--- title: "MojeRnotizenZwei" author: "jm" date: "18 6 2021" output: html\_document ---

**Hilfe**

help() or help(package = „ggplot2“)

**Unload / remove**

detach („package:ggplot2“, unload=TRUE)

remove.package(„ggplot2“)

**Which version of R:**

* Version

sessionInfo(): Infos with all packages

**Which funtions in a package**

help(package = „ggplt2“)

browseVignettes(„ggplot2“) – more Infos about the functions in a package

**Projects**

Easy organization+sharing+startback

All variables and presentations

**Versioncontrol**

**Commit**

* Push updates o repository
* Pull the repository to my local computer. Am I uptodate with the current repo
* Stage one file and commit it at the same moment. Then the next file
* My brench is not known to the repo
* Merging: pushing my new file tot he main
* Each commit should have just one issue, and purpose fort he change

**GitHub**

git config --global user.name "jankmail"

git config --global user.email jank.mail@t-online.de

git config –-list (oder –l)

Tools-> General -> GIt SHA Key

* GitHub Settings SHA Key

New Files: EnviromentQuadrant-Git

* Commit

   (New windows with Difference to previous versions)

In the upper right Quadrant: fort he commit-message

* Commit (immer noch lokal)
* Push to the repository

Generating public/private rsa key pair.

Your identification has been saved in C:/Users/Medienberater/.ssh/id\_rsa.

Your public key has been saved in C:/Users/Medienberater/.ssh/id\_rsa.pub.

The key fingerprint is:

SHA256:edyzOy4Nk8z4c0NPgVMxUY9TJ3P2wWHWcOcOwkPyWGc medienberater@MB\_AKSMZ

The key's randomart image is:

+---[RSA 3072]----+

|          . o EX@|

|           B o+%B|

|          . =o+ +|

|         o .oo.+ |

|        S+o.o. ..|

|        ..\* .o.  |

|         . =.o   |

|          + =..  |

|           =oo   |

+----[SHA256]-----+

ssh-rsa medienberater@MB\_AKSMZ

**R Markdown**

\*italics\*  \*\*bold\*\*  #HeaderBig  ##HeaderMiddle  ```{r file}  code ausführen ```

- Aufzählung hat zwei Leerzeichen am Ende

[linkname](<http://irgendwtwas.com)>

Git Tutorial mit Bitbucket

<https://www.atlassian.com/de/git/tutorials/learn-git-with-bitbucket-cloud>

<https://www.dev-insider.de/erste-schritte-mit-git-und-github-a-905364/>

**Damit der Push endlich läuft**  
Browser musste authentifiziert  werden

Medienberater@MB\_AKSMZ MINGW64 ~/Documents/R/hello-world (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

Changes not staged for commit:

  (use "git add <file>..." to update what will be committed)

  (use "git restore <file>..." to discard changes in working directory)

        modified:   MojeCheatSheetDataScience

Untracked files:

  (use "git add <file>..." to include in what will be committed)

        .gitignore hello-world.Rproj

no changes added to commit (use "git add" and/or"git commit -a")

Medienberater@MB\_AKSMZ MINGW64 ~/Documents/R/hello-world (main)

$ git add .

Medienberater@MB\_AKSMZ MINGW64 ~/Documents/R/hello-world (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

Changes to be committed:

  (use "git restore --staged <file>..." to unstage)

        new file:   .gitignore

        modified:   MojeCheatSheetDataScience

        new file:   hello-world.Rproj

Medienberater@MB\_AKSMZ MINGW64 ~/Documents/R/hello-world (main)

$ git commit -m "vllt läuft es nach git status git add . git status ist jetzt ohne Fehler UND DAnn git commit -m MESSAGE"

[main f24c2d0] vllt läuft es nach git status git add . git status ist jetzt ohne Fehler UND DAnn git commit -m MESSAGE

 3 files changed, 19 insertions(+)

 create mode 100644 .gitignore

 create mode 100644 hello-world.Rproj

Medienberater@MB\_AKSMZ MINGW64 ~/Documents/R/hello-world (main)

$ git push

Select an authentication method for '<https://github.com/':>

  1. Web browser (default)

  2. Personal access token

option (enter for default):

info: please complete authentication in your browser...

Enumerating objects: 7, done.

Counting objects: 100% (7/7), done.

Delta compression using up to 4 threads

Compressing objects: 100% (4/4), done.

Writing objects: 100% (5/5), 746 bytes | 746.00 KiB/s, done.

Total 5 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (1/1), completed with 1 local object.

To <https://github.com/jankmail/hello-world>

7d3fdac..f24c2d0  main -> main

**Programming in R**

Console:

getwd() = pwd

dir() = ls ls()

read.csv("mydata.csv")

Functions:

myfunction <- function() {

x <- rnorm(100) Random hundert Zufallswerte

  mean(x) Mittelwert

}

second <- function(x) {

  x + rnorm(length(x)) #Dem Wert x wird etwas noise gegeben

}

second(4:10) es werden die Zahlen 4 bis 10 eingegeben

source („Dateiname“) einlesen

Functions have to be in the working directory

# comment

>  x    # liefert den Wert von x  (Autoprinting without „print()“

**Objects**

Expresion = jeder Ausdruck

Object = jedes Data

Vector= contains only of objects of he same class (integer,numeric,character, complex, logical)

List= objects of several classes (eg. Integer, list, other list, character)

Vector ()

1L =1 als Integer (L behind the number)

Special numbers: Inf = infinity: 1/0 =Infinity  1/Inf=0

NaN = Not a Number  (eg 0/0 = NaN because is not defined)

Attributes: eg. Name, dimension (matrx, array), class, length, other usr-defined metadata)

attributes(objectname)

**Vectors**

c() = concatenating things together c(0.5, 0.6) c(T,F) true and false c(1+0i, 2+4i)

x<- vector(„numeric", length=10) initalised with 0 0 0 0 0 0 0 0 0 0 0 #

Vorsicht bei Anführungszeichen. Copy und Paste aus Word verändert deren Aussehen

as.numeric(x): Variable-classs converting into another

as.character(y\_n) y\_n <- as.numeric(y)

as.logical class(y\_n) liefert die Variablen-Klasse

x<-list(1,"a",3,4+3i) variablentypen quer durcheinander

**Matrizen**

m<-matrix(nrow=2, ncol=3)

dim (M) Dimension

m<-matrix(1:6,nrow=2, ncol=3) Es wird Spalte für Spalte befüllt

Es kann eine Matrix aus einem Vektor erstellt werden m<-c(1:10), dann dim(M)<-c(2,5)

Binding Column or Rows

cbind(x,y) x<-1:3 y<-8:10 es kommt eine neue Spalte dazu rbind genauso

**Factors**

For represent categorical data. Is like a integer vector with a lable for each integer Eg. 1=note sehr gut,…..6=ungeügend oder 1=männl. 2=weibl.

X<-factor(c("yes", "yes", "no", "yes", "no")

Automatisch: Levels no yes

table(x): no:2 yes:3 unclass(X) liefert 2 2 1 2 1 ( Werte in alphabetischer Reihenfolge).

Ausgabe als Spalten was was war Alphabetische Reihenfolge kann angepasst werden mit levels = c(„yes", „no") in der factor-Definition (also : X<-factor(c("yes", "yes", "no", "yes", "no"), levels = c("yes", "no"))

## Missing values NA = missing value (but they have a class) NaN= undefinde mathematical operation | is.na() is .nan()

**Data Frames**

Data- frame = special type of a list, each element hast he same length. Each element is a column with the same number of rows. Each row can be of diferent class (unlinke matrix)

Attribute: row.names (every raw has a name)

read by:   read.table() or read.csv

Convert into matrix:

data.matrix()

Create with: x<- data.frame(foo =1:4, bar = c(T,T,F,F))

nrow(x) ncol(x) Anzahl der Spalten

**Names for objects**

> x<-1:3

 names (x)

>NULL

> names(x)<- c("foo", "bar","norf")

> x

foo  bar norf

1    2    3

> names(x)

[1] "foo" "bar"  "norf"

**Names for Lists:**

x<- list(a01mb22,c=3)

x

$a 1

**Dimnames for matrices (für Spalten und Zeilen-bezeichnungen)**

m<-matrix(1:4, nrow02, ncol=2)

dimnames(m)<- list(c("a","b"), c("c","d"))

  c  d

a  1  3

b  2  4

**Read /Write Data**

read.table(file)

file: first line is header

sep (what is the separator)

colClasses(first column is numeric, second is string)

commen.char what character is  comment #?

skip (number of line to skip if necessery)

stringAsFactors (should Charakter be factor,?)

for small data: there is no need to specify the <- read.table("foo.txt")

read.css =read.table but the separator is a comma (instead of space)

initial <- read.table(database.txt, nrows=100)

read.table(file)

**Textual formats**

**Dump and dput**

y<-data.frame(a=1,b="a")

dput(y)

dput(x,file="y.R")

dget:new.y<-dget("y.R")

new.s

dump= dget but on more then one object

**Connection to the outside world**

 str(file)

 con<-file("foo.txt","r")

 data<-read.csv(con)

 close(con)

 i.e. create a line from a text file

 con<-gzfile("words.go")

 x<-readLines(con,10)

 x

**Read from URL:**

con <- url("https://www.uni-due.de%22, "r")

x<-readLines(con)

head(x)

**Subsets**

**[** can select even more objects (it will remein th esame class as in the main object

**[[** extracts a single elemetn of a list/frame, the class will not automaticaly by a list/frame

**$** extracts like [[ but by the name oft he object

> y<-1:6

> names(y)<-x

> y

a b c c d a

1 2 3 4 5 6

> y[3]

Auch eine Ausgabe einer Folge ist möglich

x[x>"a"] liefert alle Werte größer "a"

Logischer Vektor (WAHR/FALSCH)

> u<- x>"a"

> u

[1] FALSE TRUE TRUE TRUE TRUE FALSE

x[u] liefert dann nur die TRUE Werte

**Subset of a List**

x <- list (foo=1:4, bar=0.6, baz="hello") es sind 4 Integer und eine Reelle Zahl und ein String

[1] es ist eine Folge

[[1]] ist eine List (des gleichen Typs)

x$bar liefert 0.6 das gleiche wie x[["bar"]] Man muss sich nicht merken, an welcher Stelle die reelle Zahl war

x["bar"] liefert eine Liste (weil x per definitionem eine Liste war

Ausgabe der durchnummerierten Listen-Elemente: x[c(1,3)] liefert Element 1 und 3 (Folge und String)

[[ ]] - Operator kann auch mit indirekten Namen z.B. generisch erzeugenten umgehen:

name<-"foo"

x[[name]] –> liefert 1 2 3 4

**Untermengen**

> x<-matrix(1:6,2,3) erst die Anzahl der Zeilen,dann Spalten

> x[1,2] liefert [1] 3 (Erster Index ist Zeile , zweiter Spalte)

Obwohl es einfache eckige Klammer ist, liefert sie keine Matrix (wie die Obermenge, sondern das Element)

Will man eine 1x1 Matrix haben: x[1,2, drop=FALSE]

Genauso mit einer Zeile

x[1, , drop=FALSE] liefert dann eine 1x3-Matrix: 1 3 5 (statt eines Vektors)

**Teilsuche**

x$b liefert die Werte aller Elemente, deren Bezeichnug mit b anfängt

**Removing Missing Values**

x<-c(1,2,NA,4,NA,5)

bad<-is.na(x)

x[!bad] liefert 1,2,5

Vektroren zusammenfügen

good<-complete.cases(x,y) WICHTIG: gleiche Länge (BOOLEAN)

Man kann sie dennoch einzeln ausgeben, nur die die einen Wert haben

**Vektor-Operationen**

x<-1:4; y<-6:9

Vektor-Operationen x+y x-y x\*y x/y Koordiantenweise

Auch Vergleiche x<2

x+y> 8: FALSE TRUE TRUE TRUE

**Matrizen**

a<-matrix(1:4, 2,2); b<-matrix(rep(10,4),2,2)

alle Elemente gleich rep(10,4)

a\*b Elemten-weise Multiplikation

Matrix-Multipliakton a %\*% b

**Swirl**

When you are at the R prompt (>):

| – Typing skip() allows you to skip the current question.

| – Typing **play()** lets you experiment with R on your own; swirl will ignore what you do…

| – UNTIL you type **nxt()** which will regain swirl's attention.

| – Typing bye() causes swirl to exit. Your progress will be saved.

| – Typing main() returns you to swirl's main menu.

| – Typing info() displays these options again.

library(swirl)

 help.start()

? = Help i.e. ?c= what is c (but for signs use backtick(=accent aigu)) eg. `:`

^ \_ Potenz, sqrt()=Hoch2 abs()=Absolut

Vektor-Multiplikation: (3,5,7)\*3+100 wird übersetz ins (3,5,7)\*c(3,3,3)+c(100,100,100)

Variabelnnamen ergänzen lassen: erste zwei Buchstaben eintippen und dann <TAB>

**Swirl: Workspace and files**

getwd() = pwd

ls() = all the objects in your local workspace using ls().

list.files() = dir()

args(function) Welche Argumente kann eine Funktion annehmen

old.dir<-getwd() Verzeichnisnamen in einer Variable speichern

dir.create("NeuerOrdner") Neues Unterwerzeichnis erstellen im WorkingDirectory

setwd(AdresseDesDir-zB.Aus file.path()) =neues WorkingDirectory zuweisen

file.create("mytest.R") neue DAtei erstellen

file.exists("mytest.R") existiert eine Datei?

file.info("mytest.R") Infos über eine DAtei

file.info()$mtime Infos über bestimmte Argumente der Datei-Info

file.rename("vorhee","nachher") Umbennenen

file.remove('mytest.R')

file.copy("mytest2.R", "mytest3.R")

file.path() = relative path to the file

***Take nothing but results. Leave nothing but assumptions.***

*T​o get that I need an assignment token. But where to find it? There is nothing about it neither in the Coursera Help Center nor in the FAQs. The movie on Youtube is different to my dashboard.*

**Geniale Tricks**

> x <- c(17, 14, 4, 5, 13, 12, 10)

> x[x>10]<-4 setz alle Werte>10 im Vektor zu 4

[1] 4 4 4 5 4 4 10

Daten einlesen dateinameneu<- erad.csv(DAtensatz) -> rechtsObenFenster (Enviroment): Anklicken für Überblick über die Daten

data\_csv2<-read.csv(file.choose()) Datei im Prozess aussuchen lassen

**Swirl: Sequences of Numbers**

pi:10 liefert: 3.141593 4.141593 5.141593 6.141593 7.141593 8.141593 9.141 (aber nichts größer 10!)

auch runter zählen

**Folgen mit Schrittweitenund vordefineirter Länge**

seq(1,20, by=0.5) mit Schrittweite 0.5

seq(5, 10, length=30) 30 Werte in dem INtervall [5,10]

length(name der Variable) Länge der Folge rausfinden

seq\_along(my\_seq) = eine Folge erzeugen, die gleich der Länge einer anderen Folge ist

rep(0, times=40) = Replicate the same number

Auch ganze Vektoren : rep(c(1,3,5), times=10) 10mal der gleiche Vektor

oder rep(c(0,1,2), each=10) (wiederholt erste zahl 10mal dann die zweite)

**Swirl: Vectors**

tf <- num\_vect < 1 (alle Werte des Vektors <1) liefert einen logischen Vektor für jede Zeile

**Logische Operatoren:** z.B. >= == oder != (negation) A|B A&B !A

**String-Operatoren:**

paste(my\_char, collapse = " ") verbidnet die Zeilen des Vektors zu einem String, mit " " als Abstand

paste("Hello", "world!", sep = " ") verbindet mehrere Vektoren mit " " als ABstand

paste(1:3, c("X", "Y", "Z"), sep="") verbident die Vektoren zeilenweise "1X" "2Y" "3Z" (Vekroten mit ungleichen Längen werden durch den zu kurzen Veltor ergänzt)

LETTERS= vorconfig Variabe mit 26 Buchstaben des Alphabets

paste(LETTERS, 1:4, sep = "-") ergänzt den 1:4 Vektor auf 26 Zeichen.Zaheln werden zu Buchstaben umgewandelt

**Swirl: Missing values**

rnorm(100) = 100 Normalverteilte Werte um die 0 (mit STABW =1)

sample(c(y, z), 100) Stichprobe ziehen (Stichprobemgröße 100)

is.na() Prüfung ob Wert=NA ist. Liefert einen VEktor (T/F).

Bei Abfrage x[is.na(x)] liefert die Werte wo True steht (also "NA"), Bei Negation !is.na den echten Wert

sum() = summiert die Werte des Vektors (man kann auch alle TRUES zusammen rechnen (als 1))

**Swirl: Subsetting Vectors**

**KfvawvuZonc09WNo**

**Index-Vectors:**

Logical, Positive Integer, Negative Integer, Character

x[1:10] die ersten 10 Elemente eines Vektors

x[!is.na(x)] liefert alle echten (d.h. nicht-NA) Werte

x[!is.na(x) & x > 0] liefert alle positiven Werte der Stichprobe

x[c(3,5,7)] Ausgabe nur der 3., 5. und 7. Zeile eines Vektors x[c(-2,-10)] Ausgabe aller Werte bis auf 2 und 10 (geht auch: x[-c(2,10)])

vect <- c(foo = 11, bar = 2, norf = NA). Werte haben Bezeichnungen

names(vect2) <- c

vect[c("foo", "bar")] liefer eine Auswahl aus dem Ursprungsvektor

identical(vect, vect2) sind 2 Vektoren gleich

**Swirl: Matrices and data frames**

* single class of data vs many different classes

dim(my\_vector) = Dimension des Objekts

length(my\_vector) = Länge

Man kann die Dimension aber zuweisen: dim(my\_vector) <-c(4,5) und mit attributes(my\_vector) überprüfen. DAnn ist der Vektor eine Matrix | class(my\_vector) Überprüfung der Klasse | cbind() Cobine Columns

data.frame() = WIe Matrix aber unterschiedliche DAtentypen sind erlaubt

R Programming Week2

--- title: "CourseraR-2Teil-Woche2" author: "jm" date: "25 6 2021" output: html\_document editor\_options: markdown: wrap: 72 ---

**If-Else**

if(condition) {do something} else { }

y<- if(x<3){10}else{0}

**For and While (and Repeat), Next**

for(i in 1:10)

{ print(i) } # (i from sequence OR vector)

for(letter in c("a","b","c","d")) { print(letter) }

**Verschachtelt**

 x<-matrix(1:6,2,3);

 for(i in seq\_len(nrow(x))) {

                     for (j in seq\_len(ncol(x))) {

                     print(x[i,j])

                     }

 }

**While**

count<-0; while (count<10) {print(count); count<-count+1}

**Random walk (Gibt an nach wie vielen Teilstrecken beendet wurde)**

for(i in 1:10){

  counter<-0

  z<-5

  while(z >=3 && z<=10){

    #print(z)

    coin<-rbinom(1,1,0.5)

    if (coin==1) {

      z<-z+1

    } else {

          z<-z-1

            }

  counter<-counter+1

  }

  print(counter)

  }

**Repeat**

repeat { Kommandos bis if(Abfrage) {break} else{mach weiter} }

**Next**

for examle to skip some iterations for(i in 1:100) { if (i<20) {next} <other commands for i>20> }

**Functions**

above\_ten <- function(x) { # soll aus dem Vektor Zahlen größer 10 ausgeben

use<- x>10 # Logischer Vektor mit Zahlen >10

x[use] # gibt nur die Werte des Vktors x raus, die TRUE sind

} # letzter Wert wird an die Funktion zurückgegeben

 y<-1:20

above\_ten(y)

**Zwei Variablen**

above<-function (x,n) {use<- x>n; x[use] }

**Default Werte festlegen**

above<-function (x,n**=10**) {use<- x>n; x[use] }

**Matrizen-Operationen**

Mittelwert jeder Spalte

columnmean <- function(m, removeNA=TRUE) { # NA werden aus den Zeilen entfernt

  means<- numeric(ncol(m)) # Neuer Vektor der Mittelwerte jeder Spalte

  for(i in 1:ncol(m)) {

      means[i]<-mean(m[,i], na.rm=remove.NA ) # in jede Zelle des Vektors wird der Mittelwert geschrieben. NA werden entfernt

  }

means

}

Mann kann aber die NA auch behalten wenn default mit FALSE überschrieben wird ( columenmean(m,FALSE))

**Vordefinerte Funktionen**

sd() = StdAbweichung

sd(mydata, rm.na=FALSE) # rechnet mit NA-Werten

rm.na=FALSE # NA werte nicht entfernen

args(Objekt) = Liste der Argumente

eg. für LinearModell-Function: args(lm): function (formula, data, subset, weights, na.action, method = "qr", model = TRUE, x = FALSE, y = FALSE, qr = TRUE, singular.ok = TRUE,

contrasts = NULL, offset, …)

Es kann auch … (3Punkte) eingegeben werden, wenn man nicht alle Argumente einzeln eingeben möchte

eg.: paste(… , sep=" ") wobei die Dreipunkte für beliebig viele Argumente zu Anfang stehen. paste("a", "b", sep=":")

eg. cat:(…, file=" ", sep=" ")

**Scoping**

Enviroment = collection of (symbols and values) pairs, i.e. x und 7.2

search() = Which packages are installed

c und c() können gleichzeitig existieren

lexical scoping = static scoping vs dybmic scoping

global.enviroment = my workspace

closure = function and enviroment#

When calling a function it is searches in the global enviroment, then in the parent enviroemte and next parent: top-level enviroment

function in function: when a variable is in the main function it will be take into the sub-function

lexical: will take the value from the enviroment, where the value was defined. dynamic: will take the value from the enviroment where it is called (Calling enviroment)

**Data and times**

data: days since 1970-01-01

time: seconds since 1970-01-01

POSIXct = integer

POSIXlt = list ´with all the other data like weeek, quarter,

Sys.Date()

x<- Sys.time()

p<- as.POSIXlt(x)

p$sec

weekdays(p) # months() quarters()

class(p) -> Welches Klasse ist das?

names(unclass(p)) wie wir die Variabel intern gespreichert?

unclass = in welche Werte wird der Wert Time aufgesplittet?

[1] "sec" "min" "hour" "mday" "mon" "year" "wday" "yday" "isdst" "zone" "gmtoff"

p$min

str(unclasst1)) = die Werte kompakter ausgeben lassen

**Converting strings into data-strings:**

datastring <- c("Januar 10, 2020 10:40", "Dezember 9, 2021 9:10") # als deutsche Wörter!

x<-strptime(datastring, " %B %d %Y %H:%M") #strptime() converts character vectors to POSIXlt

y<-as.Date("2012-03-01")

x<-as.POSIXct(y)

difftime(t1,t2, units='days') = Differenz zw 2 Zeiten

Package for more time-functions: lubridate package by Hadley Wickham

**SWIRL: Logical functions**

TRUE && c(TRUE, FALSE, FALSE) # nur die erste zeile wird verglichen

TRUE & c(TRUE, FALSE, FALSE) # alle zeilen werden verglichen (ergibt Vektor)

Genauso | oder ||

isTRUE()

identical('twins', 'twins')

xor(5 == 6, !FALSE)\

sample(10) = Zufallsmenge ohne Zurücklegen

x<-sample(10)

x<5 liefert einen TRUE-FALSE-Vektor

which(c(TRUE, FALSE, TRUE)) liefert c(1,3)

which(x>7) liefert die Lfd-Nr (Index) der Werte >7

any() = TRUE if one or more of the elements in the logical vector is TRUE

all() = TRUE if every element in the logical vector is TRUE.

**SWIRL: Functions**

"passing arguments" to function.

**John Chambers, the creator of R**:

To understand computations in R, two slogans are helpful:

1. Everything that exists is an object.

2. Everything that happens is a function call.

Source code for any function: type the function name without any arguments or parentheses

Remainder = Division mit Rest (Modulo): 7 %% 4 = 3

the order of the arguments is unimportant when designating the arguments by name. i.e. remainder(divisor = 11, num = 5)

Partially match arguments: remainder(4, div = 2) # instead of divisor

**Argumente** der Funktion abfragen: args(remainder)

**Function in function:**

evaluate <- function(func, dat){ # Es wird an func eine vordefinerte Funktion übergeben

  func(dat) # und an diese Funktion Daten, die diese benötigt

}

**Verketten:** paste("Programming", "is", "fun!")

Elipses … functions = This is a strict rule in R programming: all arguments after an ellipses must have default values.

mytelegram <- function(…){ paste("START", …, "STOP")

What is in … ? x<-list(…)

mad\_libs <- function(…){

  args <- list(…)

  place <- args[["place"]]

  adjective <- args[["adjective"]]

  noun <- args [["noun"]]

  paste("News from", place, "today where", adjective, "students took to the streets in protest of the new", noun, "being installed on campus.")

}

User-defined binary operators have the following syntax: %[whatever]%

"%mult\_add\_one%" <- function(left, right){ # Notice the quotation marks!

  left \* right + 1

}

**Funktiosaufruf:**

source("pollutantmean.R")