Package ‘bcmaps’

March 10, 2021

Title  Map Layers and Spatial Utilities for British Columbia

Version  1.0.2

Description  Provides access to various spatial layers for B.C., such as administrative boundaries, natural resource management boundaries, etc. Most layers are imported from the ‘bcdata’ package as ‘sf’ or ‘Spatial’ objects through function calls in this package.

License  Apache License (== 2.0) | file LICENSE

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

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

Depends  sf (>= 0.9), R (>= 2.10)

Imports  bcdata (>= 0.2.0), httr (>= 1.3.1), methods, rappdirs (>= 0.3.1), progress, stats, utils, xml2, jsonlite (>= 1.7.0)

Suggests  knitr, rmarkdown, future.apply (>= 1.2.0), future (>= 1.12.0), ggplot2 (>= 3.0), glue (>= 1.1.1), raster (>= 2.5-8), rgdal (>= 1.2-13), rgeos (>= 0.3-25), sp (>= 1.2-5), lwgeom (>= 0.2-2), testthat (>= 2.1.0), withr (>= 2.3), stars (>= 0.4.3)

VignetteBuilder  knitr

LazyData  true

RoxygenNote  7.1.1

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  2021-03-09 23:40:03 UTC
R topics documented:

add_license_header ....................................................... 3
airzones ................................................................. 3
available_layers .......................................................... 4
bcmaps ................................................................. 5
bc_area ................................................................. 5
bc_bbox ................................................................. 6
bc_bound ................................................................. 7
bc_bound_hires ......................................................... 7
bc_cities ............................................................... 8
bc_neighbours .......................................................... 9
bec ................................................................. 10
bec_colours ............................................................ 10
cded ................................................................. 11
cded_raster ............................................................ 12
cded_stars ............................................................. 13
census_dissemination_area ........................................... 14
census_division ......................................................... 15
census_economic ......................................................... 15
census_metropolitan_area ............................................... 16
census_subdivision .................................................... 17
censustract ............................................................ 18
combine_nr_rd .......................................................... 18
delete_cache ............................................................ 19
ecoprovinces ............................................................ 20
ecoregions ............................................................... 21
ecosections .............................................................. 21
fix_geo_problems ......................................................... 22
fsa ................................................................. 23
get_big_data ............................................................. 23
get_layer ............................................................... 24
get_poly_attribute ........................................................ 25
gw_aquifers ............................................................. 26
health_chsa ............................................................. 26
health_ha ............................................................... 27
health_hsda ............................................................. 28
health_lha ............................................................. 29
hydrozones ............................................................. 29
make_shortcuts .......................................................... 30
mapsheets_250K ........................................................ 31
mapsheets_50K ........................................................ 32
municipalities ........................................................... 32
nr_areas ................................................................. 33
nr_districts ............................................................. 34
nr_regions .............................................................. 35
raster_by_poly .......................................................... 35
regional_districts ......................................................... 36
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

British Columbia Air Zones

Usage

airzones(class = "sf", ask = interactive(), force = FALSE)
Arguments

class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
force Should you force download the data?

Value

The spatial layer of airzones in the desired class

Source

bcdata::bcdc_get_data(record = 'e8eeefc4-2826-47bc-8430-85703d328516', resource = 'c495d082-b586-4df0-9e06-bd6b66a8acd9')

Examples

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

## Not run:
bc_bbox("sf")  
bc_bbox("sp")  
bc_bbox("raster")

## End(Not run)
bc_bound

BC Boundary

Description
BC Boundary

Usage
bc_bound(class = "sf", ask = interactive(), force = FALSE)

Arguments
class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? De-
faults to the value of interactive().
force Should you force download the data?

Value
The spatial layer of bc_bound in the desired class

Source
bcdata::bcdc_get_data('b9bd93e1-0226-4351-b943-05c6f80bd5da')

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
BC Boundary - High Resolution

Usage
bc_bound_hres(class = "sf", ask = interactive(), force = FALSE)
bc_cities

Arguments

- **class**: what class you want the object in? "sf" (default) or "sp".
- **ask**: Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
- **force**: Should you force download the data?

Value

The spatial layer of bc_bound_hres in the desired class

Source

```r
cbcdc_get_data(record = '30aeb5c1-4285-46c8-b60b-15b1a6f4258b', resource = '3d72cf36-ab53-4a2a-9988-a881c9c50966', layer = 'BC_Boundary_Terrestrial_Multipart')
```

Examples

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

---

**bc_cities**

*BC Major Cities Points*

Description

BC Major Cities Points

Usage

```r
bc_cities(class = "sf", ask = interactive(), force = FALSE)
```

Arguments

- **class**: what class you want the object in? "sf" (default) or "sp".
- **ask**: Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
- **force**: Should you force download the data?

Value

The spatial layer of bc_cities in the desired class
Source

bcdata::bcdc_get_data(record = 'b678c432-c5c1-4341-88db-0d6befa0c7f8', resource = '443dd858-2e37-4a8f-937a-f53359f16e64')

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

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

Usage

bc_neighbours(class = "sf", ask = interactive(), force = FALSE)

Arguments

class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
force Should you force download the data?

Value

The spatial layer of bc_neighbours in the desired class

Source

bcdata::bcdc_get_data('b9bd93e1-0226-4351-b943-05c6f80bd5da')

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

```r
## Not run:
if (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)
```

**cded**

*Canadian Digital Elevation Model (CDED)*

**Description**

Digital Elevation Model (DEM) for British Columbia produced by GeoBC. This data is the TRIM DEM converted to the Canadian Digital Elevation Data (CDED) format. The data consists of an ordered array of ground or reflective surface elevations, recorded in metres, at regularly spaced intervals. The spacing of the grid points is .75 arc seconds north/south. The data was converted into 1:50,000 grids for distribution. The scale of this modified data is 1:250,000 which was captured from the original source data which was at a scale of 1:20,000.

**Usage**

```r
cded( 
aoi = NULL,
tiles_50K = NULL,
.predicate = sf::st_intersects,
dest_vrt = tempfile(fileext = ".vrt"),
ask = interactive(),
check_tiles = TRUE
)
```

**Arguments**

- **aoi**
  Area of Interest. Currently supports sf and sp polygons, stars and raster objects.
- **tiles_50K**
  a character vector of 1:50,000 NTS mapsheets
- **.predicate**
  geometry predicate function used to find the mapsheets from your aoi. Default `sf::st_intersects`.
- **dest_vrt**
  The location of the vrt file. Defaults to a temporary file, but can be overridden if you’d like to save it for a project
- **ask**
  Should the function ask the user before downloading the data to a cache? Defaults to the value of `interactive()`.
check_tiles Should the tiles that you already have in your cache be checked to see if they need updating? Default TRUE. If you are running the same code frequently and are confident the tiles haven’t changed, setting this to FALSE will speed things up.

Value

path to a .vrt file of the cded tiles for the specified area of interest

Examples

```r
## Not run:
vic <- census_subdivision()$CENSUS_SUBDIVISION_NAME == "Victoria", ]
vic_cded <- cded(aoi = vic)
## End(Not run)
```

**cded_raster**  
Get Canadian Digital Elevation Model (CDED) as a raster object

**Description**

Get Canadian Digital Elevation Model (CDED) as a raster object

**Usage**

```r
cded_raster(
  aoi = NULL,
  tiles_50K = NULL,
  .predicate = sf::st_intersects,
  dest_vrt = tempfile(fileext = ".vrt"),
  check_tiles = TRUE,
  ...
)
```

**Arguments**

- **aoi**  
  Area of Interest. Currently supports sf and sp polygons, stars and raster objects.

- **tiles_50K**  
  a character vector of 1:50,000 NTS mapsheet tiles

- **.predicate**  
  geometry predicate function used to find the mapsheets from your aoi. Default `sf::st_intersects`.

- **dest_vrt**  
  The location of the vrt file. Defaults to a temporary file, but can be overridden if you’d like to save it for a project

- **check_tiles**  
  Should the tiles that you already have in your cache be checked to see if they need updating? Default TRUE. If you are running the same code frequently and are confident the tiles haven’t changed, setting this to FALSE will speed things up.

- **...**  
  Further arguments passed on to `raster::raster`
Value

a raster object of the cded tiles for the specified area of interest

Examples

```r
## Not run:
vic <- census_subdivision()[census_subdivision()$CENSUS_SUBDIVISION_NAME == "Victoria", ]
vic_cded <- cded_raster(aoi = vic)
## End(Not run)
```

Usage

```r
cded_stars(
  aoi = NULL,
  tiles_50K = NULL,
  .predicate = sf::st_intersects,
  dest_vrt = tempfile(fileext = ".vrt"),
  check_tiles = TRUE,
  ...
)
```

Arguments

- **aoi**: Area of Interest. Currently supports sf and sp polygons, stars and raster objects.
- **tiles_50K**: a character vector of 1:50,000 NTS mapsheet tiles
- **.predicate**: geometry predicate function used to find the mapsheets from your aoi. Default `sf::st_intersects`.
- **dest_vrt**: The location of the vrt file. Defaults to a temporary file, but can be overridden if you’d like to save it for a project
- **check_tiles**: Should the tiles that you already have in your cache be checked to see if they need updating? Default TRUE. If you are running the same code frequently and are confident the tiles haven’t changed, setting this to FALSE will speed things up.
- **...**: Further arguments passed on to `stars::read_stars`

Value

a stars object of the cded tiles for the specified area of interest
census_dissemination_area

Description

Current Census Dissemination Areas

Usage

census_dissemination_area(class = "sf", ask = interactive(), force = FALSE)

Arguments

class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? De-
defaults to the value of interactive().
force Should you force download the data?

Value

The spatial layer of census_dissemination_area in the desired class

Source

bcdata::bcdc_get_data(record = 'a091fd65-d682-4a24-8c0e-68de7c87e3a3', resource = 'a7fa66d4-0f95-4c58-861d-42d875e5b79f')

Examples

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

## End(Not run)
census_division

Current Census Division Boundaries

Description

Current Census Division Boundaries

Usage

census_division(class = "sf", ask = interactive(), force = FALSE)

Arguments

class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
force Should you force download the data?

Value

The spatial layer of census_division in the desired class

Source

bcdata::bcdc_get_data(record = 'ef17918a-597a-4012-8534-f8e71d8735b3',resource = '36b530c2-1de6-44a2-a6f6-c1fce36c53ed')

Examples

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

## End(Not run)

census_economic

Current Census Economic Region Boundaries

Description

Current Census Economic Region Boundaries

Usage

census_economic(class = "sf", ask = interactive(), force = FALSE)
census_metropolitan_area

Arguments

class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
force Should you force download the data?

Value

The spatial layer of census_economic in the desired class

Source

bcdata::bcdc_get_data(record = '1aebc451-a41c-496f-8b18-6f414cde93b7', resource = '3f0236cf-b1a1-4f1a-8e9e-86c2c3daff96')

Examples

## Not run:
my_layer <- census_economic()
my_layer_sp <- census_economic(class = 'sp')
## End(Not run)

census_metropolitan_area

Current Census Metropolitan Areas

Description

Current Census Metropolitan Areas

Usage

census_metropolitan_area(class = "sf", ask = interactive(), force = FALSE)

Arguments

class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
force Should you force download the data?

Value

The spatial layer of census_metropolitan_area in the desired class
census_subdivision

Source

bcdata::bcdc_get_data(record = 'a6fb34b7-0937-4718-8f1f-43dba2c0f407', resource = 'f129a965-363e-4d7e-8319-a4010e002fdb')

Examples

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

## End(Not run)

census_subdivision  Current Census Subdivision Boundaries

Description

Current Census Subdivision Boundaries

Usage

census_subdivision(class = "sf", ask = interactive(), force = FALSE)

Arguments

class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? De-
defaults to the value of interactive().
force Should you force download the data?

Value

The spatial layer of census_subdivision in the desired class

Source

bcdata::bcdc_get_data(record = '4c5618c6-38dd-4a62-a3de-9408b4974bb6', resource = '98bd1222-57bb-4504-92c2-4a5857ae7671')

Examples

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

## End(Not run)
census_tract

Current Census Tract Boundaries

Description

Current Census Tract Boundaries

Usage

census_tract(class = "sf", ask = interactive(), force = FALSE)

Arguments

class  what class you want the object in? "sf" (default) or "sp".
ask    Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
force  Should you force download the data?

Value

The spatial layer of census_tract in the desired class

Source

bcdata:::bcdc_get_data(record = '539aee5b-12f6-4934-9592-9b27acc827f8', resource = 'be76db6-0d4e-4906-a89b-564637f14db9')

Examples

## Not run:
my_layer <- census_tract()
my_layer_sp <- census_tract(class = 'sp')
## 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"))
**delete_cache**

Arguments

- **class**
  - what class you want the object in? "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.

---

**delete_cache**  
*View and delete cached files*

Description

View and delete cached files

Show the files you have in your cache

Usage

```r
delete_cache(files_to_delete = NULL)  
show_cached_files()
```

Arguments

- **files_to_delete**
  - An optional argument to specify which files or layers should be deleted from the cache. Defaults to deleting all files pausing for permission from user. If a subset of files are specified, the files are immediately deleted.

Value

- **delete_cache()**: A logical of whether the file(s) were successful deleted
- **show_cached_files()**: a data.frame with the columns:
  - file, the name of the file,
  - size_MB, file size in MB,
  - is_dir, is it a directory? If you have cached tiles from the cded() functions, there will be a row in the data frame showing the total size of the cded tiles cache directory.
  - modified, date and time last modified
ecoprovinces

Examples

## Not run:
## See which files you have
show_cached_files()

## Delete your whole cache
delete_cache()

## Specify which files are deleted
delete_cache(c('regional_districts.rds', 'bc_cities.rds'))

## End(Not run)

ecoprovinces  British Columbia Ecoprovinces

Description

British Columbia Ecoprovinces

Usage

ecoprovinces(class = "sf", ask = interactive(), force = FALSE)

Arguments

class  what class you want the object in? "sf" (default) or "sp".
ask    Should the function ask the user before downloading the data to a cache? De-
       faults to the value of interactive().
force  Should you force download the data?

Value

The spatial layer of ecoprovinces in the desired class

Source

bcdata::bcdc_get_data(record = '51832f47-efdf-4956-837a-45fc2c9032dd', resource = '811fcedb-1a53-4574-8149-454f4a740682')

Examples

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

## End(Not run)
ecoregions

Description

British Columbia Ecoregions

Usage

ecoregions(class = "sf", ask = interactive(), force = FALSE)

Arguments

class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
force Should you force download the data?

Value

The spatial layer of ecoregions in the desired class

Source

bcdata::bcdc_get_data(record = "80389e0-66da-4895-bd56-39a0dd64aa78", resource = "bd816a86-4f5e-4989-b1df-0b2f3f4a5f86")

Examples

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

ecosections

Description

British Columbia Ecosections

Usage

ecosections(class = "sf", ask = interactive(), force = FALSE)
Arguments

- **class**: what class you want the object in? "sf" (default) or "sp".
- **ask**: Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
- **force**: Should you force download the data?

Value

The spatial layer of ecossections in the desired class

Source

```r
bcdata::bcdc_get_data(record = 'ccc0f43-860d-4583-8ba4-e72d8379441e', resource = '6b6a3122-7a0b-4c0f-a72b-1e5c0e13d7e6')
```

Examples

```r
## Not run:
my_layer <- ecossections()
my_layer_sp <- ecossections(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

```r
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
**fso**

*British Columbia Forward Sortation Areas*

**Description**

British Columbia Forward Sortation Areas

**Usage**

`fso(class = "sf", ask = interactive(), force = FALSE)`

**Arguments**

- `class`: What class you want the object in? "sf" (default) or "sp".
- `ask`: Should the function ask the user before downloading the data to a cache? Defaults to the value of `interactive()`.
- `force`: Should you force download the data?

**Source**

`http://www12.statcan.gc.ca/census-recensement/2011/geo/bound-limit/files-fichiers/2016/lfsa000b16a_e.zip`

**Examples**

```r
## Not run:
my_layer <- fso()
my_layer_sp <- fso(class = 'sp')
## End(Not run)
```

---

**get_big_data**

*Download a large data file*

**Description**

Download a large data file

**Usage**

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

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"), ask = TRUE, force = FALSE, ...)

Arguments

layer the name of the layer. The list of available layers can be obtained by running available_layers()
class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().force Should you force download the data?
... arguments passed on to get_big_data if the layer needs to be downloaded from a bcmapsdata release.

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

```r
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

```r
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, 0.5), 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**

British Columbia’s developed ground water aquifers

**Usage**

```r
gw_aquifers(class = "sf", ask = interactive(), force = FALSE)
```

**Arguments**

- `class` what class you want the object in? "sf" (default) or "sp".
- `ask` Should the function ask the user before downloading the data to a cache? Defaults to the value of `interactive()`.
- `force` Should you force download the data?

**Value**

The spatial layer of `gw_aquifers` in the desired class

**Source**

`bcdata::bcdc_get_data(record = '099d69c5-1401-484d-9e19-c121ccb797c', resource = '8f421e3a-cdd3-4fab-8198-53ad6e9e2af2')`

**Examples**

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

---

**health_chsa**  
*Community Health Service Areas - CHSA*

**Description**

Community Health Service Areas - CHSA

**Usage**

```r
health_chsa(class = "sf", ask = interactive(), force = FALSE)
```
**health_ha**

**Arguments**

- **class**: what class you want the object in? "sf" (default) or "sp".
- **ask**: Should the function ask the user before downloading the data to a cache? Defaults to the value of `interactive()`.
- **force**: Should you force download the data?

**Value**

The spatial layer of `health_chsa` in the desired class

**Source**

```
bcdata::bcdc_get_data(record = '68f2f577-28a7-46b4-bca9-7e9770f2f357',resource = '59065b51-511a-4976-b77f-034168365273')
```

**Examples**

```r
## Not run:
my_layer <- health_chsa()
my_layer_sp <- health_chsa(class = 'sp')
## End(Not run)
```

---

**health_ha**

*Health Authority Boundaries*

**Description**

Health Authority Boundaries

**Usage**

```
health_ha(class = "sf", ask = interactive(), force = FALSE)
```

**Arguments**

- **class**: what class you want the object in? "sf" (default) or "sp".
- **ask**: Should the function ask the user before downloading the data to a cache? Defaults to the value of `interactive()`.
- **force**: Should you force download the data?

**Value**

The spatial layer of `health_ha` in the desired class

**Source**

```
bcdata::bcdc_get_data(record = '7bc6018f-bb4f-4e5d-845e-c529e3d1ac3b',resource = '93b79a3c-2da4-4fd4-b9c1-25870f752583')
```
## Description

Health Service Delivery Area Boundaries

## Usage

```r
health_hsda(class = "sf", ask = interactive(), force = FALSE)
```

## Arguments

- **class**: what class you want the object in? "sf" (default) or "sp".
- **ask**: Should the function ask the user before downloading the data to a cache? Defaults to the value of `interactive()`.
- **force**: Should you force download the data?

## Value

The spatial layer of `health_hsda` in the desired class

## Source

`bcdata::bcdc_get_data(record = '71c930b9-563a-46da-a10f-ead49ccbc390', resource = 'c5dad467-229b-4378-852b-ff92479a65b6')`

## Examples

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

# End(Not run)
```
**health_lha**

Local Health Area Boundaries

**Usage**

```r
health_lha(class = "sf", ask = interactive(), force = FALSE)
```

**Arguments**

- `class`: what class you want the object in? "sf" (default) or "sp".
- `ask`: Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
- `force`: Should you force download the data?

**Value**

The spatial layer of `health_lha` in the desired class

**Source**

`bcdata::bcdc_get_data(record = 'afdo21d9-7722-4410-b506-d394c6e74fc', resource = 'd6e951d3-5103-475a-8bb6-b4d275e6343f')`

**Examples**

```r
## Not run:
my_layer <- health_lha()
my_layer_sp <- health_lha(class = 'sp')
## End(Not run)
```

---

**hydrozones**

Hydrologic Zone Boundaries of British Columbia

**Description**

Hydrologic Zone Boundaries of British Columbia

**Usage**

```r
hydrozones(class = "sf", ask = interactive(), force = FALSE)
```
Arguments

class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
force Should you force download the data?

Value

The spatial layer of hydrozones in the desired class

Source

`bcdata:::bcdc_get_data(record = '\{329fd234-8835-4d44-9aaa-97c37bfc8d92\}', resource = '{baeb665e-85c7-4a7b-8e67-8b956785490a}')`

Examples

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

Description

This generates a shortcuts.R file in the R directory, with function definitions and roxygen blocks for each data object in bcmaps. This ensures that each data object 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)
## Description

NTS 250K Grid - Digital Baseline Mapping at 1:250,000 (NTS)

### Usage

```r
mapsheets_250K(class = "sf")
```

### Arguments

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

### Value

The spatial layer of `mapsheets_250K` in the desired class

### Source

https://open.canada.ca/data/en/dataset/055919c2-101e-4329-bfd7-1d0c333c0e62

## Examples

```r
## Not run:
my_layer <- mapsheets_250K()
my_layer_sp <- mapsheets_250K(class = 'sp')

## End(Not run)
```
municipalities

Description

British Columbia Municipalities

Usage

municipalities(class = "sf", ask = interactive(), force = FALSE)
Arguments

- **class**: what class you want the object in? "sf" (default) or "sp".
- **ask**: Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
- **force**: Should you force download the data?

Value

The spatial layer of municipalities in the desired class

Source

bcdata::bcdc_get_data(record = 'e3c3c580-996a-4668-8bc5-6aa7c7dc4932', resource = '25c95b07-5882-47ff-970d-36cb243b8355')

See Also

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

Examples

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

---

**nr_areas**

*British Columbia Natural Resource (NR) Areas*

Description

British Columbia Natural Resource (NR) Areas

Usage

```r
nr_areas(class = "sf", ask = interactive(), force = FALSE)
```

Arguments

- **class**: what class you want the object in? "sf" (default) or "sp".
- **ask**: Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
- **force**: Should you force download the data?
Value

The spatial layer of `nr_areas` in the desired class

Source

`bcdata::bcdc_get_data(record = 'c1861ba4-abb8-4947-b3e5-7f7c4d7257d5',resource = '4b317896-1a42-4c03-9dbd-bff996d5ea0c')`

Examples

```r
## 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

British Columbia Natural Resource (NR) Districts

Usage

```r
nr_districts(class = "sf", ask = interactive(), force = FALSE)
```

Arguments

- `class`: what class you want the object in? "sf" (default) or "sp".
- `ask`: Should the function ask the user before downloading the data to a cache? Defaults to the value of `interactive()`.
- `force`: Should you force download the data?

Value

The spatial layer of `nr_districts` in the desired class

Source

`bcdata::bcdc_get_data(record = '0bc73892-e41f-41d0-8d8e-828c16139337',resource = 'e6676e55-2a6f-4b2b-91ad-3caf291ac5d4')`

Examples

```r
## 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**
British Columbia Natural Resource (NR) Regions

**Usage**

```r
nr_regions(class = "sf", ask = interactive(), force = FALSE)
```

**Arguments**
- `class`: what class you want the object in? "sf" (default) or "sp".
- `ask`: Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
- `force`: Should you force download the data?

**Value**
The spatial layer of `nr_regions` in the desired class

**Source**
`bcdata::bcdc_get_data(record = 'dfe49c0-69c5-4c20-a6de-2c9bc99301f', resource = 'ec636f64-9c5f-4704-8e66-2dd43032c9b5')`

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

---

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

**Description**
Overlay a SpatialPolygonsDataFrame 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
)

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. If TRUE, it is up to the user to call future::plan() (or set options) to specify what parallel strategy to use.

Value

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

------------------------------------------------------------------------

regional_districts British Columbia Regional Districts

------------------------------------------------------------------------

Description

British Columbia Regional Districts

Usage

regional_districts(class = "sf", ask = interactive(), force = FALSE)

Arguments

class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
force Should you force download the data?

Value

The spatial layer of regional_districts in the desired class
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)
}

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)
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, 0.5), 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)

Arguments
raster_list  list of rasters
parallel  process in parallel? Default FALSE. If TRUE, it is up to the user to call future::plan()
(or set options) to specify what parallel strategy to use.

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.

---

**vrtr_files**

*List the files that a vrt is built on*

---

**Description**

List the files that a vrt is built on

**Usage**

vrtr_files(vrt, omit_vrt = FALSE)
Arguments

vrt path to a .vrt file
omit_vrt omit the listing of the original vrt. Default FALSE

Value

codect vector of tiles

vrt_info Get metadata about a .vrt file

Description

Get metadata about a .vrt file

Usage

vrt_info(vrt, options = character(0), quiet = FALSE)

Arguments

vrt path to a .vrt file
options options to pass to gdalinfo. See here for possible options.
quiet suppress output to the console (default FALSE)

Value

codect vector of vrt metadata

watercourses_15M British Columbia watercourses at 1:15M scale

Description

British Columbia watercourses at 1:15M scale

Usage

watercourses_15M(class = "sf", ask = interactive(), force = FALSE)

Arguments

class what class you want the object in? "sf" (default) or "sp".
ask Should the function ask the user before downloading the data to a cache? Defaults to the value of interactive().
force Should you force download the data?
**Value**

The spatial layer of `watercourses_15M` in the desired class

**Source**

https://ftp.maps.canada.ca/pub/nrcan_rncan/vector/canvec/fgdb/Hydro/canvec_15M_CA_Hydro_fgdb.zip

**Examples**

```r
## 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**

British Columbia watercourses at 1:5M scale

**Usage**

`watercourses_5M(class = "sf", ask = interactive(), force = FALSE)`

**Arguments**

- `class` what class you want the object in? "sf" (default) or "sp".
- `ask` Should the function ask the user before downloading the data to a cache? Defaults to the value of `interactive()`.
- `force` Should you force download the data?

**Value**

The spatial layer of `watercourses_5M` in the desired class

**Source**

https://ftp.maps.canada.ca/pub/nrcan_rncan/vector/canvec/fgdb/Hydro/canvec_5M_CA_Hydro_fgdb.zip

**Examples**

```r
## 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**

British Columbia’s Water Management Districts

**Usage**

```
water_districts(class = "sf", ask = interactive(), force = FALSE)
```

**Arguments**

- `class`: what class you want the object in? "sf" (default) or "sp".
- `ask`: Should the function ask the user before downloading the data to a cache? Defaults to the value of `interactive()`.
- `force`: Should you force download the data?

**Value**

The spatial layer of `water_districts` in the desired class

**Source**

`bcdata::bcdc_get_data(record = '92cb3ad8-9582-48a9-9e79-9a9d33601e50', resource = '07f9aa3f-0b66-4a49-919f-332d12bcd8f0')`

**Examples**

```r
## 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**

British Columbia’s Water Management Precincts

**Usage**

```
water_precincts(class = "sf", ask = interactive(), force = FALSE)
```

**Source**

`bcdata::bcdc_get_data(record = '92cb3ad8-9582-48a9-9e79-9a9d33601e50', resource = '07f9aa3f-0b66-4a49-919f-332d12bcd8f0')`
Arguments

   class  what class you want the object in? "sf" (default) or "sp".
   ask    Should the function ask the user before downloading the data to a cache? De-
          faults to the value of interactive().
   force  Should you force download the data?

Value

The spatial layer of water_precincts in the desired class

Source

  bcdata::bcdc_get_data(record = 'b5f436b4-532c-4ee2-ba27-90d55ec8c73f',resource = 'e482fd4a-be58-4541-8e0d-c39a764fd0a3-

Examples

## Not run:
my_layer <- water_precincts()
my_layer_sp <- water_precincts(class = 'sp')
## End(Not run)
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, 6
bc_bound, 7
bc_bound_hres, 7
bc_cities, 8
bc_neighbours, 9
bcmaps, 5
bec, 10
bec_colors (bec_colours), 10
bec_colours, 10
cded, 11
cded(), 19
cded_raster, 12
cded_stars, 13
census_dissemination_area, 14
census_division, 15
census_economic, 15
census_metropolitan_area, 16
census_subdivision, 17
census_tract, 18
combine_nr_rd, 18
combine_nr_rd(), 33, 37
delete_cache, 19
ecoprovinces, 20
ecoregions, 21
ecosections, 21
fix_geo_problems, 22
fsa, 23
future::plan(), 36, 38
get_big_data, 10, 23, 24, 39
get_layer, 24
get_poly_attribute, 25
gw_aquifers, 26
health_chsa, 26
health_ha, 27
health_hsda, 28
health_lha, 29
hydrozones, 29
make_shortcuts, 30
mapsheets_250K, 31
mapsheets_50K, 32
municipalities, 32
nr_areas, 33
nr_districts, 34
nr_regions, 35
options, 36, 38
raster::raster, 12
raster_by_poly, 35
regional_districts, 36
self_union, 37
sf::st_intersects, 11–13
show_cached_files (delete_cache), 19
stars::read_stars, 13
summarize_raster_list, 38
transform_bc_albers, 38
tsa, 39
vrt_files, 39
vrt_info, 40
water_districts, 42
water_precincts, 42
watercourses_15M, 40
watercourses_5M, 41
wsc_drainages, 43