# Package: OpenStreetMap (via r-universe)

October 13, 2026

**Maintainer** Ian Fellows <ian@fellstat.com>

**License** GPL-2 | file LICENCE

**Title** Access to Open Street Map Raster Images

**Description** Accesses high resolution raster maps using the OpenStreetMap protocol. Dozens of road, satellite, and topographic map servers are directly supported. Additionally raster maps may be constructed using custom tile servers. Maps can be plotted using either base graphics, or ggplot2. This package is not affiliated with the OpenStreetMap.org mapping project.

**SystemRequirements** Java (>= 1.8), JRI

**Version** 0.4.1

**URL** https://github.com/ifellows/ROSM https://www.fellstat.com

**Depends** methods, R (>= 4.2.0)

**Imports** grDevices, ggplot2 (>= 0.9.0), rJava, raster, sp

**Collate** 'OpenStreetMap-package.R' 'osm.R' 'autoplot.R' 'zzz.R'

**RoxygenNote** 7.3.2

**NeedsCompilation** no

**Author** Ian Fellows [aut, cre], Jan-Peter Stotz [aut]

**Config/pak/sysreqs** default-jdk

**Repository** https://cran.r-universe.dev

**Date/Publication** 2025-09-29 23:30:02 UTC

**RemoteUrl** https://github.com/cran/OpenStreetMap

**RemoteRef** HEAD

**RemoteSha** 77372ef2f075798e575eb49dc3bf4c8b94ea98b0

# Contents

autoplot . . . 2
autoplot.OpenStreetMap . . . 3
autoplot.osmtile . . . 4
getMapInfo . . . 4
LA_places . . . 4
launchMapHelper . . . 5
longlat . . . 5
openmap . . . 5
openproj . . . 7
osm . . . 8
osmtile . . . 8
plot.OpenStreetMap . . . 9
plot.osmtile . . . 10
print.OpenStreetMap . . . 10
projectMercator . . . 11
raster,OpenStreetMap-method . . . 11
raster,osmtile-method . . . 12
states . . . 12

**Index** **13**

---

`autoplot` *Create a complete ggplot appropriate to a particular data type Rexported from ggplot2. 'autoplot()' uses ggplot2 to draw a particular plot for an object of a particular class in a single command. This defines the S3 generic that other classes and packages can extend.*

---

## Description

Create a complete ggplot appropriate to a particular data type Rexported from ggplot2. 'autoplot()' uses ggplot2 to draw a particular plot for an object of a particular class in a single command. This defines the S3 generic that other classes and packages can extend.

## Arguments

| | |
|---|---|
| `object` | An object (for example an OpenStreetMap object) |
| `...` | Additional arguments |

## Value

a ggplot object

## See Also

[autoplot.OpenStreetMap()]

autoplot.OpenStreetMap *Plot an open street map using ggplot2*

## Description

Plot an open street map using ggplot2

## Usage

```
## S3 method for class 'OpenStreetMap'
autoplot(object, expand = TRUE, ...)
```

## Arguments

| | |
|---|---|
| object | an OpenStreetMap object |
| expand | if true the plotting bounds are expanded to the bounding box |
| ... | not used |

## Examples

```
## Not run:
require(maps)
require(ggplot2)

mp <- openmap(c(53.38332836757155,-130.517578125),
  c(15.792253570362446,-67.939453125),4)
mp_bing <- openmap(c(53.38332836757155,-130.517578125),
  c(15.792253570362446,-67.939453125),4,'bing')
states_map <- map_data("state")
states_map_merc <- as.data.frame(
  projectMercator(states_map$lat,states_map$long))
states_map_merc$region <- states_map$region
states_map_merc$group <- states_map$group
crimes <- data.frame(state = tolower(rownames(USArrests)), USArrests)

p <- autoplot(mp,expand=FALSE) + geom_polygon(aes(x=x,y=y,group=group),
  data=states_map_merc,fill="black",colour="black",alpha=.1) + theme_bw()
print(p)
p <- autoplot(mp_bing) + geom_map(aes(x=-10000000,y=4000000,map_id=state,fill=Murder),
  data=crimes,map=states_map_merc)
print(p)

## End(Not run)
```

## autoplot.osmtile *Plots an open street map tile using ggplot2*

### Description

Plots an open street map tile using ggplot2

### Usage

```
## S3 method for class 'osmtile'
autoplot(object, plot = FALSE, ...)
```

### Arguments

| | |
|---|---|
| object | an osmtile |
| plot | if false only the annotation_raster is returned |
| ... | not used |

## getMapInfo *Returns a table with relevant source and attribution info for each map type*

### Description

Returns a table with relevant source and attribution info for each map type

### Usage

```
getMapInfo()
```

## LA_places *Places of interest in Los Angeles*

### Description

Places of interest in Los Angeles

## launchMapHelper *Launches a Java helper GUI.*

### Description

Launches a Java helper GUI.

### Usage

```
launchMapHelper()
```

### Details

note for Mac OS X users: On the mac this can only be run from a java console such as JGR.

## longlat *Latitude Longitude projection*

### Description

Latitude Longitude projection

### Usage

```
longlat()
```

## openmap *Get a map based on lat long coordinates*

### Description

Get a map based on lat long coordinates

### Usage

```
openmap(
  upperLeft,
  lowerRight,
  zoom = NULL,
 type = c("osm", "osm-german", "esri", "esri-topo", "esri-physical", "esri-shaded",
    "esri-imagery", "esri-terrain", "esri-natgeo", "nps", "apple-iphoto",
    "osm-public-transport"),
  minNumTiles = 9L,
  mergeTiles = TRUE
)
```

## Arguments

| | |
|---|---|
| upperLeft | the upper left lat and long |
| lowerRight | the lower right lat and long |
| zoom | the zoom level. If null, it is determined automatically |
| type | the tile server from which to get the map, or the url pattern. |
| minNumTiles | If zoom is null, zoom will be chosen such that the number of map tiles is greater than or equal to this number. |
| mergeTiles | should map tiles be merged into one tile |

## Details

Type may be the url of a custom tile server (http://wiki.osgeo.org/wiki/Tile_Map_Service_Specification). should include {z}, {y}, and {x} specifying where the zoom, xtile and ytile location should be substituted. e.g.

http://api.someplace.com/.../{z}/{x}/{y}.png

## Examples

```
## Not run:
# Some of the maps available
nm <- c("osm","osm-german","esri","esri-topo","esri-physical","esri-shaded",
  "esri-imagery","esri-terrain","esri-natgeo","nps","apple-iphoto")
par(mfrow=c(3,4), mar=c(0,0,0,0))
#Korea
for(i in 1:length(nm)){
 map <- openmap(c(43.46886761482925,119.94873046875),
   c(33.22949814144951,133.9892578125),
  minNumTiles=3,type=nm[i])
plot(map)
}
# Maps from custom urls (use your own API key)
apiKey <- paste0("?access_token=",
 "pk.eyJ1IjoidGhlZmVsbCIsImEiOiJjaXN1anNwODEwMWlrMnRvZHBhamRrZjlqIn0.Gf8qLSpZ6yo5yfQhEutFfQ")
baseUrl <- "https://api.mapbox.com/styles/v1/mapbox/satellite-streets-v9/tiles/256/{z}/{x}/{y}"
map <- openmap(c(43.46886761482925,119.94873046875),
 c(33.22949814144951,133.9892578125),
  minNumTiles=4,
  type=paste0(baseUrl,apiKey))
plot(map)



# Plot Korea with ggplot2.
library(ggplot2)
map <- openmap(c(43.46886761482925,119.94873046875),
 c(33.22949814144951,133.9892578125),
 minNumTiles=4)
autoplot(map)
```

```
# Lambert Conic Conformal Map Projection
map_llc <- openproj(map, projection=
"+proj=lcc +lat_1=33 +lat_2=45 +lat_0=39 +lon_0=-96")
plot(map_llc,removeMargin=TRUE)



## End(Not run)
```

---

openproj *Projects the open street map to an alternate coordinate system*

---

## Description

Projects the open street map to an alternate coordinate system

## Usage

```
openproj(x, projection = "+proj=longlat", ...)
```

## Arguments

| | |
|---|---|
| x | an OpenStreetMap object |
| projection | a proj4 character string or CRS object |
| ... | additional parameters for projectRaster |

## Examples

```
## Not run:
library(maps)

#plot map in native mercator coords
map <- openmap(c(70,-179),
 c(-70,179),zoom=1,type="esri-imagery")
plot(map)

#using longlat projection lets us combine with the maps library
map_longlat <- openproj(map)
plot(map_longlat)
map("world",col="red",add=TRUE)

#robinson projection. good for whole globe viewing.
map_robinson <- openproj(map_longlat, projection=
"+proj=robin +lon_0=0 +x_0=0 +y_0=0 +ellps=WGS84 +datum=WGS84 +units=m +no_defs")
plot(map_robinson)

#national parks service images
upperMap <- openmap(c(70,-179),
 c(10,50),zoom=2,type='nps')
```

```
#Lambert Conic Conformal
map_llc <- openproj(upperMap, projection=
"+proj=lcc +lat_1=33 +lat_2=45 +lat_0=39 +lon_0=-96")
plot(map_llc,removeMargin=TRUE)
#add choropleth
library(sp)
data(states)
st_llc <- spTransform(states,CRS("+proj=lcc +lat_1=33 +lat_2=45 +lat_0=39 +lon_0=-96"))
plot(st_llc,add=T,col=heat.colors(48,.4)[slot(st_llc,"data")[["ORDER_ADM"]]])


## End(Not run)
```

---

`osm` *Open street map (and google) mercator projection*

---

## Description

Open street map (and google) mercator projection

## Usage

```
osm()
```

---

`osmtile` *Get an open street map tile.*

---

## Description

Get an open street map tile.

## Usage

```
osmtile(x, y, zoom, type = "osm")
```

## Arguments

| | |
|---|---|
| `x` | location in osm native coordinates |
| `y` | location in osm native coordinates |
| `zoom` | zoom level |
| `type` | the map type (see getMapInfo) |

## Value

a tile

`plot.OpenStreetMap` *Plot an OpenStreetMap object.*

## Description

Plot an OpenStreetMap object.

## Usage

```
## S3 method for class 'OpenStreetMap'
plot(x, y = NULL, add = FALSE, removeMargin = TRUE, ...)
```

## Arguments

| | |
|---|---|
| `x` | the OpenStreetMap |
| `y` | ignored |
| `add` | add to current plot |
| `removeMargin` | remove margins from plotting device |
| `...` | additional parameters to be passed to plot |

## Examples

```
## Not run:
#
# The following examples
# plot using native mercator coordinates,
# transforming the data where needed
#
library(sp)
m <- c(25.7738889,-80.1938889)
j <- c(58.3019444,-134.4197222)
miami <- projectMercator(25.7738889,-80.1938889)
jun <- projectMercator(58.3019444,-134.4197222)
data(states)
map <- openmap(j,m,4,type="esri-terrain")
plot(map,removeMargin=FALSE)
plot(states,add=TRUE)

data(LA_places)
longBeachHarbor <- openmap(c(33.760525217369974,-118.22052955627441),
 c(33.73290566922855,-118.17521095275879),14,'osm')
coords <- coordinates(LA_places)
x <- coords[,1]
y <- coords[,2]
txt <- slot(LA_places,"data")[,'NAME']
plot(longBeachHarbor)
points(x,y,col="red")
text(x,y,txt,col="white",adj=0)
```

```
## End(Not run)
```

---

`plot.osmtile` *Add tile to plot*

---

## Description

Add tile to plot

## Usage

```
## S3 method for class 'osmtile'
plot(x, y = NULL, add = TRUE, raster = TRUE, ...)
```

## Arguments

| | |
|---|---|
| `x` | the tile |
| `y` | ignored |
| `add` | add to current plot (if raster, then image is always added) |
| `raster` | use raster image |
| `...` | additional parameters to image or rasterImage |

---

`print.OpenStreetMap` *Print map*

---

## Description

Print map

## Usage

```
## S3 method for class 'OpenStreetMap'
print(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | the OpenStreetMap |
| `...` | ignored |

projectMercator *Maps long lat values to the open street map mercator projection*

## Description

Maps long lat values to the open street map mercator projection

## Usage

```
projectMercator(lat, long, drop = TRUE)
```

## Arguments

| | |
|---|---|
| lat | a vector of latitudes |
| long | a vector of longitudes |
| drop | drop to lowest dimension |

raster,OpenStreetMap-method *Create a RasterLayer from an OpenStreetMap*

## Description

Create a RasterLayer from an OpenStreetMap

## Usage

```
## S4 method for signature 'OpenStreetMap'
raster(x, ...)
```

## Arguments

| | |
|---|---|
| x | an OpenStreetMap |
| ... | unused |

## Examples

```
## Not run:
library(raster)
longBeachHarbor <- openmap(c(33.760525217369974,-118.22052955627441),
  c(33.73290566922855,-118.17521095275879),14,"esri-imagery")
ras <- raster(longBeachHarbor)
plotRGB(ras)

## End(Not run)
```

## raster,osmtile-method *Create a RasterLayer from a tile*

### Description

Create a RasterLayer from a tile

### Usage

```
## S4 method for signature 'osmtile'
raster(x, ...)
```

### Arguments

| | |
|---|---|
| x | an osmtile |
| ... | unused |

## states *The United States*

### Description

The United States

# Index

∗ **data**
LA_places, 4
states, 12

autoplot, 2
autoplot.OpenStreetMap, 3
autoplot.osmtile, 4

getMapInfo, 4

LA_places, 4
launchMapHelper, 5
longlat, 5

openmap, 5
openproj, 7
osm, 8
osmtile, 8

plot.OpenStreetMap, 9
plot.osmtile, 10
print.OpenStreetMap, 10
projectMercator, 11

raster,OpenStreetMap-method, 11
raster,osmtile-method, 12

states, 12