

01 Basic R Syntax

Martin Hanewald

2019-02-19

Grundlegende Datentypen

```
num <- 5.1 # numeric
int <- 5L # integer
char <- '433' # character
char2 <- "slkjs" # character

logic <- TRUE # logical
logic2 <- TRUE | FALSE # logical
logic3 <- T #logical

date <- as.Date('2015-01-01')

null <- NULL
na <- NA
```

Auf Datentypen testen

```
null > 5
#> logical(0)

na > 5
#> [1] NA

is.numeric(num) # TRUE
#> [1] TRUE
is.integer(num) # FALSE
#> [1] FALSE

is.numeric(int) # TRUE
#> [1] TRUE
is.integer(int) # TRUE
#> [1] TRUE

is.character(char)
#> [1] TRUE

is.logical(F)
#> [1] TRUE
```

Datenkonstrukte

Vektoren

Die Funktion `c` (für combine) erzeugt Vektoren aus Einzelelementen oder anderen Vektoren. Dabei werden unterschiedliche Datentypen ineinander überführt ('coercing'). Die Transformationsrichtung ist immer:

```

c(TRUE, TRUE, FALSE)
#> [1] TRUE TRUE FALSE
c(1,3,5,8)
#> [1] 1 3 5 8
c('a', 'b', 'c')
#> [1] "a" "b" "c"
c(num, char)
#> [1] "5.1" "433"
c(num, int)
#> [1] 5.1 5.0
c(logic, num)
#> [1] 1.0 5.1
c(logic, int)
#> [1] 1 5

vec <- c(num, int, logic, char)
vec
#> [1] "5.1" "5" "TRUE" "433"

```

Rechnen mit Vektoren

```

x <- 1:10
y <- 5:14

x + y
#> [1] 6 8 10 12 14 16 18 20 22 24

x * y
#> [1] 5 12 21 32 45 60 77 96 117 140

x / y
#> [1] 0.2000000 0.3333333 0.4285714 0.5000000 0.5555556 0.6000000 0.6363636
#> [8] 0.6666667 0.6923077 0.7142857

```

Recycling:

Ein kürzere Vektor wird immer wiederholt wenn er mit einem längeren kombiniert wird.

```

y <- 1:2

x + y
#> [1] 2 4 4 6 6 8 8 10 10 12

x * y
#> [1] 1 4 3 8 5 12 7 16 9 20

```

Indizierung

```

x[5] # das fünfte Element
#> [1] 5

x[5:10] # das fünfte bis 10te Elemente
#> [1] 5 6 7 8 9 10

```

```
x[x < 5] # alle Elemente deren Wert < 5
#> [1] 1 2 3 4

ind <- x < 4 | x > 9 # Kombination von Logischen Operatoren
x[ind]
#> [1] 1 2 3 10

x[c(T, F, F)]
#> [1] 1 4 7 10

x[8:10] <- NA
```

Nützliche Funktionen zur Vektorerzeugung

```
seq(5, 100, by = 5) # Sequenzen
#> [1] 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85
#> [18] 90 95 100
1:20 * 5 # äquivalent
#> [1] 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85
#> [18] 90 95 100

rep(1:5, 3) # Repeat 3 mal
#> [1] 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5

seq_along(letters) # 1 bis Länge eines anderen Vektors
#> [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
#> [24] 24 25 26
```

Matrizen

```
mat <- matrix(1:9, nrow=3) # numeric matrix
mat
#>      [,1] [,2] [,3]
#> [1,] 1 4 7
#> [2,] 2 5 8
#> [3,] 3 6 9

mat <- matrix(letters, nrow=2) # character matrix
mat
#>      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
#> [1,] "a" "c" "e" "g" "i" "k" "m" "o" "q" "s" "u" "w" "y"
#> [2,] "b" "d" "f" "h" "j" "l" "n" "p" "r" "t" "v" "x" "z"

# Indizierung von Matrizen: [Zeile, Spalte]
mat[2, 5:10] # 2. Zeile, 5.-10. Spalte
#> [1] "j" "l" "n" "p" "r" "t"
```

Listen

```
l <- list(first = x, second = y, third = char) # 'Named list'
l
#> $first
#> [1] 1 2 3 4 5 6 7 NA NA NA
#>
#> $second
```

```
#> [1] 1 2
#>
#> $third
#> [1] "433"

names(l)
#> [1] "first" "second" "third"

l$first
#> [1] 1 2 3 4 5 6 7 NA NA NA

l[1] # Das erste Listenelement (immer noch eine Liste)
#> $first
#> [1] 1 2 3 4 5 6 7 NA NA NA

l[[1]][5] # Das fünfte Vektor Element des ersten Listenelements
#> [1] 5

a <- list()

a$nice <- function(x) sum(x)
a$other <- 5

a
#> $nice
#> function (x)
#> sum(x)
#>
#> $other
#> [1] 5
```

Dataframes

```
names <- c('Henri', 'Klaus', 'Tiffany')
age <- c(3, 21, 55)
isadult <- age > 18

df <- data.frame(names, age, isadult)
df
#>   names age isadult
#> 1 Henri   3  FALSE
#> 2 Klaus  21   TRUE
#> 3 Tiffany 55   TRUE

df$age # Rückgabe ist Vektor
#> [1] 3 21 55

df['age'] # Rückgabe ist Data.Frame
#>   age
#> 1   3
#> 2  21
#> 3  55

df[['age']] # Rückgabe ist Vektor
#> [1] 3 21 55

df[,2] # Rückgabe ist Vektor
#> [1] 3 21 55

df[1,2] <- 22 # Änderung von Werten
```

df

```
#>  names age isadult
#> 1 Henri  22    FALSE
#> 2 Klaus  21     TRUE
#> 3 Tiffy  55     TRUE
```

str(df) # Datenstruktur

```
#> 'data.frame':  3 obs. of  3 variables:
#> $ names  : Factor w/ 3 levels "Henri","Klaus",...: 1 2 3
#> $ age    : num  22 21 55
#> $ isadult: logi  FALSE TRUE TRUE
```

summary(df) # Summary Statistiken

```
#>  names      age      isadult
#> Henri:1  Min.   :21.00  Mode :logical
#> Klaus:1  1st Qu.:21.50  FALSE:1
#> Tiffy:1  Median :22.00  TRUE  :2
#>          Mean    :32.67
#>          3rd Qu.:38.50
#>          Max.    :55.00
```