Package 'bcmaps'

April 29, 2020

Title Map Layers and Spatial Utilities for British Columbia

Version 0.18.1

Description Provides access to various spatial layers for B.C., such as administrative boundaries, natural resource management boundaries, etc. All layers are imported from the 'bcmapsdata' package as 'sf' or 'Spatial' objects through function calls in this package. All layers are in B.C. 'Albers' equal-area projection <http://spatialreference.org/ref/epsg/nad83-bc-albers/>, which is the B.C. government standard.

License Apache License (== 2.0) | file LICENSE

URL https://github.com/bcgov/bcmaps

BugReports https://github.com/bcgov/bcmaps/issues

Depends sf (>= 0.9)

Imports httr (>= 1.3.1), methods, rappdirs (>= 0.3.1), stats, utils

Suggests bcmapsdata (>= 0.3.0), future.apply (>= 1.2.0), future (>= 1.12.0), ggplot2 (>= 3.0), glue (>= 1.1.1), knitr, raster (>= 2.5-8), rgdal (>= 1.2-13), rgeos (>= 0.3-25), rmarkdown, sp (>= 1.2-5), lwgeom (>= 0.2-2), testthat (>= 2.1.0)

VignetteBuilder knitr

Additional_repositories https://bcgov.github.io/drat

LazyData true

RoxygenNote 7.1.0

Encoding UTF-8

NeedsCompilation no

Author Andy Teucher [aut, cre],
Stephanie Hazlitt [aut],
Sam Albers [aut],
Province of British Columbia [cph]

Maintainer Andy Teucher <andy.teucher@gov.bc.ca>

Repository CRAN

Date/Publication 2020-04-29 10:30:10 UTC

R topics documented:

add_license_header . . . 3
airzones . . . 3
available_layers . . . 4
bcmaps . . . 4
bc_area . . . 5
bc_bbox . . . 5
bc_bound . . . 6
bc_bound_hres . . . 7
bc_cities . . . 7
bc_neighbours . . . 8
bec . . . 9
bec_colours . . . 9
combine_nr_rd . . . 10
ecoprovinces . . . 10
ecoregions . . . 11
ecosections . . . 12
fix_geo_problems . . . 13
get_big_data . . . 13
get_layer . . . 14
get_poly_attribute . . . 15
gw_aquifers . . . 16
hydrozones . . . 16
make_shortcuts . . . 17
municipalities . . . 18
nr_areas . . . 19
nr_districts . . . 19
nr_regions . . . 20
raster_by_poly . . . 21
regional_districts . . . 22
self_union . . . 23
summarize_raster_list . . . 24
transform_bc_albers . . . 24
tsa . . . 25
watercourses_15M . . . 26
watercourses_5M . . . 26
water_districts . . . 27
water_precincts . . . 28
wsc_drainages . . . 28

Index **30**

add_license_header *Add the boilerplate Apache header to the top of a source code file*

Description

Add the boilerplate Apache header to the top of a source code file

Usage

```
add_license_header(
  file,
  year = format(Sys.Date(), "%Y"),
  copyright_holder = "Province of British Columbia"
)
```

Arguments

file	Path to the file
year	The year the license should apply (Default current year)
copyright_holder	Copyright holder (Default "Province of British Columbia")

airzones *British Columbia Air Zones*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
airzones(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::airzones for details.

Value

The spatial layer of airzones in the desired class

Examples

```
## Not run:
my_layer <- airzones()
my_layer_sp <- airzones(class = 'sp')

## End(Not run)
```

`available_layers` *List available data layers*

Description

A data.frame of all available layers in the bcmaps package. This drawn directly from the bcmapsdata package and will therefore be the most current list layers available.

Usage

```
available_layers()
```

Value

A `data.frame` of layers, with titles, and a `shortcut_function` column denoting whether or not a shortcut function exists that can be used to return the layer. If `TRUE`, the name of the shortcut function is the same as the `layer_name`. A value of `FALSE` in this column means the layer is available via `get_data()` but there is no shortcut function for it.

A value of `FALSE` in the `local` column means that the layer is not stored in the bcmapsdata package but will be downloaded from the internet and cached on your hard drive.

Examples

```
## Not run:
available_layers()

## End(Not run)
```

`bcmaps` *bcmaps: A data package providing various map layers for British Columbia*

Description

Various layers of B.C., including administrative boundaries, natural resource management boundaries, etc. All layers are available as both sf and Spatial objects, and are in BC Albers equal-area projection, which is the B.C. government standard. The layers are sourced from the British Columbia and Canadian government under open licenses, including DataBC, the Government of Canada Open Data Portal, and Statistics Canada. Each layer's individual help page contains a section describing the source for the data.

bc_area *The size of British Columbia*

Description

Total area, Land area only, or Freshwater area only, in the units of your choosing.

Usage

```
bc_area(what = "total", units = "km2")
```

Arguments

what	Which part of BC? One of 'total' (default), 'land', or 'freshwater'.
units	One of 'km2' (square kilometres; default), 'm2' (square metres), 'ha' (hectares), 'acres', or 'sq_mi' (square miles)

Details

The sizes are from Statistics Canada

Value

The area of B.C. in the desired units (numeric vector).

Examples

```
## With no arguments, gives the total area in km^2:
bc_area()

## Get the area of the land only, in hectares:
bc_area("land", "ha")
```

bc_bbox *Get an extent/bounding box for British Columbia*

Description

Get an extent/bounding box for British Columbia

Usage

```
bc_bbox(class = c("sf", "sp", "raster"), crs = 3005)
```

Arguments

class	"sf", "sp", or "raster"
crs	coordinate reference system: integer with the EPSG code, or character with proj4string. Default 3005 (BC Albers).

Value

an object denoting a bounding box of British Columbia, of the corresponding class specified in class. The coordinates will be in lat-long WGS84 (epsg:4326).

Examples

```
if (requireNamespace("bcmapsdata", quietly = TRUE)) {
  bc_bbox("sf")
  bc_bbox("sp")
  bc_bbox("raster")
}
```

---

bc_bound *BC Boundary*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
bc_bound(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::bc_bound for details.

Value

The spatial layer of bc_bound in the desired class

Examples

```
## Not run:
my_layer <- bc_bound()
my_layer_sp <- bc_bound(class = 'sp')

## End(Not run)
```

bc_bound_hres *BC Boundary - High Resolution*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
bc_bound_hres(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::bc_bound_hres for details.

Value

The spatial layer of bc_bound_hres in the desired class

Examples

```
## Not run:
my_layer <- bc_bound_hres()
my_layer_sp <- bc_bound_hres(class = 'sp')

## End(Not run)
```

bc_cities *BC Major Cities Points 1:2,000,000 (Digital Baseline Mapping)*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
bc_cities(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::bc_cities for details.

Value

The spatial layer of bc_cities in the desired class

Examples

```
## Not run:
my_layer <- bc_cities()
my_layer_sp <- bc_cities(class = 'sp')

## End(Not run)
```

---

bc_neighbours *Boundary of British Columbia, provinces/states and the portion of the Pacific Ocean that borders British Columbia*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
bc_neighbours(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::bc_neighbours for details.

Value

The spatial layer of bc_neighbours in the desired class

Examples

```
## Not run:
my_layer <- bc_neighbours()
my_layer_sp <- bc_neighbours(class = 'sp')

## End(Not run)
```

bec *British Columbia BEC Map*

Description

The current and most detailed version of the approved corporate provincial digital Biogeoclimatic Ecosystem Classification (BEC) Zone/Subzone/Variant/Phase map (version 10, August 31st, 2016).

Usage

```
bec(class = c("sf", "sp"), ...)
```

Arguments

`class`	class of object to import; one of `"sf"` (default) or `"sp"`.
`...`	arguments passed on to get_big_data

Format

An `sf` or `Spatial` polygons object with B.C.'s Biogeoclimatic Ecosystem Classification (BEC) Zone/Subzone/Variant/Phase map

Source

Original data from the B.C. Data Catalogue, under the Open Government Licence - British Columbia.

bec_colours *Biogeoclimatic Zone Colours*

Description

Standard colours used to represent Biogeoclimatic Zone colours to be used in plotting.

Usage

```
bec_colours()

bec_colors()
```

Value

named vector of hexadecimal colour codes. Names are standard abbreviations of Zone names.

Examples

```
## Not run:
if (require("bcmapsdata") && #' require(sf) && require(ggplot2)) {
 bec <- bec()
 ggplot() +
   geom_sf(data = bec[bec$ZONE %in% c("BG", "PP"),],
           aes(fill = ZONE, col = ZONE)) +
   scale_fill_manual(values = bec_colors()) +
   scale_colour_manual(values = bec_colours())
}

## End(Not run)
```

`combine_nr_rd` *Combine Northern Rockies Regional Municipality with Regional Districts*

Description

Combine Northern Rockies Regional Municipality with Regional Districts

Usage

```
combine_nr_rd(class = c("sf", "sp"))
```

Arguments

`class`	The class of the layer returned. Can be either `"sf"` (default) or `"sp"`

Value

A layer where the Northern Rockies Regional Municipality has been combined with the Regional Districts to form a full provincial coverage.

`ecoprovinces` *British Columbia Ecoprovinces*

Description

You must have the `bcmapsdata` package installed to use this function.

Usage

```
ecoprovinces(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::ecoprovinces for details.

Value

The spatial layer of ecoprovinces in the desired class

Examples

```
## Not run:
my_layer <- ecoprovinces()
my_layer_sp <- ecoprovinces(class = 'sp')

## End(Not run)
```

---

ecoregions *British Columbia Ecoregions*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
ecoregions(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::ecoregions for details.

Value

The spatial layer of ecoregions in the desired class

Examples

```
## Not run:
my_layer <- ecoregions()
my_layer_sp <- ecoregions(class = 'sp')

## End(Not run)
```

`ecosections` *British Columbia Ecosections*

Description

You must have the `bcmapsdata` package installed to use this function.

Usage

```
ecosections(class = "sf")
```

Arguments

`class`	what class you want the object in? `"sf"` (default) or `"sp"`.

Details

Type `?bcmapsdata::ecosections` for details.

Value

The spatial layer of `ecosections` in the desired class

Examples

```
## Not run:
my_layer <- ecosections()
my_layer_sp <- ecosections(class = 'sp')

## End(Not run)
```

fix_geo_problems *Check and fix polygons that self-intersect, and sometimes can fix orphan holes*

Description

For `sf` objects, uses `sf::st_make_valid`. Otherwise, uses the common method of buffering by zero.

Usage

```
fix_geo_problems(obj, tries = 5)
```

Arguments

`obj`	The SpatialPolygons* or sf object to check/fix
`tries`	The maximum number of attempts to repair the geometry. Ignored for `sf` objects.

Details

`fix_self_intersect` has been removed and will no longer work. Use `fix_geo_problems` instead

Value

The SpatialPolygons* or `sf` object, repaired if necessary

get_big_data *Download a large data file*

Description

Download a large data file

Usage

```
get_big_data(
  what,
  class = c("sf", "sp"),
  release = "latest",
  force = FALSE,
  ask = TRUE
)
```

Arguments

what	The name of the object to download
class	class of object to import; one of "sf" (default) or "sp".
release	Specific version of bcmapsdata to get the desired dataset from. Default "latest"
force	Force downloading and overwriting existing dataset. Default FALSE
ask	Ask whether or not to write to the default data directory for bcmaps. Default TRUE

---

get_layer *Get a B.C. spatial layer*

Description

Get a B.C. spatial layer

Usage

```
get_layer(layer, class = c("sf", "sp"), ...)
```

Arguments

layer	the name of the layer. The list of available layers can be obtained by running available_layers()
class	The class of the layer returned. Can be either "sf" (default) or "sp"
...	arguments passed on to get_big_data if the layer needs to be downloaded. Ignored if the layer is available locally in bcmapsdata.

Value

the layer requested

Examples

```
## Not run:
 get_layer("bc_bound_hres")

 # As a "Spatial" (sp) object
 get_layer("watercourses_15M")

## End(Not run)
```

get_poly_attribute *Get or calculate the attribute of a list-column containing nested dataframes.*

Description

For example, `self_union` produces a `SpatialPolygonsDataFrame` that has a column called `union_df`, which contains a `data.frame` for each polygon with the attributes from the constituent polygons.

Usage

```
get_poly_attribute(x, col, fun, ...)
```

Arguments

`x`	the list-column in the (SpatialPolygons)DataFrame that contains nested data.frames
`col`	the column in the nested data frames from which to retrieve/calculate attributes
`fun`	function to determine the resulting single attribute from overlapping polygons
`...`	other parameters passed on to `fun`

Value

An atomic vector of the same length as x

Examples

```
if (require(sp)) {
  p1 <- Polygon(cbind(c(2,4,4,1,2),c(2,3,5,4,2)))
  p2 <- Polygon(cbind(c(5,4,3,2,5),c(2,3,3,2,2)))
  ps1 <- Polygons(list(p1), "s1")
  ps2 <- Polygons(list(p2), "s2")
  spp <- SpatialPolygons(list(ps1,ps2), 1:2)
  df <- data.frame(a = c(1, 2), b = c("foo", "bar"),
                   c = factor(c("high", "low"), ordered = TRUE,
                              levels = c("low", "high")),
                   stringsAsFactors = FALSE)
  spdf <- SpatialPolygonsDataFrame(spp, df, match.ID = FALSE)
  plot(spdf, col = c(rgb(1, 0, 0,0.5), rgb(0, 0, 1,0.5)))
  unioned_spdf <- self_union(spdf)
  get_poly_attribute(unioned_spdf$union_df, "a", sum)
  get_poly_attribute(unioned_spdf$union_df, "c", max)
}
```

gw_aquifers *British Columbia's developed ground water aquifers*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
gw_aquifers(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::gw_aquifers for details.

Value

The spatial layer of gw_aquifers in the desired class

Examples

```
## Not run:
my_layer <- gw_aquifers()
my_layer_sp <- gw_aquifers(class = 'sp')

## End(Not run)
```

hydrozones *Hydrologic Zone Boundaries of British Columbia*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
hydrozones(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::hydrozones for details.

Value

The spatial layer of hydrozones in the desired class

Examples

```
## Not run:
my_layer <- hydrozones()
my_layer_sp <- hydrozones(class = 'sp')

## End(Not run)
```

---

make_shortcuts *Make shortcut functions for data objects in bcmapsdata*

Description

This generates a shortcuts.R file in the R directory, with function definitions and roxygen blocks for each data object in bcmapsdata. This ensures that each data object in bcmapsdata can be accessed directly from bcmaps by a function such as bc_bound(), or airzones("sp").

Usage

```
make_shortcuts(file = "R/shortcuts.R")
```

Arguments

file	the R file where the shortcut file is. Default "R/shortcuts.R"

Details

Run this function each time you add a new data object.

Value

TRUE (invisibly)

Examples

```
## Not run:
make_shortcut()

## End(Not run)
```

`municipalities` *British Columbia Municipalities*

Description

You must have the `bcmapsdata` package installed to use this function.

Usage

```
municipalities(class = "sf")
```

Arguments

`class`	what class you want the object in? `"sf"` (default) or `"sp"`.

Details

Type `?bcmapsdata::municipalities` for details.

Value

The spatial layer of `municipalities` in the desired class

See Also

`combine_nr_rd()` to combine Regional Districts and the Northern Rockies Regional Municipality into one layer

Examples

```
## Not run:
my_layer <- municipalities()
my_layer_sp <- municipalities(class = 'sp')

## End(Not run)
```

nr_areas *British Columbia Natural Resource (NR) Areas*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
nr_areas(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::nr_areas for details.

Value

The spatial layer of nr_areas in the desired class

Examples

```
## Not run:
my_layer <- nr_areas()
my_layer_sp <- nr_areas(class = 'sp')

## End(Not run)
```

nr_districts *British Columbia Natural Resource (NR) Districts*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
nr_districts(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type `?bcmapsdata::nr_districts` for details.

Value

The spatial layer of `nr_districts` in the desired class

Examples

```
## Not run:
my_layer <- nr_districts()
my_layer_sp <- nr_districts(class = 'sp')

## End(Not run)
```

---

nr_regions *British Columbia Natural Resource (NR) Regions*

Description

You must have the `bcmapsdata` package installed to use this function.

Usage

```
nr_regions(class = "sf")
```

Arguments

`class`	what class you want the object in? `"sf"` (default) or `"sp"`.

Details

Type `?bcmapsdata::nr_regions` for details.

Value

The spatial layer of `nr_regions` in the desired class

Examples

```
## Not run:
my_layer <- nr_regions()
my_layer_sp <- nr_regions(class = 'sp')

## End(Not run)
```

raster_by_poly *Overlay a SpatialPolygonsDataFrmae or sf polygons layer on a raster layer and clip the raster to each polygon. Optionally done in parallel*

Description

Overlay a SpatialPolygonsDataFrmae or sf polygons layer on a raster layer and clip the raster to each polygon. Optionally done in parallel

Usage

```
raster_by_poly(
  raster_layer,
  poly,
  poly_field,
  summarize = FALSE,
  parallel = FALSE,
  future_strategy = NULL,
  workers = NULL,
  ...
)
```

Arguments

raster_layer	the raster layer
poly	a SpatialPolygonsDataFrame layer or sf layer
poly_field	the field on which to split the SpatialPolygonsDataFrame
summarize	Should the function summarise the raster values in each polygon to a vector? Default FALSE
parallel	process in parallel? Default FALSE.
future_strategy	the strategy to use in future::plan() for parallel computation. Default NULL respects if user has already set a plan using future::plan() or an option, otherwise uses "multiprocess".
workers	number of workers if doing parallel. Default NULL uses the default of the future strategy chosen (usually future::availableCores()).
...	passed on to future::plan()

Value

a list of RasterLayers if summarize = FALSE otherwise a list of vectors.

`regional_districts` *British Columbia Regional Districts*

Description

You must have the `bcmapsdata` package installed to use this function.

Usage

```
regional_districts(class = "sf")
```

Arguments

`class`	what class you want the object in? `"sf"` (default) or `"sp"`.

Details

Type `?bcmapsdata::regional_districts` for details.

Value

The spatial layer of `regional_districts` in the desired class

See Also

`combine_nr_rd()` to combine Regional Districts and the Northern Rockies Regional Municipality into one layer

Examples

```
## Not run:
my_layer <- regional_districts()
my_layer_sp <- regional_districts(class = 'sp')

## End(Not run)
```

self_union *Union a SpatialPolygons* object with itself to remove overlaps, while retaining attributes*

Description

The IDs of source polygons are stored in a list-column called `union_ids`, and original attributes (if present) are stored as nested dataframes in a list-column called `union_df`

Usage

```
self_union(x)
```

Arguments

x	A SpatialPolygons or SpatialPolygonsDataFrame object

Value

A SpatialPolygons or SpatialPolygonsDataFrame object

Examples

```
if (require(sp)) {
  p1 <- Polygon(cbind(c(2,4,4,1,2),c(2,3,5,4,2)))
  p2 <- Polygon(cbind(c(5,4,3,2,5),c(2,3,3,2,2)))

  ps1 <- Polygons(list(p1), "s1")
  ps2 <- Polygons(list(p2), "s2")

  spp <- SpatialPolygons(list(ps1,ps2), 1:2)

  df <- data.frame(a = c("A", "B"), b = c("foo", "bar"),
                   stringsAsFactors = FALSE)

  spdf <- SpatialPolygonsDataFrame(spp, df, match.ID = FALSE)

  plot(spdf, col = c(rgb(1, 0, 0,0.5), rgb(0, 0, 1,0.5)))

  unioned_spdf <- self_union(spdf)
  unioned_sp <- self_union(spp)
}
```

summarize_raster_list *Summarize a list of rasters into a list of numeric vectors*

Description

Summarize a list of rasters into a list of numeric vectors

Usage

```
summarize_raster_list(
  raster_list,
  parallel = FALSE,
  future_strategy = NULL,
  workers = NULL,
  ...
)
```

Arguments

raster_list	list of rasters
parallel	process in parallel? Default FALSE.
future_strategy	the strategy to use in future::plan() for parallel computation. Default NULL respects if user has already set a plan using future::plan() or an option, otherwise uses "multiprocess".
workers	number of workers if doing parallel. Default NULL uses the default of the future strategy chosen (usually future::availableCores()).
...	passed on to future::plan()

Value

a list of numeric vectors

transform_bc_albers *Transform a Spatial* object to BC Albers projection*

Description

Transform a Spatial* object to BC Albers projection

Usage

```
transform_bc_albers(obj)
```

Arguments

obj	The Spatial* or sf object to transform

Value

the Spatial* or sf object in BC Albers projection

---

tsa *British Columbia Timber Supply Areas and TSA Blocks*

Description

The spatial representation for a Timber Supply Area or TSA Supply Block: A Timber Supply Area is the primary unit for allowable annual cut (AAC) determination. A TSA Supply Block is a designated area within the TSA where the Ministry approves the allowable annual cuts.

Usage

```
tsa(class = c("sf", "sp"), ...)
```

Arguments

class	class of object to import; one of "sf" (default) or "sp".
...	arguments passed on to get_big_data

Format

An sf or Spatial polygons object with B.C.'s Timber Supply Areas and TSA Blocks

Details

Updated 2017-11-03

Source

Original data from the B.C. Data Catalogue, under the Open Government Licence - British Columbia.

watercourses_15M *British Columbia watercourses at 1:15M scale*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
watercourses_15M(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::watercourses_15M for details.

Value

The spatial layer of watercourses_15M in the desired class

Examples

```
## Not run:
my_layer <- watercourses_15M()
my_layer_sp <- watercourses_15M(class = 'sp')

## End(Not run)
```

watercourses_5M *British Columbia watercourses at 1:5M scale*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
watercourses_5M(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::watercourses_5M for details.

Value

The spatial layer of watercourses_5M in the desired class

Examples

```
## Not run:
my_layer <- watercourses_5M()
my_layer_sp <- watercourses_5M(class = 'sp')

## End(Not run)
```

---

water_districts *British Columbia's Water Management Districts*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
water_districts(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::water_districts for details.

Value

The spatial layer of water_districts in the desired class

Examples

```
## Not run:
my_layer <- water_districts()
my_layer_sp <- water_districts(class = 'sp')

## End(Not run)
```

water_precincts *British Columbia's Water Management Precincts*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
water_precincts(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type ?bcmapsdata::water_precincts for details.

Value

The spatial layer of water_precincts in the desired class

Examples

```
## Not run:
my_layer <- water_precincts()
my_layer_sp <- water_precincts(class = 'sp')

## End(Not run)
```

wsc_drainages *Water Survey of Canada Sub-Sub-Drainage Areas*

Description

You must have the bcmapsdata package installed to use this function.

Usage

```
wsc_drainages(class = "sf")
```

Arguments

class	what class you want the object in? "sf" (default) or "sp".

Details

Type `?bcmapsdata::wsc_drainages` for details.

Value

The spatial layer of `wsc_drainages` in the desired class

Examples

```
## Not run:
my_layer <- wsc_drainages()
my_layer_sp <- wsc_drainages(class = 'sp')

## End(Not run)
```

Index

add_license_header, 3
airzones, 3
available_layers, 4

bc_area, 5
bc_bbox, 5
bc_bound, 6
bc_bound_hres, 7
bc_cities, 7
bc_neighbours, 8
bcmaps, 4
bec, 9
bec_colors *(*bec_colours*)*, 9
bec_colours, 9

combine_nr_rd, 10
combine_nr_rd(), *18*, *22*

ecoprovinces, 10
ecoregions, 11
ecosections, 12

fix_geo_problems, 13
future::availableCores(), *21*, *24*
future::plan(), *21*, *24*

get_big_data, *9*, 13, *14*, *25*
get_layer, 14
get_poly_attribute, 15
gw_aquifers, 16

hydrozones, 16

make_shortcuts, 17
municipalities, 18

nr_areas, 19
nr_districts, 19
nr_regions, 20

option, *21*, *24*

raster_by_poly, 21
regional_districts, 22

self_union, 23
summarize_raster_list, 24

transform_bc_albers, 24
tsa, 25

water_districts, 27
water_precincts, 28
watercourses_15M, 26
watercourses_5M, 26
wsc_drainages, 28