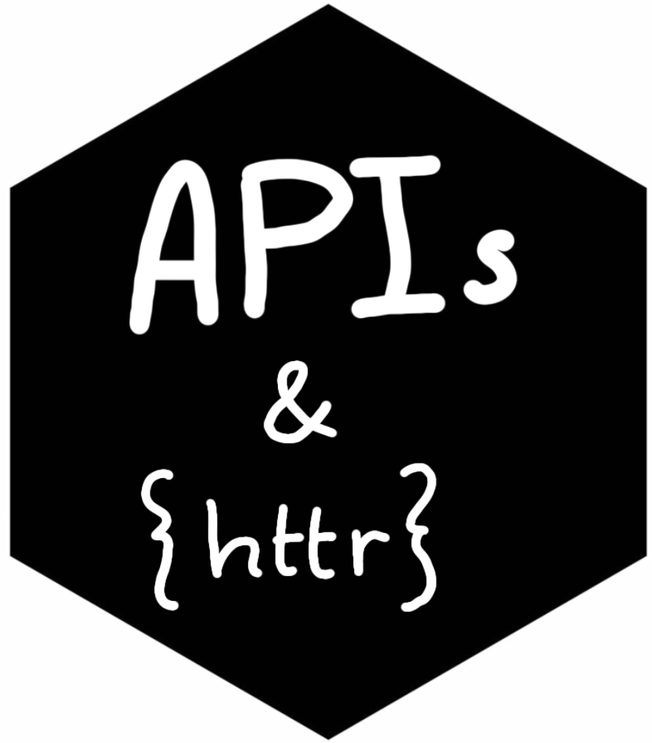
Acquiring data via an API

Data Acquisition and Distribution

Dr Zak Varty

Acquiring data via an API

- You will often have to gather data for yourself.
- There must be an easier way than scraping webpages.
- APIs to the rescue!
- See also: Introduction to APIs and DIY web data



Why do I need to know about APIs?

- APIs are a common method for sharing data within and between businesses.

> An API, or **application programming interface**, is a set of rules that allows different software applications to communicate with each other.

- Convenient way to access data programatically. Benefits include:
  - **Automation** Faster and less chance of human error;
  - **Standardisation** Replication and code your data retrieval.

What is an API?

- **Etiquette** = Rules for human communication
- **Protocol** = Rules for computer communication

APIs are a standard protocol for different programs to interact with one another.

This allows modular development of specialised tools and greater progress overall.

API Communication

There are two sides to communication and when machines communicate these are known as the **server** and the **client**.

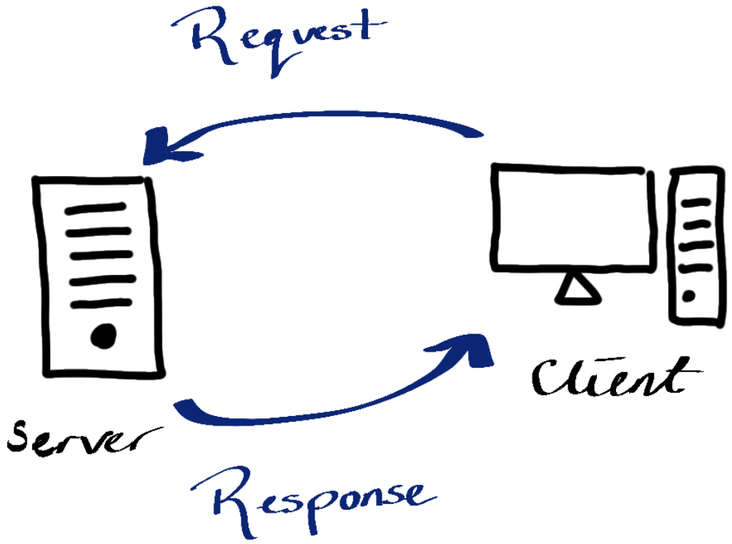
- **Server**: A program or computer used to store data or run programs on behalf of another program or computer.
- **Client**: Any program or computer that uses the server.

HTTP

An API is a set of rules for computer communication, but how do they “talk” to one another? **Hyper Text Transfer Protocol** (HTTP), or it’s secure sibling HTTPS.

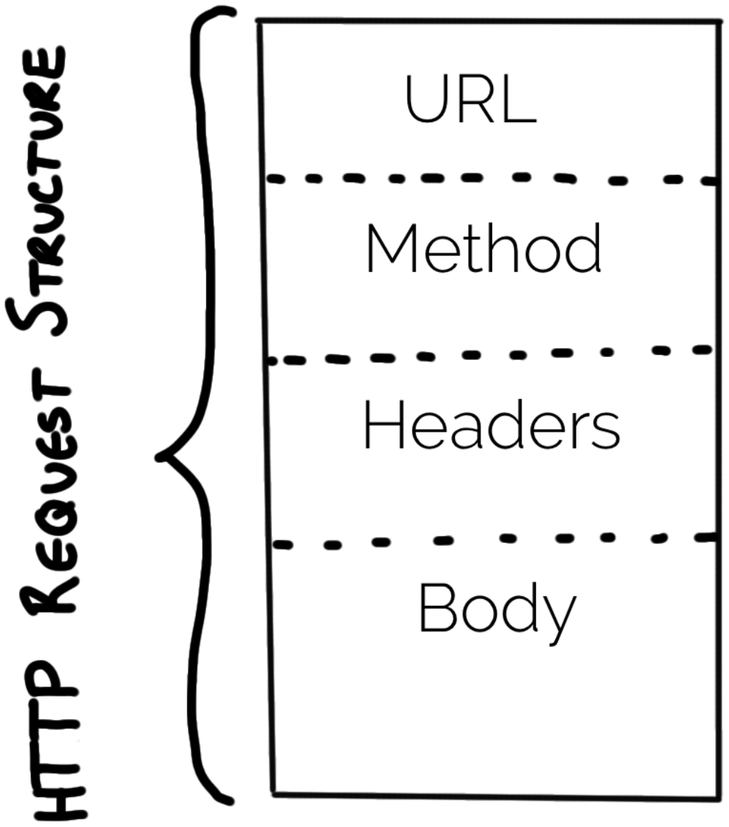
`https://www.imperial.ac.uk`

Uses a **request-response** model of communication



HTTP Requests

An HTTP request consists of:



- Uniform Resource Locator (URL)
- Method (type of action requested)
- Headers (meta-information)
- Body (data)

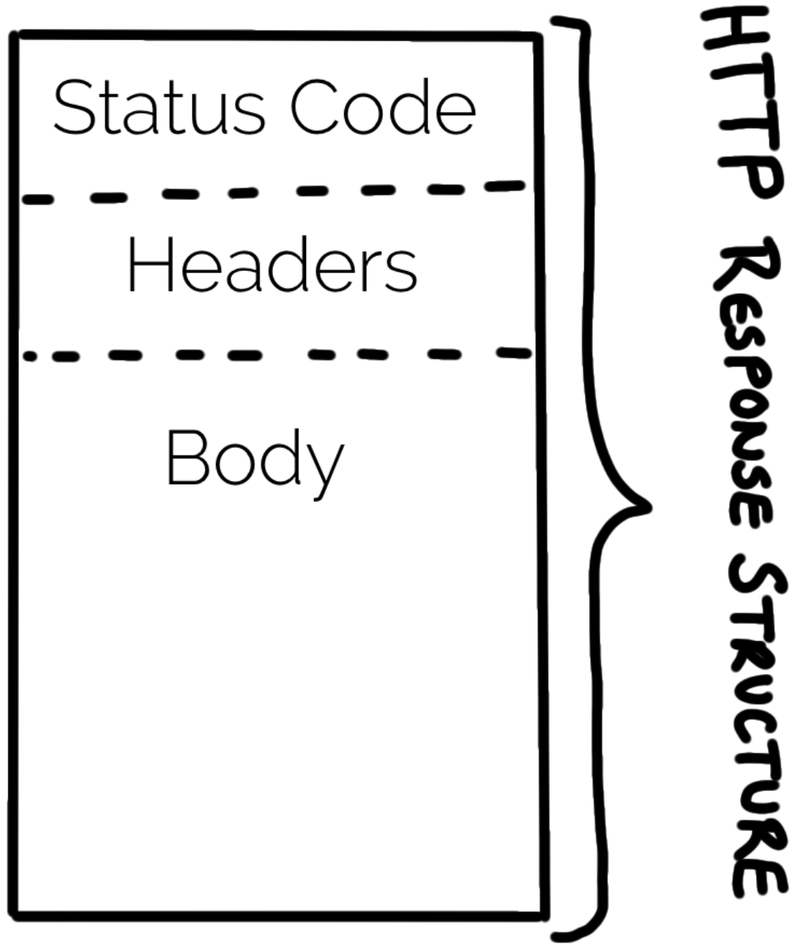
HTTP Methods

The most common HTTP Methods are:

- `GET`
- `POST`
- `PUT`
- `PATCH`
- `DELETE`

The `GET` request is all you need for data acquisition, but the others will be used if you set up your own API to share data with others.

HTTP Responses



- No URL
- No method
- Status Code

Example status codes: `200`, `404`, `503`.

Successful API access gives data in JSON or XML format.

Authentication

> **Authentication** is a way to ensure that only authorized clients are able to access an API.

- Including secrect information in each request
- We consider two methods: **Basic Authentication** and **API Keys**.

Authentication: Basic Auth vs API Keys

Basic Authentication

- User name (and password)
- Enrypted in Headers
- 401 error if not matching
- Can’t control permissions

API Keys

- random character sequence provided by server
- 401 error if not matching
- Individualised permissions
- API use tracking

```
http://example.com?api_key=my_secret_key
```

API Wrappers

We've learned a lot about how computers communicate - how do we put this into practice?

- Mostly use this new internet knowledge for debugging
- API Wrapper functions should be your go-to, if they exist

rOpenSci has a curated list of many wrappers for accessing scientific data using R.

{geonames} Wrapper

The GeoNames geographical database covers all countries and contains over eleven million place names that are available for download free of charge.

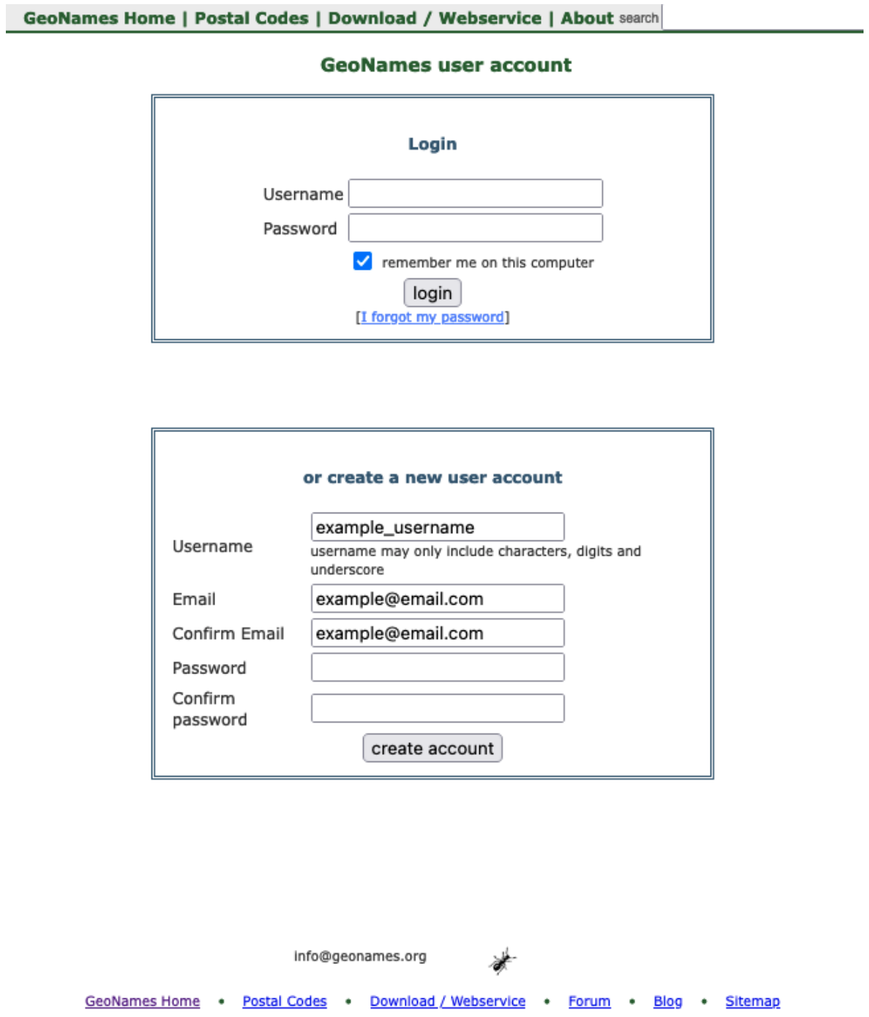
- Can access directly, but using the {geonames} is much easier.
- Purpose: Illustrate getting started with a new API.

Set up

1. Install and load {geonames} from CRAN

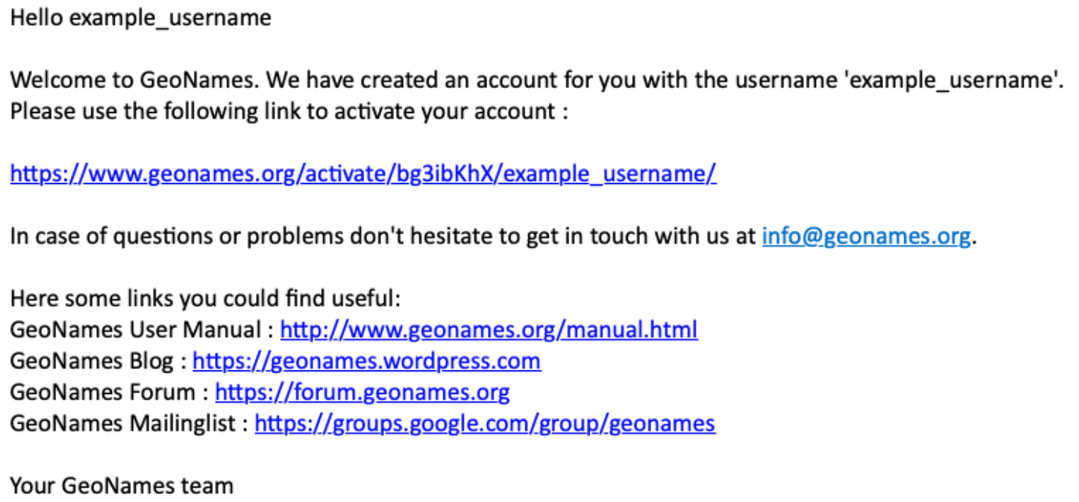
```
1 #install.packages("geonames")
2 library(geonames)
```

2. Create a user account for the GeoNames API



Set up (continued)

3. Activate the account (see activation email)



Hello example_username

Welcome to GeoNames. We have created an account for you with the username 'example_username'.
Please use the following link to activate your account :

https://www.geonames.org/activate/bg3ibKhX/example_username/

In case of questions or problems don't hesitate to get in touch with us at info@geonames.org.

Here some links you could find useful:
GeoNames User Manual : http://www.geonames.org/manual.html
GeoNames Blog : https://geonames.wordpress.com
GeoNames Forum : https://forum.geonames.org
GeoNames Mailinglist : https://groups.google.com/group/geonames

Your GeoNames team

4. Enable the free web services for your GeoNames account by logging in at this link.

Set up (final step)

5. Tell R your credentials for GeoNames.

> ⚠ **Warning**
>
> We could use the following code to tell R our credentials, but we absolutely should not.

```
options(geonamesUsername="example_username")
```

Never put credentials in your code or under version control.

Keep them secret. Keep them safe.

Storing API Credentials

Solution: Store your credentials in environment variables as part of your `.Rprofile`.

1. Open your `.Rprofile` from within R.

```
usethis::edit_r_profile()
```

2. Add your credentials to the `.Rprofile`, save and close.

```
# Add you credentials to the R profile - save and close
options(geonamesUsername="example_username")
```

3. Restart R and access your safely stored credentials.

```
# Restart R and access your safely stored credentials
getOption("geonamesUsername")
```

Gotchas: Does your `.Rprofile` end with a blank line? Did you remember to restart R?

Using {geonames}

GeoNames has a whole host of different geo-datasets.

Example: Get geo-tagged wikipedia articles within 1km of Imperial College London.

```
imperial_coords <- list(lat = 51.49876, lon = -0.1749)
search_radius_km <- 1

imperial_neighbours <- geonames::GNfindNearbyWikipedia(
  lat = imperial_coords$lat,
  lng = imperial_coords$lon,
  radius = search_radius_km,
  lang = "en",                  # English language articles
  maxRows = 500                 # maximum number of results to return
)
```

What do we get back?

```
1 str(imperial_neighbours)
```

```
'data.frame':   204 obs. of  13 variables:
 $ summary     : chr  "The Department of Mechanical Engineering is responsible for teaching and research in mechanical
engineering at "| __truncated__ "Imperial College Business School is a global business school located in London. The business
school was opened "| __truncated__ "Exhibition Road is a street in South Kensington, London which is home to several major
museums and academic est"| __truncated__ "Imperial College School of Medicine (ICSM) is the medical school of Imperial College
London in England, and one"| __truncated__ ...
 $ elevation   : chr  "20" "18" "19" "24" ...
 $ feature     : chr  "edu" "edu" "landmark" "edu" ...
 $ lng         : chr  "-0.1746" "-0.1748" "-0.17425" "-0.1757" ...
 $ distance    : chr  "0.0335" "0.0494" "0.0508" "0.0558" ...
 $ rank        : chr  "81" "91" "90" "96" ...
 $ lang        : chr  "en" "en" "en" "en" ...
 $ title       : chr  "Department of Mechanical Engineering, Imperial College London" "Imperial College Business School"
"Exhibition Road" "Imperial College School of Medicine" ...
 $ lat         : chr  "51.498524" "51.4992" "51.4989722222222" "51.4987" ...
 $ wikipediaUrl: chr  "en.wikipedia.org/wiki/Department_of_Mechanical_Engineering%2C_Imperial_College_London"
```

Sense Checking

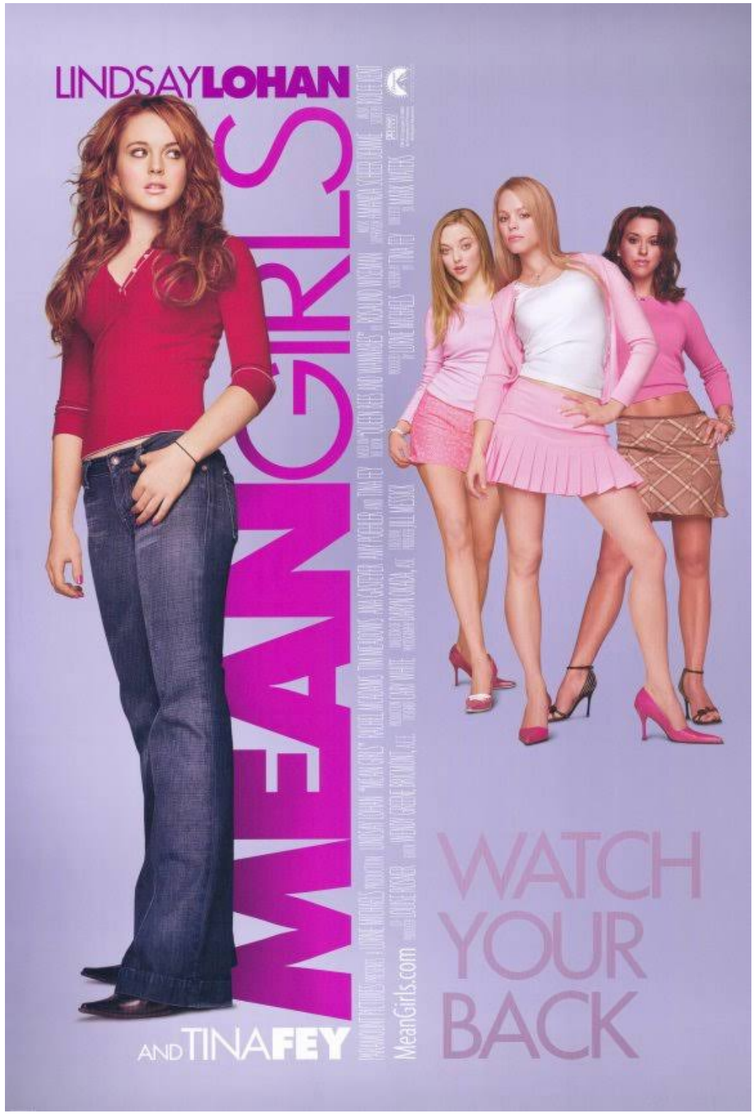
Is what we are getting back from the API sensible?

```
1 imperial_neighbours$title[1:5]
```

```
[1] "Department of Mechanical Engineering, Imperial College London"
[2] "Imperial College Business School"
[3] "Exhibition Road"
[4] "Imperial College School of Medicine"
[5] "Department of Civil and Environmental Engineering, Imperial College London"
```

What if there is no wrapper?

- No need to panic, can submit a GET request directly using `{httr}`
- Example: get Mean Girls information from OMDb, an open source version of IMDb.
- Need to get an API key, verify by email and add your API key to `.Rprofile`.



OMBb - Set Up

1. Get an API key, and verify it by clicking the email link.

2. Add this key to your `.Rprofile`, pasting in your own API key.

```
usethis::edit_r_profile()
options(OMDB_API_Key = "PASTE YOUR KEY HERE")
```

3. Restart R and safely access your API key from within your R session.

```
ombd_api_key <- getOption("OMDB_API_Key")
```

OMBb Making a Request

URL structure of OMDb API:

```
http://www.omdbapi.com/?t=<TITLE>&y=<YEAR>&plot=<LENGTH>&r=<FORMAT>&apikey=<API_KEY>
```

Function to write request URLs:

```r
#' Compose search requests for the OMBD API
#'
#' @param title String defining title to search for. Words are separated by "+".
#' @param year String defining release year to search for.
#' @param plot String defining whether "short" or "full" plot is returned.
#' @param format String defining return format. One of "json" or "xml".
#' @param api_key String defining your OMDb API key.
#'
#' @return String giving a OMBD search request URL.
#'
#' @example omdb_url("mean+girls", "2004", "short", "json", getOption(OMBD_API_Key))
omdb_url <- function(title, year, plot, format, api_key) {
  glue::glue("http://www.omdbapi.com/?t={title}&y={year}&plot={plot}&r={format}&apikey={api_key}")
}
```

Submitting a request

```
mean_girls_request <- omdb_url(
  title = "mean+girls",
  year =  "2004",
  plot = "short",
  format =  "json",
  api_key =  getOption("OMDB_API_Key"))
```

Using {httr} to construct our request and store the response we get.

```
response <- httr::GET(url = mean_girls_request)
httr::status_code(response)
```

```
[1] 200
```

Thankfully, it was a success!

Extracting the Film Data

By looking at the structure of the response we can easily extract what we want from this list.

```
1 str(httr::content(response))
```

```
List of 25
 $ Title     : chr "Mean Girls"
 $ Year      : chr "2004"
 $ Rated     : chr "PG-13"
 $ Released  : chr "30 Apr 2004"
 $ Runtime   : chr "97 min"
 $ Genre     : chr "Comedy"
 $ Director  : chr "Mark Waters"
 $ Writer    : chr "Rosalind Wiseman, Tina Fey"
 $ Actors    : chr "Lindsay Lohan, Jonathan Bennett, Rachel McAdams"
 $ Plot      : chr "Cady Heron is a hit with The Plastics, the A-list girl clique at her
new school, until she makes the mistake of"| __truncated__
```

Wrapping Up

- Learned about how computers and programs communicate.
- API keys live in your `.Rprofile` not in your code.
  - (make sure this is not under version control!)

- Wrapper > API > Scraping
  - Don’t repeat yourself, or others
  - Don’t work harder than you have to - `{omdbapi}` exists.