

Einfache Objekte

Dirk Seidensticker/Clemens Schmid

6. Februar 2016

Einführung

vectors und data.frames anlegen

vectors und data.frames adressieren

vectors und data.frames manipulieren

Informationen zu Objekten anfragen

Einführung

- R speichert Daten, Zwischenergebnisse, Funktionen etc. in verschiedenen Strukturen im **Hauptspeicher**
- Kategorien primitiver Objekttypen: **Basic Vector** (Vektor mit einem Datentyp), **Compound objects** (Listen und S4-Objekte), Special objects, R language, **Functions**, Internal, Bytecode Objects
- Datentypen in der praktischen Anwendung: **vectors**, lists, matrices, arrays, factors, **data.frames**, timeseries, formulas, shingles, connections
- vectors (dazu gehören auch einzelne Variablen) und data.frames genügen für viele Anwendungen mit R
- Objekte haben Eigenschaften (**attributes**), die auch entscheiden, wie R mit den Objekten umgeht: **class**, dim, dimnames, row.names, levels etc.

vectors und data.frames anlegen

Zuweisung von einzelnen Objekten

```
x <- ...
```

```
y -> ...
```

```
z = ...
```

Beschreibung

Variablen einfach mit Werten verknüpfen. Variablen können aber alle möglichen R-Objekte referenzieren.

Beispiel

```
a <- 5
```

```
b <- "Glockenbecher"
```

```
a -> c
```

```
d = 5
```

```
b
```

```
## [1] "Glockenbecher"
```

`c(..., recursive = FALSE)`

Beschreibung

Mehrere Objekte eines Datentyps in einem Vektor (Liste mit Reihenfolge) anordnen. Der Datentyp wird automatisch zugeordnet.

Beispiel

```
a <- c(1,2,3,4,5,10,15,35,55)
```

```
b <- c("Dechsel", "Axt", "Axt", "Beil", "Dechsel")
```

a

```
## [1] 1 2 3 4 5 10 15 35 55
```

b

```
## [1] "Dechsel" "Axt"      "Axt"      "Beil"     "Dechsel"
```

Achtung:

```
c <- c(1, "Dechsel", 4)
```

```
data.frame(..., row.names = NULL, check.rows = FALSE,  
check.names = TRUE, stringsAsFactors =  
default.stringsAsFactors())
```

Beschreibung

Mehrere Vektoren einer Klasse in einer Kreuztabelle anordnen.

Beispiel

```
a <- c("Grube 1", "Grube 2", "Grube 3", "Grube 4", "Grube 5")  
b <- c("Dechsel 1", "Axt 1", "Axt 2", "Beil 1", "Dechsel 2")  
c <- c(5,3,2,5,5)
```

```
Grabung1 <- data.frame(  
  Objekt = b,  
  Anzahl = c,  
  row.names = a,  
  stringsAsFactors = FALSE  
)
```



```
data.frame(..., row.names = NULL, check.rows = FALSE,  
check.names = TRUE, stringsAsFactors =  
default.stringsAsFactors())
```

Beschreibung

Mehrere Vektoren einer Klasse in einer Kreuztabelle anordnen.

Beispiel

Grabung1

##	Objekt	Anzahl
## Grube 1	Dechsel 1	5
## Grube 2	Axt 1	3
## Grube 3	Axt 2	2
## Grube 4	Beil 1	5
## Grube 5	Dechsel 2	5

vectors und data.frames adressieren

[...] und ...\$....

Beschreibung

Adressieren von Objekten in Vektoren, data.frames und Matrizen.

Beispiel

```
a[1]
```

```
## [1] "Grube 1"
```

```
a[1:3]
```

```
## [1] "Grube 1" "Grube 2" "Grube 3"
```

```
a[c(1,2,5)]
```

```
## [1] "Grube 1" "Grube 2" "Grube 5"
```

[...] und ...\$....

Beschreibung

Adressieren von Objekten in Vektoren, data.frames und Matrizen.

Beispiel

```
Grabung1$Objekt
```

```
## [1] "Dechsel 1" "Axt 1"      "Axt 2"      "Beil 1"     "Dechsel 2"
```

```
Grabung1$Objekt[3]
```

```
## [1] "Axt 2"
```

[...] und ...\$....

Beschreibung

Adressieren von Objekten in Vektoren, data.frames und Matrizen.

Beispiel

```
Grabung1[3,1]
```

```
## [1] "Axt 2"
```

```
Grabung1[c(1,2),]
```

```
##           Objekt Anzahl
## Grube 1 Dechsel 1      5
## Grube 2      Axt 1      3
```

```
Grabung1[,1]
```

```
## [1] "Dechsel 1" "Axt 1"      "Axt 2"      "Beil 1"     "Dechsel 2"
```

vectors und data.frames manipulieren

```
# Werte aus vector löschen
```

```
a[-2]
```

```
## [1] "Grube 1" "Grube 3" "Grube 4" "Grube 5"
```

```
a[c(-1,-3)]
```

```
## [1] "Grube 2" "Grube 4" "Grube 5"
```

```
a[c(2,4,5)]
```

```
## [1] "Grube 2" "Grube 4" "Grube 5"
```

```
# Werte aus data.frame löschen
```

```
Grabung1[3,1] <- NA
```

```
# Spalten/Zeilen nach Index aus data.frame löschen
```

```
Grabung1[c(-2,-3,-4),]
```

```
##           Objekt Anzahl
```

```
## Grube 1 Dechsel 1      5
```

```
## Grube 5 Dechsel 2      5
```

```
# Spalten nach Namen aus data.frame löschen
```

```
Grabung.dest <- Grabung1
```

```
Grabung.dest$Objekt <- NULL
```

```
Grabung.dest
```

```
##           Anzahl
```

```
## Grube 1      5
```

```
## Grube 2      3
```

```
## Grube 3      2
```

```
## Grube 4      5
```

```
## Grube 5      5
```



```
# data.frame nach Zeilen trennen
```

```
Grabung1.A <- Grabung1[c(1:2),]
```

```
Grabung1.A
```

```
##           Objekt Anzahl
```

```
## Grube 1 Dechsel 1      5
```

```
## Grube 2      Axt 1      3
```

```
Grabung1.B <- Grabung1[c(3:5),]
```

```
Grabung1.B
```

```
##           Objekt Anzahl
```

```
## Grube 3      <NA>      2
```

```
## Grube 4      Beil 1      5
```

```
## Grube 5 Dechsel 2      5
```

```
# data.frame nach Spalten trennen
```

```
Grabung1.C <- Grabung1[,1]
```

```
Grabung1.C
```

```
## [1] "Dechsel 1" "Axt 1"      NA           "Beil 1"     "Dechsel 2"
```

```
Grabung1.D <- Grabung1[,2]
```

```
Grabung1.D
```

```
## [1] 5 3 2 5 5
```

```
# vector zusammenfügen
```

```
e <- c(1,2,3)
```

```
f <- c(4,5,6)
```

```
g <- c(7,8,9)
```

```
h <- c(e,f,g,10)
```

```
h
```

```
## [1] 1 2 3 4 5 6 7 8 9 10
```

Zusammenfügen/Hinzufügen

data.frame nach Zeilen zusammenfügen

```
Grabung1.A[1,]
```

```
##           Objekt Anzahl
## Grube 1 Dechsel 1      5
```

```
Grabung1.B[1,]
```

```
##           Objekt Anzahl
## Grube 3   <NA>      2
```

```
rbind(Grabung1.A, Grabung1.B)
```

```
##           Objekt Anzahl
## Grube 1 Dechsel 1      5
## Grube 2     Axt 1      3
## Grube 3   <NA>      2
## Grube 4    Beil 1      5
## Grube 5 Dechsel 2      5
```

Zusammenfügen/Hinzufügen

```
# data.frame nach Spalten zusammenfügen
```

```
Grabung1.C
```

```
## [1] "Dechsel 1" "Axt 1"      NA           "Beil 1"     "Dechsel 2"
```

```
Grabung1.D
```

```
## [1] 5 3 2 5 5
```

```
data.frame(Grabung1.C, Grabung1.D)
```

```
##   Grabung1.C Grabung1.D
```

```
## 1 Dechsel 1          5
```

```
## 2      Axt 1          3
```

```
## 3      <NA>          2
```

```
## 4      Beil 1          5
```

```
## 5 Dechsel 2          5
```

```
# Werte ändern: adressieren + neu zuweisen
```

```
i <- c(1,2,3,4,5,6,7,8)
```

```
i[5] <- 155
```

```
i
```

```
## [1] 1 2 3 4 155 6 7 8
```

```
Grabung1[,2] <- c(133,244,355,466,577)
```

```
Grabung1
```

```
##           Objekt Anzahl
```

```
## Grube 1 Dechsel 1    133
```

```
## Grube 2      Axt 1    244
```

```
## Grube 3      <NA>    355
```

```
## Grube 4     Beil 1    466
```

```
## Grube 5 Dechsel 2    577
```

Namen ändern: Namen adressieren + neu zuweisen

```
k <- c(1,2,3,4)
```

```
names(k) <- c("Dechsel", "Landschnecke", "Bronzeobjekt", "Goldring")
```

```
k
```

```
##      Dechsel Landschnecke Bronzeobjekt      Goldring
```

```
##           1             2             3             4
```

```
colnames(Grabung1)
```

```
## [1] "Objekt" "Anzahl"
```

```
row.names(Grabung1)
```

```
## [1] "Grube 1" "Grube 2" "Grube 3" "Grube 4" "Grube 5"
```

Informationen zu Objekten anfragen

attributes(x)

attributes(x) <- value

Beschreibung

Gibt die Attribute eines Objekts an und erlaubt deren Manipulation

Beispiel

```
attributes(Grabung1)
```

```
## $names
## [1] "Objekt" "Anzahl"
##
## $row.names
## [1] "Grube 1" "Grube 2" "Grube 3" "Grube 4" "Grube 5"
##
## $class
## [1] "data.frame"
```

class(x)

Beschreibung

Gibt die Klasse eines Objekts an.

Beispiel

```
class(Grabung1)
```

```
## [1] "data.frame"
```

str(x, ...)

Beschreibung

Gibt eine strukturelle Zusammenfassung zu einem Objekt aus.

Beispiel

```
str(Grabung1)
```

```
## 'data.frame':    5 obs. of  2 variables:  
##  $ Objekt: chr  "Dechsel 1" "Axt 1" NA "Beil 1" ...  
##  $ Anzahl: num  133 244 355 466 577
```

```
summary(x, ...)
```

Beschreibung

Gibt eine statistische Zusammenfassung zu einem Objekt aus.

Beispiel

```
summary(Grabung1)
```

##	Objekt	Anzahl
##	Length:5	Min. :133
##	Class :character	1st Qu.:244
##	Mode :character	Median :355
##		Mean :355
##		3rd Qu.:466
##		Max. :577