The Priority Header Field

Kazuho Oku
Lucas Pardue

Agenda

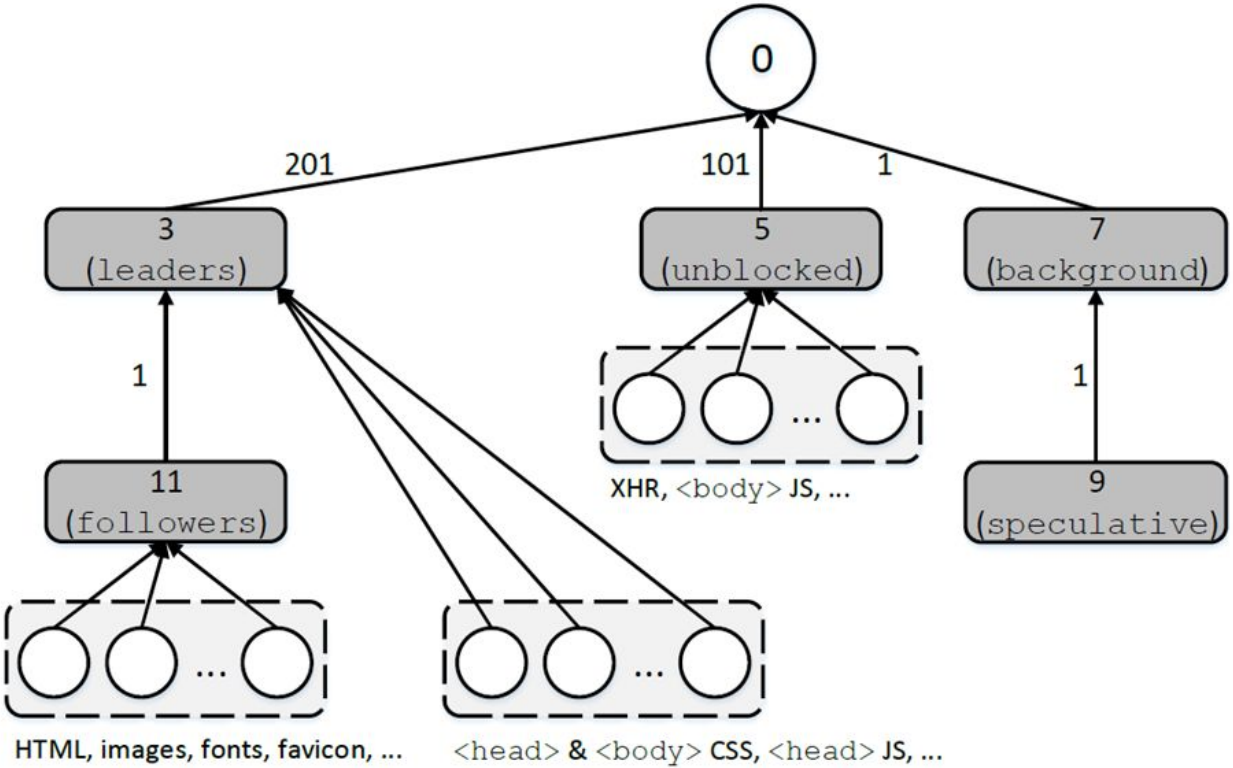
- How priorities are driven today
- Proposal for the Priority header field

https://tools.ietf.org/html/draft-kazuho-httpbis-priority-02

How should we prioritize?

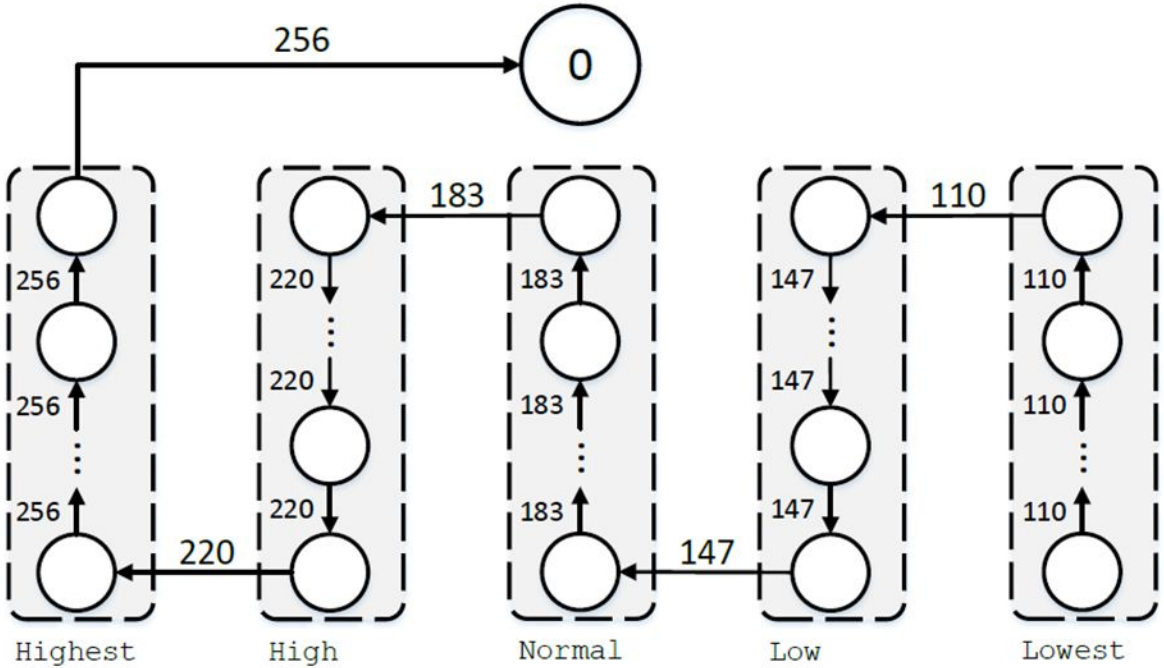
- rough consensus on the best generic ordering
  - serve render-blocking JS, CSS
  - serve HTML
  - serve images
  - serve async scripts
- ideal ordering depends on the website
  - https://lists.w3.org/Archives/Public/ietf-http-wg/2019JulSep/0008.html

Client-driven prioritization - Firefox (H2)



source: https://datatracker.ietf.org/meeting/105/materials/slides-105-httpbis-sessa-http3-priorities-01

Client-driven prioritization - Chrome (H2)



source: https://datatracker.ietf.org/meeting/105/materials/slides-105-httpbis-sessa-http3-priorities-01

Client-driven prioritization - Others (H2)

- suboptimal

Client-driven prioritization - the problem

- H2 scheme works well only when both clients and servers implement it correctly
  - lack of (or disinterest to) support on the server-side
  - some clients do not implement it in an optimal way

Server-driven prioritization - Fastly

- content-type-based prioritization as a backup
  - for browsers that do not provide good signal
- when the client does not use a placeholder
  - serve CSS, JS before other content-types
- the issues:
  - responses for <script async> provided too early
    - need more hints than just the content-type
  - the detection rule is fragile~~broken~~

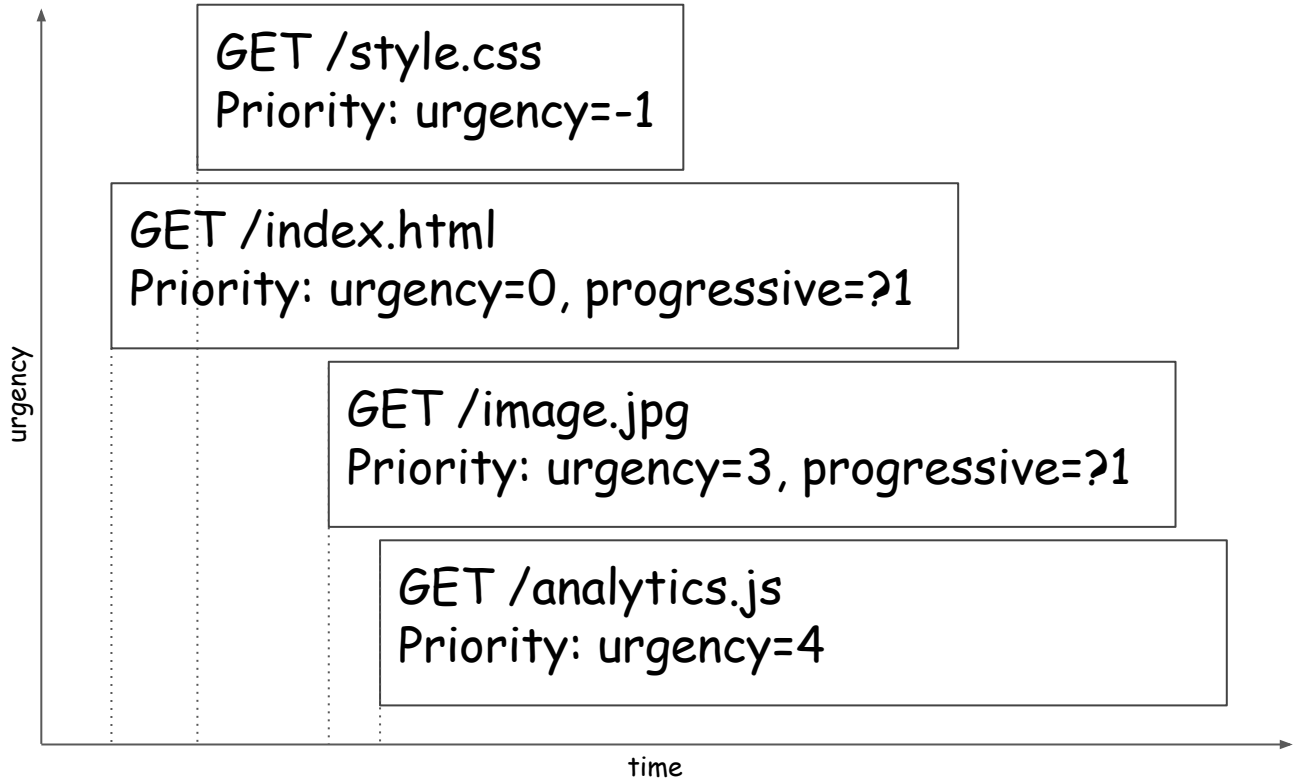
Server-driven prioritization - Cloudflare

- by default, similar approach and issues as Fastly
  - build internal prioritization model from client hints
- give chance to improve performance via tweaks
  - "cf-priority: 30/0"
  - opportunity to extend tweaking capability
- but clients use different weights (and dependencies)
  - difficult to tweak things in a way that provides consistency without encountering complexity

Server-driven prioritization

- the desire to standardize a response header, that
  - helps us tweak client-provided priorities
  - or works as a backup against a client not providing correct signal

Our proposal: Priority header field

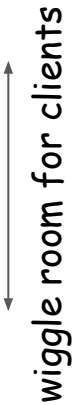


Design principles

- create a minimal spec based on how we prioritize now
  - helps us to agree on something early
  - minimizes the risk of performance becoming inferior to H2
  - server-sent signal to improve (or cover the lack of) the hints from client
- provide extensibility for the future

8 semantic urgency levels

Name	Urgency	Examples
prerequisite	-1	CSS, JS in <HEAD>
default	0	HTML, fonts
supplementary	1	(server-only)
	2	hero images
	3	images
	4	async JS
	5	(server-only)
background	6	prefetch, file download

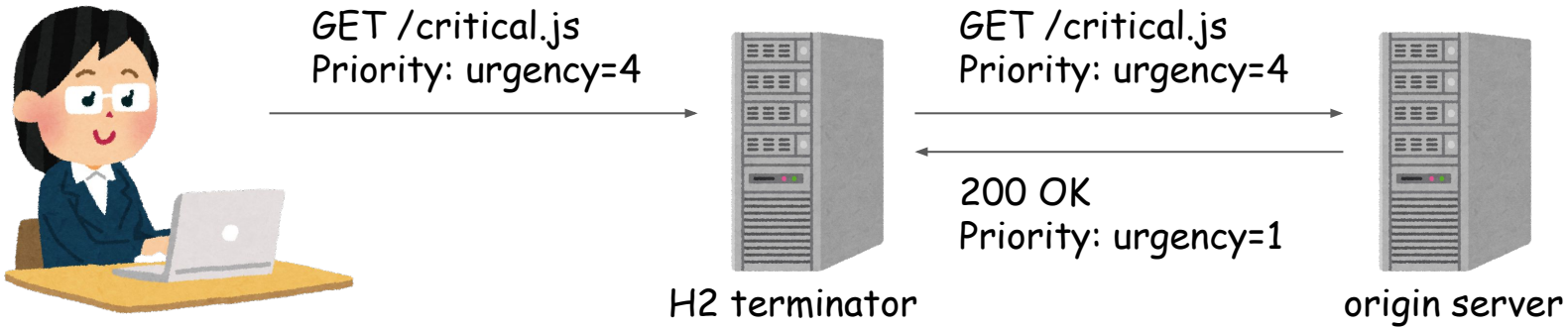


"progressive" flag

- a boolean indicating if the client anticipates it can provide some meaningful output as the chunks of the response arrives
  - images => true, CSS, JS => false

Client-server collaboration

- so that server can "fine-tune" the priorities
- server-provided parameters override those provided by the client
- example: serving async JS before images



Why semantic urgency levels?

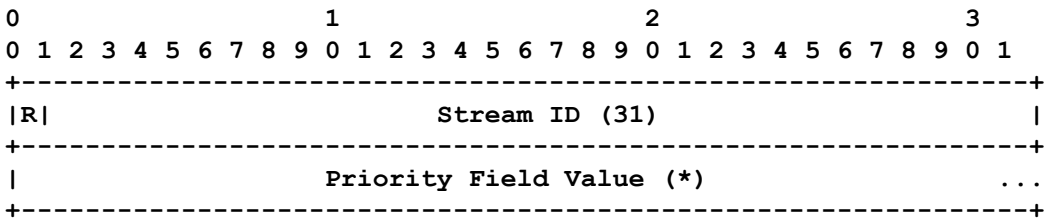
- without meanings attached to each urgency level, servers cannot tell the urgency level to which the response should be promoted / demoted

Why use a header field, instead of a frame?

- the signal has to be carried by a header between the H2 terminator and the origin
  - because some hops between the two might be H1
  - server-provided signal should be cacheable
- means that the terminator has to have the code that handles the priority "response" header
- then, why not <u>always</u> use a header?
  - bonus: can be set by Service Worker, etc.

Reprioritization

- needs to be done in a HTTP-version-specific manner
- we could transmit a frame that contains the value of the priority header field



Acknowledgements

- header-based prioritization predates to http://tools.ietf.org/agenda/83/slides/slides-83-httpbis-5.pdf
- sending the tuple of urgency and concurrency was first proposed in https://github.com/pmeenan/http3-prioritization-proposal

Proposal: summary

- 8 semantic urgency levels
- "progressive" parameter
- client-server collaboration
- room for future extension
- use header fields throughout