Package 'arcpullr'

February 23, 2021

Type Package

Title Pull Data from an 'ArcGIS REST' API

Version 0.1.2

License GPL-3

Encoding UTF-8

LazyData true

Description Functions to efficiently query 'ArcGIS REST' APIs <https://developers.arcgis.com/rest/>. Both spatial and SQL queries can be used to retrieve data. Simple Feature (sf) objects are utilized to perform spatial queries. This package was neither produced nor is maintained by Esri.

Depends R (>= 3.6.0)

Imports httr (>= 1.4.1), jsonlite (>= 1.6.1), dplyr (>= 1.0.2), ggplot2 (>= 3.3.0), sf (>= 0.9.7), tidyr (>= 1.0.2), rlang (>= 0.4.7)

RoxygenNote 7.1.1

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Paul Frater [aut, cre] (<https://orcid.org/0000-0002-7237-6563>), Zac Driscoll [aut] (<https://orcid.org/0000-0002-8233-0980>)

Maintainer Paul Frater <paul.frater@wisconsin.gov>

Repository CRAN

Date/Publication 2021-02-23 09:40:16 UTC

R topics documented:

`arcpullr-package` *arcpullr*

Description

A package for pulling spatial data from an ArcGIS REST API

Details

The role of the arcpullr package is simple...to pull spatial data from an ArcGIS REST API. These APIs are housed by various different agencies, organizations, entitites, etc., but allow a consistent format for storing and retrieving spatial data

`get_spatial_layer`

This function makes up the core of the package. It allows users to pull spatial data given a URL of an ArcGIS REST API. There are many additional query parameters that can (and probably should) be added; however, we've simplified many of these out for you with the functions below.

`get_layer_by_spatial` **family of functions**

These functions allow you to pull layers using a spatial query. The abstract syntax is wrapped into the functions, so all you have to do is pass these functions an sf object of the spatial area, line, or point you want to query by. These functions include get_layer_by_poly, get_layer_by_point, get_layer_by_line, get_layer_by_multipoint, and get_layer_by_envelope. It should be fairly obvious what type of spatial layer each function takes with the exception of get_layer_by_envelope except that it isn't particularly useful for a single point.

Helper functions

There are a few utility functions to help you along the way. The first is `plot_layer`, which is a useful way to plot the spatial layer you've tried to pull just to make sure it works. If you want fancier maps you'd be better served with ggplot2 or tmaps, though.

Other helpers include the `sf_objects` functions, which allow you to easily create sf points, lines, and polygons with a few coordinates.

Lastly, there is a `sql_where` function to help aid in building more complex SQL WHERE clauses used to query by the where argument in the retrieval functions above.

format_coords *Convert coordinates from an 'sf' object to formatted well-known text*

Description

Use this function to convert the coordinates of a sf polygon object to a string of well known text. The output can be passed to an ArcGIS REST API to perform a spatial query.

Usage

```
format_polygon_coords(sf_obj)

format_line_coords(sf_obj)

format_multipoint_coords(sf_obj)

format_point_coords(sf_obj)

format_envelope_coords(sf_obj)

format_coords(sf_obj, geom_type)
```

Arguments

sf_obj	An sf object
geom_type	Either "points", "paths", or "rings". Choose wisely

Details

Spatial queries from an ArcGIS REST API require specific text inputs formatted in a way called well-known text (WKT). ArcGIS REST APIs have their own syntax for how the text is taken. These functions will format sf objects in the correct way to be able to make spatial queries from a ArcGIS REST API

Value

String of well known text

Examples

```
mke_polygon_coords <- format_polygon_coords(mke_county)
```

get_layers_by_spatial *Retrieve ArcGIS REST API spatial layer by spatial query*

Description

These functions are wrappers around get_spatial_layer that are specialized for querying by a spatial layer. They will make a POST request to the query URL which returns data (if available) based on the appropriate spatial feature (geometry) and relationship (sp_rel).

Usage

```
get_layer_by_poly(url, geometry, sp_rel = "esriSpatialRelContains", ...)

get_layer_by_line(url, geometry, sp_rel = "esriSpatialRelIntersects", ...)

get_layer_by_point(url, geometry, sp_rel = "esriSpatialRelIntersects", ...)

get_layer_by_multipoint(
  url,
  geometry,
  sp_rel = "esriSpatialRelIntersects",
  ...
)

get_layer_by_envelope(url, geometry, sp_rel = "esriSpatialRelIntersects", ...)

get_layer_by_spatial(
  url,
  geometry,
  geom_type,
  sp_ref = NULL,
  sp_rel = "esriSpatialRelIntersects",
  ...
)
```

Arguments

url	A character string of the url for the layer to pull
geometry	An sf object used for the spatial query
sp_rel	Character. The type of relationship to query by.
...	Additional arguements to pass to get_spatial_layer
geom_type	A character of the geometry type to be used. This param is automatically specified in all get_layer_by_* functions except get_spatial_layer
sp_ref	The spatial reference value

Value

An object of class "sf" of the appropriate layer

Examples

```
base_wdnr_url <- "https://dnrmaps.wi.gov/arcgis/rest/services/"
hydro_path <- "WT_SWDV/WT_Inland_Water_Resources_WTM_Ext_v2/MapServer/2"
hydro_url <- paste0(base_wdnr_url, hydro_path)
mke_waters <- get_layer_by_poly(url = hydro_url, mke_county)
```

---

get_sf_crs *Return CRS value of an sf object*

Description

Return CRS value of an sf object

Usage

```
get_sf_crs(sf_obj)
```

Arguments

sf_obj	An object of class sf

Value

A numeric value referring to the coordinate reference system

Examples

```
get_sf_crs(iceland)
```

---

get_spatial_layer *Retrieve a spatial layer from an ArcGIS REST API*

Description

This function retrieves spatial layers present an ArcGIS REST services API located at.

Usage

```
get_spatial_layer(
  url,
  out_fields = c("*"),
  where = "1=1",
  token = "",
  sf_type = NULL,
  ...
)
```

Arguments

url	A character string of the url for the layer to pull
out_fields	A character string of the fields to pull for each layer
where	A character string of the where condition. Default is 1=1
token	A character string of the token (if needed)
sf_type	A character string specifying the layer geometry to convert to sf ("esriGeometryPolygon", "esriGeometryPoint", "esriGeometryPolyline"), if NULL (default) the server will take its best guess
...	Additional arguements to pass to the ArcGIS REST POST request

Details

This is the core function of this package. It retrieves spatial layers from an ArcGIS REST API designated by the URL. Additional querying features can be passed such as a SQL WHERE statement (`where` argument) as well as spatial queries or any other types of queries that the ArcGIS REST API accepts using `...`. However, for easier spatial querying see `get_layers_by_spatial`.

All of the querying parameters are sent via a POST request to the URL, so if there are issues with passing additional parameters via `...` first determine how they fit into the POST request and make adjustments as needed. This syntax can be tricky if you're not used to it.

Value

An object of class "sf" of the appropriate layer

Examples

```
## Not run:
# lava flows on Reykjanes (pr. 'rake-yah-ness') peninsula in Iceland
base_url <- "https://arcgisserver.isor.is:6443/arcgis/rest/services"
reykjanes_path <- "vi/uttekt_eldstodva_allt_2/MapServer/5"
reykjanes_url <- paste(base_url, reykjanes_path, sep = "/")
lava_flows <- get_spatial_layer(reykjanes_url)
plot_layer(lava_flows, outline_poly = reykjanes)
plot_layer(lava_flows, outline_poly = reykjanes) +
  ggplot2::geom_sf(data = iceland_poly, fill = NA)

## End(Not run)
```

plot_layer *Plot a spatial layer*

Description

This function plots a spatial layer as returned from `get_spatial_layer`.

Usage

```
plot_layer(sf_data, outline_poly = NULL, plot_pkg = "ggplot")
```

Arguments

`sf_data`	An sf object as returned from `get_spatial_layer`
`outline_poly`	An sf polygon to outline `sf_data` for context, or `NULL`
`plot_pkg`	Character. The plotting environment to use. Either "ggplot" (default) or "base"

Value

Either a `ggplot` object or a plot

Examples

```
## Not run:
plot_layer(iceland)
plot_layer(mke_county, outline_poly = wis_poly)
g <-
   plot_layer(wi_counties) +
   ggplot2::geom_sf(data = portage_county, fill = "red") +
   ggplot2::geom_sf(data = mke_county, fill = "red")
g

## End(Not run)
```

sf_example_polys *Various example sf polygons*

Description

These are sf polygons that are used for examples throughout the package

Usage

```
iceland

mke_county

portage_county

reykjanes

wis_counties

wis_poly
```

Format

An object of class sf and data.frame:

An object of class `sf` (inherits from `data.frame`) with 1 rows and 3 columns.

An object of class `sf` (inherits from `data.frame`) with 1 rows and 3 columns.

An object of class `sf` (inherits from `data.frame`) with 1 rows and 2 columns.

An object of class `sf` (inherits from `data.frame`) with 72 rows and 3 columns.

An object of class `sf` (inherits from `data.frame`) with 1 rows and 2 columns.

Source

`map_data`

`sf_objects` *Create sf objects from coordinates*

Description

These are simple wrapper functions for creating sf objects from points

Usage

```
sf_lines(..., crs = 4326)

sf_point(..., crs = 4326)

sf_points(..., crs = 4326)

sf_polygon(..., crs = 4326)
```

Arguments

...	The coordinates of the object
crs	The coordinate reference system. Defaults to 4326

Value

An sf object of the appropriate type

Examples

```
pt_a <- c(-90, 45)
pt_b <- c(-89, 44)
sf_pt <- sf_points(pt_a)
sf_lines <- sf_lines(pt_a, pt_b)
```

---

sql_where *Format a SQL where clause from arguments*

Description

This function will create a where statement that is compatible with get_spatial_layer). This statement can then be passed to the where argument in this function.

Usage

```
sql_where(..., rel_op = "=")
```

Arguments

...	Named objects to be queried by
rel_op	Character. The relational operator in the SQL clause (i.e. "=", "IN", "NOT IN", etc.). If a single rel_op is provide with multiple ... parameters then it will be recycled length(...) times.

Value

A character string that can be passed to the where argument of get_spatial_layer

Examples

```
## Not run:
wbics <- sql_where(WATERBODY_WBIC = c(805400, 804600), rel_op = "IN")
base_wdnr_url <- "https://dnrmaps.wi.gov/arcgis/rest/services/"
hydro_path <- "WT_SWDV/WT_Inland_Water_Resources_WTM_Ext_v2/MapServer/3"
hydro_url <- paste0(base_wdnr_url, hydro_path)
lakes <- get_spatial_layer(url = hydro_url, where = wbics)
plot_layer(lakes)

## End(Not run)
```

Index