Reading Data To/From R with readr

Outline

The readr package

Data file types

The read_*() functions

The write_*() functions

Other relevant tip and tricks

Data Import : : CHEAT SHEET

R's **tidyverse** is built around **tidy data** stored in **tibbles**, which are enhanced data frames.

The front side of this sheet shows how to read text files into R with **readr**.

The reverse side shows how to create tibbles with **tibble** and to layout tidy data with **tidyr**.

OTHER TYPES OF DATA

Try one of the following packages to import other types of files

- **haven** - SPSS, Stata, and SAS files
- **readxl** - excel files (.xls and .xlsx)
- **DBI** - databases
- **jsonlite** - json
- **xml2** - XML
- **httr** - Web APIs
- **rvest** - HTML (Web Scraping)

Save Data

Save **x**, an R object, to **path**, a file path, as:

Comma delimited file
write_csv(x, path, na = "NA", append = FALSE, col_names = !append)

File with arbitrary delimiter
write_delim(x, path, delim = " ", na = "NA", append = FALSE, col_names = !append)

CSV for excel
write_excel_csv(x, path, na = "NA", append = FALSE, col_names = !append)

String to file
write_file(x, path, append = FALSE)

String vector to file, one element per line
write_lines(x,path, na = "NA", append = FALSE)

Object to RDS file
write_rds(x, path, compress = c("none", "gz", "bz2", "xz"), ...)

Tab delimited files
write_tsv(x, path, na = "NA", append = FALSE, col_names = !append)

Read Tabular Data - These functions share the common arguments:

```
read_*(file, col_names = TRUE, col_types = NULL, locale = default_locale(), na = c("", "NA"),
  quoted_na = TRUE, comment = "", trim_ws = TRUE, skip = 0, n_max = Inf, guess_max = min(1000,
  n_max), progress = interactive())
```

```
a,b,c
1,2,3
4,5,NA
```
→

A	B	C
1	2	3
4	5	NA

Comma Delimited Files
read_csv("file.csv")
To make file.csv run:
write_file(x = "a,b,c\n1,2,3\n4,5,NA", path = "file.csv")

```
a;b;c
1;2;3
4;5;NA
```
→

A	B	C
1	2	3
4	5	NA

Semi-colon Delimited Files
read_csv2("file2.csv")
write_file(x = "a;b;c\n1;2;3\n4;5;NA", path = "file2.csv")

```
a|b|c
1|2|3
4|5|NA
```
→

A	B	C
1	2	3
4	5	NA

Files with Any Delimiter
read_delim("file.txt", delim = "|")
write_file(x = "a|b|c\n1|2|3\n4|5|NA", path = "file.txt")

```
a b c
1 2 3
4 5 NA
```
→

A	B	C
1	2	3
4	5	NA

Fixed Width Files
read_fwf("file.fwf", col_positions = c(1, 3, 5))
write_file(x = "a b c\n1 2 3\n4 5 NA", path = "file.fwf")

Tab Delimited Files
read_tsv("file.tsv") Also **read_table().**
write_file(x = "a\tb\tc\n1\t2\t3\n4\t5\tNA", path = "file.tsv")

USEFUL ARGUMENTS

```
a,b,c
1,2,3
4,5,NA
```

Example file
write_file("a,b,c\n1,2,3\n4,5,NA","file.csv")
f <- "file.csv"

1	2	3
4	5	NA

Skip lines
read_csv(f, **skip = 1**)

A	B	C
1	2	3
4	5	NA

No header
read_csv(f, **col_names = FALSE**)

A	B	C
1	2	3

Read in a subset
read_csv(f, **n_max = 1**)

x	y	z
A	B	C
1	2	3
4	5	NA

Provide header
read_csv(f, **col_names = c("x", "y", "z")**)

A	B	C
NA	2	3
4	5	NA

Missing Values
read_csv(f, **na = c("1", ".")**)

Read Non-Tabular Data

Read a file into a single string
read_file(file, locale = default_locale())

Read each line into its own string
read_lines(file, skip = 0, n_max = -1L, na = character(), locale = default_locale(), progress = interactive())

Read Apache style log files
read_log(file, col_names = FALSE, col_types = NULL, skip = 0, n_max = -1, progress = interactive())

Read a file into a raw vector
read_file_raw(file)

Read each line into a raw vector
read_lines_raw(file, skip = 0, n_max = -1L, progress = interactive())

Data types

readr functions guess the types of each column and convert types when appropriate (but will NOT convert strings to factors automatically).

A message shows the type of each column in the result.

```
## Parsed with column specification:
## cols(
##   age = col_integer(),
##   sex = col_character(),
##   earn = col_double()
## )
```

age is an integer

sex is a character

earn is a double (numeric)

1. Use **problems()** to diagnose problems.
   x <- read_csv("file.csv"); problems(x)

2. Use a col_ function to guide parsing.
   - **col_guess()** - the default
   - **col_character()**
   - **col_double()**, **col_euro_double()**
   - **col_datetime(**format = ""**)** Also **col_date(**format = ""**)**, **col_time(**format = ""**)**
   - **col_factor(**levels, ordered = FALSE**)**
   - **col_integer()**
   - **col_logical()**
   - **col_number()**, **col_numeric()**
   - **col_skip()**

   x <- read_csv("file.csv", col_types = cols(
   A = col_double(),
   B = col_logical(),
   C = col_factor()))

3. Else, read in as character vectors then parse with a parse_ function.
   - **parse_guess()**
   - **parse_character()**
   - **parse_datetime()** Also **parse_date()** and **parse_time()**
   - **parse_double()**
   - **parse_factor()**
   - **parse_integer()**
   - **parse_logical()**
   - **parse_number()**

   x$A <- parse_number(x$A)

readr

The *readr* package is the tidyverse package for reading data in and out of R

readr parses a flat file into a tibble

Outside of the tidyverse, there are a couple of other options for reading data (base, data.table)

The base functions can be up to 10x slower

The data.table function is 1.5 - 2x faster than readr functions but require specifying all columns.

Data File Types

Data can come in files that are wide ranging in format

Data may come in:

- Comma separated values (csv)
- Tab separated values (tsv)
- Fixed-width files (fwf)
- Web-log files (log)
- Etc…

read_*()

The family of read_*() files read in data from a file on your computer, a web address, or even data!

```
read_csv(file, col_names = TRUE, col_types = NULL,
  locale = default_locale(), na = c("", "NA"), quoted_na = TRUE,
  quote = "\"", comment = "", trim_ws = TRUE, skip = 0,
  n_max = Inf, guess_max = min(1000, n_max),
  progress = show_progress(), skip_empty_rows = TRUE)
```

`file`: path to file on computer or website, or even data

Lots of other defaulted arguments that allow flexibility!

Examples

File on computer if not in working directory:

```
read_csv("path/to/file/file.csv")
```

File from a website:

```
read_csv("https://github.com/tidyverse/
readr/raw/master/inst/extdata/
mtcars.csv")
```

Actual data:

```
read_csv("x,y\n1,2\n3,4")
```

read_*()

The family of write_*() files will write a flat file from the data frame / tibble you supply it with

```
write_csv(x, path, na = "NA", append = FALSE,
col_names = !append, quote_escape = "double")
```

`x`: the dataset you want to write to a flat file

`path`: where you want R to write the file to

Other defaulted arguments that allow flexibility!