Package: inbospatial (via r-universe)

August 24, 2024

Type Package

Title A Collection of Useful R Functions for Spatial Data

Version 0.0.2

Description Useful R functions developed at the Research Institute for Nature and Forest (INBO) for dealing with spatial raster or vector data.

License MIT + file LICENSE

URL https://github.com/inbo/inbospatial

BugReports https://github.com/inbo/inbospatial/issues

Imports assertthat, httr, magrittr, readr, sf, stringr, terra, xml2

Suggests dplyr, knitr, leaflet, leaflet.extras, leaflet.extras2, mapview, rmarkdown, waldo

VignetteBuilder knitr

Config/checklist/communities inbo

Config/checklist/keywords spatial; geospatial; utilities; GIS; raster; vector

Encoding UTF-8

Language en-GB

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Repository https://inbo.r-universe.dev

RemoteUrl https://github.com/inbo/inbospatial

RemoteRef HEAD

RemoteSha b3203b795d3bd96cfcaebbb0ef4540d5503cb948

Contents

`add_wms_be_cartoweb` *addWMSTiles() and addWMS() wrapper functions for WMS services*

Description

The `add_wms_*()` functions are shorthand alternatives for a fully specified `leaflet::addWMSTiles()` or `leaflet.extras2::addWMS()` statement. Appropriate attribution is added to the Leaflet map depending on the layer.

Usage

```
add_wms_be_cartoweb(
  map,
  layer = c("topo", "topo_grey", "overlay", "crossborder"),
  options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE),
  ...
)

add_wms_be_ortho(
  map,
  options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE),
  ...
)

add_wms_nl_ortho(
  map,
  options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE),
  ...
)

add_wms_fl_grbmap(
  map,
  options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE),
  ...
)

add_wms_fl_habitatmap(
```

```
  map,
  options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE, info_format
    = "text/html"),
  popupOptions = leaflet::popupOptions(maxWidth = 1200),
  add_wms_legend = FALSE,
  ...
)

add_wms_fl_habitatlabels(
  map,
  options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE),
  ...
)

add_wms_fl_forestnature(
  map,
  options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE, info_format
    = "text/html"),
  popupOptions = leaflet::popupOptions(maxWidth = 600),
  add_wms_legend = FALSE,
  ...
)

add_wms_fl_agriculture(
  map,
  year,
  options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE, info_format
    = "text/html"),
  popupOptions = leaflet::popupOptions(maxWidth = 1200),
  add_wms_legend = FALSE,
  ...
)

add_wms_nlnb_nnb(
  map,
  options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE, info_format
    = "text/html"),
  popupOptions = leaflet::popupOptions(maxWidth = 600),
  add_wms_legend = FALSE,
  ...
)

add_wms_nlnb_natte_natuurparels(
  map,
  options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE, info_format
    = "text/html"),
  popupOptions = leaflet::popupOptions(maxWidth = 2000),
  add_wms_legend = FALSE,
```

```
  ...
)

add_wms_nlnb_ambitie_landschapstypen(
  map,
 options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE, info_format
    = "text/html"),
  popupOptions = leaflet::popupOptions(maxWidth = 2000),
  add_wms_legend = FALSE,
  ...
)

add_wms_nlnb_ambitie_natuurtypen(
  map,
 options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE, info_format
    = "text/html"),
  popupOptions = leaflet::popupOptions(maxWidth = 2000),
  add_wms_legend = FALSE,
  ...
)

add_wms_nlnb_landschapstypen(
  map,
 options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE, info_format
    = "text/html", minZoom = 13),
  popupOptions = leaflet::popupOptions(maxWidth = 2000),
  add_wms_legend = FALSE,
  ...
)

add_wms_nlnb_natuurtypen(
  map,
 options = leaflet::WMSTileOptions(format = "image/png", transparent = TRUE, info_format
    = "text/html", minZoom = 13),
  popupOptions = leaflet::popupOptions(maxWidth = 2000),
  add_wms_legend = FALSE,
  ...
)
```

Arguments

map	a map widget object created from leaflet()
layer	String that defines which layer to use, if the function supports more than one. In such case, the available strings are shown in the Usage section. The first value is used if missing.
options	a list of extra options for tile layers, popups, paths (circles, rectangles, polygons, ...), or other map elements
...	Further arguments passed to leaflet::addWMSTiles() or leaflet.extras2::addWMS().

popupOptions	A Vector of popupOptions to provide popups
add_wms_legend	Logical. Is a legend to be added for this WMS? Note that the legend cannot be toggled on and off in the map; it is plotted as a separate, static legend.
year	Year to be applied in selecting the WMS

Details

On condition that these functions continue being maintained, using the shorthand functions should make your scripts more futureproof.

Value

A leaflet HTML widget object.

Abbreviations

Prefixes:

be_ Belgium
fl_ Flanders
nl_ Netherlands
nlnb_ Netherlands: Province 'Noord-Brabant'

Suffixes:

Background layers:

_ortho Orthophotographs
_cartoweb Topographic Cartoweb layers (Belgium)
_grbmap GRB basemap (Flanders; 'Grootschalig Referentiebestand')

Feature layers:

_habitatmap Natura 2000 habitat types
_habitatlabels Natura 2000 habitat type labels
_forestnature Public forest and nature areas managed by the Flemish Nature & Forest Agency
_agriculture Parcels with the agricultural use in a specific year
_nnb Netherlands; 'Natuurnetwerk Brabant (Rijk en Provincie NNB)'
_natte_natuurparels Netherlands; 'Natte natuurparels'
_ambitie_landschapstypen Netherlands; 'Natuurbeheerplan - Ambitiekaart Landschaptypen'
_ambitie_natuurtypen Netherlands; 'Natuurbeheerplan - Ambitiekaart Natuurtypen'
_landschapstypen Netherlands; 'Natuurbeheerplan - Beheertypenkaart_Landschaptypen'
_natuurtypen Netherlands; 'Natuurbeheerplan - Beheertypenkaart Natuurtypen'

Author(s)

Floris Vanderhaeghe, https://github.com/florisvdh

See Also

Other topics on using web services: add_wmts_nl_brt(), get_coverage_wcs(), get_feature_wfs()

Examples

```
library(leaflet)
leaflet() |>
  setView(lng = 4.5, lat = 51.45, zoom = 11) |>
  add_wmts_nl_brt("grijs") |>
  add_wms_be_cartoweb("topo_grey") |>
  add_wms_fl_forestnature() |>
  add_wms_nlnb_nnb()

# It can also use mapview objects, if you first extract the leaflet map slot:
library(mapview)
kmi_stations <- sf::read_sf(
  system.file("extdata/kmi_stations.geojson", package = "inbospatial")
)
mapview(kmi_stations, map.types = "CartoDB.Positron")@map |>
  add_wms_be_cartoweb()
# alternative syntax:
## Not run:
mapview(kmi_stations, map.types = "CartoDB.Positron") |>
  _@map |>
  add_wms_be_cartoweb()

## End(Not run)
```

`add_wmts_nl_brt` *addTiles() wrapper functions for WMTS services*

Description

The `add_wmts_*()` functions are shorthand alternatives for a fully specified `leaflet::addTiles()` statement. Appropriate attribution is added to the Leaflet map depending on the layer.

Usage

```
add_wmts_nl_brt(map, layer = c("standaard", "grijs", "pastel", "water"), ...)

add_wmts_nl_ortho(map, ...)
```

Arguments

`map`	a map widget object created from `leaflet()`
`layer`	String that defines which layer to use, if the function supports more than one. In such case, the available strings are shown in the Usage section. The first value is used if missing.
`...`	Further arguments passed to `leaflet::addTiles()`.

Details

On condition that these functions continue being maintained, using the shorthand functions should make your scripts more futureproof.

Value

A `leaflet` HTML widget object.

Abbreviations

Prefixes:

`be_` Belgium

`nl_` Netherlands

Suffixes:

`_ortho` Orthophotographs

`_brt` Topographic BRT layers (Netherlands; 'Basisregistratie Topografie')

Author(s)

Floris Vanderhaeghe, https://github.com/florisvdh

See Also

Other topics on using web services: `add_wms_be_cartoweb()`, `get_coverage_wcs()`, `get_feature_wfs()`

Examples

```
library(leaflet)
leaflet() |>
  setView(lng = 5.5, lat = 52.5, zoom = 9) |>
  add_wmts_nl_brt()
leaflet() |>
  setView(lng = 5.4, lat = 52.2, zoom = 14) |>
  add_wmts_nl_ortho()
```

---

get_coverage_wcs *Get a layer from a web coverage service within a bounding box*

Description

The function sends a query to a WCS service, downloads it to a temporary file from which it is read with `terra::rast()` - if needed reprojected - and returned as a `SpatRaster`object

Usage

```
get_coverage_wcs(
  wcs = c("dtm", "dsm", "omz", "omw"),
  bbox,
  layername,
  resolution,
  wcs_crs = "EPSG:4258",
  output_crs = "EPSG:31370",
  bbox_crs = "EPSG:31370",
  version = c("1.0.0", "2.0.1"),
  ...
)
```

Arguments

wcs	One of "dtm", "dsm", "omz", "omw"
bbox	An object of class bbox of length 4.
layername	Character string; name of the layer
resolution	Output resolution in meters
wcs_crs	Native CRS in which the raster layers are stored on the WCS
output_crs	Output CRS. May involve reprojection.
bbox_crs	CRS in which bbox coordinates are passed
version	WCS version to be used.
...	Additional key-value pairs passed on to the WCS query

Details

The following WCS services can currently be used:

- "omz": orthophotomosaic summer images Flanders
- "omw": orthophotomosaic winter images Flanders
- "dtm": digital terrain model Flanders
- "dsm": digital surface model Flanders See metadata Vlaanderen for more information # nolint: line_length_linter.

Value

A SpatRaster object

See Also

Other topics on using web services: add_wms_be_cartoweb(), add_wmts_nl_brt(), get_feature_wfs()

Examples

```
## Not run:
bbox <- sf::st_bbox(
  c(xmin = 155800, xmax = 155850, ymin = 132700, ymax = 132750),
  crs = sf::st_crs(31370)
)
get_coverage_wcs(
  wcs = "dsm",
  bbox = bbox,
  layername = "EL.GridCoverage.DSM",
  resolution = 1
)

## End(Not run)
```

`get_feature_wfs` *Get a layer from a web feature service*

Description

This function constructs a URL request from its arguments and either reads in the resulting vector layer as a sf object or returns the number of features that are requested. The request is made up of key-value pairs and additional key-value pairs can be passed to the function. The full documentation for the WFS standard can be consulted from https://www.ogc.org/standard/wfs/.

Usage

```
get_feature_wfs(
  wfs,
  version = "2.0.0",
  layername = NULL,
  crs = NULL,
  bbox = NULL,
  filter = NULL,
  cql_filter = NULL,
  output_format = NULL,
  property_name = NULL,
  result_type = c("results", "hits"),
  ...
)
```

Arguments

wfs	Web address for the service which you want to query features from
version	Version number for the service. For instance "2.0.0".
layername	Optional name of a layer hosted by the web feature service

crs	Optional coordinate reference system to represent the features. For instance "EPSG:31370".
bbox	Optional bounding box. Pass this as a named vector with names "xmin", "xmax", "ymin", "ymax".
filter	Optional standard OGC filter specification
cql_filter	Optional Contextual Query Language filter. This currently only works if the WFS is hosted on a GeoServer.
output_format	Optional output format supported by the WFS.
property_name	Optional character string. Which fields or columns to return? If you want to specify multiple columns, separate them by a comma. The column containing the feature geometry is usually called geom, geometry or SHAPE.
result_type	For version "2.x.x", this can be either "results" (default) or "hits". The former returns the requested features, the latter returns the number of requested features.
...	Additional key-value pairs passed on to the WFS query.

Details

See https://inbo.github.io/tutorials/tutorials/spatial_wfs_services/ for more information.

See Also

Other topics on using web services: add_wms_be_cartoweb(), add_wmts_nl_brt(), get_coverage_wcs()

Examples

```
## Not run:
vlaanderen <- get_feature_wfs(
  wfs = paste0(
    "https://eservices.minfin.fgov.be/",
    "arcgis/services/R2C/Regions/MapServer/WFSServer"
  ),
  layername = "regions",
  crs = "EPSG:31370",
  filter = paste0(
    "<Filter><PropertyIsEqualTo><PropertyName>",
    "regions:NameDUT</PropertyName><Literal>'Vlaams Gewest'",
    "</Literal></PropertyIsEqualTo></Filter>"
  )
)

## End(Not run)
```

scalefactor_lcc *Return the scale factor for a specific parallel in a Lambert Conic Conformal projection*

Description

The scale factor is the distance distortion. In this case (a conic conformal projection) it holds in any direction.

Usage

```
scalefactor_lcc(par_deg, par1_deg, par2_deg)
```

Arguments

par_deg	Numeric vector. The latitude in decimal degrees of one or more parallels for which to do the calculation.
par1_deg	Latitude of the first standard parallel of the LCC projection in decimal degrees.
par2_deg	Latitude of the second standard parallel of the LCC projection in decimal degrees.

Details

The function applies to normal LCC projections, not the oblique ones. The applied formulas are for the sphere, hence approximate.

Value

Numeric vector, same length as par_deg.

Author(s)

Floris Vanderhaeghe, https://github.com/florisvdh

References

Snyder J.P. (1987). Map Projections - A Working Manual. U.S. Geological Survey Professional Paper, Nr. 1395. United States Government Printing Office, Washington, 397 p.

See Also

Other functions to explore properties of coordinate reference systems: scalefactor_tcyl()

Examples

```
sf::st_crs(31370)
x <- seq(49.5, 52.5, 0.1)
y <- scalefactor_lcc(x, 51.1666672333333, 49.8333339)
data.frame(latitude = x, scalefactor = y)
```

`scalefactor_tcyl` *Return the scale factor for the angular distance from the central meridian in an (unscaled) transverse cylindrical projection*

Description

The scale factor is the distance distortion in the direction of the meridians. In the case of the Transverse Mercator projection (a cylindrical conformal projection) it holds in any direction.

Usage

```
scalefactor_tcyl(ang_dist)
```

Arguments

`ang_dist`	Numeric vector. One or more angular distances from the central meridian, in decimal degrees.

Details

The formulas applied are for the sphere, hence approximate.

The best known map projection system in this context is the Universal Transverse Mercator (UTM), which superposes a central meridian scale factor of 0.9996.

Value

Numeric vector, same length as `ang_dist`.

Author(s)

Floris Vanderhaeghe, https://github.com/florisvdh

References

Iliffe J. & Lott R. (2008). Datums and Map Projections. For Remote Sensing, GIS and Surveying. 2nd edn. Whittles Publishing, Caithness, UK, 208 p.

See Also

Other functions to explore properties of coordinate reference systems: `scalefactor_lcc()`

Examples

```
x <- seq(0, 3, 0.1)
y <- scalefactor_tcyl(x)
data.frame(ang_dist = x, scalefactor = y)
```

Index