

B3 Die OSM main api

Jan-Philipp Kolb

23 Oktober 2018

OSM Ausschnitte herunterladen

<www.openstreetmap.org/export>

The screenshot shows the OpenStreetMap website's export interface. At the top, there's a navigation bar with 'OpenStreetMap', 'Edit', 'History', and 'Export' buttons. Below this is a search bar and a user profile dropdown. The main area features a map of a city grid with a green bounding box highlighting a specific area. To the left of the map, there's an 'Export' section with input fields for coordinates (39.95159, -75.17569, 39.94715, -75.16556) and a 'Export' button. Below the map, there's a 'Licence' section with text about the Open Data Commons Open Database License (ODbL) and a 'Download' button. At the bottom, there's a 'Footnote' section with links to 'Overpass API', 'Planet OSM', 'Geofabrik Downloads', 'Web Services', and 'Other Sources'.

OpenStreetMap Edit History Export

Search Where are you? Go

Export

39.95159 -75.17569 39.94715 -75.16556

Licence

OpenStreetMap data is licensed under the [Open Data Commons Open Database License \(ODbL\)](#).

Download

If the above export fails, please consider using one of the sources listed below:

- [Overpass API](#)
Download this bounding box from a mirror of the OpenStreetMap database
- [Planet OSM](#)
Regularly-updated copies of the complete OpenStreetMap database
- [Geofabrik Downloads](#)
Regularly-updated extracts of continents, countries, and selected cities
- [Web Services](#)
Extracts for major world cities and their surrounding areas
- [Other Sources](#)
Additional sources listed on the OpenStreetMap wiki

Figure 1: osm export

Das R-Paket XML - Gaston Sanchez

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

Gaston Sanchez - Dataflow



Getting Data from the Web with R Part 4: Parsing XML/HTML Content

Gaston Sanchez

April-May 2014

Content licensed under [CC BY-NC-SA 4.0](#)

Figure 2: Gaston Sanchez - Webdaten bekommen

Funktionen im XML Paket

Function	Description
<code>xmlName()</code>	name of the node
<code>xmlSize()</code>	number of subnodes
<code>xmlAttrs()</code>	named character vector of all attributes
<code>xmlGetAttr()</code>	value of a single attribute
<code>xmlValue()</code>	contents of a leaf node
<code>xmlParent()</code>	name of parent node
<code>xmlAncestors()</code>	name of ancestor nodes
<code>getSibling()</code>	siblings to the right or to the left
<code>xmlNamespace()</code>	the namespace (if there's one)

Einzelne Objekte finden

<www.openstreetmap.org/export>

OpenStreetMap [Bearbeiten](#) [Chronik](#) [Export](#) [GPS-Tracks](#) [Benutzer-Blogs](#) [Urheberrecht](#) [Hilfe](#) [Über](#) [Anmelden](#) [Registrieren](#)

Suchen [Wie bin ich?](#) [Los](#) [de](#)

Relation: Berlin (62422) ✕

Reparatur Admin- und PLZ-Grenze Zehlendorf/
Nikolassee

Bearbeitet vor etwa ein Monat von [streckenruder](#)
Version #217 Änderungssatz #44753545

Attribute

ISO3166-2	DE-BE
TMC cid_56.tabcd_1: Class	Area
TMC cid_56.tabcd_1: LCLversion	12.0
TMC cid_56.tabcd_1: LocationCode	266
admin_level	4
alt_name:vi	BKc-lin
boundary	administrative
capital	yes
contact facebook	http://www.facebook.com/Berlin
contact website	http://www.berlin.de
de:amtlicher_gemeindeschlüssel	11000000
de:place	city
de:place:note	Kreisfreie Stadt
de:regionalschlüssel	110000000000
geographical_region	Barrim, Berliner Umland, Teltow/Naue ener Platte

Beispiel: administrative Grenzen Berlin

Administrative Grenzen für Deutschland

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

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

```
BE <- xmlParse("../data/62422.xml")
```

```
<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"/>
```

Figure 4: Administrative Grenzen Berlin

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] 337
```


Nutzung von Xpath

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

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

```
## [[1]]  
## <tag k="population" v="3440441"/>  
##  
## attr(,"class")  
## [1] "XMLNodeSet"
```

Quelle für die Bevölkerungsgröße

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

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

```
## <tag k="source:population" v="http://www.statistik-berlin-b"
```

```
##
```

```
## attr(,"class")
```

```
## [1] "XMLNodeSet"
```

-Statistik Berlin Brandenburg

Etwas überraschend:

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

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

```
## <tag k="name:ta" v="<U+0BAA><U+0BC6><U+0BB0><U+0BCD><U+0BB2
```

```
##
```

```
## attr(,"class")
```

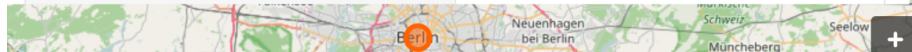
```
## [1] "XMLNodeSet"
```



OpenStreetMap



name:sw	Berlin
name:szl	Berlin
name:ta	பெர்லின்
name:te	ਬਰਲਿਨ
name:tet	Berlin

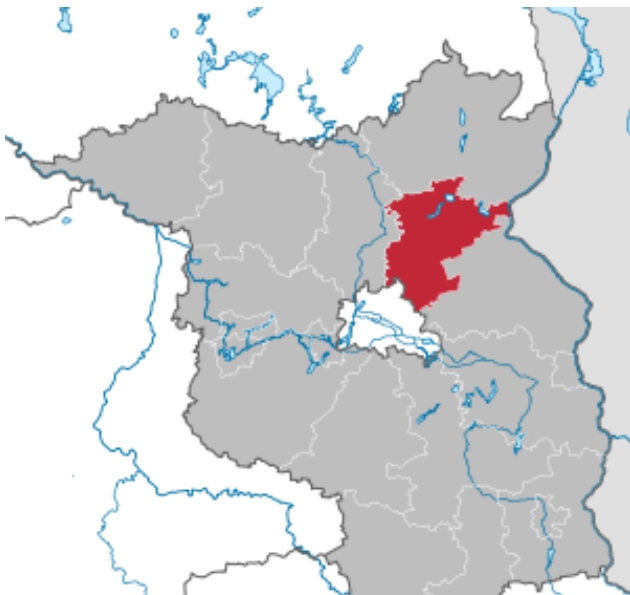


Geographische Region

```
region <- xpathApply(BE,  
  "//tag[@k = 'geographical_region']")  
# regular expressions  
region[[1]]
```

```
## <tag k="geographical_region" v="Barnim;Berliner Urstromtal;  
  
<tag k="geographical_region"  
  v="Barnim;Berliner Urstromtal;  
  Teltow;Nauener Platte"/>
```

Landkreis



Weiteres Beispiel

```
url2<-"http://api.openstreetmap.org/api/0.6/node/25113879"  
obj2<-xmlParse(url2)  
obj_amenity<-xpathApply(obj2,"//tag[@k = 'amenity']")[[1]]  
obj_amenity
```

```
## <tag k="amenity" v="university"/>
```

Wikipedia Artikel

```
xpathApply(obj2,"//tag[@k = 'wikipedia']")[[1]]
```

```
## <tag k="wikipedia" v="de:Universität Mannheim"/>
```

```
xpathApply(obj2,"//tag[@k = 'wheelchair']")[[1]]
```

```
xpathApply(obj2,"//tag[@k = 'name']")[[1]]
```

Das C und das A

```
url3<-"http://api.openstreetmap.org/api/0.6/node/303550876"  
obj3 <- xmlParse(url3)  
xpathApply(obj3,"//tag[@k = 'opening_hours']")[[1]]
```

```
## <tag k="opening_hours" v="Mo-Sa 09:00-20:00; Su,PH off"/>
```


Hin und weg

```
url4<-"http://api.openstreetmap.org/api/0.6/node/25439439"  
obj4 <- xmlParse(url4)  
xpathApply(obj4,"//tag[@k = 'railway:station_category']")[[1]]
```

```
## <tag k="railway:station_category" v="2"/>
```

• Wikipedia Artikel Bahnhofskategorien

Stufe	Bahnsteigkanten	Bahnsteiglänge	Reisende/Tag	Zughalte/Tag
6	1	bis 90 m	bis 49	bis 10
5	2	> 90 bis 140 m	50 bis 299	11 bis 50
4	3 bis 4	> 140 bis 170 m	300 bis 999	51 bis 100
3	5 bis 9	> 170 bis 210 m	1000 bis 9999	101 bis 500
2	10 bis 14	> 210 bis 280 m	10.000 bis 49.999	501 bis 1000
1	ab 15	> 280 m	ab 50.000	ab 1001

Prozent	Kategorie
> 90 %	1
> 80 bis 90 %	2
> 60 bis 80 %	3
> 50 bis 60 %	4
> 40 bis 50 %	5
> 25 bis 40 %	6
bis 25 %	7

Exkurs: Bahnhofskategorien

- rvest: Easily Harvest (Scrape) Web Pages

```
library(rvest)
```

```
## Loading required package: xml2
```

```
##
```

```
## Attaching package: 'rvest'
```

```
## The following object is masked from 'package:XML':
```

```
##
```

```
##      xml
```

```
bhfkat<-read_html(  
  "https://de.wikipedia.org/wiki/Bahnhofskategorie")  
df_html_bhfkat<-html_table(  
  html_nodes(bhfkat, "table")[[2]],fill = TRUE)
```

Bahnhofskategorien Übersicht

Stufe	Bahnsteigkanten	Bahnsteiglänge[Anm 1]	Reisende/Tag
(0)	—	—	—
1	01	> 000 bis 090 m	00.000 bis 00.049
2	02	> 090 bis 140 m	00.050 bis 00.299
3	03 bis 04	> 140 bis 170 m	00.300 bis 0.0999
4	05 bis 09	> 170 bis 210 m	01.000 bis 09.999
5	10 bis 14	> 210 bis 280 m	10.000 bis 49.999
6	00i ab 15	> 280 m bis 000	000000 ab 50.000
Gewichtung	20 %	20 %	20 %

Nur fliegen ist schöner

```
url5<-"http://api.openstreetmap.org/api/0.6/way/162149882"  
obj5<-xmlParse(url5)  
xpathApply(obj5,"//tag[@k = 'name']")[[1]]
```

```
## <tag k="name" v="City-Airport Mannheim"/>
```

```
xpathApply(obj5,"//tag[@k = 'website']")[[1]]
```

```
## <tag k="website" v="http://www.flugplatz-mannheim.de/" />
```

```
xpathApply(obj5,"//tag[@k = 'iata']")[[1]]
```

```
## <tag k="iata" v="MHG"/>
```

Das Paket osmar benutzen

```
library("osmar")
```

```
## Loading required package: RCurl
```

```
## Loading required package: bitops
```

```
## Loading required package: geosphere
```

```
##
```

```
## Attaching package: 'osmar'
```

```
## The following object is masked from 'package:utils':
```

```
##
```

```
##      find
```

```
node_ <- xmlParse("../data/162149882.xml")
```

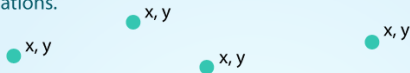
```
node_osmar <- as_osmar(node_)
```

```
node_osmar
```

Drei Typen von Vektorobjekten

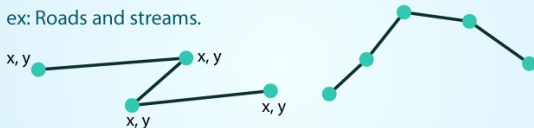
POINTS: Individual x, y locations.

ex: Center point of plot locations, tower locations, sampling locations.



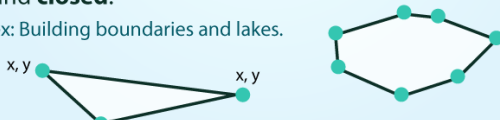
LINES: Composed of many (at least 2) vertices, or points, that are connected.

ex: Roads and streams.

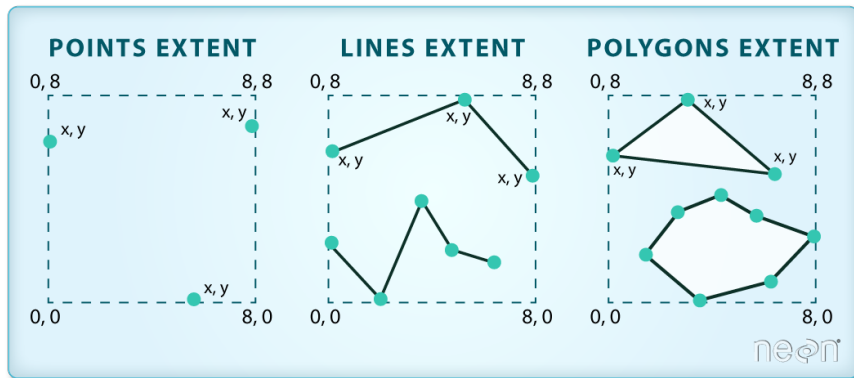


POLYGONS: 3 or more vertices that are connected and **closed**.

ex: Building boundaries and lakes.



Die Ausdehnung



Import mit dem Paket sf

```
library(sf)
```

```
## Linking to GEOS 3.6.1, GDAL 2.2.3, proj.4 4.9.3
```

- Mit dem Befehl `st_layers` kann man sehen, welche Layer verfügbar sind:

```
st_layers("../data/Amsterdam_highway_primary.osm")
```

```
## Driver: OSM
```

```
## Available layers:
```

##	layer_name	geometry_type	features	fields
## 1	points	Point	NA	10
## 2	lines	Line String	NA	9
## 3	multilinestrings	Multi Line String	NA	4
## 4	multipolygons	Multi Polygon	NA	25
## 5	other_relations	Geometry Collection	NA	4

Import von Layer lines

```
dat <- st_read("../data/Amsterdam_highway_primary.osm", "lines")
```

```
## Reading layer `lines' from data source `D:\github\geocourse'
## Simple feature collection with 1464 features and 9 fields
## geometry type:  LINESTRING
## dimension:      XY
## bbox:           xmin: 8.333102 ymin: 49.32801 xmax: 8.62799
## epsg (SRID):    4326
## proj4string:    +proj=longlat +datum=WGS84 +no_defs
```

```
plot(dat$geometry)
```



Import von Layer points

```
datp <- st_read("../data/Amsterdam_highway_primary.osm", "points")
```

```
## Reading layer `points' from data source `D:\github\geocours
## Simple feature collection with 800 features and 10 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 8.33654 ymin: 49.32801 xmax: 8.62696
## epsg (SRID):    4326
## proj4string:    +proj=longlat +datum=WGS84 +no_defs
```

```
plot(dat$geometry, pch=20, col=rgb(0,0,1,.1))
```



Mit einem anderen Paket plotten

```
library(tmap)  
qtm(dat$geometry)
```



```
st_layers("../data/ams_centraal.osm")
```

```
## Driver: OSM
```

```
## Available layers:
```

##	layer_name	geometry_type	features	fields
## 1	points	Point	NA	10
## 2	lines	Line String	NA	9
## 3	multilinestrings	Multi Line String	NA	4
## 4	multipolygons	Multi Polygon	NA	25
## 5	other_relations	Geometry Collection	NA	4

```
datm <- st_read("../data/ams_centraal.osm", "multipolygons")
```

```
## Reading layer `multipolygons' from data source `D:\github\g
```

```
## Simple feature collection with 2796 features and 25 fields
```

```
## geometry type: MULTIPOLYGON
```

```
## dimension: XY
```

```
## bbox: xmin: 4.874776 ymin: 52.36088 xmax: 4.92975
```

```
## epsg (SRID): 4326
```

Mehr Beispiele, wie man mit XML Daten umgeht:

- Deborah Nolan - **Extracting data from XML**

Noch mehr Informationen

Mehr Beispiele, wie man mit XML Daten umgeht:

- Deborah Nolan - **Extracting data from XML**
- Duncan Temple Lang - **A Short Introduction to the XML package for R**

Noch mehr Informationen

Mehr Beispiele, wie man mit XML Daten umgeht:

- Deborah Nolan - **Extracting data from XML**
- Duncan Temple Lang - **A Short Introduction to the XML package for R**

Noch mehr Informationen

- **Web Daten manipulieren**

Mehr Beispiele, wie man mit XML Daten umgeht:

- Deborah Nolan - **Extracting data from XML**
- Duncan Temple Lang - **A Short Introduction to the XML package for R**

Noch mehr Informationen

- **Web Daten manipulieren**
- **Tutorial zu xquery**

Mehr Beispiele, wie man mit XML Daten umgeht:

- Deborah Nolan - **Extracting data from XML**
- Duncan Temple Lang - **A Short Introduction to the XML package for R**

Noch mehr Informationen

- Web Daten manipulieren
- Tutorial zu xquery
- R und das Web (für Anfänger), Teil II: XML und R

Mehr Beispiele, wie man mit XML Daten umgeht:

- Deborah Nolan - **Extracting data from XML**
- Duncan Temple Lang - **A Short Introduction to the XML package for R**

Noch mehr Informationen

- **Web Daten manipulieren**
- **Tutorial zu xquery**
- **R und das Web (für Anfänger), Teil II: XML und R**
- Gaston Sanchez - **String Manipulation**

Mehr Beispiele, wie man mit XML Daten umgeht:

- Deborah Nolan - **Extracting data from XML**
- Duncan Temple Lang - **A Short Introduction to the XML package for R**

Noch mehr Informationen

- **Web Daten manipulieren**
- **Tutorial zu xquery**
- **R und das Web (für Anfänger), Teil II: XML und R**
- Gaston Sanchez - **String Manipulation**
- **Nutzung, Vor- und Nachteile OSM**

Mehr Beispiele, wie man mit XML Daten umgeht:

- Deborah Nolan - **Extracting data from XML**
- Duncan Temple Lang - **A Short Introduction to the XML package for R**

Noch mehr Informationen

- Web Daten manipulieren
- Tutorial zu xquery
- R und das Web (für Anfänger), Teil II: XML und R
- Gaston Sanchez - **String Manipulation**
- Nutzung, Vor- und Nachteile OSM
- Forschungsprojekte im Zusammenhang mit OpenStreetMap

Referenzen

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

```
##  
## To cite package 'XML' in publications use:  
##  
## Duncan Temple Lang and the CRAN Team (2018). XML: Tools for  
## Parsing and Generating XML Within R and S-Plus. R package  
## version 3.98-1.11. https://CRAN.R-project.org/package=XML  
##  
## A BibTeX entry for LaTeX users is  
##  
## @Manual{,  
##   title = {XML: Tools for Parsing and Generating XML With  
##   author = {Duncan Temple Lang and the CRAN Team},  
##   year = {2018},  
##   note = {R package version 3.98-1.11},  
##   url = {https://CRAN.R-project.org/package=XML}
```

Das neuere Paket

```
citation("xml2")
```

```
##  
## To cite package 'xml2' in publications use:  
##  
##   Hadley Wickham, James Hester and Jeroen Ooms (2018). xml2  
##   XML. R package version 1.2.0.  
##   https://CRAN.R-project.org/package=xml2  
##  
## A BibTeX entry for LaTeX users is  
##  
##   @Manual{,  
##     title = {xml2: Parse XML},  
##     author = {Hadley Wickham and James Hester and Jeroen Ooms},  
##     year = {2018},  
##     note = {R package version 1.2.0},  
##     url = {https://CRAN.R-project.org/package=xml2}
```