

# 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 'bcddata' 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** bcddata (>= 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_hres . . . . .	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
census_tract . . . . .	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* 3

<i>self_union</i> . . . . .	37
<i>summarize_raster_list</i> . . . . .	38
<i>transform_bc_albers</i> . . . . .	38
<i>tsa</i> . . . . .	39
<i>VRT_files</i> . . . . .	39
<i>VRT_info</i> . . . . .	40
<i>watercourses_15M</i> . . . . .	40
<i>watercourses_5M</i> . . . . .	41
<i>water_districts</i> . . . . .	42
<i>water_precincts</i> . . . . .	42
<i>wsc_drainages</i> . . . . .	43

**Index** 45

---

*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**

```
bcddata::bcd_get_data(record = 'e8eeefc4-2826-47bc-8430-85703d328516', resource = 'c495d082-b586-4df0-
```

**Examples**

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

## End(Not run)
```

---

available_layers	<i>List available data layers</i>
------------------	-----------------------------------

---

**Description**

A data.frame of all available layers in the bcmaps package. This drawn directly from the B.C. Data Catalogue 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 bcmaps package but will be downloaded from the internet and cached on your hard drive.

**Examples**

```
## Not run:
available_layers()

## End(Not run)
```

---

bcmaps	<i>bcmaps: A data package providing various map layers for British Columbia</i>
--------	---

---

**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	<i>The size of British Columbia</i>
---------	-------------------------------------

---

**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	<i>BC Boundary</i>
----------	--------------------

---

**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? Defaults 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::bcdata_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	<i>BC Boundary - High Resolution</i>
---------------	--------------------------------------

---

**Description**

BC Boundary - High Resolution

**Usage**

```
bc_bound_hres(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\_bound\_hres in the desired class

**Source**

```
bcdc_get_data(record = '30aeb5c1-4285-46c8-b60b-15b1a6f4258b', resource = '3d72cf36-ab53-4a2a-9988-a88
= 'BC_Boundary_Terrestrial_Multipart')
```

**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*


---

**Description**

BC Major Cities Points

**Usage**

```
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::bcdata_get_data(record = 'b678c432-c5c1-4341-88db-0d6befa0c7f8', resource = '443dd858-2e37-4a8f-
```

**Examples**

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

---

bc_neighbours	<i>Boundary of British Columbia, provinces/states and the portion of the Pacific Ocean that borders British Columbia</i>
---------------	--

---

**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::bcdata_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	<i>British Columbia BEC Map</i>
-----	---------------------------------

---

### 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 <a href="#">get_big_data</a>

### 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	<i>Biogeoclimatic Zone Colours</i>
-------------	------------------------------------

---

### 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(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

```
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 mapsheet tiles
.predicate	geometry predicate function used to find the mapsheets from your aoi. Default <a href="#">sf::st_intersects</a> .
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 <code>interactive()</code> .

`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

```
## Not run:
vic <- census_subdivision()[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

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

### Arguments

<code>aoi</code>	Area of Interest. Currently supports sf and sp polygons, stars and raster objects.
<code>tiles_50K</code>	a character vector of 1:50,000 NTS mapsheet tiles
<code>.predicate</code>	geometry predicate function used to find the mapsheets from your aoi. Default <a href="#">sf::st_intersects</a> .
<code>dest_vrt</code>	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
<code>check_tiles</code>	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.
<code>...</code>	Further arguments passed on to <a href="#">raster::raster</a>

**Value**

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

**Examples**

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

## End(Not run)
```

---

cded\_stars

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


---

**Description**

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

**Usage**

```
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 <a href="#">sf::st_intersects</a> .
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 <a href="#">stars::read_stars</a>

**Value**

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

**Examples**

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

## End(Not run)
```

---

census\_dissemination\_area

*Current Census Dissemination Areas*

---

**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? 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::bcdata_get_data(record = 'a091fd65-d682-4a24-8c0e-68de7c87e3a3', resource = 'a7fa66d4-0f95-4c58-
```

**Examples**

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

## End(Not run)
```

---

census_division	<i>Current Census Division Boundaries</i>
-----------------	---

---

**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::bcdata_get_data(record = 'ef17918a-597a-4012-8534-f8e71d8735b3', resource = '36b530c2-1de6-44a2-
```

**Examples**

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

---

census_economic	<i>Current Census Economic Region Boundaries</i>
-----------------	--

---

**Description**

Current Census Economic Region Boundaries

**Usage**

```
census_economic(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\_economic in the desired class

**Source**

```
bcdata::bcdata_get_data(record = '1aebc451-a41c-496f-8b18-6f414cde93b7', resource = '3f0236cf-b1a1-4f1a-
```

**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



**Source**

```
bcdata::bcdata_get_data(record = 'a6fb34b7-0937-4718-8f1f-43dba2c0f407', resource = 'f129a965-363e-4d7e-
```

**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? 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::bcdata_get_data(record = '4c5618c6-38dd-4a62-a3de-9408b4974bb6', resource = '98bd1222-57bb-4504-
```

**Examples**

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

---

census_tract	<i>Current Census Tract Boundaries</i>
--------------	--

---

**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::bcdata_get_data(record = '539aae5b-12f6-4934-9592-9b27acc827f8', resource = 'be767db6-0d4e-4906-
```

**Examples**

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

## End(Not run)
```

---

combine_nr_rd	<i>Combine Northern Rockies Regional Municipality with Regional Districts</i>
---------------	---

---

**Description**

Combine Northern Rockies Regional Municipality with Regional Districts

**Usage**

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

**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	<i>View and delete cached files</i>
--------------	-------------------------------------

---

**Description**

View and delete cached files

Show the files you have in your cache

**Usage**

```
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

**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? Defaults to the value of interactive().
force	Should you force download the data?

**Value**

The spatial layer of ecoprovinces in the desired class

**Source**

```
bcdata::bcdata_get_data(record = '51832f47-efdf-4956-837a-45fc2c9032dd', resource = '811fcedb-1a53-4574-
```

**Examples**

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

## End(Not run)
```

---

ecoregions                      *British Columbia 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::bcdata_get_data(record = 'd00389e0-66da-4895-bd56-39a0dd64aa78', resource = 'bd816a86-4f5e-4989-
```

**Examples**

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

## End(Not run)
```

---

ecosections                      *British Columbia 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 ecosections in the desired class

**Source**

```
bcdata::bcdata_get_data(record = 'ccc01f43-860d-4583-8ba4-e72d8379441e', resource = '6b6a3122-7a0b-4c0f-
```

**Examples**

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

## End(Not run)
```

---

fix_geo_problems	<i>Check and fix polygons that self-intersect, and sometimes can fix orphan holes</i>
------------------	---

---

**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

---

fsa *British Columbia Forward Sortation Areas*

---

**Description**

British Columbia Forward Sortation Areas

**Usage**

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

**Examples**

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

---

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 bcmappedata 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 bcmapped. Default TRUE

---

get_layer	<i>Get a B.C. spatial layer</i>
-----------	---------------------------------

---

**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 <code>available_layers()</code>
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 <code>interactive()</code> .
force	Should you force download the data?
...	arguments passed on to <a href="#">get_big_data</a> if the layer needs to be downloaded from a bcmappedata 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	<i>Get or calculate the attribute of a list-column containing nested dataframes.</i>
--------------------	--

---

### 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

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

### 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**

British Columbia's developed ground water aquifers

**Usage**

```
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::bcdata_get_data(record = '099d69c5-1401-484d-9e19-c121ccb7977c', resource = '8f421e3a-ccd3-4fab-
```

**Examples**

```
## 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**

```
health_chsa(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\_chsa in the desired class

**Source**

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

**Examples**

```
## 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::bcdata_get_data(record = '7bc6018f-bb4f-4e5d-845e-c529e3d1ac3b', resource = '93b79a3c-2da4-4fd4-
```

**Examples**

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

## End(Not run)
```

---

health\_hsda

*Health Service Delivery Area Boundaries*

---

**Description**

Health Service Delivery Area Boundaries

**Usage**

```
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::bcdata_get_data(record = '71c930b9-563a-46da-a10f-ead49ccbc390', resource = 'c5dad467-229b-4378-
```

**Examples**

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

## End(Not run)
```

---

health\_lha                      *Local Health Area Boundaries*

---

**Description**

Local Health Area Boundaries

**Usage**

```
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::bcdata_get_data(record = 'afd021d9-7722-4410-b506-d394c66e74fc', resource = 'd6e951d3-5103-475a-
```

**Examples**

```
## 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**

```
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::bcdata_get_data(record = '329fd234-8835-4d44-9aaa-97c37bfc8d92', resource = 'baeb665e-85c7-4a7b-
```

**Examples**

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

## End(Not run)
```

---

make_shortcuts	<i>Make shortcut functions for data objects in bcmaps from B.C. Data Catalogue</i>
----------------	--

---

**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)

**Examples**

```
## Not run:  
make_shortcut()  
  
## End(Not run)
```

---

mapsheets\_250K

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

---

**Description**

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

**Usage**

```
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**

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

---

mapsheets\_50K

*NTS 50K Grid - Digital Baseline Mapping at 1:50,000 (NTS)*

---

**Description**

NTS 50K Grid - Digital Baseline Mapping at 1:50,000 (NTS)

**Usage**

```
mapsheets_50K(class = "sf")
```

**Arguments**

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

**Value**

The spatial layer of mapsheets\_50K in the desired class

**Source**

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

**Examples**

```
## Not run:  
my_layer <- mapsheets_50K()  
my_layer_sp <- mapsheets_50K(class = 'sp')  
  
## End(Not run)
```

---

municipalities

*British Columbia 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::bcdata_get_data(record = 'e3c3c580-996a-4668-8bc5-6aa7c7dc4932', resource = '25c95b07-5882-47ff-
```

**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**

British Columbia Natural Resource (NR) Areas

**Usage**

```
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::bcdata_get_data(record = 'c1861ba4-abb8-4947-b3e5-7f7c4d7257d5', resource = '4b317896-1a42-4c03-
```

**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**

British Columbia Natural Resource (NR) Districts

**Usage**

```
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::bcdata_get_data(record = '0bc73892-e41f-41d0-8d8e-828c16139337', resource = 'e6676e55-2a6f-4b2b-
```

**Examples**

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

## End(Not run)
```

---

nr_regions	<i>British Columbia Natural Resource (NR) Regions</i>
------------	---

---

**Description**

British Columbia Natural Resource (NR) Regions

**Usage**

```
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::bcdata_get_data(record = 'dfc492c0-69c5-4c20-a6de-2c9bc999301f', resource = 'ec636f64-9c5f-4704-
```

**Examples**

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

## End(Not run)
```

---

raster_by_poly	<i>Overlay a SpatialPolygonsDataFrame or sf polygons layer on a raster layer and clip the raster to each polygon. Optionally done in parallel</i>
----------------	---

---

**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 <code>future::plan()</code> (or set <code>options</code> ) to specify what parallel strategy to use.

**Value**

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

---

regional_districts	<i>British Columbia Regional Districts</i>
--------------------	--

---

**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

**Source**

```
bcdata::bcdata_get_data(record = 'd1aff64e-dbfe-45a6-af97-582b7f6418b9', resource = '57c7f719-dc87-415c-
```

**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	<i>Union a SpatialPolygons* object with itself to remove overlaps, while retaining attributes</i>
------------	---

---

**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)
```

### Arguments

<code>raster_list</code>	list of rasters
<code>parallel</code>	process in parallel? Default FALSE. If TRUE, it is up to the user to call <code>future::plan()</code> (or set <a href="#">options</a> ) 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

<code>obj</code>	The Spatial* or sf object to transform
------------------	--

**Value**

the Spatial\* or sf object in BC Albers projection

---

tsa	<i>British Columbia Timber Supply Areas and TSA Blocks</i>
-----	--

---

**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 <a href="#">get_big_data</a>

**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](#).

---

vrt_files	<i>List the files that a vrt is built on</i>
-----------	--

---

**Description**

List the files that a vrt is built on

**Usage**

```
vrt_files(vrt, omit_vrt = FALSE)
```

**Arguments**

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

**Value**

character 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**

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

**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**

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

**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

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::bcdata_get_data(record = '92cb3ad8-9582-48a9-9e79-9a9d33601e50', resource = '07f9aa3f-0b66-4a49-
```

### 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

British Columbia's Water Management Precincts

### Usage

```
water_precincts(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\_precincts in the desired class

**Source**

```
bcdata::bcdata_get_data(record = 'b5f436b4-532c-4ee2-ba27-90d55ec8c73f', resource = 'e482fd4a-be58-4541-
```

**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**

Water Survey of Canada Sub-Sub-Drainage Areas

**Usage**

```
wsc_drainages(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 wsc\_drainages in the desired class

**Source**

```
bcdata::bcdata_get_data(record = '7ae18a3c-917b-4cb1-9aa8-51a172475dbb', resource = '4455072e-d33b-4685-
```

**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