Flickr

Yi Lin Liu

April 4, 2017

Contents

Introduction:

Basics of Flickr:

- Guide to use Flickr
- Guide to API

Learning API:

- Flickr API
  - **NOTE: not supported by Flickr, so might not be stable.**
  - All uses of the API must abide the community guidelines
- Results can be cached for up to 24 hours to reduce API load.
- Flickr has its own shortened URL. Shortened URL

- To perform requests, a calling convention is needed, send a request to its endpoint specifying the requried parameters (method,api_key) and other arguments, a formatted response will be recieved.
  - Endpoint: `https://api.flickr.com/services`
- All requests/data are expected to be UTF-8 enconded.
- Multiple response formats are provided: xml, json, REST
- Oauth is used for authentication.
- **Many methods for the API does not seem to require a token.**
- URLs
  - the urls are not returned, instead the parameters required to form the urls are provided in the response. Please consult the documentation

---

Rate Limit:

- Staying under aggregate 3600 requests per hour per key will be allowed.

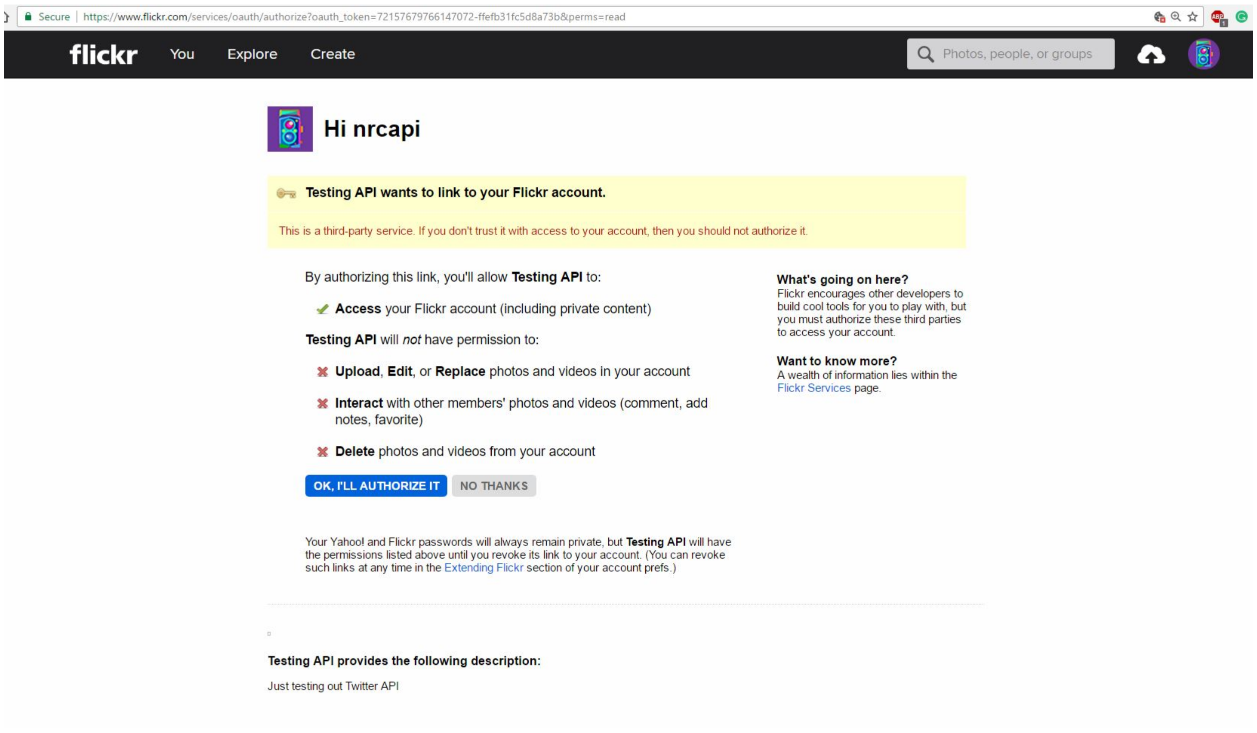
Obtaining Authorization:

- API Key
  - click on *Request an API Key* under "Get your API Key"
  - select non-commercial.
  - An call back URL must be specified.
- Generating Key
  - all the listed URLs are available on this website.
- The token does not seem to have an expiration date.
- **FOR THE FOLLOWING TO WORK, when R opens the popup, change `permission=read` to `perms=read` in the URL. The permissions must be specified or the authentication page will show `permission set not recognized` error.**

```r
rm(list = ls())

library(httr)

# ##  Retrieved from API Documentation + Generated Key
# app_Key <- "08adb0273ae63e5c07c249f5a621e7cb"
# app_Secret <- "89e0a70bfb48e7f4"
# request_URL <- "https://www.flickr.com/services/oauth/request_token"
# auth_URL <- "https://www.flickr.com/services/oauth/authorize"
# access_URL <-"https://www.flickr.com/services/oauth/access_token"
#
# token.endpoints <- oauth_endpoint(request= request_URL, authorize = auth_URL,
#                                  access = access_URL)
# token.app <- oauth_app("Testing API",key=app_Key, secret = app_Secret )
# token <- oauth1.0_token(token.endpoints, token.app,permission='read',cache=F)
#
```


Secure
https://www.flickr.com/services/oauth/authorize?oauth_token=72157679766147072-ffefb31fc5d8a73b&perms=read
flickr
You
Explore
Create
Photos, people, or groups

Hi nrcapi

Testing API wants to link to your Flickr account.

This is a third-party service. If you don't trust it with access to your account, then you should not authorize it.

By authorizing this link, you'll allow **Testing API** to:

✓ **Access** your Flickr account (including private content)

Testing API will *not* have permission to:

✖ **Upload**, **Edit**, or **Replace** photos and videos in your account

✖ **Interact** with other members' photos and videos (comment, add notes, favorite)

✖ **Delete** photos and videos from your account

OK, I'LL AUTHORIZE IT   NO THANKS

Your Yahoo! and Flickr passwords will always remain private, but **Testing API** will have the permissions listed above until you revoke its link to your account. (You can revoke such links at any time in the Extending Flickr section of your account prefs.)

What's going on here?
Flickr encourages other developers to build cool tools for you to play with, but you must authorize these third parties to access your account.

Want to know more?
A wealth of information lies within the Flickr Services page.

Testing API provides the following description:

Just testing out Twitter API

Figure 1: Correct Auth Page

```
#
# ##  Saving Token:
# saveRDS(token,file="token.R")
```

API Methods of Interest:

- flickr.contacts.getListRecentlyUploaded: retrieves a list of contacts(friends) for a user who have recently posted as well as number of photos.
- flickr.photos.search: The public search method that returns a list of photos matching the tags specified.
- flickr.photos.getExif: This method returns the EXIT/TIFF/GPS tags associated with the photo.
- More photo methods:
  - popular photos
  - recent photos
  - contact public photos
  - comments to a photo
  - location for photos
- The listed methods mainly focuses on dealing with photos. For methods that allows the exploration of friendship (contacts), finiding users, please refer to the Flickr API Documentation

---

Test Account:

Example 1:

Goal:

To use the public search to search for photos and try to extract the meta data as well as the location data and save to a file.

```r
rm(list = ls())
library(httr)
library(jsonlite)
library(xml2)
library(rvest)

token <- readRDS("./token.R")
app_Key <- "08adb0273ae63e5c07c249f5a621e7cb"
app_Secret <- "89e0a70bfb48e7f4"
```

1: Searching for public photos:

- Photo search
- PlaceId search

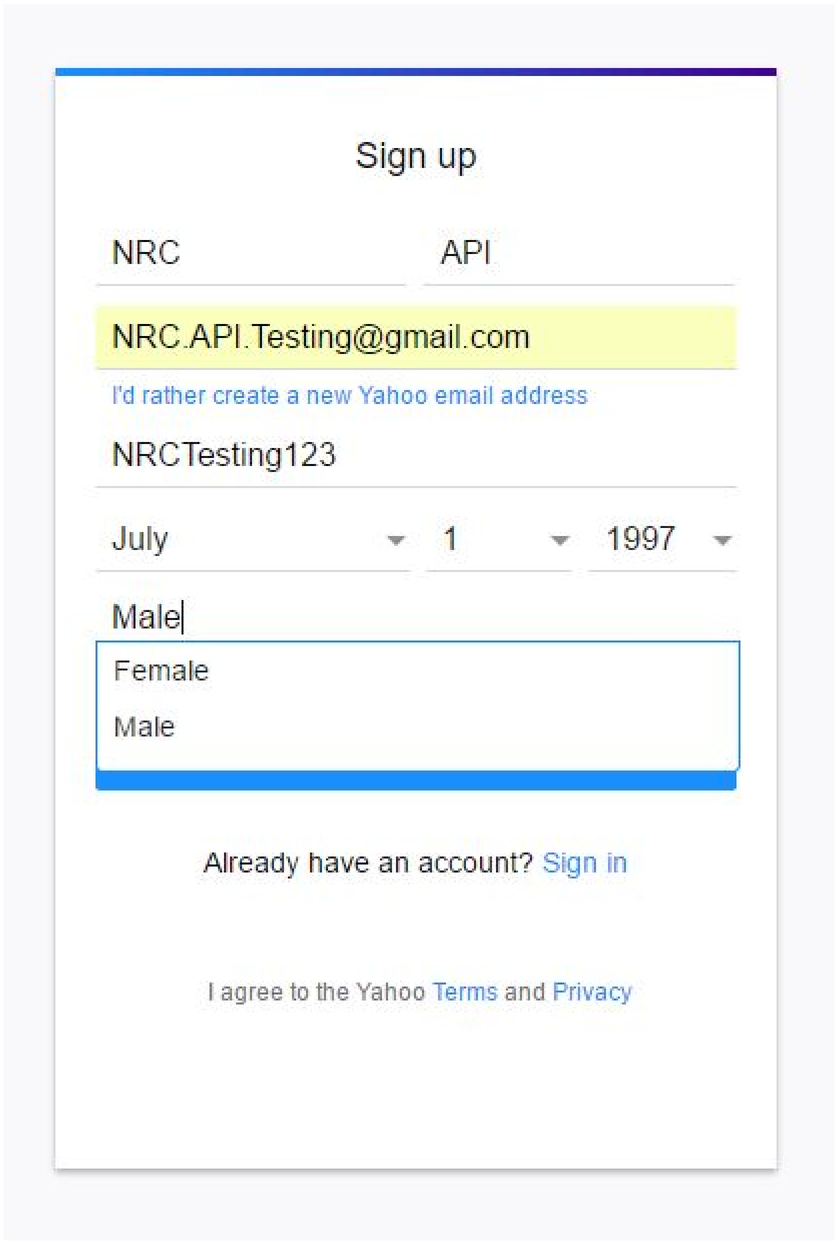
```r
endpoint <- "https://api.flickr.com/services/rest"
arguments <- c(method="flickr.photos.search",
               api_key=app_Key,
               tags = "river,mountain", placeid ="",
              has_geo=1,
              per_page=10)

##Finding the placeid for Canada:
place_args=c(method="flickr.places.find",
             api_key = app_Key,
             query="Canada")
place_Id_Request <- paste(endpoint,
    paste(names(place_args), place_args,sep="=",collapse="&"),
    sep="?")

raw_Place <- GET(place_Id_Request)

##  Validation of requests will be ignored for now.

##  Just going to take the ID of the first one,
```


Sign up

NRC

API

NRC.API.Testing@gmail.com

I'd rather create a new Yahoo email address

NRCTesting123

July 1 1997

Male

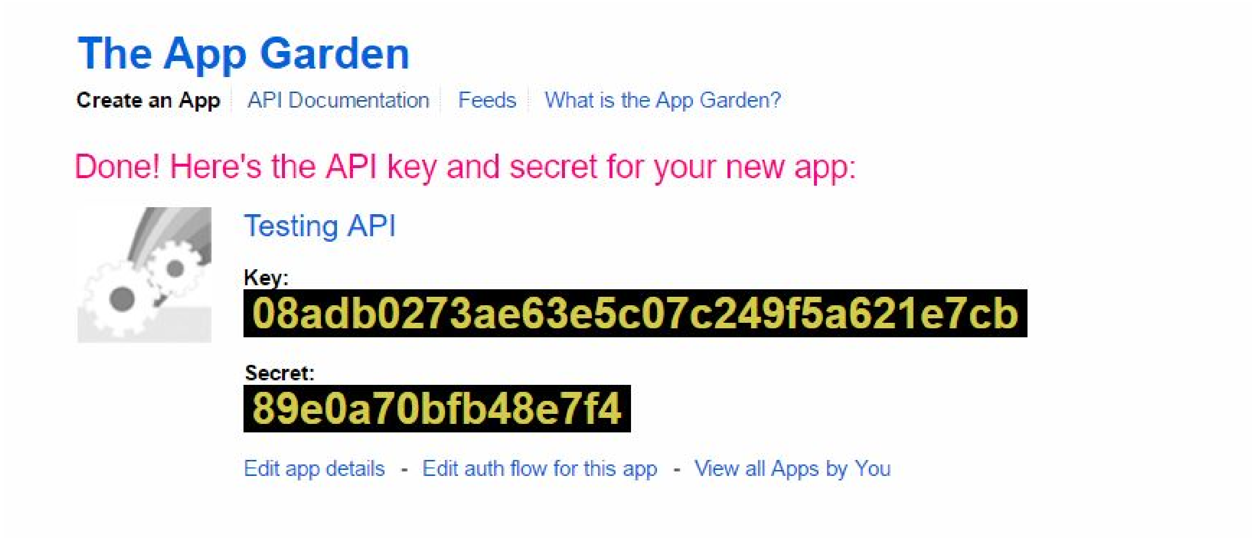
Female

Male

Already have an account? Sign in

I agree to the Yahoo Terms and Privacy

Figure 2: Registration



Done! Here's the API key and secret for your new app:

Testing API

Key:
08adb0273ae63e5c07c249f5a621e7cb

Secret:
89e0a70bfb48e7f4

Figure 3: App Key and Secret

```
##  manually verified to be canada. For more accurate,
##  use lat and lng to search for place id
place_id <- xml2::read_xml(raw_Place$content) %>%
    xml_node("places place") %>%
    xml_attr("place_id")


## adding the place id back into arguments
arguments["placeid"]=place_id



##searching

search_Req <- paste(endpoint,
    paste(names(arguments),arguments,sep="=",collapse="&"),
    sep="?")
search_Raw <- GET(search_Req)
ids <- xml2::read_xml(search_Raw$content) %>%
        xml_nodes("photos photo") %>%
        xml_attr("id")
secrets <-xml2::read_xml(search_Raw$content) %>%
        xml_nodes("photos photo") %>%
        xml_attr("secret")
```

2: Getting all Metadata:

- Saved under `meta.txt`.

```
##  Looping through to extract the metadata
##  Sometimes, the permission is not granted.
```

```r
    ##  It is still recorded

    for(i in 1:length(ids)){
        meta_Args <- c(method="flickr.photos.getExif",
                   api_key=app_Key,
                   photo_id=ids[i],
                   secret=secrets[i])

        ##  Making request
        meta_Req <- paste(endpoint,
        paste(names(meta_Args),meta_Args,sep="=",collapse="&"),
        sep="?")

        metadata = GET(meta_Req)
        metadata = rawToChar(metadata$content)

        ##  Saving returned XML to a file
        write.table(metadata,file="meta.txt",append=T,
                    sep="\n\n",col.names=F,row.names=F)
    }
```

3: Getting Location Data:

- Saved under `location.txt`.

```r
for(i in 1:length(ids)){
        args <- c(method="flickr.photos.geo.getLocation",
                   api_key=app_Key,
                   photo_id=ids[i])

        ##  Making request
        req <- paste(endpoint,
        paste(names(args),args,sep="=",collapse="&"),
        sep="?")

        data = GET(req)
        data = rawToChar(data$content)

        ##  Saving returned XML to a file
        write.table(data,file="location.txt",append=T,
                    sep="\n\n",col.names=F,row.names=F)
    }
```

Mini-Project:

Goal:

___ Search for Canada day 2015 near parliment through the twitter and flickr API.___

Twitter API:

- twitteR documentation
- search API Twitter

```
    library(twitteR)
    library(httr)

    consumer_Key_T <- "oQ3PqERg75kPtgBcgOLaFShSC"
    consumer_Secret_T <- "d4cxaKc1Dt3ugagruUNPtWzvmqGHx8WvwYAQ8MywUqTIVTTj9O"
    access_Token_T <- "833674399224061952-tL4gGOyGUrz84IbVlkkAmQzqUPahL1N"
    access_Secret_T <- "qkNmkD7TU5uZtIENW3r5K20wkqfbL6w37xyXLwelYBZg6"


    ##  Intially, the function asks the user to cache the credentials and
    ##  will be used for another session.
    setup_twitter_oauth(consumer_Key_T,consumer_Secret_T,access_Token_T,access_Secret_T)
```

```
## [1] "Using direct authentication"
```

```
    ##  Finding Lat and Lng for Canadian Parliment
    ##  can be done through google API, for the sake of time.
    ##  The lat lng was found on the internet

   lat <- 45.4236
    lng <- -75.7009
    search_Radius <- "3km"
```

1: Searching Public Twitter API:

- Since the Twitter Search API does not allow searches for tweets older than a week. This can not be done through the offical Twitter API.
- Instead there seems to be libraries for other languages that tries to solve this problem. Old Tweets

Flickr

```
    rm(list = ls())
    library(httr)
    library(jsonlite)
    library(xml2)
    library(rvest)
    library(XML)

    app_Key <- "08adb0273ae63e5c07c249f5a621e7cb"
    endpoint <- "https://api.flickr.com/services/rest"
```

1: Getting Photo Search Data

- Notes about the public search:

    - The search using the woeid which outlines the area of parliment hill returns more results that merely using the place_id.(pictures of streets around parliment hill were also shown using woeid).
    - It took some experimentation but generally, using tags to search is more effective than text.
- Returned response is saved under photo.xml.

```r
min_date <- as.POSIXct("2015-07-01")
max_date <- as.POSIXct("2015-08-01")
lng <- -75.7009
lat <- 45.4236

## finding placeId for Parliment hill
place_args=c(method="flickr.places.findByLatLon",
             api_key = app_Key,
             lat=lat,
             lon=lng)
place_Request <- paste(endpoint,
    paste(names(place_args), place_args,sep="=",collapse="&"),
    sep="?")
raw_Place <- GET(place_Request)

##Getting the place id:
placeid <- xml2::read_xml(raw_Place$content) %>%
    xml_node("places place") %>%
    xml_attr("place_id")

woeid <- xml2::read_xml(raw_Place$content) %>%
    xml_node("places place") %>%
    xml_attr("woeid")




##  Arguments for Search Request
args <- c(method="flickr.photos.search",
          api_key=app_Key,
        text = URLencode("Canada Day 2015 Parliament Hill"),
        tags = URLencode("Canada Day 2015, Parliament Hill, happy Canada Day"),
        woeid = woeid,
        min_taken_date = min_date,
        max_taken_Date = max_date)

## Search data
search_Request <- paste(endpoint,
    paste(names(args), args,sep="=",collapse="&"),
    sep="?")

raw <- GET(search_Request)
xml <- read_xml(rawToChar(raw$content))

## saving search requests
write_xml(xml,"photos.xml")
```

2: Getting Photo URLs:

- To form a photo url, refer to URL

- Photo URL saved under photoURL.txt.

```
##  Going to make a list of the URLs of the photos
##  https://farm{farm-id}.staticflickr.com/{server-id}/{id}_{secret}.jpg

## Get everything into a data frame

attrs <- xml %>% xml_node("photos") %>%
    xml_nodes("photo") %>%
    xml_attrs()


URLs <- lapply(attrs, function(x){
    x <- unlist(x)
    url <- paste("https://farm",
                 x["farm"],".staticflickr.com/",
                 x["server"],"/",x["id"],
                 "_",x["secret"],".jpg",
                 sep="")
})

write.table(URLs,"photoUrls.txt",
            col.names=F,row.names=F,
            sep="\n")
```

3: Getting information about photos:

- saved under info.txt.

```
##  Getting information, might not have
##  permissions
for(i in 1: length(attrs)){

    x<- unlist(attrs[i])
    info_args <- c(method="flickr.photos.getInfo",
          api_key=app_Key,photo_id = x[["id"]],
          secret = x[["secret"]])

    info_Req <- paste(endpoint,
    paste(names(info_args),info_args,sep="=",collapse="&"),
    sep="?")

    rawInfo <-GET(info_Req)
    info <- rawToChar(rawInfo$content)


    ##  Appending xml Files seems to be a issue
     write.table(info,file="info.txt",append=T,
                sep="\n",col.names=F,row.names=F)
}
```