Package 'rosm'

October 5, 2015

Type Package

Title Plot Raster Map Tiles from Open Street Map and Other Sources

Version 0.1.2

Date 2015-10-18

Author Dewey Dunnington <dewey@fishandwhistle.net>

Maintainer Dewey Dunnington <dewey@fishandwhistle.net>

Description Download and plot Open Street Map <http://www.openstreetmap.org/>, Mapquest <http://www.mapquest.com/>, Bing Maps <http://www.bing.com/maps> and other tiled map sources in a way that works seamlessly with plotting from the 'sp' package. Use to create high-resolution basemaps and add hillshade to vector based maps.

License GPL-2

LazyData TRUE

Imports abind, jpeg, png, sp, rgdal, digest, rjson, methods

Suggests prettymapr, rcanvec

URL https://github.com/paleolimbot/rosm

BugReports https://github.com/paleolimbot/rosm

NeedsCompilation no

R topics documented:

rosm-package 2
bmaps.plot 3
bmaps.types 4
osm.plot 4
osm.types 5

Index **7**

rosm-package *Plot Raster Map Tiles From Open Street Map and Other Sources*

Description

This package provides access and plots Open Street Map, Mapquest, and Bing Maps tiles to create high resolution basemaps and use hillshade tiles to add texture to other maps. Uses the 'sp' package to plot using base graphics. Plot Open Street Map derivative tiles using osm.plot, and plot Bing maps (Aerial, Labeled Aerial, Road) using bmaps.plot. 16 OSM and 3 Bing sources are included, with the ability to define custom tile sources based on OSM tilex, tiley, and zoom.

This package provides a simpler interface than the 'OpenStreetMap' and 'cartography' packages, but lacks the ability to reproject images and contains fewer built-in tile sources. See the cartography package vingette for more details.

Author(s)

Dewey Dunnington <dewey@fishandwhistle.net>

References

Open Street Map tile servers, Bing Maps API documentation

Examples

```
library(rcanvec)
library(prettymapr)
library(sp)

#basic ploting
nsbox <- searchbbox("nova scotia", source="google")
osm.plot(nsbox)
osm.plot(nsbox, type="mapquestsat")
bmaps.plot(nsbox)
bmaps.plot(nsbox, type="Road")

#use {prettymapr} to add scalebar and north arrow
prettymap(osm.plot(nsbox))
prettymap(bmaps.plot(nsbox, type="Road"))

#increase res argument to plot to file
pdf(height=8, width=10.5)
prettymap(osm.plot(nsbox, type="mapquestsat", res=300, stoponlargerequest=FALSE),
          scale.label.col="white", arrow.text.col = "white",
          scale.linecol = "white", arrow.border = "white")
dev.off()

#canvec.qplot and hillshade using the add=TRUE argument
```

```
prettymap({
  altabox <- prettymapr::searchbbox("Alta Lake BC", source="google")
  canvec.qplot(bbox=altabox,
               layers=c("waterbody", "forest", "river", "road"))
  osm.plot(altabox, type="hillshade", add=T, project = FALSE)
})

#define custom types by defining a tile.url.TYPE function
tile.url.darkmatter <- function(xtile, ytile, zoom) {
   paste0(paste("http://a.basemaps.cartocdn.com/dark_all",
                              zoom, xtile, ytile, sep="/"), ".png")
}
tile.maxzoom.darkmatter <- function() {return(19)} #useful it maxzoom is important
osm.plot(nsbox, type="darkmatter")
```

`bmaps.plot` *Plot Bing Maps*

Description

Identical syntax to osm.plot, but using Bing maps (https://www.bing.com/maps/) instead of Open Street Map.

Usage

```
bmaps.plot(bbox, type = "Aerial", key = NULL, ...)
```

Arguments

bbox	A bounding box as generated by `sp::bbox()` or `prettymapr::searchbbox()`
type	Use `Aerial`, `AerialWithLabels`, or `Road`.
key	If plotting a large number of images, consider getting your own (free) key at the Microsoft Website.
...	Arguments passed on to osm.plot.

Examples

```
library(prettymapr)
bmaps.plot(makebbox(47.2, -59.7, 43.3, -66.4))
bmaps.plot(makebbox(47.2, -59.7, 43.3, -66.4), type="Road")
```

bmaps.types *List types of Bing Maps*

Description

List types of Bing Maps

Usage

```
bmaps.types()
```

Value

A list of valid bing map types

Examples

```
bmaps.types()
```

osm.plot *Plot Open Street Map Tiles*

Description

Plot Open Street Map tiles using `rasterImage` and `sp::plot`. Define your own tile sources by creating a tile url function in the global environment, although most OSM listed servers are included. See osm.types for types options. By default tiles are plotted in the Spherical Mercator projection (epsg:3857); pass `project=FALSE` to keep lat/lon coordinates.

Usage

```
osm.plot(bbox, zoomin = 0, zoom = NULL, type = "osm",
  forcedownload = FALSE, stoponlargerequest = TRUE, fusetiles = TRUE,
  cachedir = NULL, res = 150, project = TRUE, ...)
```

Arguments

bbox	A bounding box as generated by `sp::bbox()` or `prettymapr::searchbbox()`
zoomin	The amount by which to adjust the automatically calculated zoom (or manually specified if the `zoom` parameter is passed). Use +1 to zoom in, or -1 to zoom out.
zoom	Manually specify the zoom level (not reccomended; adjust `zoomin` or `res` instead.
type	A map type; one of that returned by osm.types. User defined types are possible by defining `tile.url.TYPENAME <- function(xtile, ytile, zoom){}` and passing TYPENAME as the `type` argument.

forcedownload	TRUE if cached tiles should be re-downloaded. Useful if some tiles are corrupted.
stoponlargerequest	By default osm.plot will only load 32 tiles at a time. If plotting at a higher resolution it may be necessary to pass true here.
fusetiles	TRUE if tiles should be fused into a single image. This is the default because white lines appear between tiles if it is set to FALSE. PDFs appear not to have this problem, so when plotting large, high resolution PDFs it may be faster (and more memory efficient) to use fusetiles=FALSE.
cachedir	The directory in which tiles should be cached. Defaults to getwd()/rosm.cache.
res	The resolution used to calculate scale.
project	TRUE if tiles should be projected to a pseudo-mercator projection, FALSE if lat/lon should be maintained. Becuase sp::plot adjusts the aspect according to latitude for lat/lon coordinates, this makes little difference at high zoom and may make plotting overlays more convenient. Defaults to TRUE.
...	Additional parameters to be passed on to the first call to sp::plot

Examples

```
library(prettymapr)
ns <- makebbox(47.2, -59.7, 43.3, -66.4)
osm.plot(ns)
osm.plot(ns, type="mapquestsat")
prettymap(osm.plot(ns), scale.style="ticks", scale.tick.cex=0)

tile.url.darkmatter <- function(xtile, ytile, zoom) {
  paste0(paste("http://a.basemaps.cartocdn.com/dark_all",
                           zoom, xtile, ytile, sep="/"), ".png")
}
osm.plot(ns, type="darkmatter")
```

---

osm.types *Get List of Valid Types*

Description

Get List of Valid Types

Usage

```
osm.types()
```

Value

A character vector of valid type parameters.

Examples

```
osm.types()
```

Index

∗Topic **package**
rosm-package, 2

bmaps.plot, *2*, 3
bmaps.types, 4

osm.plot, *2*, *3*, 4
osm.types, *4*, 5

rosm *(*rosm-package*)*, 2
rosm-package, 2