Objects in R

R

Basic classes of objects

- Character
- Integer
- Numeric (real numbers)
- Logical (TRUE/FALSE)
- Factors (categorical information)

Three different **types** of objects

Created by Brgfx - Freepik.com

Created by Iconicbestiary - Freepik.com

Created by Freepik

Each object contains information of a certain **class**.

Storing objects

Storing objects is called **assignment**.

Assignment involves associating an object with a name.

The **assignment operators** in R are `<-` and `=`.

```
min_age <- 21
min_age = 21
```

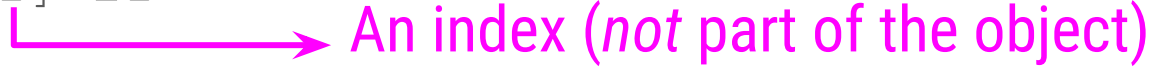
Object names

Printing objects

The > in the Console pane of RStudio Cloud indicates the **R prompt**.

Enter unassigned objects at the console to print them:

```
> 21
[1] 21
```



Print assigned objects by entering their name:

```
> min_age <- 21
> min_age
[1] 21
```

Character objects

Character objects can be created by surrounding text in double or single quotes.

```
"This is a character object."
'This is also a character object.'
```

Example of a character vector of length 1:

```
my_char <- "This is a character object."
```

Example of a character vector of length 2:

```
my_char_vec <- c("char object 1", "char object 2")
```

`c()` is the concatenation function

Integer objects

Integer objects are created by specifying `L` after an integer number.

```
num <- 1L
```

Example of an integer vector of length 3:

```
num_vec <- c(1L, 10L, 3L)
```

Creating an integer vector using the colon operator:

```
num_vec2 <- 2:5
```

Long printed output:

```
> 4:16
 [1]  4  5  6  7  8  9 10 11
 [9] 12 13 14 15 16
```

4 is the first number. 12 is the ninth.

Numeric objects

Numeric objects are created by simply specifying a number.

```
num <- 1.2
```

Example of a numeric vector of length 2:

```
num_vec <- c(1.2, 9.8)
```

Numeric objects

R as a calculator: enter mathematical expressions at the prompt.

```
> 1+5
[1] 6
> 2-3
[1] -1
> 4*2
[1] 8
> 4/5
[1] 0.8
> 3^2
[1] 9
```

Logical objects

Logical values in R are `TRUE` and `FALSE`.

```
check_condition <- TRUE
check_condition <- FALSE
```

Example of a logical vector of length 3:

```
check_condition <- c(TRUE, TRUE, FALSE)
```

Factors

Using the `factor` function:

```
> colors <- c("red", "red", "blue", "red", "blue")
> colors_factor1 <- factor(colors, levels = c("red",
"blue"))
> colors_factor1
[1] red  red  blue red  blue
Levels: red blue
```

Levels are in the specified order

Using the `as.factor` coercion function:

```
> colors_factor2 <- as.factor(colors)
> colors_factor2
[1] red  red  blue red  blue
Levels: blue red
```

Levels are in alphabetical order

Factors

The `factor` function also allows customization of labels:

```
> ozone_levels <- c(1,2,1,3,1,1)
> ozone_factor <- factor(ozone_levels, levels = 1:3,
labels = c("low", "medium", "high"))
> ozone_factor
[1] low    medium low    high   low    low
Levels: low medium high

> ozone_factor2 <- factor(ozone_levels, levels =
c(2,1,3), labels = c("medium", "low", "high"))
> ozone_factor2
[1] low    medium low    high   low    low
Levels: medium low high
```

Same labeling but different ordering of levels

Data frames

Columns correspond to variables.

model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

Rows correspond to observations.

Missing values

```
> char_vec <- c(NA, "two", "four")
> char_vec
[1] NA     "two"  "four"
> num_vec <- c(1L, 10L, NA, 3L)
> num_vec
[1]  1 10 NA  3
> num_vec <- c(1.2, 9.8, NA)
> num_vec
[1] 1.2 9.8  NA
> logi_vec <- c(TRUE, NA, FALSE, FALSE)
> logi_vec
[1]  TRUE    NA FALSE FALSE
> factor_vec <- as.factor(c(NA, "apple", "banana"))
> factor_vec
[1] <NA>   apple  banana
Levels: apple banana
> 0/0
[1] NaN
```

Most missing values are denoted with `NA`

`NaN` denotes the result of an undefined mathematical operation

Determining the class of an object: `class()`

```
> min_age <- 21
> class(min_age)
[1] "numeric"

> min_age <- 21L
> class(min_age)
[1] "integer"

> colors <- c("red", "red", "blue", "red", "blue")
> class(colors)
[1] "character"

> colors_factor1 <- factor(colors, levels = c("red", "blue"))
> class(colors_factor1)
[1] "factor"
```