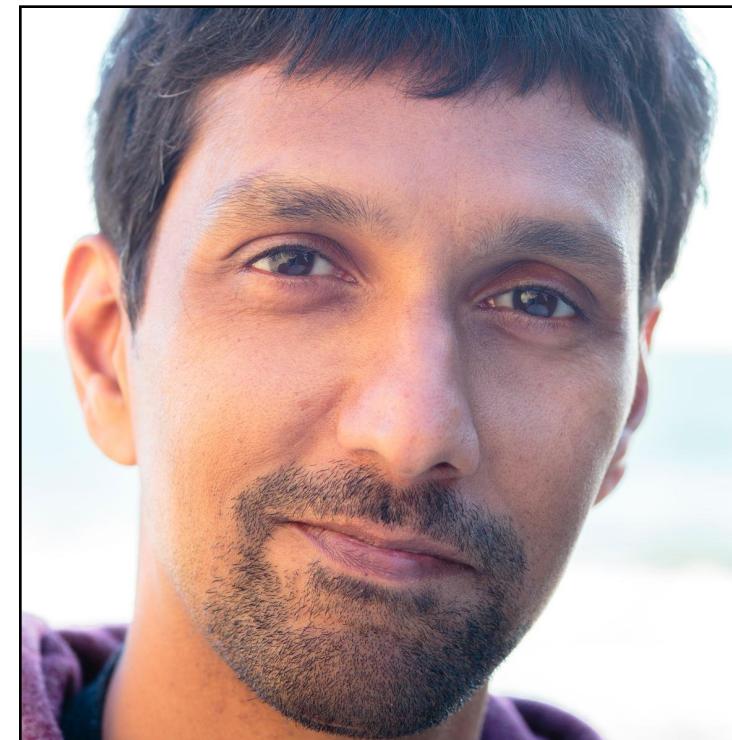


Outline

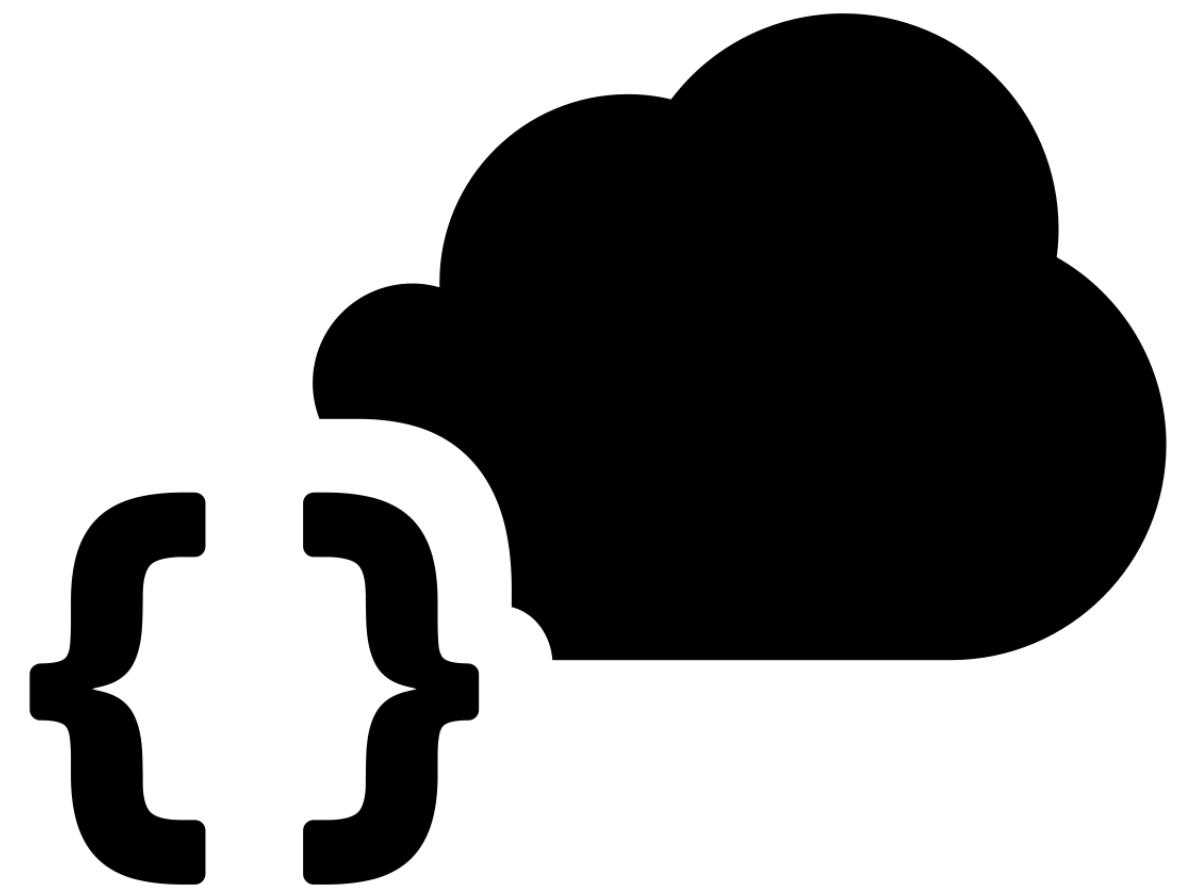
Part 1 - Collecting data from an API

Part 2 - Wrapping an API with R



Karthik Ram
Data Scientist and
Ecologist and co-founder
ROpenSci

Part II: Getting data from the web



Karthik Ram

HELLO

my name is

Karthik

@_inundata

1. Making **GET** requests
2. Extracting web content
3. Passing additional query parameters and API keys

APIs and end points



A close-up photograph of several dark-colored beer taps mounted on a wooden bar. The taps are arranged in a row, with their handles pointing towards the viewer. In the background, there is a blurred, warm-toned bokeh effect from lights, suggesting a social or celebratory atmosphere.

The Internet Movie database

Producers

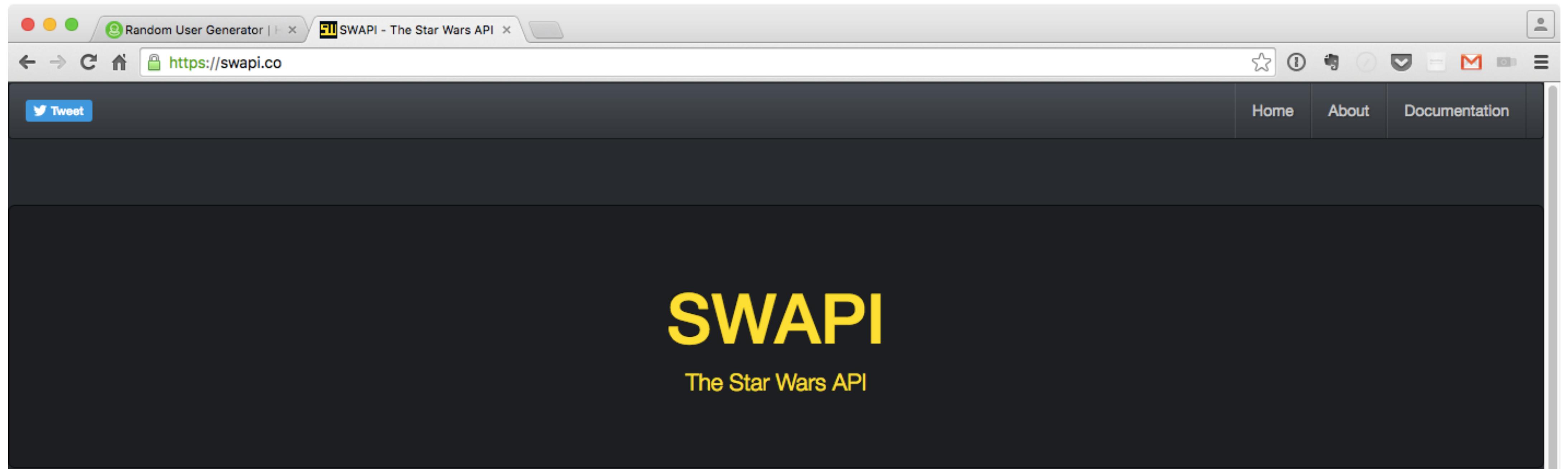
Movies

Actors



End points

Writing a GET request



Try it now!

<http://swapi.co/api/>

[people/1/](#)

request

Need a hint? try [people/1/](#) or [planets/3/](#) or [starships/9/](#)

Step 1: Write a web request

```
web_call <- GET('http://swapi.co/api/planets/1/')  
web_call
```

```
> web_call
```

```
Response [http://swapi.co/api/planets/1/]
```

```
Date: 2016-06-22 20:35
```

```
Status: 200
```

```
Content-Type: application/json
```

```
Size: 805 B
```

Fully formatted
request

```
> web_call
Response [http://swapi.co/api/planets/1/]
Date: 2016-06-22 20:35
Status: 200
Content-Type: application/json
Size: 805 B
```

Status code

Now your turn

Run a **GET** request against these end points

api.randomuser.me

api.openweathermap.org/data/2.5/forecast?id=524901

What does your result object contain?

What are the response codes?

What do they mean?



```
result <- GET('api.randomuser.me')
status_code(result)
```

```
result_2 <- GET('api.openweathermap.org/data/
2.5/forecast?id=524901')
status_code(result_2)
```

See full list of codes: <httpstatuses.com>



200
OK

2xx SUCCESS

200 OK

The request has succeeded.

The payload sent in a 200 response depends on the request method.

4xx CLIENT ERROR

403 FORBIDDEN

The server understood the request but refuses to authorize it.

A server that wishes to make public why the request has been forbidden can describe that reason in the response payload (if any).

If authentication credentials were provided in the request, the server considers them insufficient to grant access. The client SHOULD NOT automatically repeat the request with the same credentials. The client MAY repeat the request with new or different credentials. However, a request might be forbidden for reasons unrelated to the credentials.

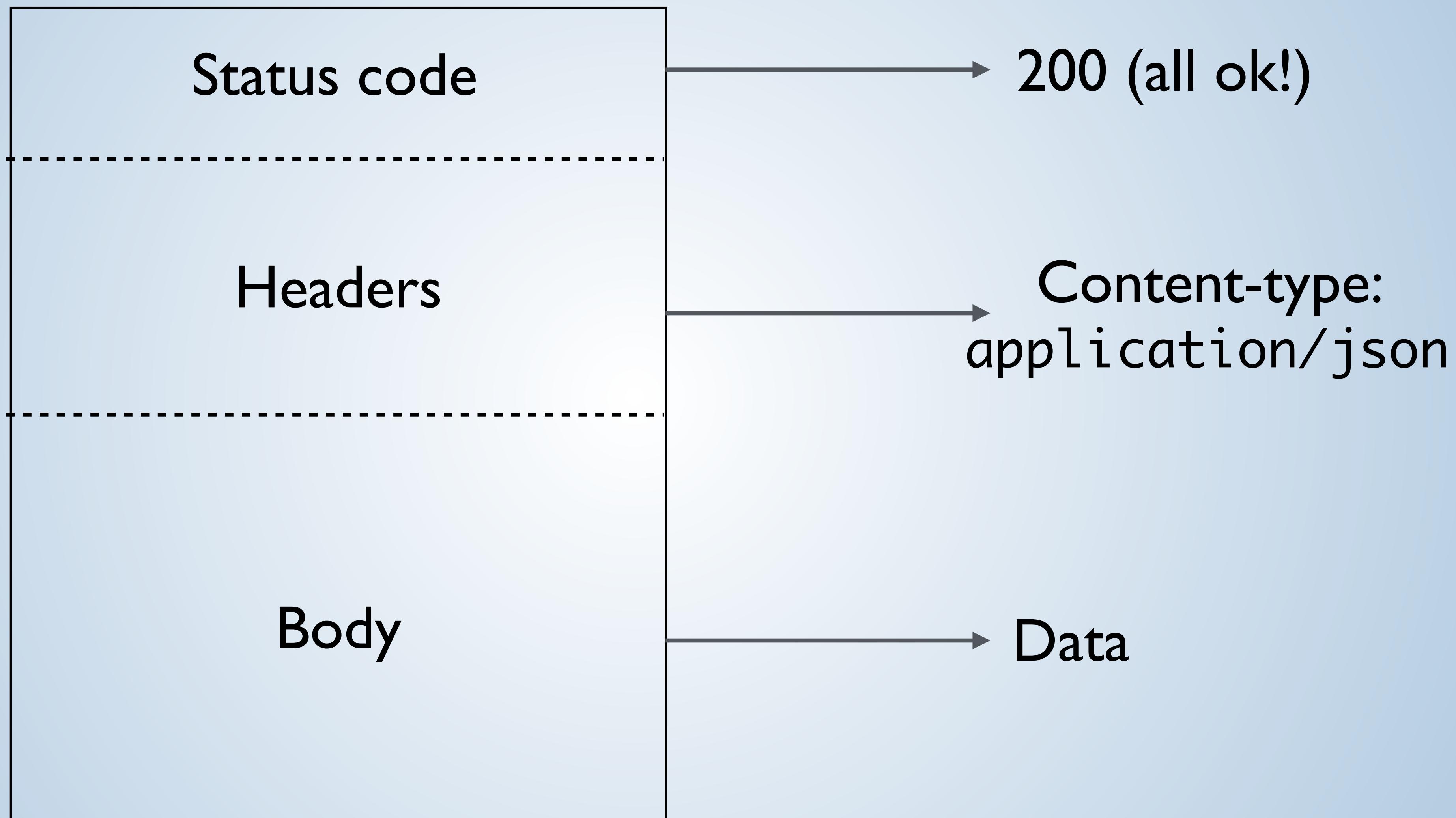


403
Forbidden

Pro tip

```
stop_for_status(request)  
warn_for_status(request)
```

2: Extracting content



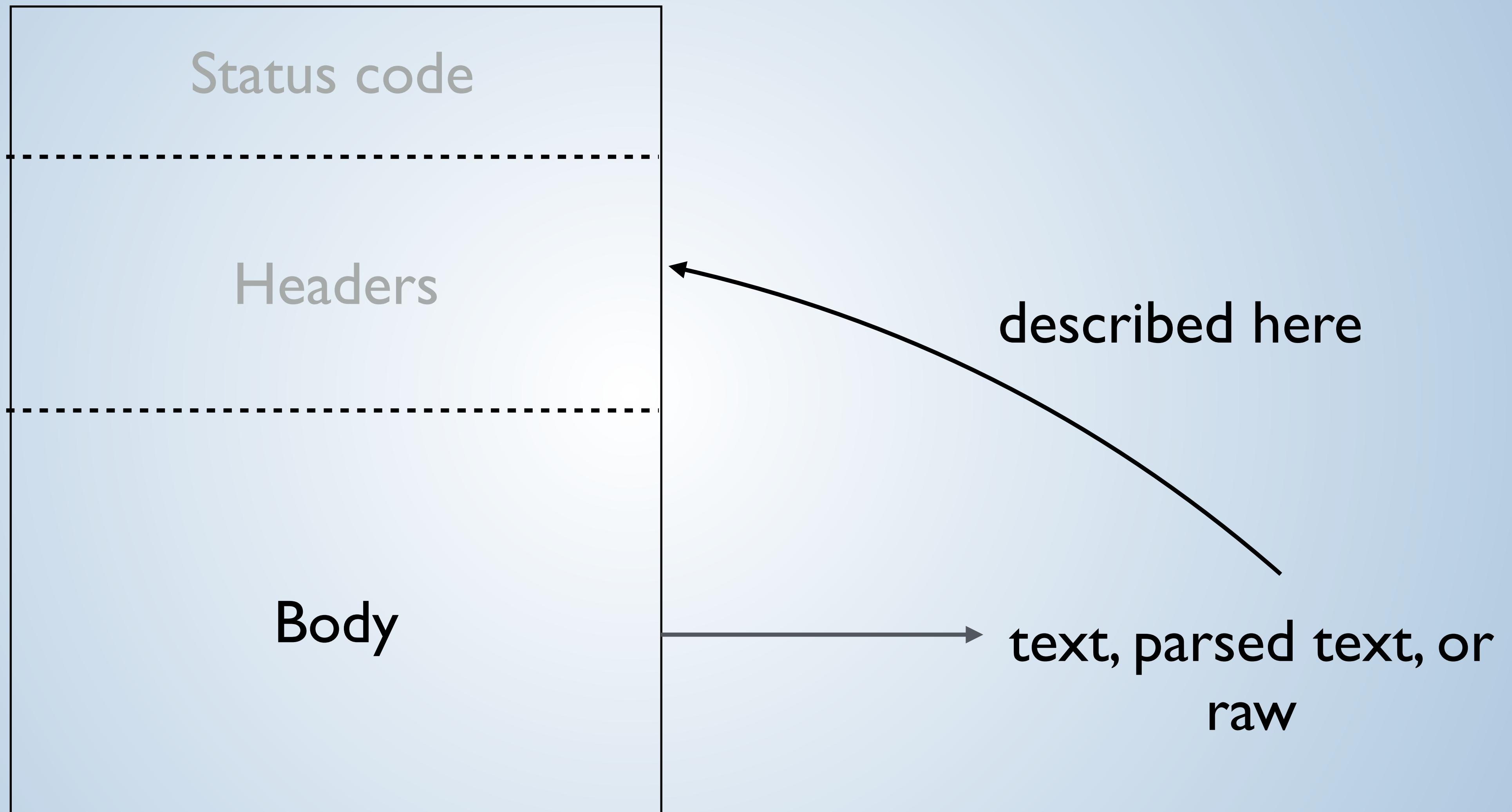
Response

Step 2: Extract content from a request

text - a *character vector*

raw - as a *raw object*

parsed - *parsed into a R object*



Text formats

Content type

text/html: `read_html`
text/xml: `read_xml`
text/csv: `read_csv`
text/tab-separated-values: `read_tsv`
application/json: `fromJSON`

Local R handler

Image formats

image/jpeg: `readJPEG`
image/png: `readPNG`

```
> call <- GET("http://google.com")
> result <- content(call, as = 'text')
> result
[1] "<!doctype html><html itemscope=\"\" itemtype=\"http://
schema.org/WebPage\" lang=\"en\"><head><meta content=\"Search the
world's information, . <truncated>
> class(result)
[1] "character"
```

api.randomuser.me

Random User Generator | X

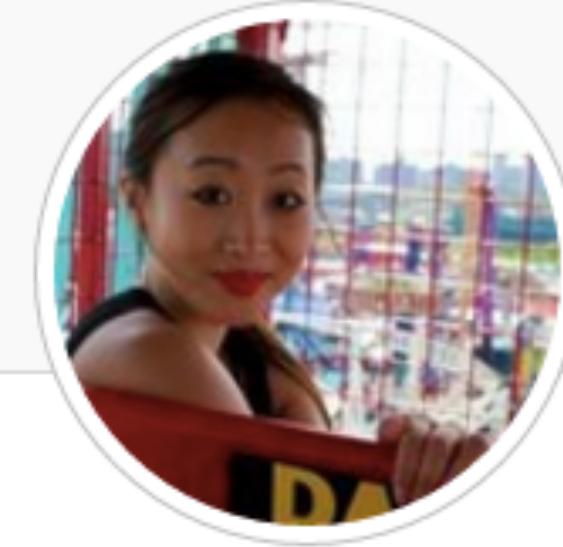
https://randomuser.me

Home User Photos Documentation Change Log Stats & Graphs Donate Copyright Notice Photoshop Extension

RANDOM USER GENERATOR

A free, [open-source API](#) for generating random user data. Like Lorem Ipsum, but for people.

[Follow us @randomapi](#)



Hi, My name is
Dawn Beck

Exercise

Process the content from the random user API

Save the result into an object called **person**

Extract the content as text and parsed.



Passing arguments

Requesting Multiple Users

Random User Generator allows you to fetch up to 5,000 generated users in one request using the **results** parameter.

`http://api.randomuser.me/?results=5000`



results: Retrieve multiple names

Specifying a gender

You can specify whether you would like to have only male or only female users generated by adding the **gender** parameter to your request. Valid values for the gender parameter are "male" or "female", or you may leave the parameter blank. Any other value will cause the service to return both male and female users.

`http://api.randomuser.me/?gender=female`



gender: specify a gender

Seeds

Seeds allow you to always generate the same set of users. For example, the seed "foobar" will always return results for [Becky Sims](#) (for version 1.0). Seeds can be any string or sequence of characters.

`http://api.randomuser.me/?seed=foobar`



seed: Set a seed

Formats

We currently offer the following data formats:

- JSON (default)

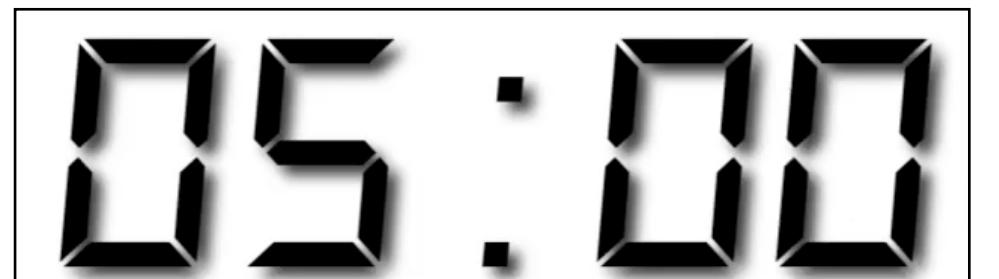
```
num_results <- 5
args <- list(results = num_results)
random_names <- GET("http://api.randomuser.me/",
query = args)
output <- content(random_names, as = 'parsed')
> length(output$results)
[1] 5
```

Exercise

Look over the api documentation and see what other parameters can be passed

Modify the example to pass a gender to the request

Can you do this with both number of results and gender?





Personal Open source Business Explore

Pricing Blog Support

Search GitHub

Personal API tokens

May 16, 2013



tclem

New Features

You can now [create your own personal API tokens](#) for use in scripts and on the command line.

Be careful, these tokens are like passwords so you should guard them carefully. The advantage to using a token over putting your password into a script is that a token can be revoked, and you can generate lots of them. [Head on over to your settings](#) to manage personal API tokens.

Personal API Access Tokens

These tokens are like passwords; guard them carefully.

4a68631afb82ba1a9f9c49892e0e3c82eaa7ef66

Create new token

Delete

Write a complete
request to a
proper API

Open Weather example

The screenshot shows the OpenWeatherMap API landing page. At the top, there's a navigation bar with links for Support Center, Weather in your city, Sign In, Sign Up, and temperature units (°C and °F). Below the navigation is the OpenWeatherMap logo and a menu with Weather, Maps, API, Price, Partners, Stations, News, and About.

The main content area is titled "Weather API". It features several sections:

- Current weather data**: Includes a list of benefits: Access current weather data for any location including over 200,000 cities; Current weather is frequently updated based on global models and data from more than 40,000 weather stations; Data is available in JSON, XML, or HTML format; Available for Free and all other paid accounts.
- 5 day / 3 hour forecast**: Includes a list of benefits: 5 day forecast is available at any location or city; 5 day forecast includes weather data every 3 hours; Forecast is available in JSON, XML, or HTML format; Available for Free and all other paid accounts.
- 16 day / daily forecast**: Includes a list of benefits: 16 day forecast is available at any location or city; 16 day forecasts includes daily weather; Forecast is available in JSON, XML, or HTML format; Available for Developer, Professional and Enterprise accounts.
- Historical data**: Includes a list of benefits: Through our API we provide city historical data; Current UV index (Clear Sky) and historical UV data.
- UV Index**: Includes a list of benefits: Current UV index (Clear Sky) and historical UV data.
- Weather map layers**: Includes a list of benefits: Weather maps include precipitation, temperature, and humidity data.

Exercise

- Get a API key from openweathermap.org/api
- Write a request for current weather data (<http://openweathermap.org/current>) for zip code 94708,us
- Pass
- Format results into a data.frame (for help with this, see: bit.ly/flatten_json)

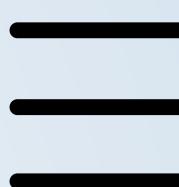
Tip: Pass the API key as:

```
APPID <- '<your_api_key>'
```



Recap

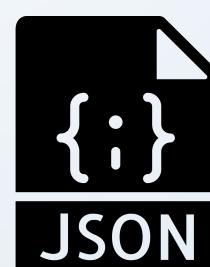
Make a **GET** request

Pass additional **arguments** to a end point 

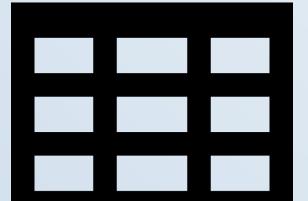
Authenticate with a **API key** or **oauth2 token**

Check for **status**

Then process content as



Format results



Thanks!