



Getting Started with R

David Keyes // R for the Rest of Us



Installation



Install R

The first thing you need to do is download the R software. Go to the [Comprehensive R Archive Network \(aka “CRAN”\) website](#) and download the software for your operating system (Windows, Mac, or Linux).



Working Directly in R



Working Directly in R

1. Enter: $2 + 2$

2. Hit return

3. View result



Your Turn

1. Download and Install R
2. Open R
3. Use any mathematical operators (+, -, /, and *) to create an expression and make sure it works as expected



RStudio

R: Engine



RStudio: Dashboard



Courtesy [Modern Dive](#)



RStudio

If you use RStudio, you'll have a graphical user interface, the ability to see all of your stored information, and much more.



Download RStudio

Download RStudio at the [RStudio website](#). Ignore the various versions listed there. All you need is the latest version of RStudio Desktop.



Tour of RStudio

The image shows the RStudio interface with four main panels:

- Scripts:** Shows an R script with code for installing packages, loading tidyverse and skimr, importing data, and examining foketucky data.
- Environment/History:** Shows the Global Environment tab with various objects listed.
- Console/Terminal:** Shows the R console output, including the R license, help information, and a prompt for exiting.
- Files/Plots/Packages/Help:** Shows the Files tab with a list of files and their details.



Your Turn

1. Download and Install RStudio
2. Open RStudio
3. Working in the console pane, use any mathematical operators (+, -, /, and *) to create an expression and make sure it works as expected



Projects



Projects

Projects allow you to keep a collection of files all together, including:

- R scripts
- RMarkdown files (more on those soon)
- Data files
- And much more!



Sample Project

The screenshot shows the RStudio interface with several panels:

- Code Editor:** Displays an R script with code for installing packages, loading tidyverse and skimr, importing data, and examining it.
- Environment:** Shows the global environment with objects like `Icon`, `images`, `intro-to-r-slides.html`, `intro-to-r-slides.Rmd`, `libs`, `setup.Rmd`, and `style.css`.
- History:** Displays the history of R commands run in the session.
- Console:** Shows the R startup message, license information, and a prompt for exiting.
- Terminal:** Shows the command `~/.Google/Drive/Work/R for the Rest of Us/courses/intro-to-r-course/` and the platform information.
- Plots:** Not visible in the screenshot.
- Packages:** Not visible in the screenshot.
- Help:** Not visible in the screenshot.
- Viewer:** Not visible in the screenshot.

Large text overlays are present in the bottom right corner:

- Scripts** (overlaid on the Environment/History panel)
- Console/Terminal** (overlaid on the Console/Terminal panel)
- Files/Plots/** (overlaid on the Environment/History panel)
- Packages/Help** (overlaid on the Environment/History panel)



How to Create a Project

1. File → New Project
2. Quit RStudio, Double-click .Rproj file to reopen project



Your Turn

1. Create a new project (doesn't matter if it's in a new or existing directory)
2. Quit RStudio, double-click the .Rproj file and reopen your project



Download Course Project





Your Turn

Enter the following into the RStudio console pane:

```
install.packages("usethis")
library(usethis)
use_course("https://bit.ly/getting-started-with-r")
```



Files in R



File Types

There are **two main file types** that you'll work with:

R scripts (.R)

Text is assumed to be executable R code unless you comment it (more on this soon)

```
# This is a comment  
data <- read_csv("data.csv")
```

RMarkdown files (.Rmd)

Text is assumed to be text unless you put it in a code chunk (more on this soon)



R Scripts

Create new script file: File → New File → R Script



How to Run Code

Run the code: control + enter on Windows, command + enter on Mac keystrokes or use Run button



Comments

Do them for others, and for your future self.

```
# This is a comment  
head(data, n = 5)
```



Packages



Packages

Packages add functionality that is not present in base R.

They're where much of the power of R is found.

R: A new phone



R Packages: Apps you can download





Packages We'll Use



tidyverse

The [tidyverse](#) is a collection of packages.

We'll use [readr](#) to import data.



Packages We'll Use

skimr

[skimr](#) provides easy summary statistics.





Install Packages

The syntax to install packages is as follows.

```
install.packages("tidyverse")
install.packages("skimr")
```

The package name must be in quotes.

Packages should be installed **once per computer** (i.e. once you've installed a package, you don't need to do it again on the same computer).



Load Packages

To load packages, use the following syntax:

```
library(tidyverse)  
library(skimr)
```

Package names don't need to be quoted here (though they can be).

Packages should be loaded **once per session** (i.e. every time you start working in R, you need to load any packages you want to use).



Your Turn

1. Open the project you downloaded before (it should be on your desktop)
2. Open the exercises.R file
3. Install the `tidyverse` and `skimr` packages using the `install.packages()` function
4. Load the `tidyverse` and `skimr` packages using the `library()` function



Import Data



Import Data

Let's read data from a CSV file.

```
faketucky <- read_csv("data/faketucky.csv")
```

We now have a data frame/tibble called faketucky that we can work with in R.



R is Case Sensitive

R is **case sensitive** so choose one of the following for all objects and **be consistent.**

Option

snake_case

camelCase

periods.in.names

Example

student_data

studentData

student.data



Directories

If the data file is in the working directory, you only need to specify its name.

```
faketucky <- read_csv("faketucky.csv")
```

If the data file is not in the working directory, you need to specify full path name.

When specifying the path name use the forward slash ("/") not backslash ("\").

```
faketucky <- read_csv("data/faketucky.csv")
```



Where Does our Data Live?

Data we have imported is available in the environment/history pane.



Your Turn

1. Open the exercises.R file
2. Import the faketucky data into a data frame called faketucky.
3. Make sure you see faketucky in your environment/history pane.



Objects and Functions



Objects and Functions

To understand computations in R, two slogans are helpful:

Everything that exists is an **object**, and

Everything that happens is a **function** call.

John Chambers, quoted in [Hadley Wickham's Advanced R](#).



```
grants_data <- read_csv("data/grants-data.csv")
```

everything that
exists is an

OBJECT

everything that
happens is a

FUNCTION



Assignment Operator

```
grants_data <- read_csv("data/grants-data.csv")
```

ASSIGNMENT OPERATOR

We assign the result of the `read_csv()` **function** to the `faketucky` **object**



IK Occhiali - Italy

SP4130 C2 48□20-140

Examine Our Data



Examine Our Data

There are many ways to look at our data. We'll talk about a few.



faketucky

If you type the name of your data frame (i.e. faketucky), R will output the following:

```
faketucky
```

```
## # A tibble: 57,855 × 12
##   student_id first_high_school_attended school_district male race_ethnicity
##       <dbl> <chr>                      <chr>          <dbl> <chr>
## 1       1622 Jackson                    Jackson        1 Multiple/Native ...
## 2       1877 Jackson                    Jackson        1 White
## 3       1941 Jackson                    Jackson        1 White
## 4       3442 Jackson                    Jackson        0 White
## 5       4623 Jackson                    Jackson        1 White
## 6       4913 Jackson                    Jackson        1 White
## 7       5754 Jackson                    Jackson        1 White
## 8       6293 Jackson                    Jackson        0 White
## 9       7010 Jackson                    Jackson        1 White
## 10      8343 Jackson                    Jackson        1 White
## # ... with 57,845 more rows, and 7 more variables: first_name <chr>, last_name <chr>, birth_date <date>, gender <chr>, grade_level <dbl>, final_gpa <dbl>, and 1 more variable: final_gpa <dbl>
```



head()

head() shows us the first X rows.

```
head(faketucky, 5)
```

```
## # A tibble: 5 × 12
##   student_id first_high_school_attended school_district  male race_ethnicity
##       <dbl> <chr>                      <chr>           <dbl> <chr>
## 1      1622 Jackson                    Jackson          1 Multiple/Native A...
## 2      1877 Jackson                    Jackson          1 White
## 3      1941 Jackson                    Jackson          1 White
## 4      3442 Jackson                    Jackson          0 White
## 5      4623 Jackson                    Jackson          1 White
## # ... with 7 more variables: free_and_reduced_lunch <dbl>,
## #   percent_absent <dbl>,
## #   gpa <dbl>, act_reading_score <dbl>, act_math_score <dbl>,
## #   received_high_school_diploma <dbl>, enrolled_in_college <dbl>
```



head()

```
head(faketucky, 5)
```

faketucky and the number 5 here are **arguments**.



tail()

tail shows us the last X rows.

```
tail(faketucky, 5)
```

```
## # A tibble: 5 × 12
##   student_id first_high_school_attended school_district  male race_ethnicity
##       <dbl> <chr>                      <chr>          <dbl> <chr>
## 1     89364 Lookout Point               Lookout Point    1 Hispanic
## 2     94764 Lookout Point               Lookout Point    0 White
## 3     97538 Lookout Point               Lookout Point    1 White
## 4    101342 Lookout Point               Lookout Point    1 White
## 5    104903 Lookout Point               Lookout Point    1 Hispanic
## # ... with 7 more variables: free_and_reduced_lunch <dbl>,
## #   percent_absent <dbl>,
## #   gpa <dbl>, act_reading_score <dbl>, act_math_score <dbl>,
## #   received_high_school_diploma <dbl>, enrolled_in_college <dbl>
```



View()

View (note capital V) opens the RStudio viewer (or click on a data frame in the environment pane).

```
View(faketucky)
```



skimr

The `skimr` package provides more detailed information about our data frame. It is also broken up by the type of variable.

```
# Install the package if necessary  
install.packages("skimr")
```

```
skim(faketucky)
```



Your Turn

1. Open the file exercises.R
2. Follow the instructions to use the `head()`, `tail()`, `View()`, and `skim()` functions to examine your data.



We've Got Issues!



We've Got Issues!

Several variables have max values of 999. This seems suspicious!

```
> skim(faketucky)
Skim summary statistics
n obs: 57855
n variables: 12

— Variable type:character —
      variable missing complete   n  min  max empty n_unique
first_high_school_attended     0    57855 57855   4   14    0       393
race_ethnicity          794    57061 57855   5   24    0       5
school_district        0    57855 57855   4   13    0       171

— Variable type:numeric —
      variable missing complete     n   mean     sd p0    p25    p50    p75   p100 hist
act_math_score            0    57855 57855 257.85 420.77  1    16     20     33    999 ━━━━
act_reading_score         0    57855 57855 258.78 420.65  2    16     22     34    999 ━━━━
enrolled_in_college       0    57855 57855 255.74 435.54  0     0     1     999    999 ━━━━
free_and_reduced_lunch    0    57855 57855 15.45 120.82  0     0     1     1    999 ━━━━
gpa                      0    57855 57855 40.22 189.95  0    2.02    2.7    3.36    999 ━━━━
male                     0    57855 57855  0.76 15.54  0     0     1     1    999 ━━━━
percent_absent           0    57855 57855 10.68 46.19  0    3.27    6.28   11.35  3153 ━━━━
received_high_school_diploma 0    57855 57855  0.74  0.44  0     0     1     1     1 ━━━━
student_id              0    57855 57855 55922.15 32332.71 1 27909.5 56070  83872.5 111990 ━━━━
```



We've Got Issues!

Several variables show up as numeric, but we know they're **not actually numeric**.

```
> skim(faketucky)
Skim summary statistics
n obs: 57855
n variables: 12

— Variable type:character —
      variable missing complete     n  min  max empty n_unique
first_high_school_attended      0    57855 57855   4   14     0       393
race_ethnicity                 794   57061 57855   5   24     0       5
school_district                  0    57855 57855   4   13     0       171

— Variable type:numeric —
      variable missing complete     n    mean      sd   p0    p25    p50    p75   p100 hist
act_math_score                  0    57855 57855 257.85 420.77  1    16     20     33    999 ━━━━━━
act_reading_score                0    57855 57855 258.78 420.65  2    16     22     34    999 ━━━━━━
enrolled_in_college              0    57855 57855 255.74 435.54  0     0     1     999    999 ━━━━
free_and_reduced_lunch            0    57855 57855 15.45 120.82  0     0     1     1     999 ━━━━
gpa                             0    57855 57855 40.22 189.95  0    2.02    2.7    3.36    999 ━━━━
male                            0    57855 57855  0.76  15.54  0     0     1     1     999 ━━━━
percent_absent                   0    57855 57855 10.68  46.19  0    3.27    6.28   11.35  3153 ━━━━
received_high_school_diploma      0    57855 57855  0.74   0.44  0     0     1     1     1 ━━━━
student_id                       0    57855 57855 55922.15 32332.71 1 27909.5 56070  83872.5 111990 ━━━━━━
```



Let's Import Our Data Again

We need to do two things:

1. Tell `read_csv()` how to handle **missing data**
2. Make sure `read_csv()` assigns the correct **data type** to each variable



Missing Data

Here's how we imported our data the first time:

```
faketucky <- read_csv("data/faketucky.csv")
```

To tell R which data is missing, simply add an argument to the `read_csv` function as follows:

```
faketucky <- read_csv("data/faketucky.csv",  
                      na = "999")
```



Missing Data

If we skim our data again, we'll see that there are no longer 999 values.

```
skim(faketucky)
```



Data Types

When you run `read_csv` you'll see the following message, which tells you the data types that have been assigned to your data frame.

```
Parsed with column specification:  
cols(  
  student_id = col_double(),  
  first_high_school_attended = col_character(),  
  school_district = col_character(),  
  male = col_double(),  
  race_ethnicity = col_character(),  
  free_and_reduced_lunch = col_double(),  
  percent_absent = col_double(),  
  gpa = col_double(),  
  act_reading_score = col_double(),  
  act_math_score = col_double(),  
  received_high_school_diploma = col_double(),  
  enrolled_in_college = col_double()  
)
```



Data Types

- **Double/Numeric** (e.g. 2.5)
- **Character** (e.g. "Male")

There are [many other data types in R](#) that we won't discuss today.



Data Types

```
faketucky <- read_csv("data/faketucky.csv",
                      na = "999",
                      col_types = list(enrolled_in_college = "c",
                                      free_and_reduced_lunch = "c",
                                      male = "c",
                                      received_high_school_diploma = "c"))
```



Your Turn

1. Open the file `exercises.R` and change the code so that you correctly import the `faketucky` data frame, telling `read_csv` which data is missing and explicitly defining column types where necessary.
2. After you make changes to how you import your data, rerun the code in the Examine Data section to make sure everything worked!



Getting Help



?function

Use the ? to get help about anything you're confused about

```
?read_csv
```



Tidyverse Website

Tidyverse



Tidyverse packages

Installation and use

- Install all the packages in the tidyverse by running `install.packages("tidyverse")`.
- Run `library(tidyverse)` to load the core tidyverse and make it available in your current R session.

Learn more about the tidyverse package at <http://tidyverse.tidyverse.org>.



Package Vignettes

Using Skimr

Elin Waring

2019-01-13

`skimr` is designed to provide summary statistics about variables. It is opinionated in its defaults, but easy to modify.

In base R, the most similar functions are `summary()` for vectors and data frames and `fivenum()` for numeric vectors:

```
summary(iris)

##   Sepal.Length   Sepal.Width    Petal.Length    Petal.Width
## Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
## 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
## Median :5.800   Median :3.000   Median :4.350   Median :1.300
```



Twitter

12 Sam Way and 12 others Retweeted

Gina Reynolds @EvaMaeRey · Feb 11

my `#ggplot2` flipbook project is online! 😎😎😎 Incrementally walks through plotting code (`#MakeoverMonday`, soon `#TidyTuesday` plots). Using `#xaringan` with reveal function; thanks, `@statsgen` `@grrrck`. `#rstats`.

evamaerey.github.io/ggplot_flipboo...

```
ggplot(data = dfmt +  
       aes(x = start.of.month) +  
       geom = "Percentage change from previous period") +  
       scale_y_continuous(breaks = c(-10, -5, 0, 5, 10)) +  
       geom_line() +  
       geom_ribbon(ymin = 0, ymax = "grey") +  
       ggtitle("Percentage change from previous period")
```

0:07 | 12.9K views

18 / 28

23

12 208

...

729

✉

...

Show this thread

R for the Rest of Us



R for Data Science Community

R4DS Online Learning Community



[HOME](#) [GET INVOLVED](#) [ABOUT](#)

**R FOR DATA SCIENCE
ONLINE LEARNING
COMMUNITY**



R for the Rest of Us



Google



E. Kale Edmiston PhD

@EKaleEdmiston

Follow



A friend/colleague who is an excellent programmer offhandedly told me the other day that coding is 90% googling error messages & 10% writing code. Until this point, I thought that all the time I spent googling error messages meant I was bad at coding. What a perspective change!

8:12 AM - 4 Jan 2019

151 Retweets 1,069 Likes

