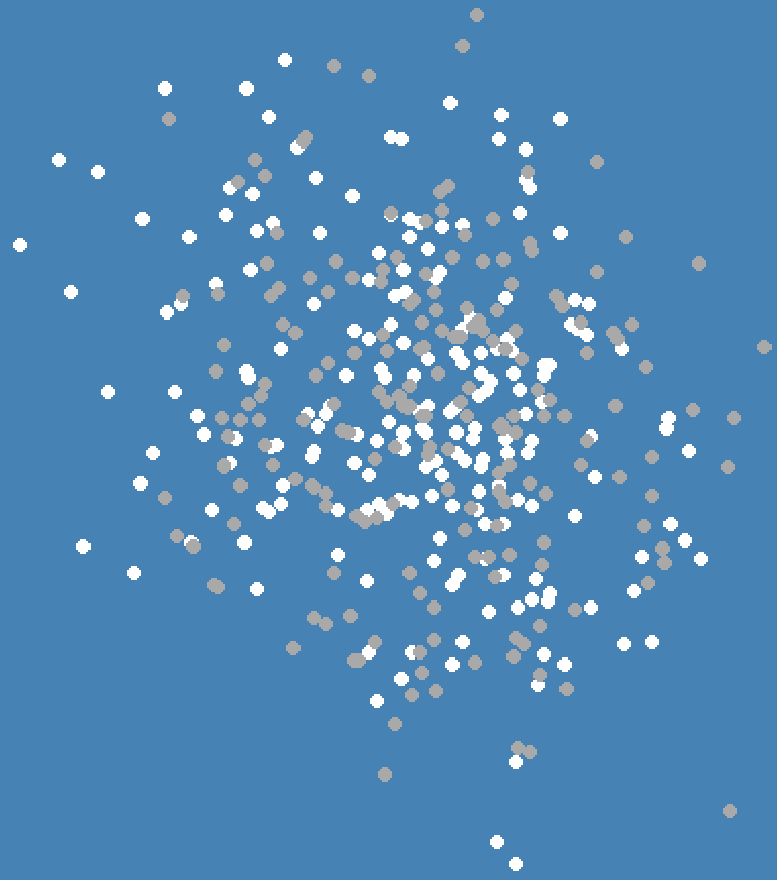
Introduction to R

1.3 R, RStudio and Basic Functionality

Functions, Packages, Tidyverse

Lion Behrens, M.Sc.



University of Mannheim
Chair of Social Data Science and Methodology
Chair of Quantitative Methods in the Social Sciences

R and RStudio

- R is a statistical programming language developed for statistical computing and graphics
    - non-commercial implementation of the S programming language
    - basic functionality is referred to as base R

- R is a statistical programming language developed for statistical computing and graphics
    - non-commercial implementation of the S programming language
    - basic functionality is referred to as base R

- R is a statistical programming language developed for statistical computing and graphics
  - non-commercial implementation of the S programming language
  - basic functionality is referred to as base R
- Packages extend the basic functionality
  - stored on the CRAN server and installed from within R
  - heavily ease ordinary data analysis tasks (data wrangling, modeling, plotting)
  - implement advanced and cutting edge statistical methods

- R is a statistical programming language developed for statistical computing and graphics
    - non-commercial implementation of the S programming language
    - basic functionality is referred to as base R
- Packages extend the basic functionality
    - stored on the CRAN server and installed from within R
    - heavily ease ordinary data analysis tasks (data wrangling, modeling, plotting)
    - implement advanced and cutting edge statistical methods
- Only comes with a basic Graphical User Interface (GUI)

- R is a statistical programming language developed for statistical computing and graphics
  - non-commercial implementation of the S programming language
  - basic functionality is referred to as base R
- Packages extend the basic functionality
  - stored on the CRAN server and installed from within R
  - heavily ease ordinary data analysis tasks (data wrangling, modeling, plotting)
  - implement advanced and cutting edge statistical methods
- Only comes with a basic Graphical User Interface (GUI)

- RStudio is an intuitive and flexible interface for R called IDE (Integretated Development Environment)

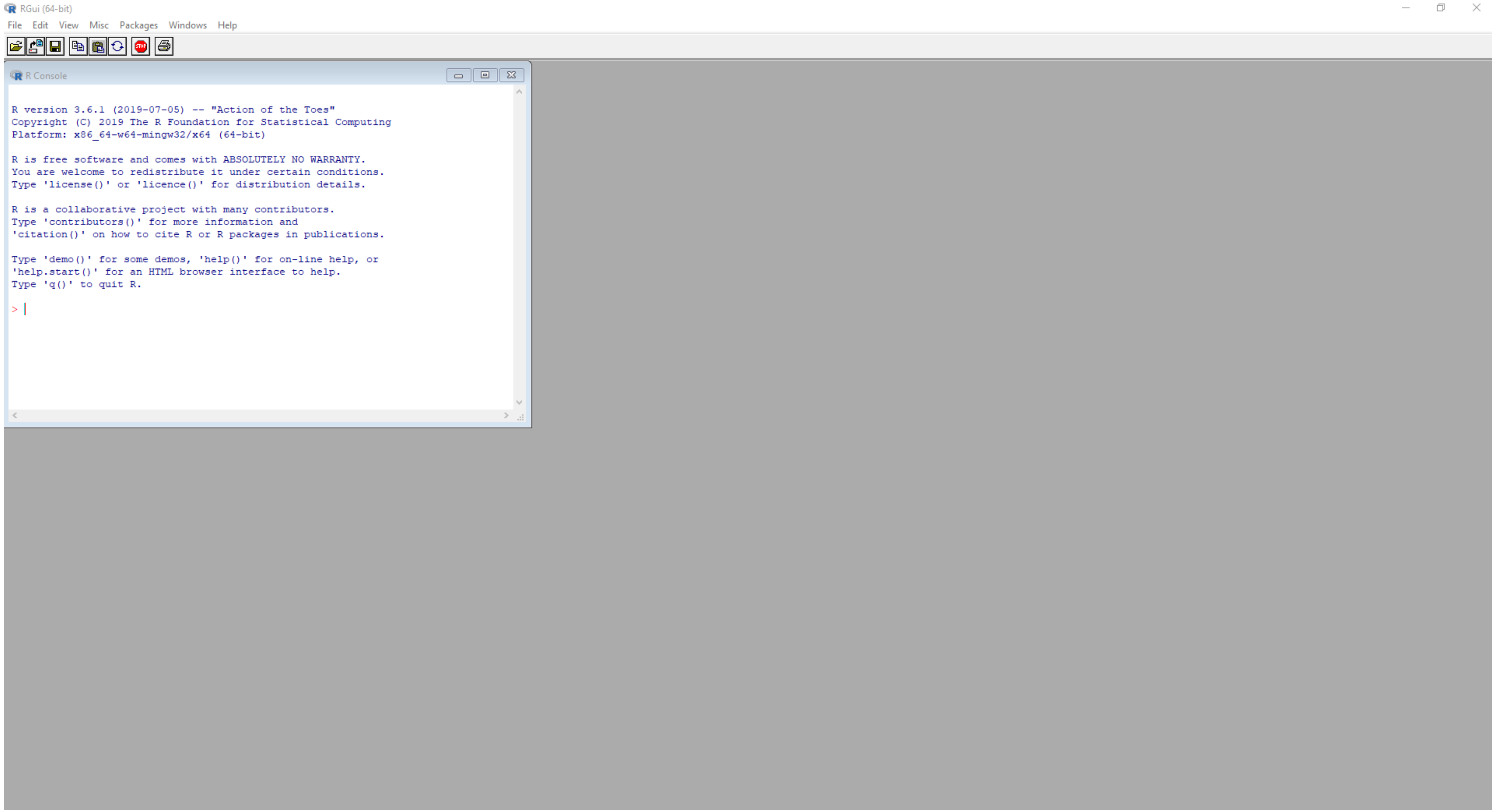
- R is a statistical programming language developed for statistical computing and graphics
  - non-commercial implementation of the S programming language
  - basic functionality is referred to as base R
- Packages extend the basic functionality
  - stored on the CRAN server and installed from within R
  - heavily ease ordinary data analysis tasks (data wrangling, modeling, plotting)
  - implement advanced and cutting edge statistical methods
- Only comes with a basic Graphical User Interface (GUI)

- RStudio is an intuitive and flexible interface for R called IDE (Integretated Development Environment)
- Makes R coding fun:
  - Nice overview of created objects, R scripts
  - Syntax highlighting
  - Spell checking
  - Auto-completion

- R is a statistical programming language developed for statistical computing and graphics
    - non-commercial implementation of the S programming language
    - basic functionality is referred to as base R
- Packages extend the basic functionality
    - stored on the CRAN server and installed from within R
    - heavily ease ordinary data analysis tasks (data wrangling, modeling, plotting)
    - implement advanced and cutting edge statistical methods
- Only comes with a basic Graphical User Interface (GUI)

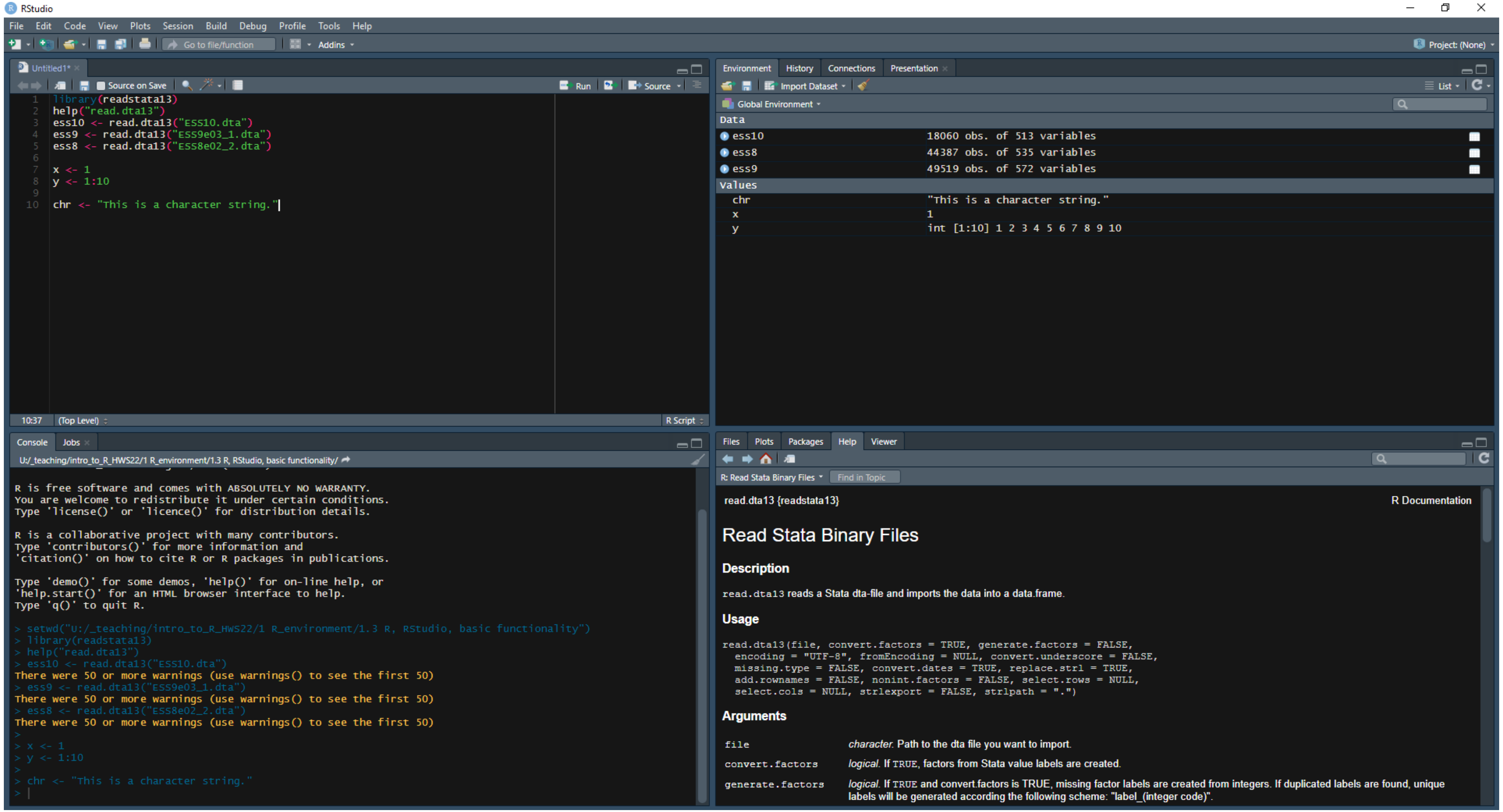
- RStudio is an intuitive and flexible interface for R called IDE (Integretated Development Environment)
- Makes R coding fun:
    - Nice overview of created objects, R scripts
    - Syntax highlighting
    - Spell checking
    - Auto-completion
- We code in R by using the RStudio Interface

R and the Graphical User Interface



```
R version 3.6.1 (2019-07-05) -- "Action of the Toes"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

RStudio Set Up



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1*
Source on Save
Run
Source
Environment History Connections Presentation
Import Dataset
List
Global Environment
10:37
(Top Level)
R Script
Console Jobs
Files Plots Packages Help Viewer
R: Read Stata Binary Files
Find in Topic

```
library(readstata13)
help("read.dta13")
ess10 <- read.dta13("ESS10.dta")
ess9 <- read.dta13("ESS9e03_1.dta")
ess8 <- read.dta13("ESS8e02_2.dta")

x <- 1
y <- 1:10

chr <- "This is a character string."
```

```
Data
ess10                    18060 obs. of 513 variables
ess8                     44387 obs. of 535 variables
ess9                     49519 obs. of 572 variables
Values
chr                      "This is a character string."
x                        1
y                        int [1:10] 1 2 3 4 5 6 7 8 9 10
```

U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic functionality/

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> setwd("U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic functionality")
> library(readstata13)
> help("read.dta13")
> ess10 <- read.dta13("ESS10.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess9 <- read.dta13("ESS9e03_1.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess8 <- read.dta13("ESS8e02_2.dta")
There were 50 or more warnings (use warnings() to see the first 50)
>
> x <- 1
> y <- 1:10
>
> chr <- "This is a character string."
> 
```

read.dta13 {readstata13} R Documentation

Read Stata Binary Files

Description

`read.dta13` reads a Stata dta-file and imports the data into a data.frame.

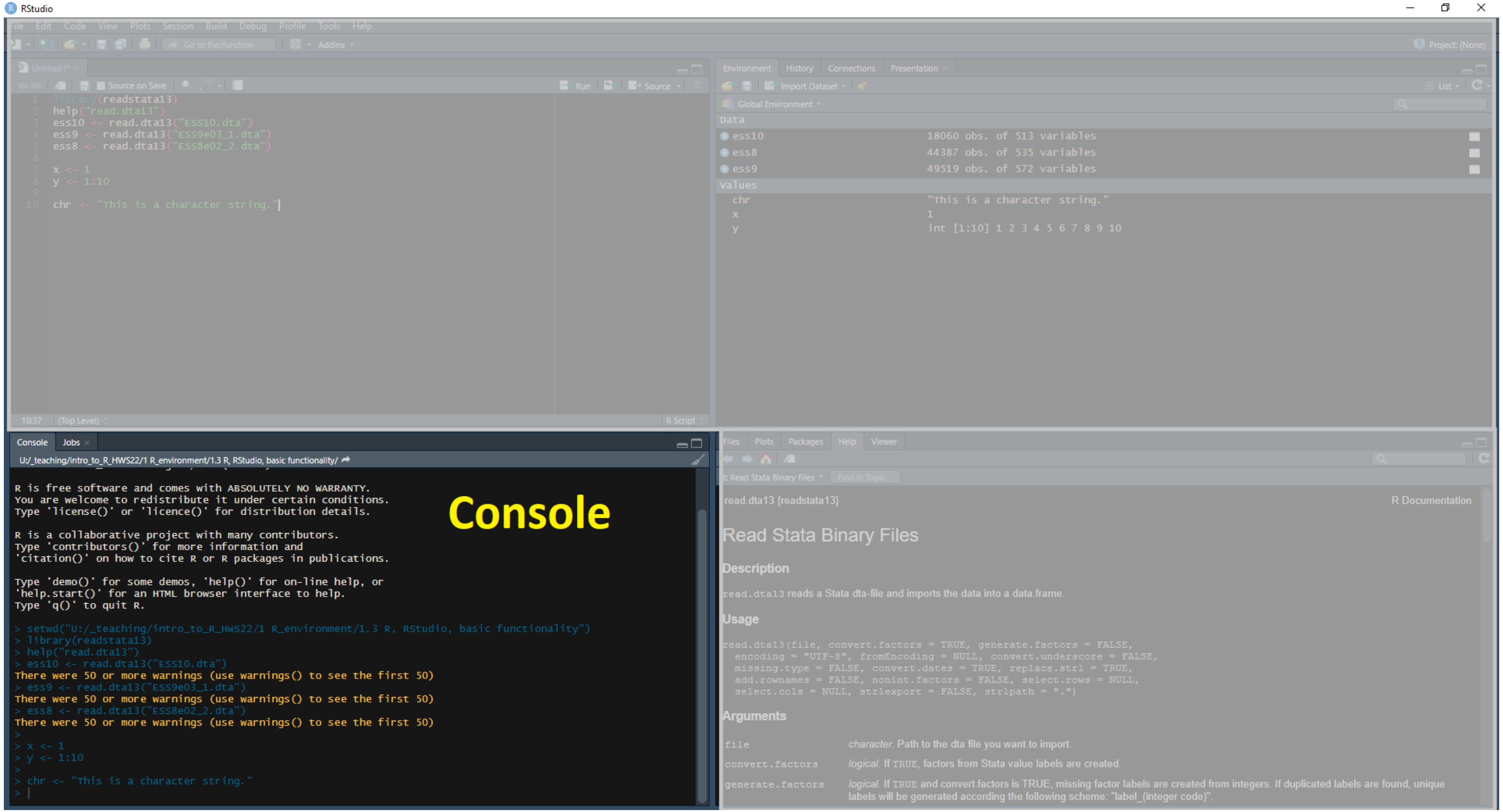
Usage

```
read.dta13(file, convert.factors = TRUE, generate.factors = FALSE,
  encoding = "UTF-8", fromEncoding = NULL, convert.underscore = FALSE,
  missing.type = FALSE, convert.dates = TRUE, replace.strl = TRUE,
  add.rownames = FALSE, nonint.factors = FALSE, select.rows = NULL,
  select.cols = NULL, strlexport = FALSE, strlpath = ".")
```

Arguments

`file`	*character.* Path to the dta file you want to import.
`convert.factors`	*logical.* If `TRUE`, factors from Stata value labels are created.
`generate.factors`	*logical.* If `TRUE` and convert.factors is TRUE, missing factor labels are created from integers. If duplicated labels are found, unique labels will be generated according the following scheme: "label_(integer code)".

RStudio Set Up



```r
library(readstata13)
help("read.dta13")
ess10 <- read.dta13("ESS10.dta")
ess9 <- read.dta13("ESS9e03_1.dta")
ess8 <- read.dta13("ESS8e02_2.dta")

x <- 1
y <- 1:10

chr <- "This is a character string."
```

```
Data
ess10                    18060 obs. of 513 variables
ess8                     44387 obs. of 535 variables
ess9                     49519 obs. of 572 variables
Values
chr                      "This is a character string."
x                        1
y                        int [1:10] 1 2 3 4 5 6 7 8 9 10
```

U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic functionality/

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> setwd("U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic functionality")
> library(readstata13)
> help("read.dta13")
> ess10 <- read.dta13("ESS10.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess9 <- read.dta13("ESS9e03_1.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess8 <- read.dta13("ESS8e02_2.dta")
There were 50 or more warnings (use warnings() to see the first 50)
>
> x <- 1
> y <- 1:10
>
> chr <- "This is a character string."
> |
```

read.dta13 {readstata13} R Documentation

Read Stata Binary Files

Description

read.dta13 reads a Stata dta-file and imports the data into a data.frame.

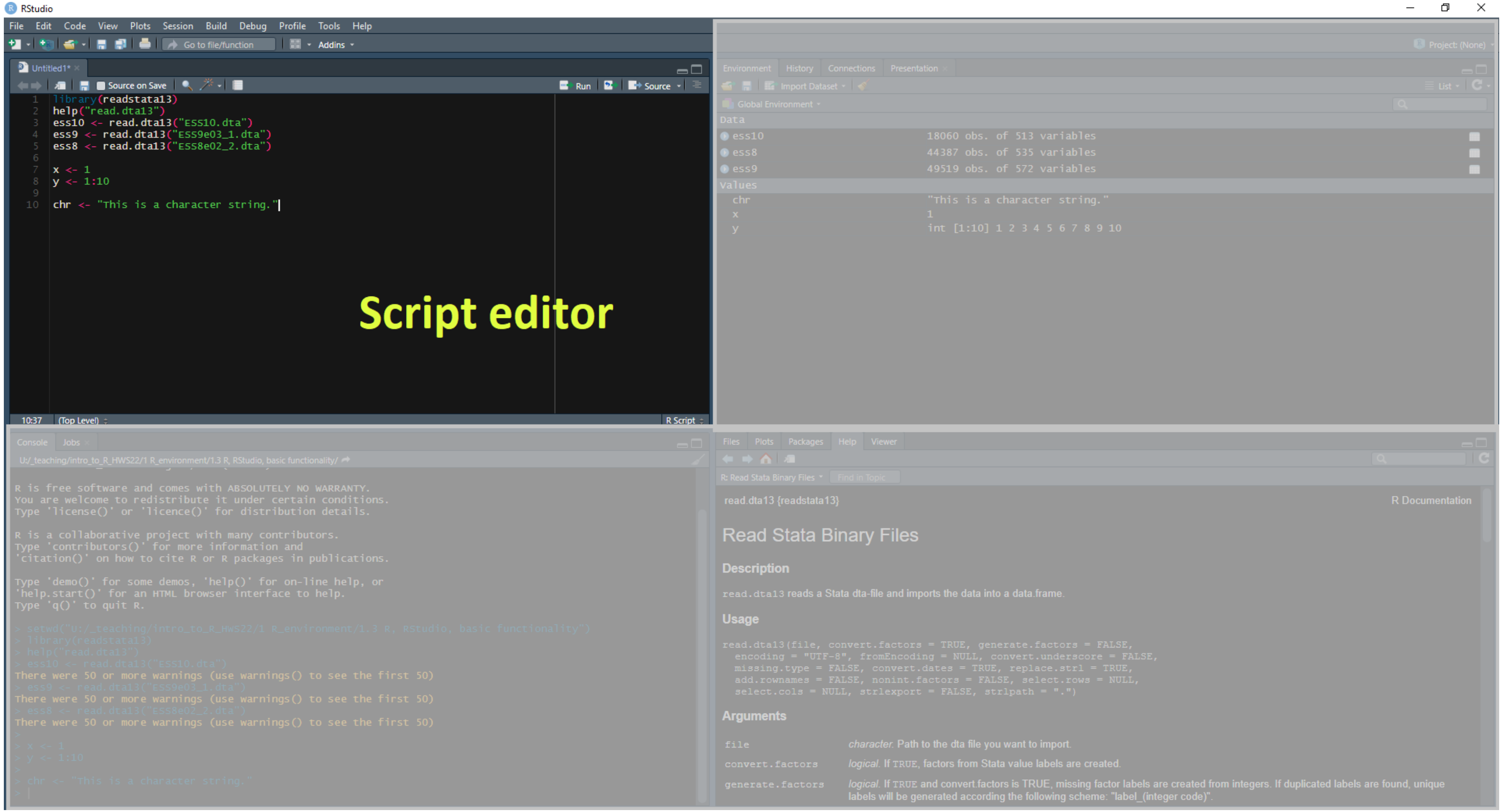
Usage

```
read.dta13(file, convert.factors = TRUE, generate.factors = FALSE,
  encoding = "UTF-8", fromEncoding = NULL, convert.underscore = FALSE,
  missing.type = FALSE, convert.dates = TRUE, replace.strl = TRUE,
  add.rownames = FALSE, nonint.factors = FALSE, select.rows = NULL,
  select.cols = NULL, strlexport = FALSE, strlpath = ".")
```

Arguments

file	*character.* Path to the dta file you want to import.
convert.factors	*logical.* If TRUE, factors from Stata value labels are created.
generate.factors	*logical.* If TRUE and convert.factors is TRUE, missing factor labels are created from integers. If duplicated labels are found, unique labels will be generated according the following scheme: "label_(integer code)".

RStudio Set Up



```r
library(readstata13)
help("read.dta13")
ess10 <- read.dta13("ESS10.dta")
ess9 <- read.dta13("ESS9e03_1.dta")
ess8 <- read.dta13("ESS8e02_2.dta")

x <- 1
y <- 1:10

chr <- "This is a character string."
```

Script editor

```
Data
ess10                 18060 obs. of 513 variables
ess8                  44387 obs. of 535 variables
ess9                  49519 obs. of 572 variables
Values
  chr                 "This is a character string."
  x                   1
  y                   int [1:10] 1 2 3 4 5 6 7 8 9 10
```

U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic functionality/

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> setwd("U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic functionality")
> library(readstata13)
> help("read.dta13")
> ess10 <- read.dta13("ESS10.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess9 <- read.dta13("ESS9e03_1.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess8 <- read.dta13("ESS8e02_2.dta")
There were 50 or more warnings (use warnings() to see the first 50)
>
> x <- 1
> y <- 1:10
>
> chr <- "This is a character string."
> 
```

read.dta13 {readstata13} R Documentation

Read Stata Binary Files

Description

read.dta13 reads a Stata dta-file and imports the data into a data.frame.

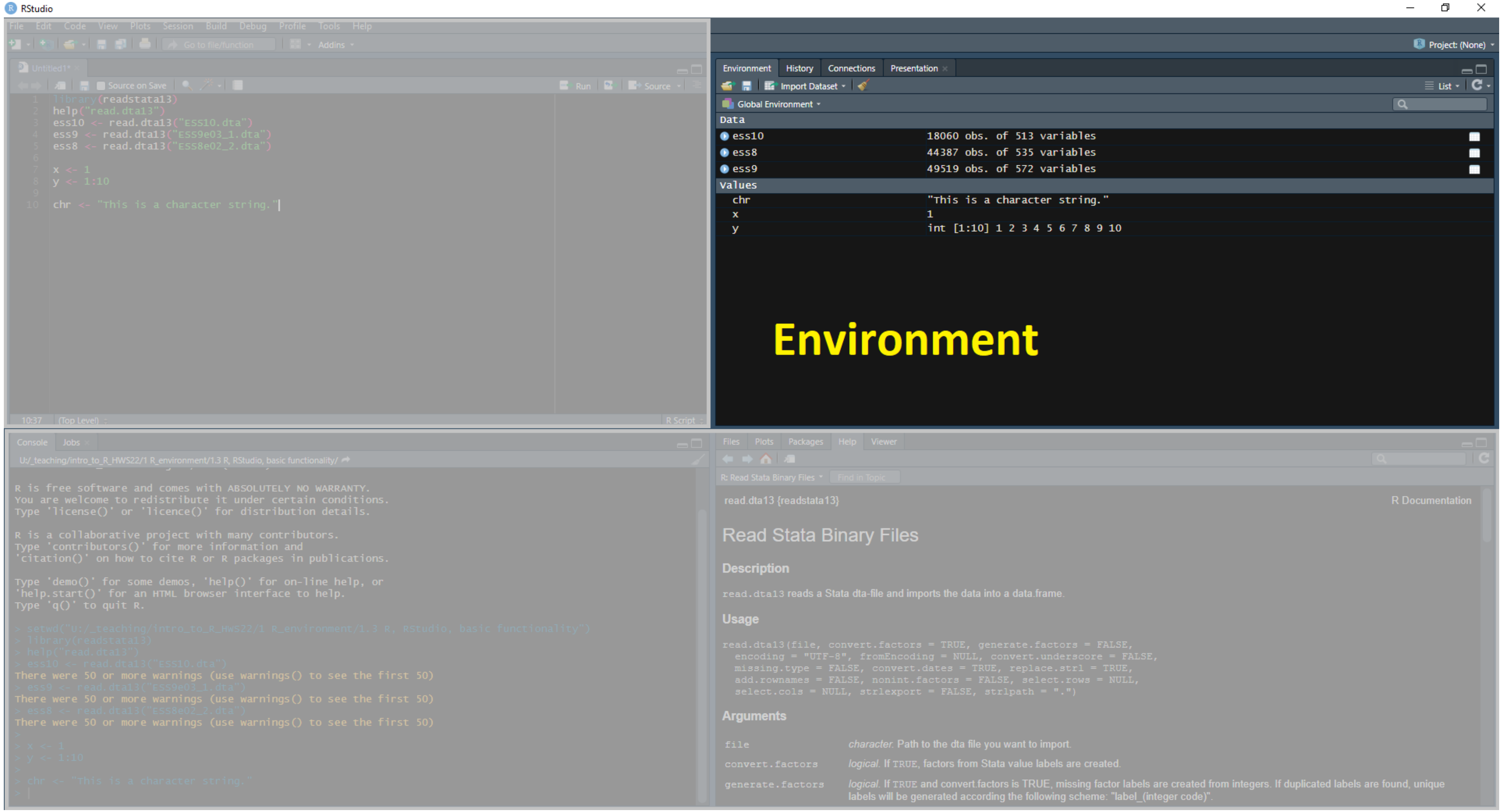
Usage

```
read.dta13(file, convert.factors = TRUE, generate.factors = FALSE,
  encoding = "UTF-8", fromEncoding = NULL, convert.underscore = FALSE,
  missing.type = FALSE, convert.dates = TRUE, replace.strl = TRUE,
  add.rownames = FALSE, nonint.factors = FALSE, select.rows = NULL,
  select.cols = NULL, strlexport = FALSE, strlpath = ".")
```

Arguments

file	*character.* Path to the dta file you want to import.
convert.factors	*logical.* If TRUE, factors from Stata value labels are created.
generate.factors	*logical.* If TRUE and convert.factors is TRUE, missing factor labels are created from integers. If duplicated labels are found, unique labels will be generated according the following scheme: "label_(integer code)".

RStudio Set Up


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Untitled1*
Source on Save
Run
Source
10:37
(Top Level)
R Script
Project: (None)
Environment History Connections Presentation
Import Dataset
List
Global Environment
Console Jobs
Files Plots Packages Help Viewer
R: Read Stata Binary Files
Find in Topic

```
library(readstata13)
help("read.dta13")
ess10 <- read.dta13("ESS10.dta")
ess9 <- read.dta13("ESS9e03_1.dta")
ess8 <- read.dta13("ESS8e02_2.dta")

x <- 1
y <- 1:10

chr <- "This is a character string."
```

```
Data
ess10                    18060 obs. of 513 variables
ess8                     44387 obs. of 535 variables
ess9                     49519 obs. of 572 variables
Values
  chr                    "This is a character string."
  x                      1
  y                      int [1:10] 1 2 3 4 5 6 7 8 9 10
```

Environment

U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic functionality/

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> setwd("U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic functionality")
> library(readstata13)
> help("read.dta13")
> ess10 <- read.dta13("ESS10.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess9 <- read.dta13("ESS9e03_1.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess8 <- read.dta13("ESS8e02_2.dta")
There were 50 or more warnings (use warnings() to see the first 50)
>
> x <- 1
> y <- 1:10
>
> chr <- "This is a character string."
> 
```

read.dta13 {readstata13} R Documentation

Read Stata Binary Files

Description

read.dta13 reads a Stata dta-file and imports the data into a data.frame.

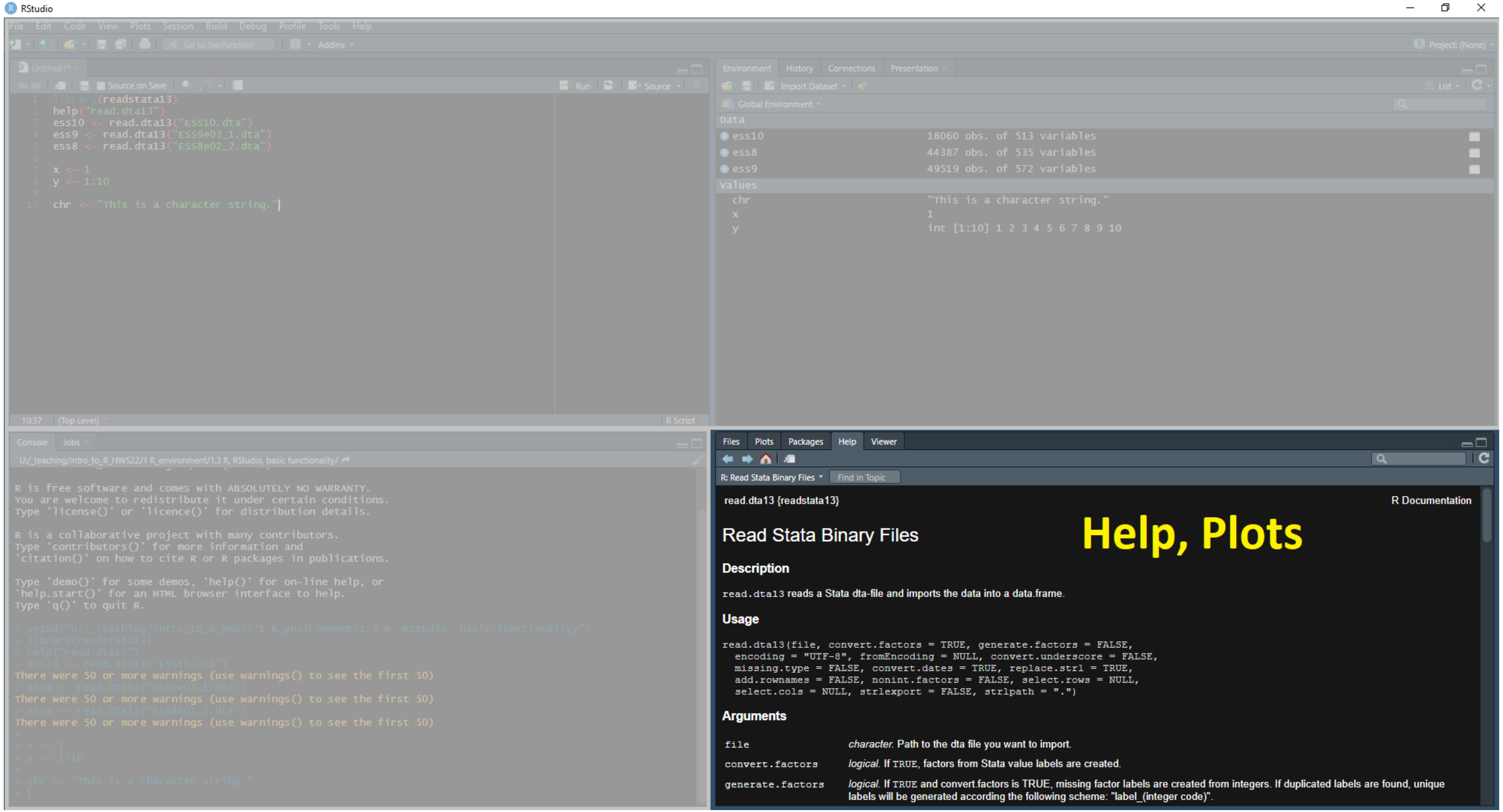
Usage

```
read.dta13(file, convert.factors = TRUE, generate.factors = FALSE,
  encoding = "UTF-8", fromEncoding = NULL, convert.underscore = FALSE,
  missing.type = FALSE, convert.dates = TRUE, replace.strl = TRUE,
  add.rownames = FALSE, nonint.factors = FALSE, select.rows = NULL,
  select.cols = NULL, strlexport = FALSE, strlpath = ".")
```

Arguments

file	*character*. Path to the dta file you want to import.
convert.factors	*logical*. If TRUE, factors from Stata value labels are created.
generate.factors	*logical*. If TRUE and convert.factors is TRUE, missing factor labels are created from integers. If duplicated labels are found, unique labels will be generated according the following scheme: "label_(integer code)".

RStudio Set Up



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1*
Source on Save
Run
Source
10:37
(Top Level)
R Script
Environment History Connections Presentation
Import Dataset
List
Global Environment
Console Jobs
Files Plots Packages Help Viewer
R: Read Stata Binary Files
Find in Topic

```
library(readstata13)
help("read.dta13")
ess10 <- read.dta13("ESS10.dta")
ess9 <- read.dta13("ESS9e03_1.dta")
ess8 <- read.dta13("ESS8e02_2.dta")

x <- 1
y <- 1:10

chr <- "This is a character string."
```

```
Data
ess10                    18060 obs. of 513 variables
ess8                     44387 obs. of 535 variables
ess9                     49519 obs. of 572 variables
Values
chr                      "This is a character string."
x                        1
y                        int [1:10] 1 2 3 4 5 6 7 8 9 10
```

U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic functionality/

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> setwd("U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic functionality")
> library(readstata13)
> help("read.dta13")
> ess10 <- read.dta13("ESS10.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess9 <- read.dta13("ESS9e03_1.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess8 <- read.dta13("ESS8e02_2.dta")
There were 50 or more warnings (use warnings() to see the first 50)
>
> x <- 1
> y <- 1:10
>
> chr <- "This is a character string."
> 
```

read.dta13 {readstata13} R Documentation

Read Stata Binary Files

Help, Plots

Description

`read.dta13` reads a Stata dta-file and imports the data into a data.frame.

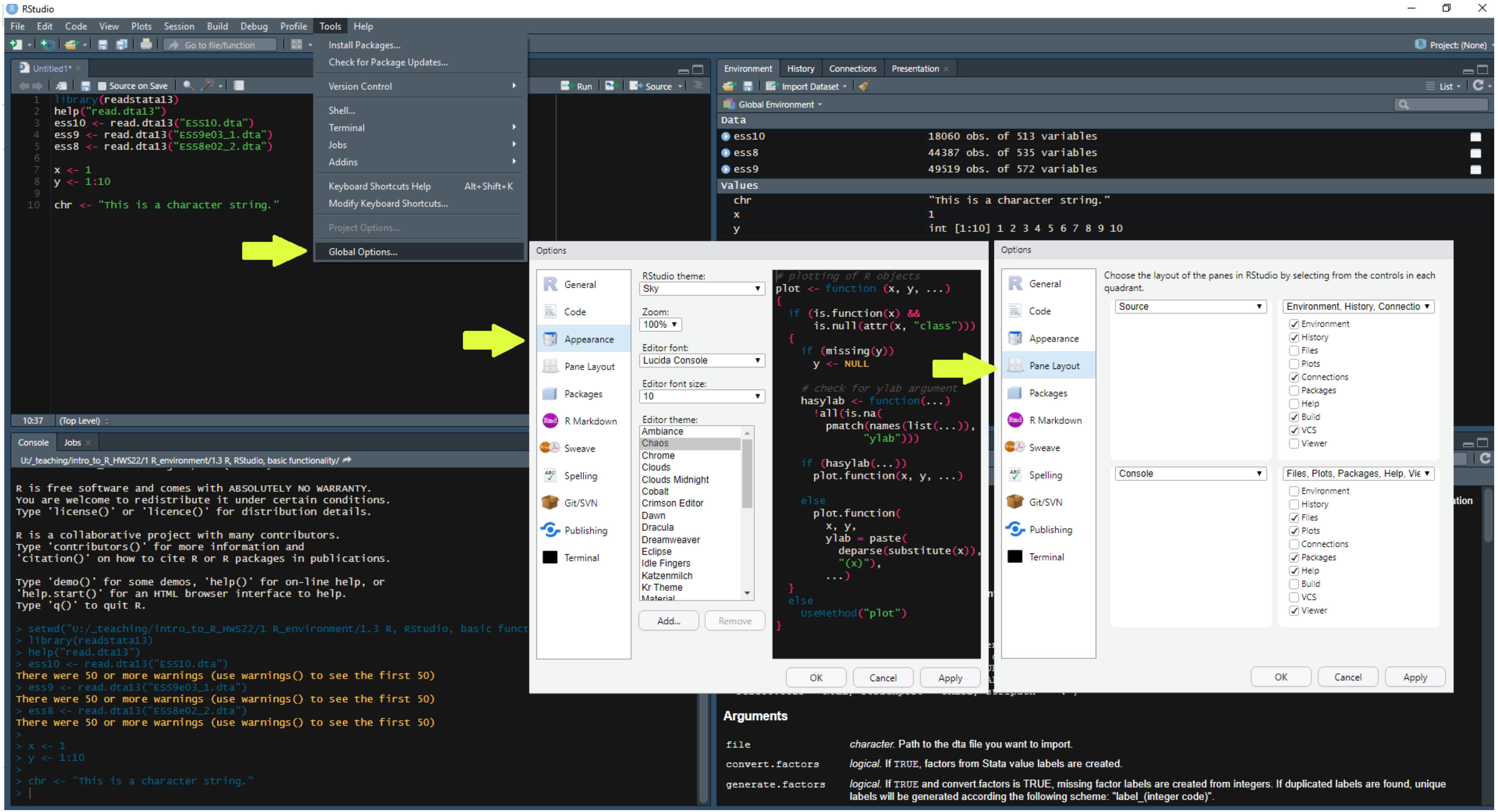
Usage

```
read.dta13(file, convert.factors = TRUE, generate.factors = FALSE,
  encoding = "UTF-8", fromEncoding = NULL, convert.underscore = FALSE,
  missing.type = FALSE, convert.dates = TRUE, replace.strl = TRUE,
  add.rownames = FALSE, nonint.factors = FALSE, select.rows = NULL,
  select.cols = NULL, strlexport = FALSE, strlpath = ".")
```

Arguments

`file`	*character.* Path to the dta file you want to import.
`convert.factors`	*logical.* If `TRUE`, factors from Stata value labels are created.
`generate.factors`	*logical.* If `TRUE` and convert.factors is TRUE, missing factor labels are created from integers. If duplicated labels are found, unique labels will be generated according the following scheme: "label_(integer code)".

RStudio Set Up



```r
library(readstata13)
help("read.dta13")
ess10 <- read.dta13("ESS10.dta")
ess9 <- read.dta13("ESS9e03_1.dta")
ess8 <- read.dta13("ESS8e02_2.dta")

x <- 1
y <- 1:10

chr <- "This is a character string."
```

```
Data
ess10                     18060 obs. of 513 variables
ess8                      44387 obs. of 535 variables
ess9                      49519 obs. of 572 variables
Values
chr                       "This is a character string."
x                         1
y                         int [1:10] 1 2 3 4 5 6 7 8 9 10
```

U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic functionality/

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> setwd("U:/_teaching/intro_to_R_HWS22/1 R_environment/1.3 R, RStudio, basic funct
> library(readstata13)
> help("read.dta13")
> ess10 <- read.dta13("ESS10.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess9 <- read.dta13("ESS9e03_1.dta")
There were 50 or more warnings (use warnings() to see the first 50)
> ess8 <- read.dta13("ESS8e02_2.dta")
There were 50 or more warnings (use warnings() to see the first 50)
>
> x <- 1
> y <- 1:10
>
> chr <- "This is a character string."
> |
```

RStudio theme: Sky

Zoom: 100%

Editor font: Lucida Console

Editor font size: 10

Editor theme: Ambiance, Chaos, Chrome, Clouds, Clouds Midnight, Cobalt, Crimson Editor, Dawn, Dracula, Dreamweaver, Eclipse, Idle Fingers, Katzenmilch, Kr Theme, Material

```r
# plotting of R objects
plot <- function (x, y, ...)
{
  if (is.function(x) &&
      is.null(attr(x, "class")))
  {
    if (missing(y))
      y <- NULL

    # check for ylab argument
    hasylab <- function(...)
      !all(is.na(
        pmatch(names(list(...)),
               "ylab")))

    if (hasylab(...))
      plot.function(x, y, ...)

    else
      plot.function(
        x, y,
        ylab = paste(
          deparse(substitute(x)),
          "(x)"),
        ...)
  }
  else
    UseMethod("plot")
}
```

Choose the layout of the panes in RStudio by selecting from the controls in each quadrant.

Source

Environment, History, Connectio

- ☑ Environment
- ☑ History
- ☐ Files
- ☐ Plots
- ☑ Connections
- ☐ Packages
- ☐ Help
- ☑ Build
- ☑ VCS
- ☐ Viewer

Console

Files, Plots, Packages, Help, Vie

- ☐ Environment
- ☐ History
- ☑ Files
- ☑ Plots
- ☐ Connections
- ☑ Packages
- ☑ Help
- ☐ Build
- ☐ VCS
- ☑ Viewer

Arguments

`file`	*character.* Path to the dta file you want to import.
`convert.factors`	*logical.* If `TRUE`, factors from Stata value labels are created.
`generate.factors`	*logical.* If `TRUE` and convert.factors is TRUE, missing factor labels are created from integers. If duplicated labels are found, unique labels will be generated according the following scheme: "label_(integer code)".

Basic Functionality

Basic R Functionality: Calculations

In it's very simplest form, R can just be used as a calculator.

```
2+2
```

```
## [1] 4
```

```
7^2
```

```
## [1] 49
```

```
1/4
```

```
## [1] 0.25
```

Basic R Functionality: Comments

When writing code, you can distinguish between two distinct elements of a code chunk

- annotations (denoted by #) which will not be executed by R
- genuine code

```r
# This is a comment
5*30
```

```
## [1] 150
```

```r
print("Hello World") # This is an in-line comment
```

```
## [1] "Hello World"
```

Basic R Functionality: Philosophy of R

"To understand computations in R, two slogans are helpful:

- *Everything that exists is an object.*
- *Everything that happens is a function call."*

--- John Chambers, developer of R.

Basic R Functionality: Functions

Functions are the workhorse of R programming. Functions take certain inputs as arguments and produces certain outputs given that all input arguments have been specified correctly.

Let's look at the sqrt function:

```
# calculate the square root of a number
sqrt(16)
```

```
## [1] 4
```

In order to understand how exactly a function works and to get help, type

```
help(sqrt)
?sqrt
```

Basic R Functionality: Arithmetic Functions

This is an overview of R's built in arithmetic functions that take a single scalar as an input.

- log()
- exp()
- sqrt()
- sin()
- cos()
- tan()

Basic R Functionality: Base R Functions

- names()
- colnames()
- rownames()

- length()
- nrow()
- ncol()
- dim()
- nchar()

- which()
- which.min()
- which.max()

- min()
- max()
- sum()

- round()
- ceiling()
- floor()

- c()
- cbind()
- unique()

- rep()
- seq()

Basic R Functionality: Objects

Since R is an object orientated programming language, values (range of values, character strings, datasets, …) can be stored in objects. This is done using the so-called assignment operator <-. An alternative would be using the equality sign =.

```r
x <- 5 # a numerical object
chr <- "Exemplary character string."
```

Of course, the output of a function can also be assigned to an object.

```r
y <- sqrt(16)
y = sqrt(16) # identical
```

Objects can be assessed and printed to the console by simply typing their name or wrapping it around print(object).

```r
y
```

```
## [1] 4
```

```r
print(y) # identical
```

```
## [1] 4
```

Packages

R Packages

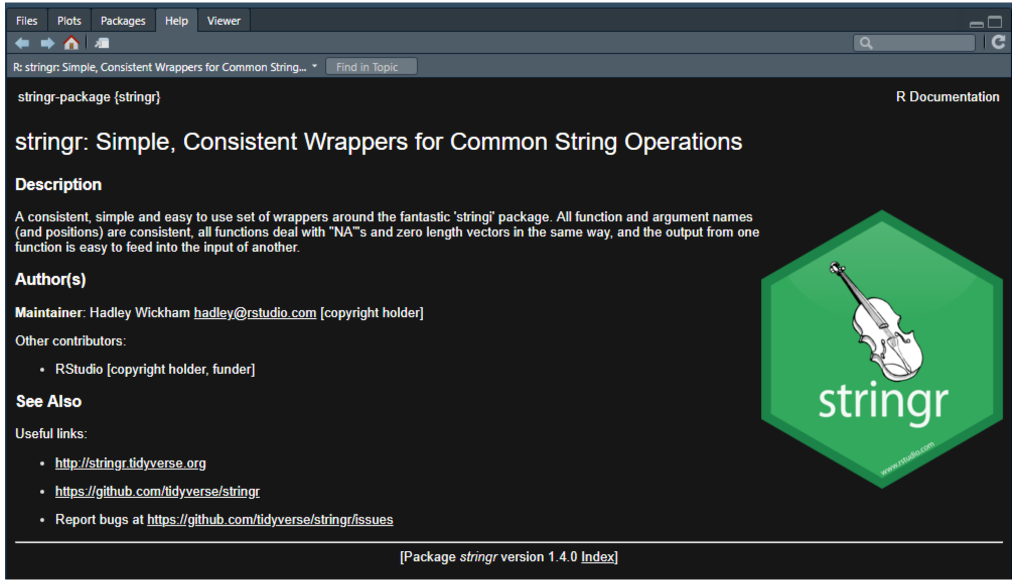
- Packages extend the basic functionality that comes with the installation of (base) R
- Packages are essentially collections of functions
  - automate tasks for you that otherwise would need your own extensive coding in base R
  - provide documentations of their functions
- Packages can be directly installed from within R and are available via The Comprehensive R Archive (CRAN)
- Working with packages requires a two-step process

```r
install.packages("stringr") # installing package, only needs to be done once
library(stringr) # loading package, required in every R session
```

An Exemplary Package: stringr

Let's have a look at an exemplary package. Let's first load it into our current R session using library() and inspect its documentation using help().

```r
library(stringr)
help(stringr)
```



stringr-package {stringr} R Documentation

stringr: Simple, Consistent Wrappers for Common String Operations

Description

A consistent, simple and easy to use set of wrappers around the fantastic 'stringi' package. All function and argument names (and positions) are consistent, all functions deal with "NA"'s and zero length vectors in the same way, and the output from one function is easy to feed into the input of another.

Author(s)

Maintainer: Hadley Wickham hadley@rstudio.com [copyright holder]

Other contributors:

- RStudio [copyright holder, funder]

See Also

Useful links:

- http://stringr.tidyverse.org
- https://github.com/tidyverse/stringr
- Report bugs at https://github.com/tidyverse/stringr/issues

[Package *stringr* version 1.4.0 Index]

The stringr packes provides:

- cohesive set of functions designed to make working with strings as easy as possible
- fast implementations of common string manipulations

```r
model_string <- str_c("Model ", 1:4)
print(model_string)
```

```
## [1] "Model 1" "Model 2" "Model 3" "Model 4"
```

Tidyverse - A Universe of R packages

Tidyverse

- The Tidyverse provides a collection of packages all developed and maintained by Hadley Wickham and his team at RStudio

Tidyverse - A Universe of R packages

Tidyverse

- The Tidyverse provides a collection of packages all developed and maintained by Hadley Wickham and his team at RStudio
- Most prominent examples include readr (data import), dplyr (data manipulation), ggplot2 (visulization) and stringr (string manipulations)

Tidyverse - A Universe of R packages

Tidyverse

- The Tidyverse provides a collection of packages all developed and maintained by Hadley Wickham and his team at RStudio
- Most prominent examples include readr (data import), dplyr (data manipulation), ggplot2 (visulization) and stringr (string manipulations)
- The packages in the tidyverse share an underlying design philosophy, grammar, and data structures and are particularly designed to work well together

Tidyverse - A Universe of R packages

Tidyverse

- The Tidyverse provides a collection of packages all developed and maintained by Hadley Wickham and his team at RStudio
- Most prominent examples include readr (data import), dplyr (data manipulation), ggplot2 (visulization) and stringr (string manipulations)
- The packages in the tidyverse share an underlying design philosophy, grammar, and data structures and are particularly designed to work well together
- You can install the complete tidyverse by typing a single line of code in the console:

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

Tidyverse vs. Base R

The Tidyverse has become so popular for data science in R that it provides substitutes for many common tasks that researchers face in their day-to-day work in R. This has led to a (friendly) divide in the R community where some try to rely on the Tidyverse as extensively as possible, whereas others prefer to code in base R.

base R

- allows for the full spectrum of R programming
- code can easily become long (like *really* long)
- often less readible
- code is built up from scratch
- you are forced to understand problems and hard-code solutions for yourself
- stable code across time

Tidyverse

- only focuses on a subset of common tasks
- code is usually clean and tidy
- often more readible
- set of ready-to-use functions
- quick fixes that are easy to implement even with little understanding
- introduces several dependencies, functionalities keep changing

Ultimately, you will develop your *own* coding style and philosophy.

References

Parts of this course are inspired by the following resources:

- Wickham, Hadley and Garrett Grolemund, 2017. *R for Data Science - Import, Tidy, Transform, Visualize, and Model Data*. O'Reilly.
- Bahnsen, Oke and Guido Ropers, 2022. *Introduction to R for Quantitative Social Science*. Course held as part of the GESIS Workshop Series.
- Breuer, Johannes and Stefan Jünger, 2021. *Introduction to R for Data Analysis*. Course held as part of the GESIS Summer School in Survey Methodology.
- Teaching material developed by Verena Kunz, David Weyrauch, Oliver Rittmann and Viktoriia Semenova.