Package: feltr (via r-universe)

September 10, 2024

Title Access the Felt API

Version 0.1.0

Description Upload, download, and edit internet maps with the Felt API (<https://feltmaps.notion.site/Felt-Public-API-reference-c01e0e6b0d954a678c608131b894e8e1>). Allows users to create new maps, edit existing maps, and extract data. Provides tools for working with layers, which represent geographic data, and elements, which are interactive annotations. Spatial data accessed from the API is transformed to work with 'sf'.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Imports cli, curl, dplyr, fs, geojsonsf, httr2, jsonlite, purrr, rlang, sf, stringr, tibble, tidyr

URL https://github.com/christopherkenny/feltr, https://christophertkenny.com/feltr/

Suggests httptest2, testthat (>= 3.0.0)

Config/testthat/edition 3

Repository https://christopherkenny.r-universe.dev

RemoteUrl https://github.com/christopherkenny/feltr

RemoteRef HEAD

RemoteSha f216514fd99bb64d3934231056f5a8278687e1ff

Contents

`felt_add_library_layer` *Add layer to library*

Description

Add layer to library

Usage

```
felt_add_library_layer(map_id, layer_id, name = NULL, clean = TRUE)
```

Arguments

`map_id`	character, map identifier from url, from `https://felt.com/map/Readable-Name-map_id`
`layer_id`	character, layer identifier from url, from `felt_get_map()`
`name`	character, name to save the layer under. Optional.
`clean`	logical, whether to turn the API response into a tibble::tibble

Value

a tibble::tibble if `clean = TRUE`, otherwise a list

Examples

```
felt_add_library_layer(map_id = 'TBI8sDkmQjuK2GX9CSiHiUA',
                       layer_id = '4Lc7RaEyRP2LfARGmR6e4C',
                       name = paste0('Test layer ', Sys.time()))
```

`felt_add_library_layer_group`
Add layer group to library

Description

Add layer group to library

Usage

```
felt_add_library_layer_group(map_id, layer_group_id, name = NULL, clean = TRUE)
```

Arguments

`map_id`	character, map identifier from url, from `https://felt.com/map/Readable-Name-map_id`
`layer_group_id`	character, layer group identifier from url, from `felt_get_map()`.
`name`	character, name to save the layer group under. Optional.
`clean`	logical, whether to turn the API response into a tibble::tibble

Value

a tibble::tibble if `clean = TRUE`, otherwise a list

Examples

```
felt_add_library_layer_group(map_id = 'TBI8sDkmQjuK2GX9CSiHiUA',
                             layer_group_id = 'rHxyTef7S9CO8W7n1PvBVwC',
                             name = paste0('Test layer group ', Sys.time()))
```

felt_add_map_elements *Add Elements to Existing Map*

Description

Add Elements to Existing Map

Usage

```
felt_add_map_elements(map_id, elements, clean = TRUE)
```

Arguments

map_id	character, map identifier from url, from https://felt.com/map/Readable-Name-map_id
elements	a sf::sf object or a path to a geojson file
clean	logical, whether to turn the API response into a tibble::tibble

Value

a tibble::tibble with the elements added

Examples

```
elem <- felt_add_map_elements(map_id = 'Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA',
                    elements = fs::path_package('feltr',  'bbox.geojson'))
elem
# and delete layer
felt_delete_map_elements(map_id = 'TBI8sDkmQjuK2GX9CSiHiUA', element_id = elem$felt_id)
```

felt_add_map_layers *Add Layers to Existing Map*

Description

Add Layers to Existing Map

Usage

```
felt_add_map_layers(
  map_id,
  name = NULL,
  file_names = NULL,
  lat = NULL,
  lng = NULL,
  zoom = NULL,
  clean = TRUE
)
```

Arguments

map_id	character, map identifier from url, from https://felt.com/map/Readable-Name-map_id
name	Name of the layer. Required.
file_names	Files to include. Required.
lat	For images, the latitude of the center of the image. Optional.
lng	For images, the longitude of the center of the image. Optional.
zoom	For images, the zoom level of the image. Optional.
clean	logical, whether to turn the API response into a tibble::tibble

Value

status of the upload

Examples

```
layer <- felt_add_map_layers(map_id = 'Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA',
                    file_names = fs::path_package('feltr',  'towns.geojson'),
                    name = 'Towns  test')
layer
```

---

felt_add_map_layers_url

Add Layers to Existing Map from URL

Description

See Felt "Upload Anything" documentation for detailed examples of potential URLs.

Usage

```
felt_add_map_layers_url(map_id, url, name = NULL, clean = TRUE)
```

Arguments

map_id	character, map identifier from url, from https://felt.com/map/Readable-Name-map_id
url	Link to layer to include. Required
name	Name of the layer. Required.
clean	logical, whether to turn the API response into a tibble::tibble

Value

a tibble::tibble for the created layer

Examples

```
# split the URL for length reasons
url <- paste0(
'https://www.rocklandgis.com/portal/sharing/rest/',
'content/items/73fc78cb0fb04580b4788937fe5ee697/data'
)
layer <- felt_add_map_layers_url(
    map_id = 'Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA',
    url = url,
    name = 'URL Parks test')
layer
# and delete the new layer
felt_delete_map_layer(map_id = 'TBI8sDkmQjuK2GX9CSiHiUA',  layer_id = layer$layer_id)
```

`felt_create_map` *Create a new map*

Description

Create a new map

Usage

```
felt_create_map(
  title = NULL,
  basemap = NULL,
  layer_urls = NULL,
  lat = NULL,
  lon = NULL,
  zoom = NULL,
  description = NULL,
  public_access = NULL,
  clean = TRUE
)
```

Arguments

`title`	Title to use for the map. Defaults to `NULL`.
`basemap`	Basemap for the new map. Defaults to `NULL`. Can be a URL or color hex code. Text options include `'default'` (same as `NULL`), `'light'`, `'dark'`, or `'satellite'`.
`layer_urls`	vector of URLs to generate layers in map. Defaults to `NULL`.
`lat`	latitude to center the map. Defaults to `NULL`.
`lon`	longitude to center the map. Defaults to `NULL`
`zoom`	zoom level to initialize the map with. Defaults to `NULL`.

description	Description for the map legend. Defaults to NULL.
public_access	Degree of public acess. Defaults to NULL, which is view_only. Text options also include 'private', 'view_and_comment', and 'view_comment_and_edit'.
clean	logical, whether to turn the API response into a tibble::tibble

Value

a tibble::tibble for the new map

Examples

```
map <- felt_create_map(title = 'feltr example')
map
# and delete it again
felt_delete_map(map_id = map$id)
```

---

felt_delete_map *Delete an existing map*

Description

Delete an existing map

Usage

```
felt_delete_map(map_id)
```

Arguments

map_id	character, map identifier from url, from https://felt.com/map/Readable-Name-map_id

Value

response code

Examples

```
map <- felt_create_map(title = 'feltr example')
felt_delete_map(map_id = map$id)
```

felt_delete_map_elements *Delete an existing element*

Description

Delete an existing element

Usage

```
felt_delete_map_elements(map_id, element_id)
```

Arguments

map_id	character, map identifier from url, from https://felt.com/map/Readable-Name-map_id
element_id	element identifier, as returned by felt_get_map_elements() or felt_add_map_elements()

Value

response code

Examples

```
elem <- felt_add_map_elements(map_id = 'Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA',
                    elements = fs::path_package('feltr',  'bbox.geojson'))
elem
# and delete layer
felt_delete_map_elements(map_id = 'TBI8sDkmQjuK2GX9CSiHiUA', element_id = elem$felt_id)
```

felt_delete_map_layer *Delete Layer from an Existing Map*

Description

Delete Layer from an Existing Map

Usage

```
felt_delete_map_layer(map_id, layer_id, clean = TRUE)
```

Arguments

map_id	character, map identifier from url, from https://felt.com/map/Readable-Name-map_id
layer_id	character, layer identifier from url, from felt_get_map()
clean	logical, whether to turn the API response into a tibble::tibble

Value

response code

Examples

```
# split the URL for length reasons
url <- paste0(
'https://www.rocklandgis.com/portal/sharing/rest/',
'content/items/73fc78cb0fb04580b4788937fe5ee697/data'
)
layer <- felt_add_map_layers_url(
    map_id = 'Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA',
    url = url,
    name = 'URL Parks test')
layer
# and delete the new layer
felt_delete_map_layer(map_id = 'TBI8sDkmQjuK2GX9CSiHiUA',  layer_id = layer$layer_id)
```

---

felt_get_comments *Export Comments on an Existing Map*

Description

Export Comments on an Existing Map

Usage

```
felt_get_comments(map_id, clean = TRUE)
```

Arguments

map_id	character, map identifier from url, from https://felt.com/map/Readable-Name-map_id
clean	logical, whether to turn the API response into a tibble::tibble

Value

a tibble::tibble for the map

Examples

```
felt_get_comments(map_id = 'Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA')
```

felt_get_library *List all layers in your workspace library*

Description

List all layers in your workspace library

Usage

```
felt_get_library(source = "workspace", clean = TRUE)
```

Arguments

source	character, source of the layers, one of `workspace`, `felt`, or `all`
clean	logical, whether to turn the API response into a tibble::tibble

Value

a tibble::tibble if `clean = TRUE`, otherwise a list

Examples

```
felt_get_library()
```

felt_get_map *Get Map Information from Map ID*

Description

- `felt_get_map()` returns identifying information for the map
- `felt_get_map_layers()` returns information about each layer in the map
- `felt_get_map_elements()` returns the shapes for each layer in the map

Usage

```
felt_get_map(map_id, clean = TRUE)

felt_get_map_layers(map_id, clean = TRUE)

felt_get_map_layer(map_id, layer_id, clean = TRUE)
```

Arguments

map_id	character, map identifier from url, from https://felt.com/map/Readable-Name-map_id
clean	logical, whether to turn the API response into a tibble::tibble
layer_id	character, layer identifier from url, from felt_get_map()

Value

a tibble::tibble for the map, if clean = TRUE, otherwise a list

Examples

```
felt_get_map('Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA')
felt_get_map_layers('Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA')

# slower, as it has to build the shapes from the API result
felt_get_map_elements('Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA')
```

---

felt_get_map_elements *Get Map Elements from Map ID*

Description

Get Map Elements from Map ID

Usage

```
felt_get_map_elements(map_id, clean = TRUE)

felt_get_map_element_groups(map_id, clean = TRUE)

felt_get_map_elements_in_group(map_id, group_id, clean = TRUE)
```

Arguments

map_id	character, map identifier from url, from https://felt.com/map/Readable-Name-map_id
clean	logical, whether to turn the API response into a tibble::tibble
group_id	group identifier, as returned by felt_get_map_element_groups()

Value

a tibble::tibble for the map

Examples

```
felt_get_map_elements(map_id = 'Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA')
felt_get_map_element_groups('TBI8sDkmQjuK2GX9CSiHiUA')
felt_get_map_elements_in_group('TBI8sDkmQjuK2GX9CSiHiUA', '3Wl5s2AqRmiYgO9CrBFxO3D')
```

`felt_get_map_layer_group` *Get information for a layer group*

Description

Get information for a layer group

Usage

```
felt_get_map_layer_group(map_id, layer_group_id, clean = TRUE)
```

Arguments

`map_id`	character, map identifier from url, from `https://felt.com/map/Readable-Name-map_id`
`layer_group_id`	character, layer group identifier from url, from `felt_get_map()`.
`clean`	logical, whether to turn the API response into a tibble::tibble

Value

a tibble::tibble for the layer group, if `clean = TRUE`, otherwise a list

Examples

```
felt_get_map_layer_group(
  map_id = 'Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA',
  layer_group_id = 'rHxyTef7S9CO8W7n1PvBVwC'
)
```

felt_get_user *Obtain information about the user*

Description

Obtain information about the user

Usage

```
felt_get_user(clean = TRUE)
```

Arguments

clean	logical, whether to turn the API response into a tibble::tibble

Value

a tibble::tibble of information about the user

Examples

```
felt_get_user()
```

felt_patch_style *Update Felt Style Information for a Layer*

Description

For details on the Felt Style Language, see https://felt.com/blog/felt-style-language.

Usage

```
felt_patch_style(map_id, layer_id, fsl, clean = TRUE)
```

Arguments

map_id	character, map identifier from url, from https://felt.com/map/Readable-Name-map_id
layer_id	character, layer identifier from url, from felt_get_map()
fsl	A list indicating the Felt style language to update the layer to. It must be valid FSL.
clean	logical, whether to turn the API response into a tibble::tibble

Value

response data

Examples

```
# URL Parks layer
# read as a list
layer <- felt_get_map_layer(map_id = 'TBI8sDkmQjuK2GX9CSiHiUA',
                            layer_id = 'eufG5hWKRRSURHE8YcGGXA',
                            clean = FALSE)
# use the existing style for easiest modification
style <- layer$style
style$label$minZoom <- round(runif(1, 0, 23))

felt_patch_style(map_id = 'Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA',
                 layer_id = 'eufG5hWKRRSURHE8YcGGXA',
                 fsl = style)
```

`felt_refresh_layer` *Refresh an existing layer*

Description

Refresh an existing layer

Usage

```
felt_refresh_layer(map_id, layer_id, file_names, clean = TRUE)
```

Arguments

`map_id`	character, map identifier from url, from `https://felt.com/map/Readable-Name-map_id`
`layer_id`	character, layer identifier from url, from `felt_get_map()`
`file_names`	file names to upload if the original layer was created with a file upload
`clean`	logical, whether to turn the API response into a tibble::tibble

Value

a tibble::tibble with upload information

Examples

```
felt_refresh_layer(map_id = 'Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA',
                   layer_id = 'rCfurZSFShycXHCBw69CdKB',
                   file_names = fs::path_package('feltr',  'towns.geojson'))
felt_refresh_layer(map_id = 'Rockland-2024-Districts-TBI8sDkmQjuK2GX9CSiHiUA',
                   layer_id = 'eufG5hWKRRSURHE8YcGGXA')
```

`felt_update_layer_details` *Update a Layer's Details*

Description

Allows for updates to the name, ordering key, and subtitle.

Usage

```
felt_update_layer_details(
  map_id,
  layer_id,
  layer_group_id = NULL,
  name = NULL,
  ordering_key = NULL,
  subtitle = NULL,
  clean = TRUE
)
```

Arguments

`map_id`	character, map identifier from url, from `https://felt.com/map/Readable-Name-map_id`
`layer_id`	character, layer identifier from url, from `felt_get_map()`
`layer_group_id`	character, layer group identifier from url, from `felt_get_map()`.
`name`	Name of the layer. Defaults to `NULL`.
`ordering_key`	Integer to order layers. Defaults to `NULL`.
`subtitle`	Subtitle for the layer. Defaults to `NULL`.
`clean`	logical, whether to turn the API response into a tibble::tibble

Value

response data

Examples

```
felt_update_layer_details(map_id = 'TBI8sDkmQjuK2GX9CSiHiUA',
                          layer_id = 'eufG5hWKRRSURHE8YcGGXA',
                          subtitle = paste0('tested ', Sys.Date()))
```

`felt_update_layer_group_details` *Update a Layer Group's Details*

Description

Allows for updates to the name, ordering key, and subtitle.

Usage

```
felt_update_layer_group_details(
  map_id,
  layer_group_id,
  name = NULL,
  ordering_key = NULL,
  subtitle = NULL,
  clean = TRUE
)
```

Arguments

`map_id`	character, map identifier from url, from `https://felt.com/map/Readable-Name-map_id`
`layer_group_id`	character, layer group identifier from url, from `felt_get_map()`.
`name`	Name of the layer group. Required.
`ordering_key`	Integer to order by. Defaults to `NULL`.
`subtitle`	Subtitle for the layer group. Defaults to `NULL`.
`clean`	logical, whether to turn the API response into a tibble::tibble

Value

response data

Examples

```
felt_update_layer_group_details(map_id = 'TBI8sDkmQjuK2GX9CSiHiUA',
                                layer_group_id = 'rHxyTef7S9CO8W7n1PvBVwC',
                                name = 'Polling Sites 2020',
                                subtitle = paste0('tested ', Sys.Date()))
```

`felt_update_map_details` *Update a Map's Details*

Description

Allows for updates to the title, description, and level of public access.

Usage

```
felt_update_map_details(
  map_id,
  title = NULL,
  description = NULL,
  public_access = NULL,
  clean = TRUE
)
```

Arguments

`map_id`	character, map identifier from url, from `https://felt.com/map/Readable-Name-map_id`
`title`	Title to use for the map. Defaults to `NULL`.
`description`	Description for the map legend. Defaults to `NULL`.
`public_access`	Degree of public acess. Defaults to `NULL`, which is `view_only`. Text options also include `'private'`, `'view_and_comment'`, and `'view_comment_and_edit'`.
`clean`	logical, whether to turn the API response into a tibble::tibble

Value

response data

Examples

```
felt_update_map_details(map_id = 'TBI8sDkmQjuK2GX9CSiHiUA',
                        title = paste0('Rockland 2024 Districts, tested ', Sys.Date()))
```

key *Check or Get Felt API Key*

Description

Check or Get Felt API Key

Usage

```
has_felt_key()

get_felt_key()

felt_get_key()

felt_has_key()
```

Value

logical if has, key if get

Examples

```
has_felt_key()
```

set_felt_key *Add Entry to Renviron*

Description

Adds Felt API key to .Renviron.

Usage

```
set_felt_key(key, overwrite = FALSE, install = FALSE, r_env = NULL)

felt_set_key(key, overwrite = FALSE, install = FALSE, r_env = NULL)
```

Arguments

key	Character. API key to add to add.
overwrite	Defaults to FALSE. Boolean. Should existing FELT_KEY in Renviron be overwritten?
install	Defaults to FALSE. Boolean. Should this be added to an environment file, r_env?
r_env	Path to install to if install is TRUE.

Value

key, invisibly

Examples

```
example_env <- tempfile(fileext = '.Renviron')
set_felt_key('1234', r_env = example_env)
# r_env should likely be: file.path(Sys.getenv('HOME'), '.Renviron')
```

Index