HTTP/2 AND ASYNCHRONOUS APIS

DAVEY SHAFIK

- Developer
- Author
- Open Source Contributor
- Release Manager for PHP 7.1
- **@dshafik**

http://developer.akamai.com

prompt

Let’s start a conversation about mental health in tech

mhprompt.org

HTTP/2

AKA: H2

HTTP/2

WHAT IS HTTP/2?

HYPERTEXT TRANSFER PROTOCOL VERSION 2

RFC7540

HPACK - HEADER COMPRESSION FOR HTTP/2

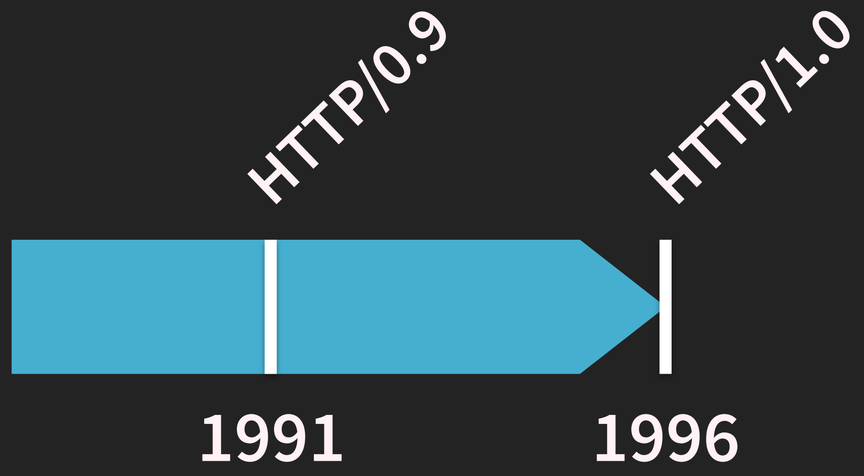
RFC7541

CHAIRED BY AKAMAI'S MARK NOTTINGHAM

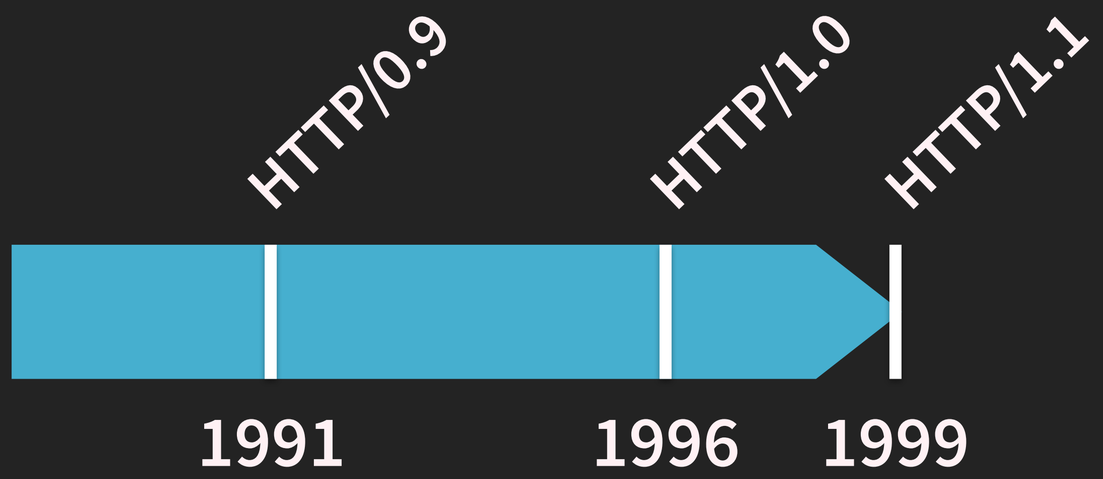
CREATED BY IETF HTTP WORKING GROUP

1991

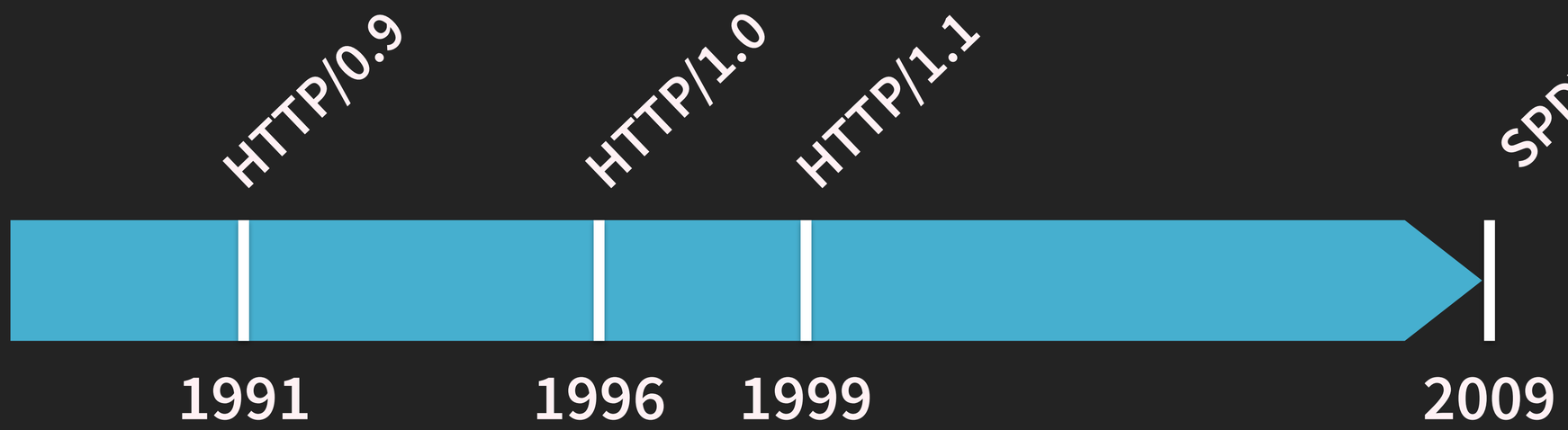
CC-BY: mrbill


HTTP/0.9
HTTP/1.0
1991
1996

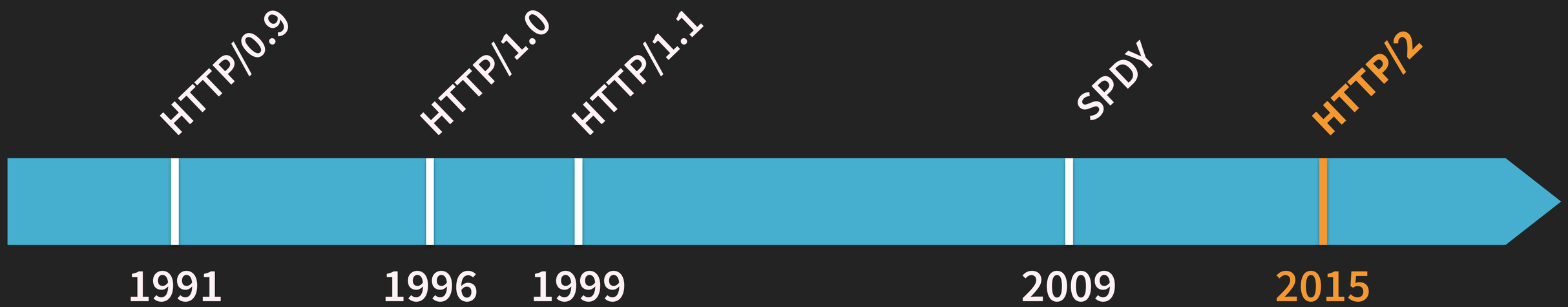
CC-BY: mrbill


HTTP/0.9
HTTP/1.0
HTTP/1.1
1991
1996
1999

CC-BY: mrbill


HTTP/0.9
HTTP/1.0
HTTP/1.1
SPDY
1991
1996
1999
2009

CC-BY: mrbill


HTTP/0.9
HTTP/1.0
HTTP/1.1
SPDY
HTTP/2
1991
1996
1999
2009
2015

BROWSER SUPPORT

BROWSER SUPPORT

- Chrome + Chrome Mobile
- Firefox
- IE 11 on Windows 10
- MS Edge
- Safari (El Capitan/iOS 9)
- Opera

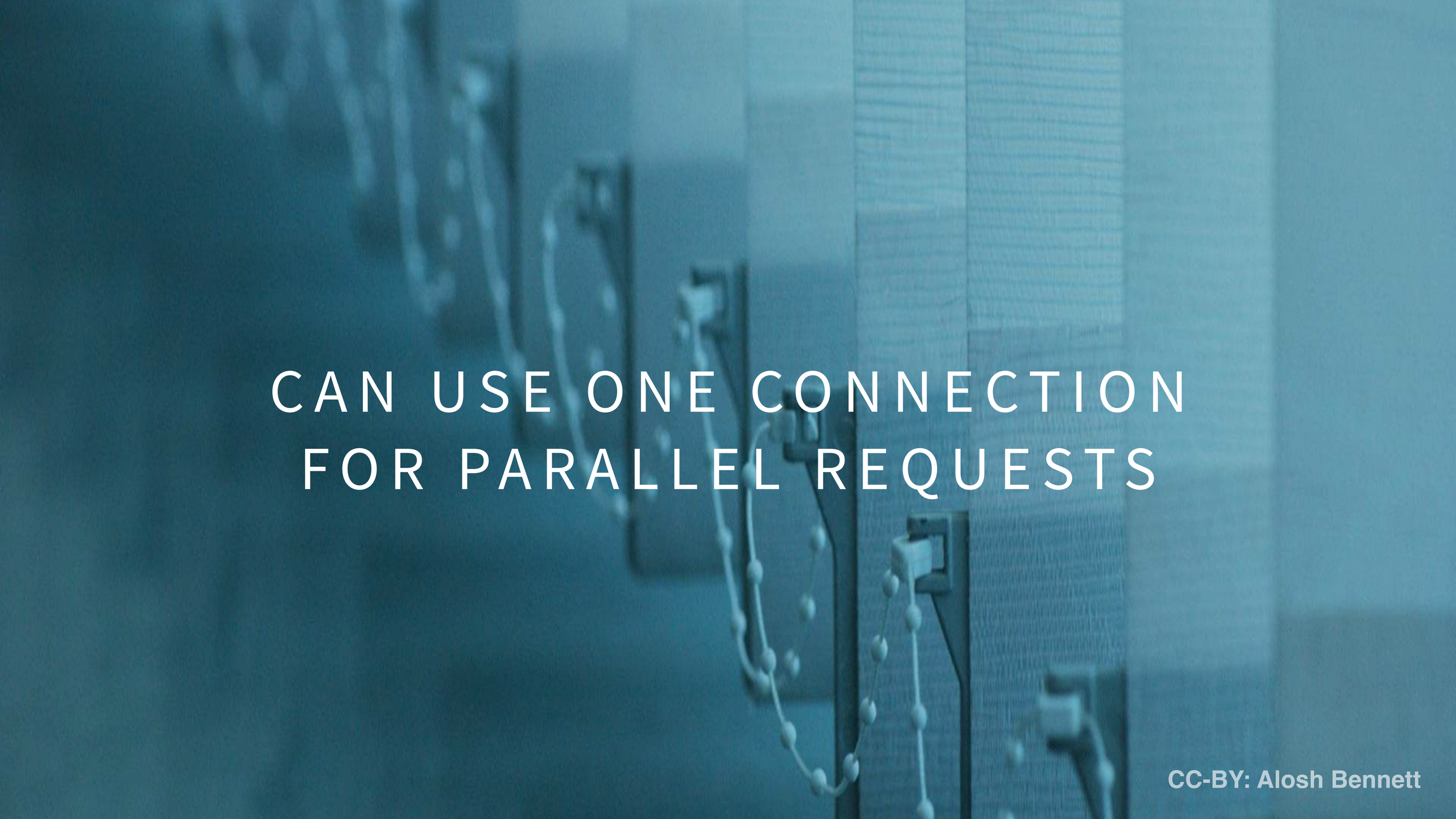
BINARY INSTEAD OF TEXT

CC-BY: brett jordan

FULLY MULTIPLEXED

INSTEAD OF ORDERED AND BLOCKING

CC-BY: vadikunc



CC-BY: Alosh Bennett

USES HEADER COMPRESSION

REDUCES OVERHEAD

CC-BY-SA: Magnus Hagdorn

SERVER PUSH

IS SUPER COOL (NO REALLY)

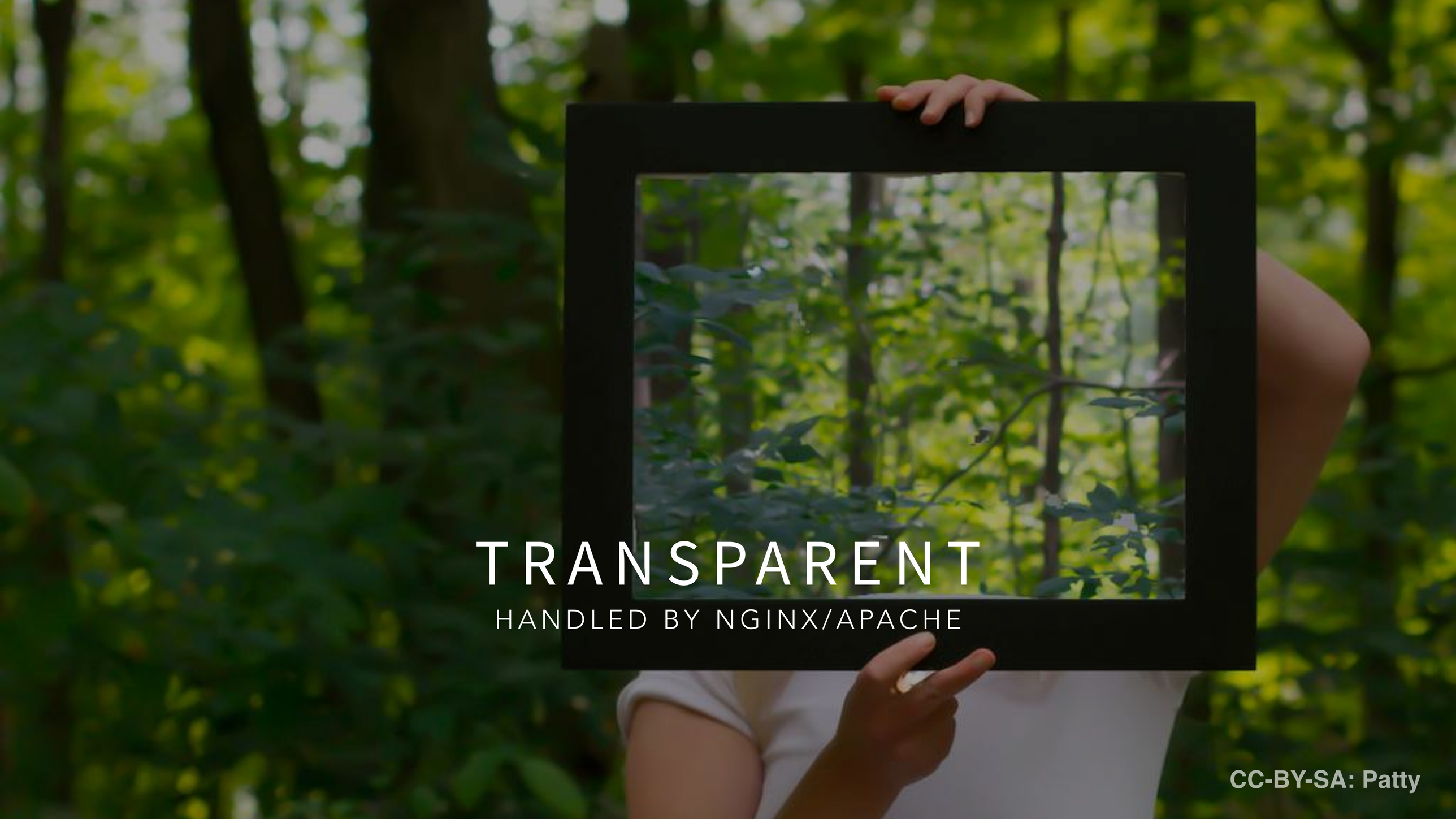
PRESS START

CC-BY-SA: Takeshi

SERVER PUSH

- Allows the server to proactively push assets like stylesheets and images to the client without them needing to parse the HTML page and make subsequent requests
- Done by pushing the assets into the client cache, avoiding the roundtrip necessary to pull them up once the client makes the request

WHAT DOES HTTP/2 MEAN FOR MY APPLICATION?



CC-BY-SA: Patty

SO WHAT'S THE POINT?

HTTP/1.X SUCKS

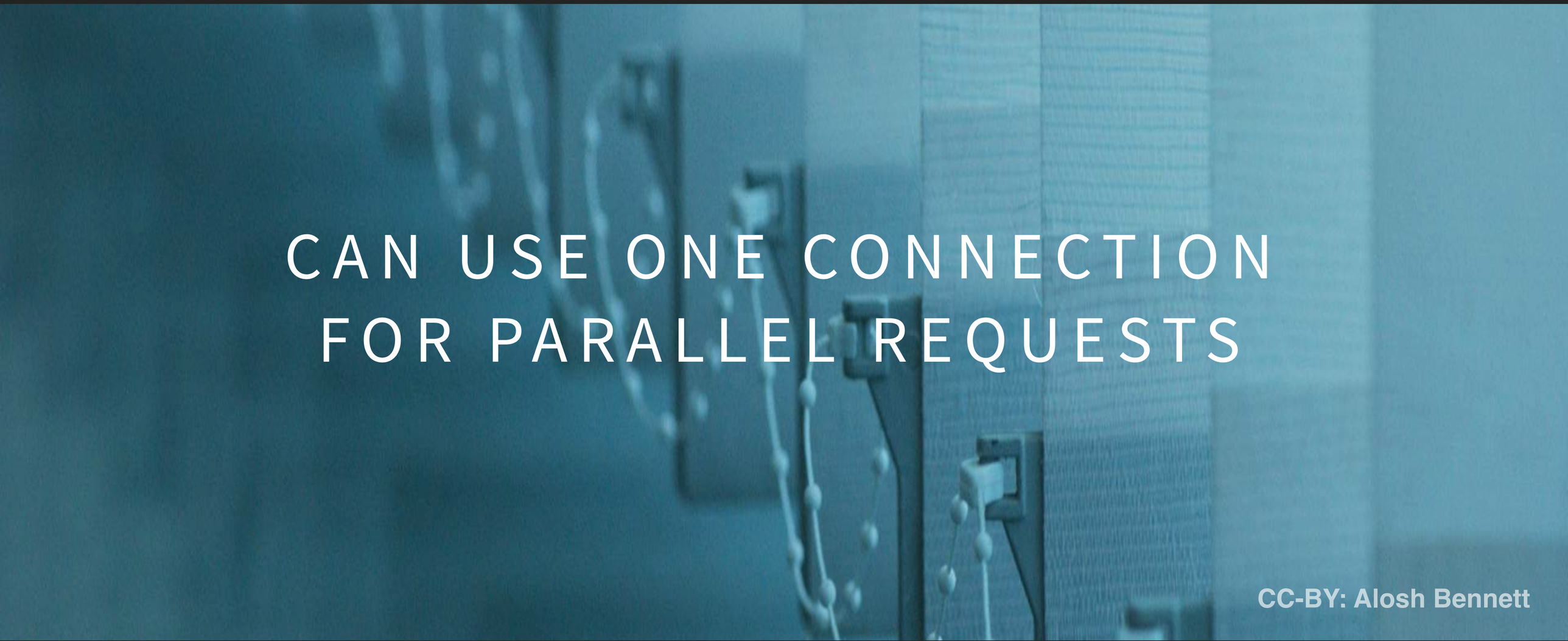
CC-BY: Flóra Soós

HTTP/1.X SUCKS

- Minify + Concat JavaScript and CSS
- Inlining small JavaScript and CSS
- Using image sprites
- Using data: URIs
- Domain sharding

THESE THINGS ARE ALL "CLEVER" HACKS
DEM16217
Arduino NG
www.arduino.cc
CC-BY: Matt Biddulph

REMEMBER THIS?

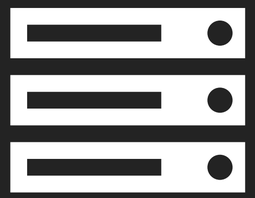


CC-BY: Alosh Bennett

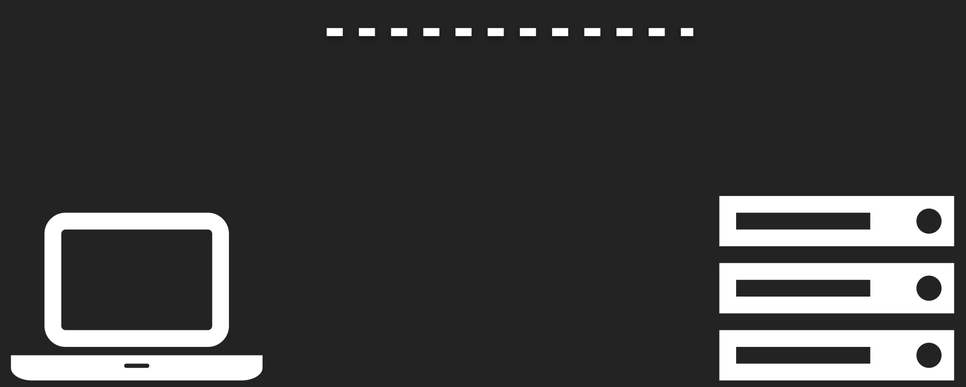
UPLOADING MULTIPLE IMAGES

CC-BY: John Trainor

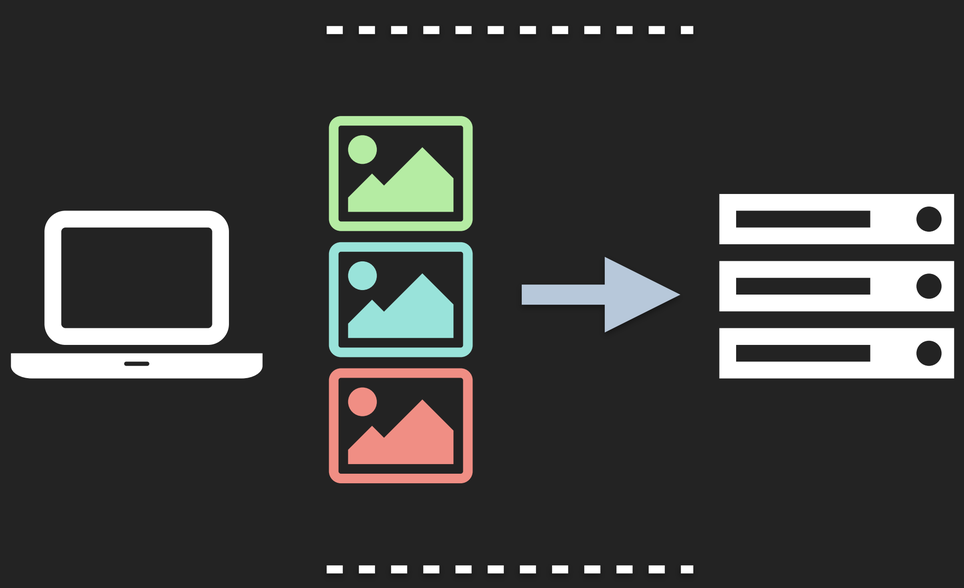
SERIAL UPLOADS



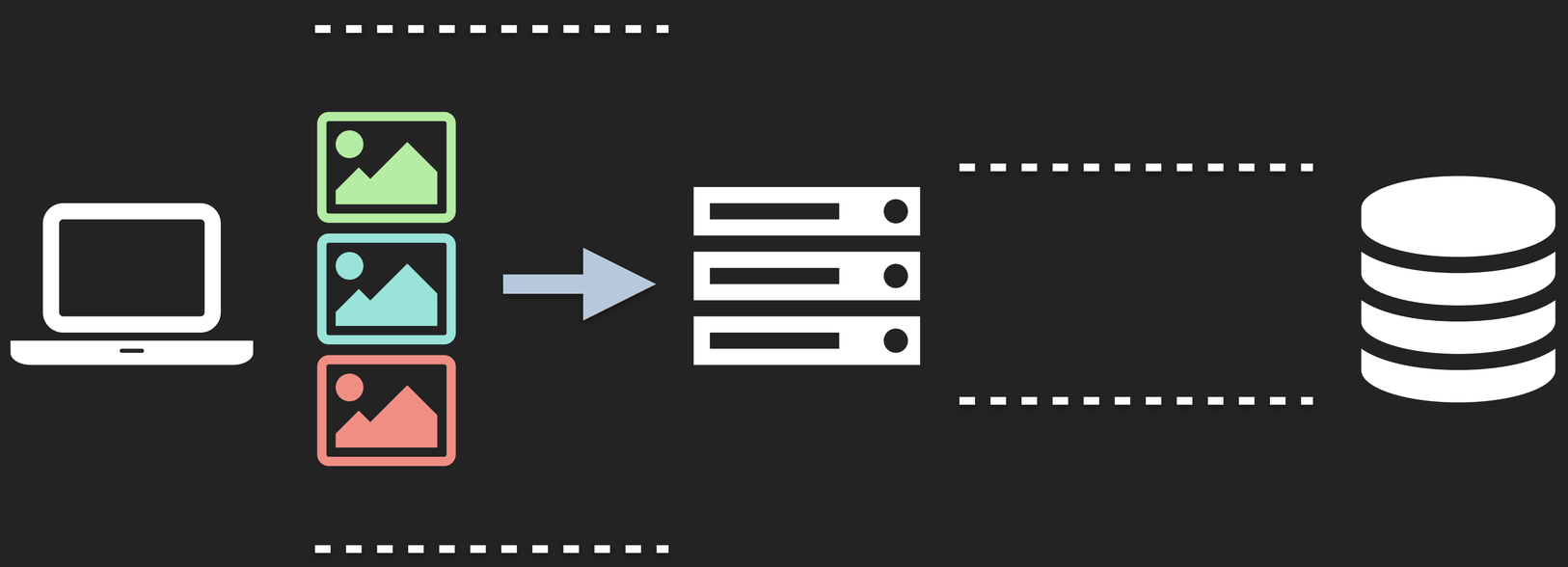
SERIAL UPLOADS



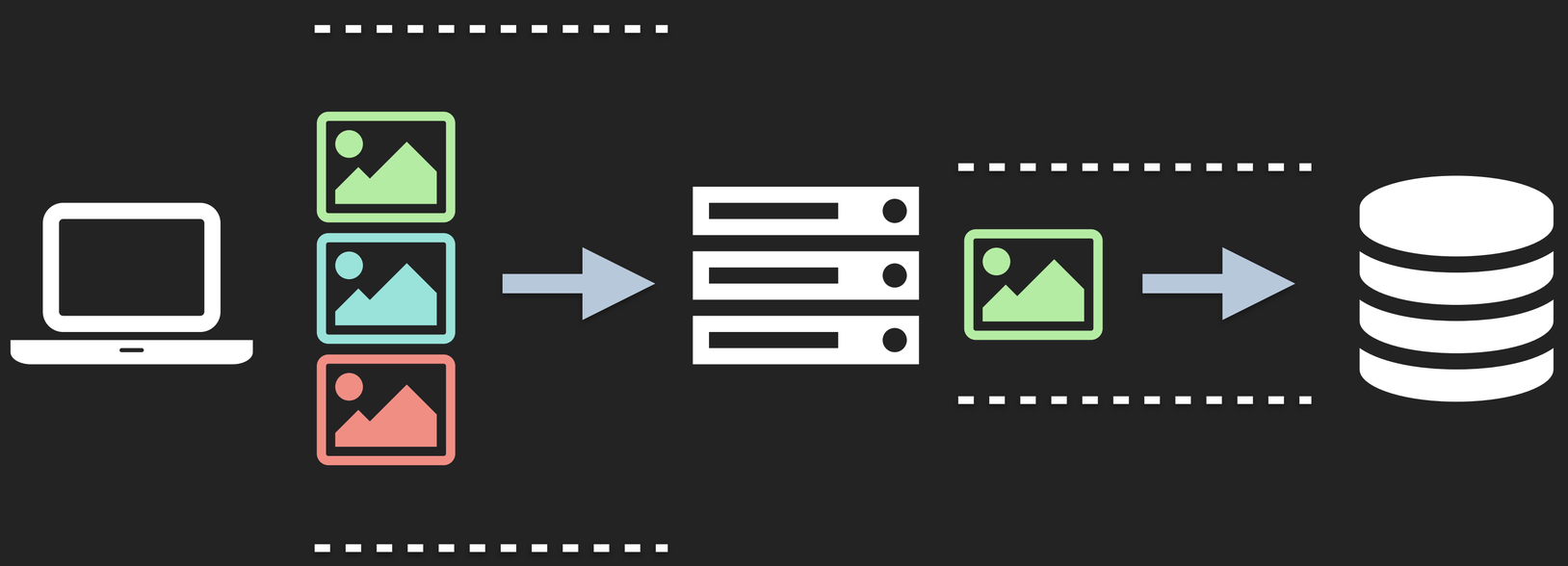
SERIAL UPLOADS



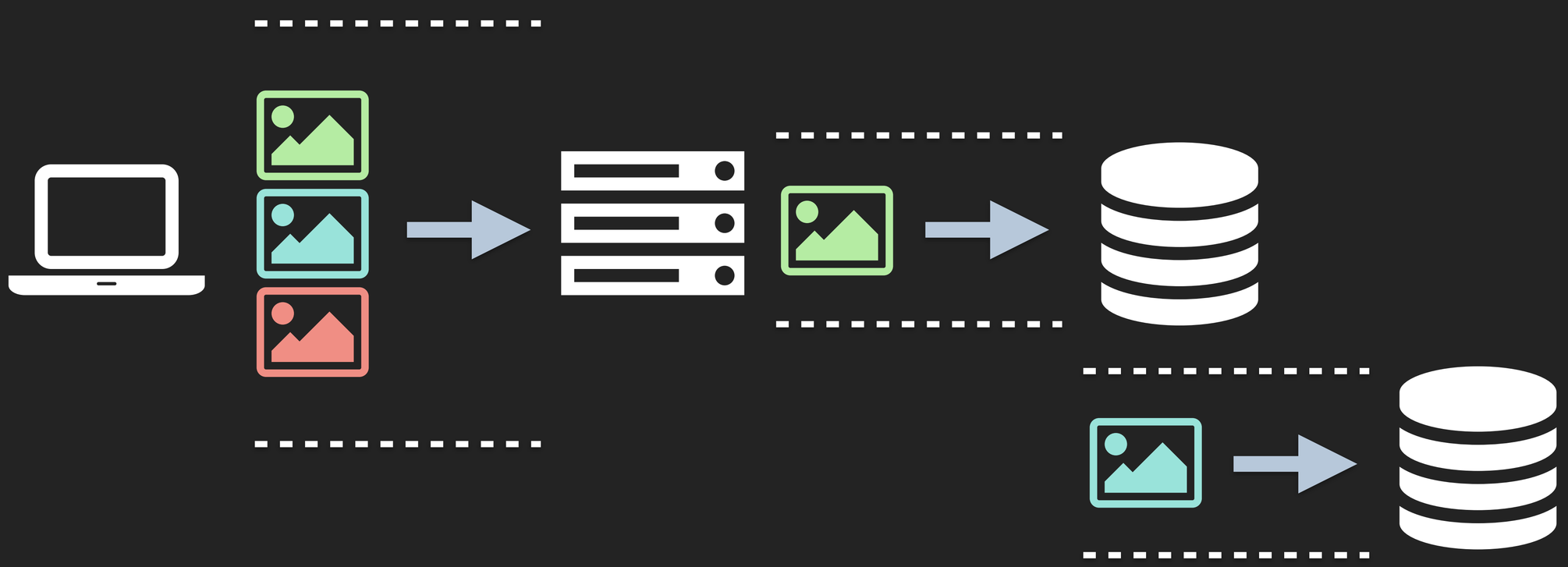
SERIAL UPLOADS



SERIAL UPLOADS

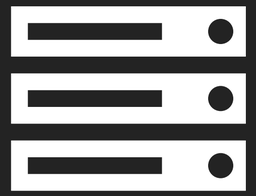


SERIAL UPLOADS

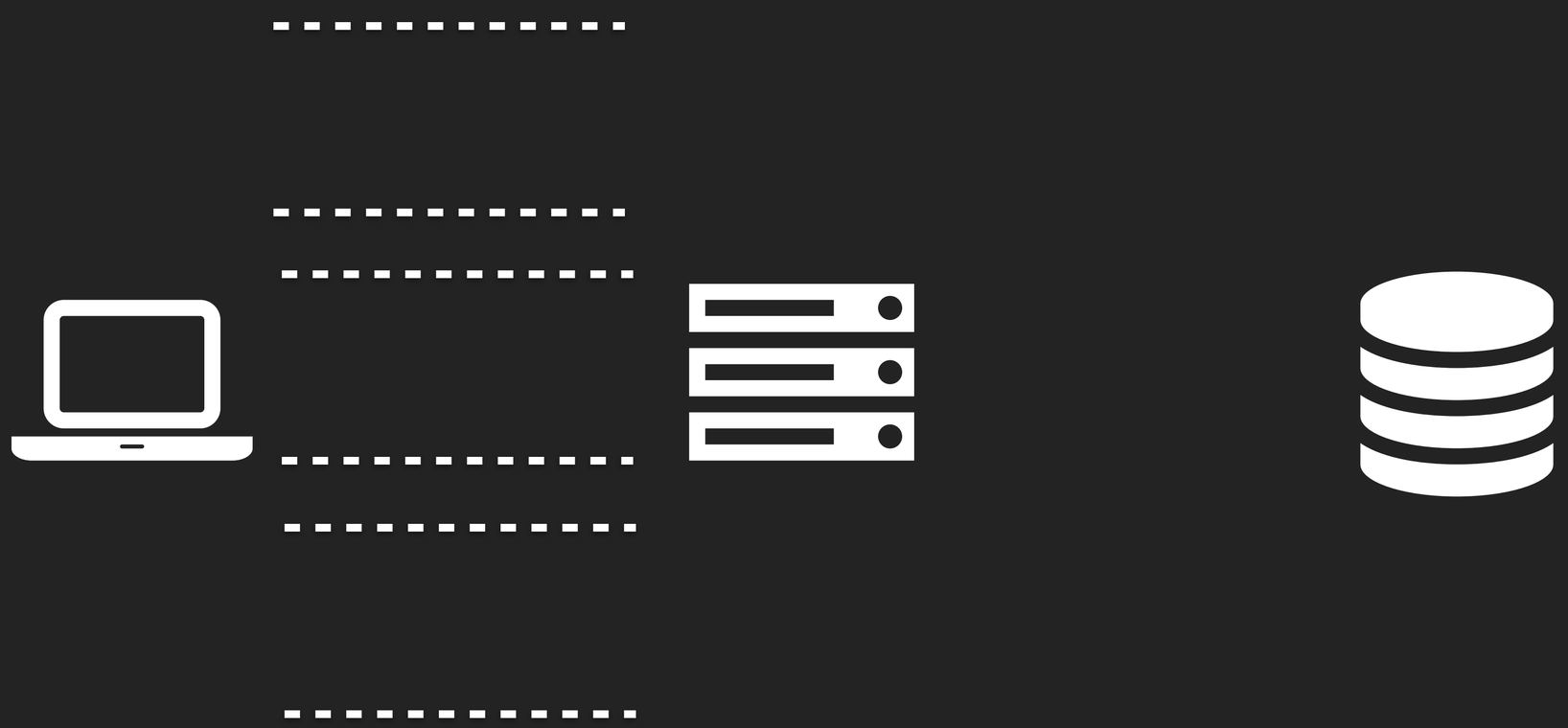


SERIAL UPLOADS

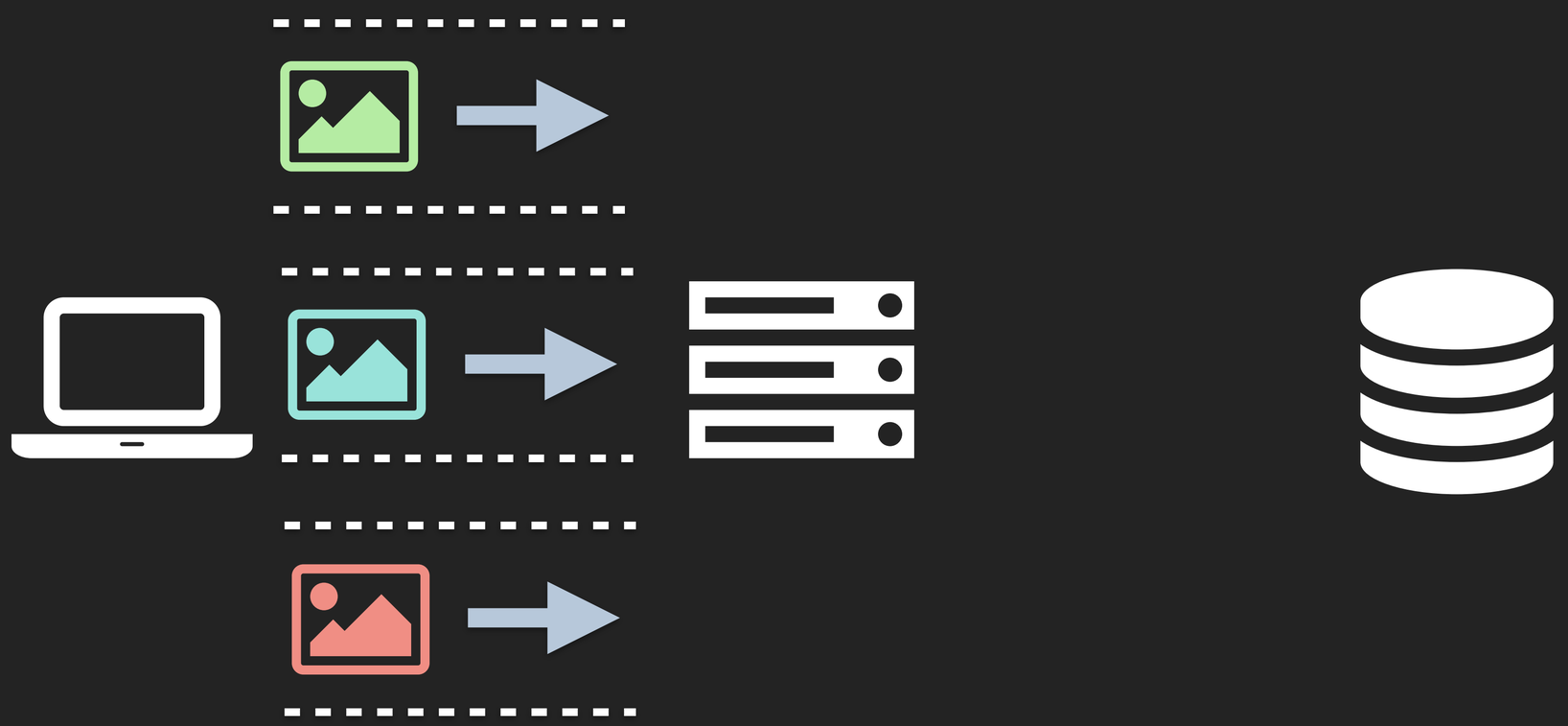
CONCURRENT



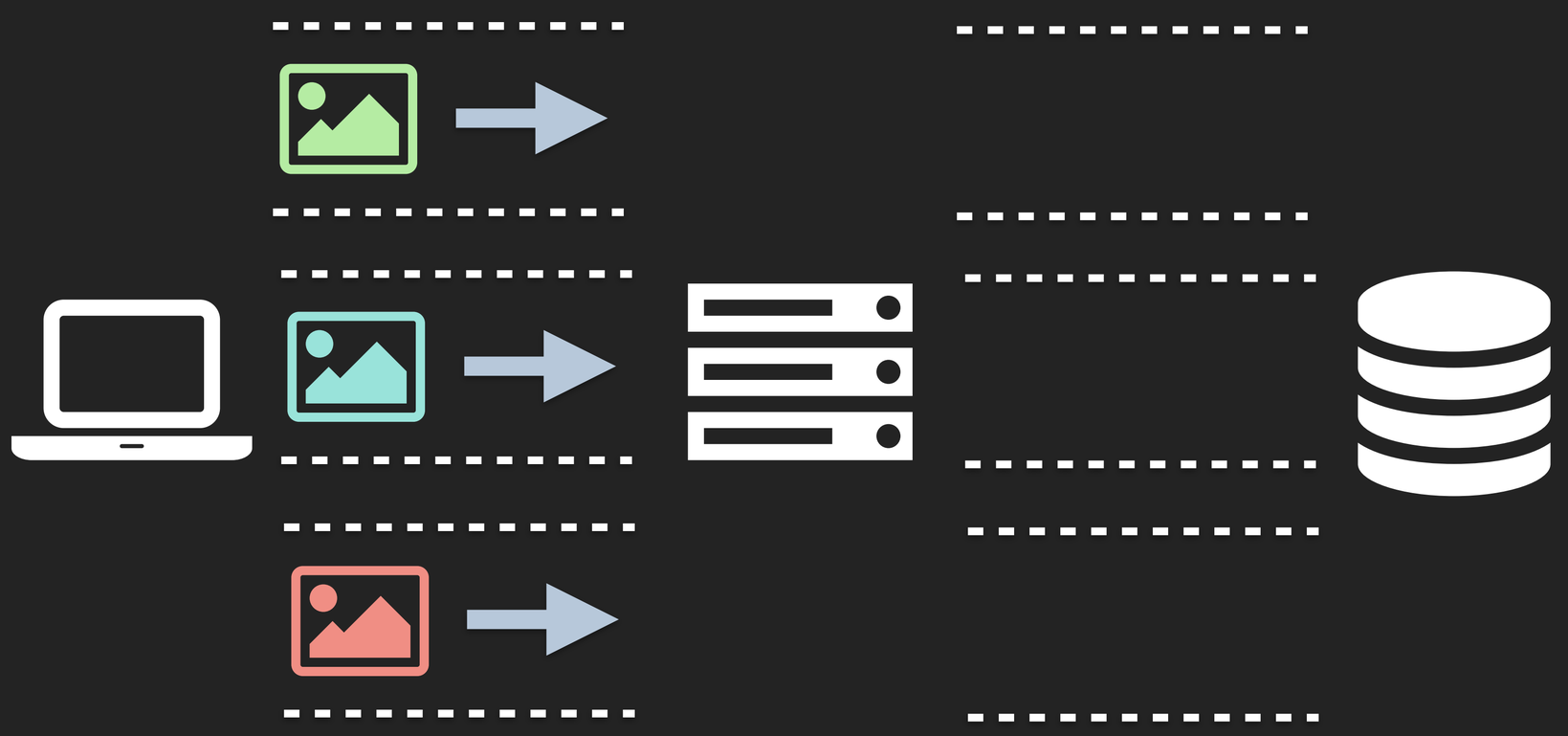
CONCURRENT



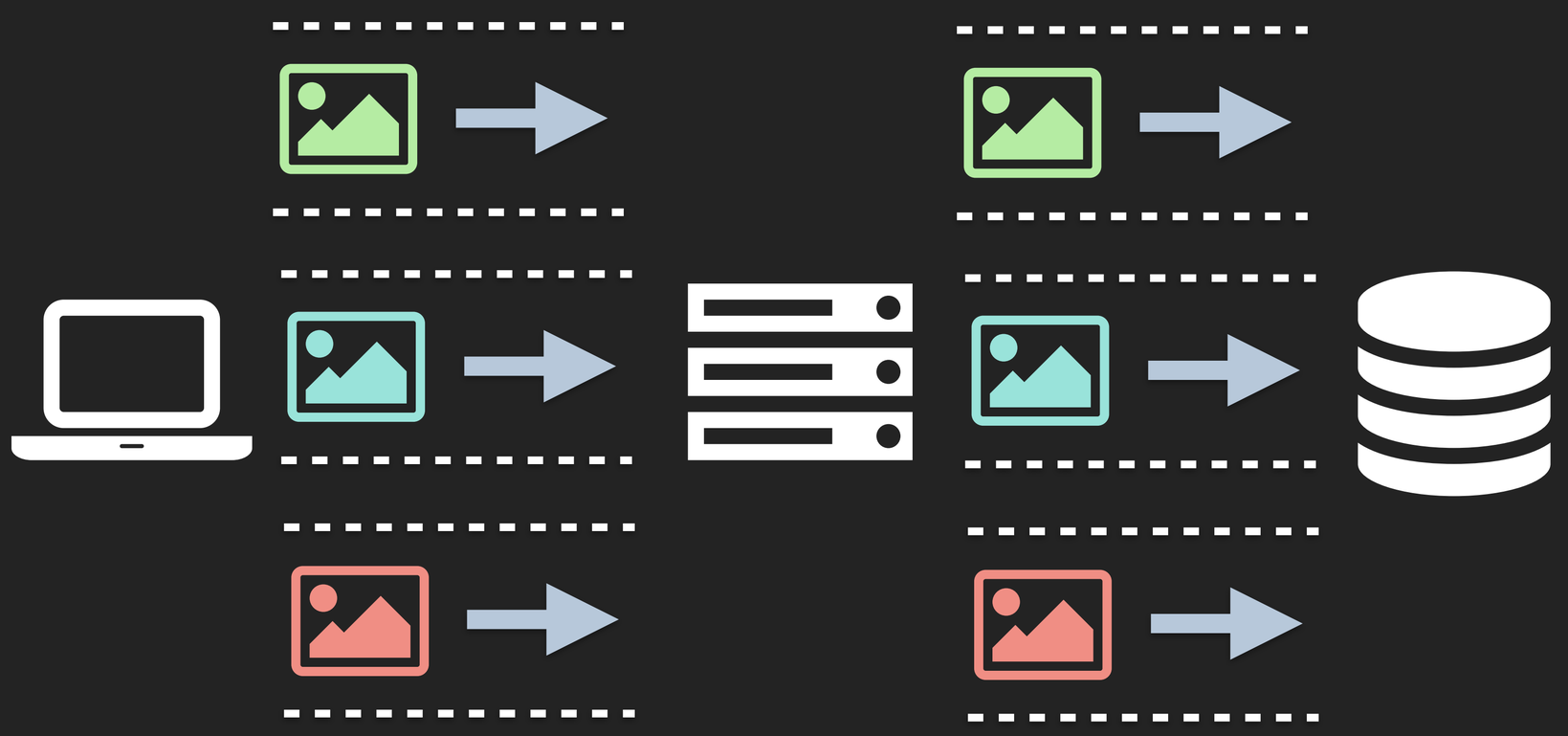
CONCURRENT



CONCURRENT



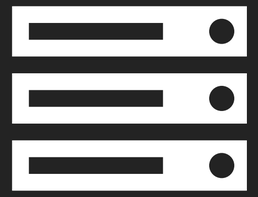
CONCURRENT

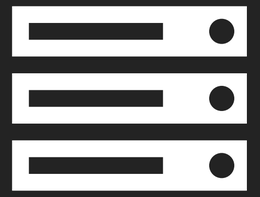


APIS

FETCHING A BLOG POST + COMMENTS

CC-BY: John Trainor

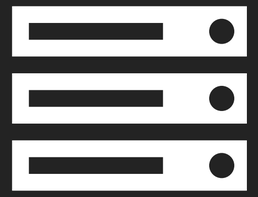


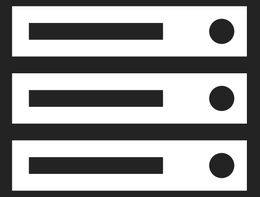


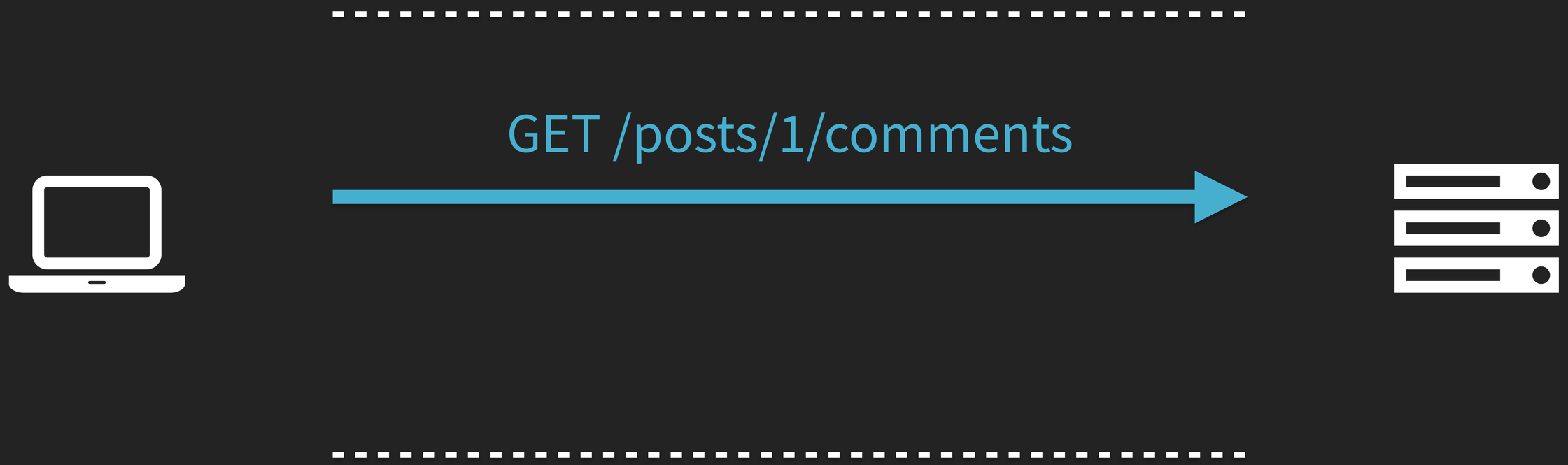
GET /post/1

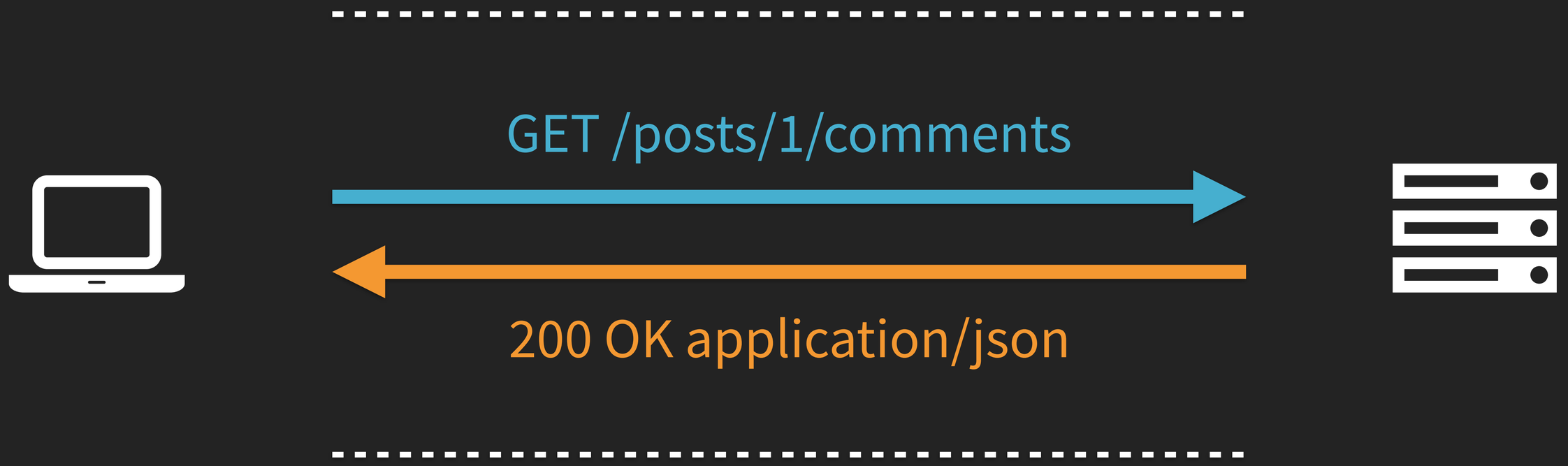
GET /post/1
200 OK application/json

```
{
  "type": "post",
  "id": "1",
  "title": "JSON API paints my bikeshed!",
  "tags": ["json", "api", "relationships"],
  "author": "http://example.com/posts/1/author",
  "comments": "http://example.com/posts/1/comments"
}
```

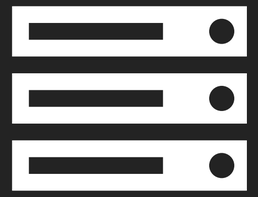


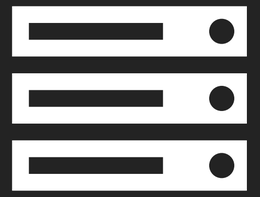



GET /posts/1/comments

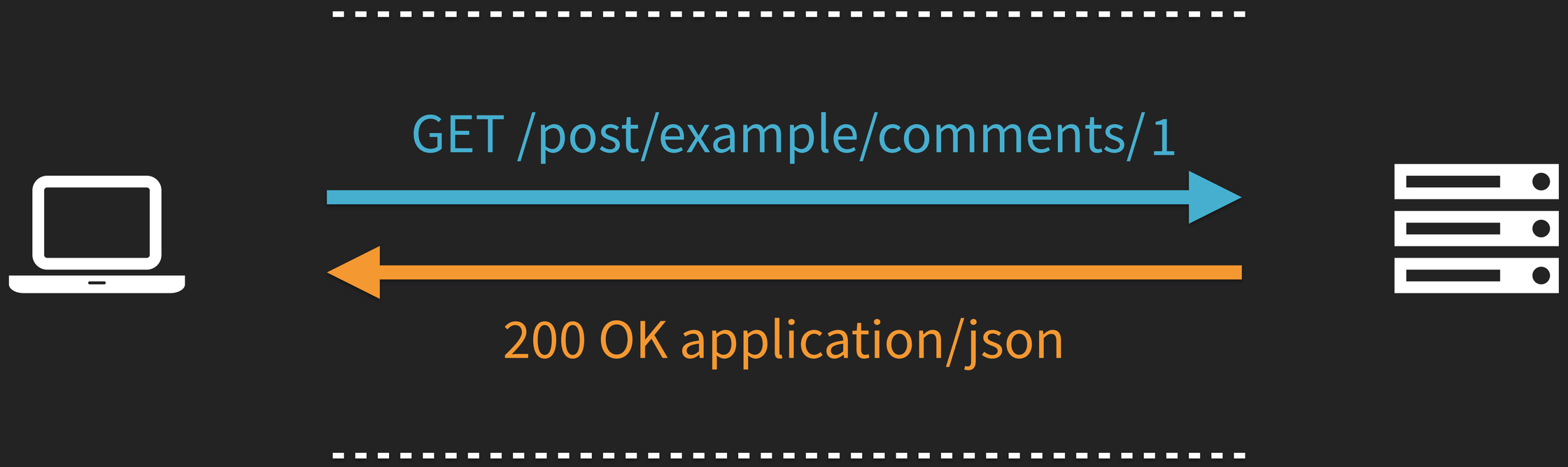

GET /posts/1/comments
200 OK application/json

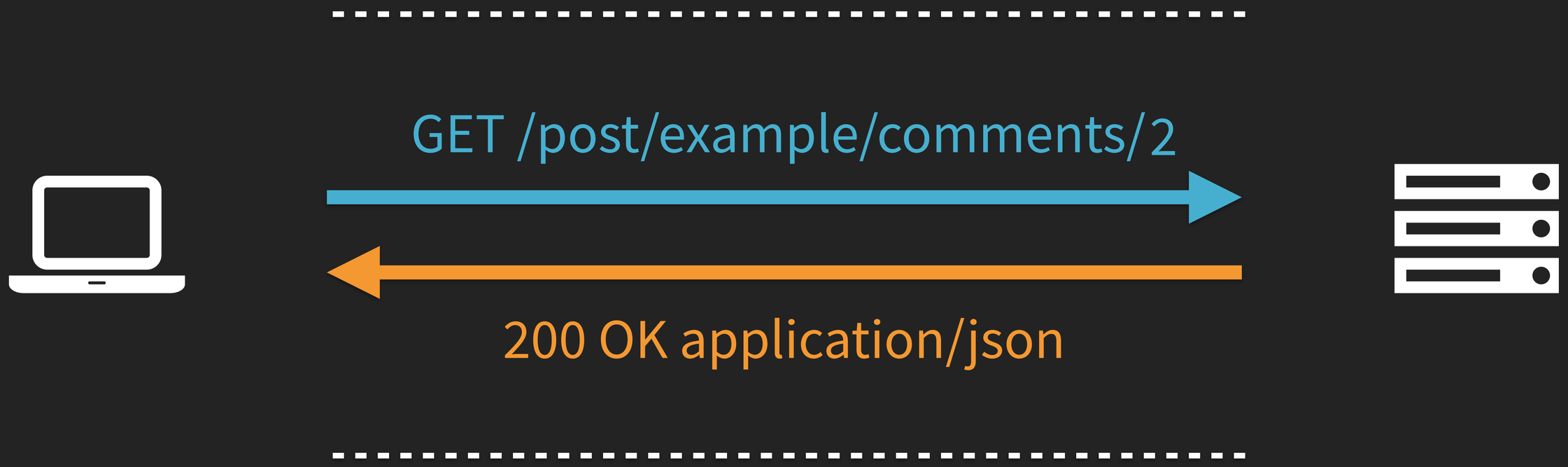
```
[
    "http://example.org/post/1/comment/1",
    "http://example.org/post/1/comment/2",
    "http://example.org/post/1/comment/3",
    "http://example.org/post/1/comment/4"
]
```

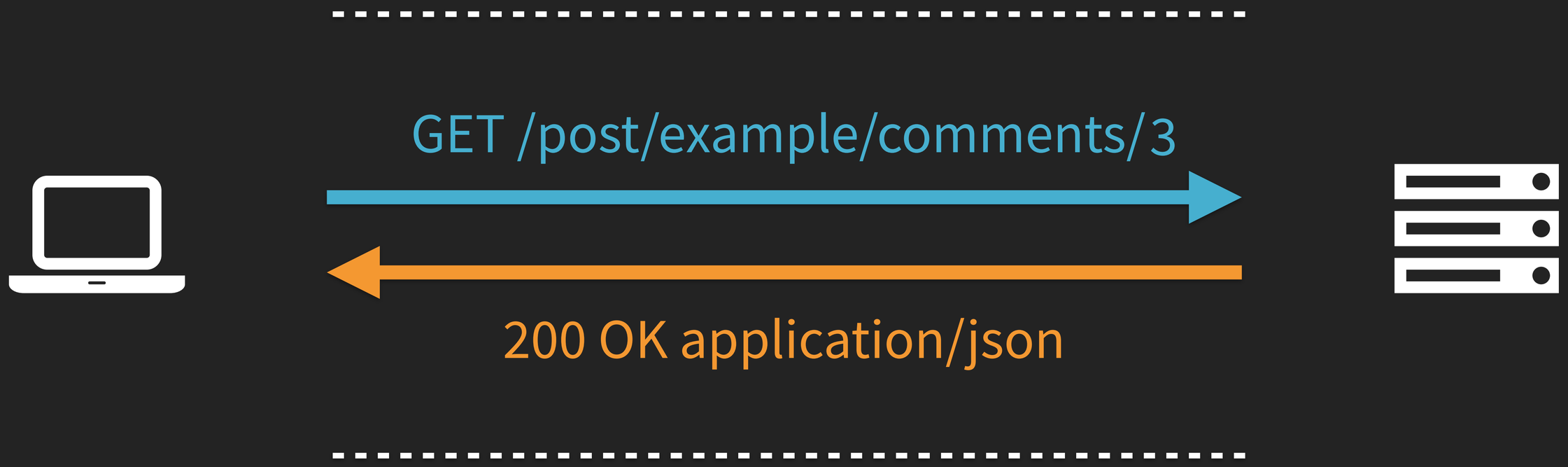


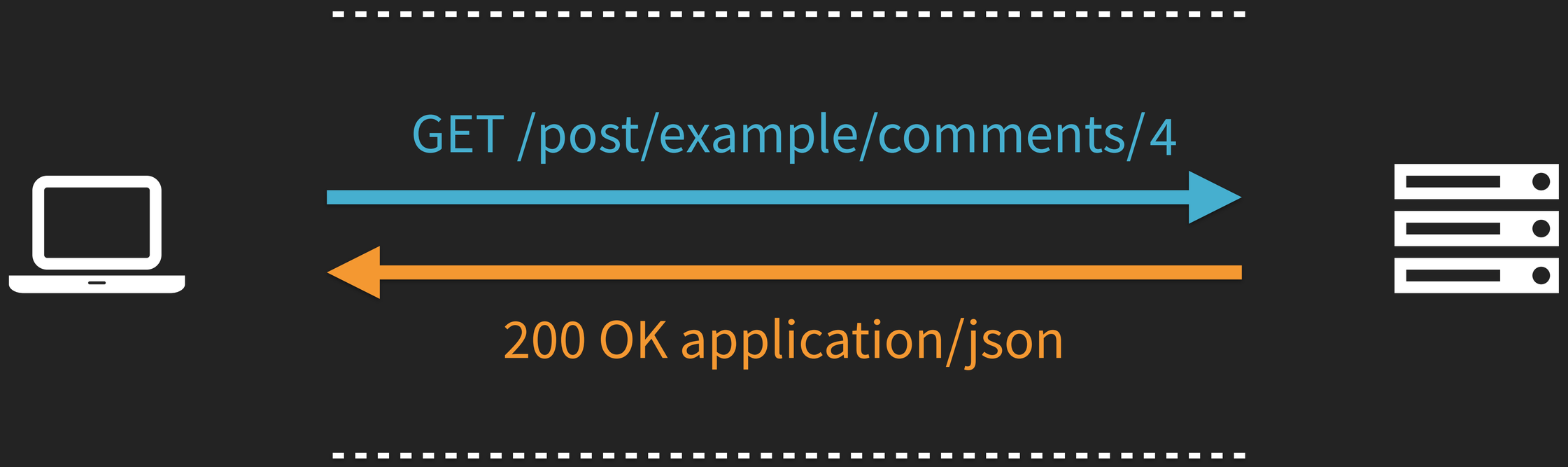


GET /post/example/comments/1


GET /post/example/comments/1
200 OK application/json


GET /post/example/comments/2
200 OK application/json


GET /post/example/comments/3
200 OK application/json


GET /post/example/comments/4
200 OK application/json

```
{
  "type": "comment",
  "id": "1",
  "title": "FIRST!",
  "author": "http://example.com/post/1/comment/1/author"
}
```

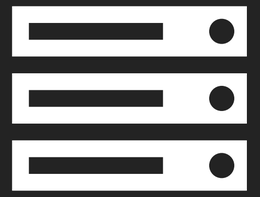
6 requests

(possibly connections)

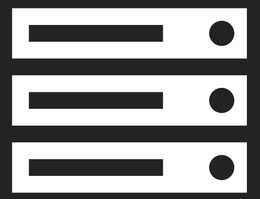
14 requests

(possibly connections)

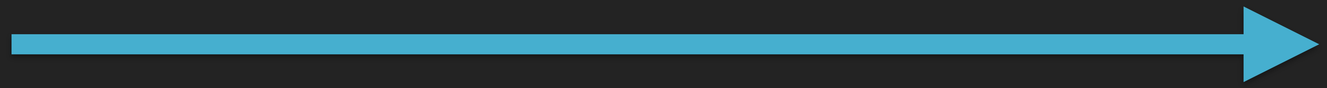
MULTIPLEXED



MULTIPLEXED

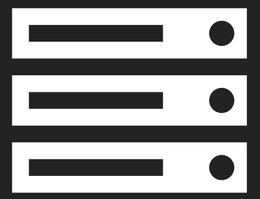


GET /post/example/comments/3



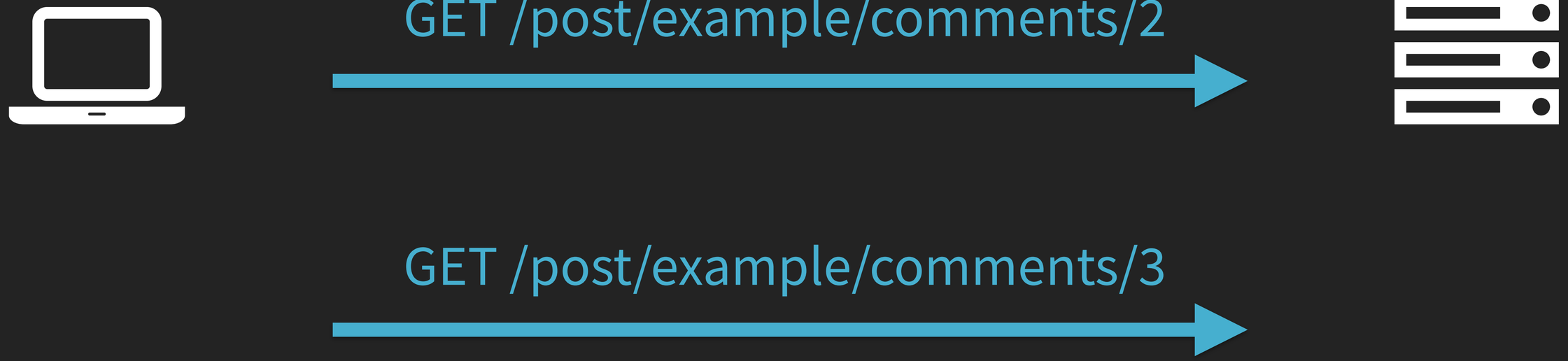
MULTIPLEXED

GET /post/example/comments/1



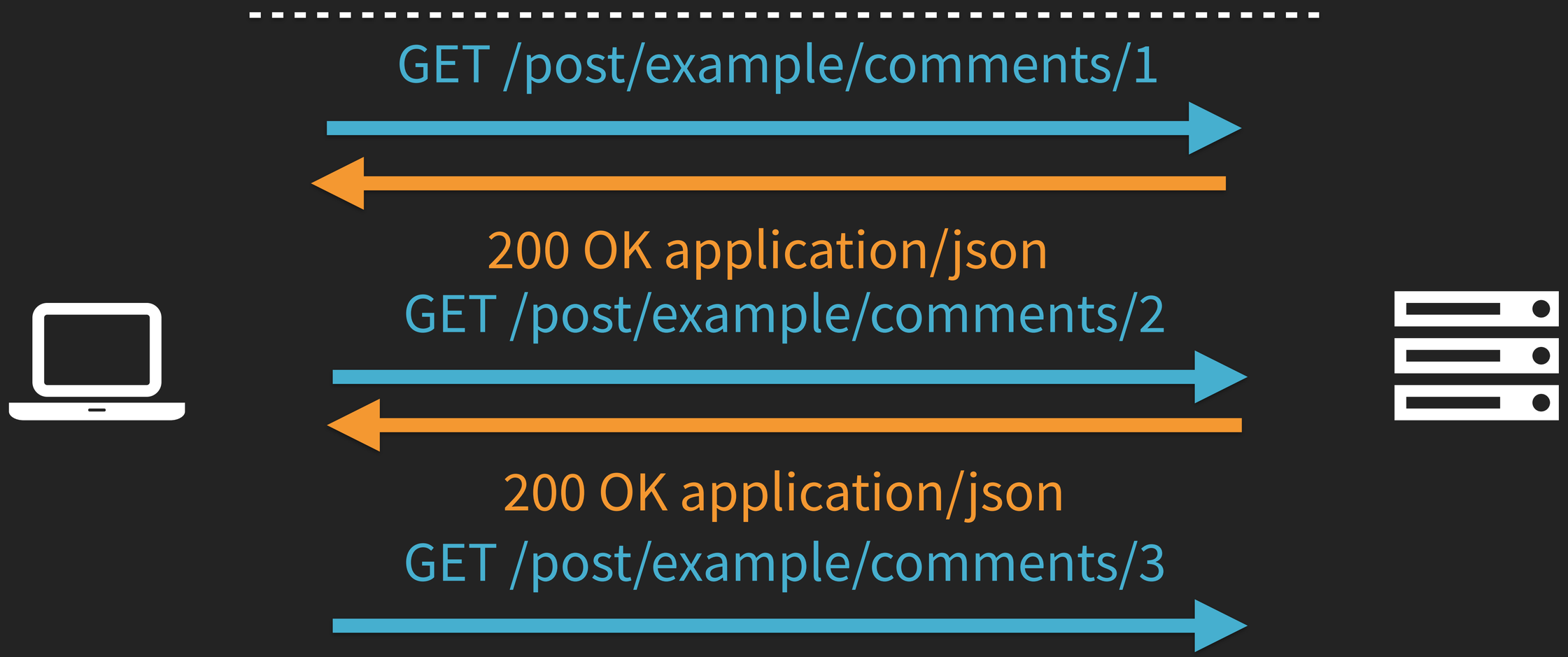
GET /post/example/comments/3

MULTIPLEXED



MULTIPLEXED

MULTIPLEXED



MULTIPLEXED

CODE

```
$numRequests = 378;
```

HTTP/1.1

Latency: **12ms**
Load time: **16.95s**

HTTP/2

Latency: **91ms**
Load time: **3.15s**

Demo concept inspired by Golang's Gophertiles

HTTP/1.1: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);
    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

HTTP/1.1: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);
    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

HTTP/1.1: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);
    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

HTTP/1.1: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);
    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

HTTP/1.1: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);
    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

HTTP/1.1: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);
    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

CC-BY: Hernán Piñera

HTTP/2: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);

    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

HTTP/2: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);
    $opts[CURLOPT_HTTP_VERSION] = HTTP_VERSION_2_0;
    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

HTTP/2: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);

    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

HTTP/2: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);
    $opts[CURLOPT_HTTP_VERSION] = HTTP_VERSION_2_0;
    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

HTTP/2: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);

    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

HTTP/2: SYNCHRONOUS

```
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, $i);
    $opts[CURLOPT_HTTP_VERSION] = HTTP_VERSION_2_0;
    curl_setopt_array($ch, $opts);

    curl_exec($ch);
    curl_close($ch);
}
```

CC-BY-NC: Scott Beckner

HTTP/1.1: CONCURRENT

```
$mh = curl_multi_init();
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $handles[] = $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, '%d');
    curl_setopt_array($ch, $opts);

    curl_multi_add_handle($mh, $ch);
}
```

HTTP/1.1: CONCURRENT

```
$mh = curl_multi_init();
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $handles[] = $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, '%d');
    curl_setopt_array($ch, $opts);

    curl_multi_add_handle($mh, $ch);
}
```

HTTP/1.1: CONCURRENT

```
$mh = curl_multi_init();
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $handles[] = $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, '%d');
    curl_setopt_array($ch, $opts);

    curl_multi_add_handle($mh, $ch);
}
```

HTTP/1.1: CONCURRENT

```
$mh = curl_multi_init();
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $handles[] = $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, '%d');
    curl_setopt_array($ch, $opts);

    curl_multi_add_handle($mh, $ch);
}
```

HTTP/1.1: CONCURRENT (CONT.)

```
do {
   $exec = curl_multi_exec($mh, $running);
} while ($exec == CURLM_CALL_MULTI_PERFORM);

while ($running && $exec == CURLM_OK) {
   $ready = curl_multi_select($mh);
   if ($ready != -1) {
      do {
         $exec = curl_multi_exec($mh, $running);
      } while ($exec == CURLM_CALL_MULTI_PERFORM);
   }
}
```

8.66
seconds

HTTP/2: MULTIPLEXED

```
$mh = curl_multi_init();
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $handles[] = $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, '%d');

    curl_setopt_array($ch, $opts);

    curl_multi_add_handle($mh, $ch);
}
```

HTTP/2: MULTIPLEXED

```
$mh = curl_multi_init();
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $handles[] = $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, '%d');
    $opts[CURLOPT_HTTP_VERSION] = HTTP_VERSION_2_0;
    curl_setopt_array($ch, $opts);

    curl_multi_add_handle($mh, $ch);
}
```

```php
$mh = curl_multi_init();
curl_multi_setopt(
    $mh,
    CURLMOPT_PIPELINING,
    CURLPIPE_MULTIPLEX
);
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $handles[] = $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, '%d');
    $opts[CURLOPT_HTTP_VERSION] = HTTP_VERSION_2_0;
    curl_setopt_array($ch, $opts);

    curl_multi_add_handle($mh, $ch);
}
```

```php
$mh = curl_multi_init();
curl_multi_setopt(
    $mh,
    CURLMOPT_PIPELINING,
    CURLPIPE_MULTIPLEX
);
$url = 'https://http2.akamai.com/demo/tile-%d.png';
for ($i = 0; $i <= $numRequests; $i++) {
    $handles[] = $ch = curl_init();
    $opts[CURLOPT_URL] = sprintf($url, '%d');
    $opts[CURLOPT_HTTP_VERSION] = HTTP_VERSION_2_0;
    curl_setopt_array($ch, $opts);

    curl_multi_add_handle($mh, $ch);
}
```

HTTP/2: MULTIPLEXED (CONT.)

```
do {
   $exec = curl_multi_exec($mh, $running);
} while ($exec == CURLM_CALL_MULTI_PERFORM);

while ($running && $exec == CURLM_OK) {
   $ready = curl_multi_select($mh);
   if ($ready != -1) {
      do {
         $exec = curl_multi_exec($mh, $running);
      } while ($exec == CURLM_CALL_MULTI_PERFORM);
   }
}
```

CC-BY: motoracereports

CC-BY-ND: dhendrix73

BACKWARDS COMPATIBILITY

PROTOCOL IDENTIFIERS

- h2 — HTTP/2 over TLS, negotiated via ALPN (Application-Layer Protocol Negotiation)
- h2c — HTTP/2 over TCP (plain text), uses an HTTP/1.1 request with a `101 Switching Protocols` status response if supported

Application-Layer Protocol Negotiation (ALPN) is a Transport Layer Security (TLS) extension for application layer protocol negotiation. ALPN allows the application layer to negotiate which protocol should be performed over a secure connection in a manner which avoids additional round trips and which is independent of the application layer protocols. It is used by HTTP/2. […] On July 11, 2014, ALPN was published as RFC 7301

– WIKIPEDIA/APPLICATION-LAYER_PROTOCOL_NEGOTIATION

H2C UPGRADE NEGOTIATION

```
> GET / HTTP/1.1
> Host: server.example.com
> Connection: Upgrade, HTTP2-Settings
> Upgrade: h2c
> HTTP2-Settings: <base64url SETTINGS payload>

< HTTP/1.1 101 Switching Protocols
< Connection: Upgrade
< Upgrade: h2c
<
< [ HTTP/2 connection ...
```

H2C UPGRADE NEGOTIATION (CONT.)

- ***May*** be an `OPTIONS` request if concurrency of multiple requests is important
- ***May*** be any request type (e.g. PUT/POST/GET/HEAD/DELETE) but any body must be sent in it's entirety before HTTP/2 can start
- A server **must** ignore an "h2" token in an Upgrade header field. Presence of a token with "h2" implies HTTP/2 over TLS, which is instead negotiated via TLS-ALPN
- It **must** include a settings payload with the initial request

CC-BY: Jason Baker

DIRECT CONNECT WITH HTTP/2

PRIOR KNOWLEDGE

- It is *possible* to set up a connection with HTTP/1.1 or ALPN negotiation when prior knowledge of HTTP/2 is known
- Performance enhancement
- Client/Server **must** send the HTTP/2 connection prefix
- **Not supported in curl yet**

CC-BY: Steven Depolo

MULTIPLEXED

STILL MAKING SUB-REQUESTS

```
{
  "type": "post",
  "id": "1",
  "title": "JSON API paints my bikeshed!",
  "tags": ["json", "api", "relationships"],
  "author": {
      "type": "author",
      "id": 1,
      "name": "Davey Shafik",
      "twitter": "@dshafik",
      "gravatar": "fee39f0c0ffb29d9ac21607ed188be6b"
  },
  "comments": [
    {
      "type": "comment",
      "id": "1",
      "content": "FIRST!",
      "author": {
        "type": "author",
        "id": 2,
        "name": "Jill Random",
        "gravatar": "706b16b2fb732ab6079a10fea61d078b"
      }
    },
    {
      "type": "comment",
      "id": "2",
      "content": "second",
      "author": {
        "type": "author",
        "id": 3,
        "name": "John Person",
        "gravatar": "8e2279479945ca4778eb3cb8d88cda58"
      }
    },
    {
      "type": "comment",
      "id": "3",
      "content": "third",
      "author": {
        "type": "author",
        "id": 1,
        "name": "Davey Shafik",
        "twitter": "@dshafik",
        "gravatar": "fee39f0c0ffb29d9ac21607ed188be6b"
      }
    },
    {
      "type": "comment",
      "id": "4",
      "content": "fourth",
      "author": {
        "type": "author",
        "id": 2,
        "name": "Jill Random",
        "gravatar": "706b16b2fb732ab6079a10fea61d078b"
      }
    }
  ]
}
```

```
{
  "type": "post",
  "id": "1",
  "title": "JSON API paints my bikeshed!",
  "tags": ["json", "api", "relationships"],
  "author": {
      "type": "author",
      "id": 1,
      "name": "Davey Shafik",
      "twitter": "@dshafik",
      "gravatar": "fee39f0c0ffb29d9ac21607ed188be6b"
  },
  "comments": [
    {
      "type": "comment",
      "id": "1",
```

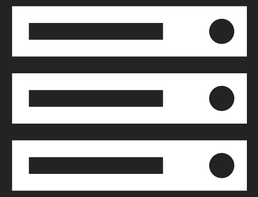
```
{
  "type": "post",
  "id": "1",
  "title": "JSON API paints my bikeshed!",
  "tags": ["json", "api", "relationships"],
  "author": {
      "type": "author",
      "id": 1,
      "name": "Davey Shafik",
      "twitter": "@dshafik",
      "gravatar": "fee39f0c0ffb29d9ac21607ed188be6b"
  },
  "comments": [
    {
      "type": "comment",
      "id": "1",
```

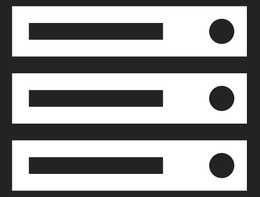
```
  },
  "comments": [
    {
      "type": "comment",
      "id": "1",
      "content": "FIRST!",
      "author": {
        "type": "author",
        "id": 2,
        "name": "Jill Random",
        "gravatar": "706b16b2fb732ab6079a10fea61d078b"
      }
    },
    {
      "type": "comment",
      "id": "2",
```

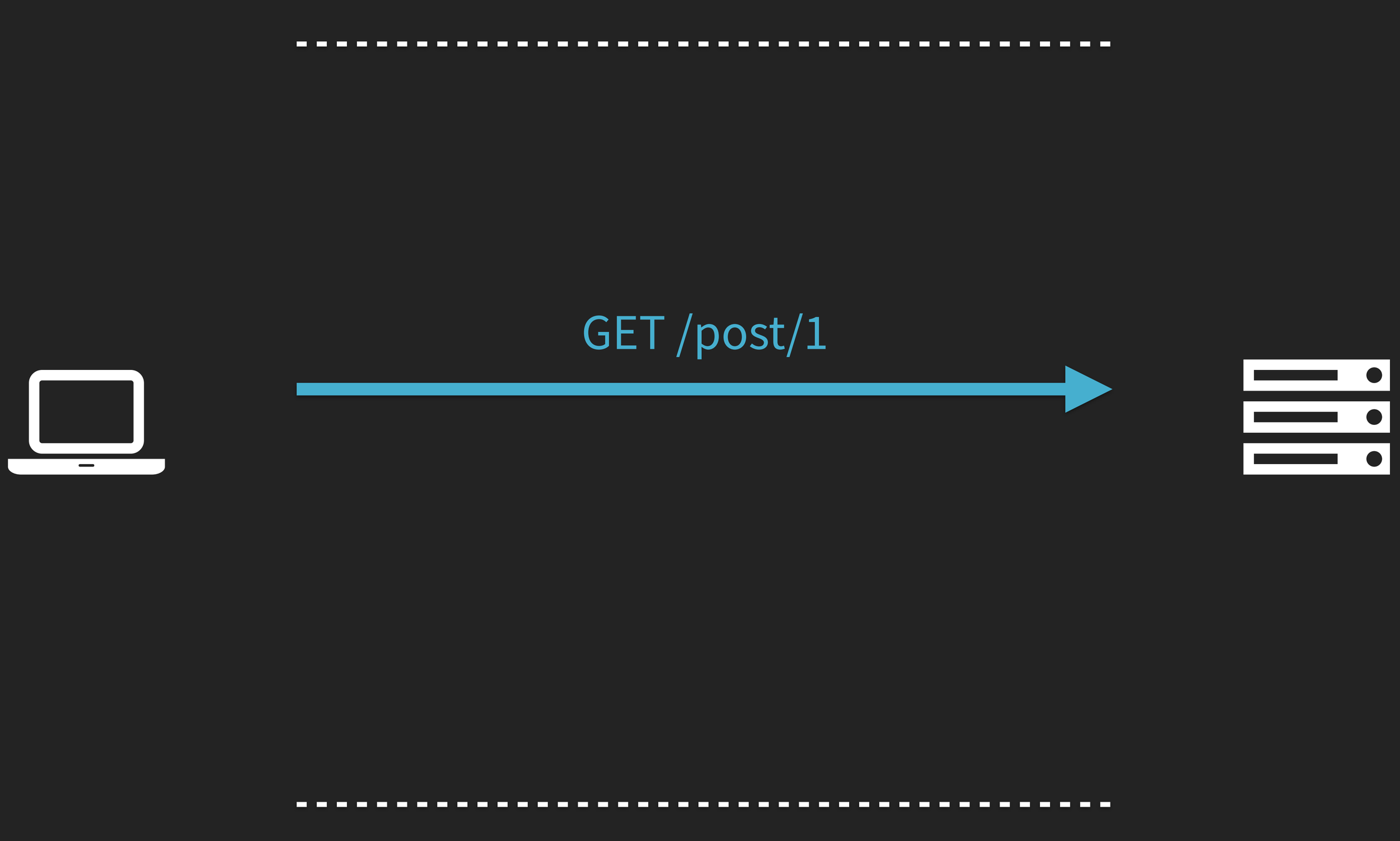
```
  },
  {
    "type": "comment",
    "id": "3",
    "content": "third",
    "author": {
      "type": "author",
      "id": 1,
      "name": "Davey Shafik",
      "twitter": "@dshafik",
      "gravatar": "fee39f0c0ffb29d9ac21607ed188be6b"
    }
  },
  {
    "type": "comment",
    "id": "4",
```

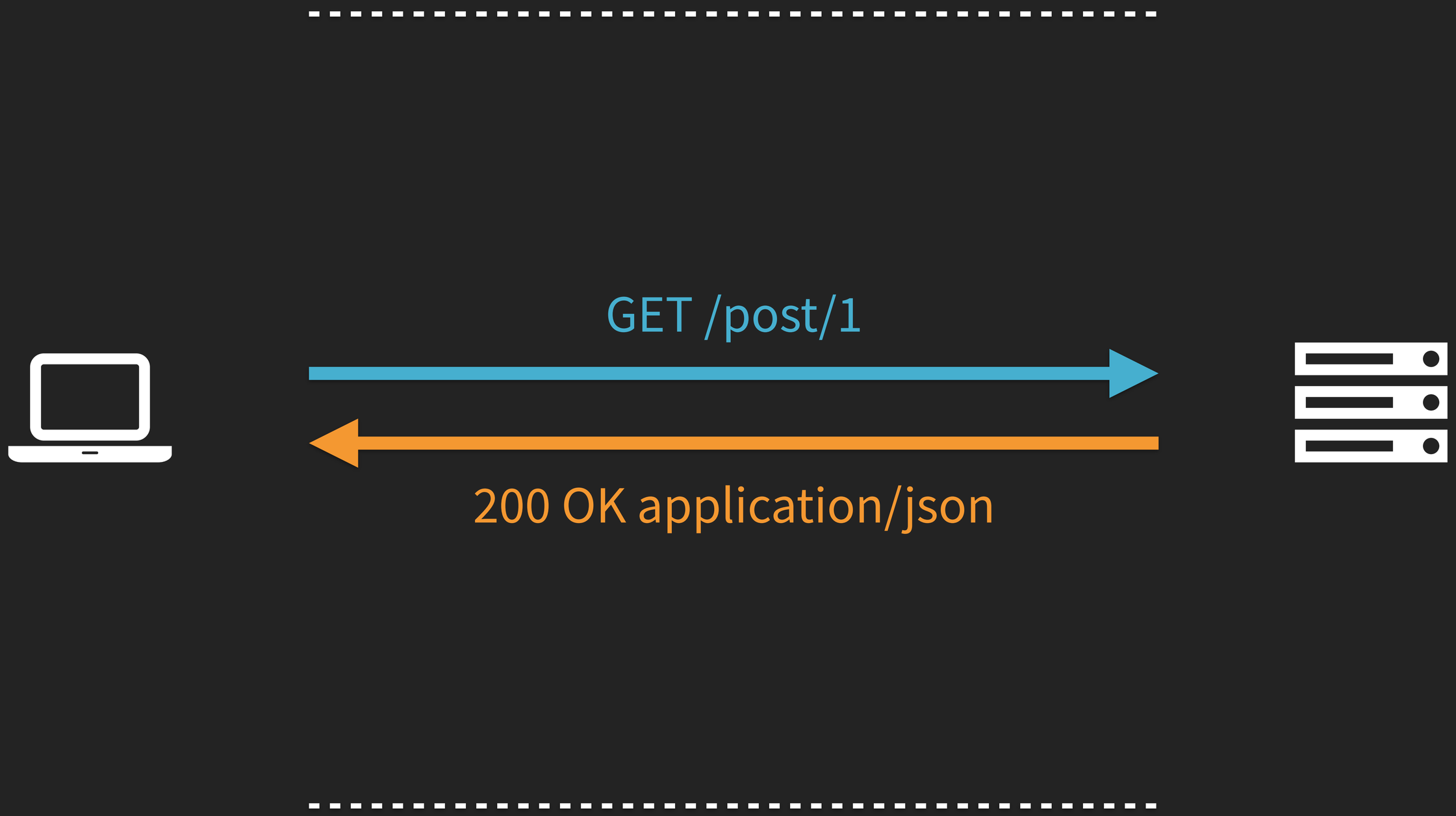
```
    },
    {
      "type": "comment",
      "id": "3",
      "content": "third",
      "author": {
        "type": "author",
        "id": 1,
        "name": "Davey Shafik",
        "twitter": "@dshafik",
        "gravatar": "fee39f0c0ffb29d9ac21607ed188be6b"
      }
    },
    {
      "type": "comment",
      "id": "4",
```

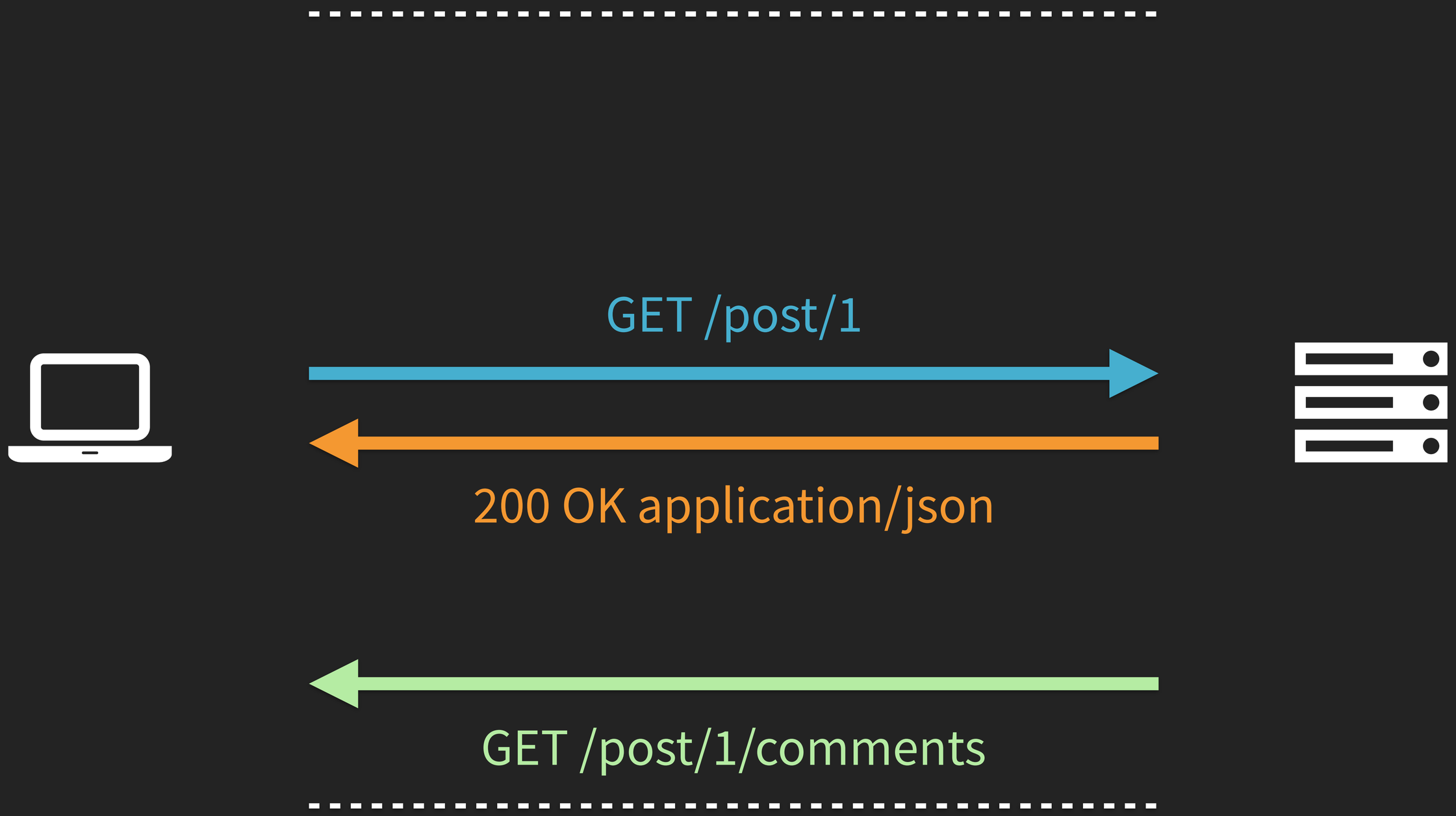
```
      }
    },
    {
      "type": "comment",
      "id": "4",
      "content": "fourth",
      "author": {
        "type": "author",
        "id": 2,
        "name": "Jill Random",
        "gravatar": "706b16b2fb732ab6079a10fea61d078b"
      }
    }
  ]
}
```

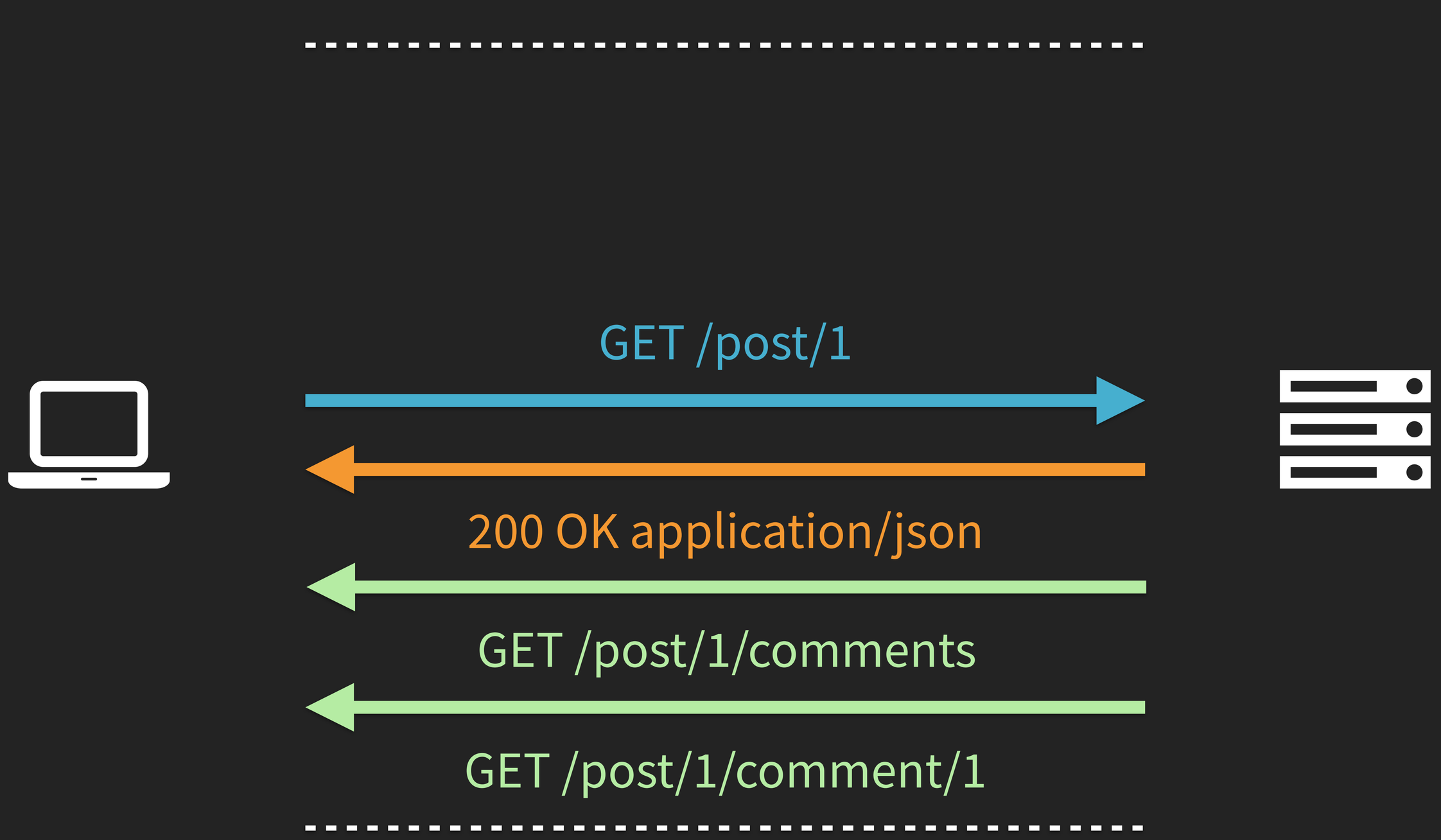


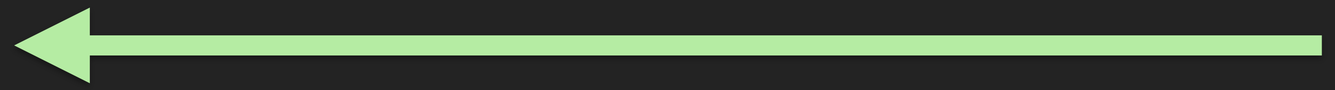



GET /post/1

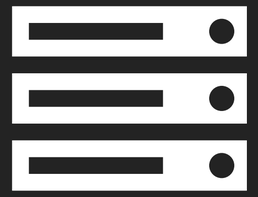

GET /post/1
200 OK application/json

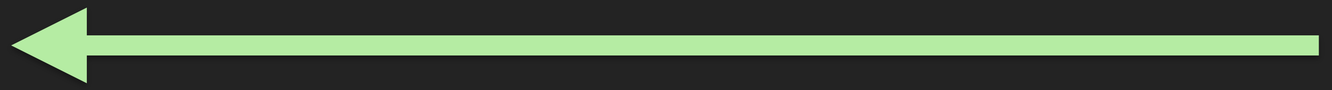

GET /post/1
200 OK application/json
GET /post/1/comments


GET /post/1
200 OK application/json
GET /post/1/comments
GET /post/1/comment/1

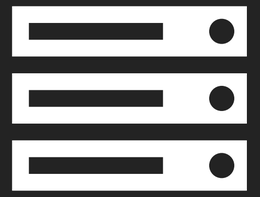


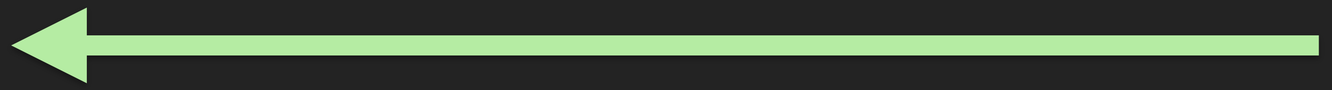
GET /post/1/comments
GET /post/1/comment/1



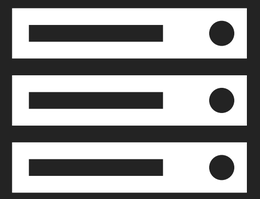


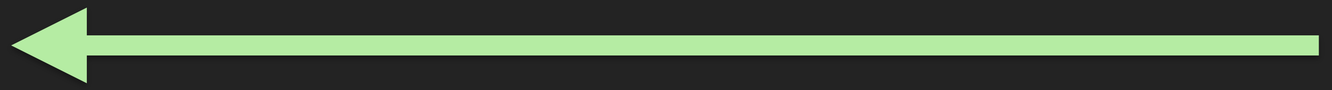
GET /post/1/comments
GET /post/1/comment/1
GET /post/1/comment/2
GET /post/1/comment/3
GET /post/1/comment/4





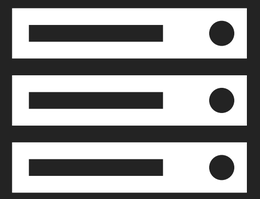
GET /post/1/comments
GET /post/1/comment/1
GET /post/1/comment/2
GET /post/1/comment/3
GET /post/1/comment/4
GET /post/1/comment/1/author
GET /post/1/comment/2/author
GET /post/1/comment/3/author
GET /post/1/comment/4/author





GET /post/1/comments
GET /post/1/comment/1
GET /post/1/comment/2
GET /post/1/comment/3
GET /post/1/comment/4

GET /post/1/comment/1/author
GET /post/1/comment/2/author
GET /post/1/comment/3/author
GET /post/1/comment/4/author



GET /post/1/comment/1/author/avatar.png
GET /post/1/comment/2/author/avatar.png
GET /post/1/comment/3/author/avatar.png
GET /post/1/comment/4/author/avatar.png

CC-BY: Patrick Metzdorf

CC-BY: Kylir Horton

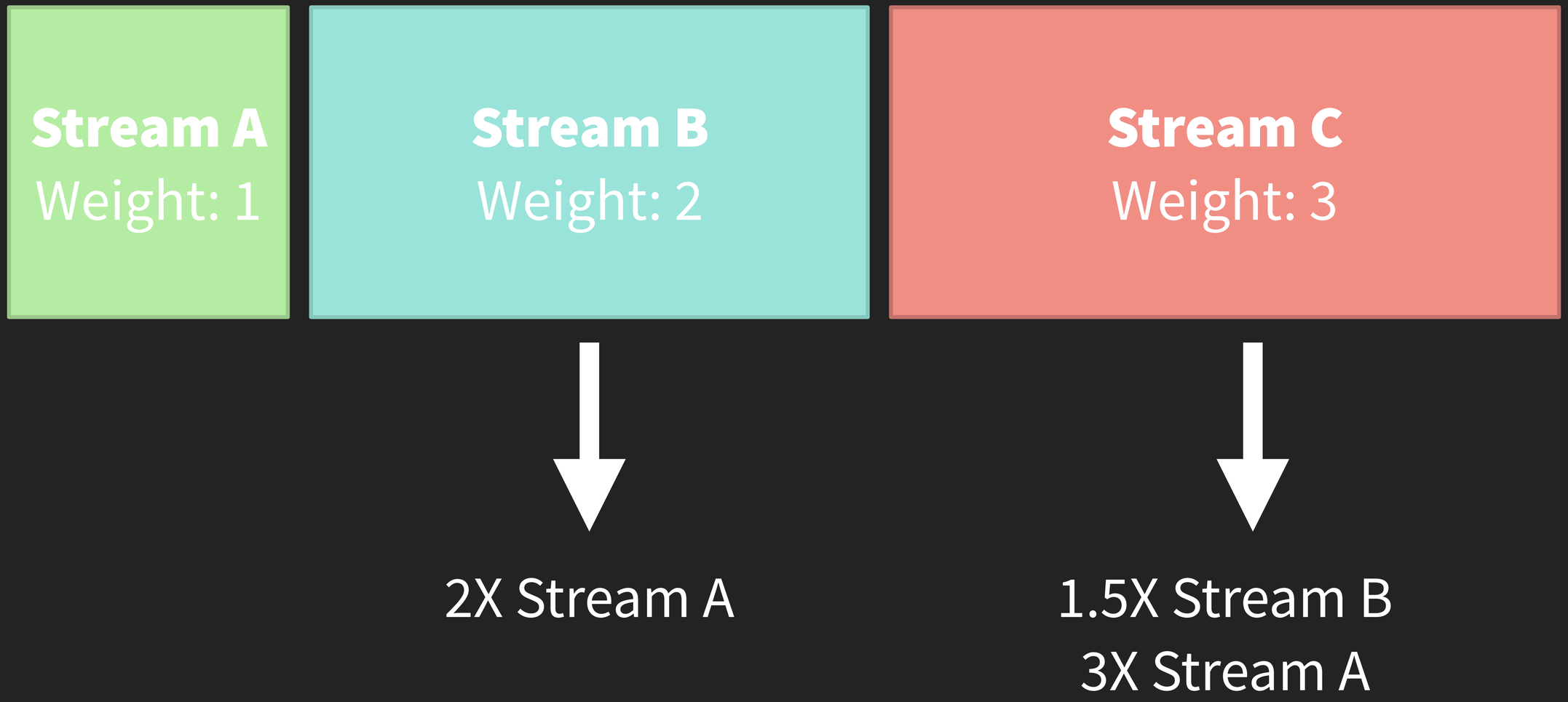
STREAMS

- Each request/response is a stream
- Streams are comprised of Frames
- Streams may have a weight (1-256)
- Streams may have a dependency

WEIGHTS

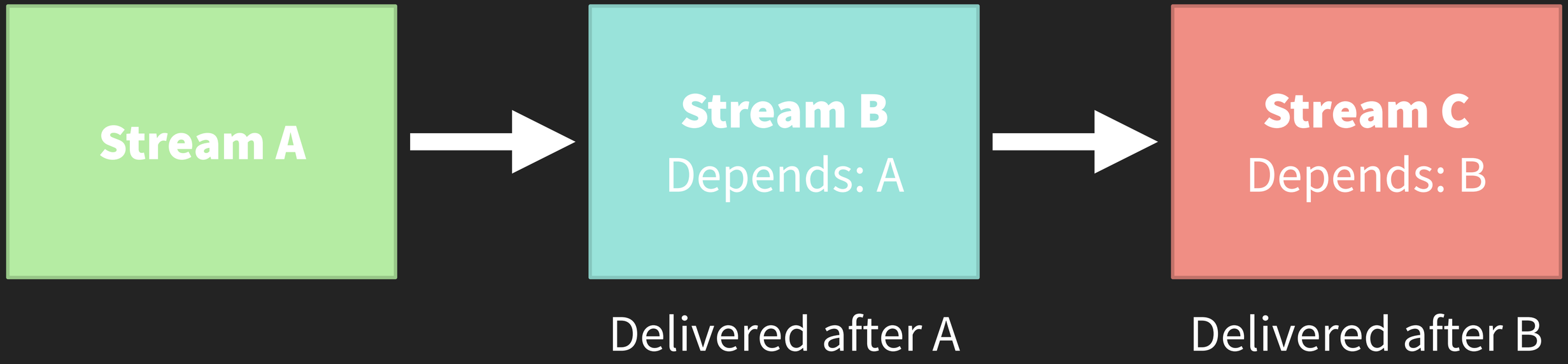
CC-BY 2.0: Tri Vu Dao

STREAM WEIGHTS



CC-BY-SA 2.0: David Gamez

STREAM DEPENDENCIES



CC-BY: Bart Everson

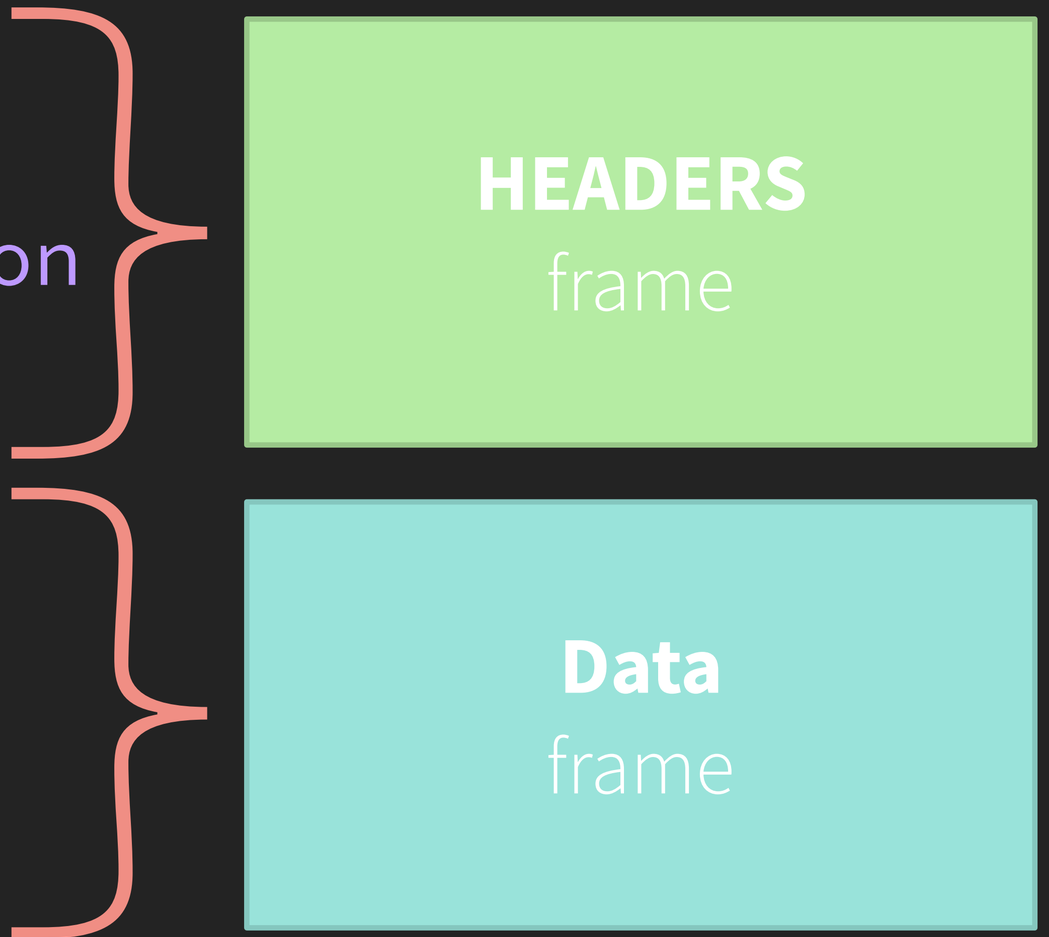
FRAMES

- Messages are composed of multiple frames, e.g. headers, data, and settings
- Each frame has a common header
  - 9 byte, length prefixed
  - Easy & efficient to parse
- Frames can be interleaved — this is multiplexing

FRAMES

```
POST /search HTTP/1.1
Host: example.org
Content-Type: application/json
Content-Length: 58

{
    "keywords":"example",
    "location":"posts"
}
```



HEADERS: HPACK

CC-BY-SA: intelligente_persona

HEADERS: HPACK

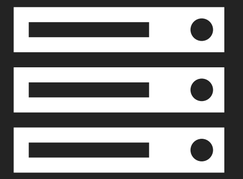
- Uses a table of known values as an index
- Can represent a header name and value (e.g. `:status: 404`), or just a header name (e.g. `accept:`)
- Values are either statically encoded, or use a static Huffman code

#	Name	Value
1	:authority	
2	:method	GET
3	:method	POST
4	:path	/
5	:path	/index.html
6	:scheme	http
7	:scheme	https
8	:status	200
9	:status	204
10	:status	206
11	:status	304
12	:status	400
13	:status	404
14	:status	500
15	accept-charset	

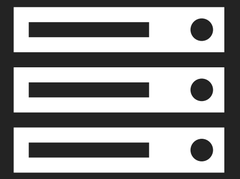
NEW ARCHITECTURE & TECHNIQUES

DELIVERING A PAGE

PAGE ASSETS

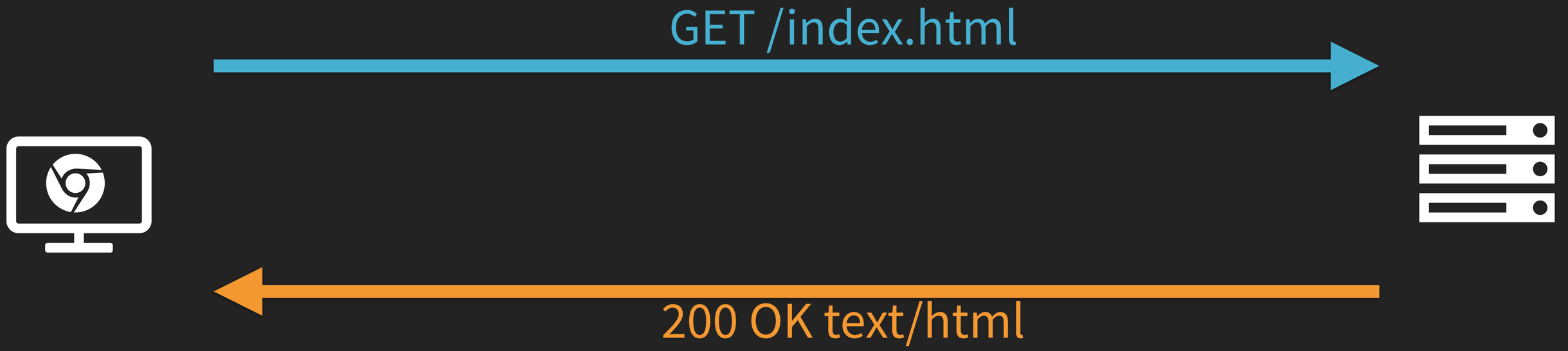


PAGE ASSETS



PAGE ASSETS

PAGE ASSETS

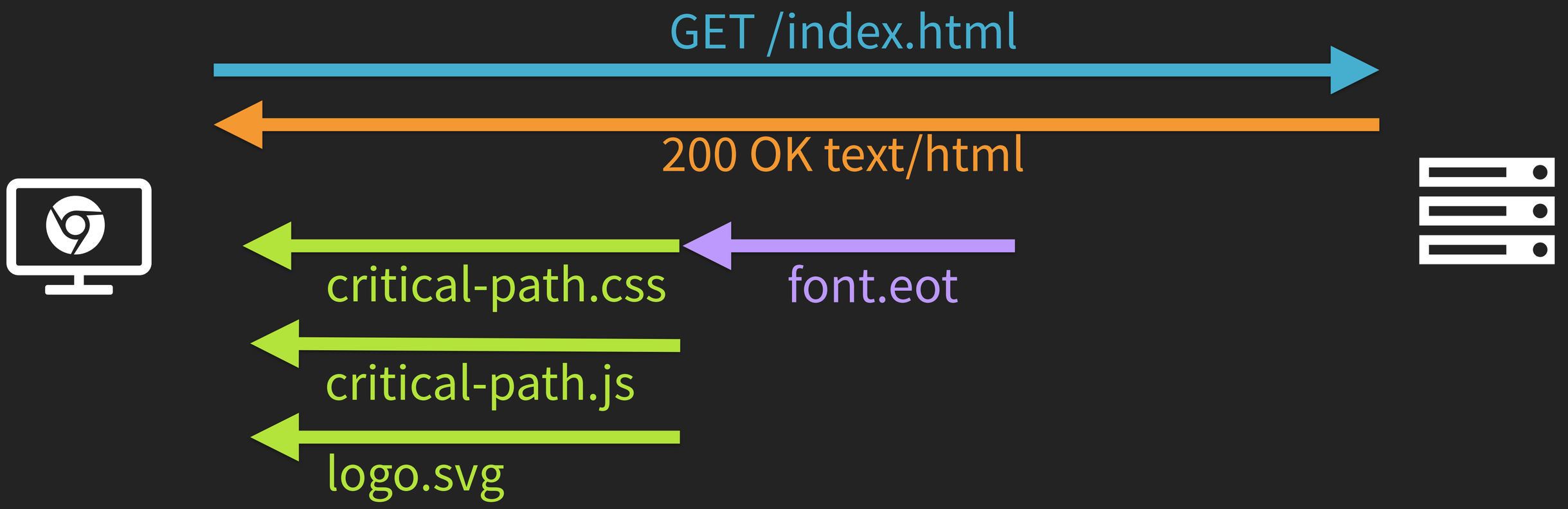


PAGE ASSETS

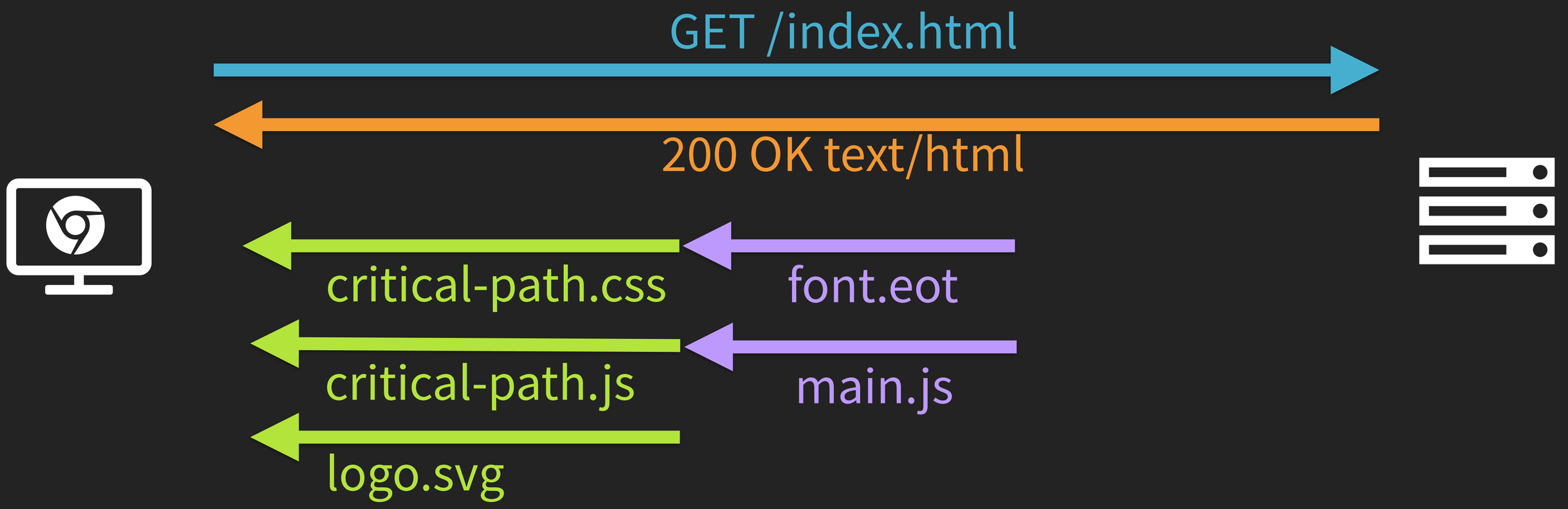
PAGE ASSETS

GET /index.html
200 OK text/html
critical-path.css
critical-path.js
logo.svg

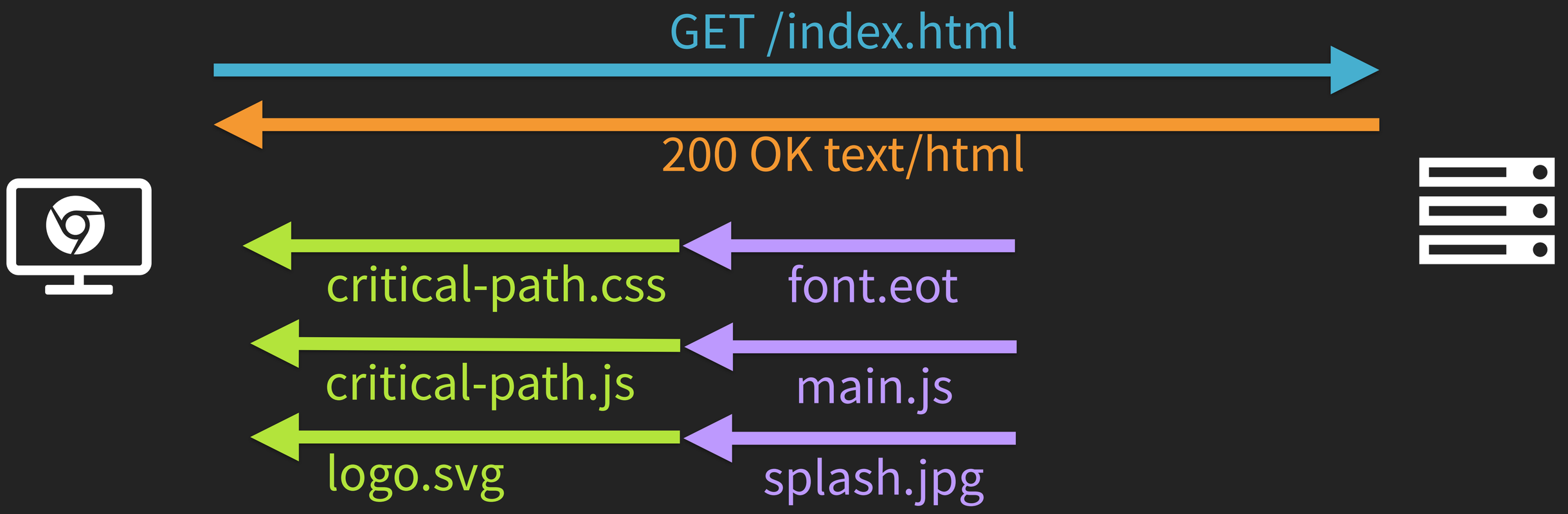
PAGE ASSETS


GET /index.html
200 OK text/html
critical-path.css
font.eot
critical-path.js
logo.svg

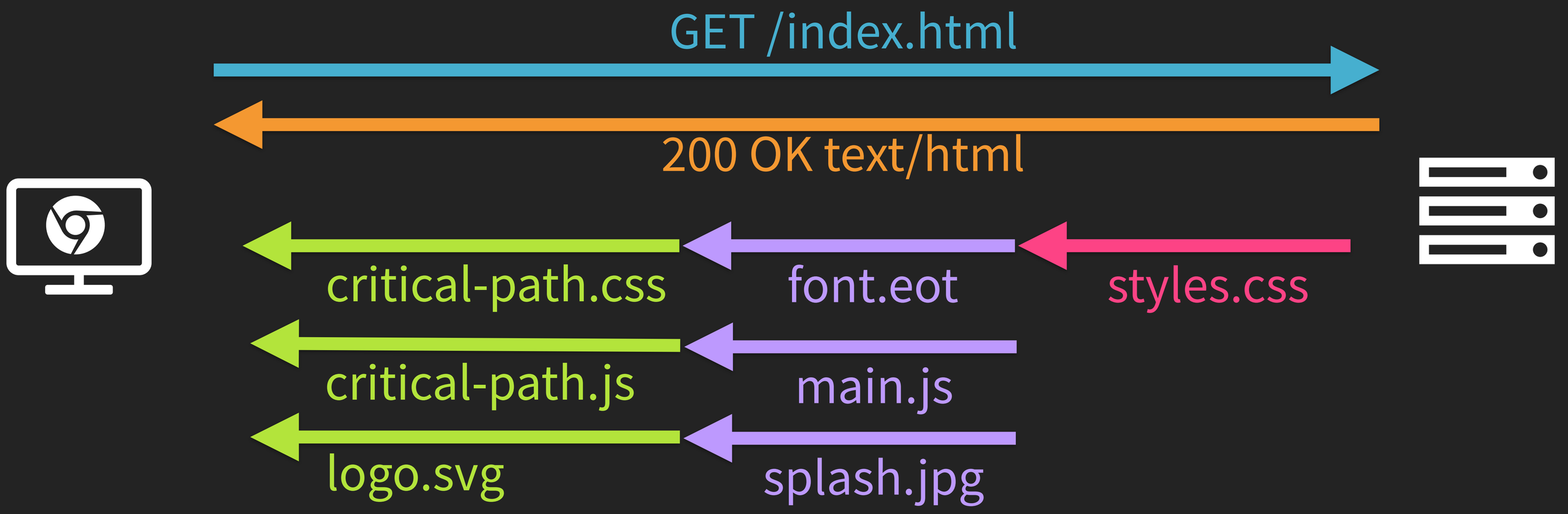
PAGE ASSETS


GET /index.html
200 OK text/html
critical-path.css
font.eot
critical-path.js
main.js
logo.svg

PAGE ASSETS

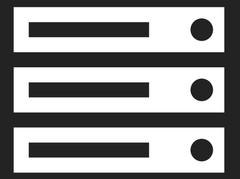

GET /index.html
200 OK text/html
critical-path.css
font.eot
critical-path.js
main.js
logo.svg
splash.jpg

PAGE ASSETS

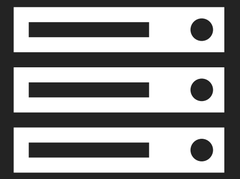

GET /index.html
200 OK text/html
critical-path.css
font.eot
styles.css
critical-path.js
main.js
logo.svg
splash.jpg

APIs

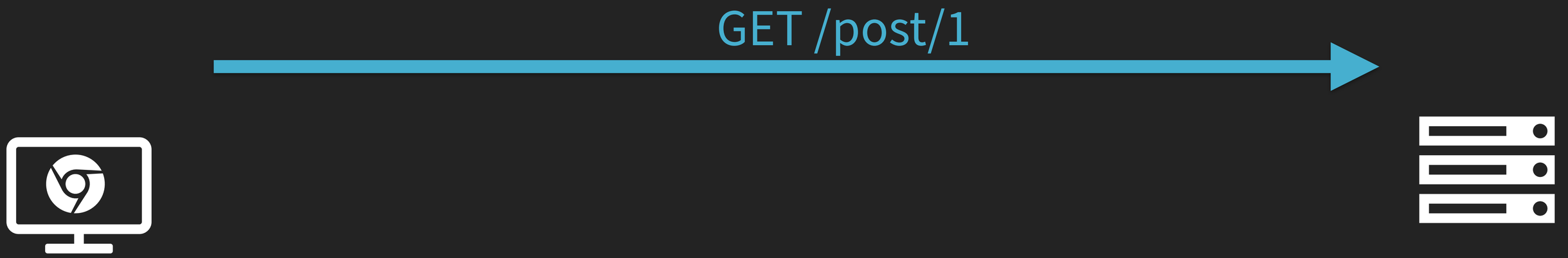
API REQUEST



API REQUEST

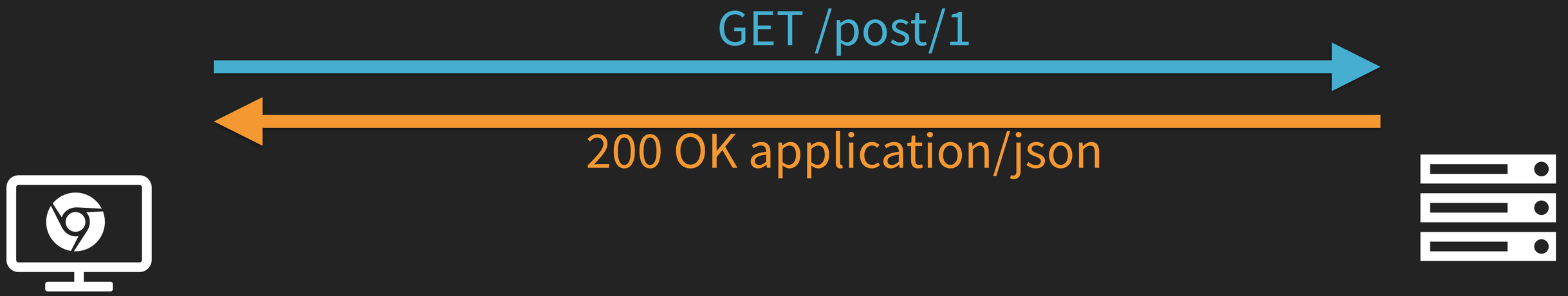


API REQUEST



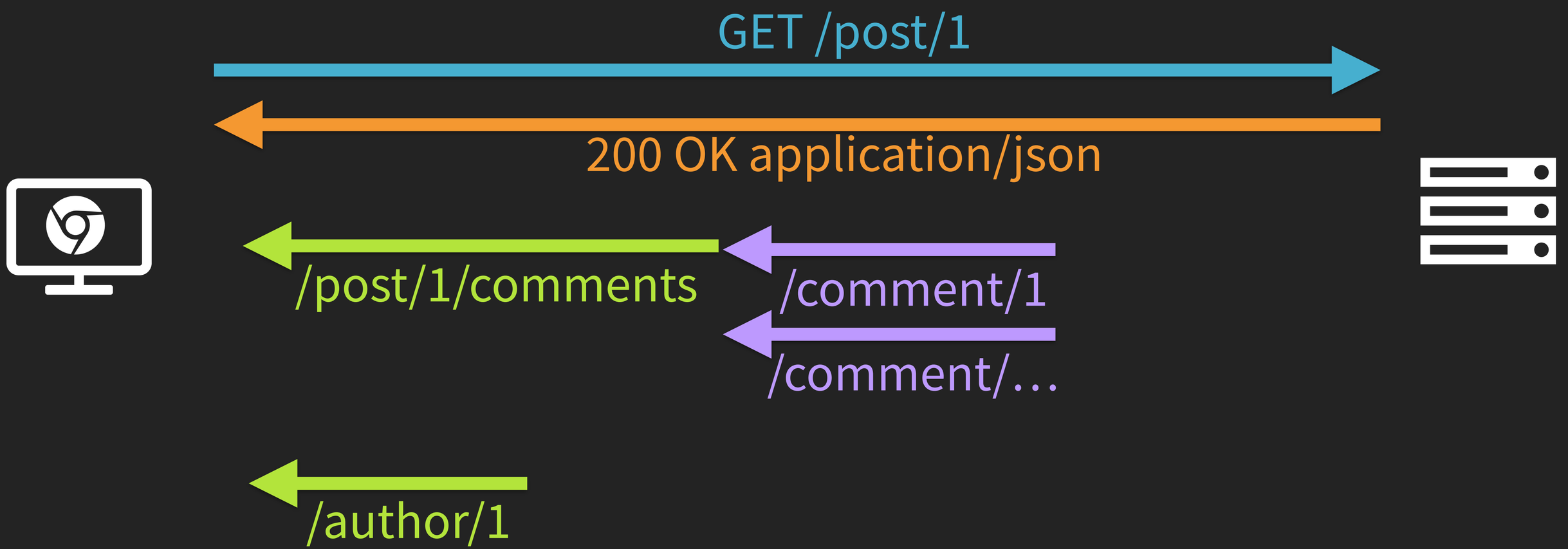
API REQUEST

SUB-RESOURCE PUSH WITH DEPENDENCIES



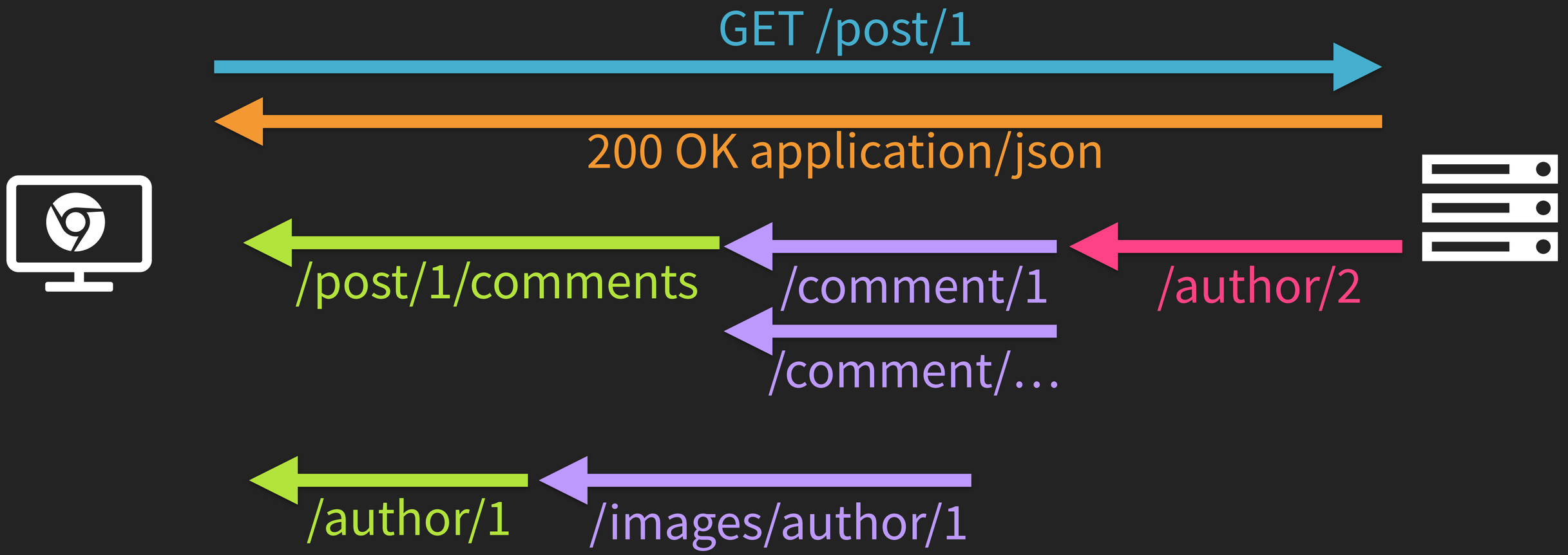
SUB-RESOURCE PUSH WITH DEPENDENCIES

SUB-RESOURCE PUSH WITH DEPENDENCIES

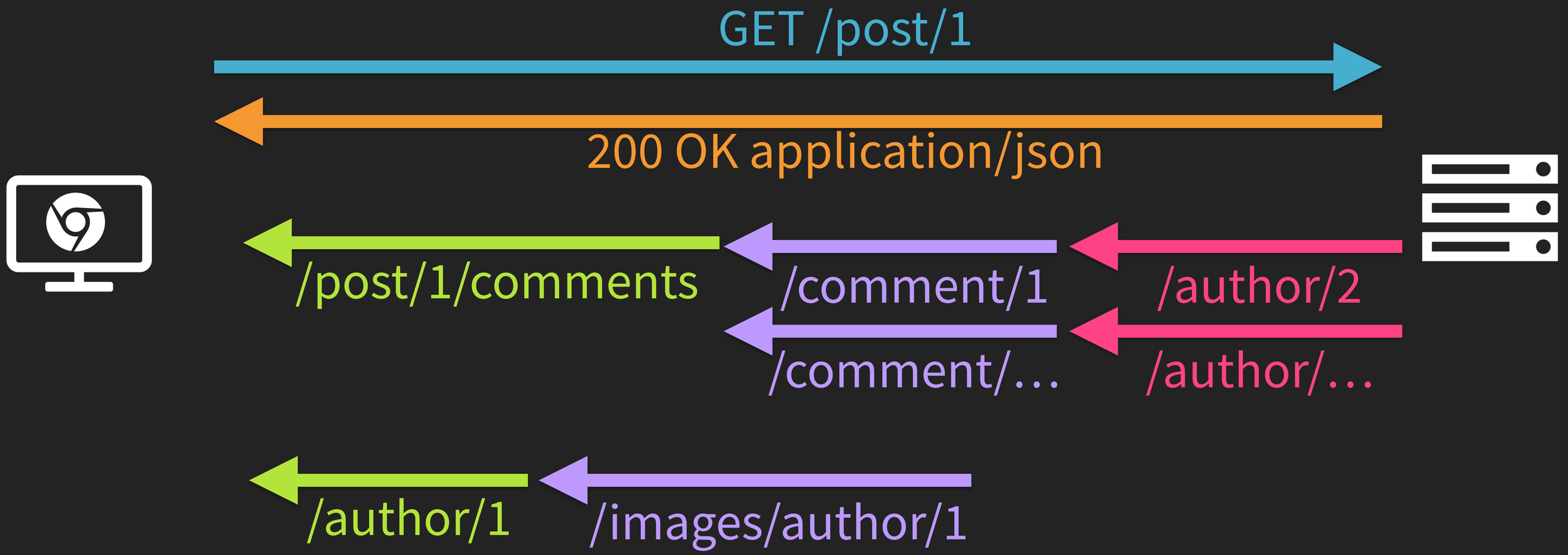


SUB-RESOURCE PUSH WITH DEPENDENCIES

SUB-RESOURCE PUSH WITH DEPENDENCIES



SUB-RESOURCE PUSH WITH DEPENDENCIES



IN SUMMARY

NO MORE HACKS!

HUGE PERFORMANCE WINS

60%+ MARKET SHARE

HTTP/2 IS AWESOME!
CC-BY-SA: Steven Gerner

FEEDBACK & QUESTIONS

Twitter: @dshafik

Email: dshafik@akamai.com

Slides: http://daveyshafik.com/slides