Import von Webdaten

Jan-Philipp Kolb

8 Mai 2017

1. Import von JSON Dateien
2. Import von XML Dateien

Import von JSON Dateien

JavaScript Object Notation (JSON)

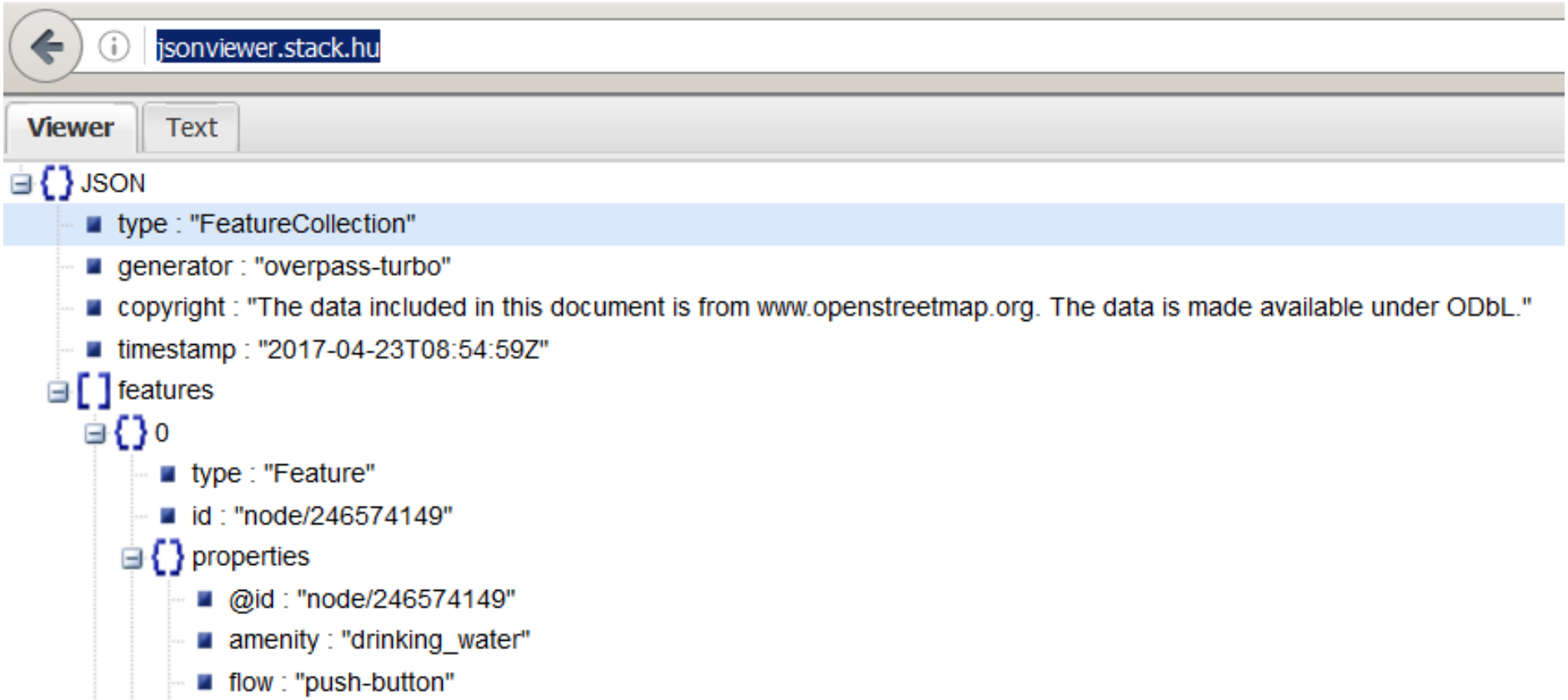
- Jedes gültige JSON-Dokument soll ein gültiges JavaScript sein
- JSON wird zur Übertragung und zum Speichern von strukturierten Daten eingesetzt
- Insbesondere bei Webanwendungen und mobilen Apps wird es in Verbindung mit JavaScript, Ajax oder WebSockets zum Transfer von Daten zwischen dem Client und dem Server häufig genutzt.

```
{
  "Herausgeber": "Xema",
  "Nummer": "1234-5678-9012-3456",
  "Deckung": 2e+6,
  "Waehrung": "EURO",
  "Inhaber":
  {
    "Name": "Mustermann",
    "Vorname": "Max",
```

Das GeoJSON Format

- GeoJSON ist ein offenes Format um geografische Daten nach der Simple-Feature-Access-Spezifikation zu repräsentieren.
- Dafür wird die JavaScript Object Notation verwendet.

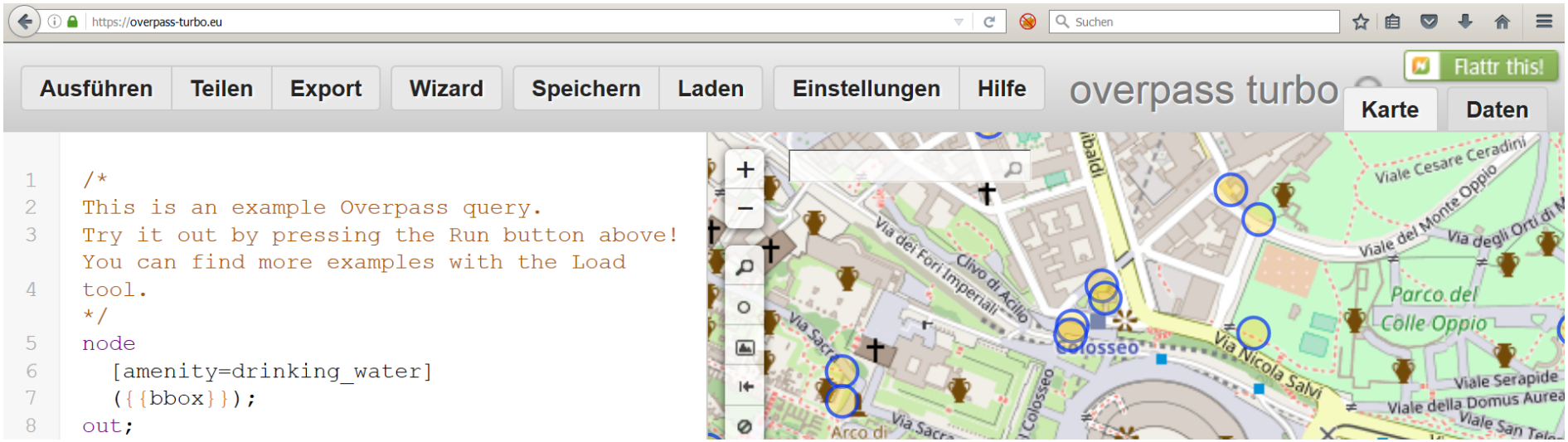
Die Struktur der Daten kann man sich mit einem JSON Viewer anschauen



Download von Beispieldaten

- Overpass Turbo kann verwendet werden um Beispieldaten zu bekommen

`https://overpass-turbo.eu/`



Exkurs OpenStreetMap Daten

- Auf Overpass Turbo können Daten für Map Features exportiert werden
- Eine Liste der erhältlichen Map Features gibt es auf `http://wiki.openstreetmap.org/wiki/DE:Map_Features`

Schlüssel	Wert	Element	Kommentar	Darstellung	Foto
			Verpflegung, Einkehr		
amenity	bar		**Bar, Nachtlokal**. Es werden hauptsächlich alkoholische Getränke serviert. Siehe auch Beschreibung von `amenity=bar` und `amenity=pub` zur Unterscheidung von Bar und Pub (Kneipe).		
amenity	bbq		**Grillplatz**. Kann mit `fuel=*` (wood/gas/electric) kombiniert werden.		
amenity	biergarten		**Biergarten**		

Beispiele für GeoJSON

Einfache Geometrien

Typ	Beispiel	
Point (Punkt)		`{ "type": "Point", "coordinates": [30, 10] }`
LineString (Linie)		`{ "type": "LineString", "coordinates": [ [30, 10], [10, 30], [40, 40] ] }`
Polygon		`{ "type": "Polygon", "coordinates": [ [[30, 10], [40, 40], [20, 40], [10, 20], [30, 10]] ] }`
		`{ "type": "Polygon", "coordinates": [ [[35, 10], [45, 45], [15, 40], [10, 20], [35, 10]], [[20, 30], [35, 35], [30, 20], [20, 30]] ] }`

JSON importieren

```
library("jsonlite")
DRINKWATER <- fromJSON("data/RomDrinkingWater.geojson")
```

```
names(DRINKWATER)
```

```
## [1] "type"      "generator" "copyright" "timestamp" "featur
```

Die Daten anschauen

```
head(DRINKWATER$features)
```

```
##      type                 id properties.@id properties.amenity
## 1 Feature node/246574149 node/246574149     drinking_water
## 2 Feature node/246574150 node/246574150     drinking_water
## 3 Feature node/246574151 node/246574151     drinking_water
## 4 Feature node/248743324 node/248743324     drinking_water
## 5 Feature node/251773348 node/251773348     drinking_water
## 6 Feature node/251773551 node/251773551     drinking_water
##   properties.type properties.name properties.name:fr proper
## 1          nasone            <NA>              <NA>
## 2            <NA>            <NA>              <NA>
## 3            <NA>            <NA>              <NA>
## 4            <NA>            <NA>              <NA>
## 5          nasone            <NA>              <NA>
## 6            <NA>   Acqua Marcia      Eau potable
```

Github JSON Daten

- Es lassen sich auch Dinge aus dem Web auslesen:

```
my_repos <- fromJSON("https://api.github.com/users/japhilko/re
```

```
names(my_repos)
```

```
##  [1] "id"                "name"              "full_name"
##  [4] "owner"             "private"           "html_url"
##  [7] "description"       "fork"              "url"
## [10] "forks_url"         "keys_url"          "collaborators
## [13] "teams_url"         "hooks_url"         "issue_events_
## [16] "events_url"        "assignees_url"     "branches_url"
## [19] "tags_url"          "blobs_url"         "git_tags_url"
## [22] "git_refs_url"      "trees_url"         "statuses_url"
## [25] "languages_url"     "stargazers_url"    "contributors_
## [28] "subscribers_url"   "subscription_url"  "commits_url"
```

```
library(jsonlite)
res <- fromJSON('http://ergast.com/api/f1/2004/1/results.json'
drivers <- res$MRData$RaceTable$Races$Results[[1]]$Driver
colnames(drivers)
```

```
## [1] "driverId"        "code"            "url"             '
## [5] "familyName"      "dateOfBirth"     "nationality"     '
```

```
article_key <- "&api-key=c2fede7bd9aea57c898f538e5ec0a1ee:6:68
url <- "http://api.nytimes.com/svc/search/v2/articlesearch.jso
req <- fromJSON(paste0(url, article_key))
articles <- req$response$docs
colnames(articles)
```

```
##  [1] "web_url"         "snippet"         "lead_paragrap
##  [4] "abstract"        "print_page"      "blog"
##  [7] "source"          "multimedia"      "headline"
## [10] "keywords"        "pub_date"        "document_type
```

Import von XML Dateien

Das XML Paket

```
library(XML)
citation("XML")
```

```
##
## To cite package 'XML' in publications use:
##
##   Duncan Temple Lang and the CRAN Team (2016). XML: Tools f
##   Parsing and Generating XML Within R and S-Plus. R package
##   version 3.98-1.5. https://CRAN.R-project.org/package=XML
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {XML: Tools for Parsing and Generating XML Wit
##     author = {Duncan Temple Lang and the CRAN Team},
##     year = {2016},
```

Erstes Beispiel

```
url <- "http://api.openstreetmap.org/api/0.6/
relation/62422"
```

```
library(xml2)
BE <- xmlParse(url)
```


```
- <osm version="0.6" generator="CGImap 0.4.0 (19884 thorn-03.openstreetmap.org)" copyright="OpenStreetMap and contributors" attribution="http://www.openstreetmap.org/copyright"
  license="http://opendatacommons.org/licenses/odbl/1-0/">
  - <relation id="62422" visible="true" version="209" changeset="36072269" timestamp="2015-12-20T19:49:52Z" user="tbicr" uid="278800">
      <member type="node" ref="240109189" role="admin_centre"/>
      <member type="way" ref="50291800" role="outer"/>
      <member type="way" ref="77913336" role="outer"/>
      <member type="way" ref="315222039" role="outer"/>
      <member type="way" ref="77487568" role="outer"/>
      <member type="way" ref="315222038" role="outer"/>
      <member type="way" ref="98035898" role="outer"/>
      <member type="way" ref="77501737" role="outer"/>
```

Das XML analysieren

- Tobi Bosede - Working with XML Data in R

```
xmltop = xmlRoot(BE)
class(xmltop)
```

```
## [1] "XMLInternalElementNode" "XMLInternalNode"
## [3] "XMLAbstractNode"
```

```
xmlSize(xmltop)
```

```
## [1] 1
```

```
xmlSize(xmltop[[1]])
```

```
## [1] 326
```

Nutzung von Xpath

Xpath, the XML Path Language, is a query language for selecting nodes from an XML document.

```
xpathApply(BE,"//tag[@k = 'source:population']")
```

```
## [[1]]
## <tag k="source:population" v="http://www.statistik-berlin-b
##
## attr(,"class")
## [1] "XMLNodeSet"
```

Node parsen

```
url2 <- "http://api.openstreetmap.org/api/0.6/node/2923760808"
RennesBa <- xmlParse(url2)
```

Way parsen

```
url3 <- "http://api.openstreetmap.org/api/0.6/way/72799743"
MadCalle <- xmlParse(url3)
```

Mehr Beispiele, wie man mit XML Daten umgeht:

- Daten aus XML extrahieren

`http://www.stat.berkeley.edu/~statcur/Workshop2/Presentations/XML.pdf`

- Duncan Temple Lang - A Short Introduction to the XML package for R

`http://www.omegahat.net/RSXML/shortIntro.pdf`

Noch mehr Informationen

- Web Daten manipulieren

`http://www.di.fc.ul.pt/~jpn/r/web/index.html#parsing-xml`

- Tutorial zu xquery

`http://www.w3schools.com/xml/xquery_intro.asp`

- R und das Web (für Anfänger), Teil II: XML und R

`http://giventhedata.blogspot.de/2012/06/r-and-web-for-beginners-part-ii-xml-in.html`

- String Manipulation

`http://gastonsanchez.com/Handling_and_Processing_Strings_in_R.pdf`

Referenzen

```
citation("XML")
```

```
##
## To cite package 'XML' in publications use:
##
##   Duncan Temple Lang and the CRAN Team (2016). XML: Tools f
##   Parsing and Generating XML Within R and S-Plus. R package
##   version 3.98-1.5. https://CRAN.R-project.org/package=XML
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {XML: Tools for Parsing and Generating XML Wit
##     author = {Duncan Temple Lang and the CRAN Team},
##     year = {2016},
##     note = {R package version 3.98-1.5},
```

Links

- XML parsen - Stackoverflow
- Processing of GeoJson data in R