Package 'maptiles'

May 8, 2026

Title Download and Display Map Tiles

Version 0.11.0

Description To create maps from tiles, 'maptiles' downloads, composes and displays tiles from a large number of providers (e.g. 'OpenStreetMap', 'Stadia', 'Esri', 'CARTO', or 'Thunderforest').

URL https://github.com/riatelab/maptiles/

BugReports https://github.com/riatelab/maptiles/issues/

License GPL-3

Depends R (>= 3.5.0)

Imports sf, curl, digest, graphics, grDevices, png, terra (>= 1.8-21), tools, utils

Suggests covr, tinytest

Encoding UTF-8

RoxygenNote 7.3.3

NeedsCompilation no

Author Timothée Giraud [cre, aut] (ORCID: <https://orcid.org/0000-0002-1932-3323>), Diego Hernangómez [ctb] (ORCID: <https://orcid.org/0000-0001-8457-4658>), Robert J. Hijmans [ctb] (ORCID: <https://orcid.org/0000-0001-5872-2872>), Hugh A. Graham [ctb], Miles McBain [cph] (Slippy map tiles functions, from slippymath package)

Maintainer Timothée Giraud <timothee.giraud@cnrs.fr>

Repository CRAN

Date/Publication 2025-12-12 16:50:02 UTC

Contents

`create_provider` *Create a new tile provider*

Description

Use this function to create new tiles provider.

Usage

```
create_provider(name, url, sub = c("a", "b", "c"), citation)
```

Arguments

`name`	name of the provider.
`url`	url of the provider. The url must contain {x}, {y} and {z} placeholders. It may also contain {s} for sub-domains or {apikey} for API keys (see Examples).
`sub`	sub-domains.
`citation`	attribution text of the provider.

Value

a list is returned. This list can be used by get_tiles.

Examples

```
stadia_toner <- create_provider(
  name = "Stadia_stamen_toner",
 url = "https://tiles.stadiamaps.com/tiles/stamen_toner/{z}/{x}/{y}{r}.png?api_key={apikey}",
  citation = "© Stadia Maps © Stamen Design © OpenMapTiles © OpenStreetMap contributors"
)
opentopomap <- create_provider(
  name = "otm",
  url = "https://{s}.tile.opentopomap.org/{z}/{x}/{y}.png",
  sub = c("a", "b", "c"),
 citation = "map data: © OpenStreetMap contributors, SRTM | map style: © OpenTopoMap (CC-BY-SA)"
)
IGN <- create_provider(
  name = "orthophoto_IGN",
```

```
  url = paste0(
    "https://data.geopf.fr/wmts?",
    "request=GetTile",
    "&service=WMTS",
    "&version=1.0.0",
    "&style=normal",
    "&tilematrixset=PM_6_18",
    "&format=image/jpeg",
    "&layer=ORTHOIMAGERY.ORTHOPHOTOS.BDORTHO",
    "&tilematrix={z}",
    "&tilerow={y}",
    "&tilecol={x}"
  ),
  citation = "IGN, BD ORTHO®"
)

# Find TileMatrixSet and Style values
layer <- "ORTHOIMAGERY.ORTHOPHOTOS.BDORTHO"
path <- "https://data.geopf.fr/wmts?"
param_info <- "service=wmts&request=GetCapabilities&version=1.0.0"
url <- paste0("WMTS:", path, param_info, ",layer=", layer)
## Not run:
tmp <- tempfile(fileext = ".xml")
sf::gdal_utils(
  util = "translate",
  source = url, destination = tmp,
  options = c("-of", "WMTS")
)
readLines(tmp)

## End(Not run)
```

`get_credit` *Get basemap tiles attribution*

Description

Get the attribution of map tiles.

Usage

```
get_credit(provider)
```

Arguments

`provider`	provider name or provider object (as produced by create_provider).

Examples

```
get_credit("OpenStreetMap")
```

get_providers *Providers*

Description

List of builtin providers with their name, URL, subdomains and attribution text.

Usage

```
get_providers()
```

Value

A list of is returned.

Examples

```
get_providers()
```

get_tiles *Get basemap tiles from map servers*

Description

Get map tiles based on a spatial object extent. Maps can be fetched from various map servers ('OpenStreetMap', 'Stadia', 'Esri', 'CARTO', or 'Thunderforest').

Usage

```
get_tiles(
  x,
  provider = "OpenStreetMap",
  zoom,
  crop = FALSE,
  project = TRUE,
  verbose = FALSE,
  apikey,
  cachedir,
  forceDownload = FALSE,
  retina = TRUE
)
```

Arguments

x	sf, sfc, bbox, SpatRaster, SpatVector or SpatExtent object. If x is a SpatExtent it must express coordinates in lon/lat WGS84 (epsg:4326).
provider	tile server to get the tiles from. It can be one of the builtin providers (see Details for the list) or a named list produced by create_provider (see Examples).
zoom	zoom level (see Details).
crop	TRUE if results should be cropped to the specified x extent, FALSE otherwise. If x is an sf object with one POINT, crop is set to FALSE.
project	if TRUE, the output is projected to the crs of x. If FALSE the output uses "EPSG:3857" (Web Mercator).
verbose	if TRUE, tiles filepaths, zoom level and attribution are displayed.
apikey	API key. Not needed for Thunderforest or Stadia servers if environment variables named "THUNDERFOREST_MAPS" or "STADIA_MAPS" are set.
cachedir	name of a folder used to cache tiles. If not set, tiles are cached in a tempdir folder.
forceDownload	if TRUE, existing cached tiles may be overwritten.
retina	if TRUE, tiles are downloaded in high resolution if they exist. Stadia and CARTO provide such tiles.

Details

Zoom levels are described in the OpenStreetMap wiki: https://wiki.openstreetmap.org/wiki/Zoom_levels.

Here is the complete list of builtin providers:

"OpenStreetMap", "OpenStreetMap.DE", "OpenStreetMap.France", "OpenStreetMap.HOT", "OpenTopoMap",

"Stadia.AlidadeSmooth", "Stadia.AlidadeSmoothDark", "Stadia.OSMBright", "Stadia.Outdoors", "Stadia.StamenToner", "Stadia.StamenTonerBackground", "Stadia.StamenTonerLines", "Stadia.StamenTonerLabels", "Stadia.StamenTonerLite", "Stadia.StamenWatercolor", "Stadia.StamenTerrain", "Stadia.StamenTerrainBackground", "Stadia.StamenTerrainLabels",

"Esri.WorldStreetMap", "Esri.WorldTopoMap", "Esri.WorldImagery", "Esri.WorldTerrain", "Esri.WorldShadedRelief", "Esri.OceanBasemap", "Esri.NatGeoWorldMap", "Esri.WorldGrayCanvas",

"CartoDB.Positron", "CartoDB.PositronNoLabels", "CartoDB.PositronOnlyLabels", "CartoDB.DarkMatter", "CartoDB.DarkMatterNoLabels", "CartoDB.DarkMatterOnlyLabels", "CartoDB.Voyager", "CartoDB.VoyagerNoLabels", "CartoDB.VoyagerOnlyLabels",

"Thunderforest.OpenCycleMap", "Thunderforest.Transport", "Thunderforest.TransportDark", "Thunderforest.SpinalMap", "Thunderforest.Landscape", "Thunderforest.Outdoors", "Thunderforest.Pioneer", "Thunderforest.MobileAtlas", "Thunderforest.Neighbourhood"

Value

A SpatRaster is returned.

Examples

```
## Not run:
library(sf)
library(maptiles)
nc <- st_read(system.file("shape/nc.shp", package = "sf"), quiet = TRUE)
nc_osm <- get_tiles(nc, crop = TRUE, zoom = 6)
plot_tiles(nc_osm)

# Create a provider from a custom url
osm_tiles <- create_provider(
  name = "osm_tiles",
  url = "https://tile.openstreetmap.org/{z}/{x}/{y}.png",
  citation = "© OpenStreetMap contributors."
)
# Download tiles and compose raster (SpatRaster)
nc_osm2 <- get_tiles(
  x = nc, provider = osm_tiles, crop = FALSE,
  zoom = 6, project = FALSE, verbose = TRUE
)
# Plot the tiles
plot_tiles(nc_osm2)
# Add attribution
mtext(get_credit(osm_tiles), side = 1, line = -1)

## End(Not run)
```

`maptiles` *Download and Display Map Tiles*

Description

To create maps from tiles, maptiles downloads, composes and displays tiles from a large number of providers (e.g. OpenStreetMap, Stamen, Esri, CARTO, or Thunderforest).

Author(s)

Maintainer: Timothée Giraud <timothee.giraud@cnrs.fr> (ORCID)

Other contributors:

- Diego Hernangómez (ORCID) [contributor]
- Robert J. Hijmans (ORCID) [contributor]
- Hugh A. Graham [contributor]
- Miles McBain (Slippy map tiles functions, from slippymath package) [copyright holder]

See Also

Useful links:

- https://github.com/riatelab/maptiles/
- Report bugs at https://github.com/riatelab/maptiles/issues/

`plot_tiles` *Plot map tiles*

Description

Plot map tiles.

Usage

```
plot_tiles(x, adjust = FALSE, add = FALSE, ...)
```

Arguments

x	a SpatRaster object.
adjust	if TRUE, plot the raster without zoom-in or zoom-out in the graphic device: add margins if the raster is smaller than the graphic device, zoom-in if the raster is larger than the graphic device. This feature does not work with an unprojected (lon/lat) raster.
add	whether to add the layer to an existing plot (TRUE) or not (FALSE).
...	bgalpha, smooth, or other arguments passed to be passed to plotRGB

Note

This function is a wrapper for plotRGB from the terra package.

Examples

```
## Not run:
library(sf)
library(maptiles)
nc <- st_read(system.file("shape/nc.shp", package = "sf"), quiet = TRUE)
nc_osm <- get_tiles(nc, crop = TRUE)
plot_tiles(nc_osm)

## End(Not run)
```

Index