Package: photosearcher (via r-universe)

October 15, 2024

Title Photo Searcher

Version 1.0

Description Queries the Flick API (https://www.flickr.com/services/api/) to return photograph metadata as well as the ability to download the images as jpegs.

Depends R (>= 3.5.0)

License GPL-3

Encoding UTF-8

LazyData true

Language en_GB

Imports xml2, httr, glue, clisymbols, crayon, sf, lubridate, dplyr

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.2

URL https://docs.ropensci.org/photosearcher

(website) https //github.com/ropensci/photosearcher

BugReports https://github.com/ropensci/photosearcher/issues

Suggests httptest, knitr, roxygen2, testthat (>= 2.1.0), rmarkdown, covr, rplos, wordcloud, tidyverse

VignetteBuilder knitr

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

RemoteUrl https://github.com/ropensci/photosearcher

RemoteRef master

RemoteSha 528c4c549b035b6160ed3d3f2d1ccda9bf56b087

Contents

`download_images` *Download images*

Description

Downloads images based on their Flickr ID. Uses the flickr.photos.getSizes API method from the Flickr API to test whether you have permission to download an image. See https://www.flickr.com/services/api/flickr.photos.getSizes.html for more information on the API method. If permission is available the image is downloaded and saved as a .jpeg in a given save directory.

Usage

```
download_images(
  photo_id,
  save_dir = NULL,
  max_image_height = NULL,
  max_image_width = NULL,
  overwrite_file = FALSE,
  quiet = FALSE
)
```

Arguments

`photo_id`	numeric or character vector. id of photo to download, can be single id, list or column for photo_search outputs
`save_dir`	character. name of directory for photos to be saved in.
`max_image_height`	numeric. maximum number of pixels for images height
`max_image_width`	numeric. maximum number of pixels for images width
`overwrite_file`	logical. Whether to overwritten existing files. if TRUE, files will be overwritten and you will be warned in the output. Default is FALSE.
`quiet`	logical. If TRUE, suppress status messages (if any), and the progress bar.

Details

Please be aware that download times will vary depending on number of photographs, size of photographs, internet speed and other factors. Downloading a large amount of photographs may take some time.

When running the function you need an API key saved as photosearcher_key.sysdata in your working directory. If this is the first function you run you will be prompted to create and enter your API key. The API key will then be saved as photosearcher_key.sysdata in your working directory and is used for all function.

Value

character. A vector of the images attempted to be downloaded and whether they were. If an image was not downloaded, information on why is provided. Images will be saved to `save_dir`.

See Also

Other Get data for known photograph: `get_exif()`, `get_photoinfo()`

Examples

```
## Not run:
download_images(photo_id = 48704764812, save_dir = ".")

download_images(photo_id = 48704764812, max_image_height = 1200,
max_image_width = 1200, save_dir = ".")
## End(Not run)
```

---

findPlaces *Find place location data*

Description

Find the WoeID and other location data for a given place

Usage

```
find_place(place)
```

Arguments

`place`	character. The place for the query

Details

Takes user defined location and returns location data for the search. Uses the flickr.places.find API method from the Flickr API. See https://www.flickr.com/services/api/flickr.places.find.html for more information on the API method.

When running the function you need an API key saved as photosearcher_key.sysdata in your working directory. If this is the first function you run you will be prompted to create and enter your API key. The API key will then be saved as photosearcher_key.sysdata in your working directory and is used for all function.

Value

data.frame. Information on locations that share the name with the search location. Nine variables are returned: place_id, woe_id, latitude, longitude, place_url, place_type, place_type_id, timezone, woe_name.

See Also

Other Information about places: location_tags()

Examples

```
## Not run:
find_place(place = "New York")

find_place(place = "England")

## End(Not run)
```

---

get_exif *Get Exif data*

Description

Returns Exchangeable image file format data for a single photograph. For more information on how Exif data differs from metadata see: https://www.flickr.com/services/api/flickr.photos.getExif.html

Usage

```
get_exif(photo_id = NULL)
```

Arguments

photo_id	Id of photograph

Details

When running the function you need an API key saved as photosearcher_key.sysdata in your working directory. If this is the first function you run you will be prompted to create and enter your API key. The API key will then be saved as photosearcher_key.sysdata in your working directory and is used for all function.

Value

A dataframe of "exchangeable image file format" information for the given photograph

See Also

Other Get data for known photograph: `download_images()`, `get_photoinfo()`

Examples

```
## Not run:
get_exif(photo_id = 48704764812)

## End(Not run)
```

`get_photoinfo` *Get metadata for a single photo*

Description

Returns image metadata for a single photograph.

Usage

```
get_photoinfo(photo_id = NULL)
```

Arguments

`photo_id`	Character, required. The id of the photo to get information for.

Details

When running the function you need an API key saved as photosearcher_key.sysdata in your working directory. If this is the first function you run you will be prompted to create and enter your API key. The API key will then be saved as photosearcher_key.sysdata in your working directory and is used for all function.

Value

Dataframe of information on given image

See Also

Other Get data for known photograph: download_images(), get_exif()

Examples

```
## Not run:
get_photoinfo(photo_id = 48704764812)

## End(Not run)
```

`interesting_list` *Interesting list*

Description

Returns a Flickr generated list of photographs deemed interesting. For information on how this list is calculated see: http://www.steves-digicams.com/knowledge-center/how-tos/online-sharing-social-networki what-is-flickr-interestingness.html#b

Usage

```
interesting_list(date = "2019-01-01")
```

Arguments

date	Character, required. Interestingness list for the date provided. The date should be in the form of "YYYY-MM-DD".

Details

When running the function you need an API key saved as photosearcher_key.sysdata in your working directory. If this is the first function you run you will be prompted to create and enter your API key. The API key will then be saved as photosearcher_key.sysdata in your working directory and is used for all function.

Value

data.frame. Output consists of 57 variables including; latitude and longitude of photograph, date and time it was taken, associated tags and image urls.

Full list of variables returned:

- id: photograph's unique id number
- owner: the unique id of the Flickr user
- secret: photograph unique secret number
- server: Flickr server data
- farm: Flickr server data

- title: photograph title
- ispublic: whether photograph is public; 1 = yes, 0 = no
- isfriend whether user is friend; 1 = yes, 0 = no
- isfamily whether user is family; 1 = yes, 0 = no
- license: use licence of the image see https://www.flickr.com/services/api/flickr.photos.licenses.getInfo.html for details
- datetaken: date and time of image capture
- datetakengranularity: accuracy of image date see https://www.flickr.com/services/api/misc.dates.html for more information on dates
- datetakenunknown: whether date is unknown see https://www.flickr.com/services/api/misc.dates.html for more information on dates
- count_views: number of view the photograph has had,
- count_comments: number of comments on the photograph
- count_faves: number of times the photograph has been favourited
- tags: user defined tags on the photograph
- latitude: latitude of where the image was taken
- longitude: longitude of where the image was taken
- accuracy: accuracy of spatial reference see https://www.flickr.com/services/api/flickr.photos.search.html for more information
- context: a numeric value representing the photo's geotagginess beyond latitude and longitude https://www.flickr.com/services/api/flickr.photos.search.html for more information
- place_id: unique numeric number representing the location of the photograph
- woeid: unique numeric number representing the location of the photograph
- geo_is_family: whether only friends can see geo; 1 = yes, 0 = no
- geo_is_friend: whether only family can see geo; 1 = yes, 0 = no
- geo_is_contact: whether only contact can see geo; 1 = yes, 0 = no
- geo_is_public whether geo is public; 1 = yes, 0 = no
- url_sq: URL for square image
- height_sq: height for square image
- width_sq: width for square image
- url_t: URL for square image thumbnail image 100 on longest side
- height_t: height for thumbnail image 100 on longest side,
- width_t: width for thumbnail image 100 on longest side
- url_s: URL for small square image 75x75
- height_s: height for small square image 75x75
- width_s: width for small square image 75x75
- url_q: URL for large square image 150x150
- height_q: height for large square image 150x150

- width_q: width for large square image 150x150
- url_m: URL for small image 240 on longest side
- height_m: height for small image 240 on longest side
- width_m: width for small image 240 on longest side
- url_n: URL for small image 320 on longest side
- height_n: height for small image 320 on longest side
- width_n: width for small image 320 on longest side
- url_z: URL for medium image 640 on longest side
- height_z: height for medium image 640 on longest side
- width_z: width for medium image 640 on longest side
- url_c: URL for medium image 800 on longest side
- height_c: height for medium image 800 on longest side
- width_c: width for medium image 800 on longest side
- url_l: URL for large image 1024 on longest side
- height_l: height for large image 1024 on longest side
- width_l: width for large image 1024 on longest side
- url_o: URL for original image, either a jpg, gif or png, depending on source format
- height_o: height for original image, either a jpg, gif or png, depending on source format
- width_o: width for original image, either a jpg, gif or png, depending on source format

See Also

Other Search for images: `photo_search()`

Examples

```
## Not run:

interesting_list(date = "2019-01-01")

interesting_list(date = "2017-05-05")

interesting_list(date = "2011-11-25")


## End(Not run)
```

location_tags *Get top tags for a location*

Description

Takes user defined location and returns the top tags related to the location. Uses the flickr.places.tagsForPlace API method from the Flickr API. See https://www.flickr.com/services/api/flickr.places.tagsForPlace.html for more information on the API method.

Usage

```
location_tags(woe_id)
```

Arguments

woe_id	numeric. a "Where on Earth" location tag.

Details

When running the function you need an API key saved as photosearcher_key.sysdata in your working directory. If this is the first function you run you will be prompted to create and enter your API key. The API key will then be saved as photosearcher_key.sysdata in your working directory and is used for all function.

Value

character. List of the top 100 tags associated with the woe_id.

See Also

Other Information about places: findPlaces

Examples

```
## Not run:
location_tags(woe_id = 35356)

## End(Not run)
```

photo_search *Search for photo metadata*

Description

Returns image metadata for photos matching the search terms.

Usage

```
photo_search(
  mindate_taken = NULL,
  maxdate_taken = NULL,
  mindate_uploaded = NULL,
  maxdate_uploaded = NULL,
  user_id = NULL,
  text = NULL,
  tags = NULL,
  tags_any = TRUE,
  bbox = NULL,
  woe_id = NULL,
  sf_layer = NULL,
  has_geo = TRUE
)
```

Arguments

mindate_taken	Character, or date required. Minimum taken date. Photos with an taken date greater than or equal to this value will be returned. The date should be in the form of "YYYY-MM-DD".
maxdate_taken	Character, or date required. Maximum taken date. Photos with an taken date less than or equal to this value will be returned. The date should be in the form of "YYYY-MM-DD".
mindate_uploaded	Character or date, optional. Minimum upload date. Photos with an upload date greater than or equal to this value will be returned. The date can be in the form of a unix timestamp or mysql datetime.
maxdate_uploaded	Character or date, optional. Maximum upload date. Photos with an upload date less than or equal to this value will be returned. The date can be in the form of a unix timestamp or mysql datetime.
user_id	Character, optional. The Flickr ID of the user who's photo to search. If this parameter isn't passed then everybody's public photos will be searched.
text	Character, optional. A free text search. Photos who's title, description or tags contain the text will be returned. You can exclude results that match a term by prepending it with a - character. Free text searches for words in order provided, for example a search for "climbing rock" will be different to "rock climbing".

tags	Character vector, optional. A comma-delimited list of tags. Photos with one or more of the tags listed will be returned. You can exclude results that match a term by prepending it with a - character.
tags_any	Logical, optional. If TRUE, photos containing any of the tags will be returned. If FALSE, only photos containing all tags will be returned. Defaulted to return any tags.
bbox	String, optional bounding box of search area provide as: "minimum_longitude,minimum_latitude,maximu
woe_id	String, optional "where on earth identifier" can be supplied instead of bbox. Use find_place to obtain woe_id for a place.
sf_layer	A simple features layer, optional, area to search for photos, can be supplied instead of a bbox or woeID.
has_geo	Logical, optional parameter for whether returned photos need associated spatial data.

Details

Uses the flickr.photos.search API method from the Flickr API. This search method requires a limiting factor to prevent parameterless searches - to ensure this is met the function requires both a minimum and a maximum date that searched photographs were taken on. See https://www.flickr.com/services/api/flickr.photos.search.html for more information on the API method.

When running the function you need an API key saved as photosearcher_key.sysdata in your working directory. If this is the first function you run you will be prompted to create and enter your API key. The API key will then be saved as photosearcher_key.sysdata in your working directory and is used for all function.

Value

data.frame. Output consists of 57 variables including; latitude and longitude of photograph, date and time it was taken, associated tags and image urls.

Full list of variables returned:

- id: photograph's unique id number
- owner: the unique id of the Flickr user
- secret: photograph unique secret number
- server: Flickr server data
- farm: Flickr server data
- title: photograph title
- ispublic: whether photograph is public; 1 = yes, 0 = no
- isfriend whether user is friend; 1 = yes, 0 = no
- isfamily whether user is family; 1 = yes, 0 = no
- license: use licence of the image see https://www.flickr.com/services/api/flickr.photos.licenses.getInfo.html for details
- datetaken: date and time of image capture

- datetakengranularity: accuracy of image date see https://www.flickr.com/services/api/misc.dates.html for more information on dates
- datetakenunknown: whether date is unknown see https://www.flickr.com/services/api/misc.dates.html for more information on dates
- count_views: number of view the photograph has had,
- count_comments: number of comments on the photograph
- count_faves: number of times the photograph has been favourited
- tags: user defined tags on the photograph
- latitude: latitude of where the image was taken
- longitude: longitude of where the image was taken
- accuracy: accuracy of spatial reference see https://www.flickr.com/services/api/flickr.photos.search.html for more information
- context: a numeric value representing the photo's geotagginess beyond latitude and longitude https://www.flickr.com/services/api/flickr.photos.search.html for more information
- place_id: unique numeric number representing the location of the photograph
- woeid: unique numeric number representing the location of the photograph
- geo_is_family: whether only friends can see geo; 1 = yes, 0 = no
- geo_is_friend: whether only family can see geo; 1 = yes, 0 = no
- geo_is_contact: whether only contact can see geo; 1 = yes, 0 = no
- geo_is_public whether geo is public; 1 = yes, 0 = no
- url_sq: URL for square image
- height_sq: height for square image
- width_sq: width for square image
- url_t: URL for square image thumbnail image 100 on longest side
- height_t: height for thumbnail image 100 on longest side,
- width_t: width for thumbnail image 100 on longest side
- url_s: URL for small square image 75x75
- height_s: height for small square image 75x75
- width_s: width for small square image 75x75
- url_q: URL for large square image 150x150
- height_q: height for large square image 150x150
- width_q: width for large square image 150x150
- url_m: URL for small image 240 on longest side
- height_m: height for small image 240 on longest side
- width_m: width for small image 240 on longest side
- url_n: URL for small image 320 on longest side
- height_n: height for small image 320 on longest side

- width_n: width for small image 320 on longest side
- url_z: URL for medium image 640 on longest side
- height_z: height for medium image 640 on longest side
- width_z: width for medium image 640 on longest side
- url_c: URL for medium image 800 on longest side
- height_c: height for medium image 800 on longest side
- width_c: width for medium image 800 on longest side
- url_l: URL for large image 1024 on longest side
- height_l: height for large image 1024 on longest side
- width_l: width for large image 1024 on longest side
- url_o: URL for original image, either a jpg, gif or png, depending on source format
- height_o: height for original image, either a jpg, gif or png, depending on source format
- width_o: width for original image, either a jpg, gif or png, depending on source format
- description: Flickr user generated text description of the photograph

See Also

Other Search for images: `interesting_list()`

Examples

```
## Not run:
photo_search(
  mindate_taken = "2019-01-01",
  maxdate_taken = "2019-01-02",
  text = "tree",
  bbox = "-13.623047,47.279229,3.251953,60.630102",
  has_geo = TRUE
  )

photo_search(
  mindate_taken = "2017-01-01",
  maxdate_taken = "2017-01-02",
  mindate_uploaded = "2017-03-04",
  maxdate_uploaded = "2017-05-05",
  tags = "lake"
  )

photo_search(
  mindate_taken = "2018-01-01",
  maxdate_taken = "2018-01-03",
  tags = c("mountain", "lake"),
  tags_any = TRUE
)

photo_search(
  mindate_taken = "2018-01-01",
```

```
  maxdate_taken = "2018-01-03",
  tags = c("mountain", "lake"),
  tags_any = FALSE
)


## End(Not run)
```

`reddit_search` *Search Reddit for posts*

Description

Searches for a a given term or set of terms across Reddit between chosen dates. Can be limited to a specific subreddit.

Usage

```
reddit_search(
  search_term = NULL,
  subreddit = NULL,
  start_date = "2020-01-01",
  end_date = "2021-01-01"
)
```

Arguments

`end_date`

`related_tags` *Get related tags*

Description

Takes a tag and returns the top tags related to that tag.

Usage

```
related_tags(tag)
```

Arguments

`tag`	character. tag to search.

Details

Uses the flickr.tags.getRelated API method from the Flickr API. See https://www.flickr.com/services/api/flickr.tags.getRelated.html for more information on the API method.

When running the function you need an API key saved as photosearcher_key.sysdata in your working directory. If this is the first function you run you will be prompted to create and enter your API key. The API key will then be saved as photosearcher_key.sysdata in your working directory and is used for all function.

Value

character. Tags most associated with input tag.

Examples

```
## Not run:
related_tags(tag = "car")

related_tags(tag = "monkey")

related_tags(tag = "river")

## End(Not run)
```

---

user_info *Get user information*

Description

Takes Flickr user ID and returns the profile data.

Usage

```
user_info(user_id)
```

Arguments

user_id	character. The id of the user you wish to obtain information for.

Details

Uses the flickr.profile.getProfile API method from the Flickr API. See https://www.flickr.com/services/api/flickr.profile.getProfile.html for more information on the API method.

See https://www.pixsy.com/academy/flickr-id/ for a guide on finding your Flickr ID.

When running the function you need an API key saved as photosearcher_key.sysdata in your working directory. If this is the first function you run you will be prompted to create and enter your API key. The API key will then be saved as photosearcher_key.sysdata in your working directory and is used for all function.

Value

data.frame. Dataframe of 5 variables from the searched users publicly available information: id, occupation, hometown, city, country.

Examples

```
## Not run:
user_info(user_id = "155421853@N05")

## End(Not run)
```

Index