Package ‘ceramic’

July 20, 2019

Title Download Online Imagery Tiles

Version 0.6.0

Description Download imagery tiles to a standard cache and load the data into raster objects.
Facilities for 'AWS' terrain <https://aws.amazon.com/public-datasets/terrain/> terrain and 'Mapbox' <https://www.mapbox.com/> servers are provided.

Depends R (>= 3.5.0)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Imports slippymath (>= 0.3.0), curl, dplyr, fs (>= 1.3.0), glue, rappdirs, tibble, stats, purrr, magrittr, rlang, sp, graphics, raster, spex, jpeg, png, utils, reproj

Suggests testthat, covr, rgdal, spelling

URL `https://github.com/hypertidy/ceramic`

BugReports `https://github.com/hypertidy/ceramic/issues`

Language en-US

NeedsCompilation no

Author Michael Sumner [aut, cre] (<https://orcid.org/0000-0002-2471-7511>), Miles McBain [ctb] (<https://orcid.org/0000-0003-2865-2548>), Ben Raymond [ctb] (regex wizardry)

Maintainer Michael Sumner `<mdsumner@gmail.com>`

Repository CRAN

Date/Publication 2019-07-20 09:00:02 UTC

R topics documented:

`ceramic-package` *Obtain imagery tiles*

Description

The ceramic package provides tools to download and load imagery and raster tiles from online servers.

Details

Any process that can trigger downloads will first check the `ceramic_cache()` in case the tile already exists.

The main functions are for downloading tiles and loading them as raster objects, and each accepts a spatial object for the first argument, alternatively a raster extent, or location:

`get_tiles`	Download tiles for a given service for an extent and resolution
`get_tiles_buffer`	Download tiles based on location and buffer (width, height) in metres
`get_tiles_dim`	Download tiles based on extent and output dimension in pixels
`get_tiles_zoom`	Download tiles base on extent and zoom level

Two helper functions will trigger the download of tiles and also collate the result into a raster object:

`cc_location`	Download tiles and build a raster object of imagery
`cc_elevation`	Download tiles and build a raster object of elevation data

Administration functions for handling the file cache and required API key for on online service:

`get_api_key`	Return the stored key for online API, or NULL
`ceramic_cache`	Report the location of the tile cache
`clear_ceramic_cache`	Delete all files in the tile cache (use with caution!)

Other functions that are either rarely used or considered subject to change:

`ceramic_tiles`	Find particular tiles from the cache
`mercator_tile_extent`	Abstract raster-extent form of the spherical Mercator tile system, expressed in tile-index and z
`plot_tiles`	Plot the tiles from `ceramic_tiles`
`tiles_to_polygon`	Convert `ceramic_tiles` to simple features format
`cc_casey`	Specific location hardcoded form of `cc_location`
`cc_davis`	Specific location hardcoded form of `cc_location`
`cc_heard`	Specific location hardcoded form of `cc_location`
`cc_kingston`	Specific location hardcoded form of `cc_location`
`cc_macquarie`	Specific location hardcoded form of `cc_location`
`cc_mawson`	Specific location hardcoded form of `cc_location`

`cc_location` *Obtain tiled imagery by location query*

Description

Obtain imagery or elevation data by location query. The first argument `loc` may be a spatial object (sp, raster, sf) or a 2-column matrix with a single longitude and latitude value. Use `buffer` to define a width and height to pad around the raw longitude and latitude in metres. If `loc` has an extent, then `buffer` is ignored.

Usage

```
cc_location(loc = NULL, buffer = 5000, type = "mapbox.satellite",
  ..., zoom = NULL, max_tiles = NULL, debug = FALSE)

cc_macquarie(loc = c(158.93835, -54.49871), buffer = 5000,
  type = "mapbox.outdoors", ..., zoom = NULL, max_tiles = NULL,
  debug = FALSE)

cc_davis(loc = c(77 + 58/60 + 3/3600, -(68 + 34/60 + 36/3600)),
  buffer = 5000, type = "mapbox.outdoors", ..., zoom = NULL,
  max_tiles = NULL, debug = FALSE)

cc_mawson(loc = c(62 + 52/60 + 27/3600, -(67 + 36/60 + 12/3600)),
  buffer = 5000, type = "mapbox.outdoors", ..., zoom = NULL,
  max_tiles = NULL, debug = FALSE)

cc_casey(loc = cbind(110 + 31/60 + 36/3600, -(66 + 16/60 + 57/3600)),
  buffer = 5000, type = "mapbox.outdoors", ..., zoom = NULL,
  max_tiles = NULL, debug = FALSE)

cc_heard(loc = c(73 + 30/60 + 30/3600, -(53 + 0 + 0/3600)),
  buffer = 5000, type = "mapbox.outdoors", ..., zoom = NULL,
  max_tiles = NULL, debug = FALSE)
```

```
cc_kingston(loc = c(147.70837, -42.98682), buffer = 5000,
  type = "mapbox.outdoors", ..., zoom = NULL, max_tiles = NULL,
  debug = FALSE)

cc_elevation(loc = NULL, buffer = 5000, ..., zoom = NULL,
  max_tiles = NULL, debug = FALSE)
```

Arguments

`loc`	a longitude, latitude pair of coordinates, or a spatial object
`buffer`	with in metres to extend around the location, ignored if 'loc' is a spatial object with extent
`type`	character string of provider imagery type (see Details)
`...`	arguments passed to internal function, specifically `base_url` (see Details)
`zoom`	desired zoom for tiles, use with caution - if `NULL` is chosen automatically
`max_tiles`	maximum number of tiles to be read into memory - if `NULL` is set by zoom constraints
`debug`	optionally print out files that will be used

Details

`cc_elevation` does extra work to unpack the DEM tiles from the RGB format.

Available types are 'elevation-tiles-prod' for AWS elevation tiles, and 'mapbox.satellite', 'mapbox.outdoors', 'mapbox.terrain-rgb' or any string accepted by Mapbox services.

Note that arguments `max_tiles` and `zoom` are mutually exclusive. One or both must be `NULL`. If both are NULL then `max_tiles = 16L`.

Value

A `raster::brick()` object, either 'RasterBrick' with three layers (Red, Green, Blue) or with a single layer in the case of `cc_elevation()`.

Custom styles

Custom Mapbox styles may be specified with the argument `base_url` in the form: `"https://api.mapbox.com/sty`

Currently must be considered in-development.

Examples

```
if (!is.null(get_api_key())) {

 img <- cc_location(cbind(147, -42), buffer = 1e5)

 ## this source does not need the Mapbox API, but we won't run the example unless it's set
 dem <- cc_kingston(buffer = 1e4, type = "elevation-tiles-prod")
 raster::plot(dem, col = grey(seq(0, 1, length = 94)))
 ## Mapbox imagery
 im <- cc_macquarie()
 library(raster)
 plotRGB(im)
 }
```

`ceramic_tiles` *Tile files*

Description

Find existing files in the cache. Various options can be controlled, this is liable to change pending generalization across providers.

Usage

```
ceramic_tiles(zoom = NULL, type = "mapbox.satellite",
  source = "api.mapbox.com", glob = NULL, regexp = NULL)
```

Arguments

`zoom`	zoom level
`type`	imagery type
`source`	imagery source
`glob`	see `fs::dir_ls`
`regexp`	see `fs::dir_ls`

Value

A data frame of tile file paths with tile index, zoom, type, version, source and spatial extent.

Examples

```
if (interactive() && !is.null(get_api_key())) {
 tiles <- ceramic_tiles(zoom = 0)
}
```

`clear_ceramic_cache` *Clear ceramic cache*

Description

Delete all downloaded files in the `ceramic_cache()`.

Usage

```
clear_ceramic_cache(clobber = FALSE, ...)
```

Arguments

`clobber`	set to `TRUE` to avoid checks and delete files
`...`	reserved for future arguments, currently ignored

Value

This function is called for its side effect, but also returns the file paths as a character vector whether deleted or not, or NULL if the user cancels.

`get-tiles-constrained` *Get tiles with specific constraints*

Description

Get tiles by zoom, by overall dimension, or by buffer on a single point.

Usage

```
get_tiles_zoom(x, zoom = 0, ..., format = "png")

get_tiles_dim(x, dim = c(512, 512), ..., format = "png")

get_tiles_buffer(x, buffer = NULL, ..., max_tiles = 9,
  format = "png")
```

Arguments

`x`	a spatial object with an extent
`zoom`	desired zoom for tiles, use with caution - cannot be unset in `get_tiles_zoom`
`...`	passed to `get_tiles()`
`format`	defaults to "png", also available is "jpg"
`dim`	for `get_tiles_dim` the overall maximum dimensions of the image (padded out to tile size of 256x256)
`buffer`	width in metres to extend around the location, ignored if 'x' is a spatial object with extent
`max_tiles`	maximum number of tiles - if `NULL` is set by zoom constraints

Details

Each function expects an extent in longitude latitude or a spatial object with extent as the first argument.

`get_tiles_zoom()` requires a zoom value, defaulting to 0

`get_tiles_dim()` requires a dim value, default to `c(512,512)`, a set of 4 tiles

`get_tiles_buffer()` requires a single location (longitude, latitude) and a buffer in metres

Value

A list with files downloaded in character vector, a data frame of the tile indices, the zoom level used and the extent in raster::extent form.

See Also

get_tiles

Examples

```
if (!is.null(get_api_key())) {
 ex <- raster::extent(146, 147, -43, -42)
 tile_infoz <- get_tiles_zoom(ex, type = "mapbox.outdoors", zoom = 1)

 tile_infod <- get_tiles_dim(ex, type = "mapbox.outdoors", dim = c(256, 256))

 tile_infob <- get_tiles_buffer(cbind(146.5, -42.5), buffer = 5000, type = "mapbox.outdoors"
}
```

`get_api_key` *Get API key for Mapbox service*

Description

Mapbox tile providers require an API key. Other providers may not need a key and so this is ignored.

Usage

```
get_api_key(api = "mapbox", ...)
```

Arguments

`api`	character string denoting which service ("mapbox" only)
`...`	currently ignored

Details

The mapdeck package[1] has a more comprehensive tool for setting the Mapbox API key, if this is in use ceramic will find it first and use it.

To set your Mapbox API key obtain a key from `https://account.mapbox.com/access-tokens/`

```
1) Run this to set for the session 'Sys.setenv(MAPBOX_API_KEY=<yourkey>)'

OR,

2) To set permanently store 'MAPBOX_API_KEY=<yourkey>' in '~/.Renviron'.
```

There is a fairly liberal allowance for the actual name of the environment variable, any of 'MAPBOX_API_KEY', 'MAPBOX_API_TOKEN', 'MAPBOX_KEY', 'MAPBOX_TOKEN', or 'MAPBOX' will work (and they are sought in that order).

If no key is available, `NULL` is returned, with a warning.

Value

The stored API key value, see Details.

Examples

```
get_api_key()
```

[1] `https://CRAN.r-project.org/package=mapdeck/`

`get_tiles` *Download Mapbox imagery tiles*

Description

Obtain imagery or elevation tiles by location query. The first argument `loc` may be a spatial object (sp, raster, sf) or a 2-column matrix with a single longitude and latitude value. Use `buffer` to define a width and height to pad around the raw longitude and latitude in metres. If `loc` has an extent, then `buffer` is ignored.

Usage

```
get_tiles(x, buffer, type = "mapbox.satellite", crop_to_buffer = TRUE,
  format = NULL, ..., zoom = NULL, debug = FALSE, max_tiles = NULL,
  base_url = NULL, verbose = TRUE)
```

Arguments

`x`	a longitude, latitude pair of coordinates, or a spatial object
`buffer`	width in metres to extend around the location, ignored if 'x' is a spatial object with extent
`type`	character string of provider imagery type (see Details)
`crop_to_buffer`	crop to the user extent, used for creation of output objects (otherwise is padded tile extent)
`format`	tile format to use, defaults to "jpg" for Mapbox satellite imagery and "png" otherwise
`...`	arguments passed to internal function, specifically `base_url` (see Details)
`zoom`	desired zoom for tiles, use with caution - if `NULL` is chosen automatically
`debug`	optionally print out files that will be used
`max_tiles`	maximum number of tiles - if `NULL` is set by zoom constraints
`base_url`	tile provider URL expert use only
`verbose`	report messages or suppress them

Details

`get_tiles()` may be run with no arguments, and will download (and report on) the default tile source at zoom 0. Arguments `type`, `zoom` (or `max_tiles`), `format` may be used without setting `loc` or `buffer` and the entire world extent will be used. Please use with caution! There is no maximum on what will be downloaded, but it can be interrupted at any time.

Use `debug` = `TRUE` to avoid download and simply report on what would be done.

`cc_elevation` does extra work to unpack the DEM tiles from the RGB format.

Available types are 'elevation-tiles-prod' for AWS elevation tiles, and 'mapbox.satellite', 'mapbox.outdoors', 'mapbox.terrain-rgb', 'mapbox.streets', 'mapbox.light', 'mapbox.dark' or any other string accepted by Mapbox services.

Value

A list with files downloaded in character vector, a data frame of the tile indices, the zoom level used and the extent in raster::extent form.

See Also

get_tiles_zoom get_tiles_dim get_tiles_buffer

Examples

```
if (!is.null(get_api_key())) {
   tile_info <- get_tiles(raster::extent(146, 147, -43, -42), type = "mapbox.outdoors", zoom
}
```

mercator_tile_extent

Tile extent

Description

Calculate tile extent for a given x, y tile at a zoom level.

Usage

```
mercator_tile_extent(tile_x, tile_y, zoom, tile_size = 256)
```

Arguments

`tile_x`	x coordinate of tile
`tile_y`	y coordinate of tile
`zoom`	zoo level
`tile_size`	tile dimensions (assumed square, i.e. 256x256)

Details

Currently only spherical Mercator is supported.

Value

A numeric vector of the spatial extent, in 'xmin', 'xmax', 'ymin', 'ymax' form.

Examples

```
mercator_tile_extent(2, 4, zoom = 10)

global <- mercator_tile_extent(0, 0, zoom = 0)
plot(NA, xlim = global[c("xmin", "xmax")], ylim = global[c("ymin", "ymax")])
rect_plot <- function(x) rect(x["xmin"], x["ymin"], x["xmax"], x["ymax"])
rect_plot(mercator_tile_extent(1, 1, zoom = 2))
rect_plot(mercator_tile_extent(2, 1, zoom = 2))
rect_plot(mercator_tile_extent(1, 2, zoom = 2))

rect_plot(mercator_tile_extent(1, 1, zoom = 4))
rect_plot(mercator_tile_extent(2, 1, zoom = 4))
rect_plot(mercator_tile_extent(1, 2, zoom = 4))
```

`plot_tiles` *Plot slippy map tiles*

Description

Create a new plot of tile rectangles, or add to an existing plot.

Usage

```
plot_tiles(x, ..., add = FALSE, label = TRUE, cex = 0.6,
  add_coast = TRUE, include_zoom = TRUE)

tiles_to_polygon(x)
```

Arguments

`x`	tiles as create by `ceramic_tiles()`
`...`	arguments passed to `graphics::rect()`
`add`	add to an existing plot?
`label`	include text label?
`cex`	relative size of text label if drawn (see `text()`)
`add_coast`	include a basic coastline on the plot?
`include_zoom`	include zoom level with text label if drawn?

Details

The extent ('xmin', 'xmax', 'ymin', 'ymax') is used directly to draw the tiles so must be in the native Mercator coordinate system used by most tile servers.

Value

`plot_tiles()` is called for its side-effect, a plot, and returns `NULL` invisibly. tiles_to_polygon returns a simple features polygon data frame.

Examples

```
if (!is.null(get_api_key())) {
  get_tiles_zoom(zoom = 1)
  tiles <- ceramic_tiles(zoom = 1)
  plot_tiles(tiles)
}
```