Grokking HTTP

Ben Ramsey

Grok?

grok • /ˈɡrɒk/

To grok is to intimately and completely share the same reality or line of thinking with another physical or conceptual entity. Author Robert A. Heinlein coined the term in his best-selling 1961 book Stranger in a Strange Land. In Heinlein's view, grokking is the intermingling of intelligence that necessarily affects both the observer and the observed.

—from Wikipedia, http://en.wikipedia.org/wiki/Grok

The basics

What is HTTP?

Hypertext Transfer Protocol:

Formally defined by RFC 2616, et al.

hypertext:

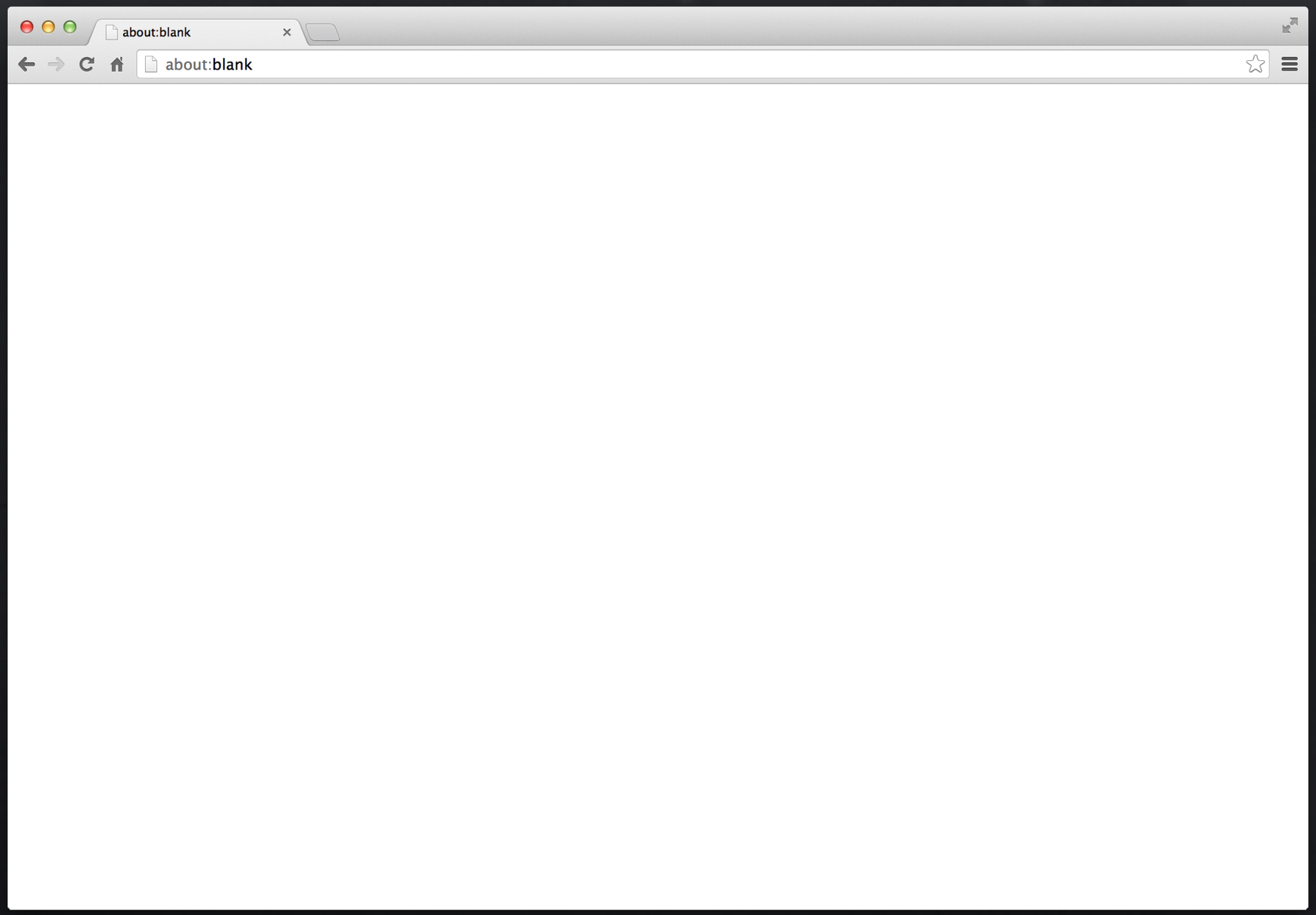
A multi-linear set of objects, building a network by using ***logical links*** (the so-called hyperlinks) between the nodes (e.g. text or words).

protocol:

A set of rules and regulations that define how data is transmitted across a network.

HTTP is a set of rules for *transferring hypertext* across the Internet.

It forms the basis of *everything* we do *on the Web.*


about:blank
about:blank

```
GET / HTTP/1.1
Host: benramsey.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.7; rv:15.0)
    Gecko/20100101 Firefox/15.0.1
Accept: text/html,application/xhtml+xml, application/xml;q=0.9,
    */*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: ...
Pragma: no-cache
Cache-Control: no-cache
```

Ben Ramsey

benramsey.com

Ben Ramsey

Blog | Archives | About Me

Search

MAR 22ND, 2014

Will Encryption Catch on With Keybase?

Email is not secure. Let's stop fooling ourselves. Just because I use Gmail, and I'm using it over HTTPS does not mean that the email I send or receive is encrypted while being transmitted outside of Google's network. ~~Inside Google's network, even, the contents are not encrypted.~~[1] So, why do we keep sending sensitive information through email, and why do our banks and mortgage brokers and HR departments keep asking for us to send our Social Security number, bank accounts, and other private details through email?

Is it because we are oblivious, naïve, or do we just not care? I suspect it's a little of all three, but mainly it's because encryption is hard, and the difficulty barrier keeps us from adopting it.

The alpha launch of Keybase has got me excited. It uses the public-key cryptography (a.k.a. PGP/GnuPG) model to identify yourself, prove your identity, and allow others to vouch for your identity. I hope it paves the way to making encryption easier for us all, from the technologically-skilled to the technologically-challenged.

Recent Posts

- Will Encryption Catch on With Keybase?
- Dates Are Hard
- The Fall of PEAR and the Rise of Composer
- Wild Garlic
- Contributing to PHP Core

Follow @ramsey 5,159 followers

GitHub Repos

uuid
A PHP 5.3+ library for generating RFC 4122 version 1, 3, 4, and 5 universally unique identifiers (UUID).

array_column
Provides functionality for array_column() to projects using PHP earlier than version 5.5.

```
HTTP/1.1 200 OK
Date: Tue, 09 Oct 2012 21:38:43 GMT
Server: Apache
Last-Modified: Fri, 05 Oct 2012 10:18:18 GMT
Accept-Ranges: bytes
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 4155
Content-Type: text/html

<!DOCTYPE html>
<!--[if IEMobile 7 ]><html class="no-js iem7"><![endif]-->
<!--[if lt IE 9]><html class="no-js lte-ie8"><![endif]-->
<!--[if (gt IE 8)|(gt IEMobile 7)|!(IEMobile)|!(IE)]><!--><html class="no-js"
lang="en"><!--<![endif]-->
<head>
  <meta charset="utf-8">
  <title>Ben Ramsey</title>
  <meta name="author" content="Ben Ramsey">

...
```

How do I see all that?

Ben Ramsey

benramsey.com

Ben Ramsey

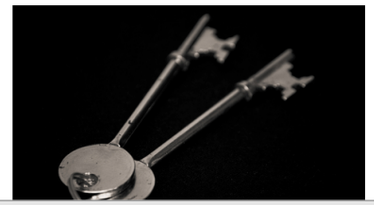
Blog | Archives | About Me

Search

MAR 22ND, 2014

Will Encryption Catch on With Keybase?

Email is not secure. Let's stop fooling ourselves. Just because I use Gmail, and I'm using it over HTTPS does not mean that the email I send or receive is encrypted while being transmitted outside of Google's network. ~~Inside Google's network, even, the contents are not encrypted.~~[1] So, why do we keep sending



Recent Posts

- Will Encryption Catch on With Keybase?
- Dates Are Hard
- The Fall of PEAR and the Rise of Composer
- Wild Garlic
- Contributing to PHP Core

Elements | Network | Sources | Timeline | Profiles | Resources | Audits | Console

Preserve log

Name / Path

- benramsey.com
- screen.css /stylesheets
- modernizr-2.0.js /javascripts
- jquery.min.js ajax.googleapis.com/ajax/
- octopress.js /javascripts

37 requests | 1.5 MB transferre...

Headers | Preview | Response | Cookies | Timing

Remote Address: 75.119.205.123:80
Request URL: http://benramsey.com/
Request Method: GET
Status Code: 200 OK
Request Headers view source
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8
Cache-Control: no-cache
Connection: keep-alive
Cookie: __qca=P0-343654428-1376246898326; __utma=146594599.918676432.1370927938.1398306053.1398442532.115; __utmb=146594599.2.10.1398442532; __utmc=146594599; __utmz=146594599.1395583227.98.25.utmcsr=feedly.com|utmccn=(referral)|utmcmd=referral|utmcct=/
Host: benramsey.com

Favorite tools

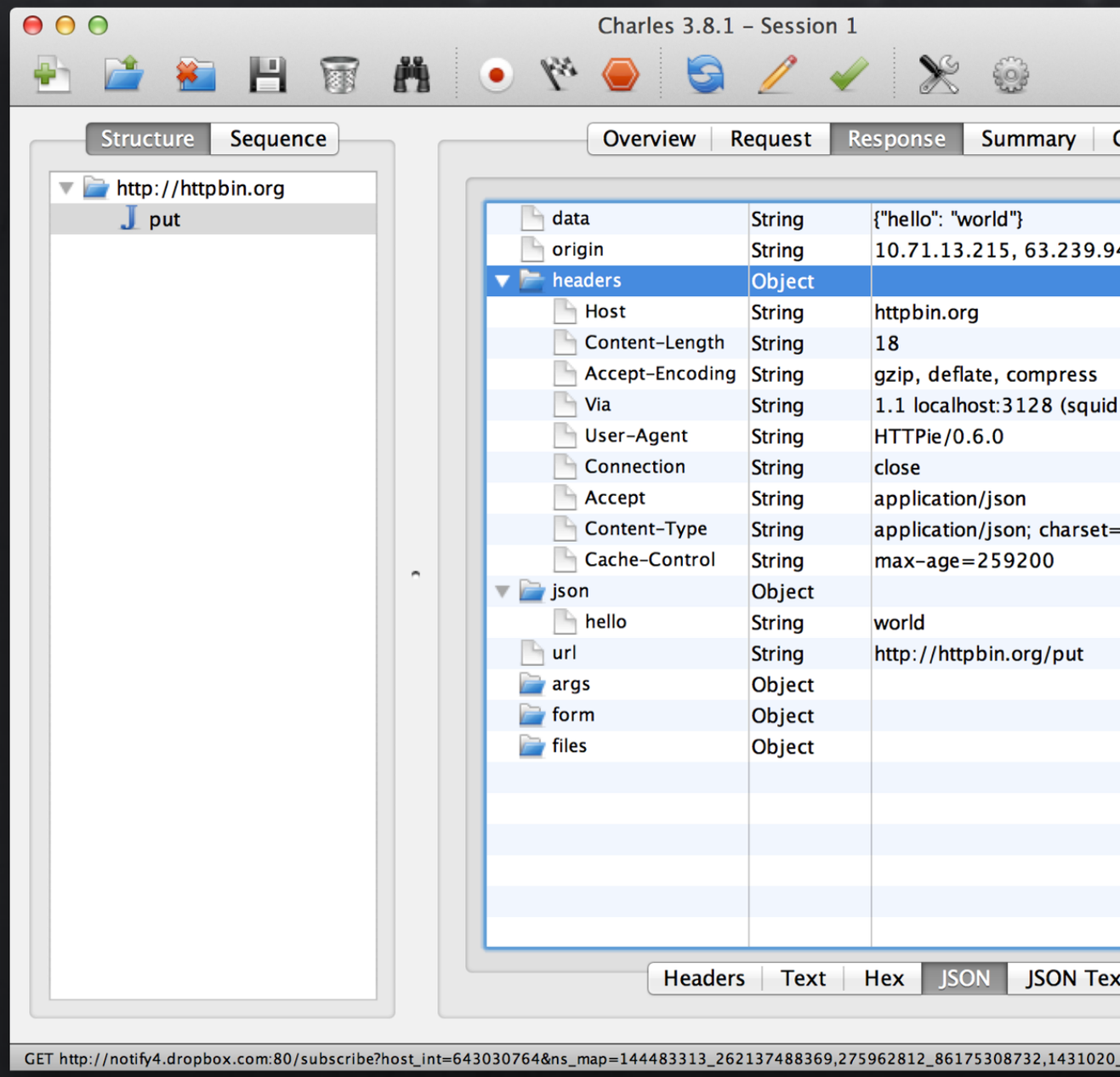
Charles

I cannot recommend this enough!

charlesproxy.com

Perfect for debugging Ajax and Flash remoting (AMF) requests

Well worth the $50 license fee



HTTPie

Ditch cURL.
Use HTTPie.

httpie.org

Perfect for testing and debugging APIs

Free; requires Python

1. ramsey@gamgee: ~ (zsh)

```
ramsey at gamgee in ~
$ http -v PUT httpbin.org/put hello=world
PUT /put HTTP/1.1
Accept: application/json
Accept-Encoding: gzip, deflate, compress
Content-Length: 18
Content-Type: application/json; charset=utf-8
Host: httpbin.org
User-Agent: HTTPie/0.6.0

{
    "hello": "world"
}

HTTP/1.0 200 OK
Access-Control-Allow-Origin: *
Connection: close
Content-Length: 562
Content-Type: application/json
Date: Wed, 09 Oct 2013 03:28:51 GMT
Server: gunicorn/0.17.4
Via: 1.1 localhost:3128 (squid/2.7.STABLE3)
X-Cache: MISS from localhost
X-Cache-Lookup: MISS from localhost:3128

{
    "args": {},
    "data": "{\"hello\": \"world\"}",
    "files": {},
    "form": {},
    "headers": {
        "Accept": "application/json",
        "Accept-Encoding": "gzip, deflate, compress",
        "Cache-Control": "max-age=259200",
        "Connection": "close",
        "Content-Length": "18",
        "Content-Type": "application/json; charset=utf-8",
        "Host": "httpbin.org",
        "User-Agent": "HTTPie/0.6.0",
        "Via": "1.1 localhost:3128 (squid/2.7.STABLE3)"
    },
    "json": {
        "hello": "world"
    },
    "origin": "10.71.13.215, 63.239.94.10",
    "url": "http://httpbin.org/put"
}


ramsey at gamgee in ~
$
```

The protocol

Properties of HTTP

A client-server architecture

Atomic

Cacheable

A uniform interface

Layered

Code on demand

RESTful!

RFC 2616

GET

POST

PUT

DELETE

HEAD

OPTIONS

TRACE

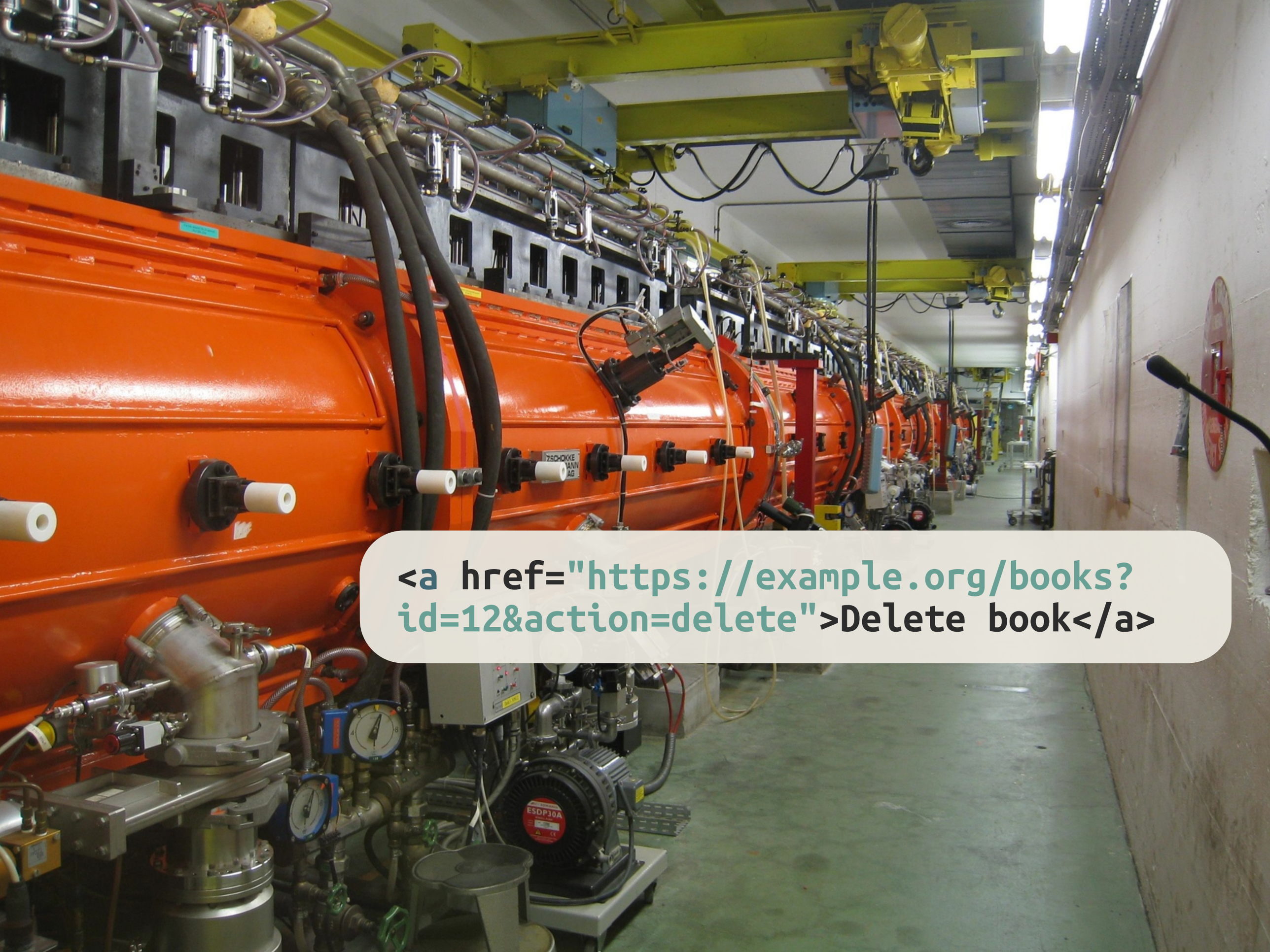
CONNECT

Safe methods

GET and HEAD should not take action other than retrieval

These are considered safe

This allows user agents to represent POST, PUT, and DELETE in a special way



```
<a href="https://example.org/books?
id=12&action=delete">Delete book</a>
```

Idempotence

Side effects of N > 0 identical requests is the same as for a single request

GET, HEAD, PUT, and DELETE share this property

OPTIONS and TRACE are inherently idempotent

GET

Usually used for retrieval of information

Transfers a representation of the resource from the server to the client

Safe & idempotent

```
GET /get?foo=bar HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate, compress
Host: httpbin.org
User-Agent: HTTPie/0.6.0
```

```
HTTP/1.0 200 OK
Connection: close
Content-Length: 391
Content-Type: application/json
Date: Wed, 09 Oct 2013 03:09:15 GMT
Server: gunicorn/0.17.4

{
    "args": {
        "foo": "bar"
    },
    "headers": {...},
    "origin": "...",
    "url": "http://httpbin.org/get?foo=bar"
}
```

HEAD

Identical to GET, except...

Returns only the headers, not the body

Useful for getting details about a resource representation before retrieving the full representation

Safe & idempotent

```
HEAD /get?foo=bar HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate, compress
Host: httpbin.org
User-Agent: HTTPie/0.6.0
```

POST

The body content should be accepted as a new subordinate of the resource

Append, annotate, paste after

Not safe or idempotent

```
POST /post HTTP/1.1
Accept: application/json
Accept-Encoding: gzip, deflate, compress
Content-Length: 14
Content-Type: application/json; charset=utf-8
Host: httpbin.org
User-Agent: HTTPie/0.6.0

{
    "foo": "bar"
}
```

PUT

Storage of information

Transfers a *full* representation of a resource from the client to the server

Not safe

Idempotent

```
PUT /put HTTP/1.1
Accept: application/json
Accept-Encoding: gzip, deflate, compress
Content-Length: 14
Content-Type: application/json; charset=utf-8
Host: httpbin.org
User-Agent: HTTPie/0.6.0

{
    "foo": "bar"
}
```

DELETE

Requests that the resource identified be removed from public access

Not safe

Idempotent

```
DELETE /delete HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate, compress
Content-Length: 0
Host: httpbin.org
User-Agent: HTTPie/0.6.0
```

Why are PUT & DELETE idempotent?

The data on the server changes, right?

Right. But...

The state remains the same for every request.

What’s the difference between POST and PUT?

```
POST /books HTTP/1.1

PUT /books/decd0562 HTTP/1.1
```

POST vs. PUT

The fundamental difference between the POST and PUT requests is reflected in the ***different meaning of the Request-URI***. The URI in a POST request identifies the resource that will handle the enclosed entity. That resource might be a data-accepting process, a gateway to some other protocol, or a separate entity that accepts annotations. In contrast, the URI in a PUT request identifies the entity enclosed with the request—the user agent knows what URI is intended and the server MUST NOT attempt to apply the request to some other resource.

—from RFC 2616, Section 9.6

Status codes

1xx: Informational

2xx: Successful

3xx: Redirection

4xx: Client error

5xx: Server error

Diving deeper

Content negotiation

Caching

Conditional requests

Range requests

Content negotiation

a.k.a. conneg

Server-driven negotiation

Agent-driven negotiation

Server-driven

The client may send headers to help the server guess: **Accept**, **Accept-Language**, **Accept-Encoding**, **Accept-Charset**, and **User-Agent**

The server can use other factors

It's the server's best guess, so the response could be different on subsequent identical requests

```
GET /books/9790482c HTTP/1.1
Accept-Charset: utf-8
Host: example.com
Accept-Language: en-us, en-gb;q=0.8, en;q=0.7
Accept-Encoding: gzip
Accept: application/hal+json
User-Agent: HTTPie/0.2.0
```

```
HTTP/1.1 200 OK
Date: Mon, 30 Jul 2012 02:42:26 GMT
Server: Apache/2.2.22 (Ubuntu)
X-Powered-By: PHP/5.3.10-1ubuntu3.2
Content-Language: en-us
ETag: "9790482c-1"
Vary: Accept,Accept-Charset,Accept-Language,Accept-
Encoding
Content-Encoding: gzip
Content-Length: 213
Content-Type: application/hal+json; charset=utf-8

{
...
}
```

Agent-driven

Requires multiple requests from the client, sometimes

First request results in a response listing available representations either in the headers or in the entity body

Second request is either automatic (client chooses) or manual (user chooses) for the desired representation

```
GET /books/9790482c HTTP/1.1
Host: example.com
```

```
HTTP/1.1 300 Multiple Choices
Date: Mon, 30 Jul 2012 02:57:42 GMT
Server: Apache/2.2.22 (Ubuntu)
X-Powered-By: PHP/5.5.4
Content-Length: 444
Content-Type: application/hal+json
```

```
{
  "_links": {
    "alternate": [
      {
        "href": "http://example.com/books/9790482c.en-us.html",
        "hreflang": "en-us",
        "type": "text/html; charset=utf-8"
      },
      {
        "href": "http://example.com/books/9790482c.en-us.json",
        "hreflang": "en-us",
        "type": "application/hal+json; charset=utf-8"
      },
      {
        "href": "http://example.com/books/9790482c.en-us.xml",
        "hreflang": "en-us",
        "type": "application/hal+xml; charset=utf-8"
      }
    ],
    "self": {
      "href": "http://example.com/books/9790482c"
    }
  }
}
```

Caching

Expires

Cache-Control

Cache properties

max-age

s-maxage

public

private

no-cache

no-store

must-revalidate

proxy-revalidate

```
Cache-Control: max-age=3600, must-revalidate
```

Conditional requests

If-Modified-Since

If-Unmodified-Since

If-Match

If-None-Match

If-Range

```
GET /books/9790482c HTTP/1.1
Host: example.com
Accept-Encoding: identity, deflate, compress, gzip
Accept: application/hal+json
User-Agent: HTTPie/0.2.0
If-Modified-Since: Sun, 15 Jul 2012 16:34:23 GMT
```

```
HTTP/1.1 304 Not Modified
Date: Mon, 30 Jul 2012 03:39:51 GMT
Server: Apache/2.2.22 (Ubuntu)
Vary: Accept-Encoding
```

Range requests

Used when requests are made for ranges of bytes from a resource

Determine whether a server supports range requests by checking for the Accept-Ranges header with HEAD

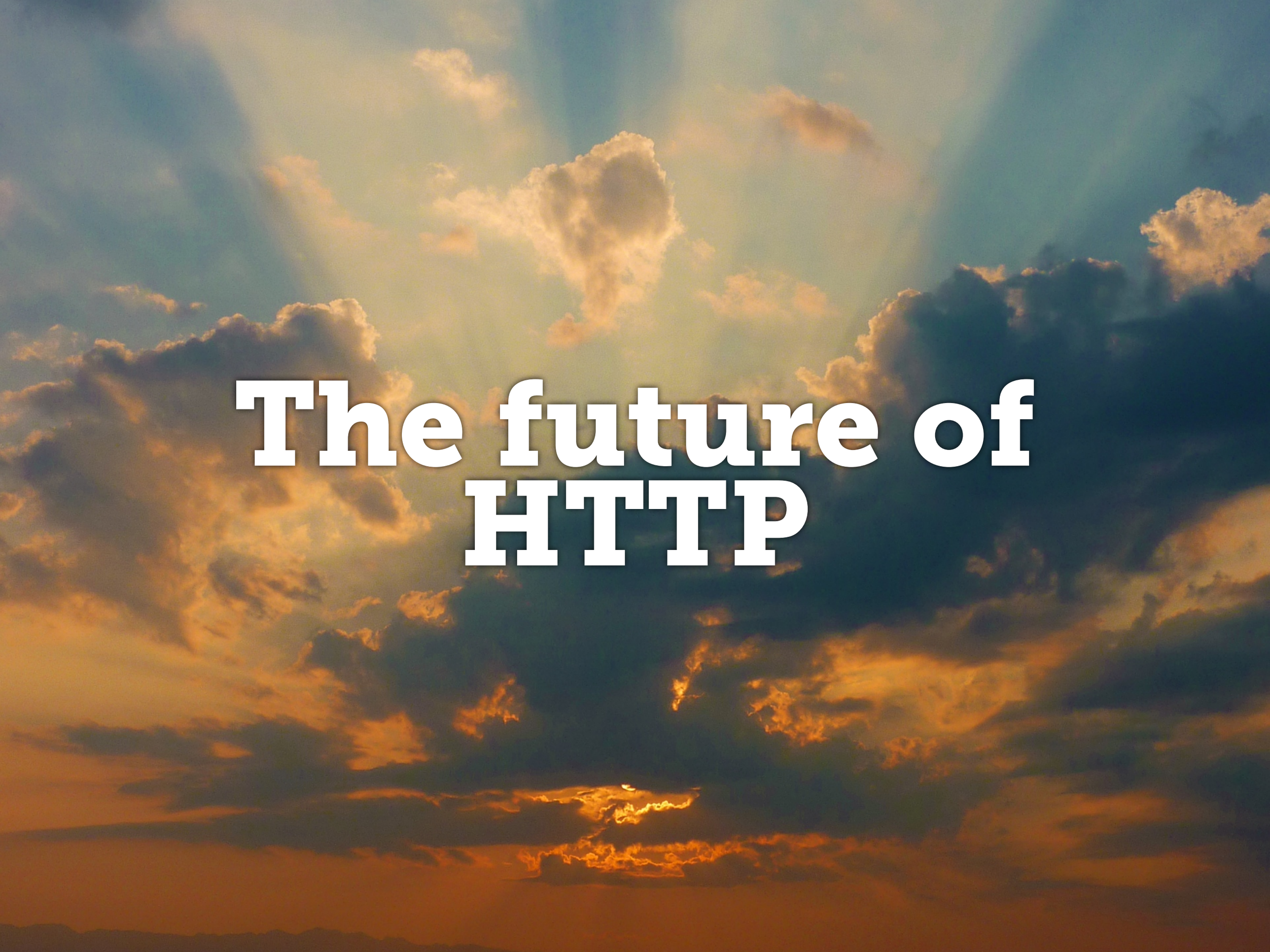
```
HEAD /2390/2253727548_a413c88ab3_s.jpg HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate, compress
Host: farm3.static.flickr.com
User-Agent: HTTPie/0.6.0
```

```
HTTP/1.0 200 OK
Accept-Ranges: bytes
Cache-Control: max-age=315360000,public
Content-Length: 3980
Content-Type: image/jpeg
Date: Wed, 09 Oct 2013 04:31:35 GMT
Expires: Mon, 09 Oct 2023 14:39:15 UTC
Last-Modified: Sat, 09 Feb 2008 23:04:10 GMT
```

```
GET /2390/2253727548_a413c88ab3_s.jpg HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate, compress
Host: farm3.static.flickr.com
Range: bytes=0-999
User-Agent: HTTPie/0.6.0
```

```
HTTP/1.0 206 Partial Content
Accept-Ranges: bytes
Cache-Control: max-age=315360000,public
Content-Length: 1000
Content-Range: bytes 0-999/3980
Content-Type: image/jpeg
Date: Wed, 09 Oct 2013 04:31:50 GMT
Expires: Mon, 09 Oct 2023 14:39:30 UTC
Last-Modified: Sat, 09 Feb 2008 23:04:10 GMT

{binary data}
```



The future of HTTP

PATCH

Allows a set of ***partial*** changes to be described, rather than the full entity body.

RFC 5789

```
OPTIONS /books/1984 HTTP/1.1
Host: example.org

HTTP/1.1 200 OK
Allow:
    GET, HEAD, PUT, PATCH, OPTIONS, DELETE
Accept-Patch:
    application/json-patch+json, text/diff
```

```http
PATCH /books/1984 HTTP/1.1
Host: example.org
Content-Length: 188
Content-Type: application/json-patch+json

[
    {
        "op": "replace",
        "path": "/isbn",
        "value": "978-0452262935"
    },
    {
        "op": "add",
        "path": "/asin",
        "value": "0452262933"
    }
]
```

More status codes

RFC 6585 defines more status codes

428 Precondition Required

429 Too Many Requests

431 Request Header Fields Too Large

Web linking

Defines a framework for typed links not specific to an application, and introduced the Link header.

RFC 5988

```
GET /books/?page=2 HTTP/1.1
Host: example.org

HTTP/1.1 200 OK
Content-Type: text/html
Link: <http://example.org/books/?page=1>;
      rel="previous"; title="Page 1",
      <http://example.org/books/?page=3>;
      rel="next"; title="Page 3"
```

Prefer header

Defines a header used by the client to request certain server behaviors when processing a request.

draft-snell-http-prefer-18

```http
POST /collection HTTP/1.1
Host: example.org
Content-Type: text/plain
Prefer: respond-async

{Data}
```

```http
HTTP/1.1 202 Accepted
Location: http://example.org/collection/123
Preference-Applied: respond-async
```

```http
POST /collection HTTP/1.1
Host: example.org
Content-Type: text/plain
Prefer: return=minimal

{Data}
```

```http
HTTP/1.1 201 Created
Location: http://example.org/collection/123
Preference-Applied: return=minimal
```

HTTPbis

bis is a Latin adverb meaning “twice”

Creating RFCs to clarify and supersede 1.1

Creating registries of method and authentication schemes

Drafting what will become HTTP 2.0

http://datatracker.ietf.org/wg/httpbis/

But wait!
There's more!

Resources

1. RFC 2616, http://tools.ietf.org/html/rfc2616
2. HTTPbin, for playing around with HTTP, http://httpbin.org/
3. HTTPie, http://httpie.org/
4. Charles Proxy, http://www.charlesproxy.com/
5. Mark Nottingham's Caching Tutorial, http://www.mnot.net/cache_docs/
6. PATCH Method for HTTP, http://tools.ietf.org/html/rfc5789
7. Additional HTTP Status Codes, http://tools.ietf.org/html/rfc6585
8. Web Linking, http://tools.ietf.org/html/rfc5988
9. Prefer Header for HTTP, http://tools.ietf.org/html/draft-snell-http-prefer
10. HTTPbis Working Group, http://datatracker.ietf.org/wg/httpbis/
11. HTTP 2.0, http://tools.ietf.org/html/draft-ietf-httpbis-http2
12. JSON Patch, http://tools.ietf.org/html/rfc6902
13. HTTP Status Code Registry, http://www.iana.org/assignments/http-status-codes/http-status-codes.xhtml
14. Message Headers Registry, http://www.iana.org/assignments/message-headers/message-headers.xhtml

Thank you

Ben Ramsey
benramsey.com
@ramsey

joind.in/10816

Ramsey, Ben. "Grokking HTTP." Lone Star PHP Conference. Addison Conference Centre, Addison, TX. 26 Apr. 2014. Conference Presentation.

Photo Credits

1. “GROK” by Cassidy Curtis, flickr.com/photos/cassidy/2519309017/
2. “Tools IMG_0171” by OZinOH, flickr.com/photos/75905404@N00/7126146307/
3. “LINAC2” by André Goerres, flickr.com/photos/gewuerzmandel/3314451829/
4. “Diving the Willaurie & Anthony Bell - Nassau, Bahamas” by Marc AuMarc, flickr.com/photos/theactionitems/3966877991/
5. “sunrise” by Sean MacEntee, flickr.com/photos/smemon/5783321374/