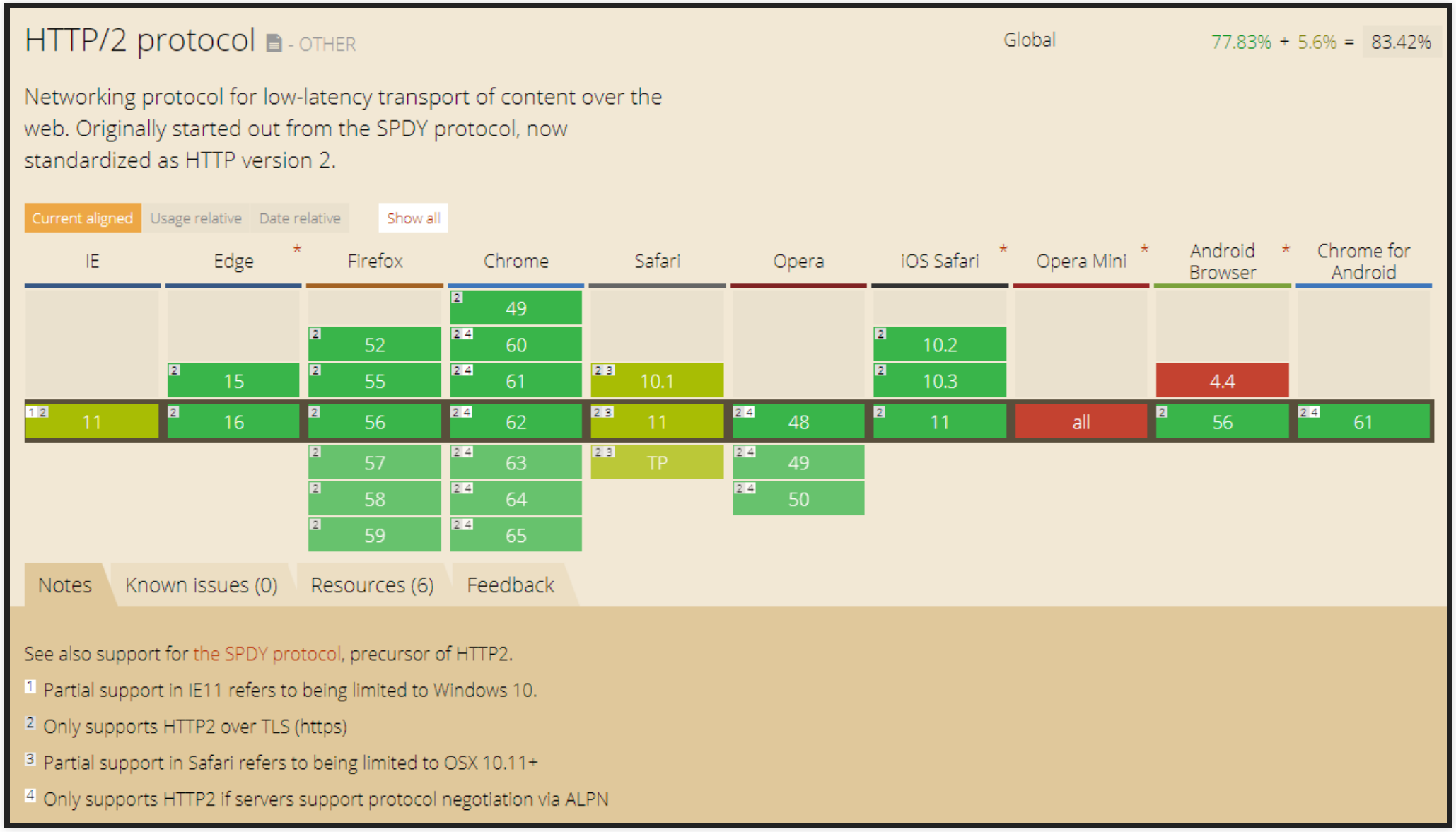
HTTP/2



https://caniuse.com/#feat=http2

HTTP/1.1 → SPDY → HTTP/2

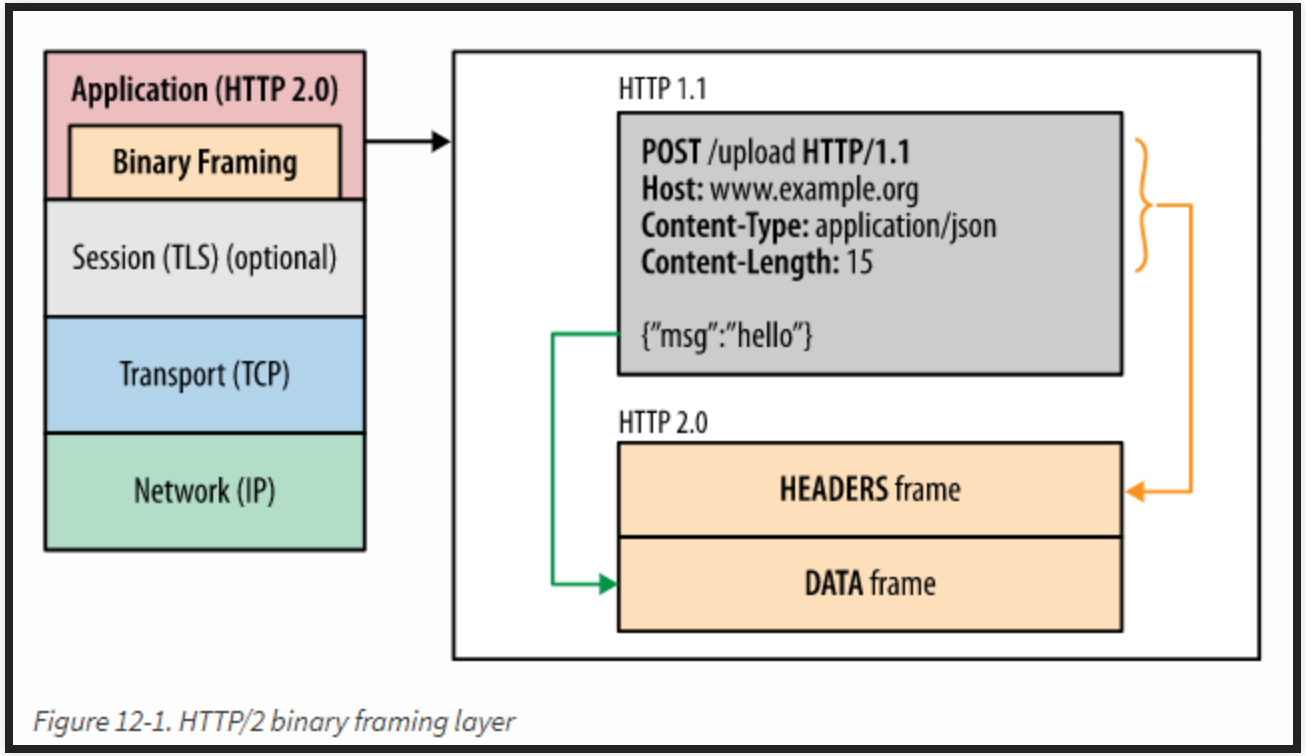


Figure 12-1. HTTP/2 binary framing layer

https://hpbn.co/http2/

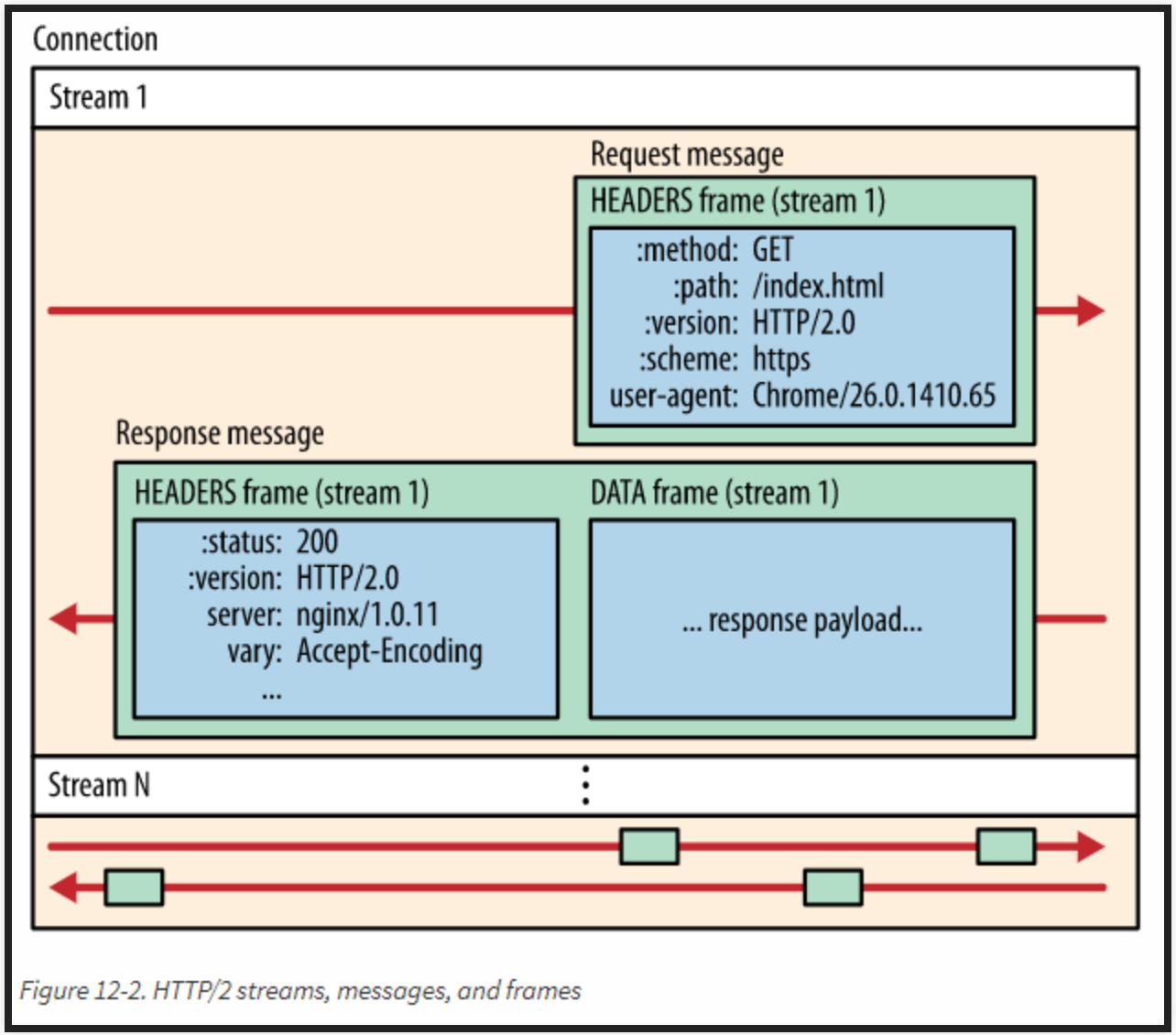


Figure 12-2. HTTP/2 streams, messages, and frames

https://hpbn.co/http2/

- HPACK
- multiplexing
- server push

HPACK Compression – Huffman Encoding

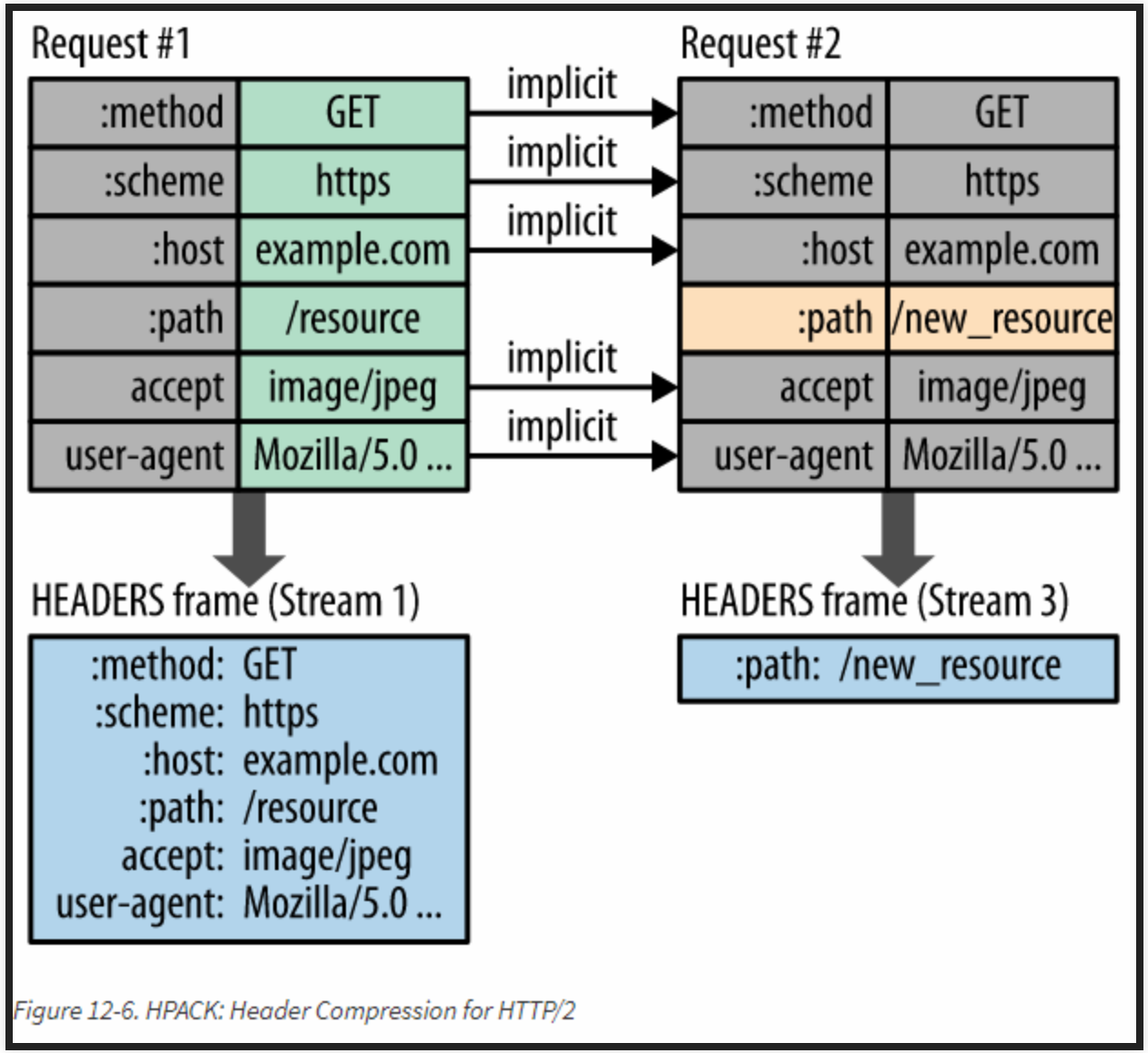


Figure 12-6. HPACK: Header Compression for HTTP/2

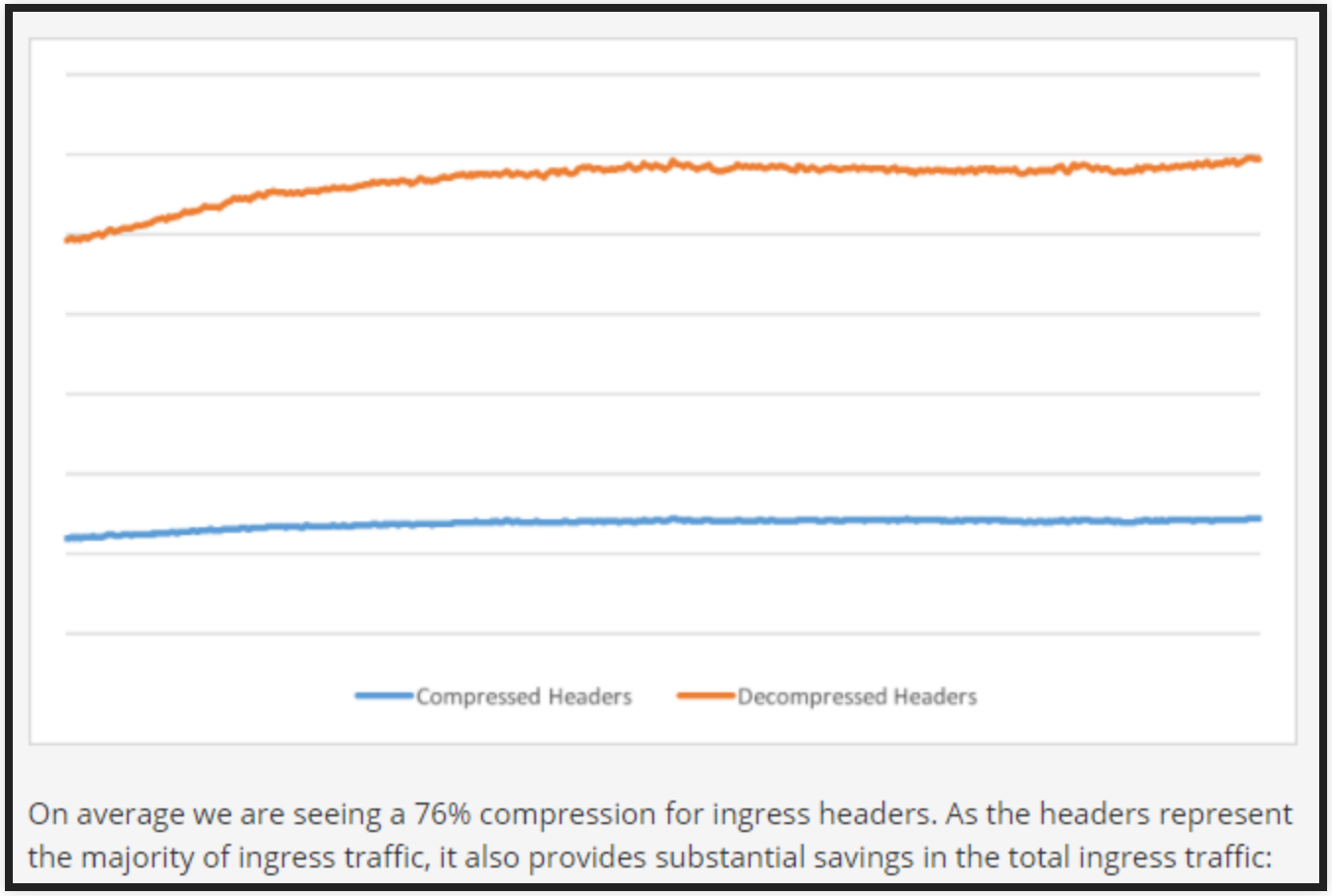
https://hpbn.co/http2/

- Static Dictionary: A predefined dictionary of 61 commonly used header fields, some with predefined values.
- Dynamic Dictionary: A list of actual headers that were encountered during the connection. This dictionary has limited size, and when new entries are added, old entries might be evicted.
- Huffman Encoding: A static Huffman code can be used to encode any string: name or value. This code was computed specifically for HTTP Response/Request headers - ASCII digits and lowercase letters are given shorter encodings. The shortest encoding possible is 5 bits long, therefore the highest compression ratio achievable is 8:5 (or 37.5% smaller).

https://blog.cloudflare.com/hpack-the-silent-killer-feature-of-http-2/

https://blog.cloudflare.com/hpack-the-silent-killer-feature-of-http-2/

https://blog.cloudflare.com/hpack-the-silent-killer-feature-of-http-2/

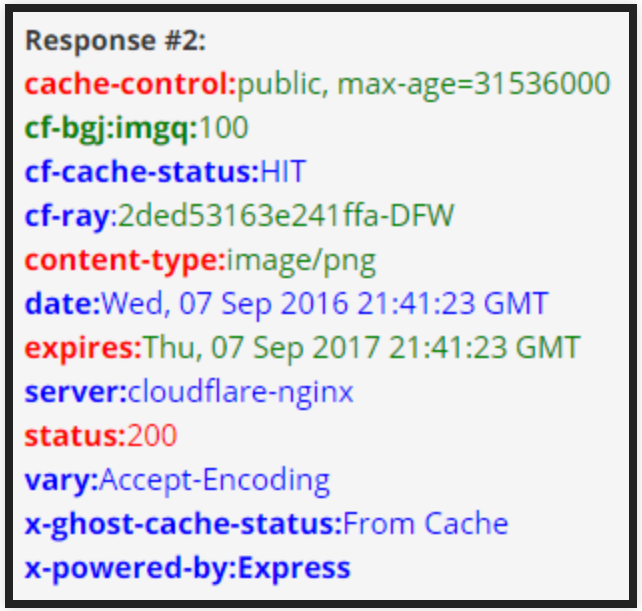


On average we are seeing a 76% compression for ingress headers. As the headers represent the majority of ingress traffic, it also provides substantial savings in the total ingress traffic:

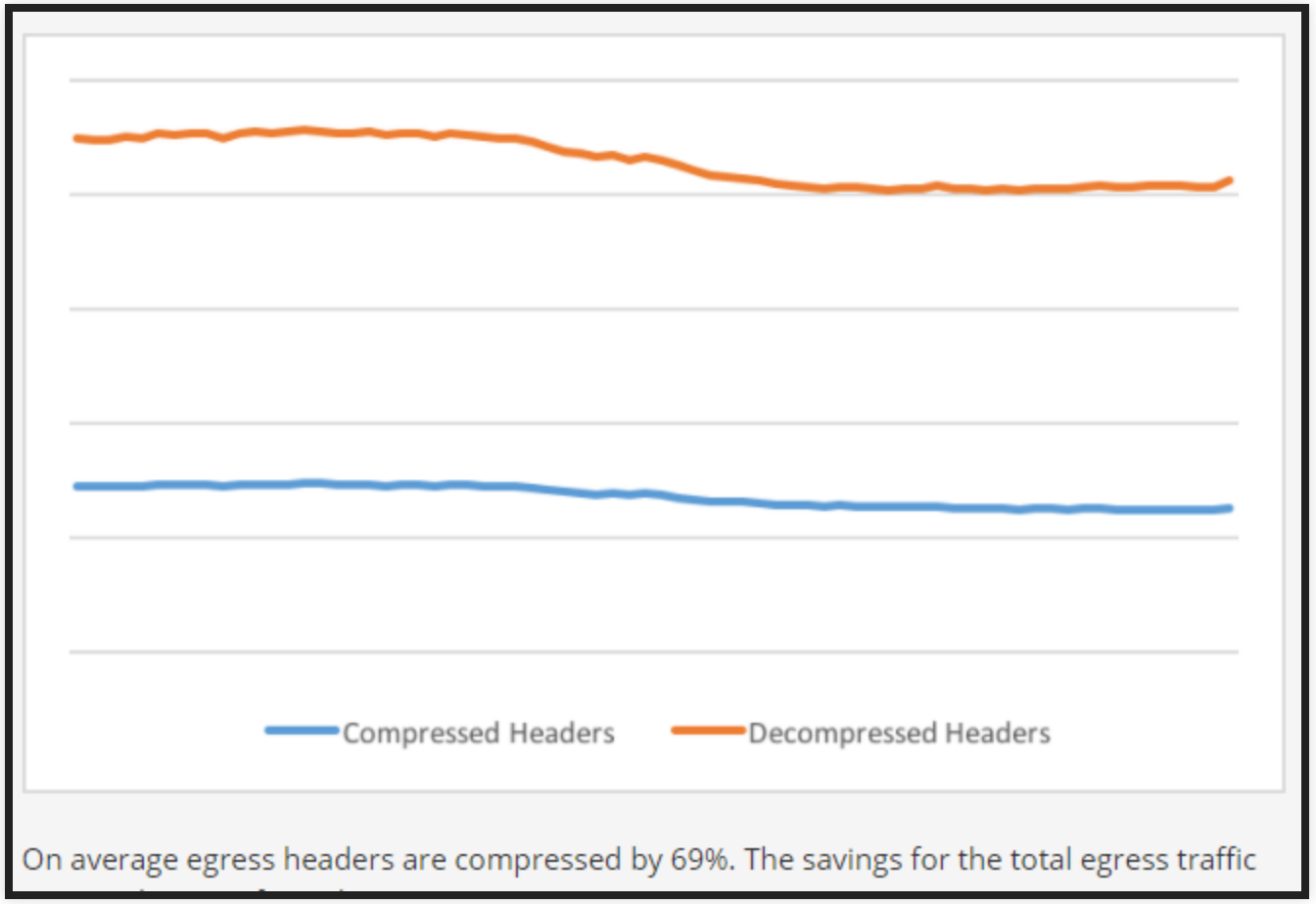
https://blog.cloudflare.com/hpack-the-silent-killer-feature-of-http-2/

Response #1:
cache-control:public, max-age=30
cf-cache-status:HIT
cf-h2-pushed:</assets/css/screen.css?v=2237be22c2>,</assets/js/jquery.fitvids.js?v=2237be22c2>
cf-ray:2ded53145e0c1ffa-DFW
content-encoding:gzip
content-type:text/html; charset=utf-8
date: Wed, 07 Sep 2016 21:41:23 GMT
expires: Wed, 07 Sep 2016 21:41:53 GMT
link: <//cdn.bizible.com/scripts/bizible.js>; rel=preload; as=script, <https://code.jquery.com/jquery-1.11.3.min.js>; rel=preload; as=script
server:cloudflare-nginx
status:200
vary: Accept-Encoding
x-ghost-cache-status: From Cache
x-powered-by: Express

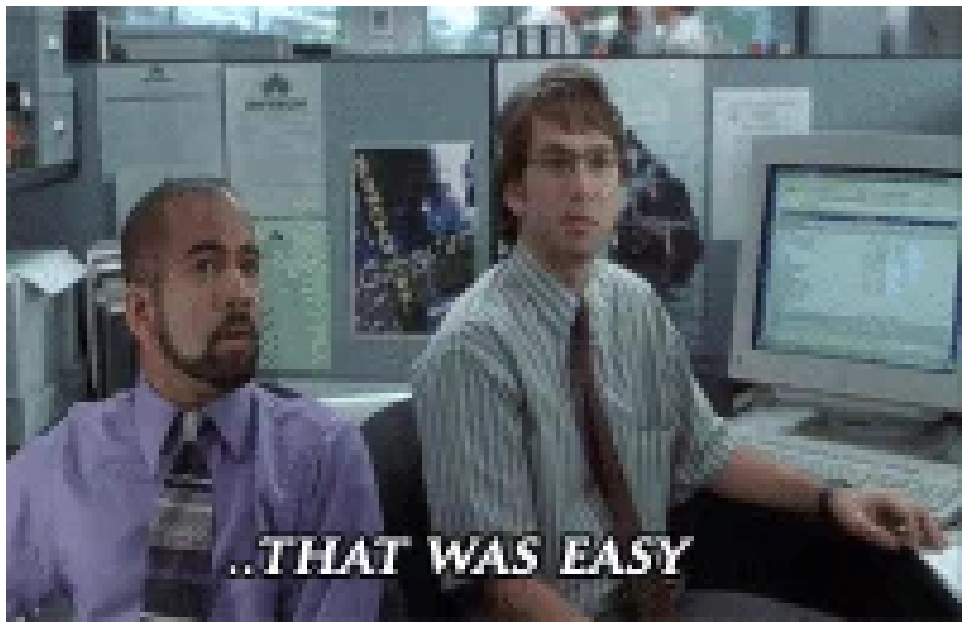
https://blog.cloudflare.com/hpack-the-silent-killer-feature-of-http-2/



https://blog.cloudflare.com/hpack-the-silent-killer-feature-of-http-2/



https://blog.cloudflare.com/hpack-the-silent-killer-feature-of-http-2/


..THAT WAS EASY

https://http2.github.io/http2-spec/compression.html
https://http2.github.io/http2-spec/compression.html#static.table.definition
https://http2.github.io/faq/#why-do-we-need-header-compression

multiplexing

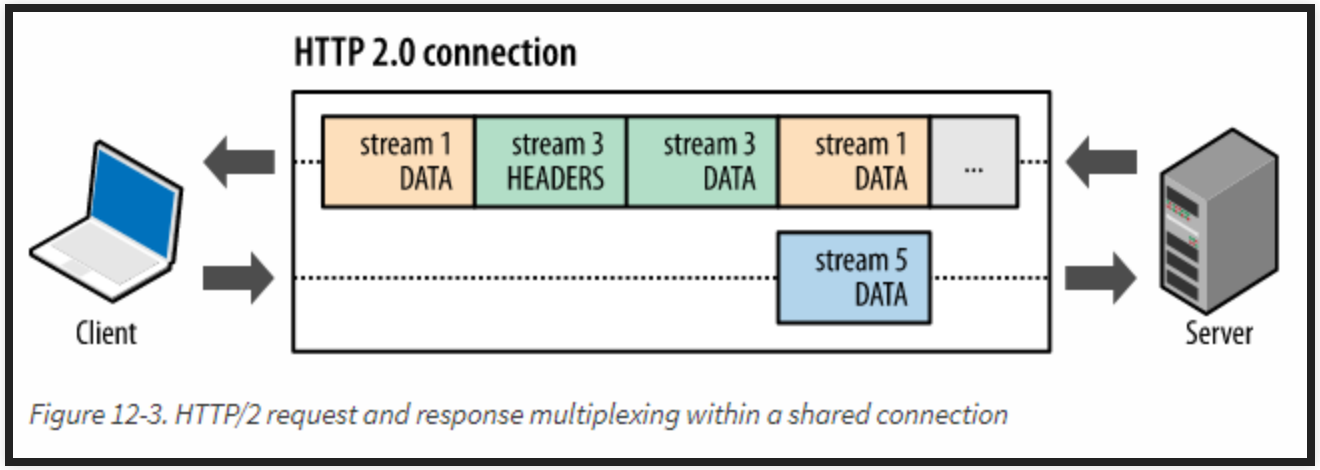
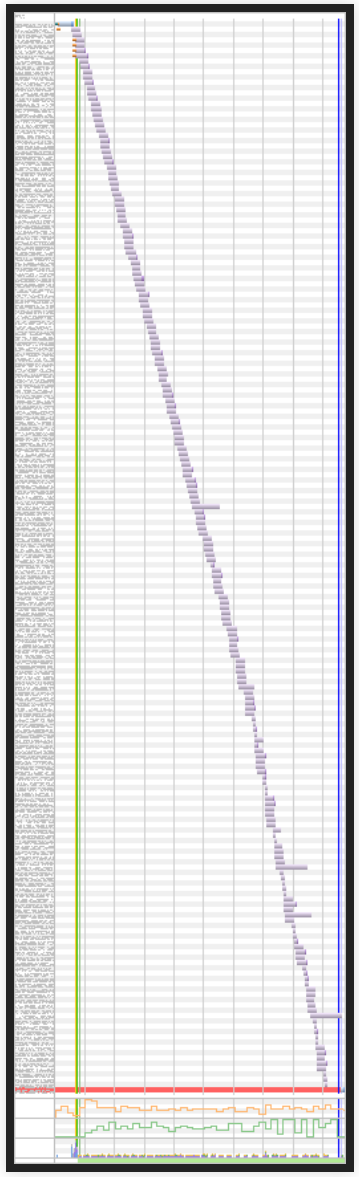
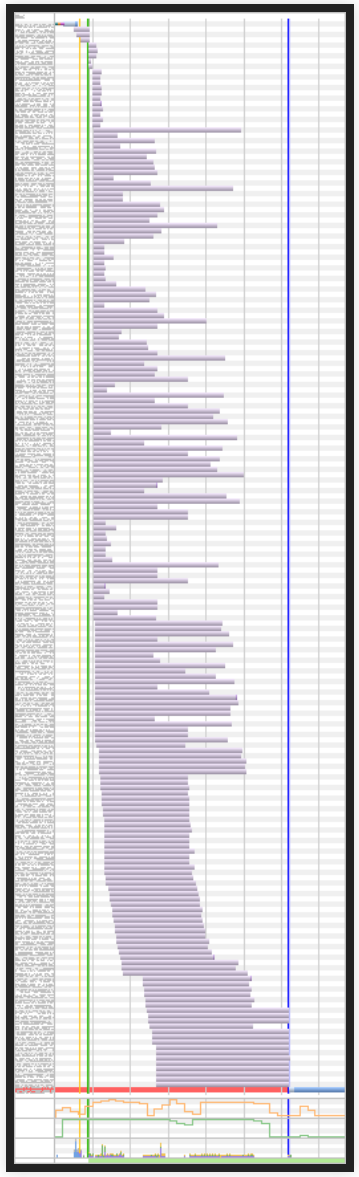


Figure 12-3. HTTP/2 request and response multiplexing within a shared connection

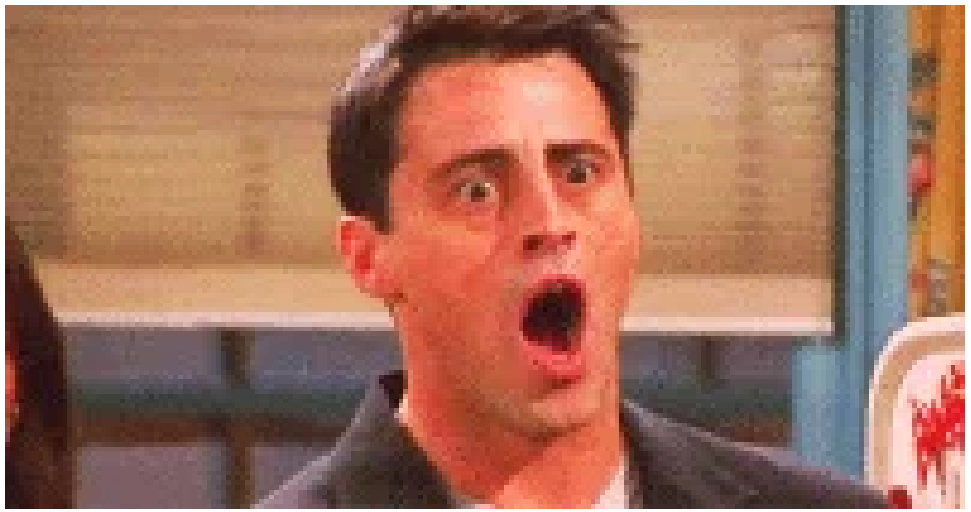
https://hpbn.co/http2/



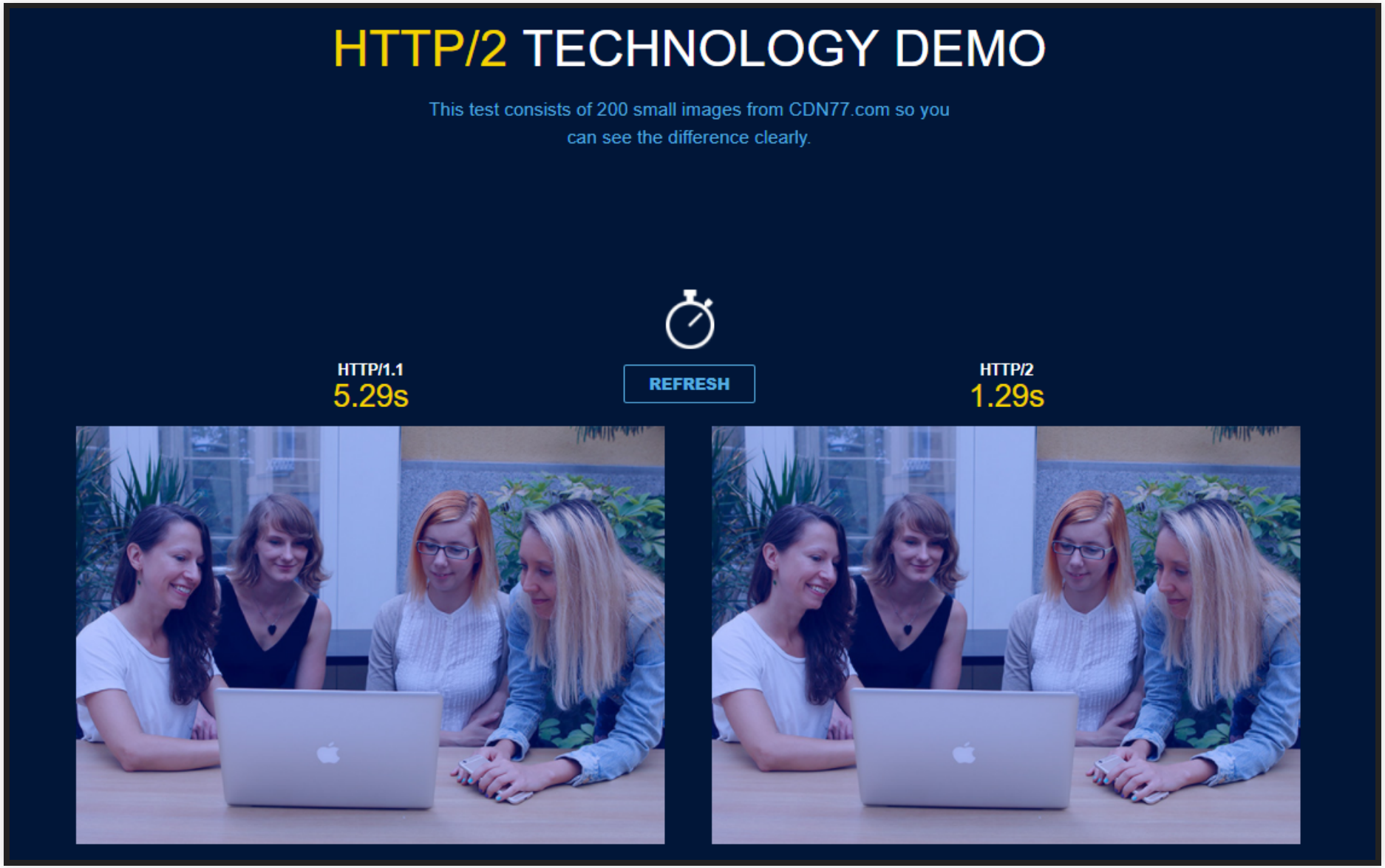
http://www.webpagetest.org/result/171107_JH_cf777098020e891be6f8b6dacce76c81/



http://www.webpagetest.org/result/171107_V7_06052b5e21052c4e209afe926d2bcea0/



https://http2.github.io/faq/#why-is-http2-multiplexed
https://http2.github.io/faq/#why-just-one-tcp-connection



http://www.http2demo.io/

- WiFi / Cable: 4x
- LTE: 6x
- 3G: 15x
- 2G: 2x

https://medium.com/apps-and-networking/http-2-makes-media-loading-3-15-times-faster-on-mobile-a455c3e68135

server push

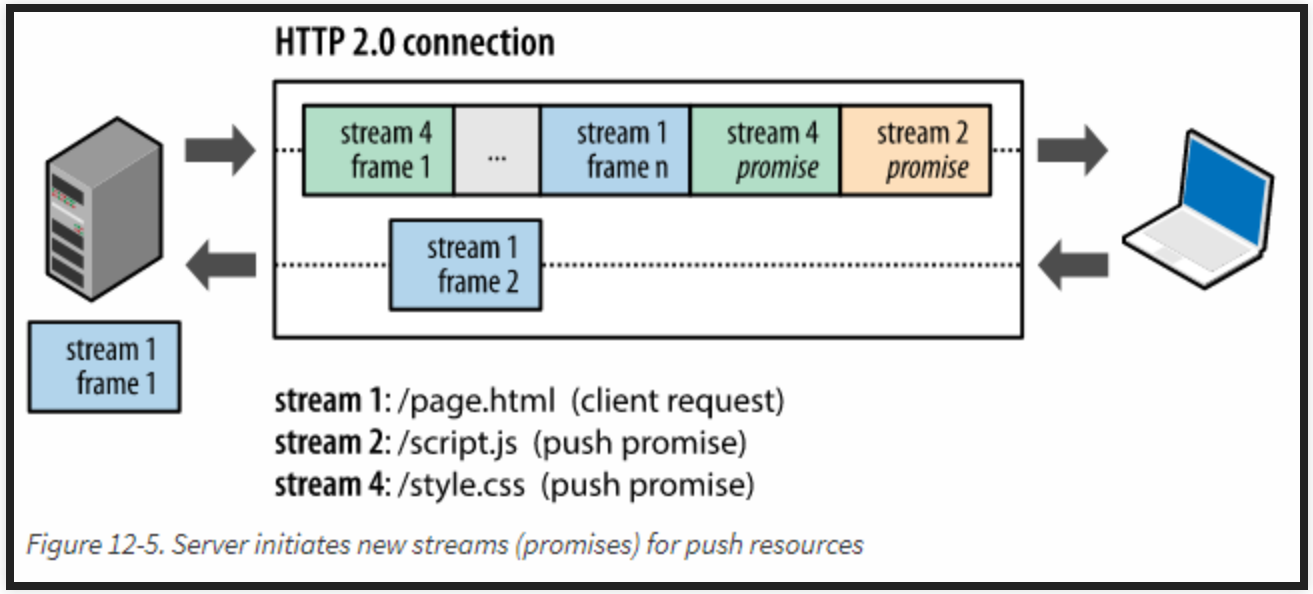


Figure 12-5. Server initiates new streams (promises) for push resources

https://http2.github.io/faq/#whats-the-benefit-of-server-push

don'ts

- ~~concatenation~~
- ~~domain sharding~~
- ~~CDNs~~
- ~~spritesheets~~

https://hpbn.co/http2/
https://http2.github.io