Web scraping 1
Advanced Crime Analysis UCL

Bennett Kleinberg

14 Jan 2019

Getting data from the Internet

Webscraping 1

Today

- Types of webscraping
- Using APIs: Twitter + YouTube
- Crime data ‘wrappers’
- “Real” webscraping: basics of a webpage

What is webscraping anyway?

The game changer!

- direct broadcasting of ideas
- “unfiltered” and “uncensored” (?)
- location-enabled
- and: *en masse*

Crime Prediction Using Twitter Sentiment and Weather

Xinyu Chen, Youngwoon Cho, and Suk young Jang
University of Virginia, xc7xn, yc5ac, sj2rh@virginia.edu

Using Twitter for Next-Place Prediction, with an Application to Crime Prediction

Mingjun Wang and Matthew S. Gerber
Department of Systems and Information Engineering
University of Virginia
Charlottesville, Virginia 22903
Email: {mw4cc, msg8u}@virginia.edu

Relationships Between Crime and Twitter Activity Around Stadiums

Alina Ristea, Chad Langford
Department of Geoinformatics, Doctoral College
University of Salzburg
Salzburg, Austria
mihaela-alina.ristea@sbg.ac.at; chad.langford@sbg.ac.at

Michael Leitner
Department of Geography and Anthropology
Louisiana State University
Baton Rouge, USA
mleitne@lsu.edu

Public Perceptions on Organised Crime, Mafia, and Terrorism: A Big Data Analysis based on Twitter and Google Trends

Panos Kostakos[1]
University of Oulu, Finland

Articles

The impact of using social media data in crime rate calculations: shifting hot spots and changing spatial patterns

Nick Malleson & Martin A. Andresen

Pages 112-121 | Received 28 Nov 2013, Accepted 13 Mar 2014, Published online: 10 Apr 2014

Language Usage on Twitter Predicts Crime Rates

Abdulaziz Almehmadi
Sensor Networks and Cellular Systems Research Center (SNCS)
Faculty of Computing and IT
University of Tabuk
Tabuk, Saudi Arabia
aalmehmadi@ut.edu.sa

Zeinab Joudaki
University of Ontario Institute of Technology
Oshawa, Canada
zeinab.joudaki@uoit.ca

Roozbeh Jalali
University of Ontario Institute of Technology
Oshawa, Canada
roozbeh.Jalali@uoit.ca

Types of webscraping

	Data shared	**Data not shared**
Ready-made table	Download	*closed source*
Not ready-made	API	Real webscraping

Application programming interfaces (APIs)

API: basics

Goal:

- help developers interact with the platform
- facilitates interaction in an automatable manner
- analogous to the GUI
- part of it: enabling data access
- contains precise documentation

What an API does not do:

- give you all the data
- be free forever
- give you full control

There’ no free lunch!

Using an API

Core elements of an API:

- GET requests
- POST requests

Implementable in different ways…

Using an API

Classes of APIs:

1. Web APIs
   - send requests through the browser
   - add URL parameters

     ```
     https://data.police.uk/api/crimes-at-locati
     ```
2. Libraries/packages for APIs
   - depending on the API: python, js, php, ruby
   - = frameworks to access the API
   - = methods implemented in different languages
3. API wrappers
   - R packages that use the API

Using an API

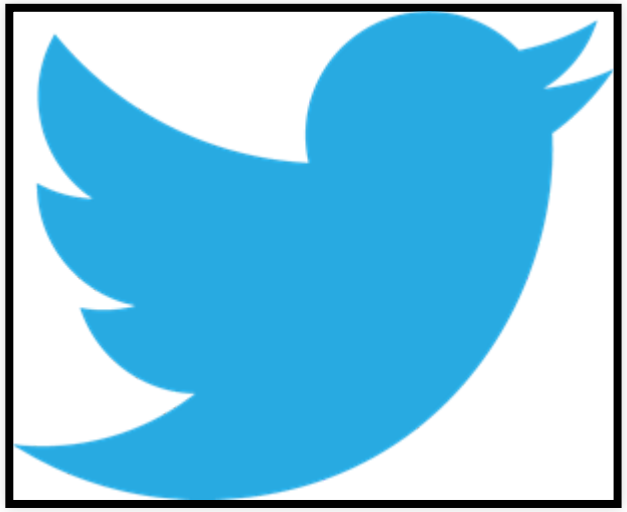
Identify API capabilities

Official API docs Twitter

Useful websites that have an API

- Twitter
- YouTube
- Instagram
- Facebook
- Reddit

Case 1: Twitter's API



Getting access

Basic steps

1. Twitter account
2. Apply for a developer's account
3. Create project
4. Obtain access credentials

Tutorial here

The `rtweet` package

```
library(twitteR)
```

- Documentation: R Docs
- Demos: [basic first steps]https://towardsdatascience.com/access-data-from-twitter-api-using-r-and-or-python-b8ac342d3efe) + next week's tutorial

Note: check out the newer rtweet package.

Authenitication through R

```
my_consumer_key = "5tc2oAVLyO8DkCKW1k8ny2H6e"
my_consumer_secret = "qEQYGX6IKs6NiSUsENprBZlOOdoM9lWkoIht3p1sVnAMraQpq2
my_access_token = "858383409986625537-Fy9Ai5eFyf23VZHguRJEdXqell6Q8Jl"
my_access_secret = "nT5Z0eQjAvBdf2ZjxMgiaoRb7hiHVxB8jYh7lT74CW1Um"

setup_twitter_oauth(consumer_key = my_consumer_key
                    , consumer_secret = my_consumer_secret
                    , access_token = my_access_token
                    , access_secret = my_access_secret
                    )
```

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

How to search?

Remember the problem-solving approach?

Start with the problem…

…then do what's necessary to solve it.

The research question determines the method

How to search?

Depends on the problem:

- Tweets in a certain time frame (e.g. December 2018)
- Tweets with a certain key-word (e.g. “#metoo”)
- Tweets by a certain author (e.g. Elon Musk)
- Tweets in a certain location (e.g. London)
- Tweets in a certain language
- Combined search queries

API possibilities

Always look at two sources:

1. The original API (Twitter's API docs)
2. The API interface (`twitteR R package`)

Note: mostly original API options > API interface options.

Tweets by date

Search:

- tweets since December 2018
- with #metoo

```
metoo_tweets_december = searchTwitter(searchString = '#metoo'
                                      , n = 10
                                      , since = '2018-12-01'
                                      )
metoo_tweets_december
```

```
## [[1]]
## [1] "zee45427557: #RajkumarHirani @aamir_khan #AamirKhan #aamir #3idi
##
## [[2]]
## [1] "metoozoo: #MeToo Merch - YellowMaps Beaver Island MI topo map, 1
##
## [[3]]
## [1] "Mirbia3: RT @la_patilla: Los aspirantes demócratas a la Casa Bla
##
## [[4]]
## [1] "gulfkannadiga: RT @timesofindia: #MeToo movement: Filmmaker #Raj
##
## [[5]]
```

```
## [1] "WeForNews: Rajkumar Hirani accused of sexual assault during maki
##
## [[6]]
## [1] "worldwidetoto10: RT @12ji10pun: 怒りが収まらない\U0001f4a2\n海外では
```

Tweets by date

Display as dataframe with meta information:

```
twListToDF(metoo_tweets_december)
```

```
##
## 1
## 2
## 3
## 4
## 5
## 6  RT @12ji10pun: 怒りが収まらない\U0001f4a2\n海外では問題になりそうな #松本人
## 7
## 8
## 9
## 10
##    favorited favoriteCount    replyToSN             created truncated
## 1      FALSE             0         <NA> 2019-01-13 11:38:59     FALSE
## 2      FALSE             0         <NA> 2019-01-13 11:38:51      TRUE
## 3      FALSE             0         <NA> 2019-01-13 11:38:47     FALSE
## 4      FALSE             0         <NA> 2019-01-13 11:38:44     FALSE
## 5      FALSE             0         <NA> 2019-01-13 11:38:33      TRUE
## 6      FALSE             0         <NA> 2019-01-13 11:38:24     FALSE
```

Tweets by date

Example: Popular crime tweet in 2019?

```
crime_tweets_2019 = searchTwitter(searchString = 'crime'
                                  , n = 1000
                                  , since = '2019-01-01'
                                  , resultType = 'popular'
                                  )
```

```
## Warning in doRppAPICall("search/tweets", n, params = params,
## retryOnRateLimit = retryOnRateLimit, : 1000 tweets were requested but
## API can only return 63
```

```
df.crime_tweets_2019 = twListToDF(crime_tweets_2019)

df.crime_tweets_2019[order(df.crime_tweets_2019$created, decreasing = F),
```

```
## [1] "#NewsUpdate ออกประกาศด่วน! หยุดเดินเรือข้ามเกาะสมุย ชาวบ้านแห่กักตุนอาหาร พ
## [2] "Reform-minded prosecutors can repair our broken #CriminalJustice
## [3] "Lembrando sempre que não gostar de alguém, além de não ser crime,
```

Tweets by keyword

Search:

- “fake news” tweets
- since the start of the year

```
fakenews_tweets_2019 = searchTwitter(searchString = 'fake+news'
                                     , n = 1000
                                     , since = '2019-01-01'
                                     , resultType = 'popular'
                                     )
```

```
## Warning in doRppAPICall("search/tweets", n, params = params,
## retryOnRateLimit = retryOnRateLimit, : 1000 tweets were requested but
## API can only return 58
```

```
df.fakenews_tweets_2019 = twListToDF(fakenews_tweets_2019)
df.fakenews_tweets_2019[order(df.fakenews_tweets_2019$retweetCount, decre
```

```
##
## 21 The Mainstream Media has NEVER been more dishonest than it is now.
## 22 ....The Fake News Media in our Country is the real Opposition Party
## 31 With all of the success that our Country is having, including the
## 1  The Fake News Media keeps saying we haven't built any NEW WALL. Bel
## 30  The story in the New York Times regarding Jim Webb being considere
##    favorited favoriteCount replyToSN             created truncated
## 21     FALSE        125233      <NA> 2019-01-10 03:43:13      TRUE
## 22     FALSE        130273      <NA> 2019-01-07 13:31:00      TRUE
## 31     FALSE        135224      <NA> 2019-01-07 12:56:19      TRUE
## 1      FALSE        101249      <NA> 2019-01-11 17:50:04      TRUE
## 30     FALSE        105462      <NA> 2019-01-04 21:45:27      TRUE
##    replyToSID                  id replyToUID
## 21       <NA> 1083207607412760576       <NA>
## 22       <NA> 1082268365081767936       <NA>
## 31       <NA> 1082259636227620865       <NA>
## 1        <NA> 1083783112973320192       <NA>
## 30       <NA> 1081305634115674112       <NA>
```

Tweets by keyword

Search:

- knife crime
- yesterday

```
knife_crime_yesterday = searchTwitter(searchString = 'knife+crime'
                                      , since = '2019-01-08'
                                      )
```

```
knife_crime_yesterday[1:10]
```

```
## [[1]]
## [1] "IsmailRahiman: RT @DailyMailUK: Police are armed with metal detec
##
## [[2]]
## [1] "johnbrissenden: RT @natalieisonline: There are a few alarming thi
##
## [[3]]
## [1] "Bob4719: RT @JuanDiablo4d: @MayorofLondon @BBCSPLondon @Jo_Coburn
##
## [[4]]
## [1] "ot7_trash: RT @SeemaChandwani: The child is dead. Murdered. \n\n(
##
## [[5]]
## [1] "amitysv: RT @incorrectbucko: bucky, singing to himself: coming ou
##
## [[6]]
## [1] "steer266: If May stays at No 10, I'm starting Project Hope, the p
##
```

Tweets by combined keywords

Example: Tweets about knife killings in London in 2019

```
knife_killings_london = searchTwitter(searchString = 'knife+killing+londo
                                      , since = '2019-01-01'
                                      )
knife_killings_london[1:5]
```

```
## [[1]]
## [1] "Isabel_Cavazos: RT @Condor_Law: Khan's London: 14-Year-Old 'Butch
##
## [[2]]
## [1] "Brasssneck: @I_R_DyLaNe @joshwoolcott Last week a man was killed
##
## [[3]]
## [1] "sterling_poetry: RT @Condor_Law: Khan's London: 14-Year-Old 'Butc
##
## [[4]]
## [1] "WantBigHammer: RT @Condor_Law: Khan's London: 14-Year-Old 'Butche
##
## [[5]]
## [1] "joeyd541: RT @Condor_Law: Khan's London: 14-Year-Old 'Butchered
```

Tweets by combined keywords

Zoom in further...

Example: Tweets about murder on Surrey train (4th of Jan)

```
surrey_train_killings = searchTwitter(searchString = 'surrey+murder+knife
                                      , since = '2019-01-01'
                                      , n = 100
                                      )
```

```
## Warning in doRppAPICall("search/tweets", n, params = params,
## retryOnRateLimit = retryOnRateLimit, : 100 tweets were requested but t
## API can only return 82
```

```
df.surrey_train_killings = twListToDF(surrey_train_killings)

#how many are retweets?
table(df.surrey_train_killings$isRetweet)
```

```
##
## FALSE  TRUE
##    16    66
```

Tweets by author

Search:

- tweets in 2019
- by the Mayor of London

```
mol_2019 = searchTwitter(searchString = 'from:MayorofLondon'
                         , since = '2019-01-01'
                         , n = 100
                         )
```

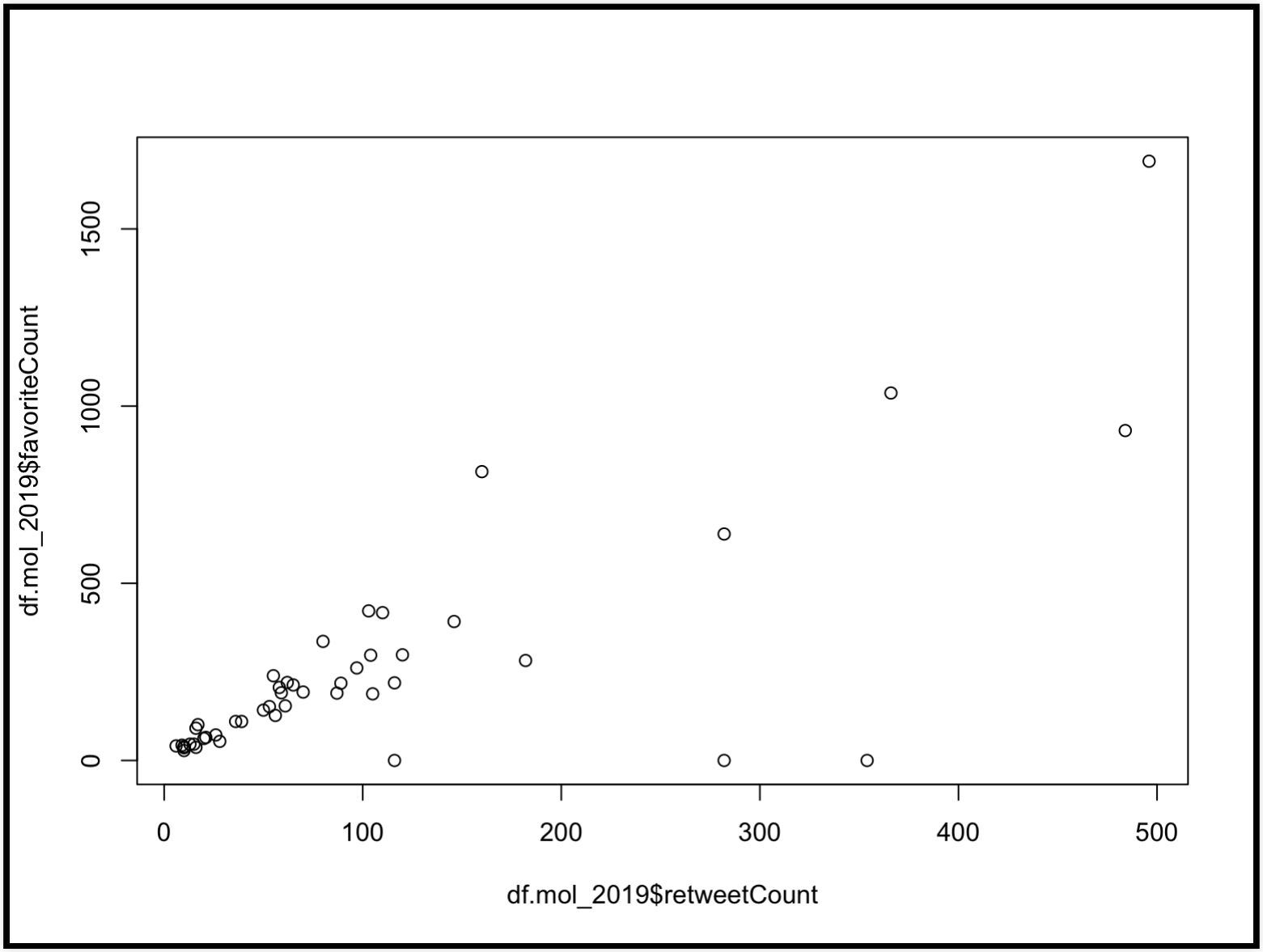
```
## Warning in doRppAPICall("search/tweets", n, params = params,
## retryOnRateLimit = retryOnRateLimit, : 100 tweets were requested but t
## API can only return 46
```

```
df.mol_2019 = twListToDF(mol_2019)
head(df.mol_2019)
```

```
##
## 1 Tune in to @BBCSPLondon from 11am where I'll be speaking live with (
## 2                          I'll be on @BBC5Live with @JPonpolitics to
## 3 No one should be sleeping rough tonight. We're doing everything we c
## 4 Can you guess how many Hopper journeys are made every day? Join Tall
## 5 Violent crime has no place in our city, and I'm determined to do eve
## 6           What an incredible celebration of Waltham Forest this ever
##   favorited favoriteCount replyToSN             created truncated
## 1     FALSE             41      <NA> 2019-01-13 10:49:23      TRUE
## 2     FALSE             36      <NA> 2019-01-13 09:01:32     FALSE
## 3     FALSE            152      <NA> 2019-01-12 17:14:08      TRUE
## 4     FALSE             43      <NA> 2019-01-12 14:06:36      TRUE
## 5     FALSE            239      <NA> 2019-01-12 10:10:16      TRUE
## 6     FALSE            191      <NA> 2019-01-11 20:14:01      TRUE
##   replyToSID                  id replyToUID
## 1       <NA> 1084402019174096896       <NA>
## 2       <NA> 1084374879321894915       <NA>
## 3       <NA> 1084136458137583618       <NA>
```

Tweets by author

```
plot(df.mol_2019$retweetCount, df.mol_2019$favoriteCount)
```


df.mol_2019$favoriteCount
0
500
1000
1500
0
100
200
300
400
500
df.mol_2019$retweetCount

The "outlier"?

```
df.mol_2019[which.max(df.mol_2019$favoriteCount), 'text']
```

```
## [1] "The PM must finally rule out a no deal Brexit once and for all t
```

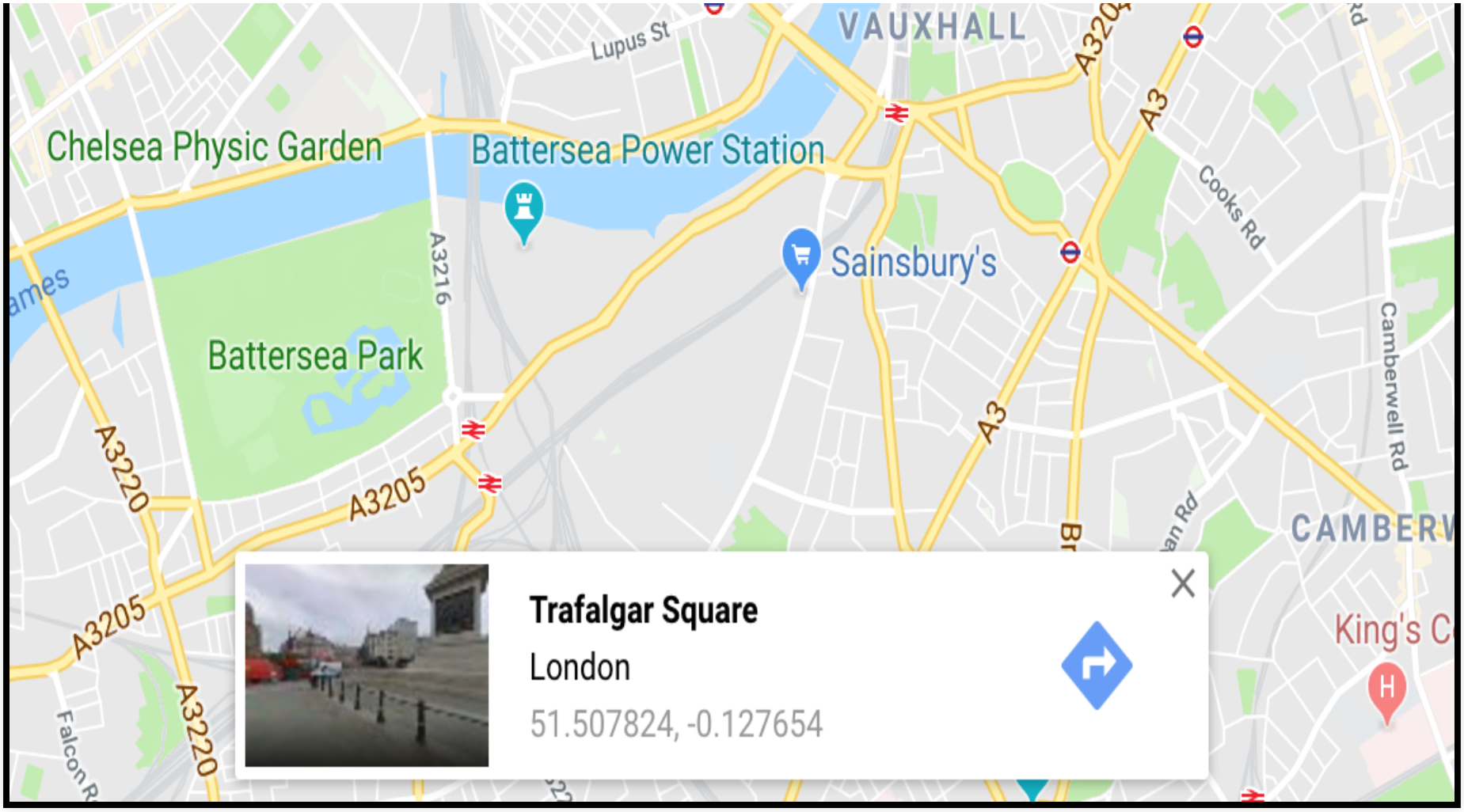
Tweets by location

Search:

- most retweeted tweets –> **sort by `retweetCount`**
- yesterday –> **set date to 8 Jan 2019**
- in London –> **???**
- about: …

Solution: geo coordinates.

Tweets by location



(51.507824, -0.127654)

We also add a radius around that point location.

Tweets by location

```
tweets_in_london = searchTwitter(searchString = ' '
                         , since = '2019-01-01'
                         , n = 100
                         , geocode='51.507824,-0.127654,15km'
                         )
head(tweets_in_london)
```

```
## [[1]]
## [1] "niamhethandarcy: RT @matthewsyed: British tennis was amateurish,
##
## [[2]]
## [1] "gtrlucie: RT @aveirjapan: https://t.co/VI1jTcLU2t"
##
## [[3]]
## [1] "preshn9: RT @donnyc1975: @JolyonMaugham @damocrat With one party
##
## [[4]]
## [1] "flawlessftmalik: RT @esnysophie: LIAM HAS BEEN NOMINATED FOR A BI
##
## [[5]]
## [1] "FrauVonHerzL: RT @streetartmagic: Zabou - French Street Artist -
##
## [[6]]
## [1] "Ege2503: RT @NZ_Bazou: T'es en couple? — Célibataire depuis 1956
```

Tweets by language

Common problem:

- you search for a keyword
- but it's in multiple languages

```
?searchTwitter
```

`lang`:

If not NULL, restricts tweets to the given language, given by an ISO 639-1 code

Tweets by language

ISO 639-1 language codes:

https://www.loc.gov/standards/iso639-2/php/code_list.php

```
searchTwitter(searchString = '#metoo'
                           , since = '2019-01-01'
                           , n = 5
                           , lang = 'en'
                           )
```

```
## [[1]]
## [1] "LeadingWPassion: How to Tap Into Your Greatest Leadership Potenti
##
## [[2]]
## [1] "Tudumonstu: RT @RoshanKrRai: So #RajkumarHirani , the biggest ima
##
## [[3]]
## [1] "metoozoo: #MeToo Merch - YellowMaps Beaver Island MI topo map, 1
##
## [[4]]
## [1] "gulfkannadiga: RT @timesofindia: #MeToo movement: Filmmaker #Raj
##
## [[5]]
## [1] "WeForNews: Rajkumar Hirani accused of sexual assault during maki
```

Tweets by language

```
searchTwitter(searchString = '#metoo'
                           , since = '2019-01-01'
                           , n = 5
                           , lang = 'es'
                           )
```

```
## [[1]]
## [1] "Mirbia3: RT @la_patilla: Los aspirantes demócratas a la Casa Bla
##
## [[2]]
## [1] "77diegoleon: RT @NunkMasKKs: Las del hashtag #JuntoAActricesArge
##
## [[3]]
## [1] "Marlyndreams: RT @NunkMasKKs: Las del hashtag #JuntoAActricesArg
##
## [[4]]
## [1] "lucaluna2015: RT @NunkMasKKs: Las del hashtag #JuntoAActricesArg
##
## [[5]]
## [1] "Remanso4: RT @NunkMasKKs: Las Griseldas Sicilianis de la vida di
```

Combined search queries

Example: Burglaries since Christmas?

```
combined_query_1 = searchTwitter(searchString = 'burgled'
                , since = '2018-12-24'
                #, until = '2019-01-07'
                , n = 100
                , lang = 'en'
                )
head(combined_query_1)
```

```
## [[1]]
## [1] "RobertCongreve: RT @K9Finn: Riding school charity for the disable
##
## [[2]]
## [1] "jwoodford74: RT @paul_samouelle: Anybody seen this piece of scum
##
## [[3]]
## [1] "RobertAlanWint2: RT @K9Finn: Riding school charity for the disabl
##
## [[4]]
## [1] "PhoebeBellend: can’t believe my gaffs been burgled haha what a da
##
## [[5]]
## [1] "RobertAlanWint2: RT @neenaw: My mother (72) was burgled last weel
##
```

```
## [[6]]
```

Combined search queries

Example: Reactions to the Soubry issue in two cities?

```
soubry_london = searchTwitter(searchString = 'soubry'
              , since = '2019-01-01'
              , n = 100
              , lang = 'en'
              , geocode='51.507824,-0.127654,15km'
              )
head(soubry_london, 2)
```

```
## [[1]]
## [1] "baneman21: RT @Frankhaviland: If calling Anna Soubry a ‘Nazi’ is
##
## [[2]]
## [1] "journopoly: @Anna_Soubry @carolecadwalla The citizens see past t
```

```
soubry_manchester = searchTwitter(searchString = 'soubry'
              , since = '2019-01-01'
              , n = 100
              , lang = 'en'
              , geocode='53.480874,-2.242588,15km'
              )
head(soubry_manchester, 2)
```

```
## [[1]]
## [1] "jameslynn38: RT @StephenWadswor2: Absolutely shameful. Crocodile
##
## [[2]]
## [1] "JamesIsherwoo15: BBC News - Brexit failure a catastrophic breach
```

Combined search queries

Your turn: what does this search do?

```
searchTwitter(searchString = '@Anna_Soubry + nazi'
              , since = '2019-01-01'
              , n = 5
              , lang = 'en'
              , resultType = 'popular'
              )
```

Combined search queries

Your turn: what does this search do?

```
searchTwitter(searchString = '@Anna_Soubry + nazi'
              , since = '2019-01-01'
              , n = 5
              , lang = 'en'
              , resultType = 'popular'
              )
```

```
## [[1]]
## [1] "SuzanneEvans1: And after screaming blue murder when @Anna_Soubry
##
## [[2]]
## [1] "BBCNormanS: Is this what its come to ...? @Anna_Soubry faces \"na
##
## [[3]]
## [1] "BBCPolitics: \"This is astonishing. This is what has happened to
##
## [[4]]
## [1] "AngelaRayner: What has our Country come to when watching @BBCNews
##
## [[5]]
## [1] "davidkurten: I agree with @Anna_Soubry - it's wrong to call somec
```

Get Twitter trends

```
?getTrends
```

```
getTrends(woeid, exclude=NULL, ...)
```

woeid: A numerical identification code describing a location, a Yahoo! Where On Earth ID

–> Cross-reference WOEID against search string/address here

getTrends...

San Francisco, California –> `woied = 2487956`

```
trends_sf = getTrends(woeid = 2487956)

head(trends_sf)
```

```
##          name                                                  url
## 1     Cowboys           http://twitter.com/search?q=Cowboys
## 2   #dalvslar       http://twitter.com/search?q=%23dalvslar
## 3 CJ Anderson http://twitter.com/search?q=%22CJ+Anderson%22
## 4        Rams              http://twitter.com/search?q=Rams
## 5    #INDvsKC        http://twitter.com/search?q=%23INDvsKC
## 6       Colts             http://twitter.com/search?q=Colts
##                query   woeid
## 1          Cowboys 2487956
## 2       %23dalvslar 2487956
## 3 %22CJ+Anderson%22 2487956
## 4             Rams 2487956
## 5       %23INDvsKC 2487956
## 6            Colts 2487956
```

Additional stuff...

Twitter API/twitteR package

- URLs
- followers
- user info
- ...

Twitter Search API vs Twitter Stream API

- query quality
- result quantity

Problems with Twitter/YouTube data?

Some issues:

- Sample representativeness
- Location accuracy
- Location availability
- Sampling through API

Case 2: Youtube's API

Getting access

Basic steps

1. Google account
2. Login to Google Developers Console
3. Create project/app
4. Obtain access credentials

Tutorial here

The `tuber` package

```
library(tuber)
```

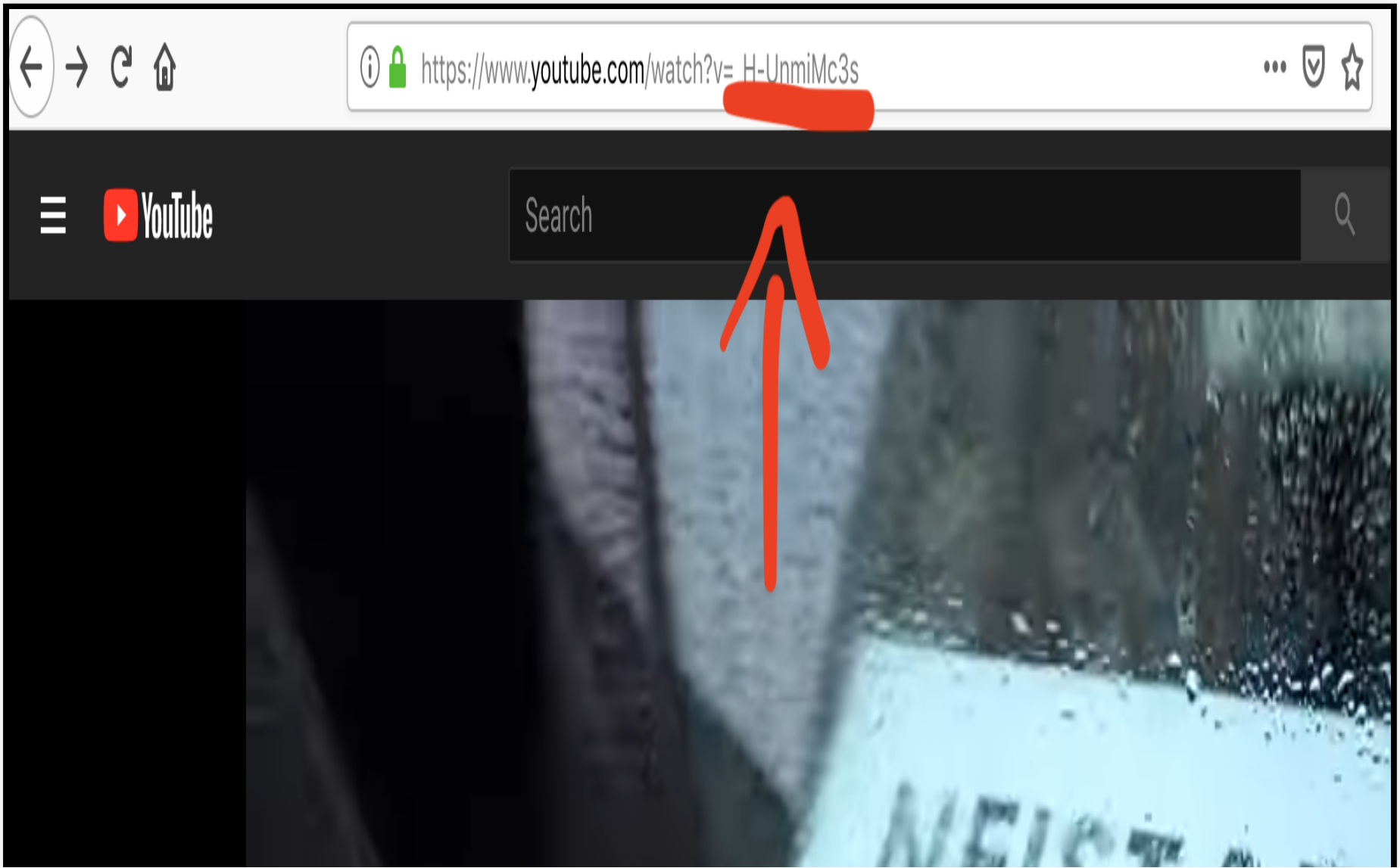
- Documentation: pdf
- Source code: GitHub
- Demos: developer's tutorial, basic first steps + next week's tutorial

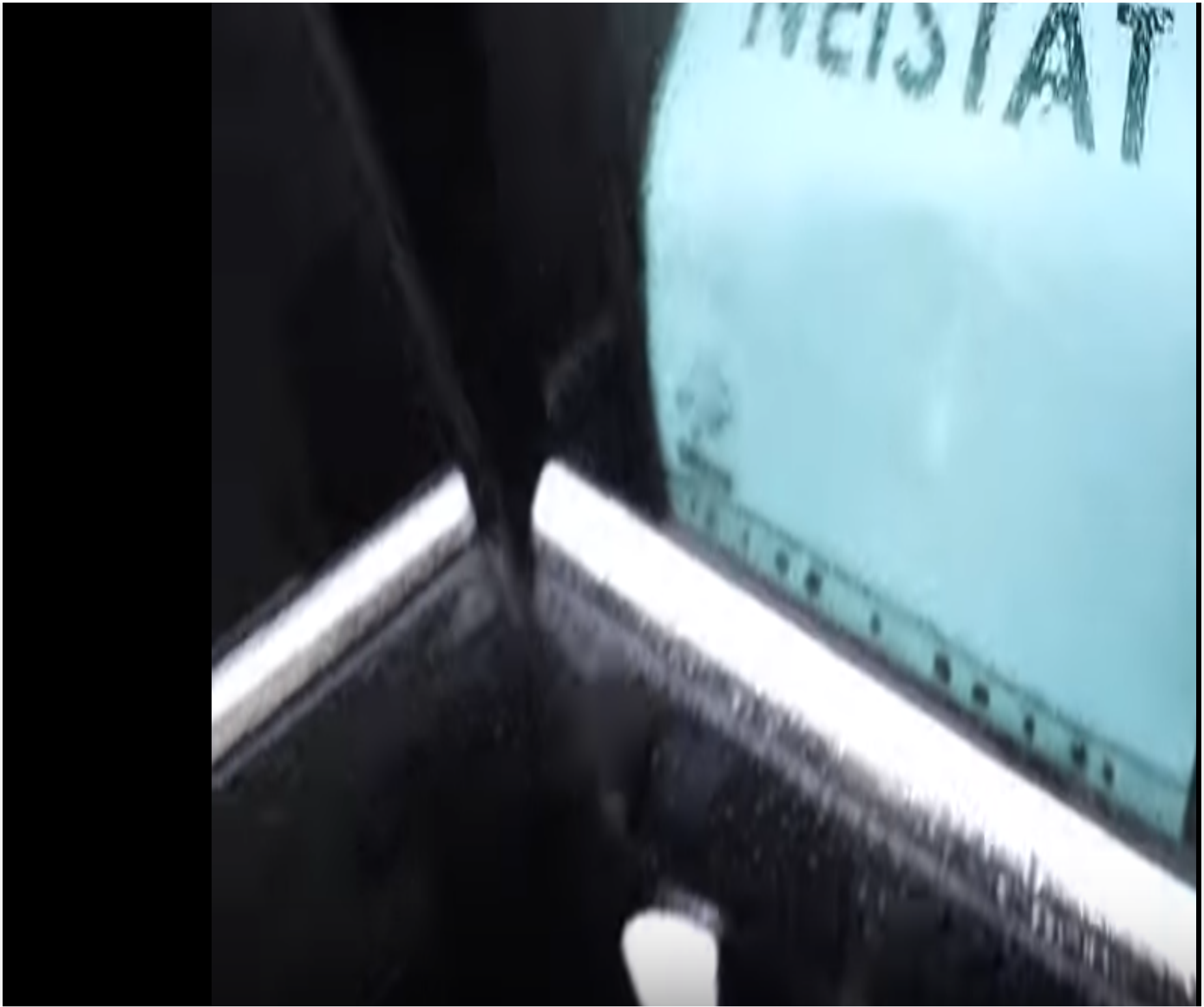
Authentication

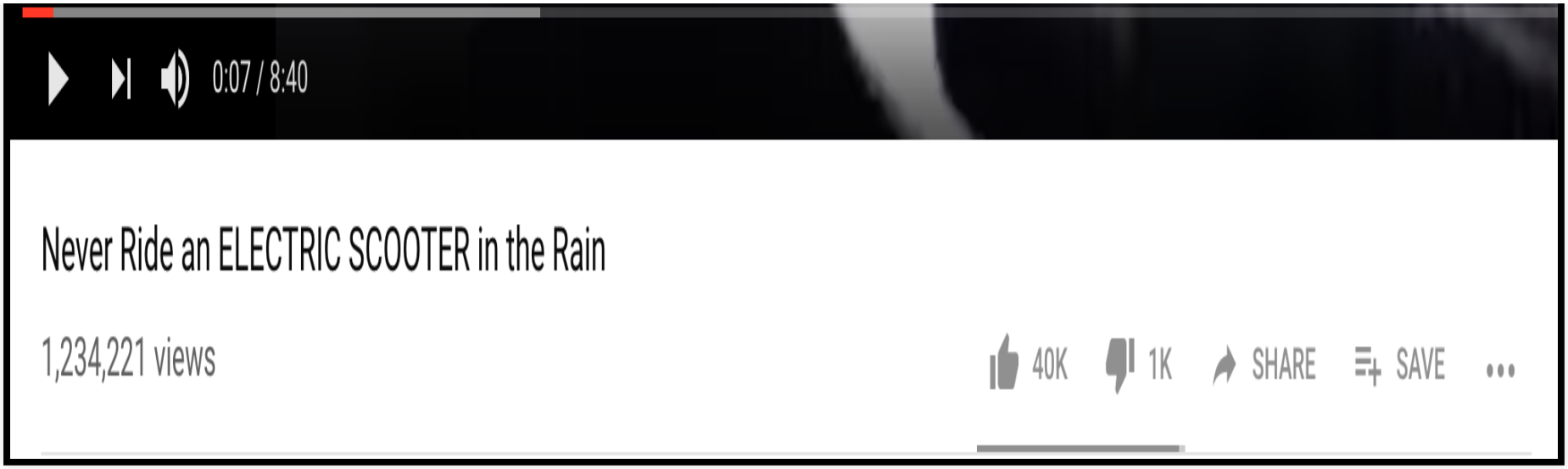
```
client_secret = 'rwHJJDPf_xdvIWmQ4TL00HKz'
client_id = '625618111946-mf44nomvi5m9ot668b59k7koq122jmaa.apps.googleuse

yt_oauth(app_id = client_id, app_secret = client_secret, token='')
```

Meta data per video

First step: get the video ID.






0:07 / 8:40
Never Ride an ELECTRIC SCOOTER in the Rain
1,234,221 views
40K
1K
SHARE
SAVE

Meta data per video

Get the “meta” stats for this video:

```
video_identifier = '_H-UnmiMc3s'

video_stats = get_stats(video_id = video_identifier)
as.data.frame(video_stats)
```

```
##            id viewCount likeCount dislikeCount favoriteCount commentCo
## 1 _H-UnmiMc3s   1248936     41311          1041             0         1
```

Detailed data per video

Want more depth of detail?

```
video_details = get_video_details(video_id = video_identifier)
video_details
```

```
## $kind
## [1] "youtube#videoListResponse"
##
## $etag
## [1] "\"XpPGQXPnxQJhLgs6enD_n8JR4Qk/jSWIMSiXSwvh-NUIY6h8ErCkZhw\""
##
## $pageInfo
## $pageInfo$totalResults
## [1] 1
##
## $pageInfo$resultsPerPage
## [1] 1
##
##
## $items
## $items[[1]]
## $items[[1]]$kind
## [1] "youtube#video"
```

Detailed data per video

A closer look:

```
items[[1]]$snippet$thumbnails$high$url
```

https://i.ytimg.com/vi/_H-UnmiMc3s/hqdefault.jpg

Think of:

- thumbnails for propaganda
- clickbait understanding

Comments per video

Comments made below the video:

New video: Introducing the Numberphile Podcast

```
video_identifier_2 = '0GzhWPj4-cw'

video_comments = get_all_comments(video_id = video_identifier_2)
names(video_comments)
```

```
##  [1] "authorDisplayName"      "authorProfileImageUrl"
##  [3] "authorChannelUrl"       "authorChannelId.value"
##  [5] "videoId"                "textDisplay"
##  [7] "textOriginal"           "canRate"
##  [9] "viewerRating"           "likeCount"
## [11] "publishedAt"            "updatedAt"
## [13] "id"                     "parentId"
## [15] "moderationStatus"
```

Comments per video

A closer look:

```
head(video_comments)
```

```
##   authorDisplayName
## 1          For Phone
## 2       Martin Wujet
## 3         DarknessFX
## 4        vignesh war
## 5        Knives Town
## 6    Leonardo Castro
##
## 1 https://yt3.ggpht.com/-zzteRRx5IS4/AAAAAAAAAAI/AAAAAAAAAAA/n5vSdvlu
## 2 https://yt3.ggpht.com/-M-LYAfoZxEw/AAAAAAAAAAI/AAAAAAAAAAA/YSgUM7jn
## 3 https://yt3.ggpht.com/-LCk4tCJGpyw/AAAAAAAAAAI/AAAAAAAAAAA/i9d94vyW
## 4 https://yt3.ggpht.com/-Vp8GoRNPR3s/AAAAAAAAAAI/AAAAAAAAAAA/qwchRgCi
## 5 https://yt3.ggpht.com/-xOvjHZvnw2c/AAAAAAAAAAI/AAAAAAAAAAA/BFjyPiiy
## 6 https://yt3.ggpht.com/-2Vp8jMULXk8/AAAAAAAAAAI/AAAAAAAAAAA/oH74EQ4o
##                                          authorChannelUrl
## 1 http://www.youtube.com/channel/UC5VzcYt1NpagomAQ1I6gksw
## 2 http://www.youtube.com/channel/UCIF-cT20nIjlamcp2qny_Xg
## 3 http://www.youtube.com/channel/UCZSYME-OGU4tZCIj2BcG6EA
```

Transcript (captions)

YouTube transcrips yet untapped source!

Few exceptions:

- Identifying the sentiment styles of YouTube's vloggers
- Social Media Sellout: The Increasing Role of Product Promotion on YouTube

Transcript (captions)

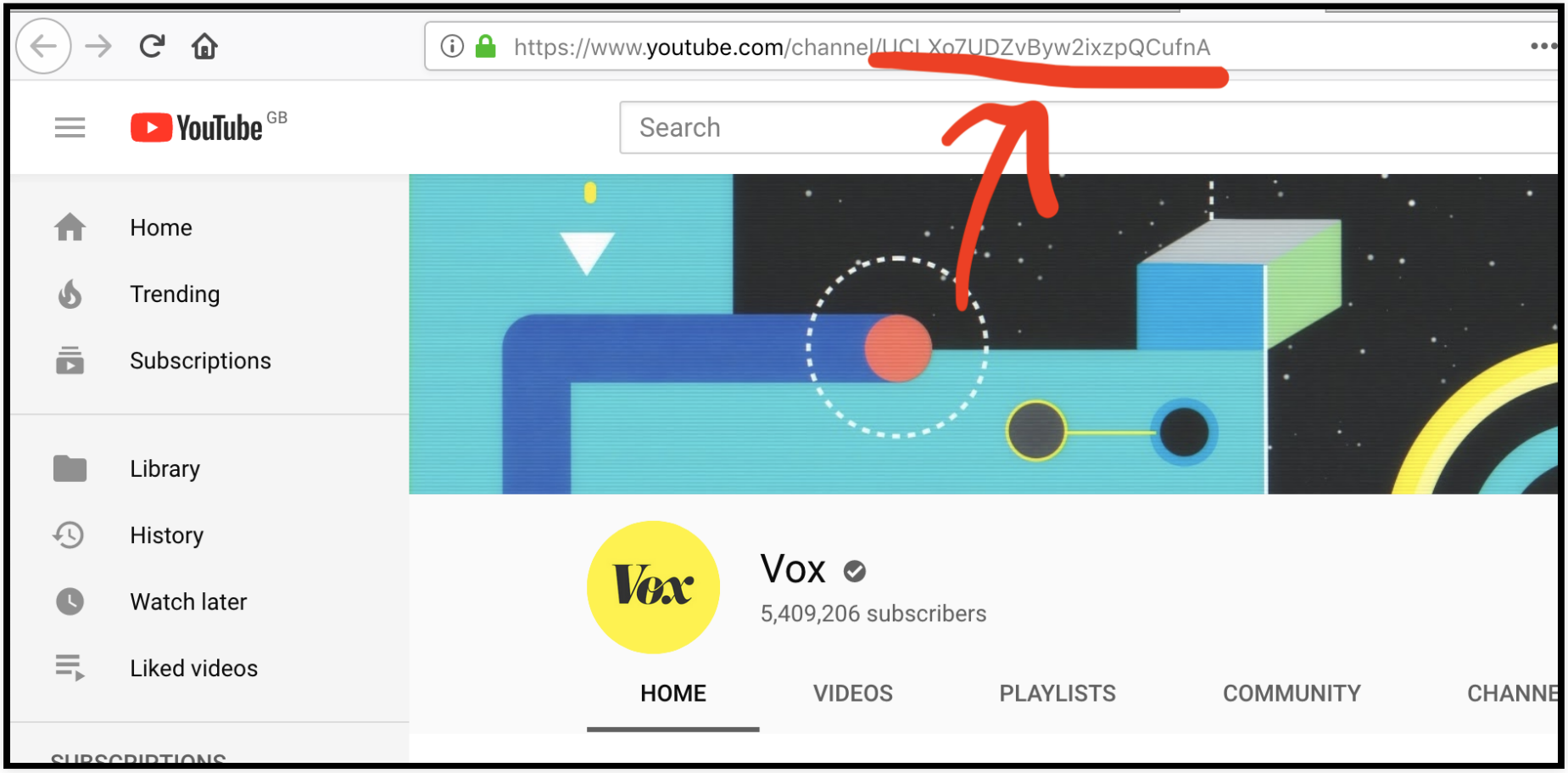
Tricky to get transcripts.

- uploader needs to provide proper transcript (high quality, low availibility)
- workaround approach (moderate/low quality, high availibility)

```
?get_captions
?list_caption_tracks
```

Meta data per channel

First: identify the channel ID.



Meta data per channel

```
channel_identifier = 'UCLXo7UDZvByw2ixzpQCufnA'
channel_stats = get_channel_stats(channel_id = channel_identifier)
```

```
## Channel Title: Vox
## No. of Views: 1209094639
## No. of Subscribers: 5426703
## No. of Videos: 946
```

```
channel_stats
```

```
## $kind
## [1] "youtube#channel"
##
## $etag
## [1] "\"XpPGQXPnxQJhLgs6enD_n8JR4Qk/u5LBm16TDKiRba0uYloNu5kz1aQ\""
##
## $id
## [1] "UCLXo7UDZvByw2ixzpQCufnA"
##
## $snippet
## $snippet$title
## [1] "Vox"
##
## $snippet$description
## [1] "Vox helps you cut through the noise and understand what's drivin
##
## $snippet$customUrl
## [1] "voxdotcom"
```

Meta data per channel

Some statistics per channel:

```
channel_stats$statistics
```

```
## $viewCount
## [1] "1209094639"
##
## $commentCount
## [1] "0"
##
## $subscriberCount
## [1] "5426703"
##
## $hiddenSubscriberCount
## [1] FALSE
##
## $videoCount
## [1] "946"
```

Activity data per channel

New channel (smaller): JSNation

```
channel_identifier_2 = 'UCQM428Hwrvxla8DCgjGONSQ'
channel_activity = list_channel_activities(filter = c(channel_id = channe
                                                      max_results = 50)
names(channel_activity)
```

```
##  [1] "publishedAt"                "channelId"
##  [3] "title"                      "description"
##  [5] "thumbnails.default.url"     "thumbnails.default.width"
##  [7] "thumbnails.default.height"  "thumbnails.medium.url"
##  [9] "thumbnails.medium.width"    "thumbnails.medium.height"
## [11] "thumbnails.high.url"        "thumbnails.high.width"
## [13] "thumbnails.high.height"     "thumbnails.standard.url"
## [15] "thumbnails.standard.width"  "thumbnails.standard.height"
## [17] "thumbnails.maxres.url"      "thumbnails.maxres.width"
## [19] "thumbnails.maxres.height"   "channelTitle"
## [21] "type"                       "groupId"
```

Activity data per channel

```
head(channel_activity)
```

```
##                publishedAt                channelId
## 1 2018-12-10T09:58:22.000Z UCQM428Hwrvxla8DCgjGONSQ
## 2 2018-12-13T16:07:13.000Z UCQM428Hwrvxla8DCgjGONSQ
## 3 2018-12-10T09:57:31.000Z UCQM428Hwrvxla8DCgjGONSQ
## 4 2018-12-13T16:07:05.000Z UCQM428Hwrvxla8DCgjGONSQ
## 5 2018-12-10T09:54:19.000Z UCQM428Hwrvxla8DCgjGONSQ
## 6 2018-12-13T16:06:52.000Z UCQM428Hwrvxla8DCgjGONSQ
##                                                                  t
## 1 How to refactor JavaScript with JavaScript on a massive scale – Kers
## 2 How to refactor JavaScript with JavaScript on a massive scale – Kers
## 3           Creating IoT Applications with Web Bluetooth – Martin Wool
## 4           Creating IoT Applications with Web Bluetooth – Martin Wool
## 5               The impostor syndrome aka I'm a fraud – Claudio Semer
## 6               The impostor syndrome aka I'm a fraud – Claudio Semer
##
## 1 Talk recording from AmsterdamJS December 2018 Meetup https://www.mee
## 2 Talk recording from AmsterdamJS December 2018 Meetup https://www.mee
## 3                    Talk recording from AmsterdamJS December 2018 Me
```

Video stats per channel

Stats for all videos in a channel:

```
all_video_stats = get_all_channel_video_stats(channel_id = channel_identi
names(all_video_stats)
```

Video stats per channel

```
head(all_video_stats)
```

```
##           id
## 1 _4nrh6mTt4E
## 2 _iIxC8ziZNM
## 3 -BGxJn3c7NA
## 4 -CGpVrydTyg
## 5 0t9FERJRShQ
## 6 1eH9-cLMXQg
##
## 1                                    Amsterdam JSNation Conference 201
## 2                                 Smart Contracts in JavaScript - Mikha
## 3                            TypeScript Ruined My Life (In a Good Way) -
## 4 SonarJS: How To Build a Static Code Analyzer - Elena Vilchik & Carlo
## 5                                          The dark ages of IoT - Sebas
## 6                       In the Ocean of Angular Web Applications - Yap
##           publication_date viewCount likeCount dislikeCount favoriteCo
## 1 2018-06-01T16:58:23.000Z      2886        61             0
## 2 2018-04-08T17:11:16.000Z       351         8             0
## 3 2018-06-08T14:34:50.000Z       332         1             1
```

Additional queries

- subscriber info: `get_subscriptions`
- list all videos: `get_`

Problems with Twitter/YouTube data?

Some issues:

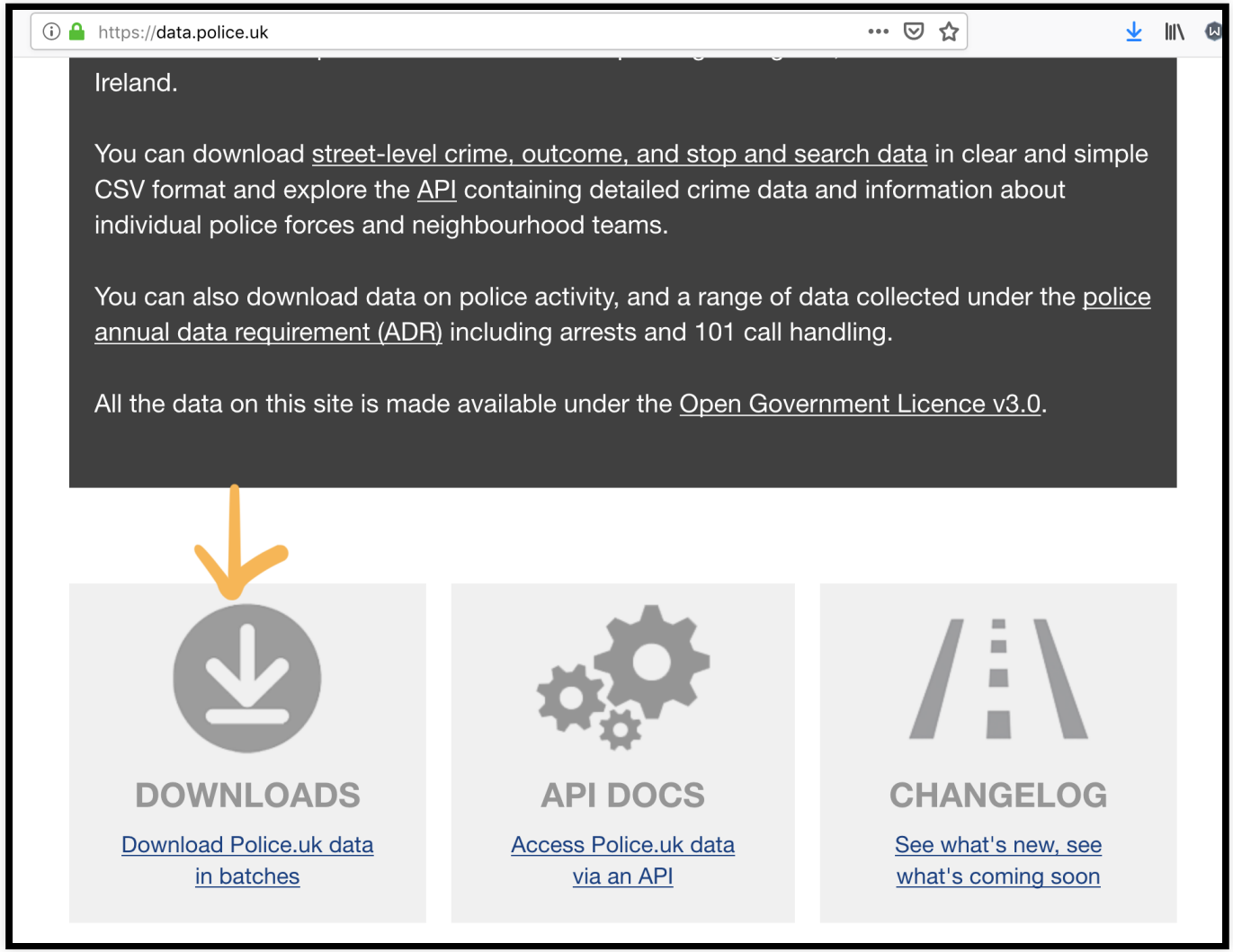
- Sample representativeness
- ~~Location accuracy~~
- ~~Location availability~~
- ~Sampling through API~
- Transcript quality
- Real representations (filtered)
- Censored?

Crime data interfaces

- police.uk
- R package “crimedata”

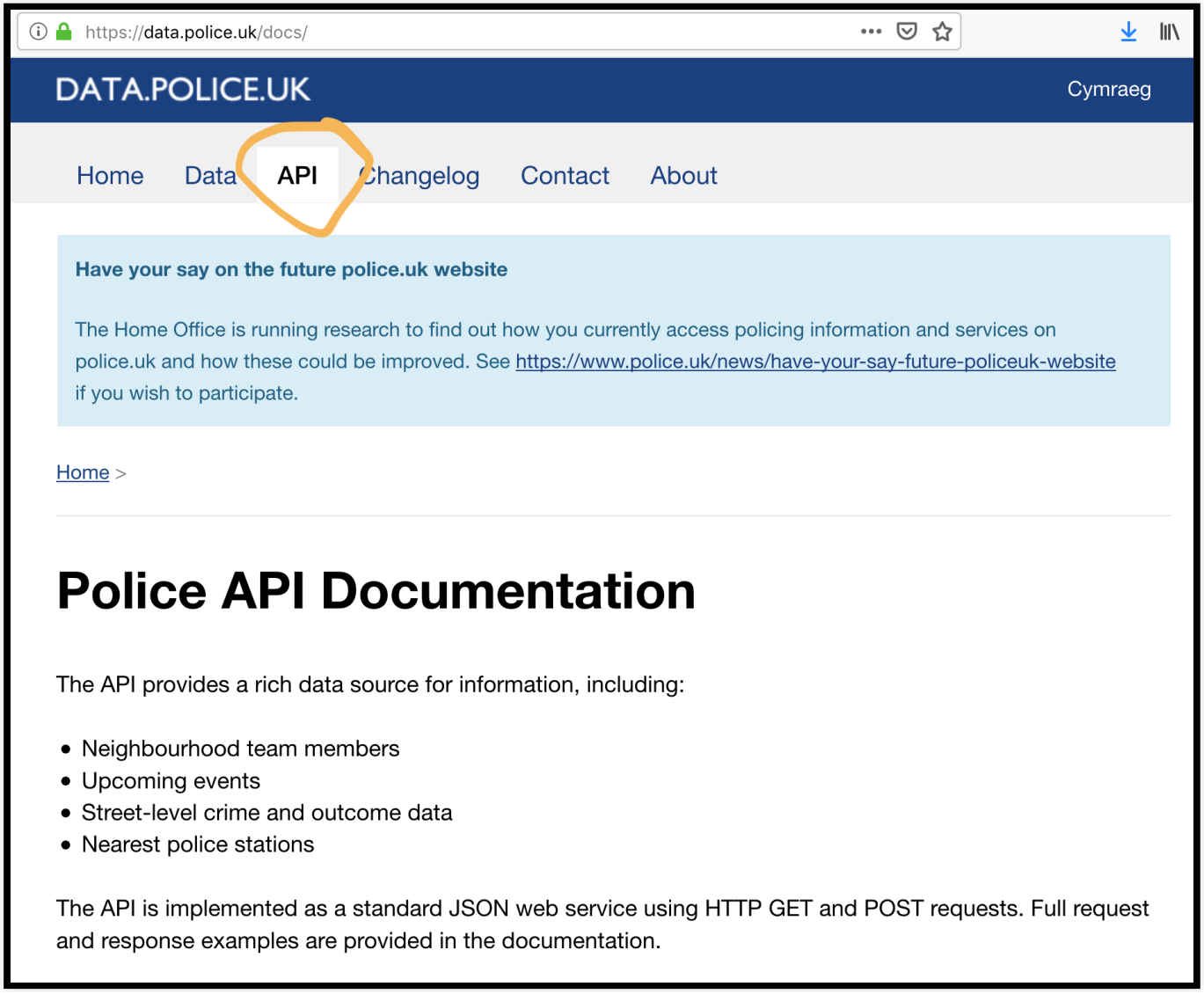
police.uk as data repository

Public database of "open police data"



police.uk as data repository

API?



police.uk as data repository

Using the API:

- different method
- calls direct from the browser
- no R implementation

The API is implemented as a standard JSON web service using HTTP GET and POST requests. Full request and response examples are provided in the documentation.

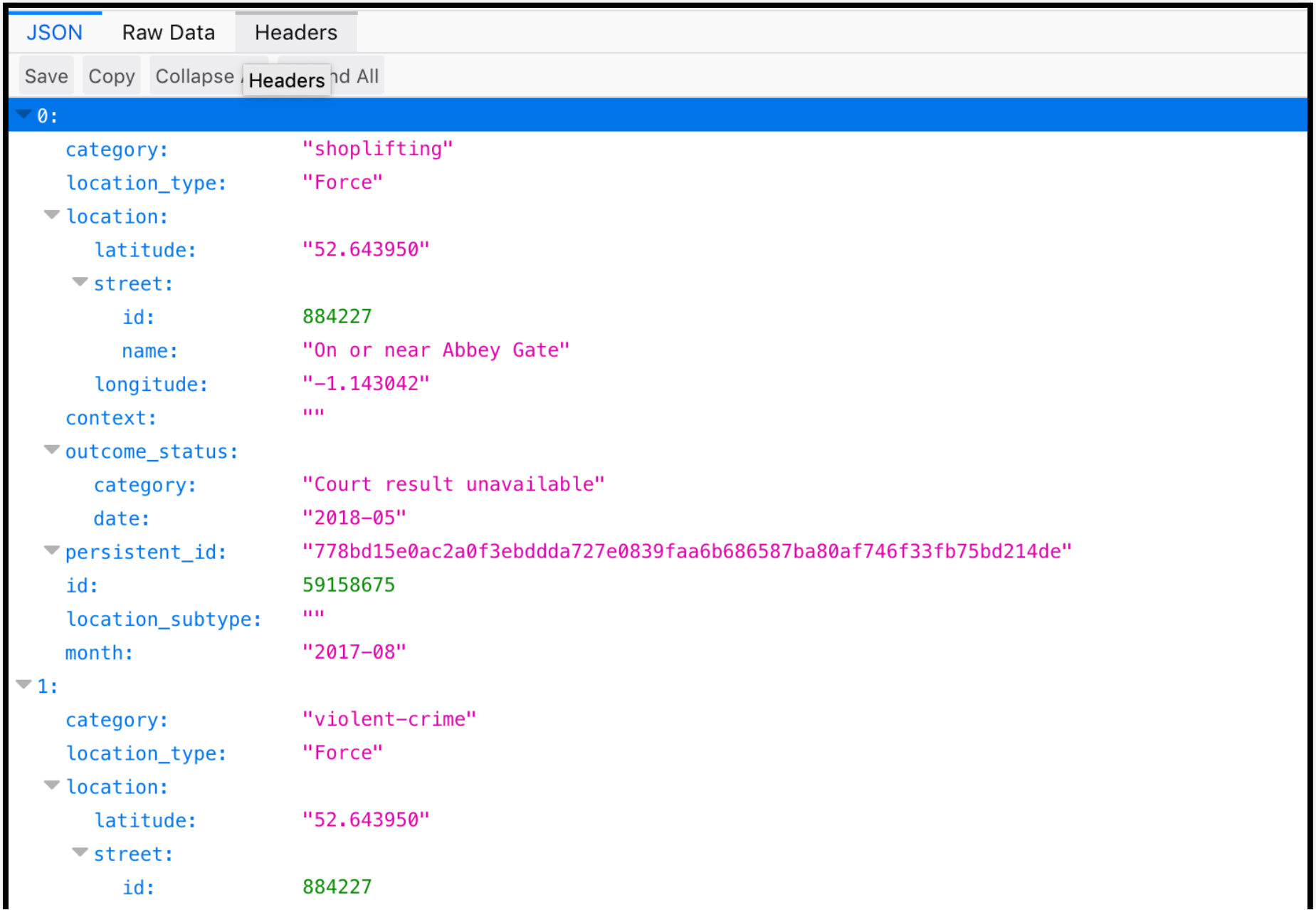
police.uk as data repository

“Crimes at location”

Search query example:

```
https://data.police.uk/api/crimes-at-location?date=2017-08&location_id=8
```

police.uk as data repository



```
      name:          "On or near Abbey Gate"
    longitude:       "-1.143042"
  context:           ""
▼ outcome_status:
    category:        "Unable to prosecute suspect"
    date:            "2017-08"
▼ persistent_id:     "79aeb5c45e8fb2843306d70b969c3012762271c794c0c973010d1d3b53539797"
  id:                59157496
  location_subtype:  ""
  month:             "2017-08"
```

`crimedata` package

```
library(crimedata)
```

`crimedata` package

Aim:

Gives convenient access to publicly available police-recorded open crime data from large cities in the United States that are included in the Crime Open Database

Open Crime Database

`crimedata` package

Which data are available?

```
list_crime_data(quiet = FALSE)
```

```
## Downloading list of URLs for data files. This takes a few seconds but
```

```
## # A tibble: 11 x 2
##    city          years
##    <chr>         <chr>
##  1 all cities    2007 to 2017
##  2 Chicago       2007 to 2017
##  3 Detroit       2009 to 2017
##  4 Fort Worth    2007 to 2017
##  5 Kansas City   2009 to 2017
##  6 Los Angeles   2010 to 2017
##  7 Louisville    2009 to 2017
##  8 New York      2007 to 2017
##  9 San Francisco 2007 to 2017
## 10 Tucson        2009 to 2017
## 11 Virginia Beach 2013 to 2017
```

`crimedata` package

Getting data:

```
crime_data_ny_2017 = get_crime_data(years = 2017
                                    , cities = c("New York"))
```

```
## Using cached URLs to get data from server. These URLs rarely change ar
```

```
## Downloading sample data for New York in 2017
```

```
##
   |
   |                                                                      |    0
   |
   |======================================================================| 100
##
   |
   |                                                                      |    0
   |
   |=                                                                     |    2
   |
   |==                                                                    |    3
   |
   |==                                                                    |    4
   |
```

```
===
```

```
names(crime_data_ny_2017)
```

```
##  [1] "uid"                "city_name"          "offense_code"
##  [4] "offense_type"       "offense_group"      "offense_against"
##  [7] "date_single"        "date_start"         "date_end"
## [10] "longitude"          "latitude"           "location_type"
## [13] "location_category" "census_block"
```

```
head(crime_data_ny_2017)
```

```
## # A tibble: 6 x 14
##       uid city_name offense_code offense_type offense_group offense_aga
##     <int> <chr>     <chr>        <chr>         <chr>         <chr>
## 1 1.38e7 New York  90Z          all other o… all other of… other
## 2 1.38e7 New York  22B          non-residen… burglary/bre… property
## 3 1.38e7 New York  13B          simple assa… assault offe… persons
## 4 1.38e7 New York  90Z          all other o… all other of… other
## 5 1.38e7 New York  12A          personal ro… robbery       property
## 6 1.38e7 New York  90Z          all other o… all other of… other
## # ... with 8 more variables: date_single <chr>, date_start <chr>,
## #   date_end <chr>, longitude <dbl>, latitude <dbl>, location_type <cl
## #   location_category <chr>, census_block <chr>
```

`crimedata` package

Multiple cities, multiple years:

```
crime_data_2010_2015 = get_crime_data(years = 2010:2015
                                      , cities = c("Chicago", "Detroit"))
```

```
## Warning in if (cities != "all" & !all(cities %in% unique(urls$city))) {
## the condition has length > 1 and only the first element will be used
```

```
##
  |
  |                                                                      |   0
  |
  |======================================================================| 100
##
  |
  |                                                                      |   0
  |
  |                                                                      |   1
  |
  |=                                                                     |   2
  |
  |==                                                                    |   4
  |
  |====                                                                  |   5
```

```
table(crime_data_2010_2015$city_name, crime_data_2010_2015$offense_agains
```

```
##
##           other persons property society
##   Chicago  1434    4811     9658    2992
##   Detroit   480    1445     4096     492
```

`crimedata` package

Additional features:

- `nycvehiclethefts`: Dataset containing records of thefts of motor vehicles in New York City from 2014 to 2017
- `homicides15`: Dataset containing records of homicides in nine large US cities in 2015

APIs: Pros & Cons

Pro

- easiy to access
- nicely documentation
- **works even if website changes**

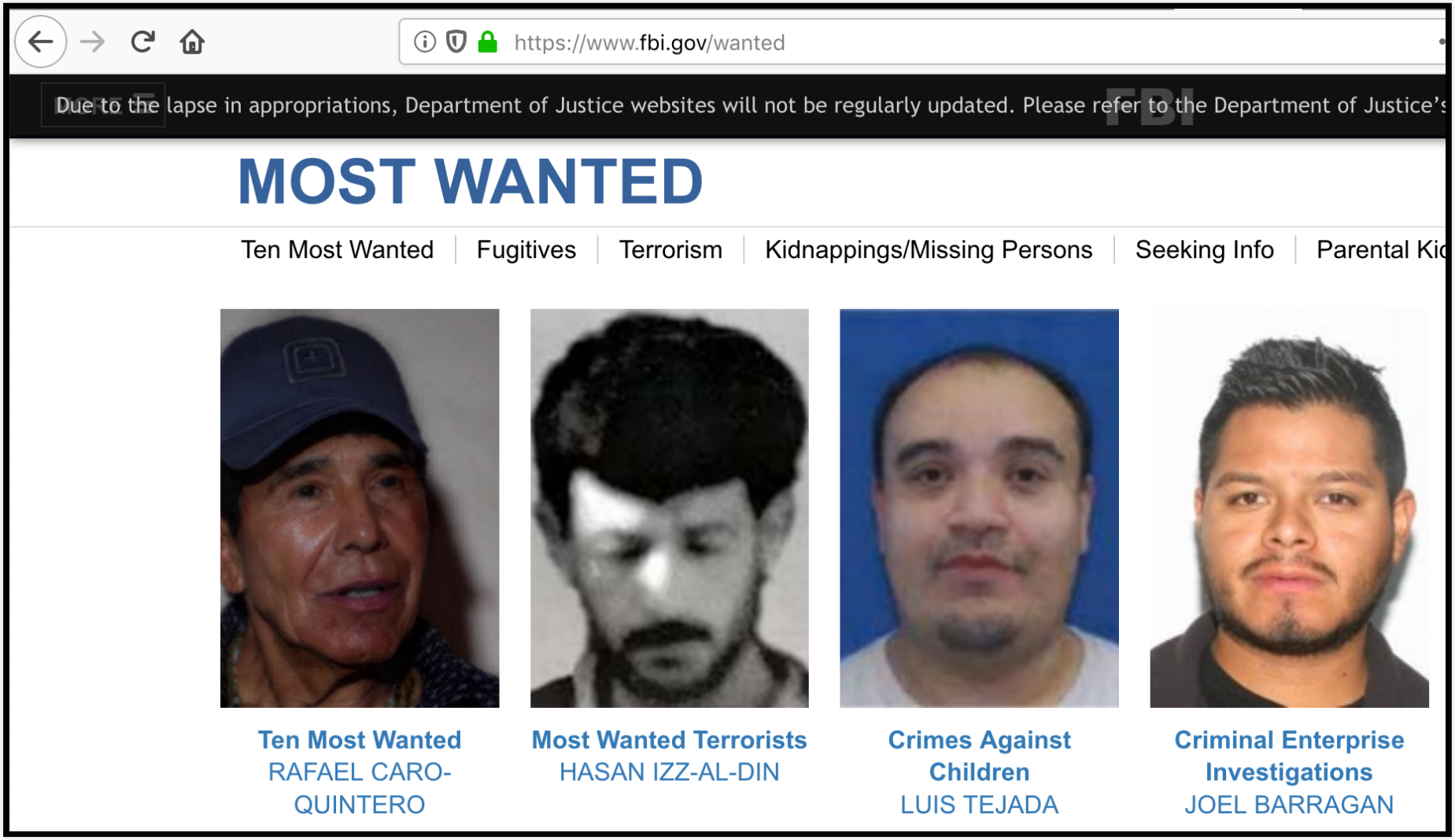
Cons

- quota limits ($ $ $)
- under the platforms' control
- only for few platforms

Don’t let the data determine your research!

COOL

But what about:



UK MISSING PERSONS UNIT

National Crime Agency (GB) | https://www.missingpersons.police.uk/en-gb/case-

Bureau Reference:	18-009282
Location	London
Gender	Male
Date found	21 September 2018
Age	30 - 40
Ethnicity	White European

View case details

View case details

Bureau Reference:	18-006195
Location	Dunsden
Gender	Male
Date found	15 May 2018
Age	16 - 100
Ethnicity	White European

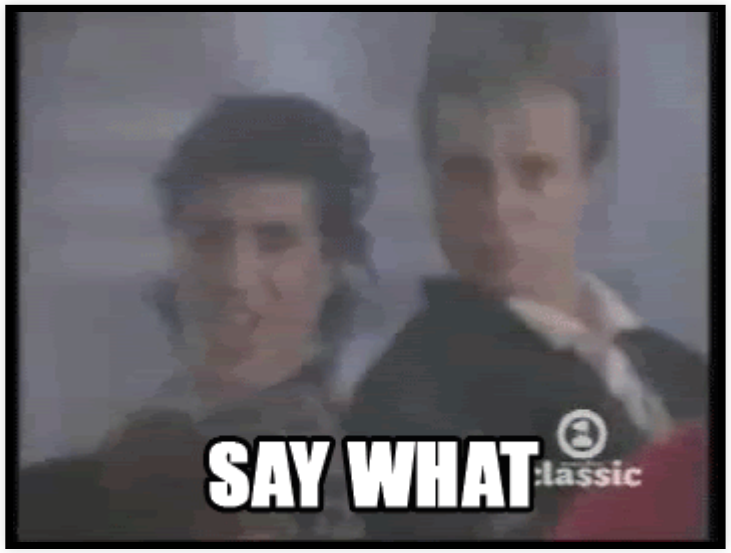
View case details

No APIs

- incels.me
- stormfront
- 4chan
- **APIs are restrictive!**

... what about:

Your research question –> no API?



Main problem:

Really ‘juicy’ data of the Internet vs **APIs**

“Real” webscraping: basics of a webpage

Three elements of a webpage

1. Structure
2. Behaviour
3. Style

Three elements of a webpage

1. Structure
2. Behaviour
   - JavaScript (**!= Java**)
   - user interaction
   - examples: alerts, popups, server-interaction
3. Style

Three elements of a webpage

1. Structure
2. Behaviour
3. Style
    - CSS (Cascading Style Sheets)
    - formatting, design, responsiveness
    - examples: submit buttons, app interaces

Three elements of a webpage

1. Structure
    - HTML (hypertext markup language)
    - structured with `<tags>`
    - contains the pure content of the webpage
2. Behaviour
3. Style

For now: HTML

The very basics of HTML:

Raw architecture of a webpage

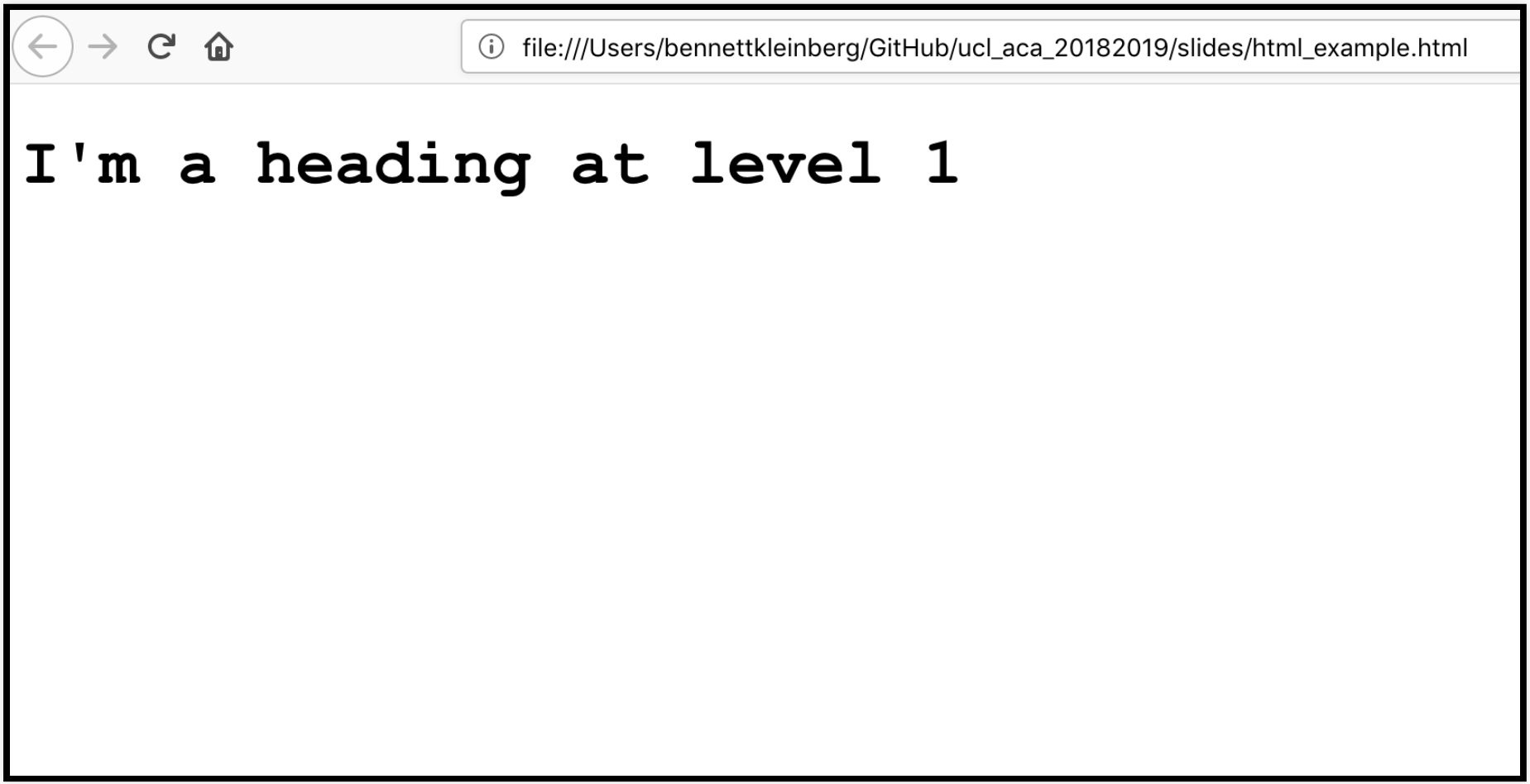
```
<!DOCTYPE html>
<html>
<body>

HERE COMES THE VISIBLE PART!!

</body>
</html>
```

Note: Every tags `< >` is closed `< />`. Content is contained within the tag.

HTML basics

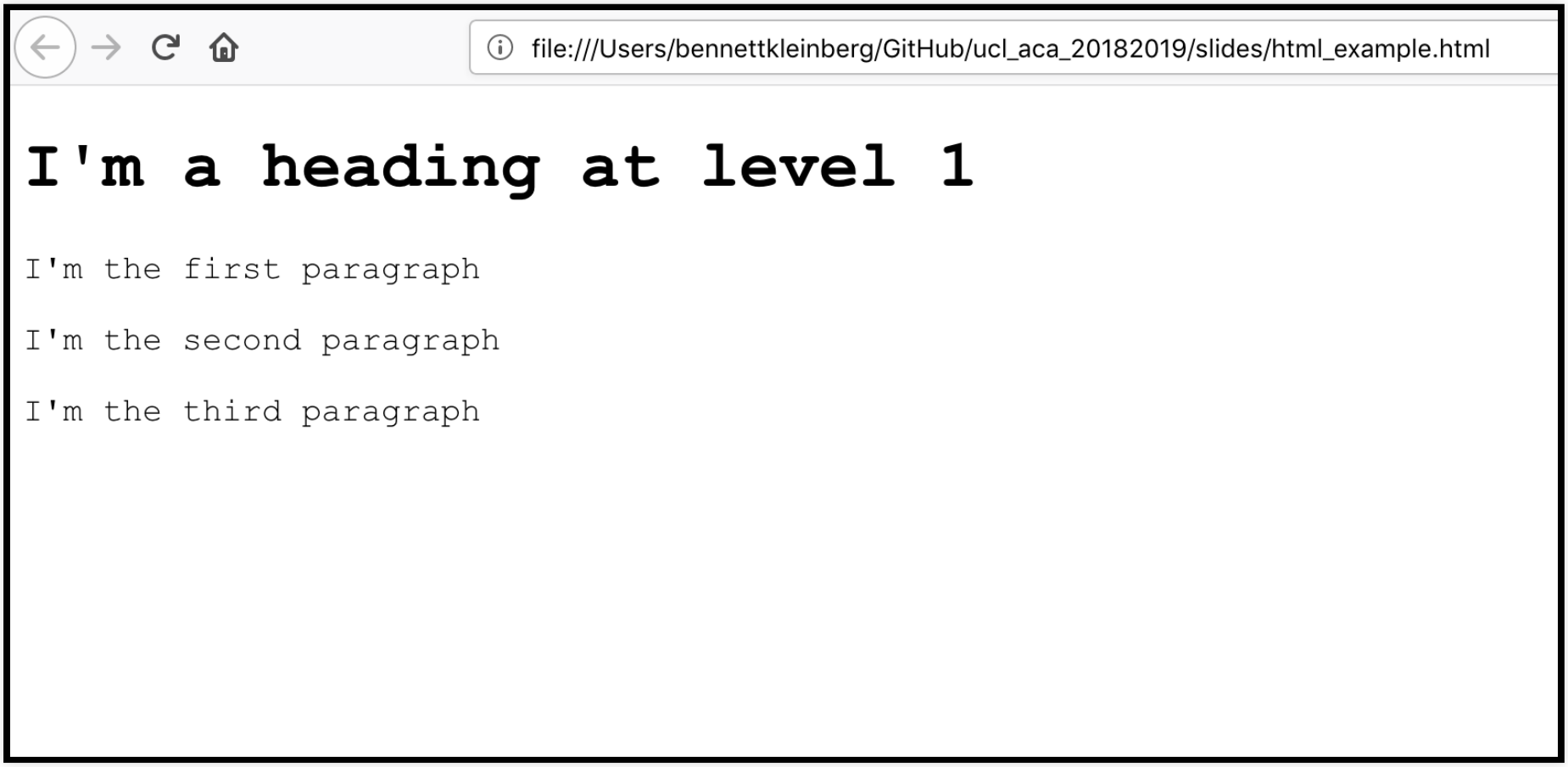
Ways to put content in the `<body> ... </body>` tag:

- headings: `<h1>I'm a heading at level 1</>`



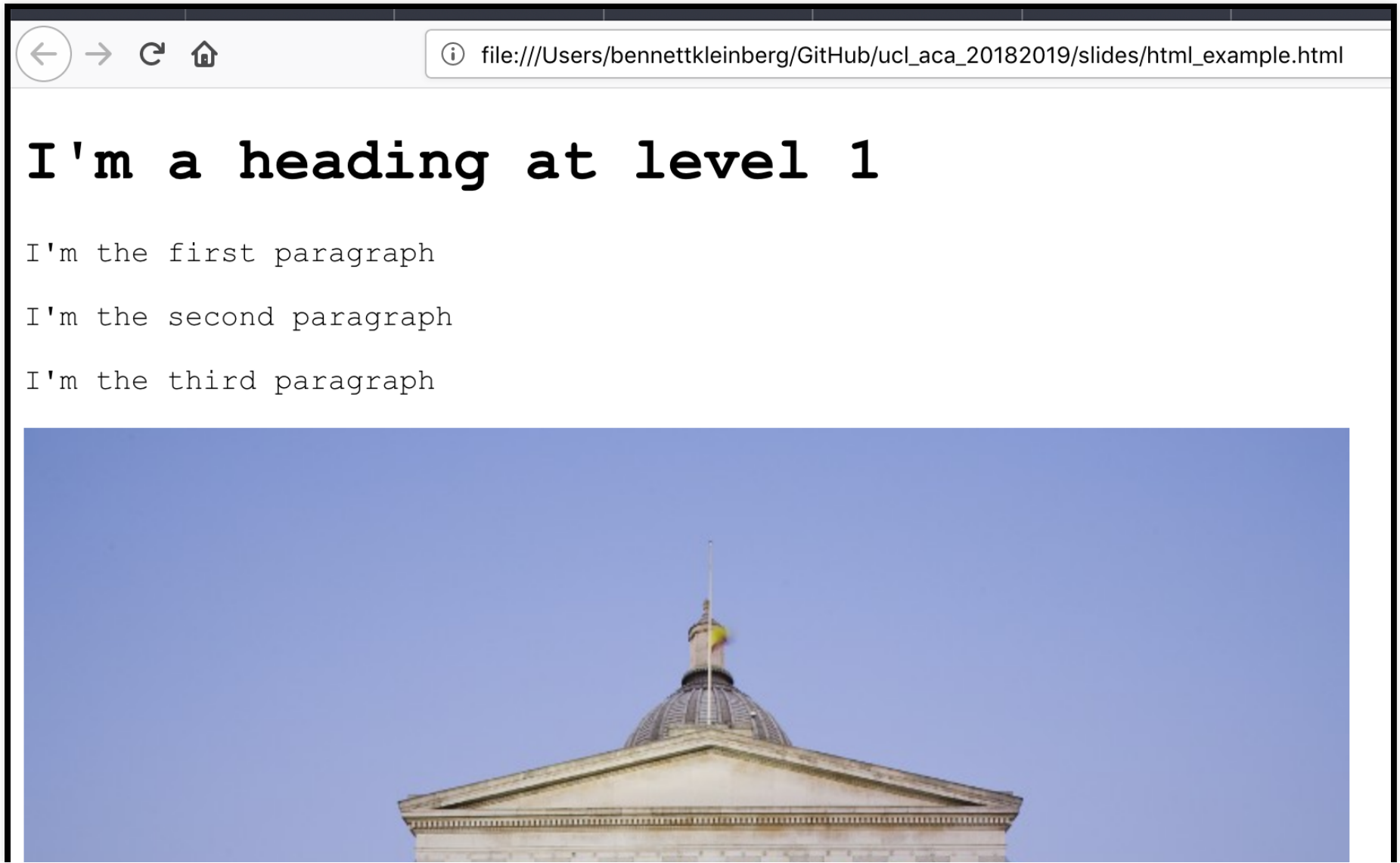
Content in the body tag

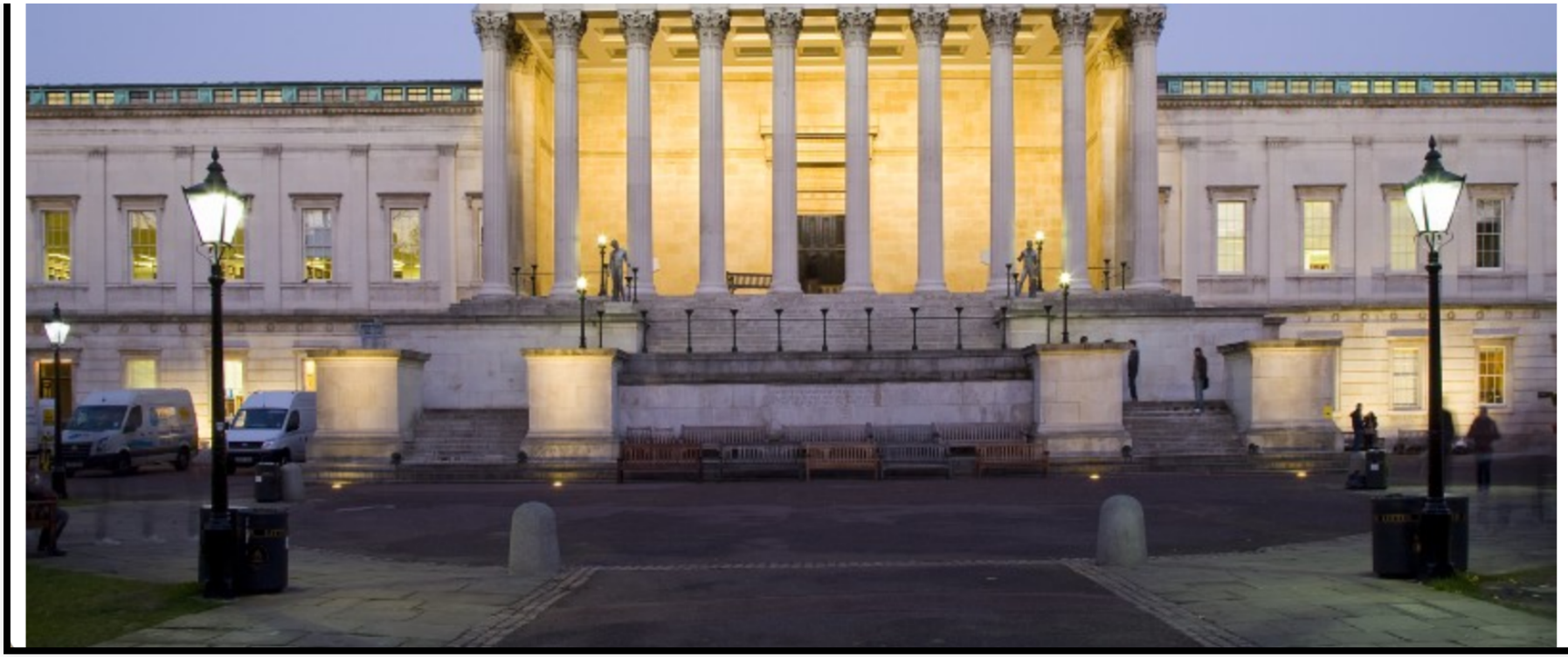
- paragraphs: `<p>This is a paragraph</p>`



Content in the body tag

- images: `<img src="./img/ucl.jpg">`

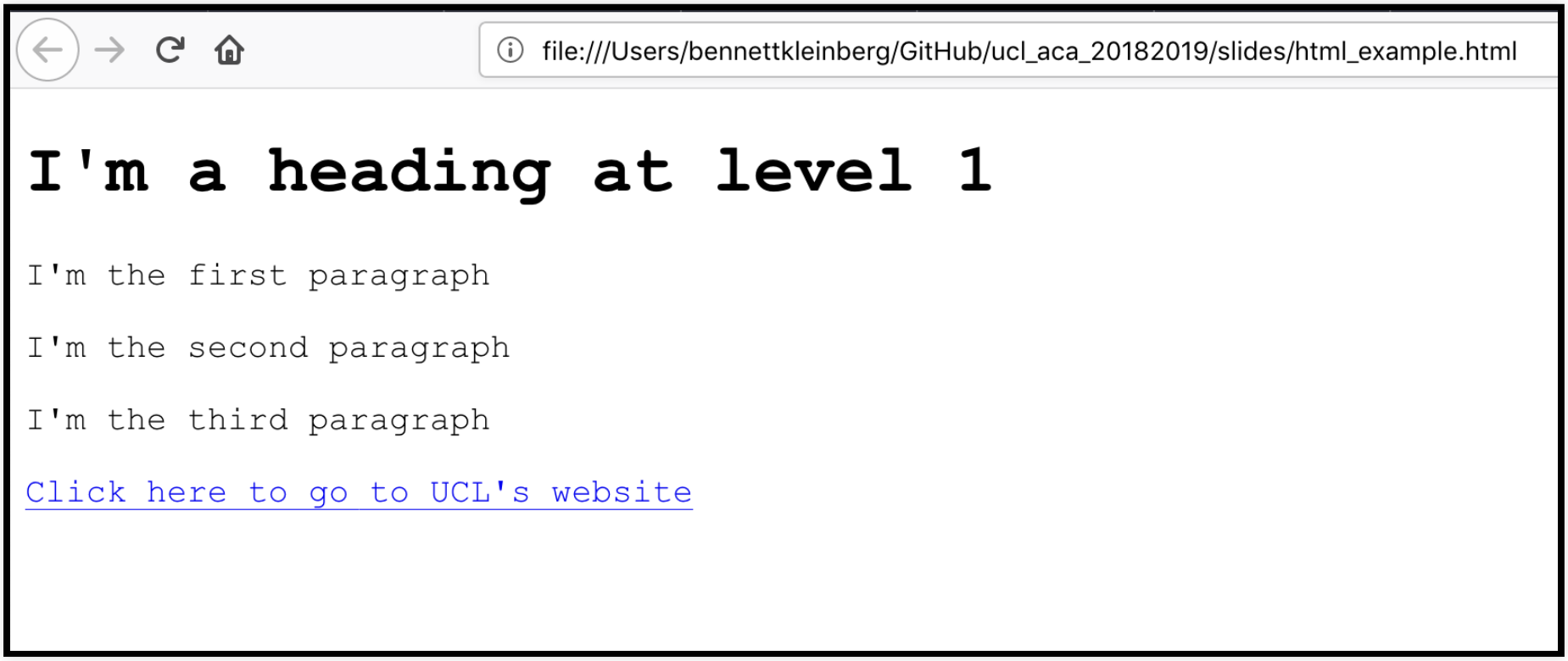




Content in the body tag

- links:

```
<a href="https://www.ucl.ac.uk/">Click here t
```

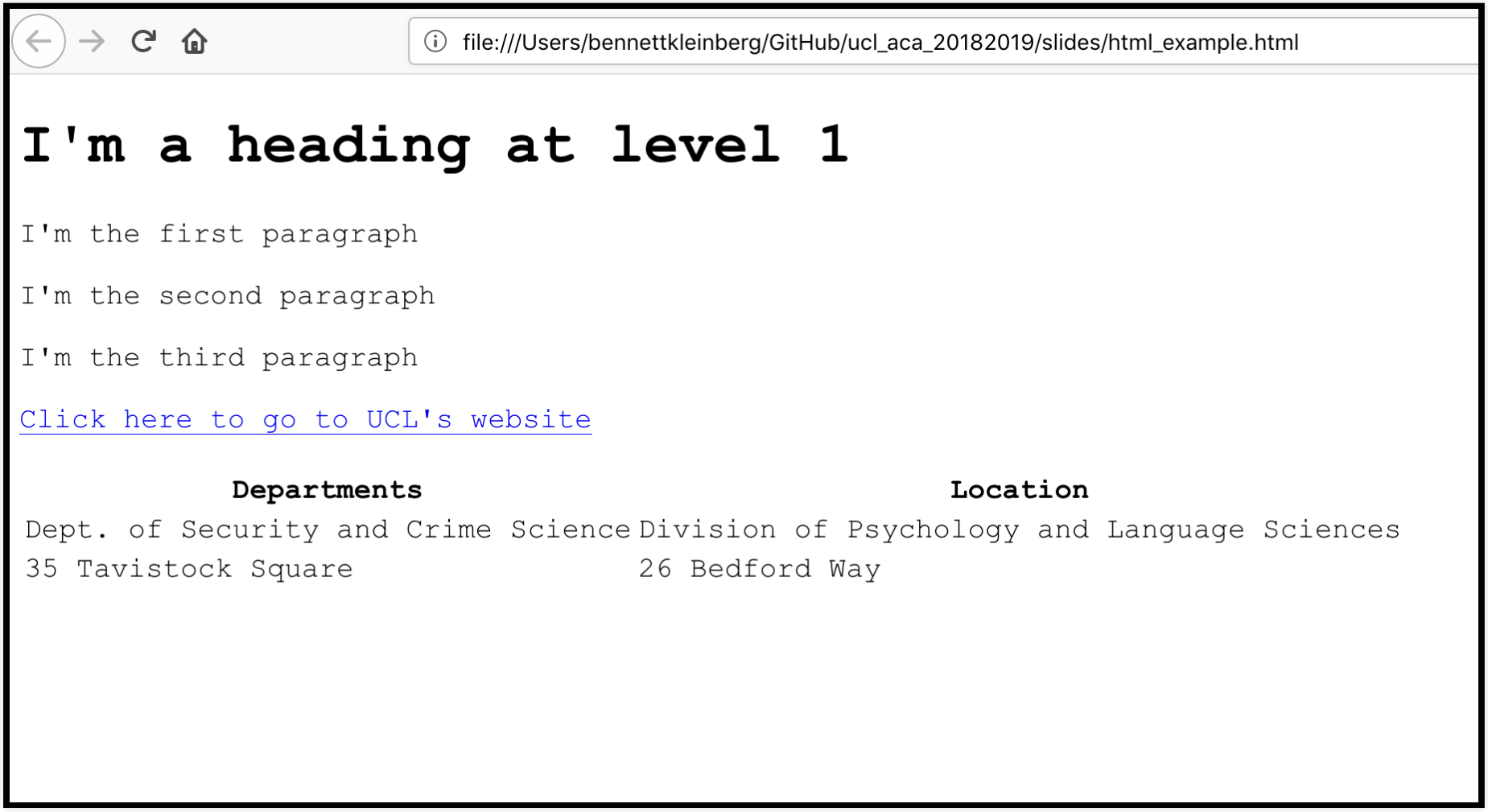


Content in the body tag

- tables

```
<table>
  <tr>
    <th>Departments</th>
    <th>Location</th>
  </tr>
  <tr>
    <td>Dept. of Security and Crime Science</td>
    <td>Division of Psychology and Language Sciences</td>
  </tr>
  <tr>
    <td>35 Tavistock Square</td>
    <td>26 Bedford Way</td>
  </tr>
</table>
```

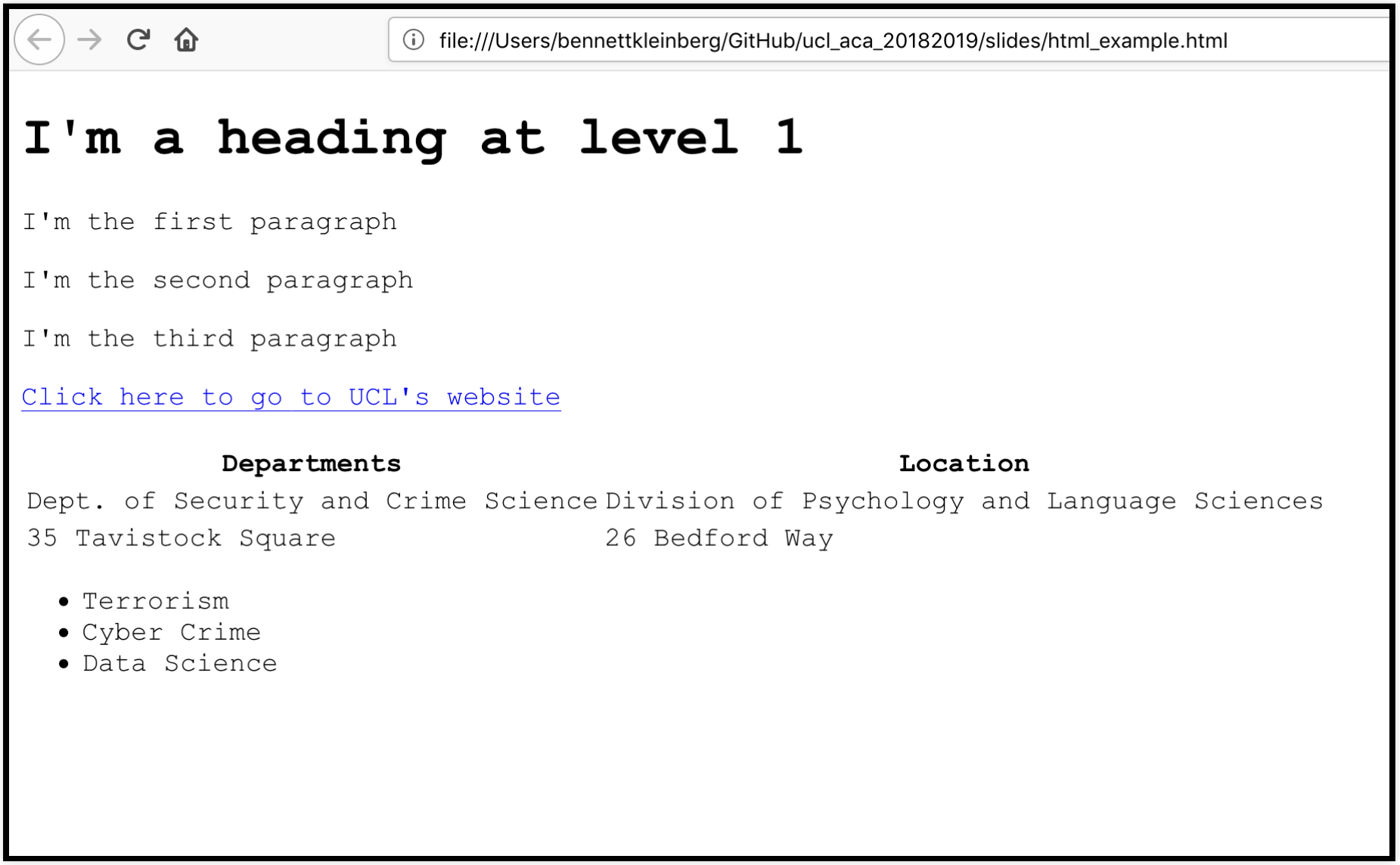
Html `<table>...</table>`



Content in the body tag

- lists

```
<ul>
  <li>Terrorism</li>
  <li>Cyber Crime</li>
  <li>Data Science</li>
</ul>
```


file:///Users/bennettkleinberg/GitHub/ucl_aca_20182019/slides/html_example.html
I'm a heading at level 1
I'm the first paragraph
I'm the second paragraph
I'm the third paragraph
Click here to go to UCL's website
Departments
Location
Dept. of Security and Crime Science
35 Tavistock Square
Division of Psychology and Language Sciences
26 Bedford Way
Terrorism
Cyber Crime
Data Science

HTML basics

Elements (can) have IDs:

```
<p id='paragraph1'>This is a paragraph</p>
<img id='ucl_image' src="./img/ucl.jpg">
```

Same for tables, links, etc.

Every element can have an ID.

You need unique IDs! Two elements cannot have the same ID.

HTML basics

Common elements (can) have CLASSES:

```
<p id="paragraph1" class="paragraph_class">I am the first paragraph</p>
<p class="paragraph_class">I am the second paragraph</p>
<p class="paragraph_class">I am the third paragraph</p>
```

Multiple elements can have the same class.

Now what?

Web scraping logic

If all webpages are built in this structure…

… then we could access this structure programmatically.

But where do I find that structure?

Is it just “there”?

YES!!

How to see the html structure?

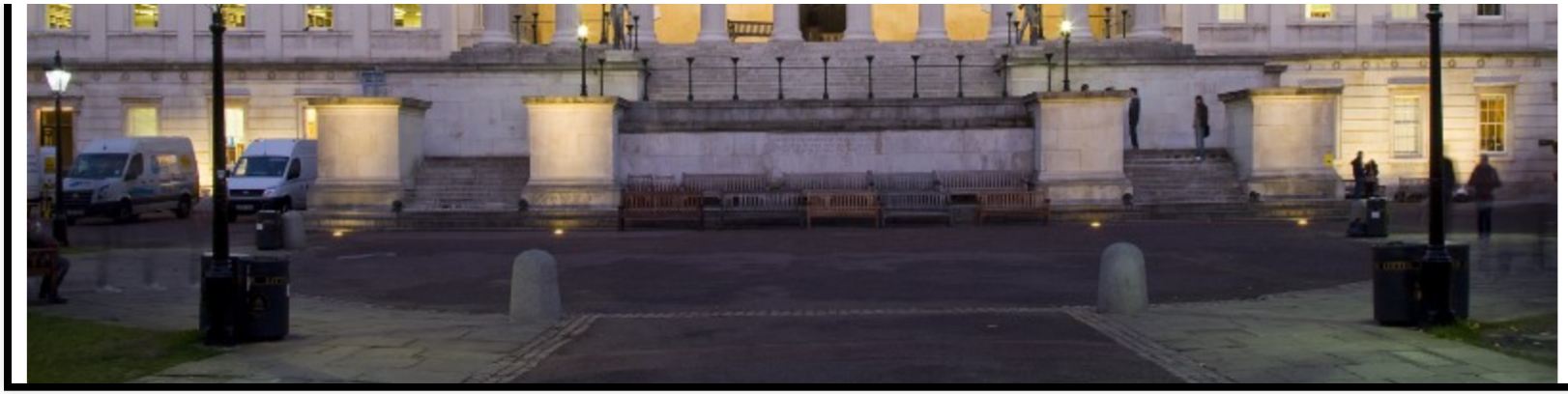
file:///Users/bennettkleinberg/GitHub/ucl_aca_20182019/slides/html_example.html

I'm a heading at level 1

I'm the first paragraph

I'm the second paragraph

I'm the third paragraph



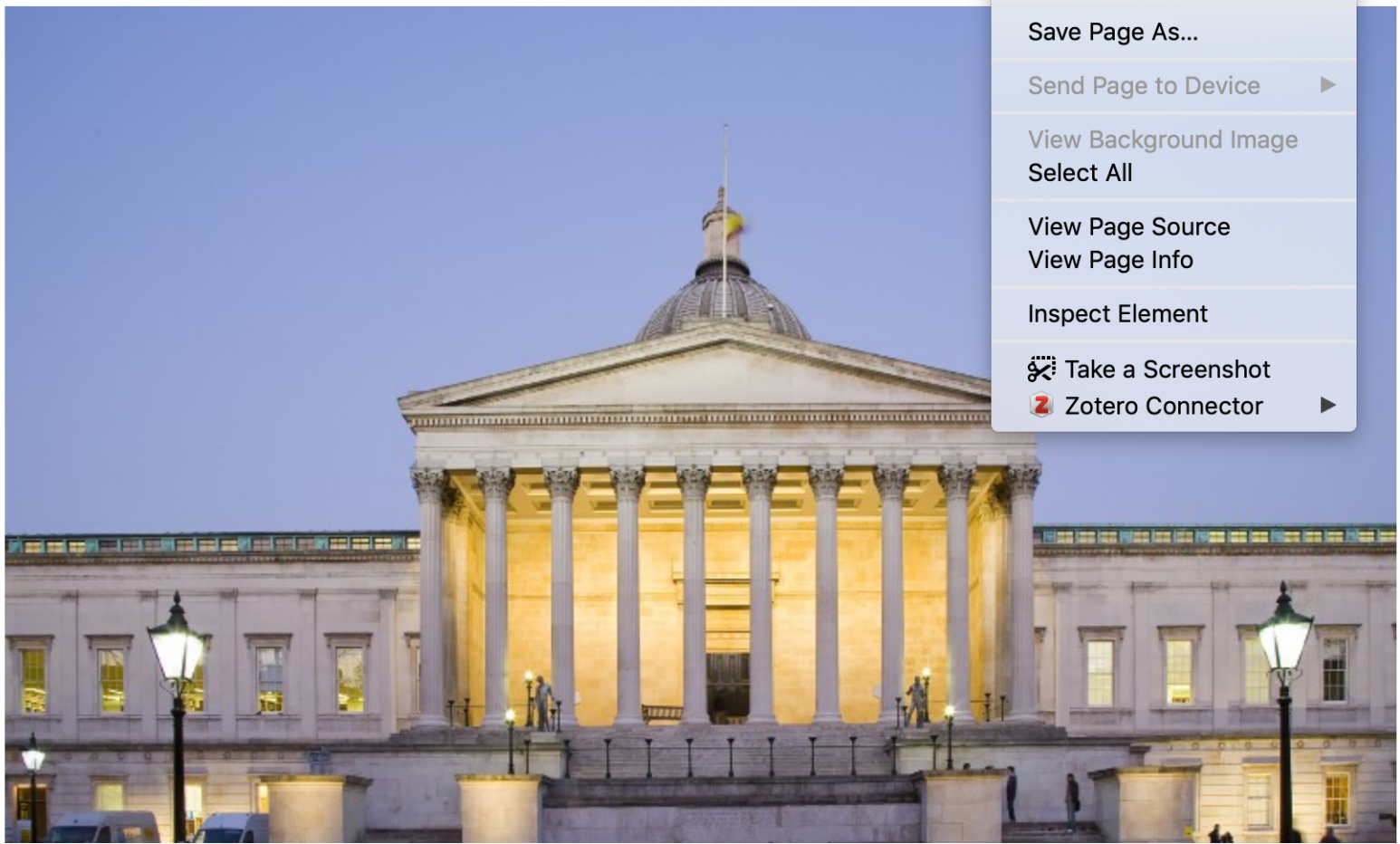
file:///Users/bennettkleinberg/GitHub/ucl_aca_20182019/slides/html_example.html

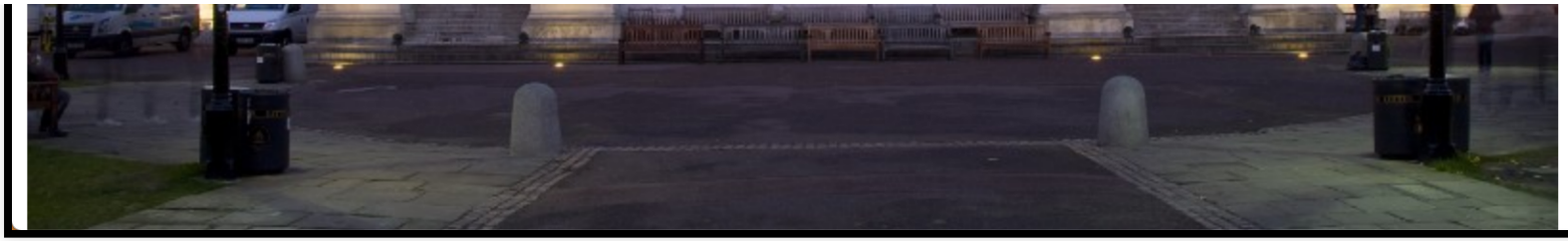
I'm a heading at level 1

I'm the first paragraph

I'm the second paragraph

I'm the third paragraph





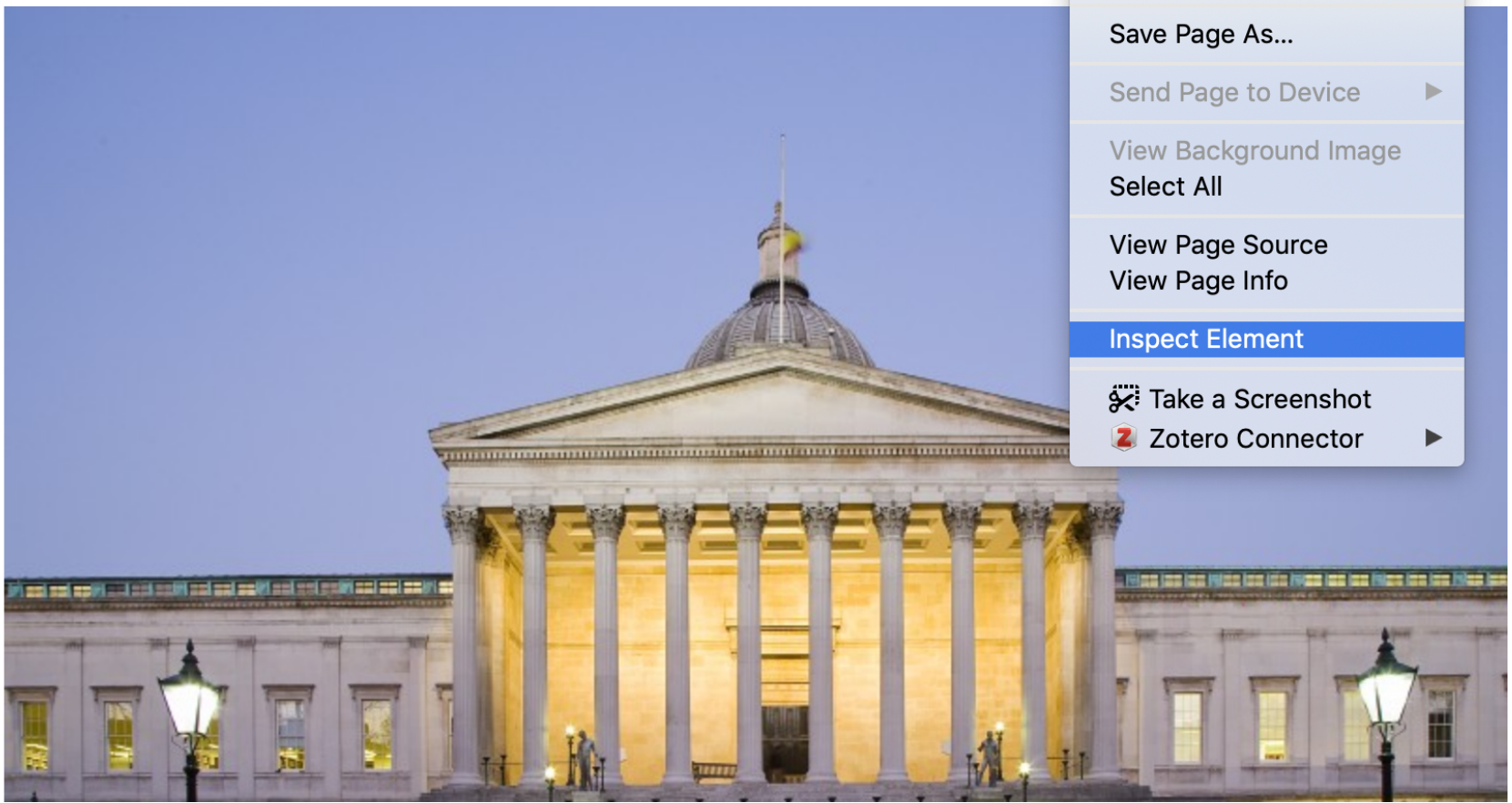
file:///Users/bennettkleinberg/GitHub/ucl_aca_20182019/slides/html_example.html

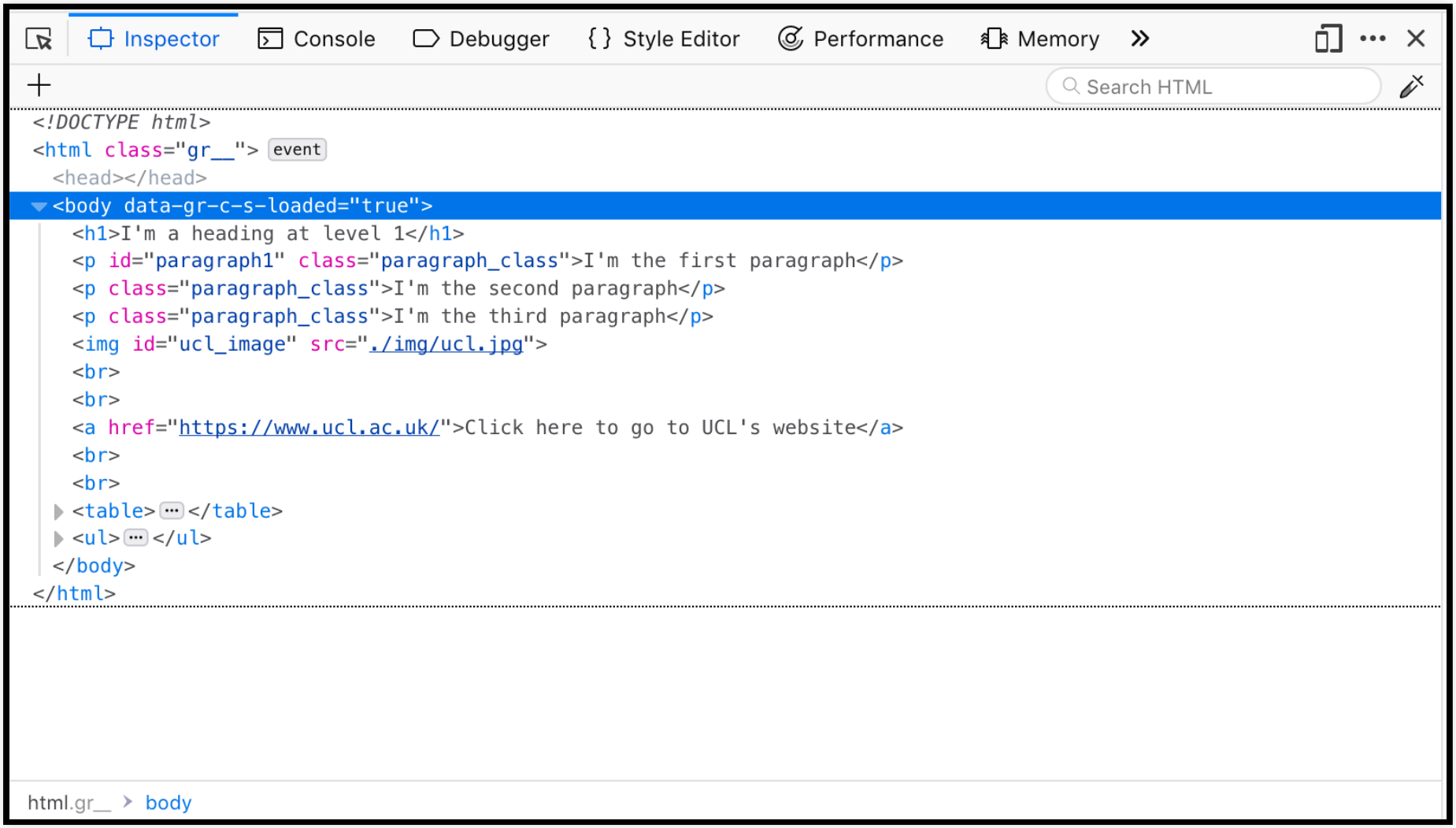
I'm a heading at level 1

I'm the first paragraph

I'm the second paragraph

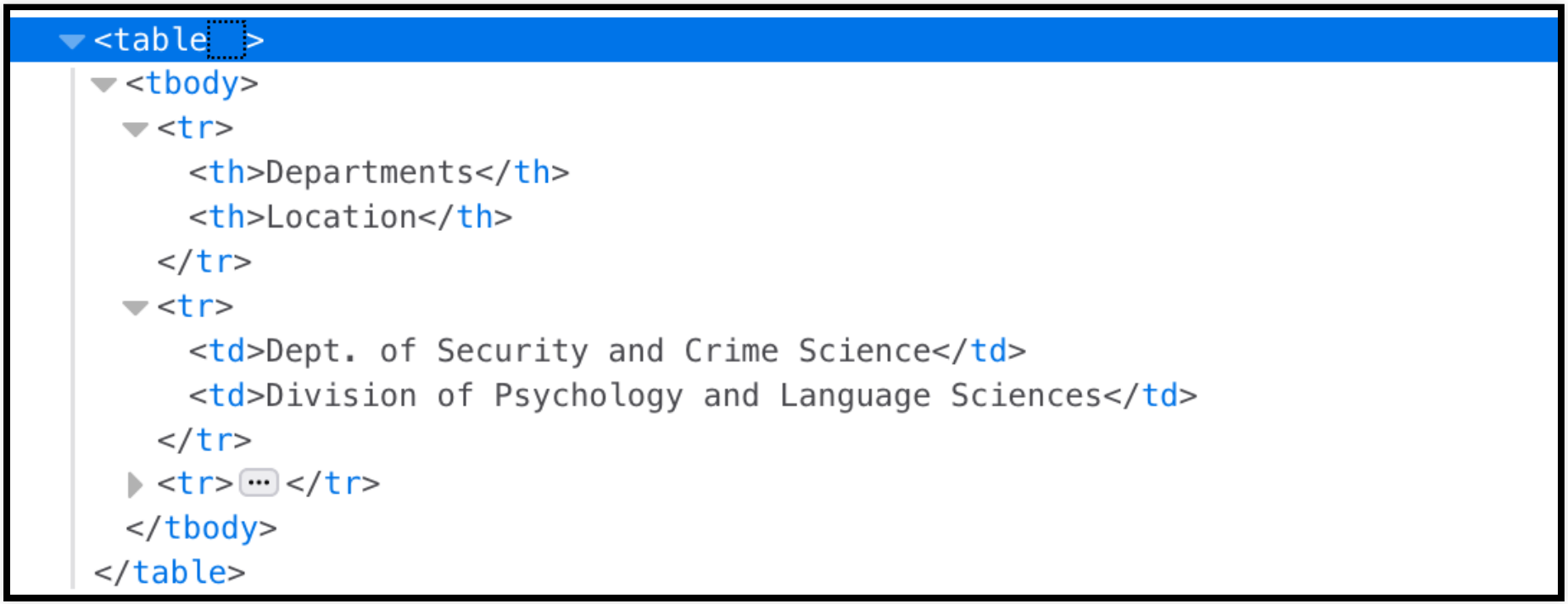
I'm the third paragraph



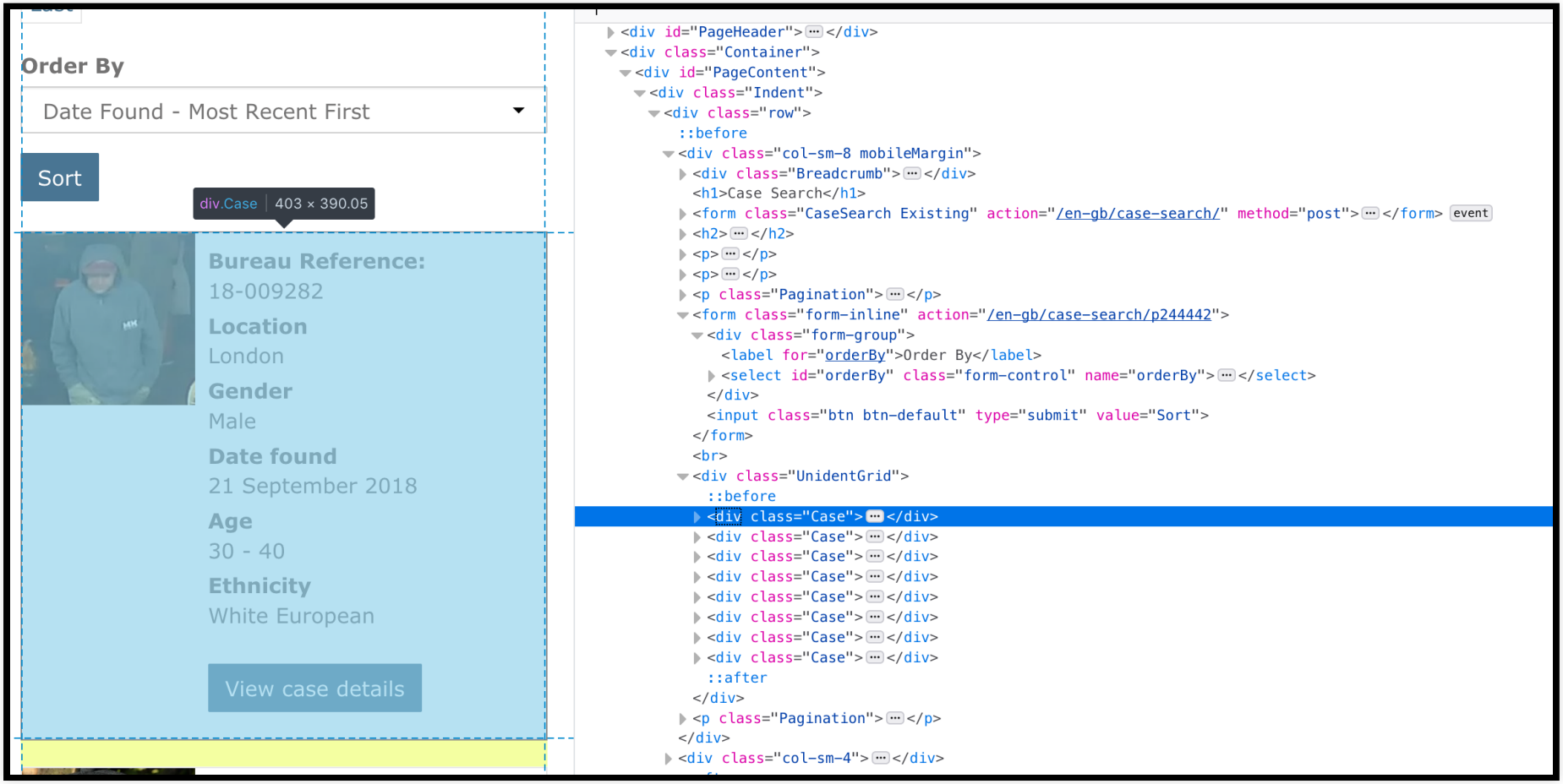

Inspector
Console
Debugger
Style Editor
Performance
Memory
Search HTML
<!DOCTYPE html>
<html class="gr__"> event
<head></head>
<body data-gr-c-s-loaded="true">
<h1>I'm a heading at level 1</h1>
I'm the first paragraph
I'm the second paragraph
I'm the third paragraph
<img id="ucl_image" src="./img/ucl.jpg">

Click here to go to UCL's website

<table> ··· </table>
 ···
</body>
</html>
html.gr__ > body

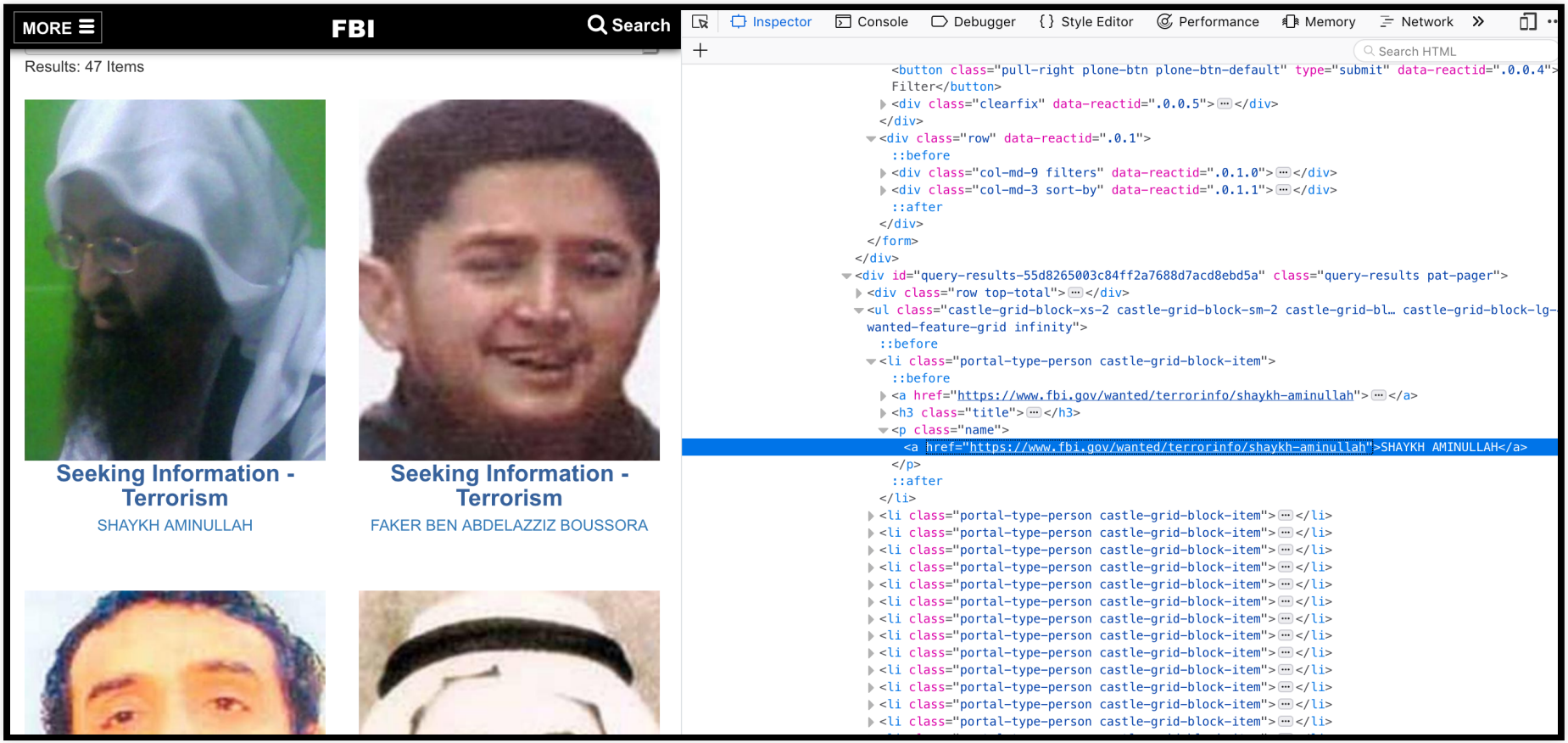

<table>
<tbody>
<tr>
<th>Departments</th>
<th>Location</th>
</tr>
<tr>
<td>Dept. of Security and Crime Science</td>
<td>Division of Psychology and Language Sciences</td>
</tr>
<tr>...</tr>
</tbody>
</table>

Example 1: Missing persons



Example 1: Missing persons

Example 2: FBI most wanted



Webscraping in a nutshell

1. understand the structure of a webpage
2. exploit that structure for web-scraping

RECAP

- **Always: problem first, never the method first!**
- **Method follows problem!**
- APIs give you structured access
- R packages `twitteR` and `tuber`
- HTML structure key to ‘real’ webscraping

Outlook

Next week

- Webscraping on the html structure
- Tutorial: APIs + webscraping in R

Homework

- Getting access to YouTube and Twitter API.

END