Einführung in R - Housekeeping

Jan-Philipp Kolb

13 Juni, 2019

Pakete automatisch installieren

- Mit `installed.packages()` kann ich herausfinden, welche Pakete installiert sind.
- Bei mir sind es momentan 317 Pakete.

```r
my_packages <- installed.packages()
mypack <- my_packages[,"Package"]
```

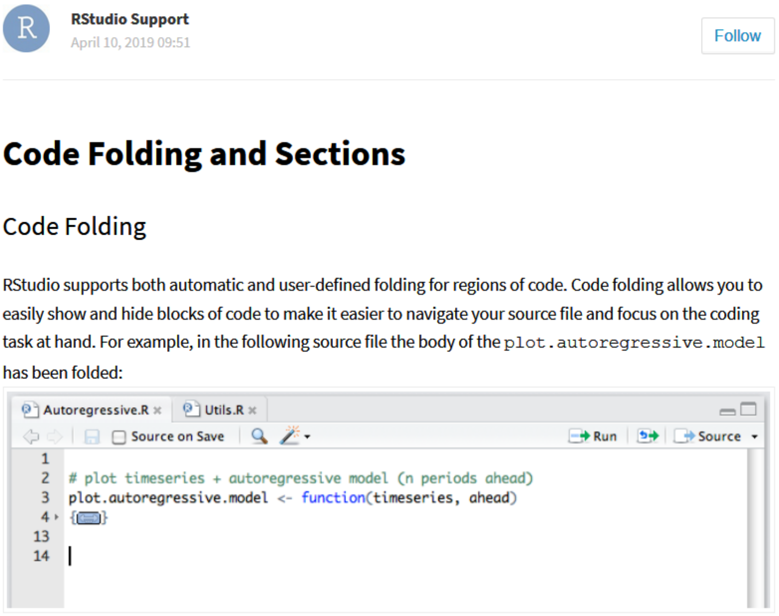
```r
my_packages[,"Package"]
```

```
##                 abind               acepack                   AER
##               "abind"             "acepack"                 "AER"
##           AmesHousing              antiword               aplpack
##         "AmesHousing"            "antiword"             "aplpack"
##                   arm               askpass            assertthat
##                 "arm"             "askpass"          "assertthat"
##             backports                  base             base64enc
##           "backports"                "base"           "base64enc"
##            bayestestR              beanplot                    BH
##          "bayestestR"            "beanplot"                  "BH"
##                 bindr              bindrcpp                bitops
##               "bindr"            "bindrcpp"              "bitops"
```

Fehlende Pakete installieren

```r
packlist <- c("ggplot2", "Rcpp")
new.packages <- packlist[!(packlist %in% mypack)]
if(length(new.packages)) install.packages(new.packages)
```

Codeblöcke zusammenklappen



RStudio Support
April 10, 2019 09:51

Code Folding and Sections

Code Folding

RStudio supports both automatic and user-defined folding for regions of code. Code folding allows you to easily show and hide blocks of code to make it easier to navigate your source file and focus on the coding task at hand. For example, in the following source file the body of the `plot.autoregressive.model` has been folded:

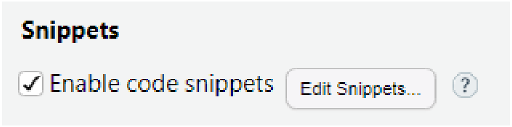
```

# plot timeseries + autoregressive model (n periods ahead)
plot.autoregressive.model <- function(timeseries, ahead)
{⟺}

```

Die Funktion source

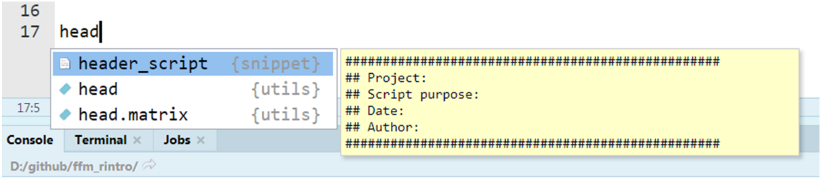
```r
source("../rcode/load_packages.R")
```

Snippets - um automatisch Codekopf einzufügen

Tools > Golbal Options

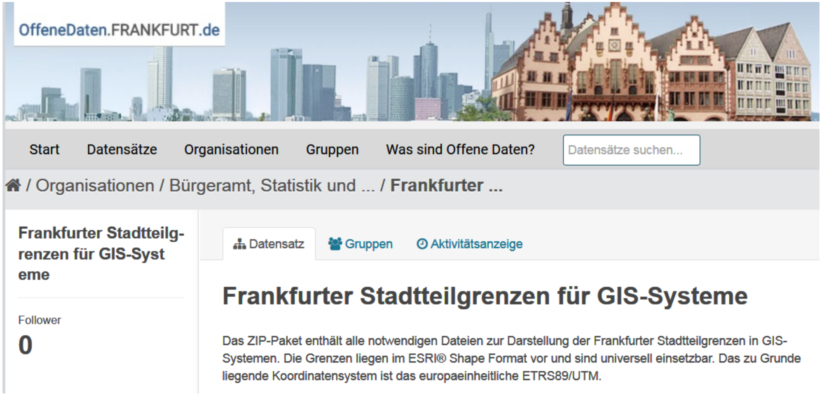

Snippets

✓ Enable code snippets


```
head
```

```
###################################################
## Project:
## Script purpose:
## Date:
## Author:
###################################################
```

Shapefiles herunterladen



Frankfurter Stadtteilg-
renzen für GIS-Syst
eme

Follower

0

Frankfurter Stadtteilgrenzen für GIS-Systeme

Das ZIP-Paket enthält alle notwendigen Dateien zur Darstellung der Frankfurter Stadtteilgrenzen in GIS-Systemen. Die Grenzen liegen im ESRI® Shape Format vor und sind universell einsetzbar. Das zu Grunde liegende Koordinatensystem ist das europaeinheitliche ETRS89/UTM.

Shapefiles importieren

```
ffm_shp <- rgdal::readOGR("../data/Stadtteile_Frankfurt_am_Main.
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "D:\github\ffm_rintro\data\Stadtteile_Frankfurt_am_Ma
## with 46 features
## It has 2 fields
```

Der Datenslot

```r
head(ffm_shp@data)
```

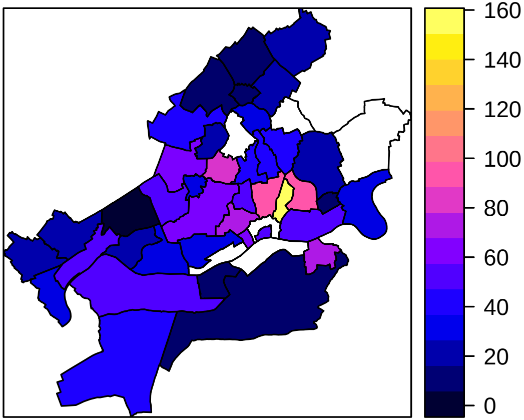
```
##   STTLNR         STTLNAME
## 0      1         Altstadt
## 1      2       Innenstadt
## 2      3 Bahnhofsviertel
## 3      4     Westend-SÃ¼d
## 4      5     Westend-Nord
## 5      6     Nordend-West
```

Inhaltliche Daten hinzufügen

```
dat <- read.csv2("../data/bauenwohnen.csv")
ffm_shp@data$Einwohnerdichte <-
  dat$Wohnumfeld...öffentlicher.Raum.Einwohnerdichte.je.ha.2012
```

Eine thematische Karte plotten

```
sp::spplot(ffm_shp,"Einwohnerdichte")
```



Thematische Karte mit tmap

```
install.packages("tmap")
```

```
tmap::qtm(ffm_shp,"Einwohnerdichte")
```

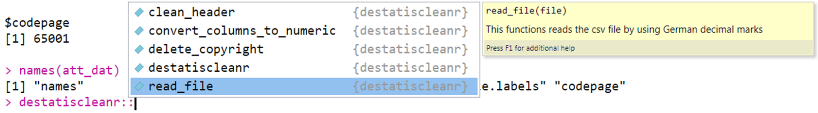
Belabelter SPSS Datensatz

```
data_path <- "D:/gitlab/IntroDataAnalysis/data/"
gpdat <- foreign::read.spss(paste0(data_path,
            "ZA5666_v1-0-0.sav"),to.data.frame=TRUE)
att_dat <- attributes(gpdat)
names(att_dat)
att_dat$variable.labels
```

Destatis Daten

```r
devtools::install_github("cutterkom/destatiscleanr")
```

```r
library(destatiscleanr)
```



```
$codepage
[1] 65001

> names(att_dat)
[1] "names"                                         e.labels" "codepage"
> destatiscleanr::
```

```
clean_header                 {destatiscleanr}
convert_columns_to_numeric   {destatiscleanr}
delete_copyright             {destatiscleanr}
destatiscleanr               {destatiscleanr}
read_file                    {destatiscleanr}
```

read_file(file)

This functions reads the csv file by using German decimal marks

Press F1 for additional help