Hidden Gems in HTTP

Ben Ramsey
Code Works





Because you are a Web developer.

HTTP is the Web.

That's all I have to say about that.

Some properties of HTTP...

- A client-server architecture
- Atomic
- Cacheable
- A uniform interface
- Layered
- Code on demand

Now, what does that sound like?



And, that's all I have to say about that, too.

Our focus today...

- Methods you've never used
- Status codes you didn't know existed
- Working with HTTP in PHP

Methods you've never used...

Well, not really never.

GET

- You know GET
- Retrieval of information
- Transfers a representation of a resource from the server to the client
- Safe & idempotent

```
GET /user/ramsey HTTP/1.1
Host: atom.example.org
```

```
HTTP/1.1 200 OK
Date: Tue, 22 Sep 2009 17:28:14 GMT
Server: Apache/2.2.11 (Unix) DAV/2 PHP/5.3.0
X-Powered-By: PHP/5.3.0
Content-Length: 594
Content-Type: application/atom+xml;type=entry
```

</entry>

He just thinks he's funny.

Stop laughing. You're just encouraging him.

POST

- You know POST
- The body content should be accepted as a new subordinate of the resource
- Append, annotate, paste after
- Not safe or idempotent

```
POST /user HTTP/1.1
Host: atom.example.org
Content-Type: application/atom+xml;type=entry
Content-Length: 474
```

```
</entry>
```

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 /content/1234.mp4 HTTP/1.1
Host: atom.example.org
```

```
HTTP/1.1 200 OK
Date: Tue, 22 Sep 2009 17:28:14 GMT
Server: Apache/2.2.11 (Unix) DAV/2 PHP/5.3.0
X-Powered-By: PHP/5.3.0
Content-Length: 12334753
Content-Type: application/mp4
```

PUT

- Opposite of GET
- Storage of information
- Transfers a representation of a resource from the client to the server
- Not safe
- Idempotent

```
PUT /user/ramsey/ HTTP/1.1
Host: atom.example.org
Content-Type: application/atom+xml;type=entry
Content-Length: 594
<?xml version="1.0" encoding="utf-8"?>
<entry xmlns="http://www.w3.org/2005/Atom"</pre>
       xml:base="http://atom.example.org/">
 <title>ramsey</title>
</entry>
HTTP/1.1 200 OK
Date: Tue, 22 Sep 2009 17:47:27 GMT
Server: Apache/2.2.11 (Unix) DAV/2 PHP/5.3.0
X-Powered-By: PHP/5.3.0
Content-Length: 594
Content-Type: application/atom+xml;type=entry
<?xml version="1.0" encoding="utf-8"?>
<entry xmlns="http://www.w3.org/2005/Atom"</pre>
```

```
xml:base="http://atom.example.org/">
<title>ramsey</title>
```

```
</entry>
```



- Requests that the resource identified be removed from public access
- Not safe
- Idempotent

```
DELETE /content/1234/ HTTP/1.1
Host: example.org
```

```
HTTP/1.1 204 No Content
Date: Tue, 22 Sep 2009 18:06:37 GMT
Server: Apache/2.2.11 (Unix) DAV/2 PHP/5.3.0
X-Powered-By: PHP/5.3.0
Content-Length: 0
Content-Type: text/html; charset=utf-8
```

What the hell are safe & idempotent methods?

Safe methods

- GET & HEAD should not take action other than retrieval
- These are considered safe
- Allows agents to represent POST, PUT, & DELETE in a special way

Idempotence

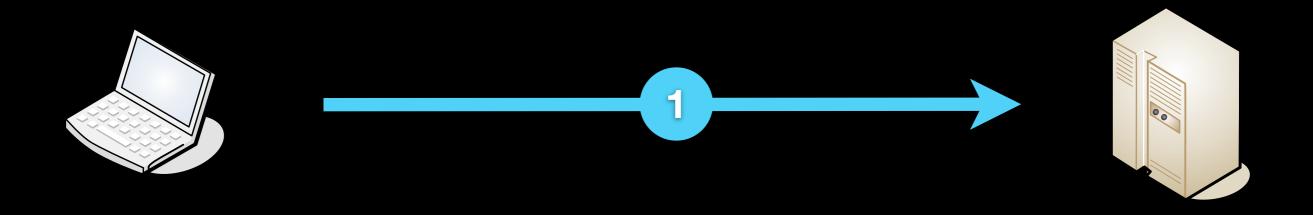
- Side-effects of N > 0 identical requests is the same as for a single request
- GET, HEAD, PUT and DELETE share this property
- OPTIONS and TRACE are inherently idempotent

Status codes you didn't know existed

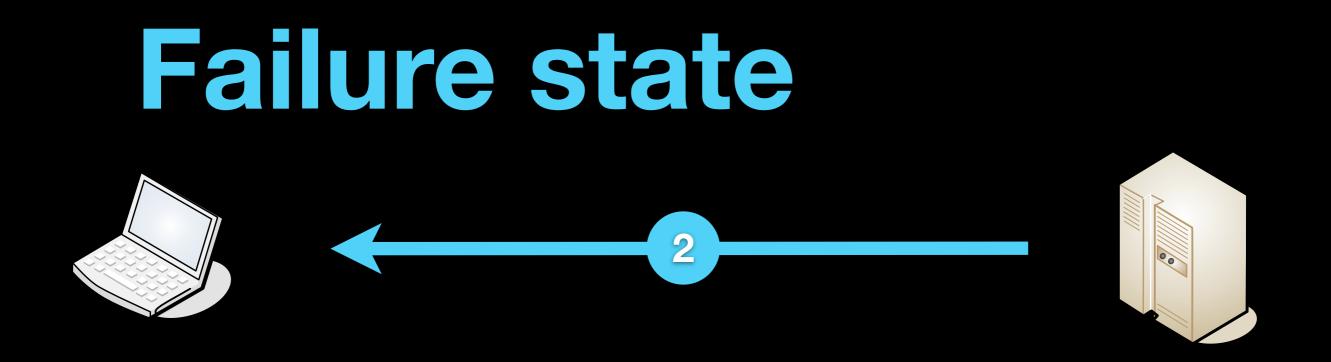
- Informational (1xx)
- Successful (2xx)
- Redirection (3xx)
- Client error (4xx)
- Server error (5xx)

The look-beforeyou-leap request (LBYL)

- Client sends a request without a body and includes the Expect: 100-continue header and all other headers
- Server determines whether it will accept the request and responds with 100 Continue (or a 4xx code on error)
- 3. Client sends the request again with the body and without the Expect header



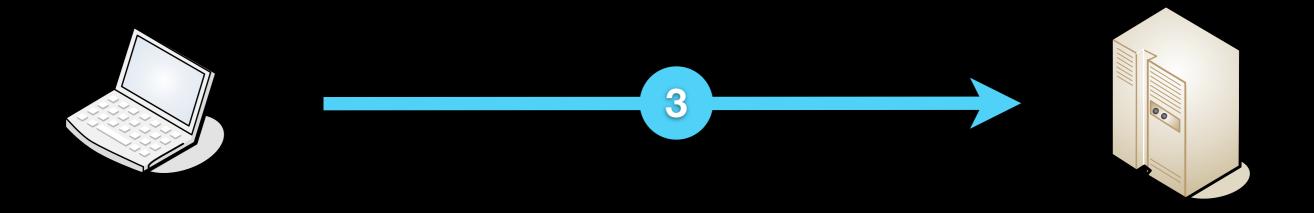
POST /content/videos HTTP/1.1
Host: example.org
Content-Type: video/mp4
Content-Length: 115910000
Authorization: Basic bWFkZTp5b3VfbG9vaw==
Expect: 100-continue



HTTP/1.1 413 Request Entity Too Large Date: Thu, 21 May 2009 23:05:15 GMT Server: Apache/2.2.11 (Unix) DAV/2 PHP/5.3.0 X-Powered-By: PHP/5.3.0 Content-Length: 0 Connection: close Content-Type: text/html

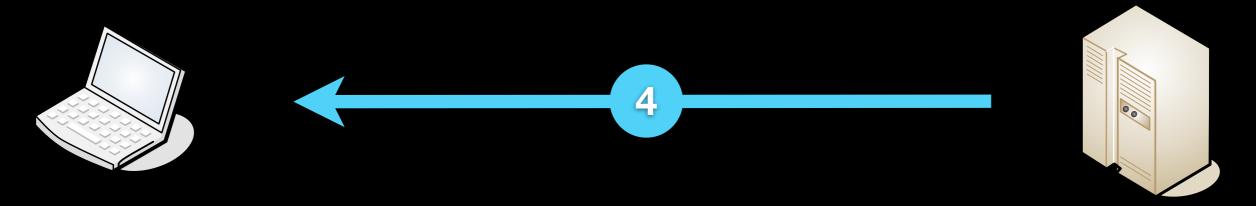


HTTP/1.1 100 Continue
Date: Thu, 21 May 2009 23:05:15 GMT
Server: Apache/2.2.11 (Unix) DAV/2 PHP/5.3.0
X-Powered-By: PHP/5.3.0
Content-Length: 0
Content-Type: text/html



POST /content/videos HTTP/1.1 Host: example.org Content-Type: video/mp4 Content-Length: 115910000 Authorization: Basic bWFkZTp5b3VfbG9vaw==

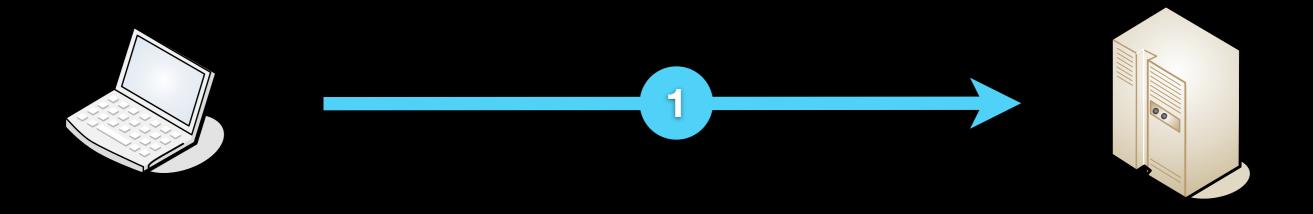
{binary video data}



HTTP/1.1 201 Created Date: Thu, 21 May 2009 23:05:34 GMT Server: Apache/2.2.11 (Unix) DAV/2 PHP/5.3.0 X-Powered-By: PHP/5.3.0 Content-Length: 119 Content-Type: text/html Location: http://example.org/content/videos/1234

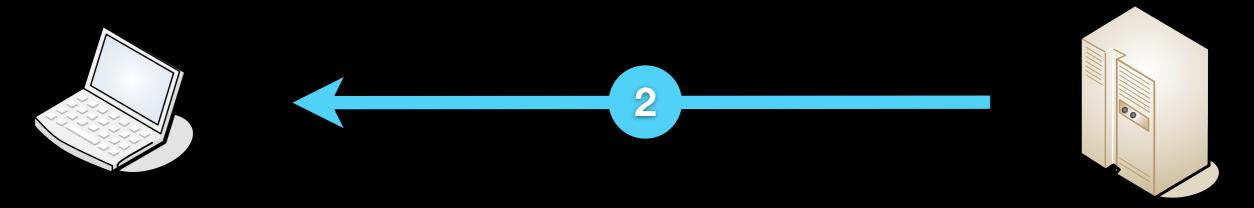
<html><body>Video uploaded! Go <a
href="http://example.org/content/videos/
1234">here to see it.</body></html>

The created at another location response



POST /content/videos HTTP/1.1
Host: example.org
Content-Type: video/mp4
Content-Length: 115910000
Authorization: Basic bWFkZTp5b3VfbG9vaw==

{binary video data}

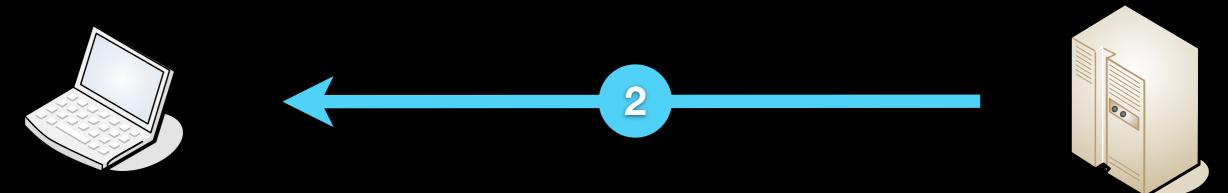


HTTP/1.x 201 Created Date: Thu, 21 May 2009 23:05:34 GMT Server: Apache/2.2.11 (Unix) DAV/2 PHP/5.3.0 X-Powered-By: PHP/5.3.0 Content-Length: 120 Content-Type: text/html Location: http://example.org/content/videos/1234

<html><body>Video uploaded! Go <a
href="http://example.org/content/videos/
1234">here to see it.</body></html>

The "it's not you it's me" response

i.e. I've accepted it but might have to do more processing



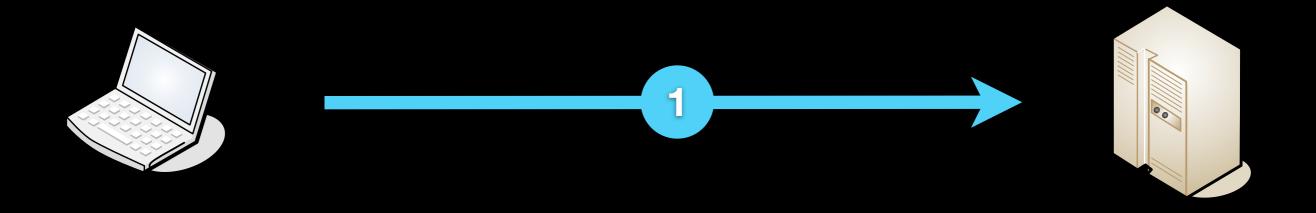
HTTP/1.x 202 Accepted Date: Thu, 21 May 2009 23:05:34 GMT Server: Apache/2.2.11 (Unix) DAV/2 PHP/5.3.0 X-Powered-By: PHP/5.3.0 Content-Length: 137 Content-Type: text/html Location:

http://example.org/content/videos/1234/status

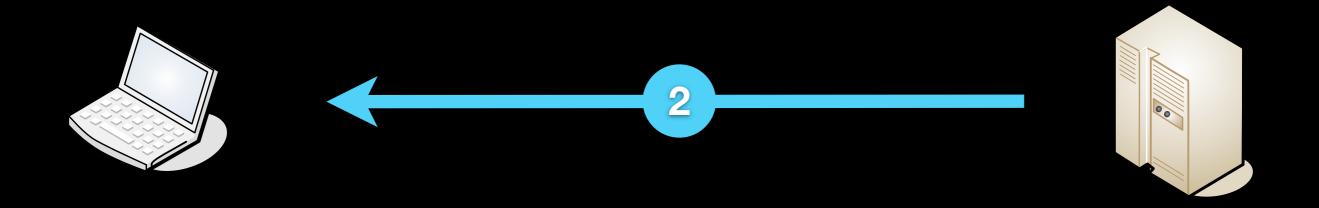
<html><body>Video processing! Check <a
href="http://example.org/content/videos/1234/
status">here for the status.</body></
html>

The "I have nothing to say to you" response...

...but you were still successful



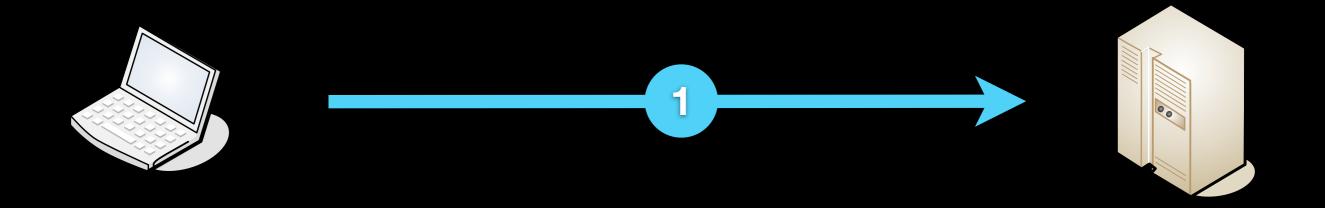
DELETE /content/videos/1234 HTTP/1.1 Host: example.org Authorization: Basic bWFkZTp5b3VfbG9vaw==



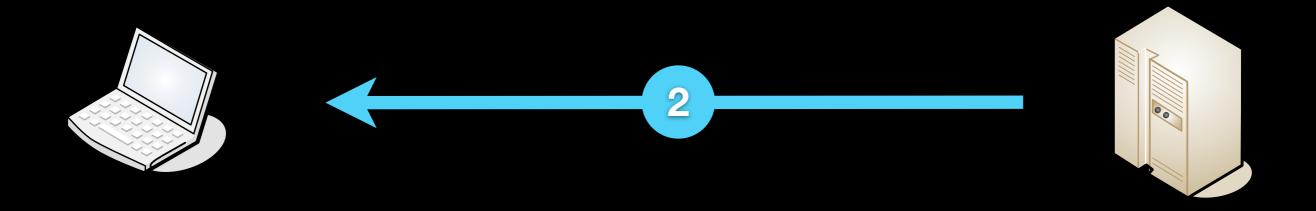
HTTP/1.x 204 No Content Date: Thu, 21 May 2009 23:28:34 GMT

The ranged request

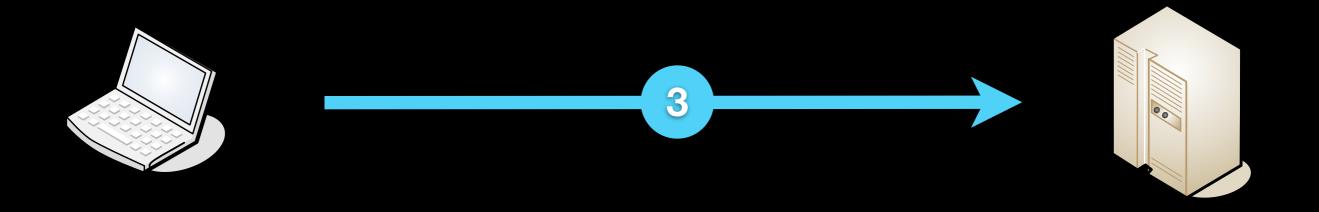
- Used when requests are made for ranges of bytes from a resource
- Determine whether a server supports range requests by checking for the Accept-Ranges header with HEAD



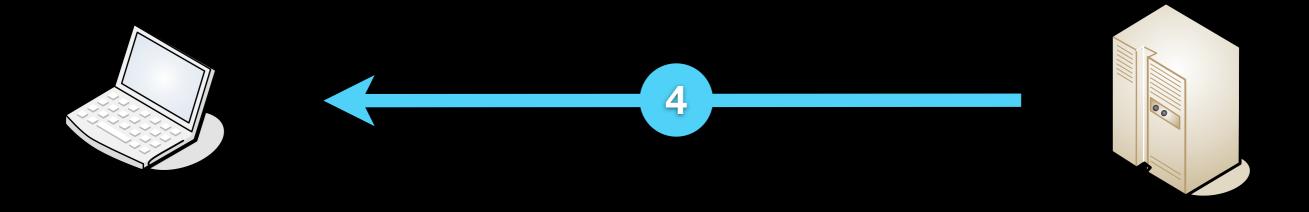
HEAD /2390/2253727548_a413c88ab3_s.jpg HTTP/1.1 Host: farm3.static.flickr.com



```
HTTP/1.0 200 OK
Date: Mon, 05 May 2008 00:33:14 GMT
Server: Apache/2.0.52 (Red Hat)
Accept-Ranges: bytes
Content-Length: 3980
Content-Type: image/jpeg
```



GET /2390/2253727548_a413c88ab3_s.jpg HTTP/1.1 Host: farm3.static.flickr.com Range: bytes=0-999

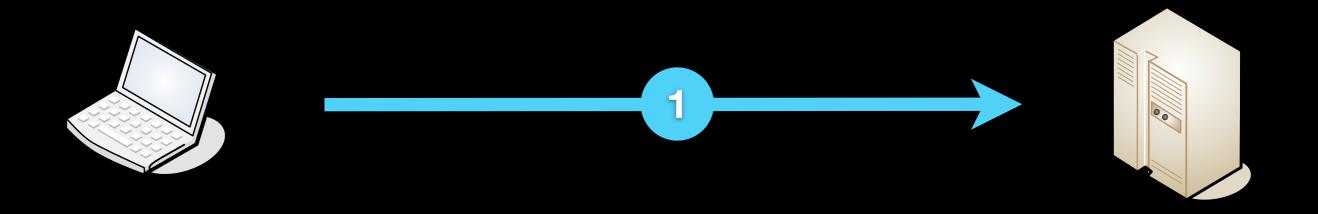


HTTP/1.0 206 Partial Content Date: Mon, 05 May 2008 00:36:57 GMT Server: Apache/2.0.52 (Red Hat) Accept-Ranges: bytes Content-Length: 1000 Content-Range: bytes 0-999/3980 Content-Type: image/jpeg

{binary data}

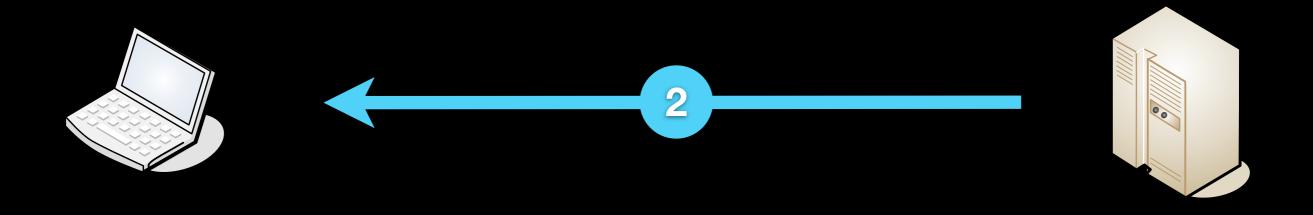
The GET me from another location response

- 303 See Other
- The response to your request can be found at another URL identified by the Location header
- The client should make a GET request on that URL
- The Location is not a substitute for this URL



POST /contact HTTP/1.1 Host: example.org Content-Type: application/x-www-form-urlencoded Content-Length: 1234

{url-encoded form values from a contact form}



HTTP/1.1 303 See Other Date: Tue, 22 Sep 2009 23:41:33 GMT Server: Apache/2.2.11 (Unix) DAV/2 PHP/5.3.0 X-Powered-By: PHP/5.3.0 Location: http://example.org/thankyou Content-Length: 0

The find me temporarily at this place response

- 307 Temporary Redirect
- The resource resides temporarily at the URL identified by the Location
- The Location may change, so don't update your links
- If the request is not GET or HEAD, then you must allow the user to confirm the action

The permanent forwarding address response

- 301 Moved Permanently
- The resource has moved permanently to the URL indicated by the Location header
- You should update your links accordingly
- Great for forcing search engines, etc. to index the new URL instead of this one

But what about just finding the resource at another location?

302 Found

- The resource has been found at another URL identified by the Location header
- The new URL might be temporary, so the client should continue to use this URL
- Redirections SHOULD be confirmed by the user (in practice, browsers don't respect this)

The data validation error response

- 400 Bad Request
- Generic error message
- The client sent malformed syntax
- The client needs to modify the request before sending it again (to fix errors)

```
POST /user/ HTTP/1.1
Host: atom.example.org
Content-Type: application/atom+xml;type=entry
Content-Length: 474
```

```
</entry>
```

```
HTTP/1.1 400 Bad Request
Date: Tue, 22 Sep 2009 23:51:00 GMT
Server: Apache/2.2.11 (Unix) DAV/2 PHP/5.3.0
X-Powered-By: PHP/5.3.0
Content-Length: 123
Connection: close
Content-Type: text/html; charset=utf-8
<div class="error">
The following errors occurred:

            Title contained invalid characters

</div>
```

But wait! There's more...

Working with HTTP in PHP

- header() function <u>http://php.net/header</u>
- Client URL library (cURL) <u>http://php.net/curl</u>
- Streams <u>http://php.net/streams</u>
- HTTP extension (pecl/http) <u>http://php.net/http</u>

Questions?

- My website is <u>benramsey.com</u>
- <u>@ramsey</u> on Twitter
- Rate this talk at joind.in
- Read the HTTP spec at tools.ietf.org/html/rfc2616
- My company is Schematic <u>schematic.com</u>

Hidden Gems in HTTP Copyright © <u>Ben Ramsey</u>. Some rights reserved.

This work is licensed under a <u>Creative Commons</u> <u>Attribution-Noncommercial-No Derivative Works 3.0 United</u> <u>States License</u>.

For uses not covered under this license, please contact the author.

