

# Advanced Data IO

Data Wrangling in R

# Google Sheets



🖨️ ⚙️ 100% View only

A1 | fx | country

	A	B	C	D	E	F
1	country	continent	year	lifeExp	pop	gdpPercap
2	Australia	Oceania	1952	69.12	8691212	10039.59564
3	Australia	Oceania	1957	70.33	9712569	10949.64959
4	Australia	Oceania	1962	70.93	10794968	12217.22686
5	Australia	Oceania	1967	71.1	11872264	14526.12465
6	Australia	Oceania	1972	71.93	13177000	16788.62948
7	Australia	Oceania	1977	73.49	14074100	18334.19751
8	Australia	Oceania	1982	74.74	15184200	19477.00928
9	Australia	Oceania	1987	76.32	16257249	21888.88903
10	Australia	Oceania	1992	77.56	17481977	23424.76683
11	Australia	Oceania	1997	78.83	18565243	26997.93657
12	Australia	Oceania	2002	80.37	19546792	30687.75473
13	Australia	Oceania	2007	81.235	20434176	34435.36744
14	New Zealand	Oceania	1952	69.39	1994794	10556.57566
15	New Zealand	Oceania	1957	70.26	2229407	12217.22686



Africa

Americas

Asia

Europe

Oceania



Explore

[https://docs.google.com/spreadsheets/d/1U6Cf\\_qEOhiR9AZqTqS3mbMF3zt2db48ZP5v](https://docs.google.com/spreadsheets/d/1U6Cf_qEOhiR9AZqTqS3mbMF3zt2db48ZP5v)

# Reading data with the `googlesheets4` package

First, set up Google credentials

```
library(googlesheets4)
```

```
# Prompts a browser pop-up
gs4_auth()
```

```
# Once set up, you can automate this process by passing your email
gs4_auth(email = "avamariehoffman@gmail.com")
```

## Reading data with the `googlesheets4` package

You can also supply an authorization token directly, but make sure to add any files to your `.gitignore`!

```
library(googledrive)
drive_auth(email= "<email>",
           token = readRDS("google-sheets-token.rds")) # Saved in a file
```

# Reading data with the `googlesheets4` package

Read in using `googlesheets4::read_sheet`

```
sheet_url = paste0("https://docs.google.com/spreadsheets/d/1U6Cf_qEOhiR9",
                   "AZqTqS3mbMF3zt2db48ZP5v3rkrAEJY/edit#gid=780868077")
sheet_dat_1 = read_sheet(sheet_url)
```

Reading from "gapminder"

Range "Africa"

```
head(sheet_dat_1)
```

```
# A tibble: 6 x 6
  country continent year lifeExp      pop gdpPerCap
  <chr>   <chr>    <dbl>   <dbl>     <dbl>    <dbl>
1 Algeria Africa     1952    43.1  9279525    2449.
2 Algeria Africa     1957    45.7  10270856    3014.
3 Algeria Africa     1962    48.3  11000948    2551.
4 Algeria Africa     1967    51.4  12760499    3247.
5 Algeria Africa     1972    54.5  14760787    4183.
6 Algeria Africa     1977    58.0  17152804    4910.
```

# Reading data with the `googlesheets4` package

Specify the sheet name if necessary:

```
sheet_url = paste0("https://docs.google.com/spreadsheets/d/1U6Cf_qEohiR9",
                   "AZqTqS3mbMF3zt2db48ZP5v3rkrAEJY/edit#gid=780868077")
sheet_dat_oceania = read_sheet(sheet_url, sheet = "Oceania")
```

Reading from "gapminder"

Range "'Oceania'"

```
head(sheet_dat_oceania)
```

```
# A tibble: 6 x 6
  country continent year lifeExp      pop gdpPercap
  <chr>    <chr>   <dbl>   <dbl>     <dbl>    <dbl>
1 Australia Oceania  1952     69.1 8691212 10040.
2 Australia Oceania  1957     70.3 9712569 10950.
3 Australia Oceania  1962     70.9 10794968 12217.
4 Australia Oceania  1967     71.1 11872264 14526.
5 Australia Oceania  1972     71.9 13177000 16789.
6 Australia Oceania  1977     73.5 14074100 18334.
```

# Reading data with the `googlesheets4` package

Alternatively, list out the sheet names using `sheet_names()`.

```
sheet_names(sheet_url)
```

```
[1] "Africa"    "Americas"  "Asia"       "Europe"    "Oceania"
```

# Reading data with the `googlesheets4` package

Iterate through the sheet names:

```
gapminder_sheets = sheet_names(sheet_url)

data_list = list()
for(g_sheet in gapminder_sheets) {
  data_list[[g_sheet]] = read_sheet(sheet_url, sheet = g_sheet)
}
```

Reading from "gapminder"

Range "'Africa'"

Reading from "gapminder"

Range "'Americas'"

Reading from "gapminder"

Range "'Asia'"

Reading from "gapminder"

Range "'Europe'"

Reading from "gapminder"

# Reading data with the `googlesheets4` package

Iterate through the sheet names:

```
str(data_list)
```

```
List of 5
$ Africa : tibble [624 × 6] (S3:tbl_df/tbl/data.frame)
..$ country : chr [1:624] "Algeria" "Algeria" "Algeria" "Algeria" ...
..$ continent: chr [1:624] "Africa" "Africa" "Africa" "Africa" ...
..$ year     : num [1:624] 1952 1957 1962 1967 1972 ...
..$ lifeExp   : num [1:624] 43.1 45.7 48.3 51.4 54.5 ...
..$ pop       : num [1:624] 9279525 10270856 11000948 12760499 14760787 ...
..$ gdpPercap: num [1:624] 2449 3014 2551 3247 4183 ...
$ Americas: tibble [300 × 6] (S3:tbl_df/tbl/data.frame)
..$ country : chr [1:300] "Argentina" "Argentina" "Argentina" "Argentina" ...
..$ continent: chr [1:300] "Americas" "Americas" "Americas" "Americas" ...
..$ year     : num [1:300] 1952 1957 1962 1967 1972 ...
..$ lifeExp   : num [1:300] 62.5 64.4 65.1 65.6 67.1 ...
..$ pop       : num [1:300] 17876956 19610538 21283783 22934225 24779799 ...
..$ gdpPercap: num [1:300] 5911 6857 7133 8053 9443 ...
$ Asia    : tibble [396 × 6] (S3:tbl_df/tbl/data.frame)
..$ country : chr [1:396] "Afghanistan" "Afghanistan" "Afghanistan" "Afghanistan" ...
..$ continent: chr [1:396] "Asia" "Asia" "Asia" "Asia" ...
..$ year     : num [1:396] 1952 1957 1962 1967 1972 ...
..$ lifeExp   : num [1:396] 28.8 30.3 32 34 36.1 ...
..$ pop       : num [1:396] 8425333 9240934 10267083 11537966 13079460 ...
..$ gdpPercap: num [1:396] 779 821 853 836 740 ...
$ Europe  : tibble [360 × 6] (S3:tbl_df/tbl/data.frame)
..$ country : chr [1:360] "Albania" "Albania" "Albania" "Albania" ...
```

# Writing data with the `googlesheets4` package

```
sheet_dat_oceania = sheet_dat_oceania %>%
  mutate(lifeExp_days = lifeExp * 365)
sheet_out = gs4_create("Oceania-days",
                      sheets = list(Oceania_days = sheet_dat_oceania))
```

Creating new Sheet: "Oceania-days"

```
# Opens a browser window
gs4_browse(sheet_out)
```

## Append data with the `googlesheets4` package

```
sheet_append(sheet_out, data = sheet_dat_oceania, sheet = "Oceania_days")
```

Writing to "Oceania-days"

Appending 24 row(s) to "Oceania\_days"

# JHU Tidyverse Book

<https://jhubdatascience.org/tidyversecourse/get-data.html#google-sheets>

googlesheets Lab  
<http://sisbid.github.io/Data-Wrangling/labs/advanced-io-lab.Rmd>

---

# JSON: JavaScript Object Notation

## Lists of stuff

## Example [\[edit\]](#)

The following example shows a possible JSON representation describing a person.

```
{  
    "firstName": "John",  
    "lastName": "Smith",  
    "isAlive": true,  
    "age": 27,  
    "address": {  
        "streetAddress": "21 2nd Street",  
        "city": "New York",  
        "state": "NY",  
        "postalCode": "10021-3100"  
    },  
    "phoneNumbers": [  
        {  
            "type": "home",  
            "number": "212 555-1234"  
        },  
        {  
            "type": "office",  
            "number": "646 555-4567"  
        }  
    ],  
    "children": [],  
    "spouse": null  
}
```

# Why JSON matters

The screenshot shows a browser window displaying the GitHub REST API documentation at [docs.github.com/en/rest/reference/search](https://docs.github.com/en/rest/reference/search). The left sidebar contains a list of API endpoints, and the main content area shows a "Default response" example.

**Default response**

Status: 200 OK

```
{  
  "total_count": 7,  
  "incomplete_results": false,  
  "items": [  
    {  
      "name": "classes.js",  
      "path": "src/attributes/classes.js",  
      "sha": "d7212f9dee2dcc18f084d7df8f417b80846ded5a",  
      "url": "https://api.github.com/repos/167174/contents/src/attribut  
      "git_url": "https://api.github.com/repos/167174/git/blobs/d7212  
      "html_url": "https://github.com/jquery/jquery/blob/825ac3773694e0cd23e  
      "repository": {  
        "id": 167174,  
        "node_id": "MDEwOlJlcG9zaXRvcnkxNjcxNzQ=",  
        "name": "jquery",  
        "full_name": "jquery/jquery",  
        "owner": {  
          "login": "jquery",  
          "id": 70142,  
          "node_id": "MDQ6VXNlcjcwMTQy",  
          "avatar_url": "https://0.gravatar.com/avatar/6906f317a4733f4379b06  
          "gravatar_id": "",  
          "url": "https://api.github.com/users/jquery",  
          "html_url": "https://github.com/jquery",  
          "followers_url": "https://api.github.com/users/iauerv/followers".  
        }  
      }  
    }  
  ]  
}
```

<https://docs.github.com/en/rest/reference/search>

```
#install.packages("jsonlite")
library(jsonlite)
jsonData <- fromJSON("https://raw.githubusercontent.com/Biuni/PokemonGO-Pokedex/master/pokedex.json")
head(jsonData)
```

\$pokemon

1	1	001	Bulbasaur	http://www.serebii.net/pokemongo/pokemon/001.pr	im
2	2	002	Ivysaur	http://www.serebii.net/pokemongo/pokemon/002.pr	
3	3	003	Venusaur	http://www.serebii.net/pokemongo/pokemon/003.pr	
4	4	004	Charmander	http://www.serebii.net/pokemongo/pokemon/004.pr	
5	5	005	Charmeleon	http://www.serebii.net/pokemongo/pokemon/005.pr	
6	6	006	Charizard	http://www.serebii.net/pokemongo/pokemon/006.pr	
7	7	007	Squirtle	http://www.serebii.net/pokemongo/pokemon/007.pr	
8	8	008	Wartortle	http://www.serebii.net/pokemongo/pokemon/008.pr	
9	9	009	Blastoise	http://www.serebii.net/pokemongo/pokemon/009.pr	
10	10	010	Caterpie	http://www.serebii.net/pokemongo/pokemon/010.pr	
11	11	011	Metapod	http://www.serebii.net/pokemongo/pokemon/011.pr	
12	12	012	Butterfree	http://www.serebii.net/pokemongo/pokemon/012.pr	
13	13	013	Weedle	http://www.serebii.net/pokemongo/pokemon/013.pr	
14	14	014	Kakuna	http://www.serebii.net/pokemongo/pokemon/014.pr	
15	15	015	Beedrill	http://www.serebii.net/pokemongo/pokemon/015.pr	
16	16	016	Pidgey	http://www.serebii.net/pokemongo/pokemon/016.pr	
17	17	017	Pidgeotto	http://www.serebii.net/pokemongo/pokemon/017.pr	
18	18	018	Pidgeot	http://www.serebii.net/pokemongo/pokemon/018.pr	
19	19	019	Rattata	http://www.serebii.net/pokemongo/pokemon/019.pr	
20	20	020	Raticate	http://www.serebii.net/pokemongo/pokemon/020.pr	
21	21	021	Spearow	http://www.serebii.net/pokemongo/pokemon/021.pr	
22	22	022	Fearow	http://www.serebii.net/pokemongo/pokemon/022.pr	
23	23	023	Ekans	http://www.serebii.net/pokemongo/pokemon/023.pr	18/51
24	24	024	Arbok	http://www.serebii.net/pokemongo/pokemon/024.pr	

# Data frame structure from JSON

```
dim(jsonData$pokemon)  
[1] 151 17  
  
class(jsonData$pokemon$type) # Can be lists  
[1] "list"  
  
jsonData$pokemon$type  
[[1]]  
[1] "Grass"  "Poison"  
  
[[2]]  
[1] "Grass"  "Poison"  
  
[[3]]  
[1] "Grass"  "Poison"  
  
[[4]]  
[1] "Fire"  
  
[[5]]  
[1] "Fire"  
  
[[6]]  
[1] "Fire"   "Flying"
```

# Data frame structure from JSON

```
class(jsonData$pokemon$next_evolution[[1]]) # Or lists of data.frames!
```

```
[1] "data.frame"
```

```
jsonData$pokemon$next_evolution
```

```
[[1]]
```

```
  num      name
1 002  Ivysaur
2 003 Venusaur
```

```
[[2]]
```

```
  num      name
1 003 Venusaur
```

```
[[3]]
```

```
NULL
```

```
[[4]]
```

```
  num      name
1 005 Charmeleon
2 006 Charizard
```

```
[[5]]
```

```
  num      name
1 006 Charizard
```

```
[[6]]
```

# JSON Lab

<http://sisbid.github.io/Data-Wrangling/labs/advanced-io-lab.Rmd>

---

# Web Scraping

# This is data

<http://bowtie-bio.sourceforge.net/recount/>



## » The Datasets

Study	PMID	Species	Number of biological replicates	Number of uniquely aligned reads	ExpressionSet	Count table	Phenotype table	Notes
bodymap	not published, but publicly available <a href="#">here</a>	human	19	2,197,622,796	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	Illumina Human BodyMap 2.0 -- tissue comparison
cheung	<a href="#">20856902</a>	human	41	834,584,950	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	HapMap - CEU
core	<a href="#">19056941</a>	human	2	8,670,342	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	lung fibroblasts
gilad	<a href="#">20009012</a>	human	6	41,356,738	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	liver; males and females
maqc	<a href="#">20167110</a>	human	14 (technical)** 2 (biological)	71,970,164	<a href="#">original pooled</a>	<a href="#">original pooled</a>	<a href="#">original pooled</a>	experiment: MAQC-2
montgomery	<a href="#">20220756</a>	human	60	*886,468,054	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	HapMap - CEU
pickrell	<a href="#">20220758</a>	human	69	*886,468,054	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	HapMap - YRI
sultan	<a href="#">18599741</a>	human	4	6,573,643	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	cell type comparison
wang	<a href="#">18978772</a>	human	22	223,929,919	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	tissue comparison
								control vs

# View the source

Please note that to use the expressionsets below, you will need to install Bioconductor and run the command library(BioBase)

## » The Datasets

Study	PMID	Species	Number of biological replicates	Number of uniquely aligned reads	Expotype	Notes
bodymap	not published, but publicly available <a href="#">here</a>	human	19	2,197,622,796	<a href="#">link</a>	Illumina Human BodyMap 2.0 - - tissue comparison
cheung	<a href="#">20856902</a>	human	41	834,584,950	<a href="#">link</a>	HapMap - CEU
core	<a href="#">19056941</a>	human	2	8,670,342	<a href="#">link</a>	lung fibroblasts
gilad	<a href="#">20009012</a>	human	6	41,356,738	<a href="#">link</a>	liver; males and females
maqc	<a href="#">20167110</a>	human	14 (technical)** 2 (biological)	71,970,164	<a href="#">original pooled</a>	original pooled original pooled experiment: MAQC-2
montgomery	<a href="#">20220756</a>	human	60	*886,468,054	<a href="#">link</a>	<a href="#">link</a> <a href="#">link</a> HapMap - CEU
pickrell	<a href="#">20220758</a>	human	69	*886,468,054	<a href="#">link</a>	<a href="#">link</a> <a href="#">link</a> HapMap - YRI
sultan	<a href="#">18599741</a>	human	4	6,573,643	<a href="#">link</a>	<a href="#">link</a> <a href="#">link</a> cell type comparison
wang	<a href="#">18978772</a>	human	22	223,929,919	<a href="#">link</a>	<a href="#">link</a> <a href="#">link</a> tissue comparison
katz.mouse	<a href="#">21057496</a>	mouse	4	14,368,471	<a href="#">link</a>	<a href="#">link</a> <a href="#">link</a> control vs. CUG-BP1

# What the computer sees

```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4 <script src="sorttable.js" type="text/javascript"></script>
5 <title>ReCount: analysis-ready RNA-seq gene count datasets</title>
6 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
7 <link rel="stylesheet" type="text/css" href="css/style.css" media="screen" />
8 <script type="text/javascript">
9
10 var _gaq = _gaq || [];
11 _gaq.push(['_setAccount', 'UA-26478269-2']);
12 _gaq.push(['_trackPageview']);
13
14 (function() {
15   var ga = document.createElement('script'); ga.type = 'text/javascript'; ga.async = true;
16   ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js';
17   var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s);
18 })();
19
20 </script>
21
22 </head>
23
24 <body class="c20">
25 <div id="wrap">
26   <div id="top">
27     <div class="lefts">
28       <table width="100%" cellpadding="2">
29         <tr><td>
30           <a href=".index.shtml"><h1>ReCount</h1></a>
31           <h2>A multi-experiment resource of analysis-ready RNA-seq gene count datasets</h2>
32           </td><td align="right" valign="middle">
33             <h1><a href="http://www.biostat.jhsph.edu/"></a>&ampnbsp&ampnbsp</h1>
34           </td></tr>
35         </table>
36       </div>
37     </div>
38
39     <div id="subheader">
40       <p><b>There is now <a href="https://jhbiostatistics.shinyapps.io/recount/">a new version of recount</a> that provides processed and summarized express data for nearly 60,000 human RNA-seq samples from the Sequence Read Archive (SRA). The <a href="https://github.com/leekgroup/recount">associated Bioconductor package</a> provides a convenient API for querying, downloading, and analyzing the data. Each processed study consists of meta- and phenot data, the expression levels of genes and their underlying exons and splice junctions, and corresponding genomic annotation. See <a
```

# Ways to see the source

Chrome:

1. right click on page
2. select “view source”

Firefox:

1. right click on page
2. select “view source”

Microsoft Edge:

1. right click on page
2. select “view source”

Safari

1. click on “Safari”
2. select “Preferences”
3. go to “Advanced”
4. check “Show Develop menu in menu bar”
5. click on “Develop”
6. select “show page source”
7. alternatively to 5./6., right click on page and select “view source”

<https://github.com/simonmunzert/rscraping-jsm-2016/blob/c04fd91fec711df65c838e07723125155a7f2cda/02-scraping-with-rvest.r>

# Inspect element

Not Secure | bowtie-bio.sourceforge.net/recount/

Please note that to use the expressionsets below, you will need to install Bioculator and run the command library (biocore).

✖ The Datasets

Study	PMID	Species	Number of biological replicates	Number of uniquely aligned reads	Expression	Back	Forward	Reload	Notes
maqc	<a href="#">20167110</a>	human	14 (technical)** 2 (biological)	71,970,164	original pooled	<a href="#">Save As...</a>	<a href="#">Print...</a>	<a href="#">Cast...</a>	experiment: MAQC-2
modencodefly	<a href="#">21179090</a>	fly	147 (technical)** 30 (biological)	2,278,788,557	original pooled	<a href="#">Translate to English</a>	<a href="#">View Page Source</a>	<a href="#">View Frame Source</a>	developmental time course
modencodeworm	<a href="#">19181841</a>	worm	46	1,451,119,823	<a href="#">link</a>	<a href="#">Reload Frame</a>	<a href="#">Inspect</a>	<a href="#">Speech ►</a>	developmental time course
hammer	<a href="#">20452967</a>	rat	8	158,178,477	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	experimental vs. control at 2 time points
nagalakshmi	<a href="#">18451266</a>	yeast	4	7,688,602	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	priming technique comparison
bottomly	<a href="#">21455293</a>	mouse	21	343,445,340	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	2 inbred mouse strains
yang	<a href="#">20363980</a>	mouse	1	27,883,862	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	hybrid cell line, X always inactive
trapnell	<a href="#">20436464</a>	mouse	4	111,376,152	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	time course
mortazavi	<a href="#">18516045</a>	mouse	3	61,732,881	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	<a href="#">link</a>	tissue comparison

# Copy XPath

The screenshot shows a web browser window with the URL [Not Secure | bowtie-bio.sourceforge.net/recount/](http://bowtie-bio.sourceforge.net/recount/). A context menu is open over a table row with the ID `recounttab`. The table has columns for Study, PMID, Species, Number of biological replicates, Number of uniquely aligned reads, ExpressionSet, Count table, Phenotype table, and Notes. The rows contain data for bodymap, cheung, core, and gilad. The context menu options include: Add attribute, Edit as HTML, Delete element, Copy, Hide element, Force state, Break on, Expand recursively, Collapse children, Scroll into view, Focus, Store as global variable, Speech, Cut element, Copy element, Paste element, Copy outerHTML, Copy selector, Copy JS path, Copy styles, Copy XPath (which is highlighted), and Copy full XPath.

Study	PMID	Species	Number of biological replicates	Number of uniquely aligned reads	ExpressionSet	Count table	Phenotype table	Notes
bodymap	not published, but publicly available here	human	19	2,197,622,796	link	link	link	Illumina Human BodyMap 2.0 - tissue comparison
cheung	20856902	human	41	834,584,950	link	link	link	HapMap - CEU
core	19056941	human	2	8,670,342	link	link	link	lung fibroblasts
gilad	2000			41,356,738	link	link	link	liver; males and females

Brief description of experiment:  
<br >  
<br >  
"  
Please note that to use the Bioconductor library, run the command "  
<a href="http://www.bioconductor.org" style="color: blue; text-decoration: none">http://www.bioconductor.org</a>" and run the command "  
<tt>library(Bioconductor)</tt>".  
<h3>The Datasets</h3>

Copy element  
Copy element  
Paste element  
Copy outerHTML  
Copy selector  
Copy JS path  
Copy styles  
Copy XPath  
Copy full XPath  
Store as global variable  
Speech

Styles Computed Event Listeners DOM Breakpoints >  
Filter :hover .cls +  
element.style {}  
\* {  
padding: 0;  
margin: 0;  
}  
div {  
display: block;  
}  
Inherited from div#leftside  
#leftside {  
padding-left: 8px;  
color: #555;  
}

## Use SelectorGadget

<https://cran.r-project.org/web/packages/rvest/vignettes/selectorgadget.html>

---

# rvest package

```
recount_url = "http://bowtie-bio.sourceforge.net/recount/"
# install.packages("rvest")
library(rvest)
htmlfile = read_html(recount_url)

nds = html_nodes(htmlfile,
xpath='//*[@id="recounttab"]/table')
dat = html_table(nds)
dat = as.data.frame(dat)
head(dat)
```

	X1	X2	X3
1	Study	PMID	Species
2	bodymap not published, but publicly available here		human
3	cheung	20856902	human
4	core	19056941	human
5	gilad	20009012	human
6	magc	20167110	human
	X4	X5	
1	Number of biological replicates	Number of uniquely aligned reads	
2	19	2,197,622,796	
3	41	834,584,950	
4	2	8,670,342	
5	6	41,356,738	
6	14 (technical)** 2 (biological)	71,970,164	
	X6	X7	X8
1	ExpressionSet link	Count table link	Phenotype table link
2			

# Little cleanup

```
colnames(dat) = as.character(dat[1,])
dat = dat[-1,]
head(dat)
```

	Study	PMID	Species
2	bodymap not published, but publicly available here		human
3	cheung	20856902	human
4	core	19056941	human
5	gilad	20009012	human
6	maqc	20167110	human
7	montgomery	20220756	human
	Number of biological replicates	Number of uniquely aligned reads	
2	19	2,197,622,796	
3	41	834,584,950	
4	2	8,670,342	
5	6	41,356,738	
6	14 (technical)** 2 (biological)	71,970,164	
7	60	*886,468,054	
	ExpressionSet	Count table	Phenotype table
2	link	link	link
3	link	link	link
4	link	link	link
5	link	link	link
6	original pooled	original pooled	original pooled
7	link	link	link
		Notes	
2	Illumina Human BodyMap 2.0 -- tissue comparison		
3		HapMap - CEU	

<http://motherboard.vice.com/read/70000-okcupid-users-just-had-their-data-published>

MOTHERBOARD  
TECH BY VICE

# 70,000 OkCupid Users Just Had Their Data Published

Just because data is sort-of public, doesn't mean that it's ethical to collect en masse.

By [Joseph Cox](#)

<https://www.theguardian.com/science/2012/may/23/text-mining-research-tool-forbidden>

theguardian.com/science/2012/may/23/text-mining-research-tool-forbidden

Peer review and scientific publishing

## Text mining: what do publishers have against this hi-tech research tool?

Researchers push for end to publishers' default ban on computer scanning of tens of thousands of papers to find links between genes and diseases



Alok Jha, Science correspondent

Wed 23 May 2012 11.27 EDT

[f](#) [t](#) [e](#)

3 62



▲ A DNA sequence is examined: text mining allows links to be discovered that would never have been noticed during manual research. Photograph: Simon Dolbner/Alamy

# APIs

# Application Programming Interfaces

<https://developers.facebook.com/>

The screenshot shows the Facebook Developers homepage with a navigation bar at the top. The main section is titled "DEVELOPER TOOLS" and features six cards, each representing a different tool or program:

- Messenger**: Build lasting customer relationships through conversation. [Learn more](#)
- Instagram**: Create tools for businesses, creators, and people to enhance the Instagram experience. [Learn more](#)
- Business Tools**: Build and scale your business across the Facebook family of apps. [Learn more](#)
- Open Source**
- Artificial Intelligence**
- AR/VR**

# In biology too!

<http://www.ncbi.nlm.nih.gov/books/NBK25501/>

The screenshot shows a web browser displaying the "Entrez Programming Utilities Help" page. The URL in the address bar is <http://www.ncbi.nlm.nih.gov/books/NBK25501/>. The page content includes a sidebar with links to "Entrez Programming Utilities Help", "NCBI Help Manual", and the "National Center for Biotechnology Information". The main content area features a search bar and sections on "Introduction to the E-utilities", "Contents", and "Recent Activity". The right side of the page contains a sidebar with links to "Views", "PubReader", "Print View", "Cite this Page", "PDF version of this title (1.8M)", "Other titles in this collection" (listing "NCBI Help Manual"), "Related information" (listing "NLM Catalog"), and "Recent Activity" (listing several recent publications).

Entrez Programming Utilities Help

Bethesda (MD): [National Center for Biotechnology Information \(US\)](#); 2010-.  
[Copyright and Permissions](#)

Search this book

Introduction to the E-utilities

- [YouTube E-utilities Introduction](#)
- Please see the [Release Notes](#) for details and changes.

The Entrez Programming Utilities (E-utilities) are a set of eight server-side programs that provide a stable interface into the Entrez query and database system at the National Center for Biotechnology Information (NCBI). The E-utilities use a fixed URL syntax that translates a standard set of input parameters into the values necessary for various NCBI software components to search for and retrieve the requested data. The E-utilities are therefore the structured interface to the Entrez system, which currently includes 38 databases covering a variety of biomedical data, including nucleotide and protein sequences, gene records, three-dimensional molecular structures, and the biomedical literature.

Contents

E-utilities Quick Start

Created: December 12, 2008; Last Update: October 24, 2018.

Expand All [Collapse All](#)

Views

PubReader

Print View

Cite this Page

PDF version of this title (1.8M)

Other titles in this collection

NCBI Help Manual

Related information

NLM Catalog

Recent Activity

Turn Off Clear

- Entrez Programming Utilities Help
- A fully automated pipeline for brain structure segmentation in multiple sclerosis...
- Validation of Accelerometer Wear and Nonwear Time Classification Algorithm
- The Evolution of Earned, Transparent, and Quantifiable Faculty Salary Compensation...
- Evaluating the Impact of Intensity Normalization on MR Image

# Step 0: Did someone do this already

<https://ropensci.org/packages/>

The screenshot shows a web browser displaying the [ropensci.org/packages/](https://ropensci.org/packages/) page. The page features a search bar at the top with the placeholder text "Search by: name, maintainer, or keyword". Below the search bar is a table with columns: NAME, MAINTAINER, DESCRIPTION, and STATUS. The STATUS column includes icons for CRAN and GitHub, along with a speech bubble icon for each package. The packages listed are:

NAME	MAINTAINER	DESCRIPTION	STATUS
gert	Jeroen Ooms	Simple Git Client for R	CRAN
tic	Kirill Müller	CI-Agnostic Workflow Definitions	CRAN
rotl	Francois Michonneau	Interface to the 'Open Tree of Life' API	CRAN
colpluz	Scott Chamberlain	Catalogue of Life Plus Client	CRAN
crul	Scott Chamberlain	HTTP Client	CRAN
c14bazAAR	Clemens Schmid	Download and Prepare C14 Dates from Different Source Databases	CRAN
stplanr	Robin Lovelace	Sustainable Transport Planning	CRAN
webchem	Tamás Stirling	Chemical Information from the Web	CRAN

# Do it yourself: read the Docs

<https://docs.github.com/en/rest>

The screenshot shows a web browser displaying the GitHub REST API documentation at <https://docs.github.com/en/rest>. The page has a dark blue header with the GitHub logo and the text "GitHub Docs". On the left, there's a sidebar with a dark blue background containing navigation links for "All products", "REST API", "OVERVIEW", "Resources in the REST API", "Media types", "Other authentication methods", "Troubleshooting", "API previews", "Libraries", "Endpoints available for GitHub Apps", and "REFERENCE" (which includes "Actions", "Activity", "Apps", "Billing", "Checks", "Code Scanning", and "Codes of conduct"). The main content area has a light blue header with a message: "We've unified all of GitHub's product documentation in one place! Check out the content for [REST API](#), [GraphQL API](#), and [Developers](#). Learn more on the [GitHub blog](#)." Below this, there's a search bar with the placeholder "Search topics, products..." and a language selector set to "English". The main content starts with the heading "GitHub REST API" and a paragraph: "You can use the GitHub REST API to create calls to get the data you need to integrate with GitHub." There are two sections below: "REST API overview" (describing resources, libraries, previews, and troubleshooting) and "Reference" (providing reference documentation for resources). A button in the top right corner says "Article version: GitHub.com".

# Read the docs

<https://docs.github.com/en/rest/reference/repos>

The screenshot shows a web browser displaying the GitHub REST API documentation for repositories. The URL in the address bar is <https://docs.github.com/en/rest/reference/repos>. The page title is "Repositories". On the left, there is a sidebar with a dark blue background containing a list of links: Code Scanning, Codes of conduct, Emojis, Gists, Git database, Gitignore, Interactions, Issues, Licenses, Markdown, Meta, Migrations, OAuth Authorizations, Organizations, Projects, Pulls, Rate limit, Reactions, Repositories, SCIM, Search, Teams, and Users. The main content area has a white background. At the top, it shows the REST API / Reference / Repositories path. Below that is a large section titled "Repositories". Under this, there is a sub-section titled "List organization repositories" with the description "Lists repositories for the specified organization." It includes a "GET /orgs/{org}/repos" endpoint. Below this are sections for "Parameters" and "Responses". The "Parameters" table has columns for Name, Type, In, and Description. It lists three parameters: "accept" (string, header), "org" (string, path), and "type" (string, query). The "Responses" table lists two possible responses: "200 OK" and "404 Not Found". To the right of the main content, there is a sidebar titled "In this article" which lists various repository-related endpoints. At the bottom right of the page is a "Article version: GitHub.com" dropdown menu.

Code Scanning  
Codes of conduct  
Emojis  
Gists  
Git database  
Gitignore  
Interactions  
Issues  
Licenses  
Markdown  
Meta  
Migrations  
OAuth Authorizations  
Organizations  
Projects  
Pulls  
Rate limit  
Reactions  
Repositories  
SCIM  
Search  
Teams  
Users

REST API / Reference / Repositories

# Repositories

## List organization repositories

Lists repositories for the specified organization.

GET /orgs/{org}/repos

### Parameters

Name	Type	In	Description
accept	string	header	Setting to <code>application/vnd.github.v3+json</code> is recommended See preview notices.
org	string	path	
type	string	query	Specifies the types of repositories you want returned. Can be one of <code>all</code> , <code>public</code> , <code>private</code> , <code>forks</code> , <code>sources</code> , <code>member</code> , <code>internal</code> . Default: <code>all</code> . If your organization is associated with an enterprise account using GitHub

### Responses

Status	Description
200 OK	The requested repositories were successfully listed.
404 Not Found	The organization or repository was not found.

In this article

- List organization repositories
- Create an organization repository
- Get a repository
- Update a repository
- Delete a repository
- Enable automated security fixes
- Disable automated security fixes
- List repository contributors
- Create a repository dispatch event
- List repository languages
- List repository tags
- List repository teams
- Get all repository topics
- Replace all repository topics
- Transfer a repository
- Check if vulnerability alerts are enabled for a repository
- Enable vulnerability alerts
- Disable vulnerability alerts
- Create a repository using a template

# Read the docs

<https://docs.github.com/en/rest/reference/rate-limit>

The screenshot shows a web browser displaying the GitHub REST API Rate Limit documentation at <https://docs.github.com/en/rest/reference/rate-limit>. The page has a dark blue sidebar on the left containing a list of links to other API documentation sections. The main content area has a light gray background. At the top of the content area is a navigation bar with icons for star, R, apple, magnifying glass, and other common web functions. The main title 'Rate limit' is centered above a paragraph of text. To the right of the main content, there is a sidebar titled 'In this article' with two links: 'Understanding your rate limit status' and 'Get rate limit status for the authenticated user'. The main text discusses the REST API overview documentation's rate limit rules and how to check current status using the Rate Limit API.

Code Scanning  
Codes of conduct  
Emojis  
Gists  
Git database  
Gitignore  
Interactions  
Issues  
Licenses  
Markdown  
Meta  
Migrations  
OAuth Authorizations  
Organizations  
Projects  
Pulls  
Rate limit  
Reactions  
Repositories  
SCIM  
Search  
Teams  
Users

## Rate limit

The REST API overview documentation describes the [rate limit rules](#). You can check your current rate limit status at any time using the Rate Limit API described below.

### Understanding your rate limit status

The Search API has a [custom rate limit](#), separate from the rate limit governing the rest of the REST API. The GraphQL API also has a [custom rate limit](#) that is separate from and calculated differently than rate limits in the REST API.

For these reasons, the Rate Limit API response categorizes your rate limit. Under `resources`, you'll see four objects:

- The `core` object provides your rate limit status for all non-search-related resources in the REST API.
- The `search` object provides your rate limit status for the [Search API](#).
- The `graphql` object provides your rate limit status for the [GraphQL API](#).
- The `integration_manifest` object provides your rate limit status for the [GitHub App Manifest code conversion](#) endpoint.

For more information on the headers and values in the rate limit response, see ["Rate limiting."](#)

# Read the docs

## Example

Suppose you want to find the definition of the `addClass` function inside `jQuery`. Your query would look something like this:

```
https://api.github.com/search/code?q=addClass+in:file+language:js+repo:jquery/jquery
```

Here, we're searching for the keyword `addClass` within a file's contents. We're making sure that we're only looking in files where the language is JavaScript. And we're scoping the search to the `repo:jquery/jquery` repository.

## A dissected example

[https://api.github.com/search/repositories?  
q=created:2014-08-13+language:r+-user:cran](https://api.github.com/search/repositories?q=created:2014-08-13+language:r+-user:cran)

---

## The base URL

[https://api.github.com/search/repositories?  
q=created:2014-08-13+language:r+-user:cran](https://api.github.com/search/repositories?q=created:2014-08-13+language:r+-user:cran)

## The Path: Search repositories

[https://api.github.com/search/repositories?  
q=created:2014-08-13+language:r+-user:cran](https://api.github.com/search/repositories?q=created:2014-08-13+language:r+-user:cran)

Create a query - pass the `q` parameter

[`https://api.github.com/search/repositories?`](https://api.github.com/search/repositories?)  
`q=created:2014-08-13+language:r+-user:cran`

Date repo was created

[https://api.github.com/search/repositories?  
q=created:2014-08-13+language:r+-user:cran](https://api.github.com/search/repositories?q=created:2014-08-13+language:r+-user:cran)

Language repo is in

[https://api.github.com/search/repositories?  
q=created:2014-08-13+language:r+-user:cran](https://api.github.com/search/repositories?q=created:2014-08-13+language:r+-user:cran)

Ignore repos from “cran”

[https://api.github.com/search/repositories?  
q=created:2014-08-13+language:r+-user:cran](https://api.github.com/search/repositories?q=created:2014-08-13+language:r+-user:cran)

```
#install.packages("httr")
library(httr)

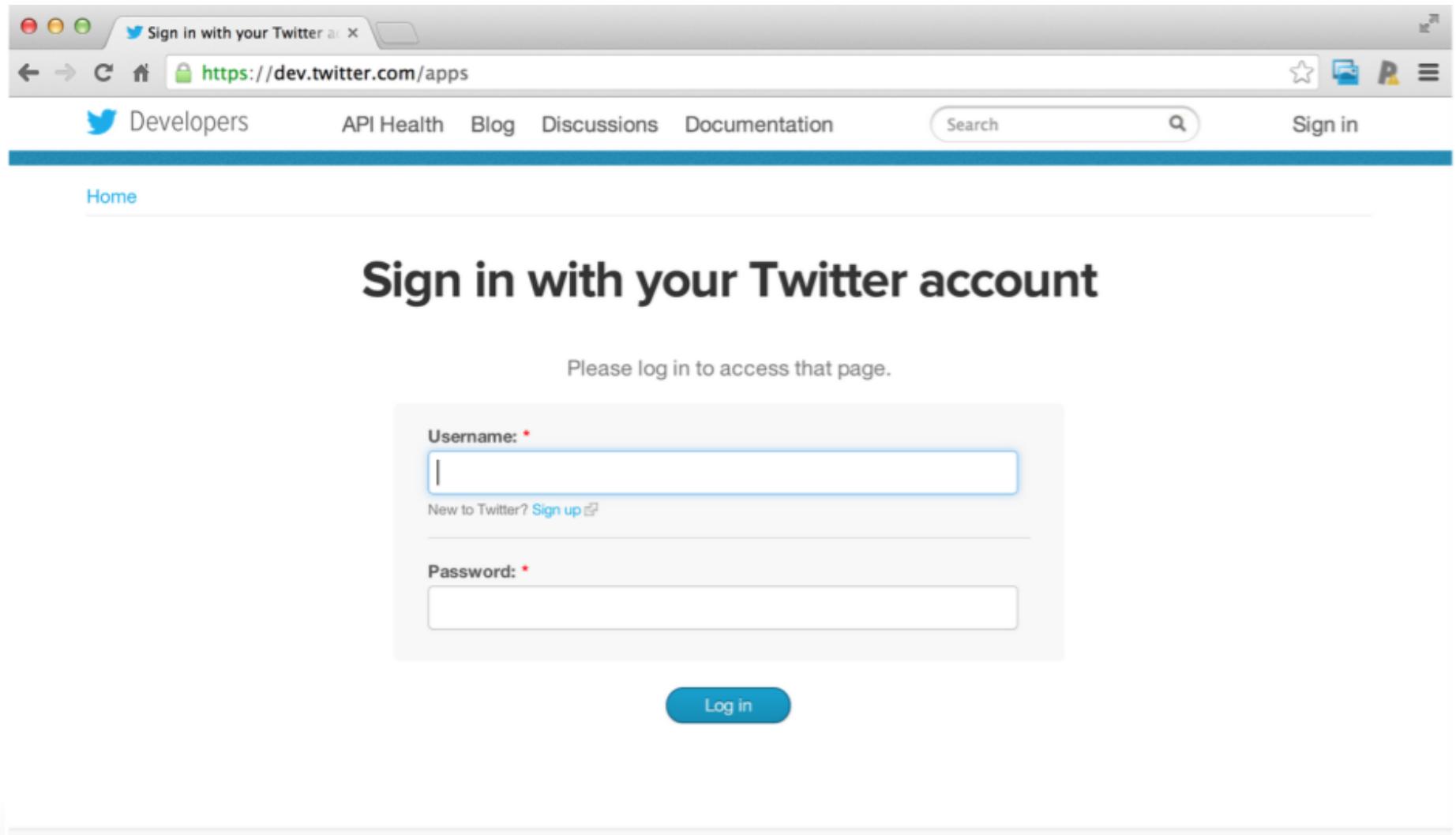
query_url = paste0("https://api.github.com/", "search/repositories",
                  "?q=created:2014-08-13", "+language:r", "+-user:cran")

req = GET(query_url)
names(content(req))
```

```
[1] "total_count"      "incomplete_results" "items"
```

# Not all APIs are “open”

<https://apps.twitter.com/>



A screenshot of a web browser window displaying the Twitter API sign-in page. The URL in the address bar is <https://dev.twitter.com/apps>. The page title is "Sign in with your Twitter account". The content area includes fields for "Username:" and "Password:", both marked with a red asterisk indicating they are required. A "Log in" button is at the bottom. The browser interface shows standard navigation buttons, a search bar, and a "Sign in" link in the top right.

(see also `twitteR` package)

```
myapp = oauth_app("twitter",
                  key="yourConsumerKeyHere", secret="yourConsumerSecretHere")
sig = sign_oauth1.0(myapp,
                     token = "yourTokenHere",
                     token_secret = "yourTokenSecretHere")
homeTL = GET("https://api.twitter.com/1.1/statuses/home_timeline.json", sig)

# But you can get cool data
json1 = content(homeTL)
json2 = jsonlite::fromJSON(toJSON(json1))
json2[1,1:4]
```

	created_at	id	id_str
1	Mon Jan 13 05:18:04 +0000 2014	4.225984e+17	422598398940684288

1 Now that P. Norvig's regex golf IPython notebook hit Slashdot, let's see if