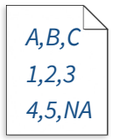
Data Import : : **CHEAT SHEET**

Read Tabular Data with readr

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

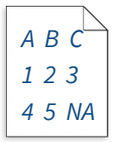
A	B	C
1	2	3
4	5	NA

read_delim("file.txt", delim = "|"**)** Read files with any delimiter. If no delimiter is specified, it will automatically guess.
To make file.txt, run:
write_file(x = "A|B|C\n1|2|3\n4|5|NA", file = "file.txt")



A	B	C
1	2	3
4	5	NA

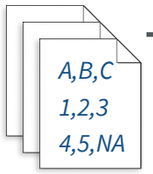
read_csv("file.csv"**)** Read a comma delimited file.
Also **read_csv2(**"file2.csv") to read a semi-colon delimited file.
write_file(x = "A,B,C\n1,2,3\n4,5,NA", file = "file.csv")



A	B	C
1	2	3
4	5	NA

read_tsv("file.tsv"**)** Read a tab delimited file. Also **read_table()**.
write_file(x = "A\tB\tC\n1\t2\t3\n4\t5\tNA", file = "file.tsv")

read_fwf("file.fwf", col_positions = c(1, 3, 5)**)** Read a fixed width file.
write_file(x = "A\tB\tC\n1\t2\t3\n4\t5\tNA", file = "file.tsv")



A	B	C
1	2	3
4	5	NA

read_csv(file = c("file1.csv", "file2.csv", "file3.csv")**)** Read multiple files by passing file a vector of file paths.

USEFUL READ ARGUMENTS

write_file("A,B,C\n1,2,3\n4,5,NA","file.csv")
f <- "file.csv"

A	B	C
1	2	3
4	5	NA

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

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

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

1	2	3
4	5	NA

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

A	B	C
1	2	3

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

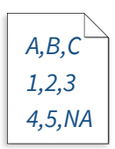
A	B	C
NA	2	3
4	5	NA

Read values as missing
read_csv(f, **na = c("1", ".")**)

Save Data with readr

write_*(x, file, na = "NA", append = FALSE, col_names = !append, quote_escape = "double", eol = "\n"**)**

A	B	C
1	2	3
4	5	NA



write_csv(x, file**)** Write a comma delimited file.

write_csv2(x, file**)** Write a semi-colon delimited file.

write_delim(x, file, delim = " "**)** Write files with any delimiter.

write_tsv(x, file**)** Write a tab delimited file.

Often one of the first steps of a project is to import outside data into R. Data is often stored in tabular formats, like csv files or spreadsheets.

The front page of this sheet shows how to import and save text files into R using **readr**.

The back page shows how to import spreadsheet data from Excel files using **readxl** or Google Sheets using **googlesheets4**.

OTHER TYPES OF DATA

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

- **haven** - SPSS, Stata, and SAS files
- **DBI** - databases
- **jsonlite** - json
- **xml2** - XML
- **httr** - Web APIs
- **rvest** - HTML (Web Scraping)
- **readr::read_lines()** - text data

Column Specification with readr

Column specifications define what data type each column of a file will be imported as. By default readr will generate a column spec when a file is read.

```
## 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)

COLUMN TYPES

Each column type has a function and corresponding string abbreviation.

- **col_logical()** - "l"
- **col_integer()** - "i"
- **col_double()** - "d"
- **col_number()** - "n"
- **col_character()** - "c"
- **col_factor(**levels, ordered = FALSE**)** - "f"
- **col_datetime(**format = ""**)** - "T"
- **col_date(**format = ""**)** - "D"
- **col_time(**format = ""**)** - "t"
- **col_skip()** - "-", "_"
- **col_guess()** - "?"

See readr.tidyverse.org/articles/readr for more information on parsing and debugging.

DEFINE COLUMN SPECIFICATION

If the default column specification isn't accurate, you can define it manually using the **col_type** argument.

Guess all columns
To "guess", **read_*()** looks at the first 1000 rows of data to guess what type a column is. Increase with the **guess_max** argument.

```
read_csv(path, col_types = NULL,
         guess_max = Inf)
```

Set a default type

```
read_csv(file,
         col_type = cols(.default = col_double()))
```

Use column type or string abbreviation

```
read_csv(file,
         col_type = cols(
                      x = col_double(),
                      y ="l",
                      z = "_"))
```

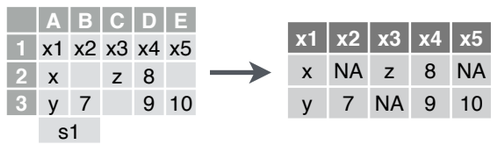
Use a single string of abbreviations

```
# col types: skip, guess, integer, logical, character
read_csv(file,
         col_type = "_?ilc")
```

Import Spreadsheets

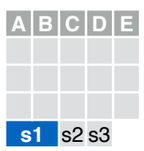
with readxl

READ EXCEL FILES

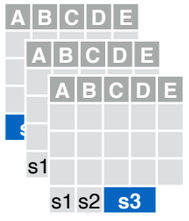


read_excel(path, sheet = NULL, range = NULL**)**
Read a .xls or .xlsx file based on the file extension. See front page for more read arguments. Also **read_xls()** and **read_xlsx()**.
read_excel("excel_file.xlsx")

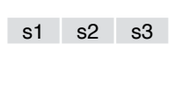
READ SHEETS



read_excel(path, **sheet = NULL)** Specify which sheet to read by position or name.
read_excel(path, sheet = 1)
read_excel(path, sheet = "s1")



To **read multiple sheets:**
1. From a file path, create a vector of sheet names.
2. Set the vector names to be the sheet names.
3. Use purrr::map_dfr() to read multiple files into one data frame



excel_sheets(path**)** Get a vector of sheet names.
excel_sheets("excel_file.xlsx")

```
path <- "your_file_path.xlsx"
path %>% excel_sheets() %>%
          set_names() %>%
          map_dfr(read_excel, path = path)
```

READXL COLUMN SPECIFICATION

Column specifications define what data type each column of a file will be imported as.

Use the **col_types** argument of **read_excel()** to set the column specification.

Guess all columns
To "guess", **read_excel()** looks at the first 1000 rows of data to guess what type a column is. Increase with the **guess_max** argument.
read_excel(path, col_types = NULL,
guess_max = Inf)

Set all columns to same type, e.g. character
read_excel(path, col_types = "text")

Set each column individually
read_excel(path,
col_types = c("text",
"guess",
"guess",
"numeric"))

COLUMN TYPES

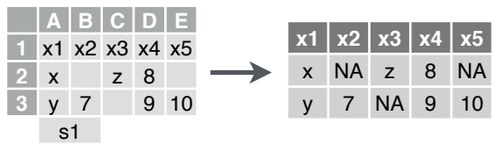
logical	numeric	text	date	list
TRUE	2	hello	1947-01-08	hello
FALSE	3.45	world	1956-10-21	1

- skip
- guess
- logical
- numeric
- text
- date
- list

Use **list** for columns that include multiple data types. See **tidyr** and **purrr** for list-column data.

with googlesheets4

READ SHEETS



read_sheet(ss, sheet = NULL, range = NULL**)**
Read a sheet from a URL, a Sheet ID, or a dribble from the googledrive package. See front page for more read arguments. Same as **range_read()**.

SHEETS METADATA

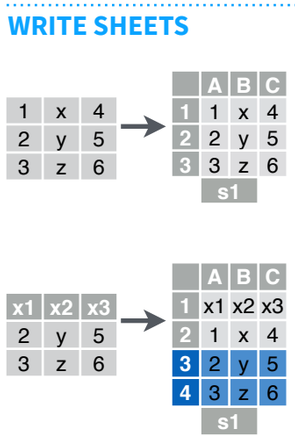
URLs are in the form:
https://docs.google.com/spreadsheets/d/
SPREADSHEET_ID/edit#gid=**SHEET_ID**

gs4_get(ss**)** Get spreadsheet meta data.

gs4_find(...**)** Get data on all spreadsheet files.

sheet_properties(ss**)** Get a tibble of properties for each worksheet. Also **sheet_names()**.

WRITE SHEETS



sheet_write(data, ss = NULL, sheet = NULL**)**
Write a data frame into a new or existing Sheet.

sheet_append(ss, data, sheet = 1**)** Add rows to the end of a worksheet.
Also **range_write()**, **range_flood()**, and **range_clear()**.

GOOGLESHEETS4 COLUMN SPECIFICATION

Column specifications define what data type each column of a file will be imported as.

Use the **col_types** argument of **read_sheet()/ range_read()** to set the column specification.

Guess all columns
To "guess", **read_sheet()/range_read()** looks at the first 1000 rows of data to guess what type a column is. Change with **guess_max**.
read_sheet(path, col_types = NULL,
guess_max = Inf)

Set all columns to same type, e.g. character
read_sheet(path, col_types = "c")

Set each column individually
col types: skip, guess, integer, logical, character
read_sheets(ss,
col_types = "_?ilc")

COLUMN TYPES

l	n	c	D	L
TRUE	2	hello	1947-01-08	hello
FALSE	3.45	world	1956-10-21	1

- skip - "_" or "-"
- guess - "?"
- logical - "l"
- integer - "i"
- double - "d"
- numeric - "n"
- date - "D"
- datetime - "T"
- character - "c"
- list-column - "L"
- cell - "C" Returns list of raw cell data.

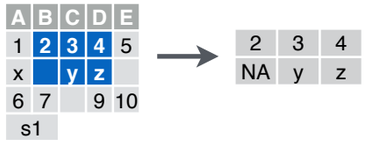
Use list for columns that include multiple data types. See **tidyr** and **purrr** for list-column data.

OTHER USEFUL EXCEL PACKAGES

For functions to write data to Excel files, see:
- **openxlsx** - **ycphs.github.io/openxlsx/**
- **writexl** - **docs.ropensci.org/writexl/**

For working with non-tabular Excel data, see:
- **tidyxl** - **nacnudus.github.io/tidyxl/**

CELL SPECIFICATION FOR READXL AND GOOGLESHEETS4



Use the **range** argument of **readxl::read_excel()** or **googlesheets4::read_sheet()** to read a subset of cells from a sheet.
read_excel(path, range = "Sheet1!D12:F15")
read_sheet(ss, range = "D12:F15")

Also use the cell specification functions **cell_limits, cell_rows, cell_cols,** and **anchored.**

FILE LEVEL OPERATIONS

googlesheets4 also offers ways to modify other aspects of Sheets (e.g. freeze rows, set column width, manage (work)sheets. Go to **googlesheets4.tidyverse.org** to read more.

For whole-file operations (e.g. renaming, sharing, placing within a folder), see the tidyverse package **googledrive** at **googledrive.tidyverse.org**.