

Lecture 21: Oct 26, 2018

Web APIs

- *HTTP(S) Requests*
- *Web + APIs*
- *httr with REST*
- *JSON*
- *Resources*

James Balamuta
STAT 385 @ UIUC



Announcements

- **hw07** is due **Friday, Nov 2nd, 2018 at 6:00 PM**
- **Group project** member choice phase **has begun!**
 - <https://github.com/stat385-fa2018/disc/issues/75>
- **Quiz 10** covers Week 9 contents @ [CBTF](#).
 - Window: Oct 30th - Nov 1st
 - Sign up: <https://cbtf.engr.illinois.edu/sched>
- Want to review your homework or quiz grades?
Schedule an appointment.

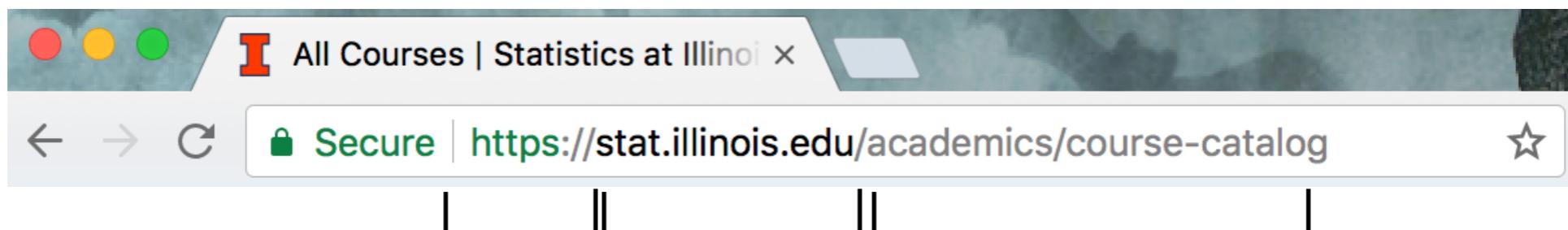
Lecture Objectives

- **Summarize** how an HTTP/HTTPS request is formed
- **Communicate** with a Web API
- **Understand** the semi-structured form of JSON
- **Extract** data from Web API Response

HTTP(S) Requests

HTTP(S) Request

... HyperText Transfer Protocol (**S**ecure) ...



Protocol/Scheme	Domain	Path
Communication medium	Web Address	Location of Resource
for request		



All Courses

[Collapse All Course Levels](#) [Expand All Course Levels](#)

► 100 Level Courses

ACADEMICS

Graduate Programs

Undergraduate Program

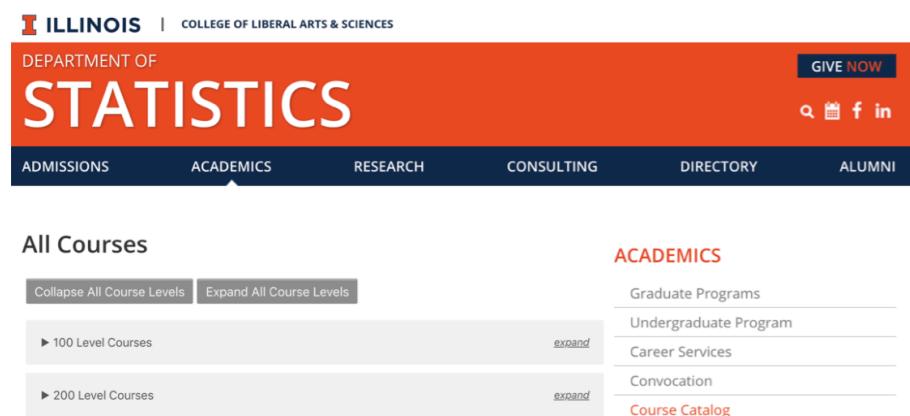
Career Services

Convocation

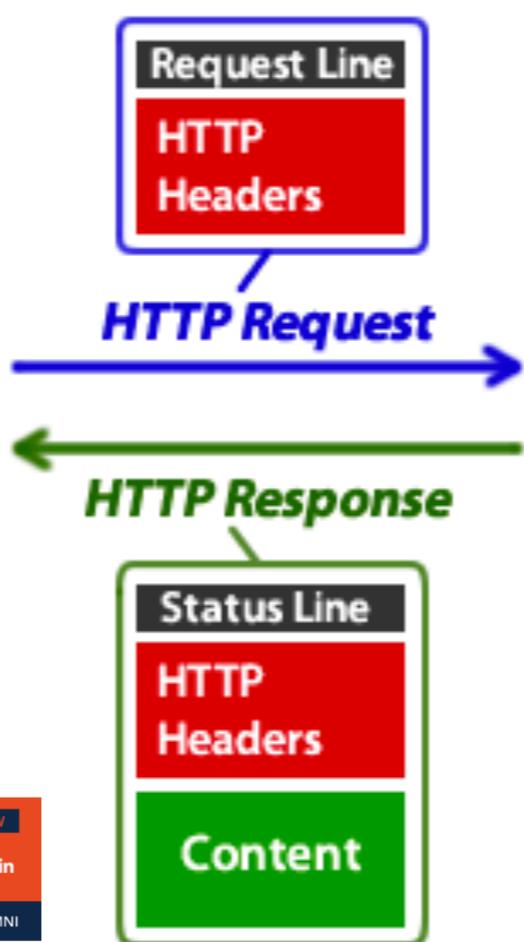
Course Catalog

► 200 Level Courses

Hello stat.illinois.edu,
I'd like to see the
course catalog



Hello stat.illinois.edu,
thank you!



Hello person's web
browser, let me pull
that up...

[Source](#)

Name
▼ academics
careers-statistics.htm
convocation.htm
course-catalog.htm
grad-programs.htm
registration.htm
statistics-seminars.htm
statistics-tutoring.htm
student-groups.htm
undergrad-programs.htm
▶ admissions
▶ alumni
▶ assets
▶ consulting
▶ directory
▶ research

Hello person's web
browser, I found it.
Here you go!

URI*

```
scheme://user:pass@example.org:8182/path/to/file;type=foo?name=val#fragm  
\\_ / \\_ / \\_ / \\_ / \\_ / \\_ / \\_ / \\_ / \\_ /  
| | | | | | | | | | | | | | | | | |  
scheme user info hostname port path filename param query fragment  
\\_ / (matrix)  
authority
```

Example: <http://www.google.se/search?q=openehr&ie=utf-8&oe=utf-8>

Source

- * **URLs** (**Uniform Resource Locators**) or web links are actual a subset of **URIs**. They capture the **scheme** required to access the resource.

Downloading a File

... retrieving a file using a web browser ...

Student Enrollment by Curriculum and Class Level

This report uses the final enrollment statistics from the official tenth day of on-campus terms. (Preliminary enrollment figures may be posted before the tenth day.) Reports for terms before Fall 2004 are sorted by college and curriculum code, with degree-granting and advising departments listed for each curriculum. Beginning in Fall 2004, reports are sorted by college, department code, and major code. Only students taking at least one on-campus, credit-bearing class are included in these reports. The following categories of students are excluded: auditors (students taking **only** non-credit classes); students taking **only** extramural or off-campus classes; Medical Scholars taking **no** on-campus, non-MSP classes. (Note: Illini Center MBA students are included in these statistics.)

NOTE: As of Spring 2017, these enrollment reports no longer separate students into "on-campus" and "extramural" categories. All students enrolled in classes on the 10th day are reported.

Student system	Fall - HTML	Spring - HTML	Summer 1 - HTML	Summer 2 - HTML	Fall - Excel	Spring - Excel	Summer - Excel
Banner: All Students	Fall, 2017	Spring, 2018 <small>NEW</small> Spring, 2017	Summer, 2017		Fall, 2017	Spring, 2018 <small>NEW</small>	Summer, 2017
Banner: On-Campus Students	Fall, 2016 Fall, 2015 Fall, 2014 Fall, 2013	Spring, 2016 Spring, 2015 Spring, 2014 Spring, 2013	Summer, 2016 Summer, 2015 Summer, 2014 Summer, 2013	Starting Summer, 2005, summer 1 and summer 2 terms were consolidated	Fall, 2016 Fall, 2015 Fall, 2014 Fall, 2013	Open Link in New Tab Open Link in New Window Open Link in Incognito Window Save Link As... Copy Link Address	

Website: <http://www.dmi.illinois.edu/stuenr/index.htm#class>

CSV File: <http://www.dmi.illinois.edu/stuenr/class/enrfa17.xls>

Downloading a File

... retrieving a file online using R ...

Uniform Resource Locator Save Destination
Website link to the file Place to store file on computer

```
download.file(url = <address>, destfile = <save-to>)
```

Downloading a File

... retrieving a file online from R ...

```
# URL of file to retrieve  
url = "http://www.dmi.illinois.edu/stuenr/class/enrfa17.xls"
```

```
# Save this file as...  
destfile = "enrfa17.xls"
```

```
# Download the file  
download.file(url = url, destfile = destfile)
```

```
# Read the file into R  
enrollfa17 = readxl::read_excel(x = destfile)
```

How can we:

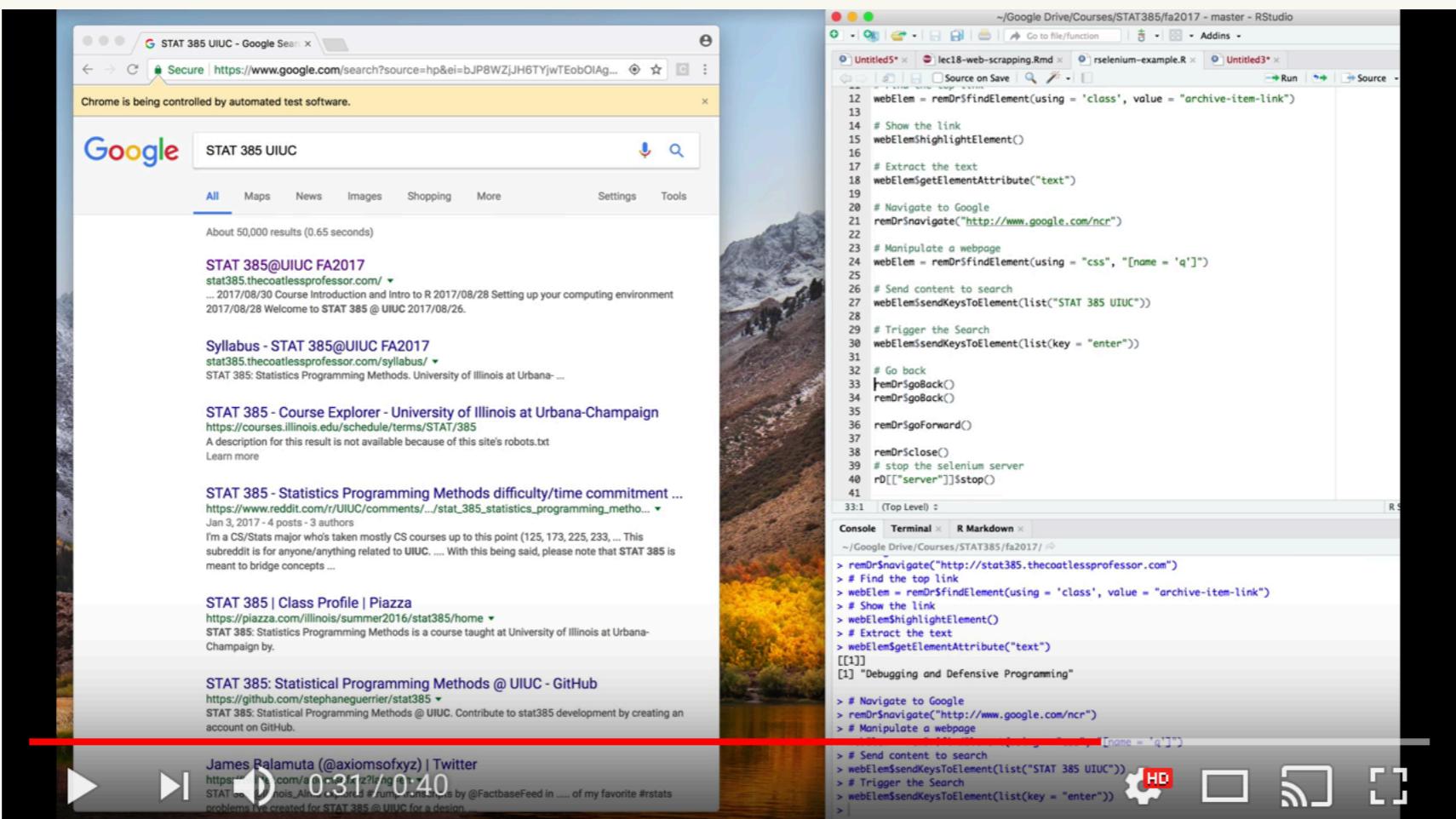
- 1. obtain a list of files; and**
- 2. download them automatically?**

More on this next time when we cover web scraping ...

Web + APIs

Definition:

Web scraping (web harvesting or web data extraction) is a computer software technique of extracting information from websites. [Source](#)



<https://www.youtube.com/watch?v=Vt6f8A35-1w>

Why not just Web Scrape?

... reasons to not scrape ...

1

- **Websites change...**

Scraping is not robust.

2

- **Legality of scraping...**

Check the Terms of Service (TOS)

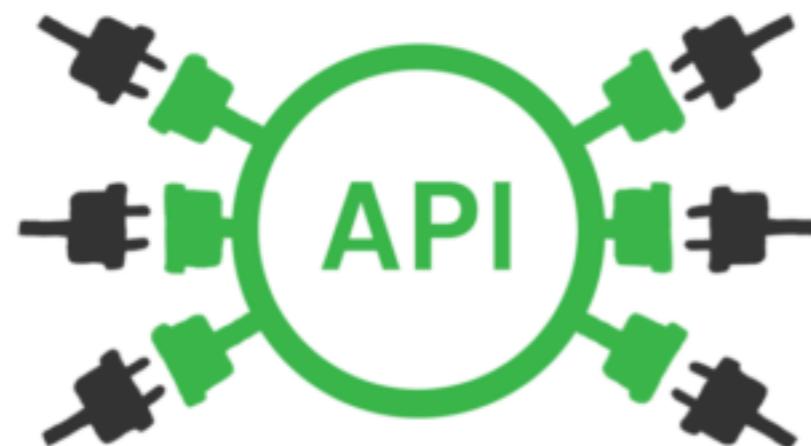
3

- **Lack of Documentation**

Not as structured as an API

Definition:

Application programming interfaces (APIs) describe different ways to interact with software.



Interface

Function Name
Actual name of the function that can be called e.g. add(1, 2)

```
add = function(a, b) {  
    summed = a + b  
    return (summed)  
}
```

Statements in between {} that are run when the function is called

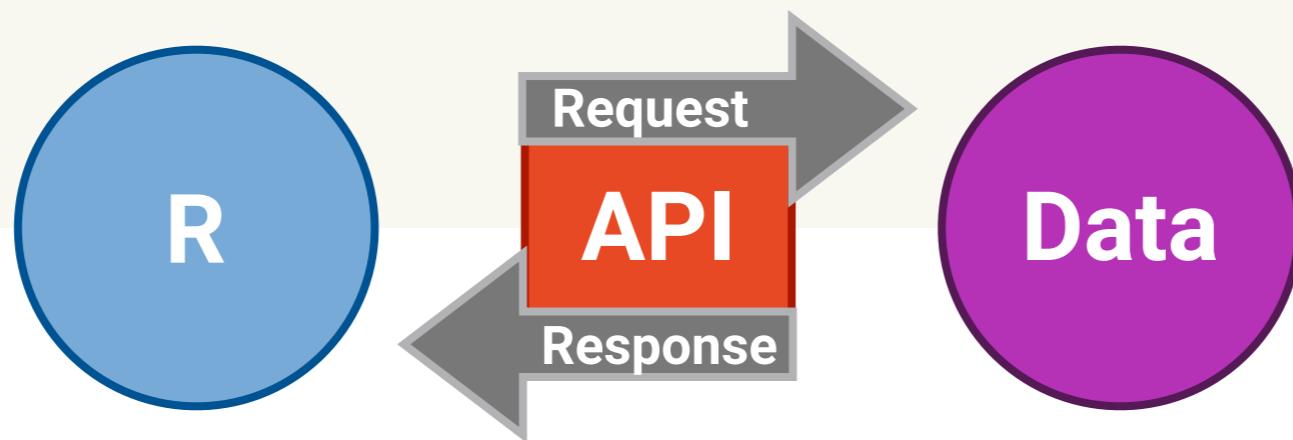
Parameters
Variables that receive expressions that can be used in the function's body

Return Value
Result made available from running the body statements

[Source](#)

Definition:

Web APIs refer to an *Application Programming Interface* (API) for a web server or web browser.



Twitter Developer | Use cases | Products | Docs | More | Apply | Sign in

Search all documentation...

Docs

Basics

- Accounts and users
- Tweets
- Direct Messages
- Media
- Trends
- Geo
- Ads
- Metrics
- Publisher tools & SDKs
- Developer utilities
- API reference index

Below are some of the key areas where developers typically engage with the Twitter platform. For additional information, use the navigation to the left, or the search box, to explore our documentation.

Ads API
Programmatically create and manage your ad campaigns
[Getting started](#)

Filter realtime Tweets
Get only the Tweets you need in realtime
[Filter realtime Tweets](#)

Search Tweets
Use the Search API to gather Tweets
[Search Tweets](#)

Direct Message API
Build personalized customer experiences with our Direct Message platform
[Build private experiences](#)

<https://developer.twitter.com/en/docs>

GitHub Developer | API Docs | Blog | Early Access | Versions | Search...

REST API v3

Overview

This describes the resources that make up the official GitHub REST API v3. If you have any problems or requests please contact GitHub support.

- i. Current version
- ii. Schema
- iii. Authentication
- iv. Parameters
- v. Root endpoint
- vi. GraphQL global relay IDs
- vii. Client errors
- viii. HTTP redirects
- ix. HTTP verbs
- x. Hypermedia
- xi. Pagination
- xii. Rate limiting
- xiii. User agent required
- xiv. Conditional requests
- xv. Cross origin resource sharing
- xvi. JSON-P callbacks
- xvii. Timezones

Reference | Guides | Libraries

Overview | Media Types | OAuth Authorizations API | Other Authentication Methods | Troubleshooting | Pre-release Program | API Previews | Versions | Activity | Gists | Git Data | GitHub Apps | Issues | Migration | Miscellaneous | Organizations

<https://developer.github.com/v3/>

Definition:

Endpoints indicate where resources are on the server that can be requested by users.

`https://api.website.com/v1/resource/`

first API version

Endpoint



URI for APIs

... setting up ways to connect ...

Base URI

https://api.github.com



Endpoint

/user/repos
/repos/:owner/:repo
/orgs/:org/repos

Variables



<https://developer.github.com/v3/repos/#repositories>

https://api.github.com/**user/repos**

https://api.github.com/**repos/:owner/:repo**

https://api.github.com/**orgs/:org/repos**

httr with REST

REST

Representation **S**tate **T**ransfer

defines how networked resources are accessed

HTTP Information

... when visiting a website ...



CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

Open Chrome's Web Developer Tools

- Windows: **Ctrl + Shift + I**
- macOS: **Command + Option + I**

The screenshot shows the Network tab in Chrome DevTools. A red arrow points to the 'Network' tab itself. Another red arrow points to the list of network requests, where 'cran.r-project.org' is highlighted. The details panel on the right shows a request for 'https://cran.r-project.org/' with various headers and response details. Three large arrows point from the text 'Response', 'Request', and 'HTTP Request' to the 'Response' section of the DevTools, the 'Request' section, and the 'Request Headers' section respectively.

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages. **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux](#)
- [Download R for \(Mac\) OS X](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2018-07-02, Feather Spray) [R-3.5.1.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension [packages](#)

Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.

Request URL: https://cran.r-project.org/
Request Method: GET
Status Code: 200 OK
Remote Address: 137.208.57.37:443
Referrer Policy: no-referrer-when-downgrade

Response Headers

Accept-Ranges: bytes
Connection: Keep-Alive
Content-Encoding: gzip
Content-Length: 458
Content-Type: text/html
Date: Tue, 17 Jul 2018 17:56:34 GMT
ETag: "352-52743448b67b6-gzip"
Keep-Alive: timeout=15, max=99
Last-Modified: Sat, 19 Dec 2015 17:05:49 GMT
Server: Apache/2.4.10 (Debian)
Vary: Accept-Encoding

Request Headers

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate, br

Response

Request

HTTP Response

... breaking down the 3 components ...



← **Status Code**

Summary of request state

← **Header**

Metadata on body content
think addressing an envelope

← **Body:**

Content of request
think the words of the letter

HTTP Verbs

... navigating APIs ...

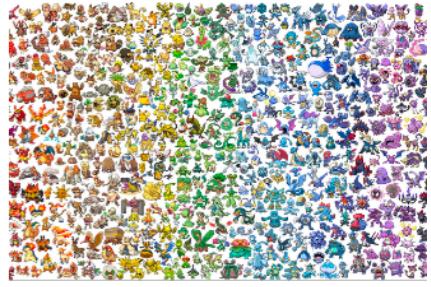
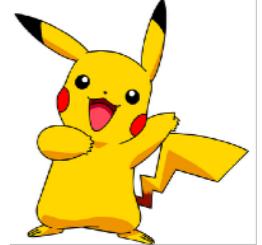
Verb	Description
POST	Create a new resource given data
GET	Retrieve a resource from collection.
PUT	Update an existing resource with new data.
DELETE	Delete a resource from collection.

* These verbs are collectively known as "[CRUD](#)"

** There are [additional verbs](#) that are less frequently used:
CONNECT, TRACE, OPTIONS, HEAD, PATCH

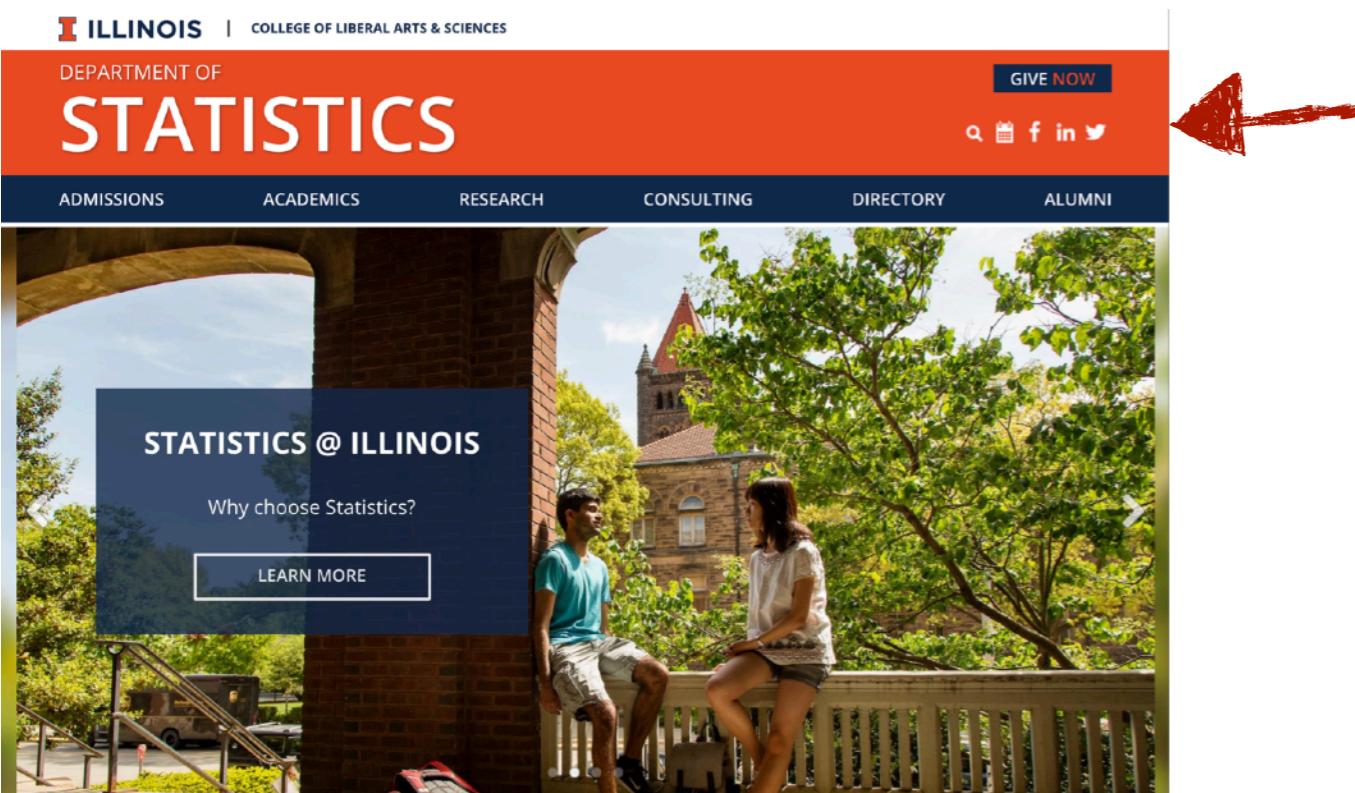
API and Verbs

... how an API relates to HTTP Verbiage...

Verb	Path	Description	
GET	/v2/pokemon/	Retrieve all Pokemon	
GET	/v2/pokemon/{id}	Retrieve a specific Pokemon	
POST	/v2/pokemon/	Add a new pokemon	
PUT	/v2/pokemon/{id}	Update an existing Pokemon	
DELETE	/v2/pokemon/{id}	Delete an existing Pokemon	

httr

... making *HTTP* requests in *R* ...



A status of **200** means everything went smoothly

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

```
# Form a GET request to obtain the
STAT department website
web_page =
```

```
GET("https://stat.illinois.edu/")
```

```
# Response [https://stat.illinois.edu/]
```

```
# Date: 2018-07-10 18:37
```

```
# Status: 200
```

```
# Content-Type: text/html;
charset=UTF-8
```

```
# Size: 60.3 kB
```

```
# ...
```

```
# Check status of HTTP GET
request
```

```
status_code(web_page)
```

```
# [1] 200
```

HTTP Status Code Breakdown

... classes of the status codes ...

Code	Meaning
1xx	Informational
2xx	Successful
3xx	Redirection
4xx	Client Error
5xx	Server Error

* Status codes may *not* always be correct. May need info in header.

** [Breakdown of Common Status Codes](#)

HTTP STATUS DOGS

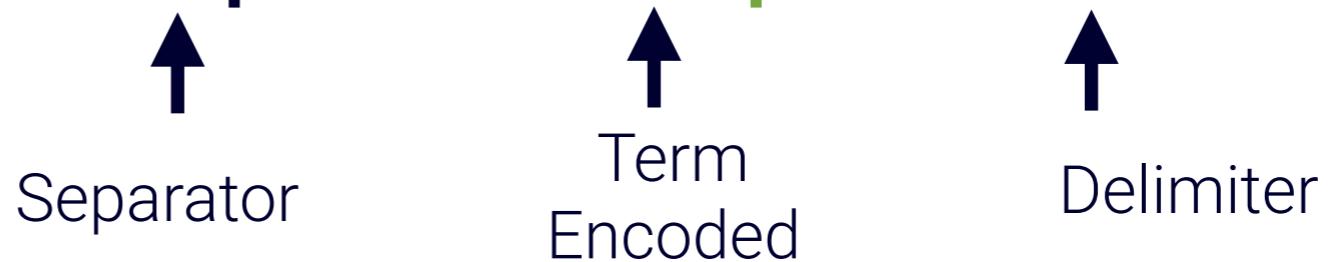
Common
HTTP STATUS CODES



Supplying Query Terms

... writing a query term ...

`https://www.google.com/search?q=uiuc+stat+department&sourceid=chrome`



More in-depth formatting: https://en.wikipedia.org/wiki/Query_string

```
# Provide a query term  
query_terms = list(q = "uiuc stat department")
```

```
# Form a GET request to search google  
google_search = GET("https://www.google.com/search",  
                    query = query_terms)
```

Retrieving Header Contents

... metadata on the request ...

The screenshot shows a search results page for "uiuc stat department" and the Network tab of the developer tools. The search results page lists various links related to the University of Illinois Statistics department. The Network tab shows a list of requests, with one specific request highlighted. The highlighted request is for a Google search result, with its details shown in the Headers section:

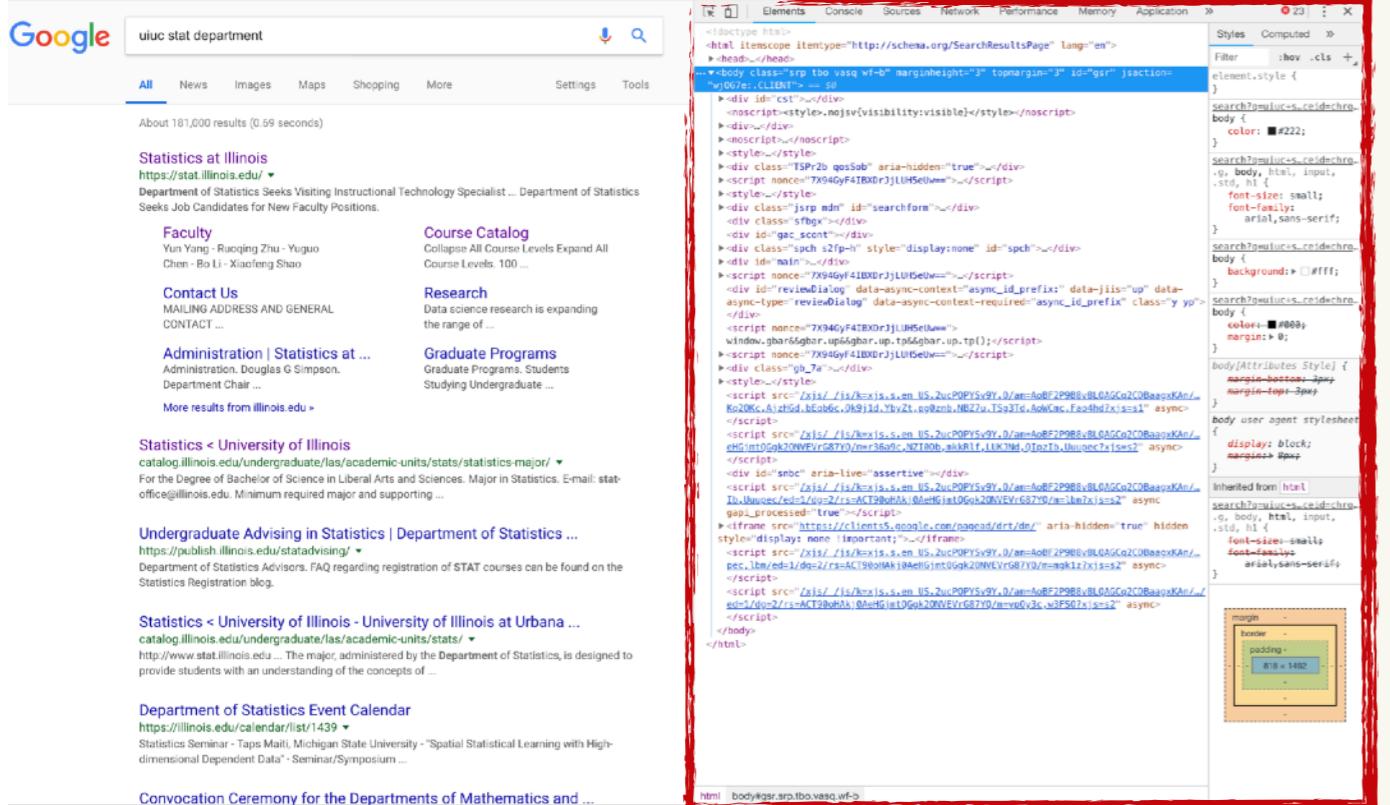
Name	Headers
gen_204?atyp=&ct=webfont...	Request URL: https://notifications.google.com/u/0/widget?sourceid=1&hl=en&origin=https%3A%2F%2Fwww.google.com&usegap=1&jsh=m%3B%2F.%2Fscs%2Fabc-static%2F_%2F%2Fk%3Dgapi.gapi.en.Bst0mEhp0_0_0%2Fr%3Dj%2Fd%3D1%2Fr%30Alp0o-oqKd0JtcyAi5YyA6ph5kp_0j%2F%3D..._features_
gen_204?atyp=&ei=Q0vT...	Request Method: GET
widget?sourceid=1&hl=en&ori...	Status Code: 200
m=wmwg8b	Remote Address: 216.58.192.238:443
m=OJUrb	Referrer Policy: origin
m=Cy0we,EFQ78c,F3ZVPC...	
api.js	
m=NTM2ac,SKR2Dw	
m=ZxDnqc,wHhDv	
cb=gapi.loaded_0	
m=FCpbqb,WhJNk,_latency	
log?format=json&offset=true	
gen_204?atyp=&ei=Q0vT...	

The Headers section contains the following content:

```
alt-svc: quic=":443"; ma=2592000; v="44,43,39,35"
cache-control: private, max-age=21600
content-encoding: gzip
content-security-policy: script-src 'report-sample' 'nonce-uaTUvK95LZt1M1wz7WrguKBeI' 'unsafe-inline' 'unsafe-eval';object-src 'none';base-uri 'self';report-uri /_Notifications0gbUi/cspreport;worker-src 'self'
content-security-policy: script-src 'nonce-uaTUvK95LZt1M1wz7WrguKBeI' 'self' 'unsafe-eval' https://apis.google.com https://ssl.gstatic.com https://www.gstatic.com http://www.gstatic.com/report-uri /_Notifications0gbUi/cspreport;frame-ancestors https://www.gstatic.com Go to Google Home
content-security-policy-report-only: script-src 'report-sample' 'unsafe-inline' https://htp://report-uri /_Notifications0gbUi/cspreport
content-type: text/html; charset=utf-8
date: Fri, 26 Oct 2018 17:13:45 GMT
expires: Fri, 26 Oct 2018 17:13:45 GMT
server: ESF
set-cookie: SIDCC=A0tHo-GCPHhYIsAcUdGQ74_NG6kD5bRlhH10tuAYcGrwApQYlq6A-z-pVFN4t:G7ZR_01K0ZH0kg; expires=Thu, 24-Jan-2019 17:13:45 GMT; path=/; domain=.google.com; priority=high
status: 200
strict-transport-security: max-age=31536000
x-content-type-options: nosniff
x-xss-protection: 1; mode=block
```

```
# Retrieve the header
headers(google_search)
# $date
# [1] "Tue, 10 Jul 2018 12:17:39
GMT"
# $expires
# [1] "-1"
# `$cache-control`
# [1] "private, max-age=0"
# `$content-type`
# [1] "text/html;
charset=ISO-8859-1"
# ...
```

Retrieving Body Contents ... grabbing web data ...



```
# Retrieve body of request as text
content(google_search, "text")
# [1] "<!doctype html><html
itemscope="" itemtype="http://
schema.org/SearchResultsPage"
lang="en"><head><meta
content="text/html; charset=UTF-8"
http-equiv="Content-Type"><meta
content="/images/branding/googleleg/
1x/googleleg_standard_color_128dp.png"
itemprop="image"><link href="/
images/branding/product/ico/
googleleg_lodp.ico" rel="shortcut
icon"><title>uiuc stat department -
Google Search</title>
```

```
# Default content is either:
# xml document or raw vector
content(google_search)
# {xml_document}
# <html itemtype="http://schema.org/
SearchResultsPage" lang="en">
# ...
```

Encoding/Decoding Special Characters

Decoded	Encoded
space	%20
!	%21
"	%22
#	%23
\$	%24
%	%25
&	%26
	%27
(%28
)	%29
*	%2A
+	%2B
,	%2C
-	%2D
.	%2E
/	%2F
[%5B
\	%5C
]	%5D
^	%5E
<	%3C
=	%3D
>	%3E
?	%3F
@	%40

Encoding a URL

```
URLencode("https://www.google.com/search?q=URL encoded")  
# [1] "https://www.google.com/search?q=URL %20encoded"
```

Decoding the URL

```
URLdecode("https://www.google.com/search?q=URL %20encoded")  
# [1] "https://www.google.com/search?q=URL encoded"
```

Encoding Special Characters

```
URLencode("+ * - \ ^ # $ % &")  
# [1] "+%20*%20-%20%20%5E%20#%20$%20%25%20&"
```

Decode Special Characters

```
URLdecode(  
  URLencode("+ * - \ ^ # $ % &")  
)  
# [1] "+ * - ^ # $ % &"
```

Your Turn

Send a **GET** request to <https://httpbin.org/get>

Check:

Status

Header

Body

What happens if a **POST** request is sent to

<https://httpbin.org/post?>

JSON

Previously

Structures of Data

... how data is shaped ...

Structured*

Rectangular

~5 - 10%

Semistructured**

key: value

~5 - 10%

title: "Untitled"

author: "JJB"

date: "1/27/2018"

output: html_document

Unstructured***

?????????

~80 - 90%

Pinky said,
"Gee, Brain. What are we
going to do tonight?"
The Brain replied, "The
same thing we do every
night, Pinky. Try to take
over the world."

* Typical form for scientific experiments and company databases

** RMarkdown Document Properties (YAML), JavaScript Object Notation (JSON), XML

*** Pure text documents, images, social media posts, and so on. No visible relationship.

Previously

What kind of data is this?

```
{  
  "id": "file",  
  "value": "File Menu",  
  "showuser": true,  
  "dropdown": {  
    "menuitem": [  
      {"label": "New Document", "onclick": "create_document()"},  
      {"label": "Save Document", "onclick": "save_document()"},  
      {"label": "Open Document", "onclick": "open_document()"},  
      {"label": "Close Document", "onclick": "close_document()"}  
}
```

JSON

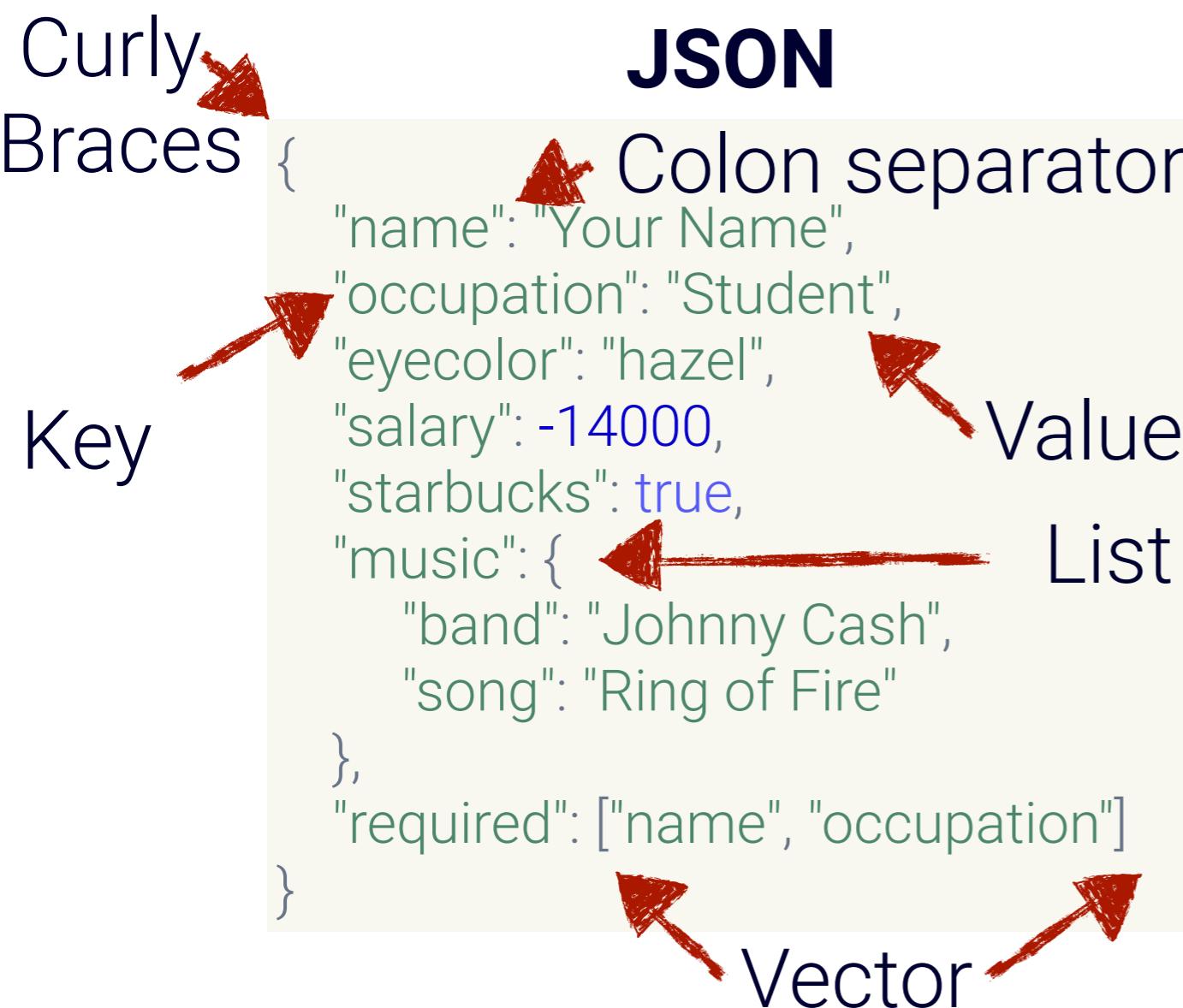
... JavaScript Object Notation ...

most common response type from a Web API

```
{  
  "name": "Your Name",  
  "occupation": "Student",  
  "eyecolor": "hazel",  
  "salary": -14000,  
  "starbucks": true,  
  "music": {  
    "band": "Johnny Cash",  
    "song": "Ring of Fire"  
  },  
  "required": ["name", "occupation"]  
}
```

Behind JSON

... semi-structure format ...



R

```
my_data = list(
  "name" = "Your Name",
  "occupation" = "Student",
  "eyecolor" = "hazel",
  "salary" = -14000,
  "starbucks" = true,
  "music" = list(
    "band" = "Johnny Cash",
    "song" = "Ring of Fire"
  ),
  "required" = c("name", "occupation")
)
```

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

```
fromJSON('{
  "name": "Your Name",
  "occupation": "Student",
  "eyecolor": "hazel",
  "salary": -14000,
  "starbucks": true,
  "music": {
    "band": "Johnny Cash",
    "song": "Ring of Fire"
  },
  "required": ["name", "occupation"]
}')
```

Parsing JSON

Querying GitHub's API

... connecting to GitHub ...

```
# Specify base URL  
base_url = "https://api.github.com"  
  
# Specify user  
gh_user = "tidyverse"  
  
# Create a request for information  
endpoint_resource = paste0("/users/", gh_user ,"/repos")  
  
# Make a url  
url = paste0(base_url, endpoint_resource)  
  
# Form a GET request to obtain a list of repos on GitHub  
gh_get_repos = GET(url)
```

Manipulating Results

... viewing data in an API response ...

```
# Acquire results in a list
gh_list_repos = content(gh_get_repos)
# May generate a list OR a data frame
gh_data_repos = fromJSON(content(gh_get_repos, "text"))
```

```
# Check underlying structure
str(gh_data_repos)
# 'data.frame': 28 obs. of 72 variables:
# $ id      : int 72109608 23932217 86504302...
# ...
```

```
# Directly verify if a data frame
is.data.frame(gh_data_repos)
# [1] TRUE
```

Your Turn

Retrieve the tag information for the **RcppCore/Rcpp** repository based on the details provided in:

<https://developer.github.com/v3/repos/#list-tags>

Your Turn

Obtain the list of houses from the Ice and Fire API

[HOME](#) [ABOUT](#) [DOCUMENTATION](#) [SPONSOR](#)

An API of Ice And Fire

All the data from the universe of Ice And Fire you've ever wanted!

Try it out!

<https://anapioficeandfire.com/api/characters/583>

Psst, need a hint? Try [/books/1](#), [/characters/583](#) or [/houses/378](#)

Response:

```
{  
    "url": "https://anapioficeandfire.com/api/characters/583",  
    "name": "Jon Snow",  
    "gender": "Male",  
    "culture": "Northmen",  
    "born": "In 283 AC",  
    "died": "",  
    "titles": [  
        "Lord Commander of the Night's Watch"  
    ],  
    "aliases": [  
        "Lord Snow",  
        "Ned Stark's Bastard",  
        "The Snow of Winterfell",  
        "The Night's Watch Commander",  
        "The King in the North",  
        "The King in the North (alias)",  
        "The King in the North (title)",  
        "The King in the North (name)",  
        "The King in the North (culture)",  
        "The King in the North (born)",  
        "The King in the North (died)",  
        "The King in the North (titles)",  
        "The King in the North (aliases)",  
        "The King in the North (url)"  
    ]  
}
```

<https://anapioficeandfire.com/Documentation#houses>

Resources

Public APIs

... APIs for all !!!

 README.md

Public APIs

A collective list of free APIs for use in web development.

A public API for this project can be found [here](#) - thanks to [DigitalOcean](#) for helping us provide this service!

For information on contributing to this project, please see the [contributing guide](#).

Please note a passing build status indicates all listed APIs are available since the last update. A failing build status indicates that 1 or more services may be unavailable at the moment.

Index

- [Animals](#)
- [Anime](#)
- [Anti-Malware](#)
- [Art & Design](#)
- [Books](#)
- [Business](#)
- [Calendar](#)
- [Cloud Storage & File Sharing](#)
- [Continuous Integration](#)
- [Cryptocurrency](#)
- [Currency Exchange](#)
- [Data Validation](#)
- [Development](#)

<https://github.com/toddmotto/public-apis>

REST API Tutorial

... Crux of the API Request ...

REST API Tutorial Home Tutorials ▾ HTTP Status Codes Resources

Using HTTP Methods for RESTful Services

The HTTP verbs comprise a major portion of our “uniform interface” constraint and provide us the action counterpart to the noun-based resource. The primary or most-commonly-used HTTP verbs (or methods, as they are properly called) are POST, GET, PUT, PATCH, and DELETE. These correspond to create, read, update, and delete (or CRUD) operations, respectively. There are a number of other verbs, too, but are utilized less frequently. Of those less-frequent methods, OPTIONS and HEAD are used more often than others.

Below is a table summarizing recommended return values of the primary HTTP methods in combination with the resource URIs:

HTTP Verb	CRUD	Entire Collection (e.g. /customers)	Specific Item (e.g. /customers/{id})
POST	Create	201 (Created), 'Location' header with link to /customers/{id} containing new ID.	404 (Not Found), 409 (Conflict) if resource already exists..
GET	Read	200 (OK), list of customers. Use pagination, sorting and filtering to navigate big lists.	200 (OK), single customer. 404 (Not Found), if ID not found or invalid.
PUT	Update/Replace	405 (Method Not Allowed), unless you want to update/replace every resource in the entire collection.	200 (OK) or 204 (No Content). 404 (Not Found), if ID not found or invalid.
PATCH	Update/Modify	405 (Method Not Allowed), unless you want to modify the collection itself.	200 (OK) or 204 (No Content). 404 (Not Found), if ID not found or invalid.
DELETE	Delete	405 (Method Not Allowed), unless you want to delete the whole collection—not often desirable.	200 (OK). 404 (Not Found), if ID not found or invalid.

Below is a more-detailed discussion of the main HTTP methods. Click on a tab for more information about the desired HTTP method.

[POST](#) [GET](#) [PUT](#) [PATCH](#) [DELETE](#)

The POST verb is most-often utilized to **create** new resources. In particular, it's used to create subordinate resources. That is, subordinate to some other (e.g. parent) resource. In other words, when creating a new resource, POST to the parent and the service takes care of associating the new resource with the parent, assigning an ID (new resource URI), etc.

On successful creation, return HTTP status 201, returning a Location header with a link to the newly-created resource with the 201 HTTP status.

POST is neither safe nor idempotent. It is therefore recommended for non-idempotent resource requests. Making two identical POST requests will most-likely result in two resources containing the same information.

Examples:

- POST <http://www.example.com/customers>
- POST <http://www.example.com/customers/12345/orders>

<http://www.restapitutorial.com/lessons/httpmethods.html>

Recap

- **HTTP(S)**
 - Communicating over the internet.
- **Web + APIs**
 - APIs provide structured interaction with software.
 - Web APIs are more stable than web pages
- **httr with REST**
 - Connect with a REST API using HTTP Verbs
 - Formatting of HTTP Response is: Status, Header, Body
- **JSON**
 - Semi-structured output based on key-value form
 - Common Web API response that is converted to an *R* list

This work is licensed under the
Creative Commons
Attribution-NonCommercial-
ShareAlike 4.0 International
License

