RSoundCloud vignette

Jasper Ginn

July 13, 2015

Introduction

The RSoundCloud package contains two functions with which to query the SoundCloud API. It further contains one convenience function aimed at helping the user to obtain a client id. This id is necessary to query the API.

Package information

- *package name:* RSoundCloud
- *version:* v0.1a
- *maintainer:* Jasper Ginn (jasperginn@gmail.com)
- *issue reports:* https://github.com/JasperHG90/RSoundCloud/issues

Installing

The package is available from GitHub. We can install it via the 'devtools' package.

```r
# Install devtools if not already installed
if(!require("devtools", character.only = T)) install.packages("devtools")
# Load
library(devtools)
# Install RSoundCloud
install_github("JasperHG90/RSoundCloud")
# Load
library(RSoundCloud)
```

Setting up RSoundCloud

In order to query the API, you need to register as a soundcloud developer. You can simply do this with your existing soundcloud account. Then, you need to register an application and copy the 'client id' to R. You can do this by calling the `loginDetails` function.

```r
# Set up RSoundCloud
#clientID <- loginDetails() # Copy-paste your client ID in the R Console below
clientID <- list("clientID" = "45afab70863909e48cfb2f1314d9380a")
```

This function simply returns a list with your client ID. At this point, you're all set.

Querying the SoundCloud API

RSoundCloud contains two functions to query the SoundCloud API: `SCapi_general` and `SCapi_specific`.

Making generic queries

`SCapi_general` can be used for generic queries (e.g. users, comments, tracks). This is similar to the Twitter firehose.

```r
# Query users 1-50
t1 <- Sys.time()
users <- SCapi_general(clientID$clientID, type = "users", limit = 50)
#Sys.time() - t1 # 0.108 secs
```

By default, the `SCapi_general` function returns 50 results. Additionally, the SoundCloud API returns at most 200 results per query. Querying more than 200 results is possible by paginating through the results.

```r
# Query users 1-200
t1 <- Sys.time()
users <- SCapi_general(clientID$clientID, type = "users", limit = 200)
#Sys.time() - t1 # 0.737 secs

# Query users 1-1000 (== 5 separate calls to the API)
t1 <- Sys.time()
users <- SCapi_general(clientID$clientID, type = "users", limit = 1000)
#Sys.time() - t1 # 5.63 secs
```

You can also specify an offset. The offset value determines at what point the query starts. That is, with an offset value of 1000 we start querying from user 1000.

```r
# Query users 1000-2000
t1 <- Sys.time()
users <- SCapi_general(clientID$clientID, type = "users", limit = 1000,
                       filter = NULL, offset = 1000)
#Sys.time() - t1 # 4.67 secs
```

The `filter` argument can be used to filter the results.

```r
filt <- SCapi_general(clientID$clientID, type = "users", limit = 50,
                      filter = list("q" = "the-bugle"))
```

Note that this argument **only** takes a list as an value. For an overview of the various filters, you can consult the SoundCloud API documentation

Searches are similar for tracks and comments.

```r
# Tracks
t1 <- Sys.time()
tracks <- SCapi_general(clientID$clientID, type = "tracks", limit = 500)
#Sys.time() - t1 # 1.007 secs.
# Comments
t1 <- Sys.time()
comments <- SCapi_general(clientID$clientID, type = "comments", limit = 500)
#Sys.time() - t1 # 0.923 secs.
```

Filters can also be used for tracks and comments.

```r
# Tracks
t1 <- Sys.time()
tracks <- SCapi_general(clientID$clientID, type = "tracks", limit = 500,
                        filter = list("created_at" = "last_hour"))
#Sys.time() - t1 # 3.33 secs.
```

Althrough you *can* use multiple filters, a lot of combinations return errors. Consult the SoundCloud API documentation for more information.

```r
# This returns an error
tracks <- SCapi_general(clientID$clientID, type = "tracks", limit = 500,
                        filter = list("created_at" = "last_hour",
                                      "q" = "Emily"))
```

This returns the following error:

```
Error in file(con, "r") : cannot open the connection
In addition: Warning message:
In file(con, "r") : cannot open: HTTP status was '400 Bad Request'`
```

When querying groups, it is not uncommon to get a '502 Bad Gateway' - error. As such, queries for groups should be limited to 25.

```r
# Groups
t1 <- Sys.time()
groups <- SCapi_general(clientID$clientID, type = "groups", limit = 25)
#Sys.time() - t1 # 8.32 secs.
```

Currently, it is not possible to increase the limit and paginate per 25 results. If you really want to paginate through more than 25 results for groups, you can change `offset` to paginate through the results.

```r
# Query 100 results for groups
master <- list()
t1 <- Sys.time()
for(i in seq(25,100,25)) {
  groups <- SCapi_general(clientID$clientID, type = "groups", limit = 25, offset = i)
  master <- c(master,groups)
}
#Sys.time() - t1 # 25.86 secs.
```

Making specific queries

`SCapi_specific` can be used to query information about specific users, tracks, groups etc. It allows you to search by soundcloud name, soundcloud id, or by url.

```r
# Query by name
t1 <- Sys.time()
us <- SCapi_specific(clientID$clientID, "the-bugle", type="name", query_type="users")
#Sys.time() - t1 # 0.103 secs
# Query by soundcloud ID
```

```
t1 <- Sys.time()
us <- SCapi_specific(clientID$clientID, "9818871", type="id", query_type="users")
#Sys.time() - t1 # 0.0439 secs (faster because it does not have to resolve a url)
# Query by url
t1 <- Sys.time()
us <- SCapi_specific(clientID$clientID, "https://soundcloud.com/the-bugle",
                     type="url", query_type="users")
#Sys.time() - t1 # 0.105 secs
```

Note that the URL does not necessarily have to point to the main page of a user.

```
# Query by url
us <- SCapi_specific(clientID$clientID, "https://soundcloud.com/the-bugle/likes",
                     type="url", query_type="users")
```

This works because we specify the `query_type` argument to look for users (and not likes). Selecting "users" will always return the generic information about a user.

If we want to query the tracks belonging to a user, we have several options.

```
# Get tracks for user # METHOD1
t1 <- Sys.time()
us <- SCapi_specific(clientID$clientID, "https://soundcloud.com/the-bugle/likes",
                     type="url", query_type="users", get="tracks")
#Sys.time() - t1 # 0.492 secs
# Get tracks for user # METHOD2
t1 <- Sys.time()
us <- SCapi_specific(clientID$clientID, "https://soundcloud.com/the-bugle/tracks",
                     type="url", query_type="tracks")
#Sys.time() - t1 # 0.128 secs (faster because less computation needed)
```

Both of these options work. In method 1, the URL we provide is merely used to resolve the user name (see SoundCloud documentation on resolving). This is governed by the `query_type` argument, which is specified to look for "users". The `get` argument then stipulates that we are looking for the tracks belonging to that user (see the SoundCloud documentation on GET).

For method 2, we provide a direct URL to the tracks of the user, and tell `SCapi_specific` to query tracks directly via the `query_type` argument.

As with the `SCapi_general` function, we can add filters to narrow down our search.

```
# Filter for tracks created after 20 May 2015
us <- SCapi_specific(clientID$clientID, "9818871", type="id",
                     query_type="users", get="tracks",
                     filter=list("created_at[from]" = "2015-05-20"))
```