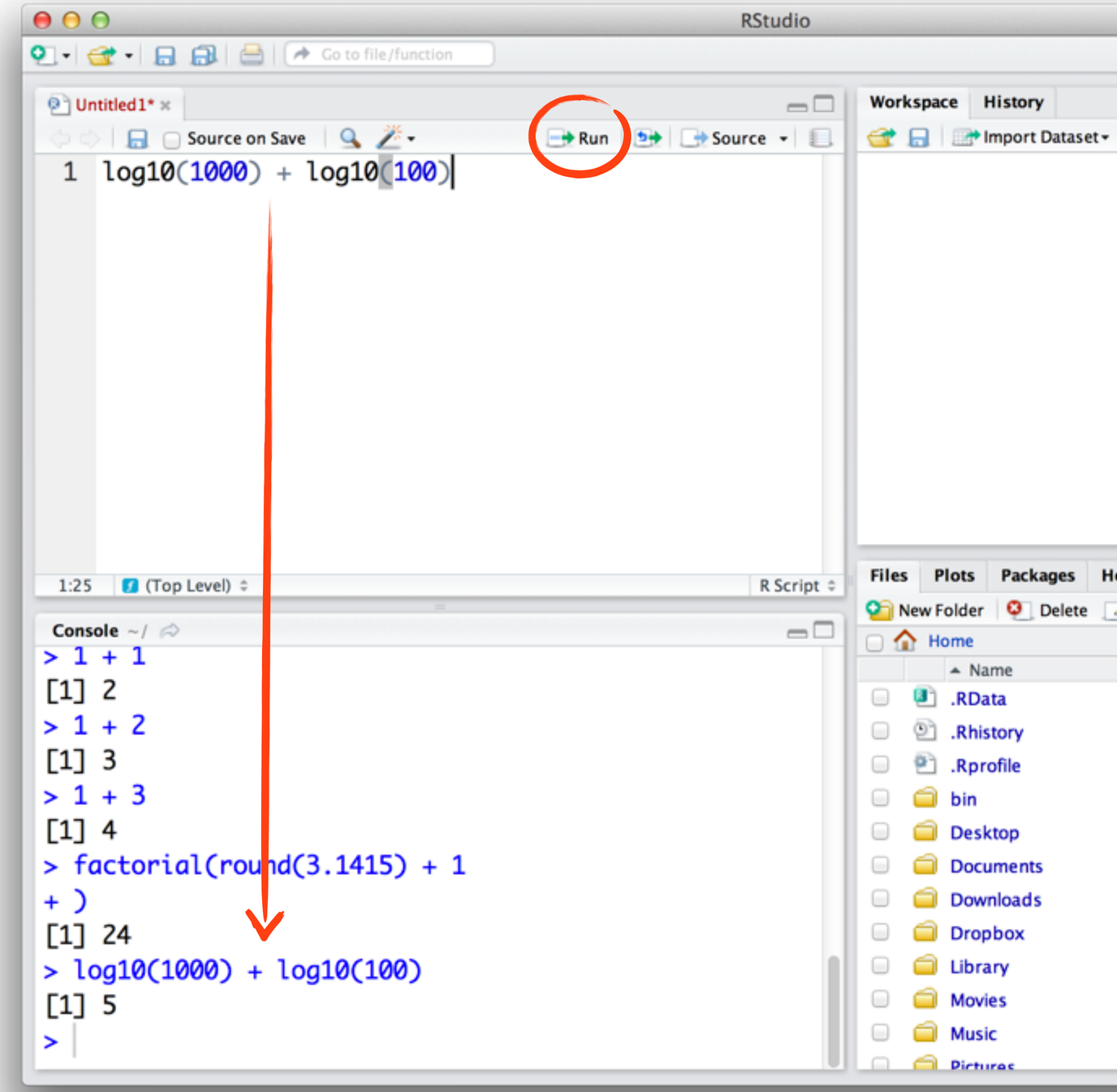
Common workflow

1. Write code in an R script



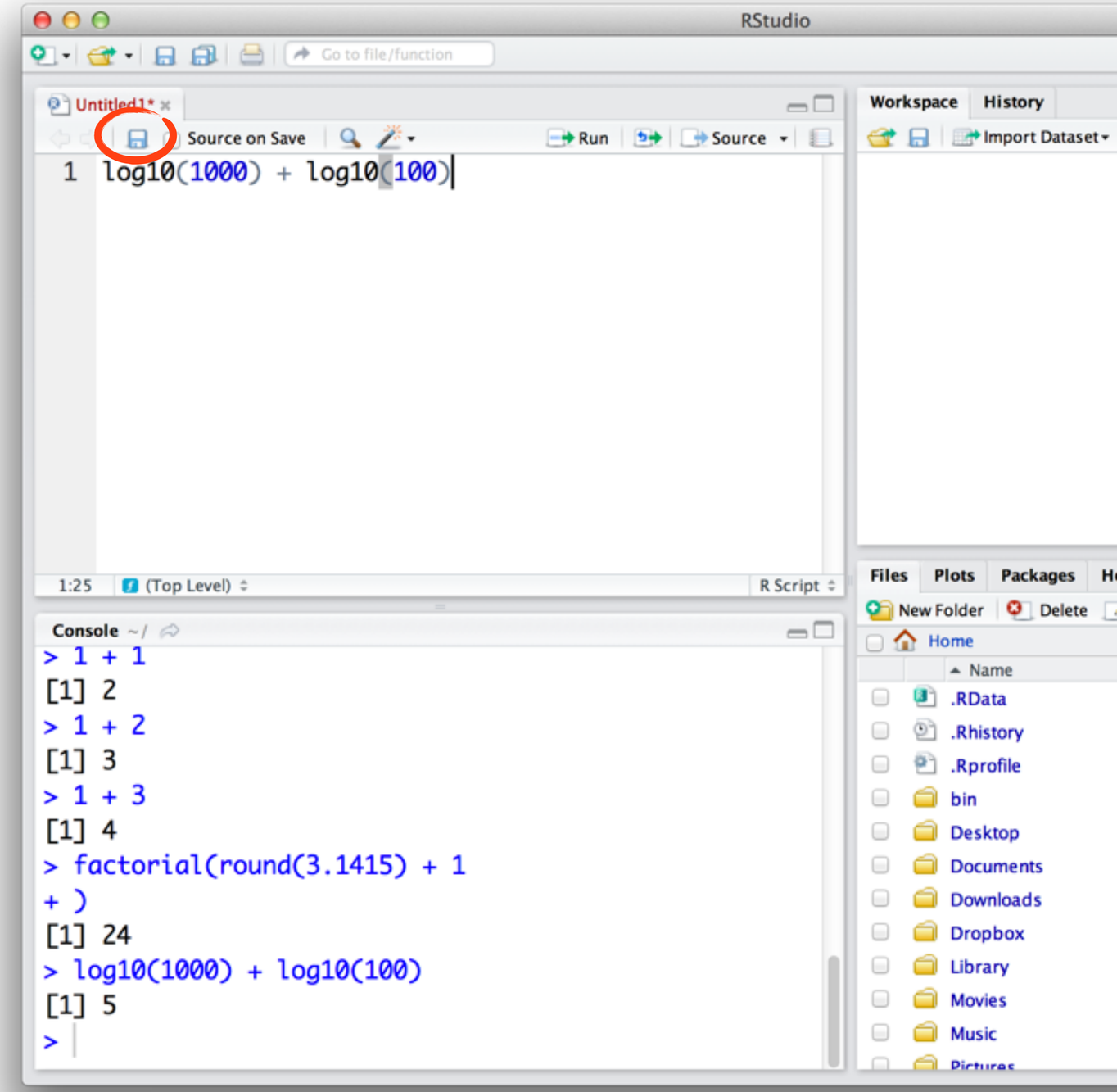
Common workflow

1. Write code in an R script
2. Run code in console with Run

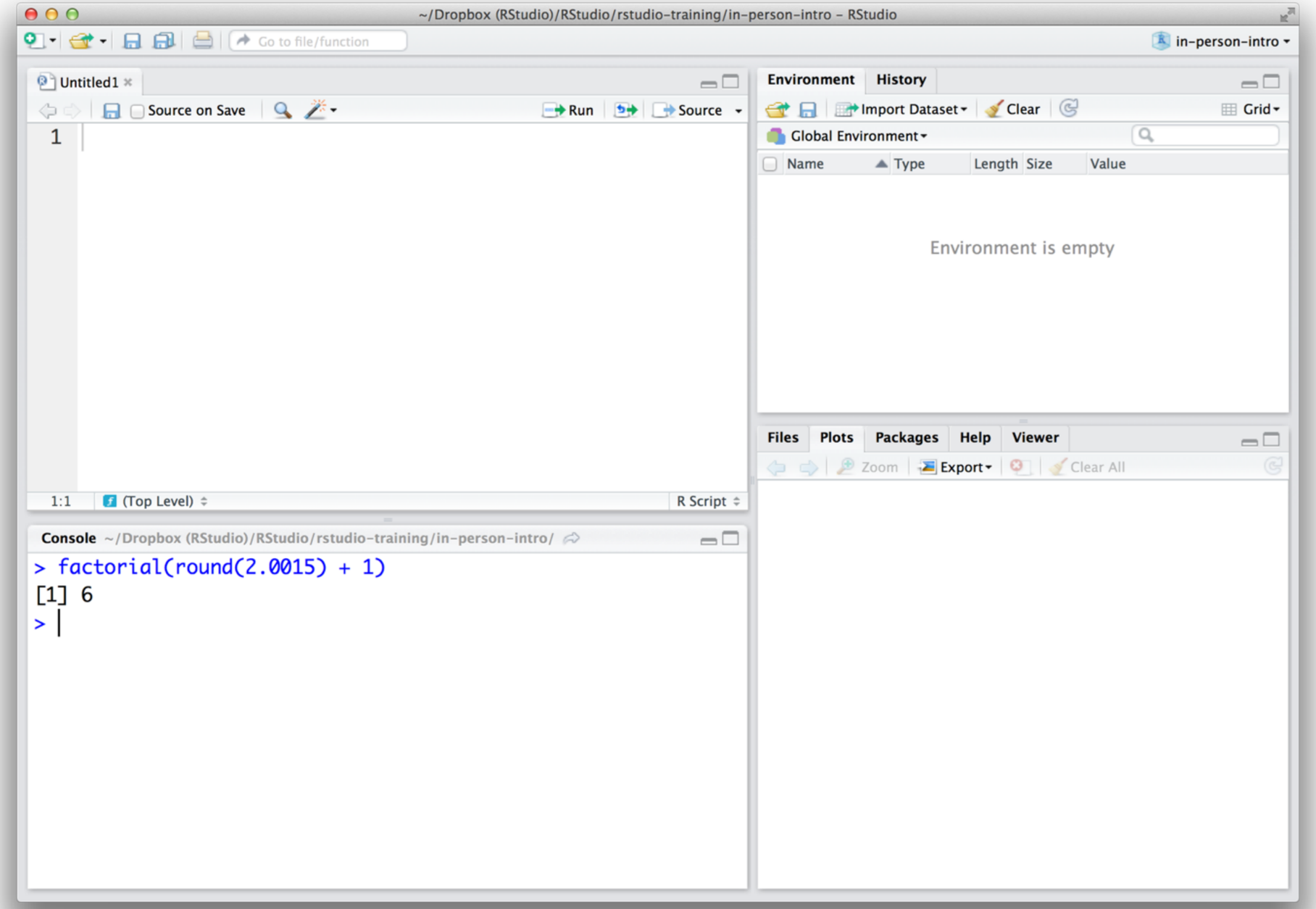


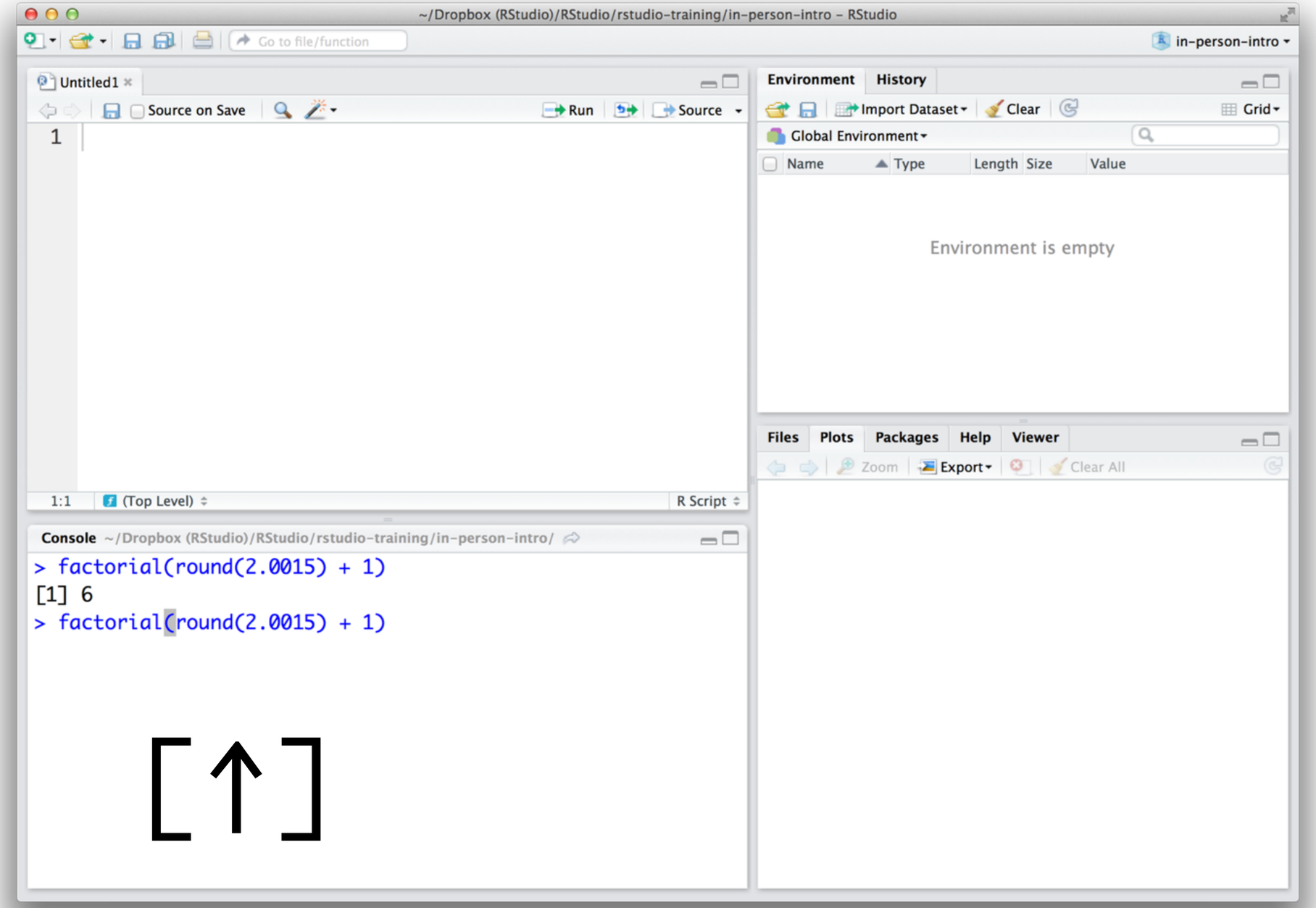
Common workflow

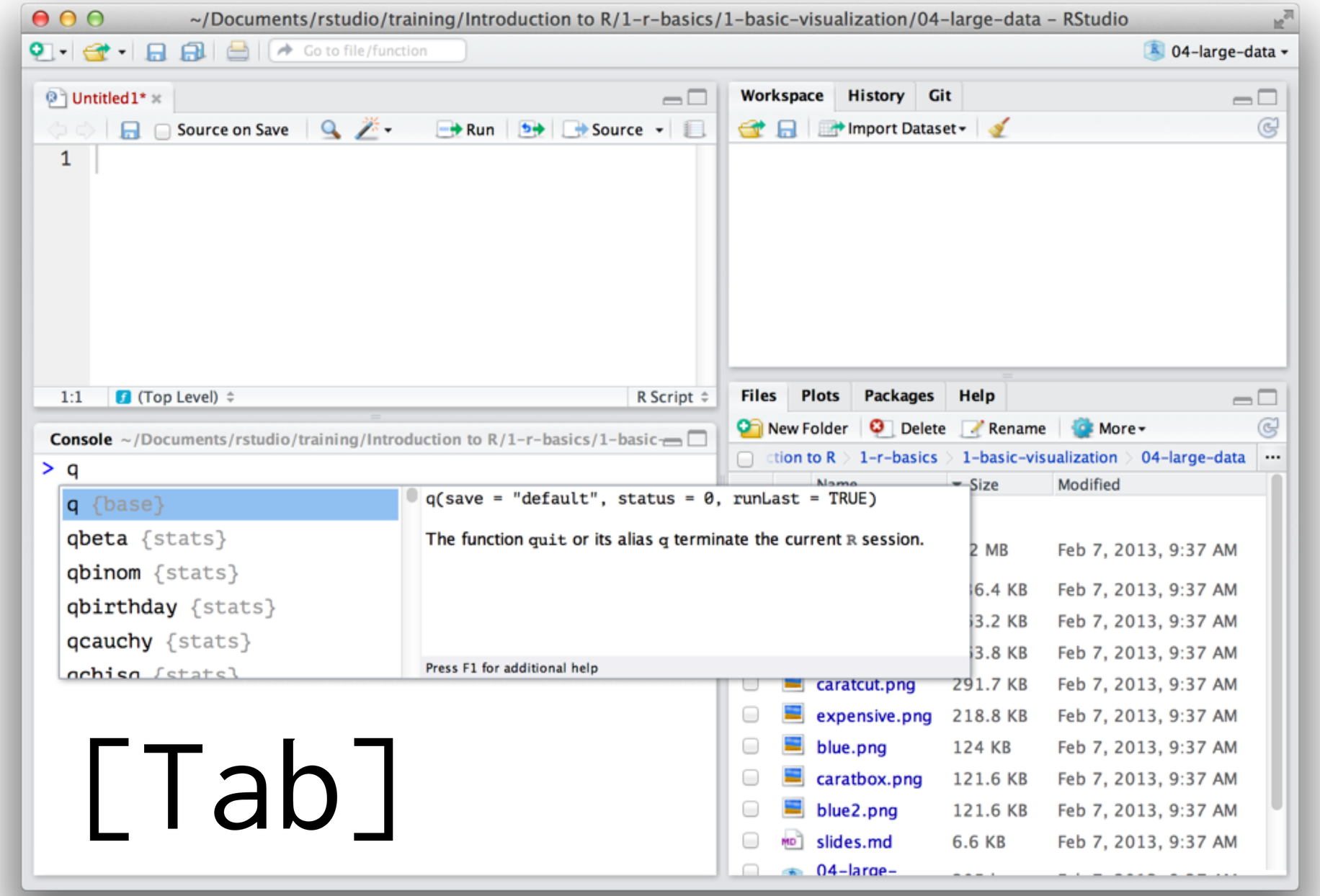
1. Write code in an R script
2. Run code in console with Run
3. Save R Script when finished

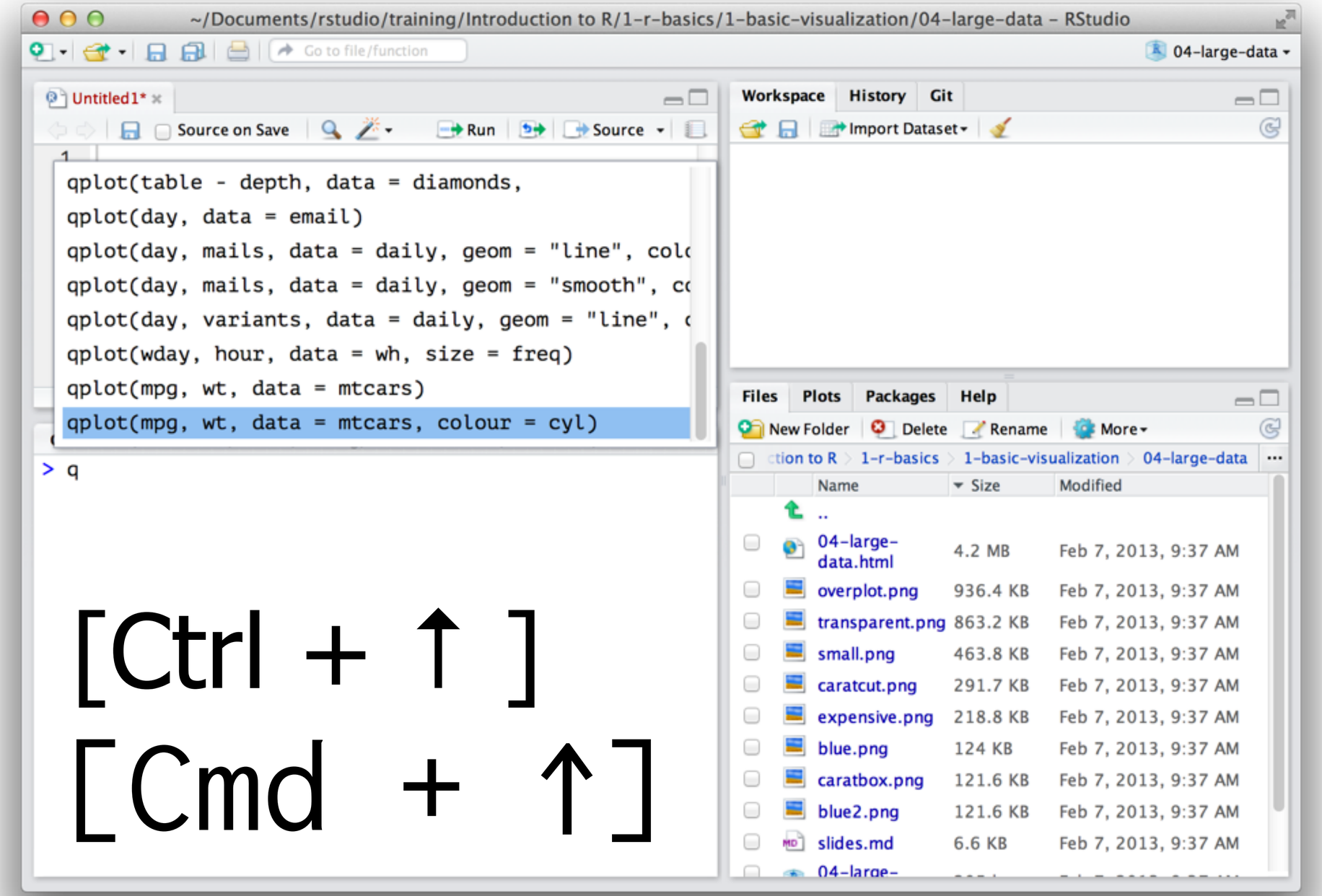


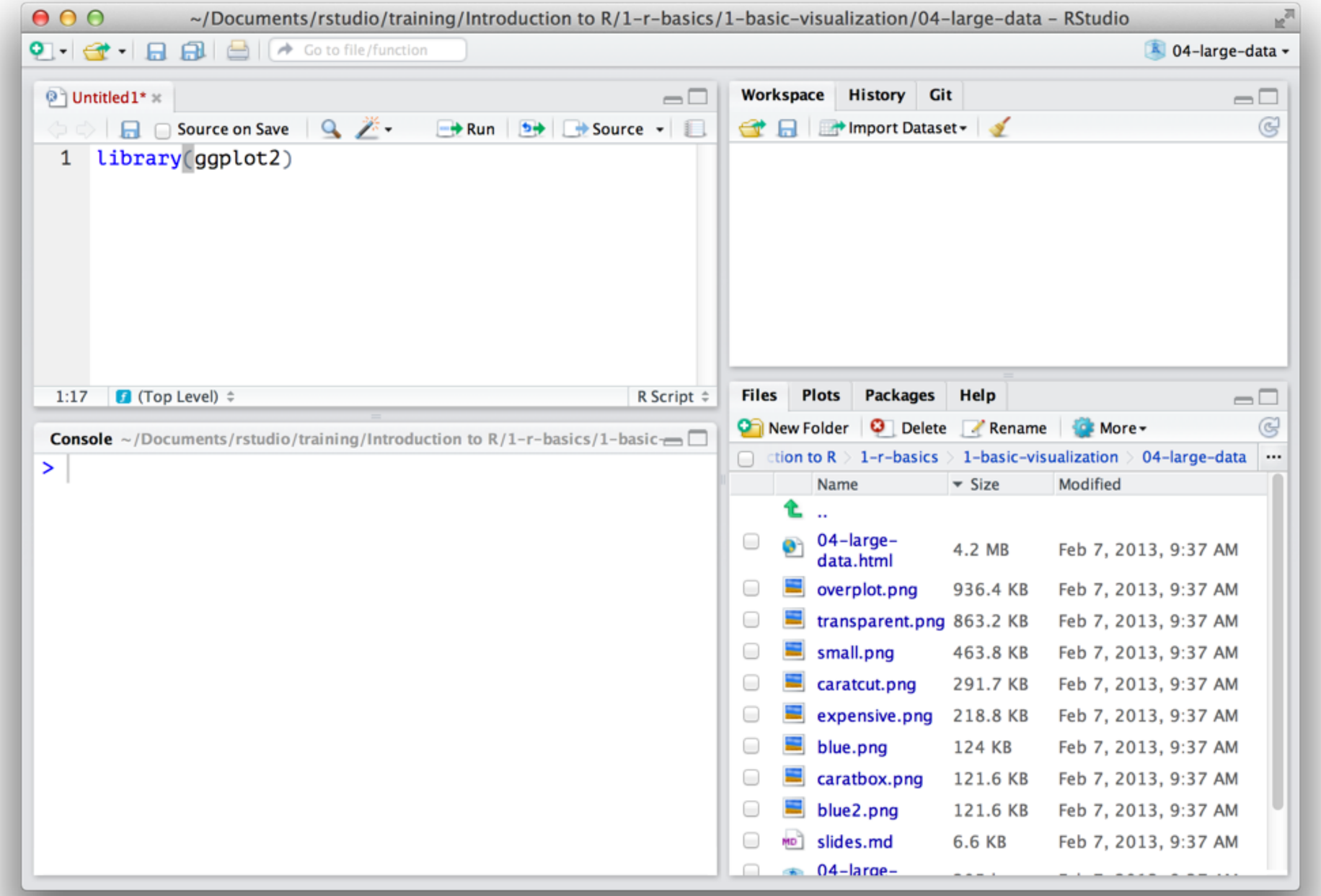
Shortcuts

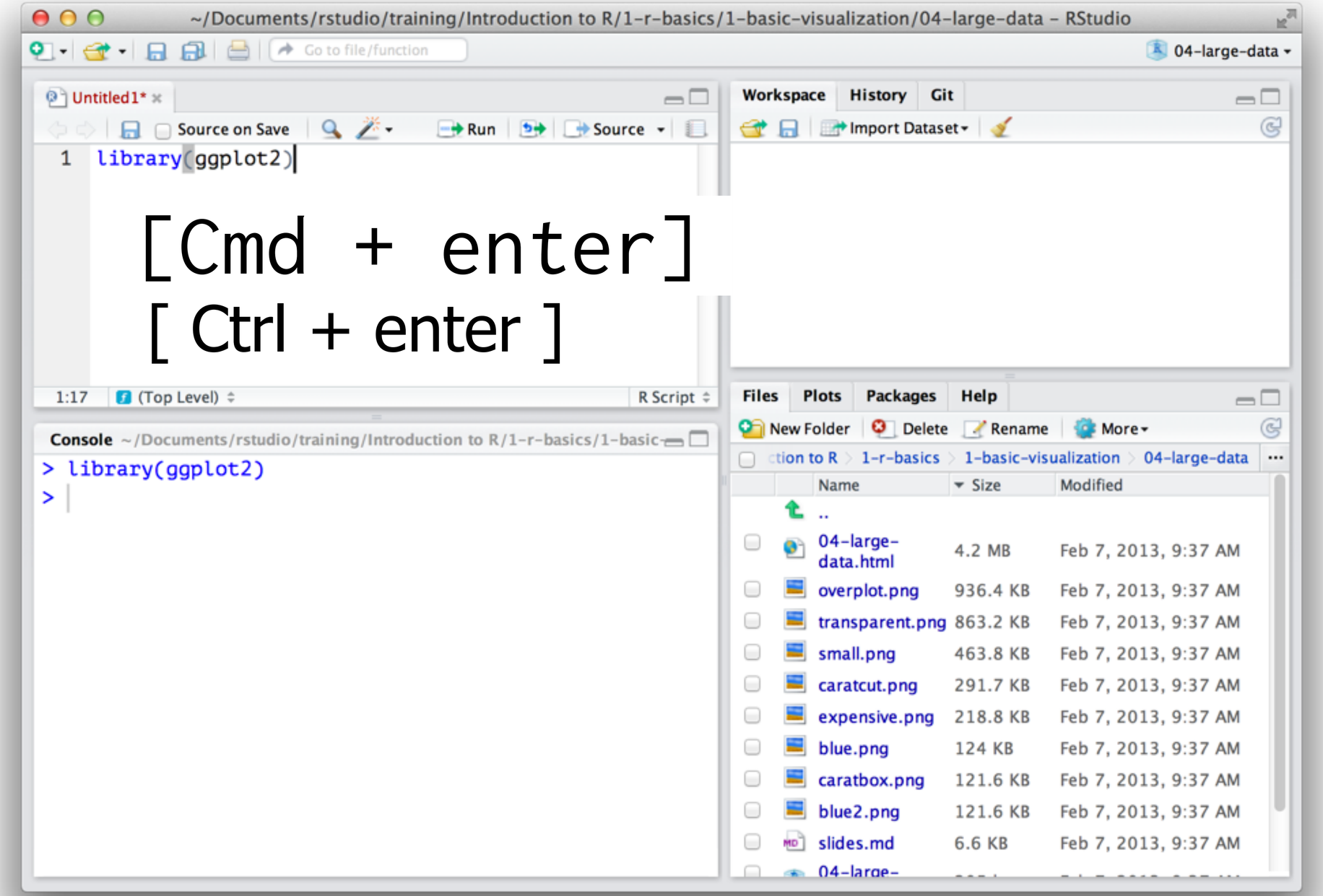

~/Dropbox (RStudio)/RStudio/rstudio-training/in-person-intro – RStudio
Go to file/function
in-person-intro
Untitled1
Source on Save
Run
Source
1
1:1
(Top Level)
R Script
Console ~/Dropbox (RStudio)/RStudio/rstudio-training/in-person-intro/
> factorial(round(2.0015) + 1)
[1] 6
>
Environment
History
Import Dataset
Clear
Grid
Global Environment
Name
Type
Length
Size
Value
Environment is empty
Files
Plots
Packages
Help
Viewer
Zoom
Export
Clear All


~/Dropbox (RStudio)/RStudio/rstudio-training/in-person-intro – RStudio
Go to file/function
in-person-intro
Untitled1
Source on Save
Run
Source
1
1:1
(Top Level)
R Script
Console ~/Dropbox (RStudio)/RStudio/rstudio-training/in-person-intro/
> factorial(round(2.0015) + 1)
[1] 6
> factorial(round(2.0015) + 1)
[↑]
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


~/Documents/rstudio/training/Introduction to R/1-r-basics/1-basic-visualization/04-large-data - RStudio
Go to file/function
04-large-data
Untitled1*
Source on Save
Run
Source
1
1:1
(Top Level)
R Script
Workspace
History
Git
Import Dataset
Files
Plots
Packages
Help
New Folder
Delete
Rename
More
1-r-basics
1-basic-visualization
04-large-data
Size
Modified
Console ~/Documents/rstudio/training/Introduction to R/1-r-basics/1-basic-
> q
q {base}
qbeta {stats}
qbinom {stats}
qbirthday {stats}
qcauchy {stats}
q(save = "default", status = 0, runLast = TRUE)
The function quit or its alias q terminate the current R session.
Press F1 for additional help
caratcut.png 291.7 KB Feb 7, 2013, 9:37 AM
expensive.png 218.8 KB Feb 7, 2013, 9:37 AM
blue.png 124 KB Feb 7, 2013, 9:37 AM
caratbox.png 121.6 KB Feb 7, 2013, 9:37 AM
blue2.png 121.6 KB Feb 7, 2013, 9:37 AM
slides.md 6.6 KB Feb 7, 2013, 9:37 AM
[Tab]


~/Documents/rstudio/training/Introduction to R/1-r-basics/1-basic-visualization/04-large-data - RStudio
Go to file/function
04-large-data
Untitled1*
Source on Save
Run
Source
qplot(table - depth, data = diamonds,
qplot(day, data = email)
qplot(day, mails, data = daily, geom = "line", colo
qplot(day, mails, data = daily, geom = "smooth", co
qplot(day, variants, data = daily, geom = "line", c
qplot(wday, hour, data = wh, size = freq)
qplot(mpg, wt, data = mtcars)
qplot(mpg, wt, data = mtcars, colour = cyl)
> q
[Ctrl + ↑]
[Cmd + ↑]
Workspace
History
Git
Import Dataset
Files
Plots
Packages
Help
New Folder
Delete
Rename
More
1-r-basics
1-basic-visualization
04-large-data
Name
Size
Modified
04-large-data.html 4.2 MB Feb 7, 2013, 9:37 AM
overplot.png 936.4 KB Feb 7, 2013, 9:37 AM
transparent.png 863.2 KB Feb 7, 2013, 9:37 AM
small.png 463.8 KB Feb 7, 2013, 9:37 AM
caratcut.png 291.7 KB Feb 7, 2013, 9:37 AM
expensive.png 218.8 KB Feb 7, 2013, 9:37 AM
blue.png 124 KB Feb 7, 2013, 9:37 AM
caratbox.png 121.6 KB Feb 7, 2013, 9:37 AM
blue2.png 121.6 KB Feb 7, 2013, 9:37 AM
slides.md 6.6 KB Feb 7, 2013, 9:37 AM


~/Documents/rstudio/training/Introduction to R/1-r-basics/1-basic-visualization/04-large-data - RStudio
Go to file/function
04-large-data
Untitled1*
Source on Save
Run
Source
1 library(ggplot2)
1:17
(Top Level)
R Script
Console ~/Documents/rstudio/training/Introduction to R/1-r-basics/1-basic-
>
Workspace
History
Git
Import Dataset
Files
Plots
Packages
Help
New Folder
Delete
Rename
More
ction to R
1-r-basics
1-basic-visualization
04-large-data
Name
Size
Modified
..
04-large-data.html 4.2 MB Feb 7, 2013, 9:37 AM
overplot.png 936.4 KB Feb 7, 2013, 9:37 AM
transparent.png 863.2 KB Feb 7, 2013, 9:37 AM
small.png 463.8 KB Feb 7, 2013, 9:37 AM
caratcut.png 291.7 KB Feb 7, 2013, 9:37 AM
expensive.png 218.8 KB Feb 7, 2013, 9:37 AM
blue.png 124 KB Feb 7, 2013, 9:37 AM
caratbox.png 121.6 KB Feb 7, 2013, 9:37 AM
blue2.png 121.6 KB Feb 7, 2013, 9:37 AM
slides.md 6.6 KB Feb 7, 2013, 9:37 AM
04-large-


~/Documents/rstudio/training/Introduction to R/1-r-basics/1-basic-visualization/04-large-data - RStudio
Go to file/function
04-large-data
Untitled1*
Source on Save
Run
Source
1 library(ggplot2)
[Cmd + enter]
[Ctrl + enter]
1:17
(Top Level)
R Script
Workspace
History
Git
Import Dataset
Files
Plots
Packages
Help
New Folder
Delete
Rename
More
1-r-basics
1-basic-visualization
04-large-data
Name
Size
Modified
Console ~/Documents/rstudio/training/Introduction to R/1-r-basics/1-basic-
> library(ggplot2)
>
04-large-data.html 4.2 MB Feb 7, 2013, 9:37 AM
overplot.png 936.4 KB Feb 7, 2013, 9:37 AM
transparent.png 863.2 KB Feb 7, 2013, 9:37 AM
small.png 463.8 KB Feb 7, 2013, 9:37 AM
caratcut.png 291.7 KB Feb 7, 2013, 9:37 AM
expensive.png 218.8 KB Feb 7, 2013, 9:37 AM
blue.png 124 KB Feb 7, 2013, 9:37 AM
caratbox.png 121.6 KB Feb 7, 2013, 9:37 AM
blue2.png 121.6 KB Feb 7, 2013, 9:37 AM
slides.md 6.6 KB Feb 7, 2013, 9:37 AM

R objects

Save information as an R object with the greater than sign followed by a minus, e.g, an arrow: <-

```
foo <- 42
```

Save information as an R object with the greater than sign followed by a minus, e.g, an arrow: <-

name of new object

```
foo <- 42
```

Save information as an R object with the greater than sign followed by a minus, e.g, an arrow: <-

assignment operator, "gets"

```
foo <- 42
```

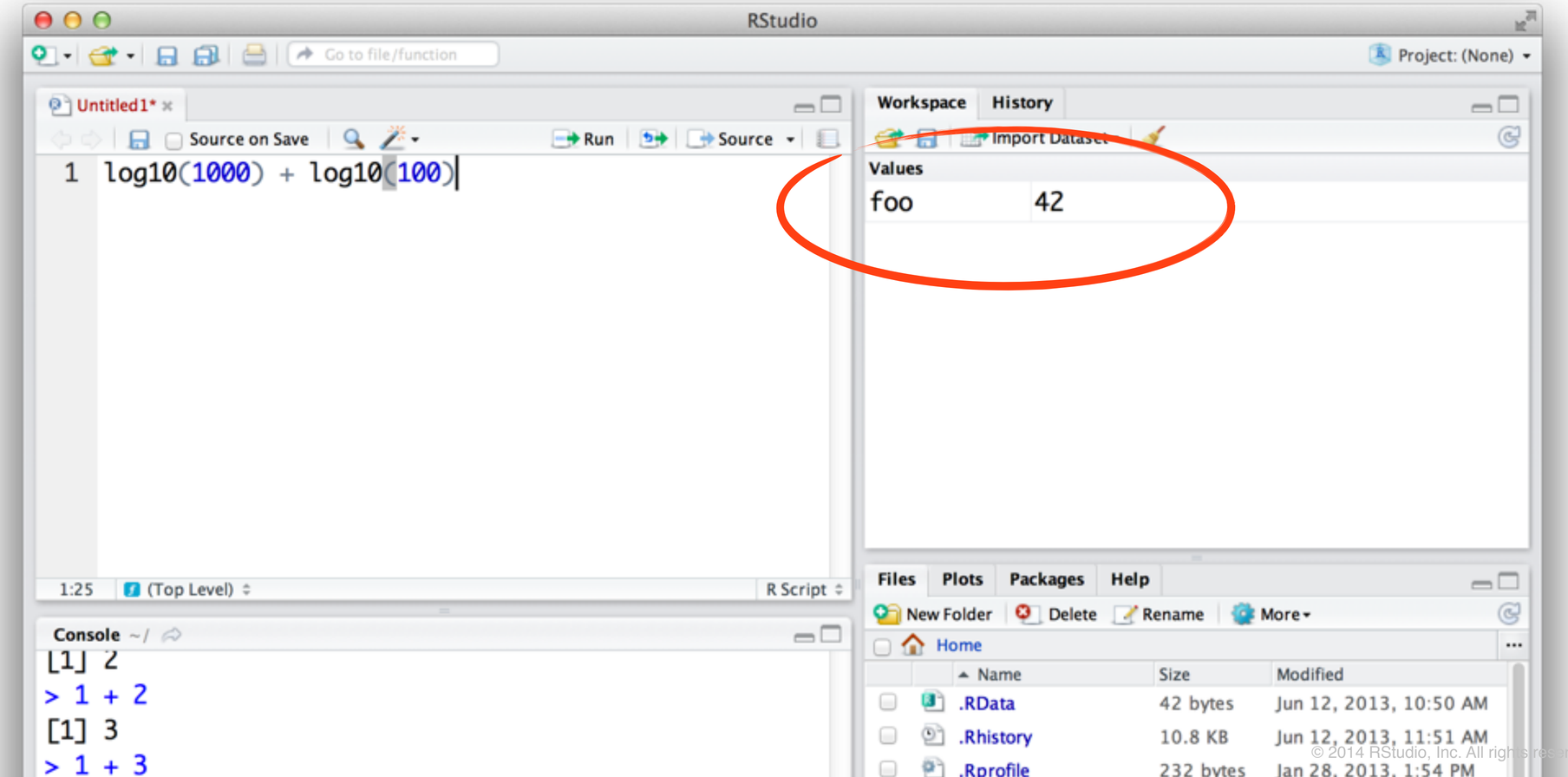
Save information as an R object with the greater than sign followed by a minus, e.g, an arrow: <-

information to store in the object

```
foo <- 42
```

When you create an R object, you'll see it appear in your workspace pane

(or Environment pane)



Common R workflow

Save output of one function as an R object to use in a second function.

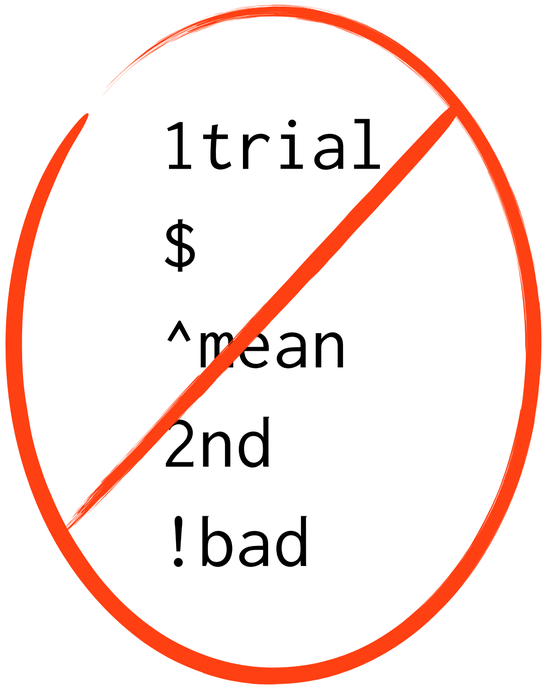
```
foo <- round(3.1415) + 1
foo
# 4

factorial(foo)
# 24
```

Object names

Object names cannot begin with numbers.

Wise to avoid names already in use.

```
a
b
FOO
my_var
.day
```



Capitalization matters

R will treat each of these as a different object

```
a       A
b       B
sum     SUM
```

rm

You can remove an object with `rm`

```
foo
# 4

rm(foo)
foo
# Error: object 'foo' not found
```

This can be useful if you overwrite an object that comes with R

```
pi
# 3.141593

pi <- 1
pi
# 1

rm(pi)
pi
# 3.141593
```

Data structures

Warm up

Look at the R object `WorldPhones` (by typing its name in the console).

What is inside of `WorldPhones`?

```
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
```

You can save more than a single number in an object by creating a *vector*, *matrix*, or *array*.

vectors

Combine multiple elements into a one dimensional array.

Create with the c function.

```
vec <- c(1, 2, 3, 10, 100)
vec
# 1   2   3  10 100
```

vectors

multiple elements stored in a one dimensional array.

Create with the c function.

```
vec <- c(1, 2, 3, 10, 100)
vec
# 1   2   3  10 100
```

matrices

multiple elements stored in a two dimensional array.

Create with the `matrix` function.

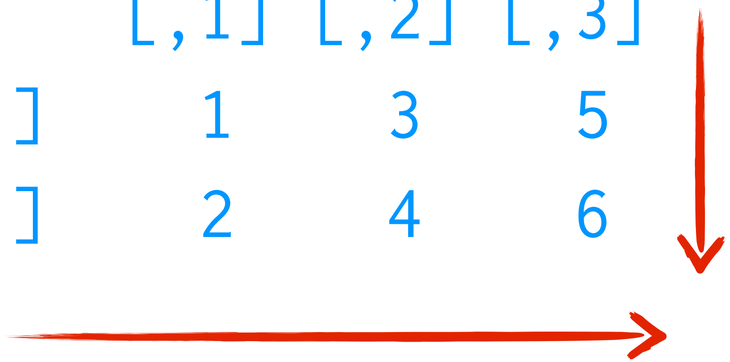
```
mat <- matrix(c(1, 2, 3, 4, 5, 6), nrow = 2)
mat
#      [,1] [,2] [,3]
# [1,]    1    3    5
# [2,]    2    4    6
```

matrices

Combine multiple elements into a two dimensional array.

Create with the `matrix` function.

```
mat <- matrix(c(1, 2, 3, 4, 5, 6), nrow = 2)
mat
#      [,1] [,2] [,3]
# [1,]    1    3    5
# [2,]    2    4    6
```



vector of elements to go in the matrix

```
matrix(c(1, 2, 3, 4, 5, 6), nrow = 2)
#      [,1] [,2] [,3]
# [1,]    1    3    5
# [2,]    2    4    6
```

number of rows for matrix

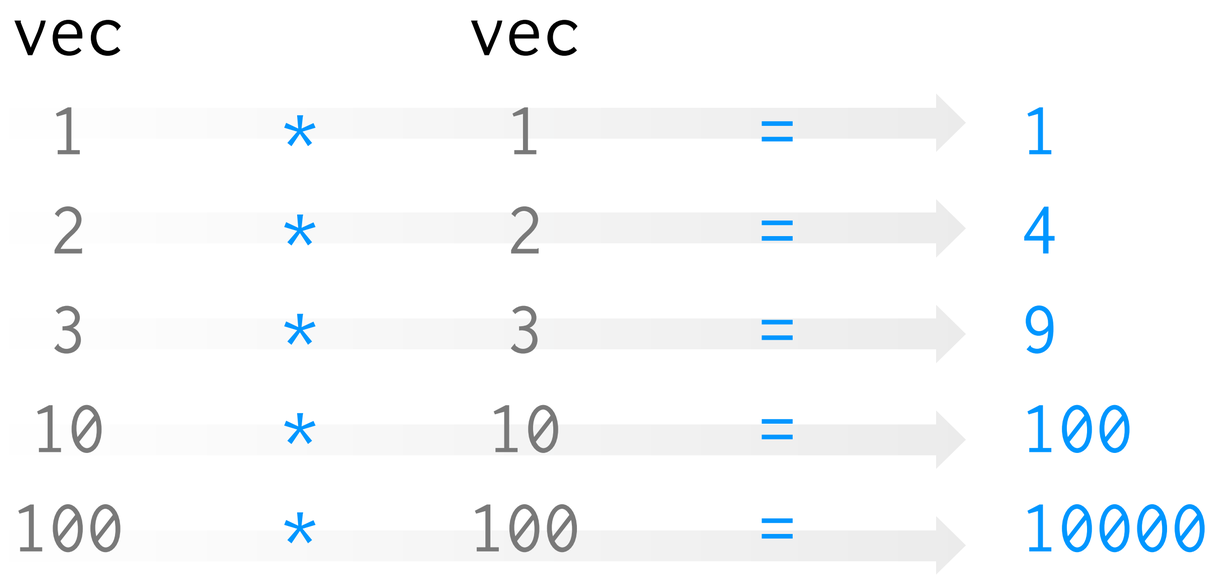
```
matrix(c(1, 2, 3, 4, 5, 6), nrow = 2)
#      [,1] [,2] [,3]
# [1,]    1    3    5
# [2,]    2    4    6
```

```
matrix(c(1, 2, 3, 4, 5, 6), nrow = 3)
#      [,1] [,2]
# [1,]    1    4
# [2,]    2    5
# [3,]    3    6
```

Math: element-wise

```
vec + 4
# 5   6   7  14 104

vec * 4
# 4   8  12  40 400

vec * vec
# 1     4     9   100 10000
```

```
vec * vec
# 1     4     9   100 10000
```



Matrix multiplication

```
vec %*% vec # inner
#      [,1]
# [1,] 10114
```

```
vec %o% vec # outer
#      [,1] [,2] [,3] [,4]  [,5]
# [1,]    1    2    3   10   100
# [2,]    2    4    6   20   200
# [3,]    3    6    9   30   300
# [4,]   10   20   30  100  1000
# [5,]  100  200  300 1000 10000
```

```
mat
#      [,1] [,2] [,3]
# [1,]    1    3    5
# [2,]    2    4    6
```

```
t(mat)
#      [,1] [,2]
# [1,]    1    2
# [2,]    3    4
# [3,]    5    6
```

arrays

Combine multiple elements into an array that has three or more dimensions.

Create with the `array` function.

```
array(c(1, 2, 3, 4, 5, 6), dim = c(2, 2, 3))
```

arrays

Combine multiple elements into an array that has three or more dimensions.

Create with the `array` function.

```
array(c(1, 2, 3, 4, 5, 6), dim = c(2, 2,
```

We won't focus on arrays

Data types

Warm up

	A	B	C	D
1	date	president	democrat	unemploy
2	Mar 31, 1968	Lyndon Johnson	TRUE	2709
3	Apr 30, 1968	Lyndon Johnson	TRUE	2740
4	May 31, 1968	Lyndon Johnson	TRUE	2938
5	Jun 30, 1968	Lyndon Johnson	TRUE	2883
6	Jul 31, 1968	Lyndon Johnson	TRUE	2768
7	Aug 31, 1968	Lyndon Johnson	TRUE	2686
8	Sep 30, 1968	Lyndon Johnson	TRUE	2689
9	Oct 31, 1968	Lyndon Johnson	TRUE	2715
10	Nov 30, 1968	Lyndon Johnson	TRUE	2685
11	Dec 31, 1968	Lyndon Johnson	TRUE	2718
12	Jan 31, 1969	Richard Nixon	FALSE	2692
13	Feb 28, 1969	Richard Nixon	FALSE	2712
14	Mar 31, 1969	Richard Nixon	FALSE	2758
15	Apr 30, 1969	Richard Nixon	FALSE	2713
16	May 31, 1969	Richard Nixon	FALSE	2816
17	Jun 30, 1969	Richard Nixon	FALSE	286
18	Jul 31, 1969	Richard Nixon	FALSE	2
19	Aug 31, 1969	Richard Nixon	FALSE	
20	Sep 30, 1969	Richard Nixon	FALSE	
21	Oct 31, 1969	Richard Nixon	FALSE	
22	Nov 30, 1969	Richard Nixon	FALSE	

What **types** of data appear in this spreadsheet?

data types

Like Excel, Numbers, etc., R can recognize different types of data.

We'll look at four basic types:

- numbers
- character strings (text)
- logical
- factor

numeric

Any number, no quotes.

Appropriate for math.

```
1 + 1
```

```
3000000
```

```
class(0.00001)
# "numeric"
```

character

Any symbols surrounded by quotes.

Appropriate for words, variable names, messages, any text.

```
"hello"

class("hello")
# "character"
```

```
"hello" + "world"
# Error

nchar("hello")
# 5

paste("hello", "world")
# "hello world"
```

Warm up

Which of these are numbers? **What are the others?** How can you tell?

1 "1" "one"

logical

TRUE or FALSE

R's form of binary data. Useful for logical tests.

```
3 < 4
# TRUE

class(TRUE)
# "logical"

class(T)
# "logical"
```

factor

R's form of categorical data. Saved as an integer with a set of labels (e.g. levels).

```
fac <- factor(c("a", "b", "c"))
fac
# a b c
# Levels: a b c

class(fac)
# factor
```

Quiz

```
x <- c(1, 2, 3)
```

What is the difference between these?

```
x
"x"
```

Type	Examples
numeric	0, 1, -2, 3.1415, 0.0005
character	"gender", "date", "31"
logical	TRUE, FALSE
factor	a c c b Levels: a b c

Aside: dates and times

Surprisingly difficult for computers!

We won't cover, but I recommend the following resources

http://www.r-statistics.com/2012/03/do-more-with-dates-and-times-in-r-with-lubridate-1-1-0/

http://www.jstatsoft.org/v40/i03/

Your turn

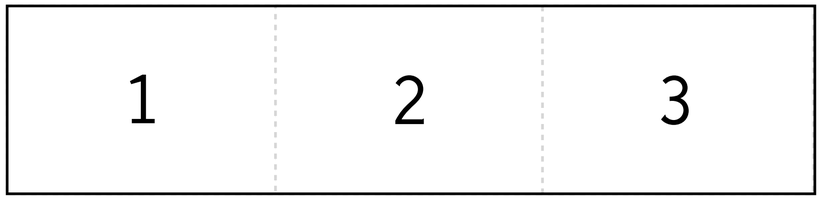
Make a vector that contains the number 1, the letter R, and the logical TRUE.

What class of data is the vector?

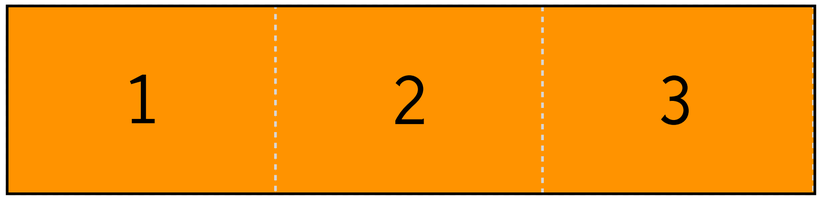
```
vec <- c(1, "R", TRUE)
class(vec)
# "character"

vec
# "1"    "R"    "TRUE"

# What is R doing?
```

Vector

Vector

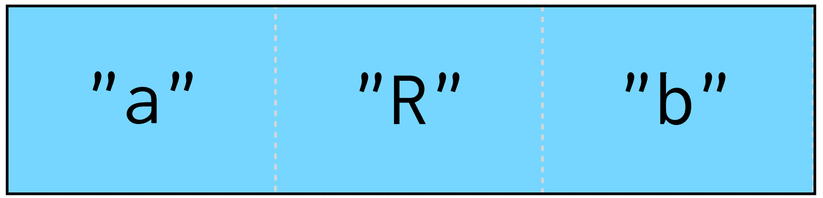


Vector



numeric

Vector

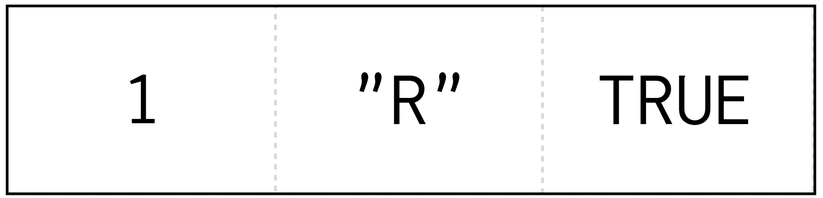


character

Vector

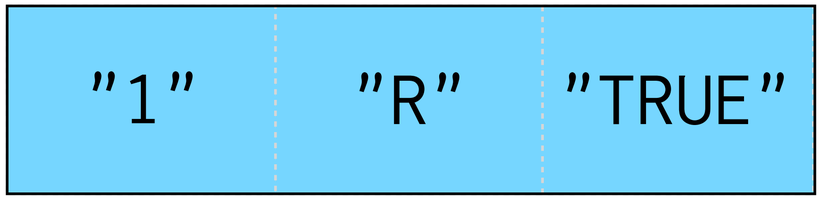
logical

Vector



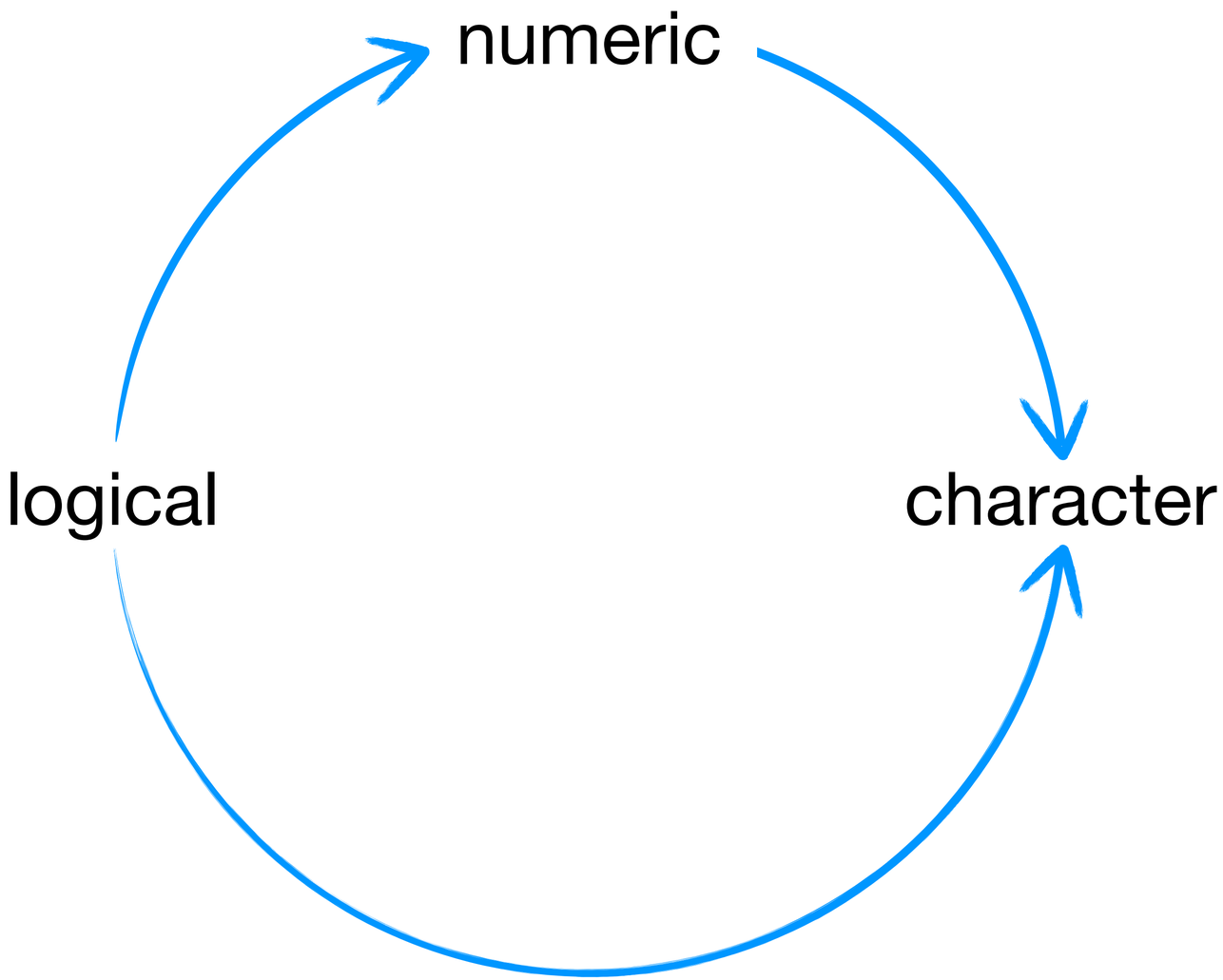
?

Vector



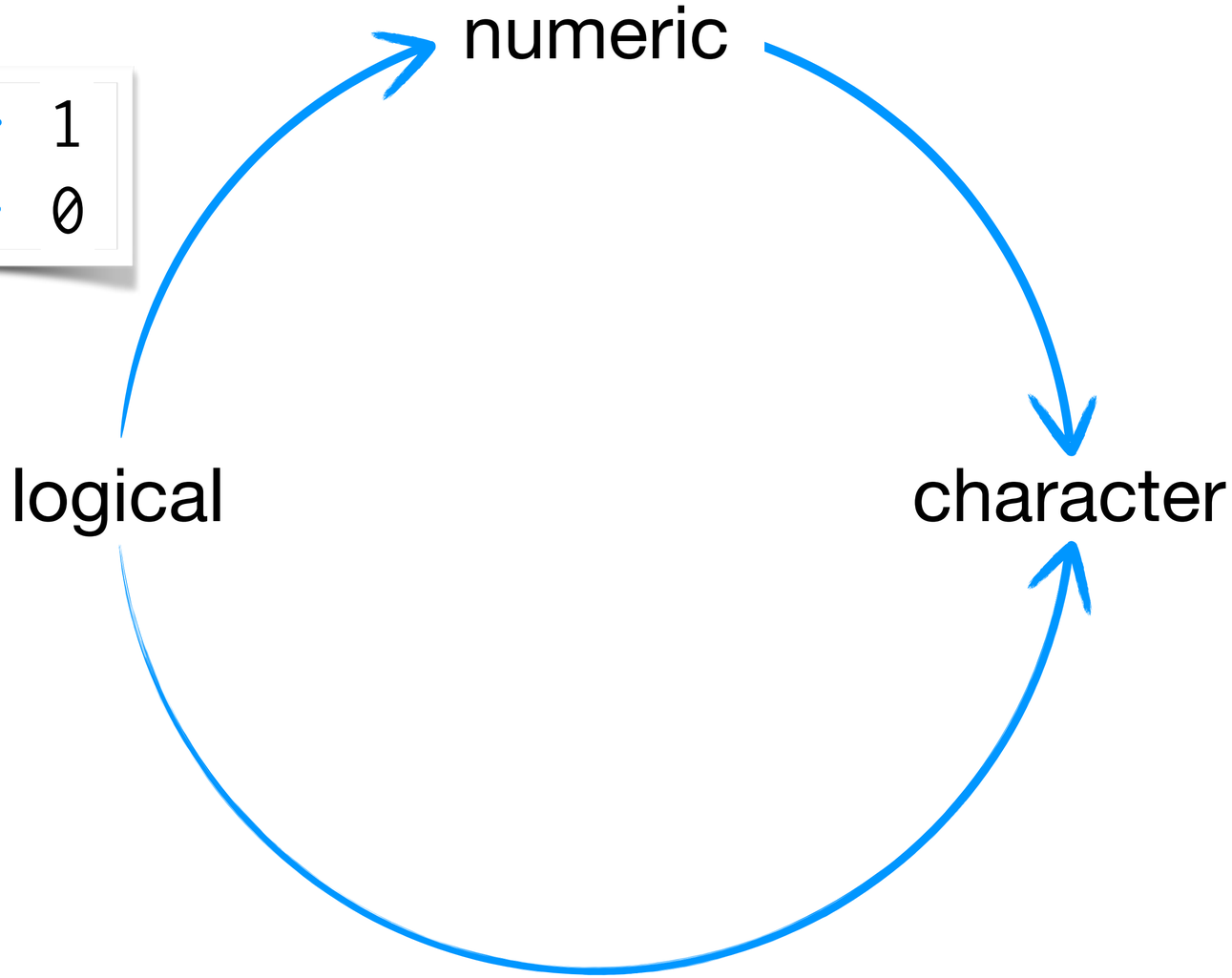
character

coercion



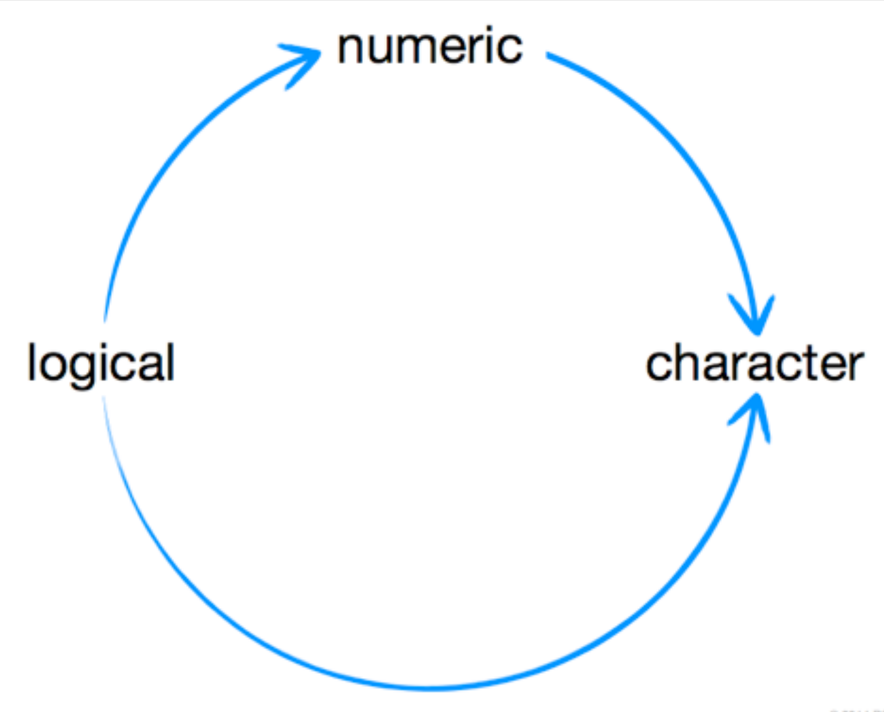
coercion

```
TRUE  → 1
FALSE → 0
```



Quiz

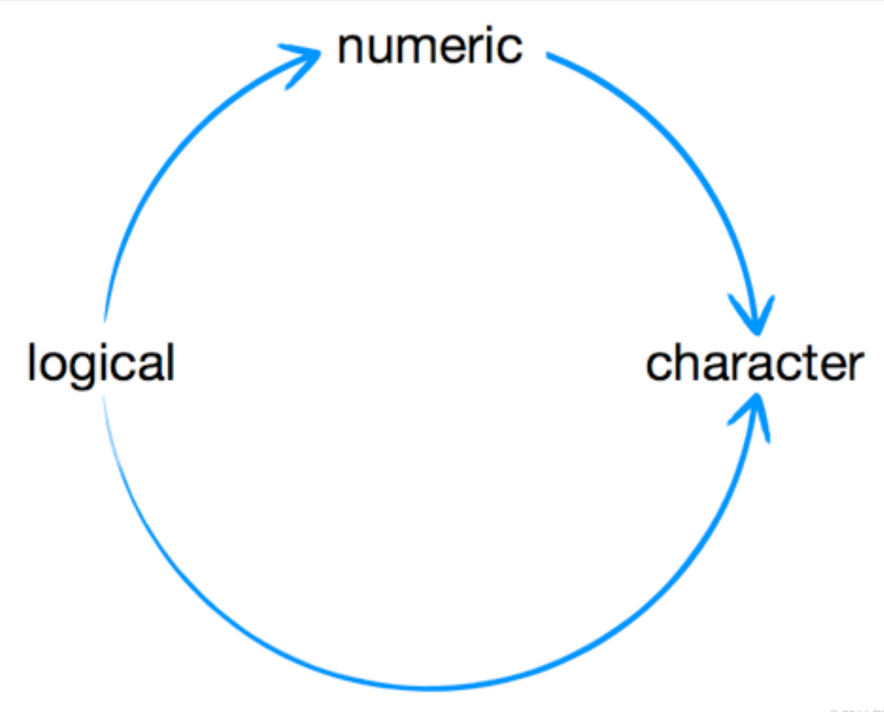
What type of data will result?




```
c(5, "two")

c(TRUE, "a")

c(1, "TRUE")

TRUE + 5
```

Quiz

What type of data will result?




```
c(5, "two")
```
character

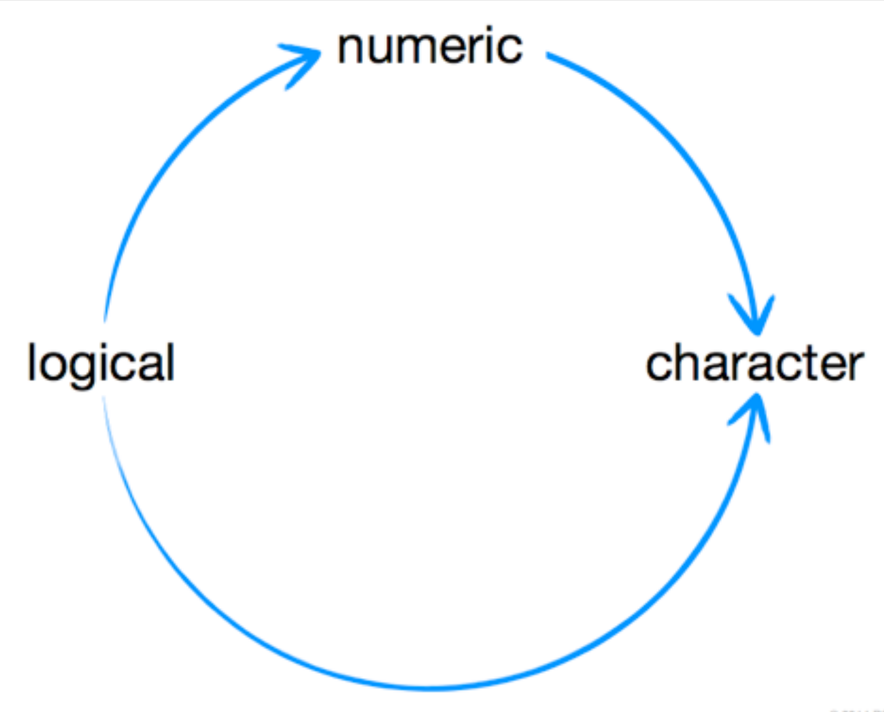
```
c(TRUE, "a")
```

```
c(1, "TRUE")
```

```
TRUE + 5
```

Quiz

What type of data will result?




```
c(5, "two")
```
character

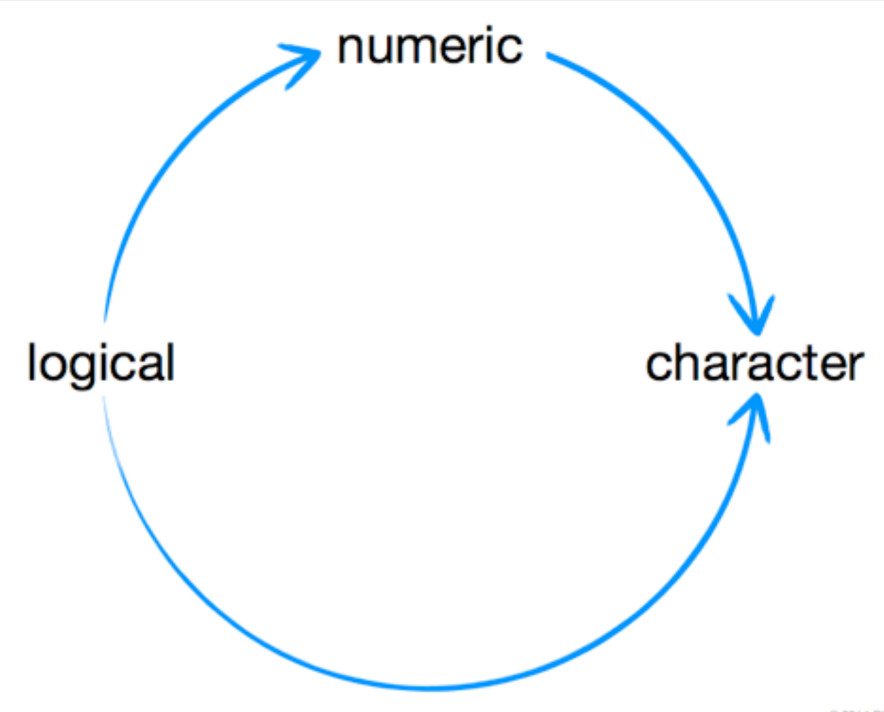
```
c(TRUE, "a")
```
character

```
c(1, "TRUE")
```

```
TRUE + 5
```

Quiz

What type of data will result?




```
c(5, "two")
```
character

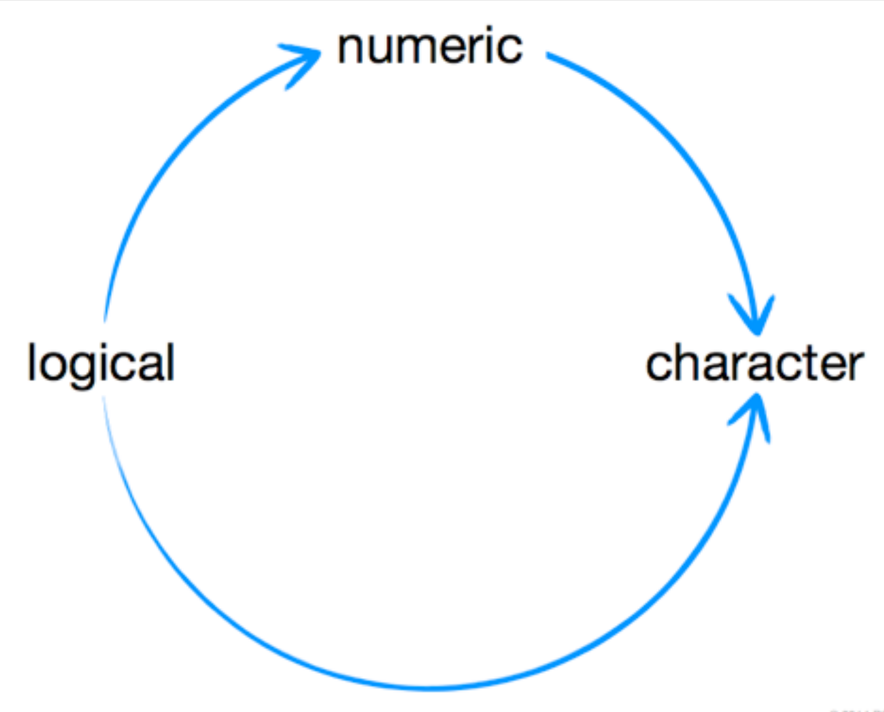
```
c(TRUE, "a")
```
character

```
c(1, "TRUE")
```
character

```
TRUE + 5
```

Quiz

What type of data will result?




```
c(5, "two")
```
character

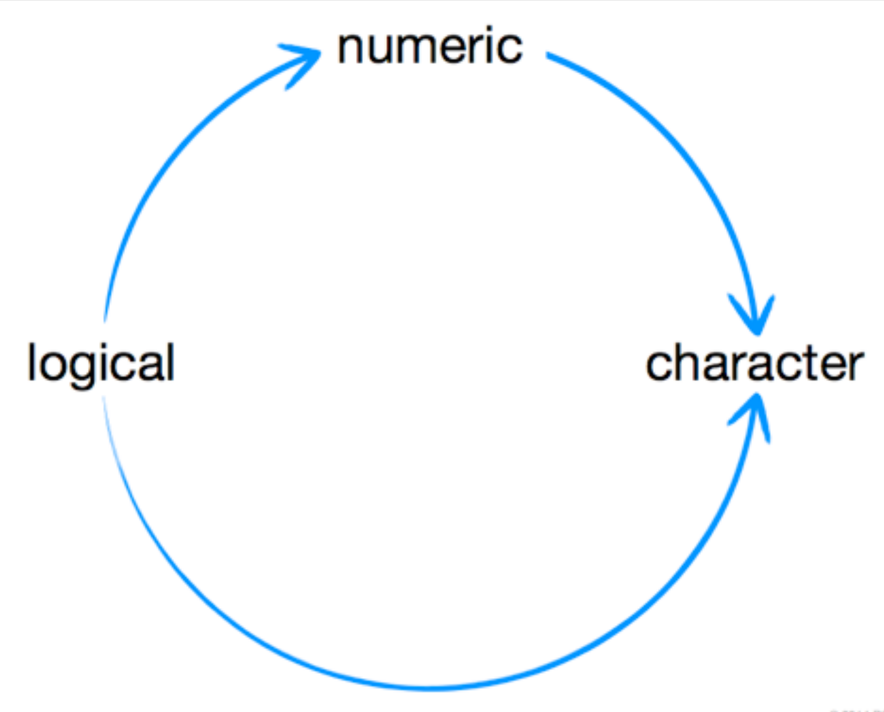
```
c(TRUE, "a")
```
character

```
c(1, "TRUE")
```
character

```
TRUE + 5
```

Quiz

What type of data will result?




```
c(5, "two")
```
character

```
c(TRUE, "a")
```
character

```
c(1, "TRUE")
```
character

```
TRUE + 5
```
numeric

manual coercion

function	coerces data to
as.numeric	numeric
as.character	character
as.logical	logical
as.factor	factor

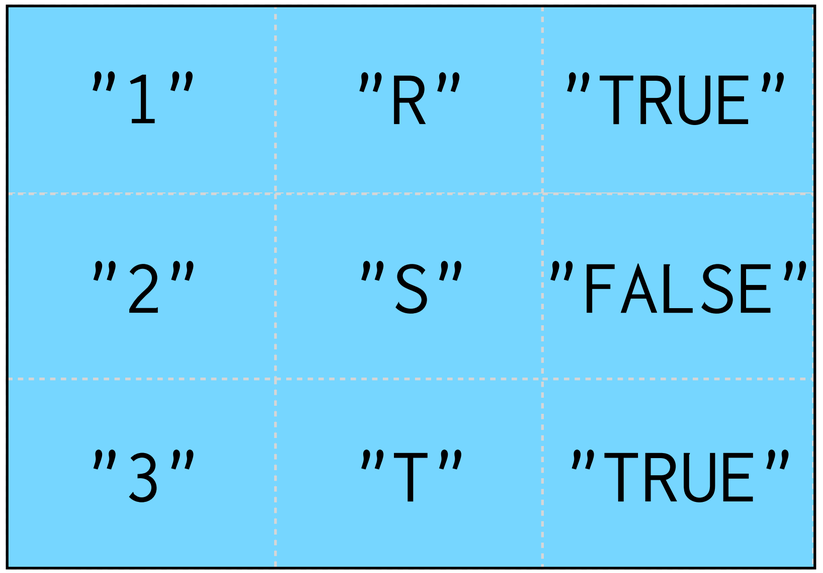
```
as.numeric("1")
as.character(TRUE)
```

Matrix

1	"R"	TRUE
2	"S"	FALSE
3	"T"	TRUE

?

Matrix



character

Matrix

"1"	"R"	"TRUE"
"2"	"S"	"FALSE"
"3"	"T"	"TRUE"

What if you want different data types in the same object?

lists and data frames

lists and *data frames* generalize vectors and matrices to allow multiple types of data

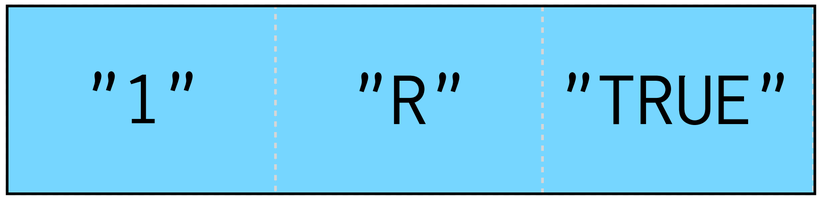
lists

A list is a one dimensional group of R objects.

Create lists with `list`

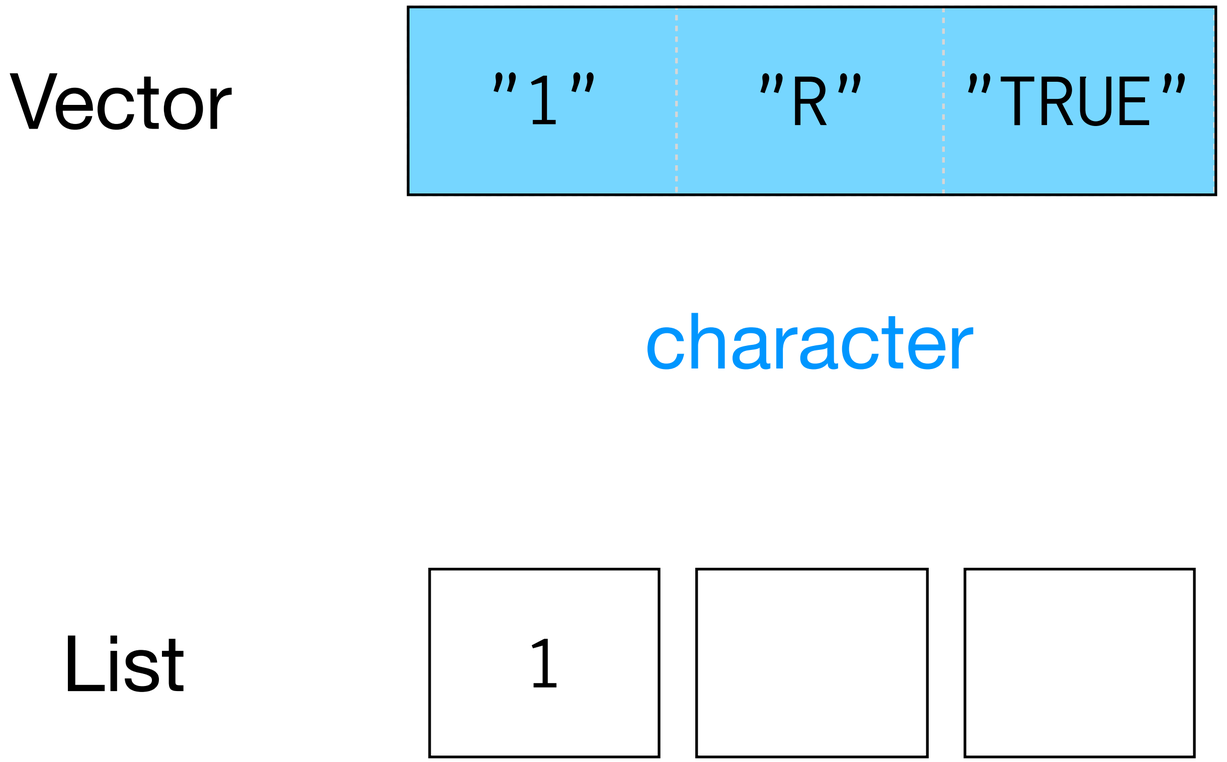
```
lst <- list(1, "R", TRUE)

class(lst)

# "list"
```

Vector

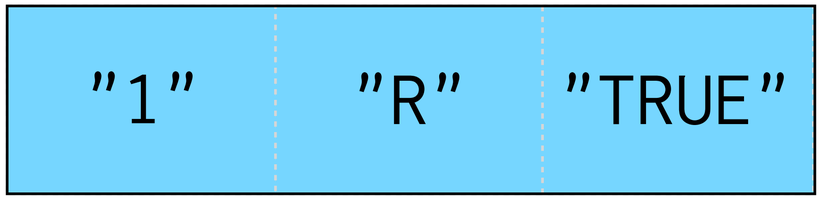


character

List



Vector

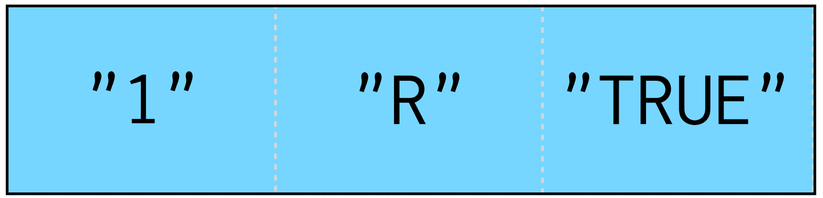


character

List

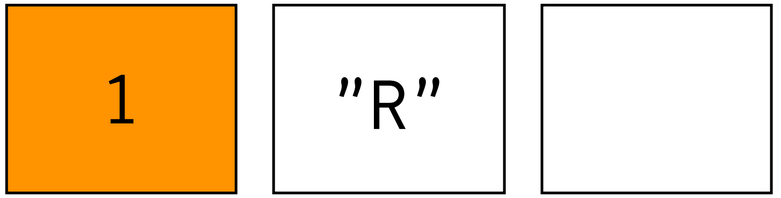
numeric

Vector



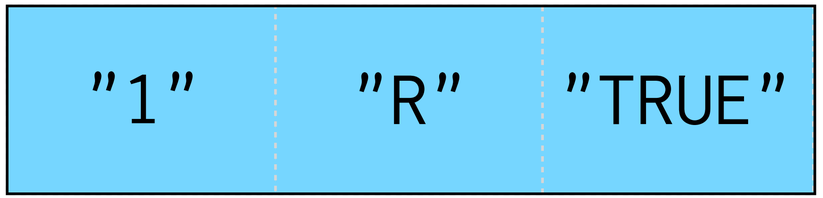
character

List



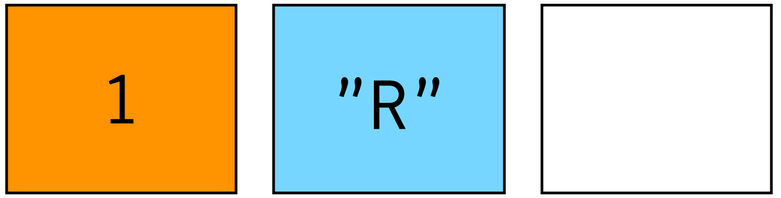
numeric

Vector



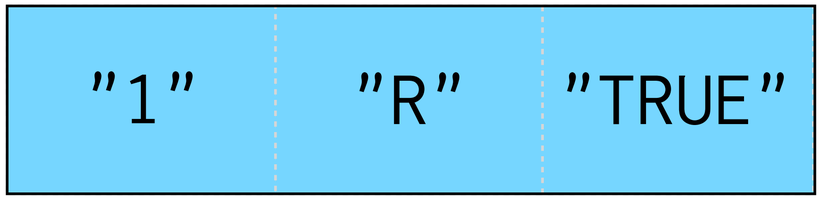
character

List



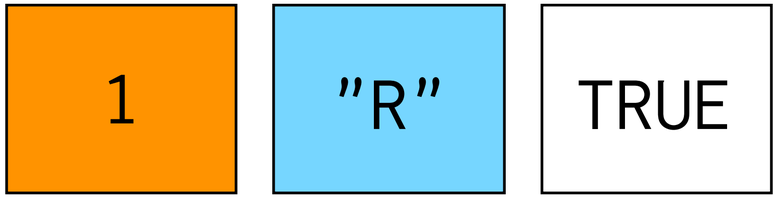
numeric character

Vector



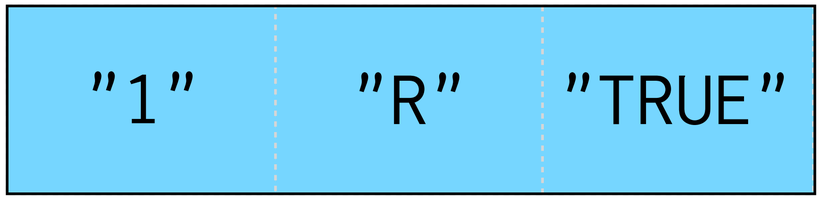
character

List



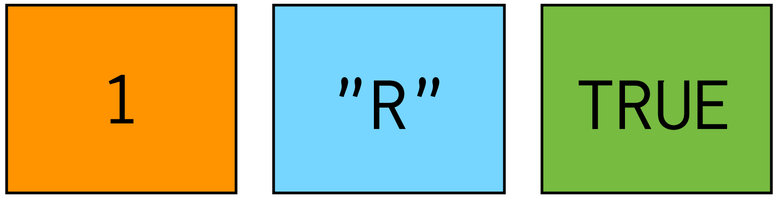
numeric character

Vector



character

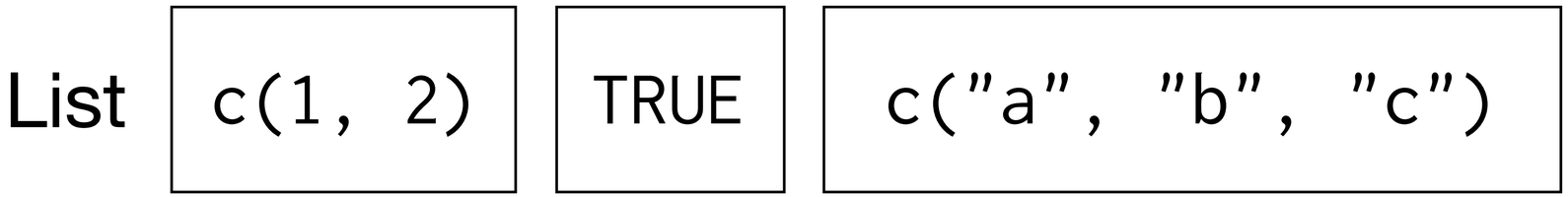
List



numeric character logical

The elements of a list can be anything. Even vectors or other lists.

```
list(c(1, 2), TRUE, c("a", "b", "c"))
```



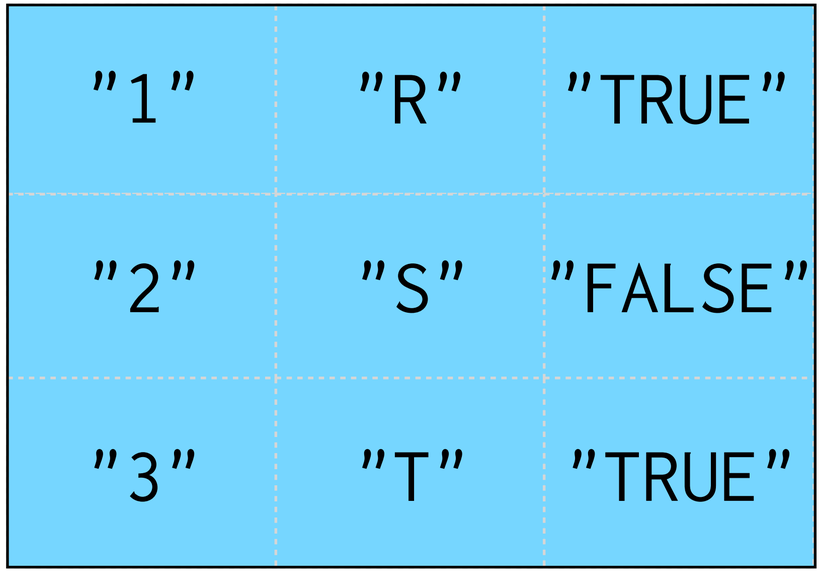
data frame

A data frame is a two dimensional group of R objects.

Each column in a data frame can be a different type

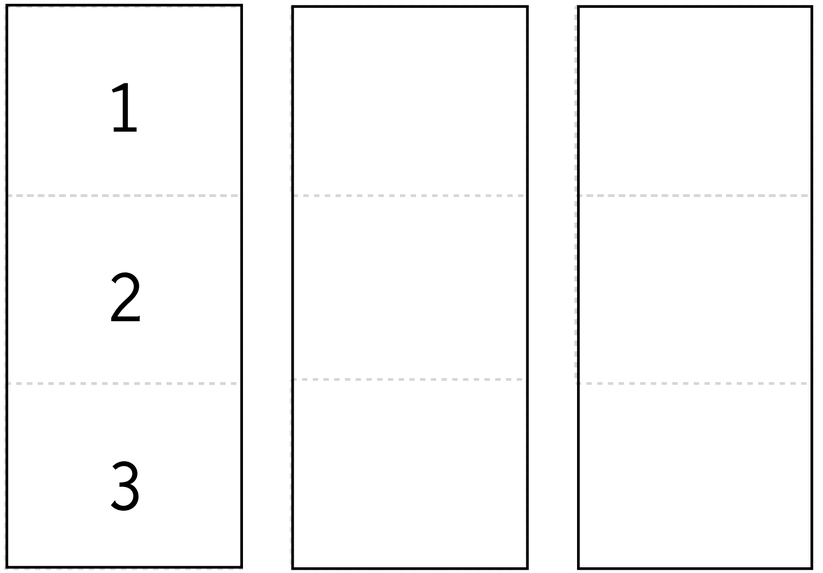
```
df <- data.frame(c(1, 2, 3),
 c("R","S","T"), c(TRUE, FALSE, TRUE))
class(df)
# "data.frame"
```

Matrix

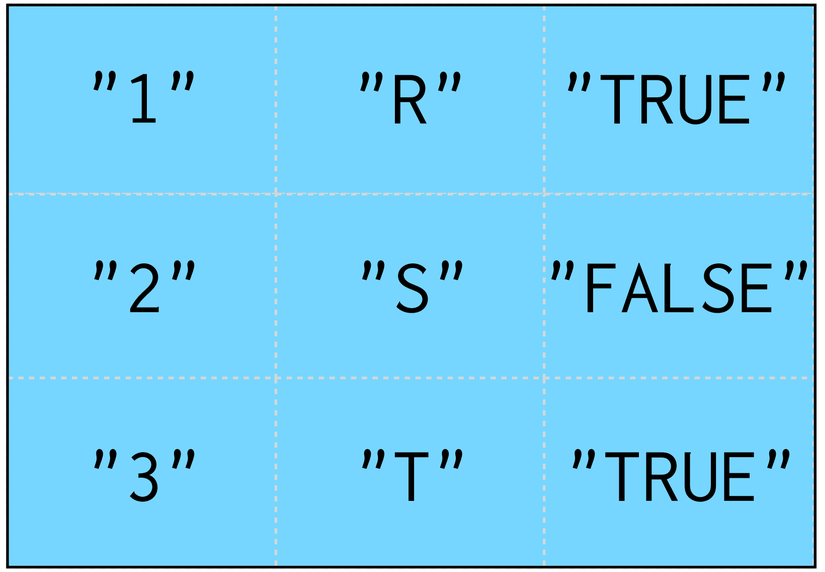


character

data frame

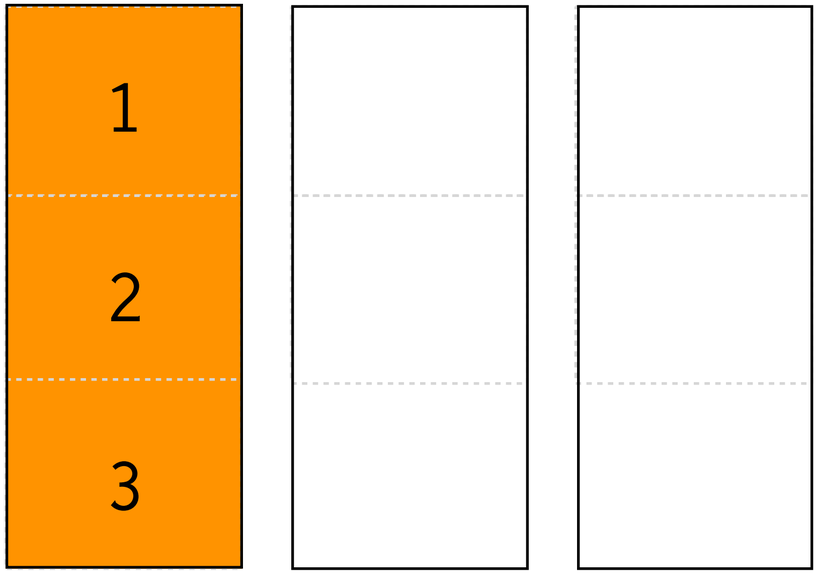


Matrix



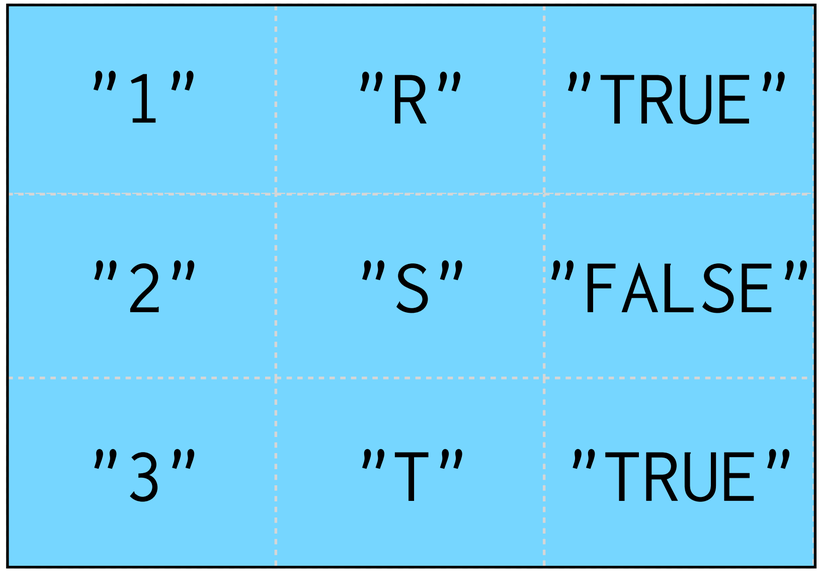
character

data frame



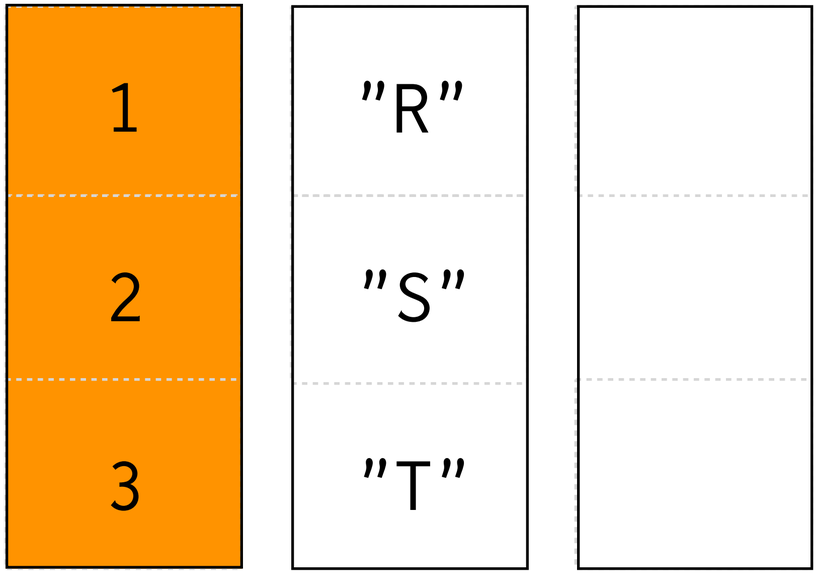
numeric

Matrix



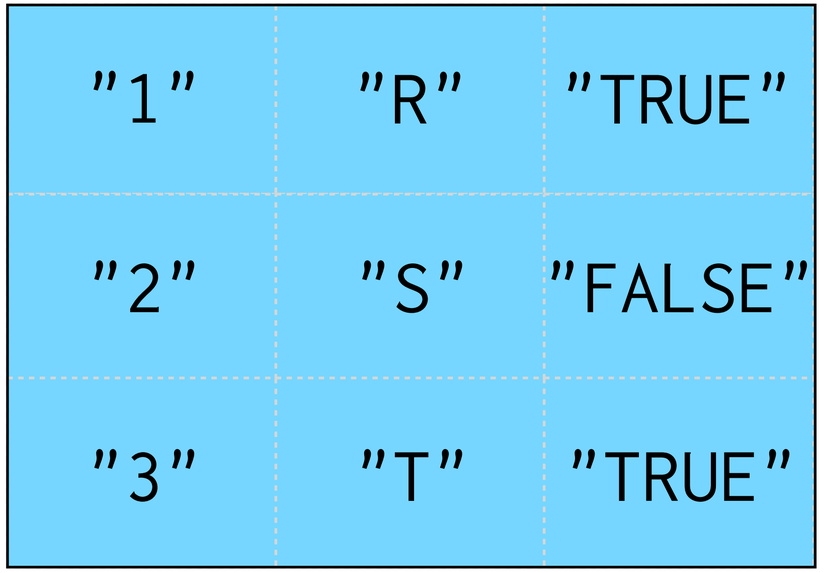
character

data frame

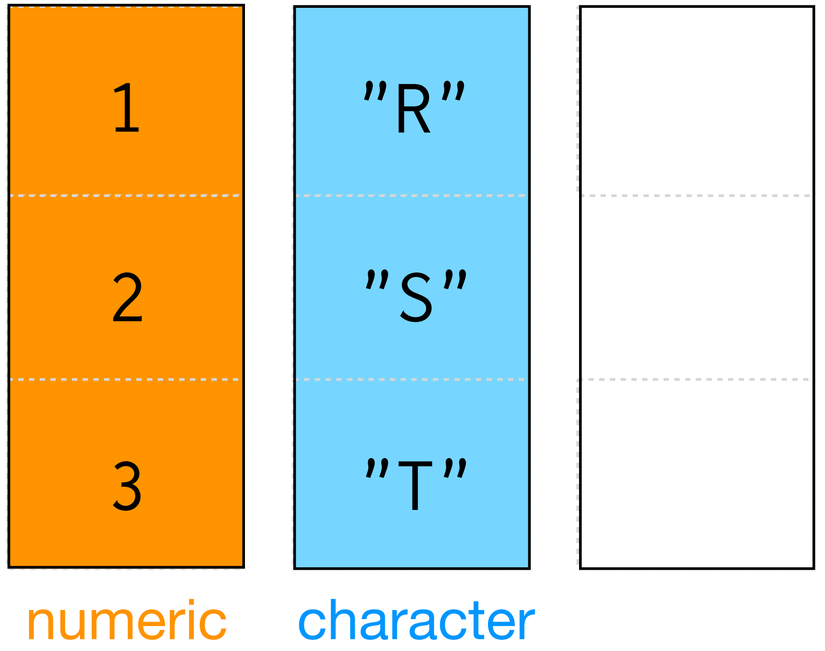


numeric

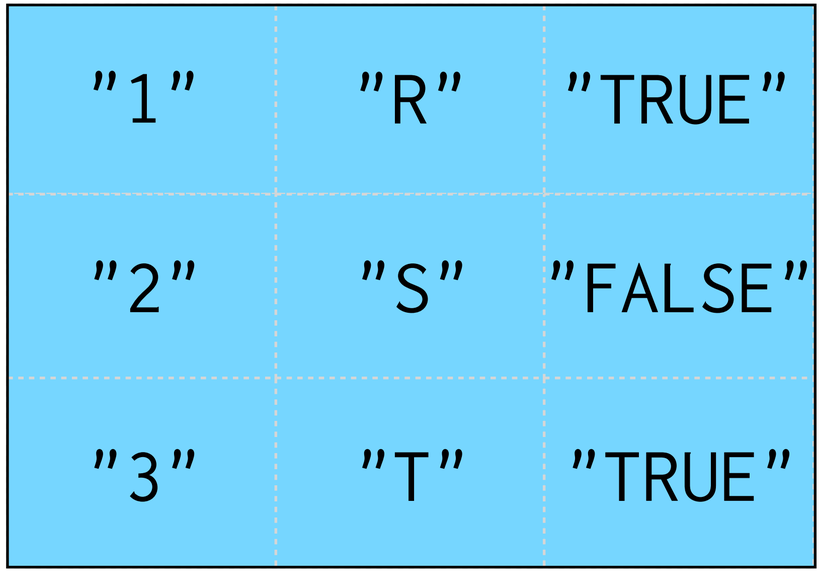
Matrix



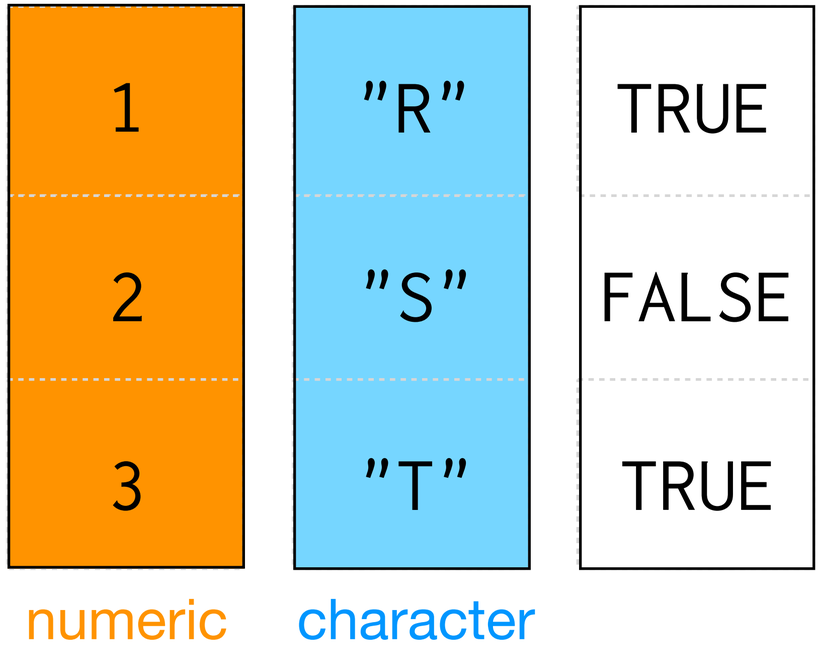
data frame



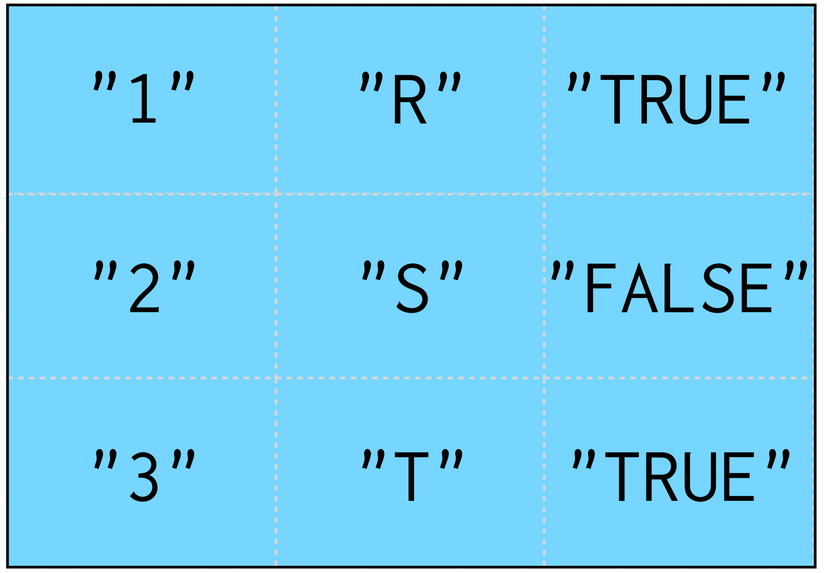
Matrix



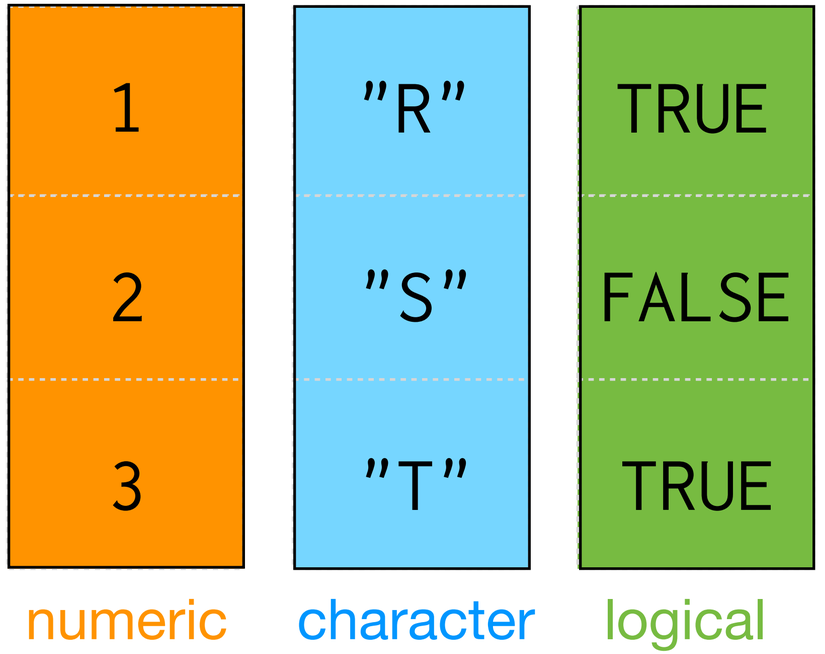
data frame



Matrix



data frame



names

You can name the elements of a vector, list, or data frame when you create them.

```
nvec <- c(one = 1, two = 2, three = 3)
```

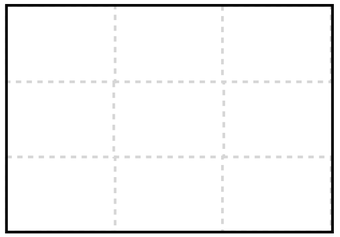
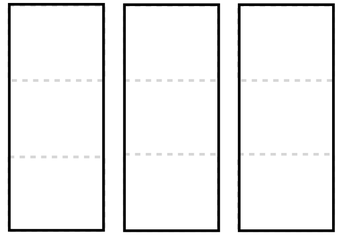
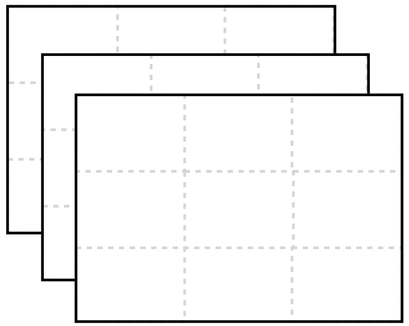
```
nvec
# one  two three
#   1    2     3
```

```
nlst <- list(one = 1, two = 2,
             many = c(3, 4, 5))

nlst
# $one
# [1] 1
#
# $two
# [1] 2
#
# $many
# [1] 3 4 5
```

```
ndf <- data.frame(numbers = c(1, 2, 3),
                  letters = c("R","S","T"),
                  logic = c(TRUE, FALSE, TRUE))

ndf
#   numbers letters logic
# 1       1       R  TRUE
# 2       2       S FALSE
# 3       3       T  TRUE
```

You can also see and set the names with `names`

```
names(ndf)
# [1] "numbers" "letters" "logic"

names(nvec)
# [1] "one"   "two"   "three"

names(nvec) <- c("uno", "dos", "tres")
nvec
# uno  dos tres
#   1    2    3
```

	single type	multiple types
1D	Vector 	List 
2D	Matrix 	Data frame 
nD	Array 	

helper functions for data structures

	create	change to	check	get names	get dimensions
vector	`c, vector`	`as.vector`	`is.vector`	`names`	`length`
matrix	`matrix`	`as.matrix`	`is.matrix`	`rownames, colnames`	`dim, nrow, ncol`
array	`array`	`as.array`	`is.array`	`dimnames`	`dim`
list	`list`	`as.list`	`is.list`	`names`	`length`
data frame	`data.frame`	`as.data.frame`	`is.data.frame`	`names`	`dim, nrow, ncol`

PELOTONIA®

TOGETHER UNSTOPPABLE