

readxl (1)

INTRODUCTION TO IMPORTING DATA IN R



Filip Schouwenaars
Instructor, DataCamp


Microsoft Excel

- Common data analysis tool
- Many R packages to interact with Excel
- readxl - Hadley Wickham

Typical Structure Excel Data

- Different sheets with tabular data

Capital	Population
New York	16044000
Berlin	3433695
Madrid	3010492
Stockholm	1683713



	Population
	17800000
	3382169
Madrid	2938723
Stockholm	1942362

year_1990

year_2000

readxl

- `excel_sheets()`
 - list different sheets
- `read_excel()`
 - actually import data into R

```
install.packages("readxl")  
library(readxl)
```

excel_sheets()

```
dir()
```

```
"cities.xlsx" "the_rest_is_secret.txt"
```

```
excel_sheets("cities.xlsx")
```

```
"year_1990" "year_2000"
```

read_excel()

```
read_excel("cities.xlsx")
```

```
# A tibble: 4 × 2
  Capital Population
  <chr>         <dbl>
1 New York    16044000
2 Berlin      3433695
3 Madrid      3010492
4 Stockholm   1683713
```

```
read_excel("cities.xlsx", sheet = 2)
read_excel("cities.xlsx", sheet = "year_2000")
```

```
# A tibble: 4 × 2
  Capital Population
  <chr>         <dbl>
1 New York    17800000
2 Berlin      3382169
3 Madrid      2938723
4 Stockholm   1942362
```

readxl (2)

INTRODUCTION TO IMPORTING DATA IN R



Filip Schouwenaars
Instructor, DataCamp

read_excel()

```
read_excel(path, sheet = 1,  
           col_names = TRUE,  
           col_types = NULL,  
           skip = 0)
```

Capital	Population		
New York	16044000		
Berlin	3433695		
Madrid	3010492		
Stockholm	1683713		
year_1990			
year_2000			



read_excel() - col_names

```
read_excel(path, sheet = 1,  
           col_names = TRUE,  
           col_types = NULL,  
           skip = 0)
```

- col_names = FALSE: R assigns names itself
- col_names = character vector: manually specify

read_excel() - col_types

```
read_excel(path, sheet = 1,  
           col_names = TRUE,  
           col_types = NULL,  
           skip = 0)
```

```
read_excel("cities.xlsx", col_types = c("text", "text"))
```

```
# A tibble: 4 × 2  
  Capital Population  
  <chr>      <chr>  
1 New York  16044000  
2 Berlin   3433695  
3 Madrid   3010492  
4 Stockholm 1683713
```

read_excel() - col_types

```
read_excel(path, sheet = 1,  
           col_names = TRUE,  
           col_types = NULL,  
           skip = 0)`
```

```
read_excel("cities.xlsx",  
           col_types = c("text", "blank"))
```

```
# A tibble: 4 × 1  
  Capital  
  <chr>  
1 New York  
2 Berlin  
3 Madrid  
4 Stockholm
```

read_excel() - skip

```
read_excel(path, sheet = 1,  
           col_names = TRUE,  
           col_types = NULL,  
           skip = 0)
```

```
read_excel("cities.xlsx",  
           col_names = c("Capital", "Population"),  
           skip = 2)
```

```
# A tibble: 3 × 2  
  Capital Population  
  <chr>      <dbl>  
1 Berlin    3433695  
2 Madrid    3010492  
3 Stockholm 1683713
```

- n_max not (yet) available

Wrap-up

- `excel_sheets()`
- `read_excel()`
- Everything you need!
- Fast
- Same arguments as in `readr` package
- Consistency

gdata

INTRODUCTION TO IMPORTING DATA IN R



Filip Schouwenaars
Instructor, DataCamp

gdata

- Gregory Warnes
- Entire suite of tools for data manipulation
- Supercharges basic R
- `read.xls()`
- Support for XLS
- Support for XLSX with additional driver
- No `readxl::excel_sheets()` equivalent

gdata

XLS $\xrightarrow{\text{Perl}}$ CSV

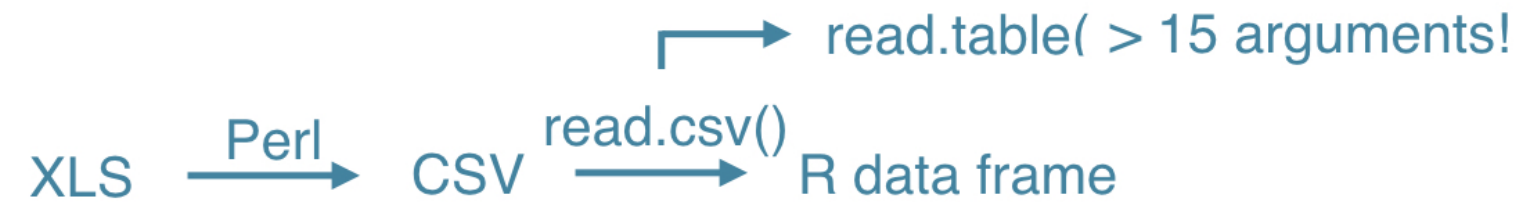
gdata

XLS $\xrightarrow{\text{Perl}}$ CSV $\xrightarrow{\text{read.csv()}}$ R data frame

gdata



gdata



- Elegant extension of utils package
- Easy if familiar with utils
- Extremely inefficient
- readxl < v1.x

cities.xls

Capital	Population		
New York	16044000		
Berlin	3433695		
Madrid	3010492		Population
Stockholm	1683713		17800000
			3382169
year_1990		Madrid	2938723
		Stockholm	1942362
		year_2000	



read.xls()

```
install.packages("gdata")  
library(gdata)
```

```
read.xls("cities.xls")
```

```
      Capital Population  
1 New York    16044000  
2   Berlin    3433695  
3   Madrid    3010492  
4 Stockholm   1683713
```

```
read.xls("cities.xls", sheet = "year_2000")
```

```
      Capital Population  
1 New York    17800000  
2   Berlin    3382169  
3   Madrid    2938723  
4 Stockholm   1942362
```

Reading sheets

INTRODUCTION TO IMPORTING DATA IN R



Filip Schouwenaars
Instructor, DataCamp



XLConnect

- Martin Studer
- Work with Excel through R
- Bridge between Excel and R
- XLS and XLSX
- Easy-to-use functionality

Installation

```
install.packages("XLConnect")
```

```
also installing the dependencies 'XLConnectJars', 'rJava'  
...
```

- Problems?
 - Install Oracle's Java Development Kit (JDK)
 - Google your error!

loadWorkbook()

```
library("XLConnect")  
book <- loadWorkbook("cities.xlsx")  
str(book)
```

```
Formal class 'workbook' [package "XLConnect"] with 2 slots  
  ..@ filename: chr "cities.xlsx"  
  ..@ jobj      : ...
```

getSheets()

```
getSheets(book)
```

```
"year_1990" "year_2000"
```

```
library(readxl)  
excel_sheets("cities.xlsx")
```

```
"year_1990" "year_2000"
```

readWorksheet()

```
readWorksheet(book, sheet = "year_2000")
```

	Capital	Population
1	New York	17800000
2	Berlin	3382169
3	Madrid	2938723
4	Stockholm	1942362

readWorksheet()

Capital	Population
New York	17800000
Berlin	3382169
Madrid	2938723
Stockholm	1942362

year_2000 col 2

row 3
row 4

```
readWorksheet(book, sheet = "year_2000",  
             startRow = 3,  
             endRow = 4,  
             startCol = 2,  
             header = FALSE)
```

```
Col1  
1 3382169  
2 2938723
```

Adapting sheets

INTRODUCTION TO IMPORTING DATA IN R



Filip Schouwenaars
Instructor, DataCamp

New data!

```
pop_2010 <- data.frame(Capital = c("New York", "Berlin", "Madrid", "Stockholm"),  
  Population = c(8191900, 3460725, 3273000, 1372565))
```

```
pop_2010
```

	Capital	Population
1	New York	8191900
2	Berlin	3460725
3	Madrid	3273000
4	Stockholm	1372565

createSheet()

```
pop_2010 <- ... # truncated
library(XLConnect)
book <- loadWorkbook("cities.xlsx")
```

Capital	Population
New York	16044000
Berlin	3382169
Madrid	2938723
Stockholm	1942362

Capital	Population
New York	17800000
Berlin	3382169
Madrid	2938723
Stockholm	1942362

year_1990

year_2000



createSheet()

```
pop_2010 <- ... # truncated
library(XLConnect)
book <- loadWorkbook("cities.xlsx")
createSheet(book, name = "year_2010")
```

The diagram shows a transformation of a single table into a wide table format. On the left, a single table with 5 rows and 2 columns is shown:

Capital	Population
New York	16044000
Berlin	3500000
Madrid	3200000
Stockholm	1200000

Below this table, the text "year_1990" is displayed. In the center, a second table with 5 rows and 2 columns is shown:

Capital	Population
New York	17800000
Berlin	3500000
Madrid	2800000
Stockholm	1200000

Below this table, the text "year_2000" is displayed. On the right, a wide table with 5 rows and 2 columns is shown:

Below this table, the text "year_2010" is displayed. An Excel icon is located in the top right corner of the diagram.

writeWorksheet()

```
pop_2010 <- ... # truncated
library(XLConnect)
book <- loadWorkbook("cities.xlsx")
createSheet(book, name = "year_2010")
writeWorksheet(book, pop_2010, sheet = "year_2010")
```

The diagram shows a transformation of a single table into a wide table format. On the left, a single table with 2 columns (Capital, Population) and 5 rows (New York, Berlin, Madrid, Stockholm, and an unlabeled row) is shown. This table is transformed into a wide table with 3 columns (Capital, Population, and an unlabeled column) and 5 rows. The transformation is labeled 'year_1990' and 'year_2000' at the bottom. An Excel icon is shown in the top right corner.

Capital	Population
New York	16044000
Berlin	3500000
Madrid	3000000
Stockholm	1000000

year_1990


Capital	Population	
New York	17800000	
Berlin	3500000	
Madrid	2800000	
Stockholm	1000000	

year_2000

year_2010

saveWorkbook()

```
pop_2010 <- ... # truncated
library(XLConnect)
book <- loadWorkbook("cities.xlsx")
createSheet(book, name = "year_2010")
writeWorksheet(book, pop_2010, sheet = "year_2010")
```



Capital	Population
New York	16044000
Berlin	3382169
Madrid	2938723
Stockholm	1942362

year_1990

Capital	Population
New York	17800000
Berlin	3382169
Madrid	2938723
Stockholm	1942362

year_2000

Capital	Population
New York	17800000
Berlin	3382169
Madrid	2938723
Stockholm	1942362


year_2010

saveWorkbook()

```
pop_2010 <- ... # truncated
library(XLConnect)
book <- loadWorkbook("cities.xlsx")
createSheet(book, name = "year_2010")
writeWorksheet(book, pop_2010, sheet = "year_2010")

saveWorkbook(book, file = "cities2.xlsx")
```

Capital	Population
New York	16044000
Berlin	3382169
Madrid	2938723
Stockholm	1942362
year_1990	
Capital	Population
New York	17800000
Berlin	3382169
Madrid	2938723
Stockholm	1942362
year_2000	
Capital	Population
New York	17800000
Berlin	3382169
Madrid	2938723
Stockholm	1942362
year_2010	

 cities2.xlsx

renameSheet()

```
renameSheet(book, "year_1990", "Y1990")
renameSheet(book, "year_2000", "Y2000")
renameSheet(book, "year_2010", "Y2010")
```

Capital	Population
New York	16044000
Berlin	3382169
Madrid	2938723
Stockholm	1942362


year_1990

Capital	Population
New York	17800000
Berlin	3382169
Madrid	2938723
Stockholm	1942362

year_2000

Capital	Population
New York	17800000
Berlin	3382169
Madrid	2938723
Stockholm	1942362


year_2010



renameSheet()

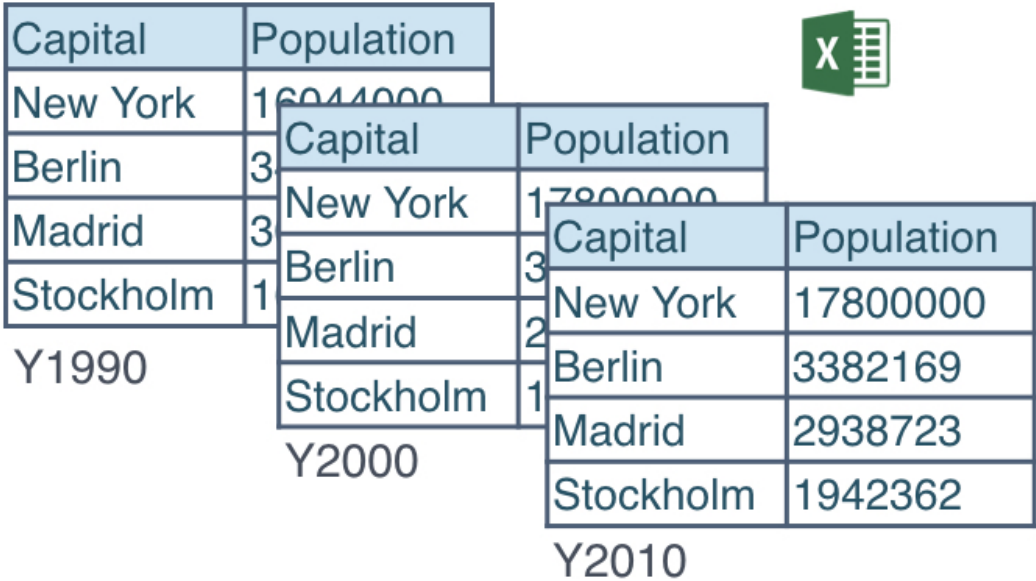
```
renameSheet(book, "year_1990", "Y1990")
renameSheet(book, "year_2000", "Y2000")
renameSheet(book, "year_2010", "Y2010")
saveWorkbook(book, file = "cities3.xlsx")
```

Capital	Population
New York	16044000
Berlin	3382169
Madrid	2938723
Stockholm	1942362
Y1990	
Capital	Population
New York	17800000
Berlin	3382169
Madrid	2938723
Stockholm	1942362
Y2000	
Capital	Population
New York	17800000
Berlin	3382169
Madrid	2938723
Stockholm	1942362
Y2010	

 cities3.xlsx

removeSheet()

```
removeSheet(book, sheet = "Y2010")
```



removeSheet()

```
removeSheet(book, sheet = "Y2010")  
saveWorkbook(book, file = "cities4.xlsx")
```

Capital	Population
New York	16044000
Berlin	3382169
Madrid	2938723
Stockholm	1942362
Y1990	
Capital	Population
New York	17800000
Berlin	3382169
Madrid	2938723
Stockholm	1942362
Y2000	



Wrap-up

- Basic operations
- Reproducibility is the key!
- More functionality
 - Styling cells
 - Working with formulas
 - Arranging cells
 - ...