

# Data import with the tidyverse : : CHEAT SHEET



## Read Tabular Data with readr

**read\_\***(file, col\_names = TRUE, col\_types = NULL, col\_select = NULL, id = NULL, locale, n\_max = Inf, skip = 0, na = c("", "NA"), guess\_max = min(1000, n\_max), show\_col\_types = TRUE) See ?**read\_delim**

A|B|C  
1|2|3  
4|5|NA



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("A|B|C\n1|2|3\n4|5|NA", file = "file.txt")

A,B,C  
1,2,3  
4,5,NA



A	B	C
1	2	3
4	5	NA

**read\_csv**("file.csv") Read a comma delimited file with period decimal marks.  
write\_file("A,B,C\n1,2,3\n4,5,NA", file = "file.csv")

A;B;C  
1,5;2;3  
4,5;5;NA



A	B	C
1.5	2	3
4.5	5	NA

**read\_csv2**("file2.csv") Read semicolon delimited files with comma decimal marks.  
write\_file("A;B;C\n1,5;2;3\n4,5;5;NA", file = "file2.csv")

A B C  
1 2 3  
4 5 NA



A	B	C
1	2	3
4	5	NA

**read\_tsv**("file.tsv") Read a tab delimited file. Also **read\_table()**.  
**read\_fwf**("file.tsv", fwf\_widths(c(2, 2, NA))) Read a fixed width file.  
write\_file("A\tB\tC\n1\t2\t3\n4\t5\tNA\n", file = "file.tsv")

### USEFUL READ ARGUMENTS

A	B	C
1	2	3
4	5	NA

#### No header

read\_csv("file.csv", col\_names = FALSE)

1	2	3
4	5	NA

#### Skip lines

read\_csv("file.csv", skip = 1)

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

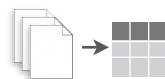
#### Provide header

read\_csv("file.csv",  
col\_names = c("x", "y", "z"))

A	B	C
1	2	3

#### Read a subset of lines

read\_csv("file.csv", n\_max = 1)



#### Read multiple files into a single table

read\_csv(c("f1.csv", "f2.csv", "f3.csv"),  
id = "origin\_file")

A	B	C
NA	2	3
4	5	NA

#### Read values as missing

read\_csv("file.csv", na = c("1"))

A;B;C  
1,5;2;3,0

#### Specify decimal marks

read\_delim("file2.csv", locale =  
locale(decimal\_mark = ";"))

## Save Data with readr

**write\_\***(x, file, na = "NA", append, col\_names, quote, escape, eol, num\_threads, progress)

A	B	C
1	2	3
4	5	NA



A,B,C  
1,2,3  
4,5,NA

**write\_delim**(x, file, delim = " ") Write files with any delimiter.

**write\_csv**(x, file) Write a comma delimited file.

**write\_csv2**(x, file) Write a semicolon delimited file.

**write\_tsv**(x, file) Write a tab delimited file.

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**.

## 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 and output a summary.

**spec(x)** Extract the full column specification for the given imported data frame.

```
spec(x)
# cols(
#   age = col_integer(),
#   sex = col_character(),
#   earn = col_double()
# )
```

age is an integer

earn is a double (numeric)

sex is a character

### 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()** - "?"

### 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

### USEFUL COLUMN ARGUMENTS

#### Hide col spec message

read\_\*(file, show\_col\_types = FALSE)

#### Select columns to import

Use names, position, or selection helpers.

read\_\*(file, col\_select = c(age, earn))

#### Guess column types

To guess a column type, read\_\*(file) looks at the first 1000 rows of data. Increase with **guess\_max**.  
read\_\*(file, guess\_max = Inf)

### DEFINE COLUMN SPECIFICATION

#### Set a default type

```
read_csv(
  file,
  col_type = list(default = col_double())
)
```

#### Use column type or string abbreviation

```
read_csv(
  file,
  col_type = list(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

	A	B	C	D	E
1	x1	x2	x3	x4	x5
2	x		z	8	
3	y	7		9	10

s1

x1	x2	x3	x4	x5
x	NA	z	8	NA
y	7	NA	9	10

**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

A	B	C	D	E

s1 s2 s3

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

**excel\_sheets(path)** Get a vector of sheet names.  
`excel_sheets("excel_file.xlsx")`

A	B	C	D	E

s1

A	B	C	D	E

s1 s2 s3

**To read multiple sheets:**

1. Get a vector of sheet names from the file path.
2. Set the vector names to be the sheet names.
3. Use `purrr::map_dfr()` to read multiple files into one data frame.

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

### OTHER USEFUL EXCEL PACKAGES

For functions to write data to Excel files, see:

- **openxlsx**
- **writexl**

For working with non-tabular Excel data, see:

- **tidyxl**



### 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 column types

To guess a column type, `read_excel()` looks at the first 1000 rows of data. Increase with the **guess\_max** argument.

```
read_excel(path, 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

	A	B	C	D	E
1	x1	x2	x3	x4	x5
2	x		z	8	
3	y	7		9	10

s1

x1	x2	x3	x4	x5
x	NA	z	8	NA
y	7	NA	9	10

**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

	A	B	C
1	x	4	
2	y	5	
3	z	6	

s1

A	B	C	D

s1

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

**gs4\_create(name, ..., sheets = NULL)** Create a new Sheet with a vector of names, a data frame, or a (named) list of data frames.

**sheet\_append(ss, data, sheet = 1)** Add rows to the end of a worksheet.

x1	x2	x3
2	y	5
3	z	6

s1

A	B	C
1	x1	x2
2	1	x
3	2	y
4	3	z

s1

### CELL SPECIFICATION FOR READXL AND GOOGLESHEETS4

A	B	C	D	E
1	1	2	3	4
2	x		y	z
3	6	7		9

s1

2	3	4
NA	y	z

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!B1:D2")
read_sheet(ss, range = "B1:D2")
```

Also use the range argument with cell specification functions **cell\_limits()**, **cell\_rows()**, **cell\_cols()**, and **anchored()**.



### 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 column types

To guess a column type `read_sheet()`/`range_read()` looks at the first 1000 rows of data. Increase with **guess\_max**.

```
read_sheet(path, 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.

### 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**.