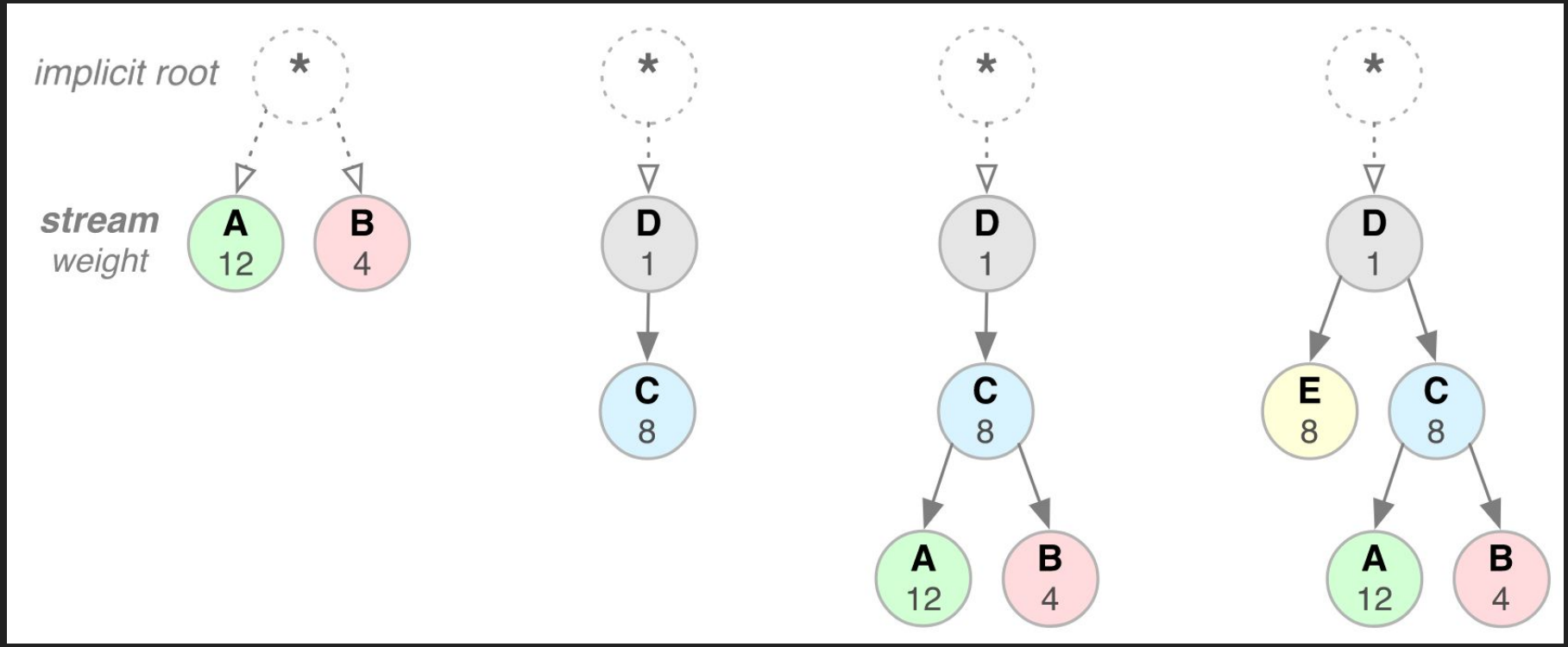
H3 Priorities

IETF 105 July 2019

It all began with a

And the H2 priority tree was born...



See Section 5.3 of RFC7540 for a full description

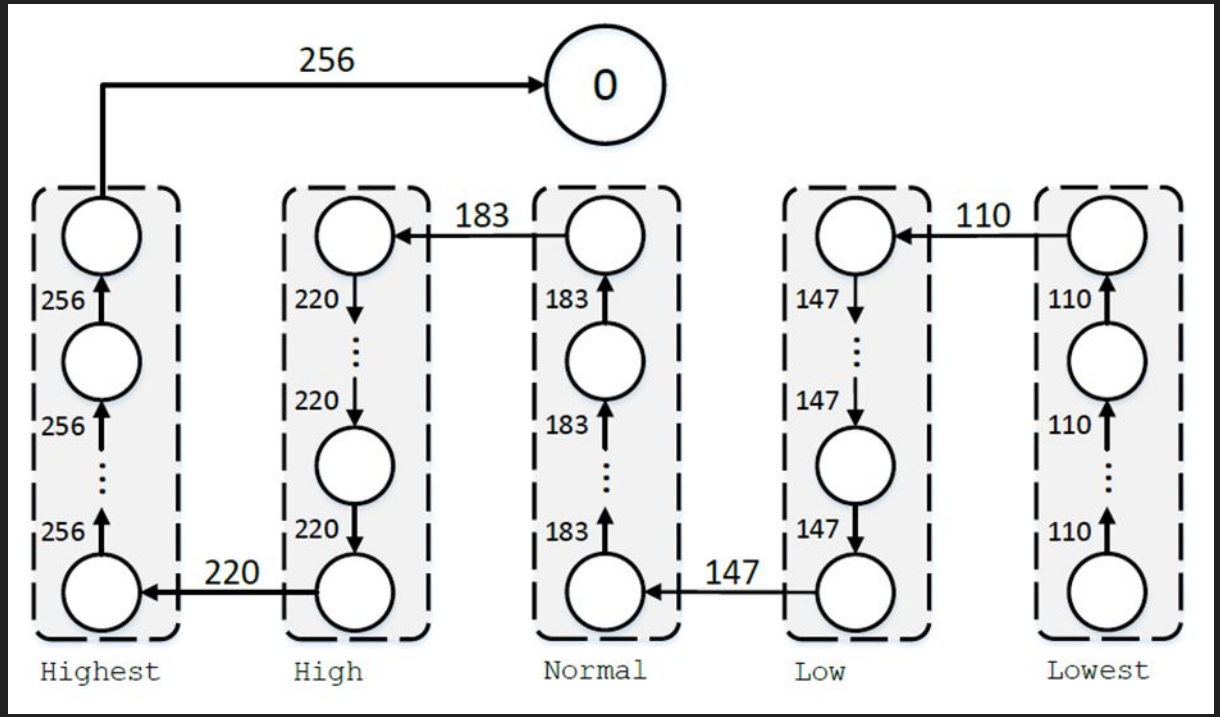
What are H2 priorities really?

Strict prioritization is implicit in the tree structure

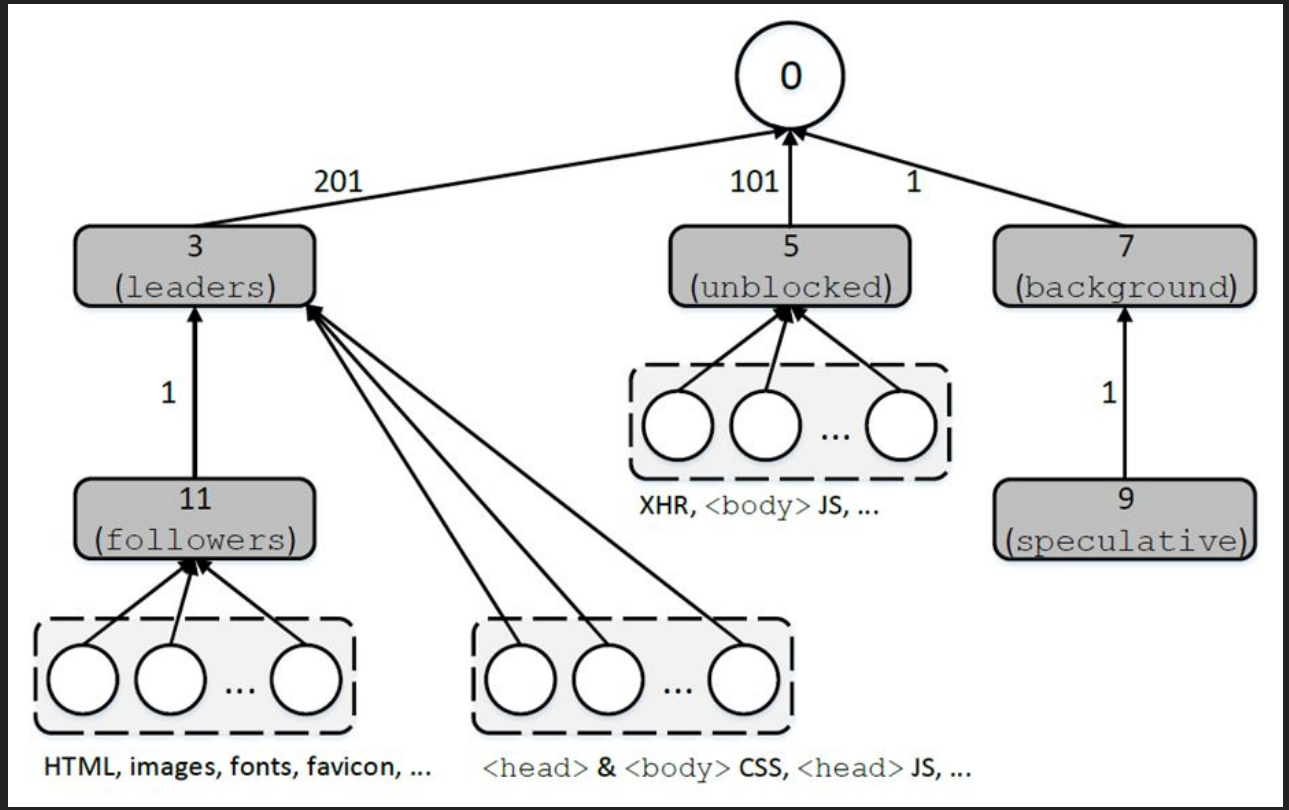
Weights to share bandwidth between nodes

How do browsers use H2 priority today?

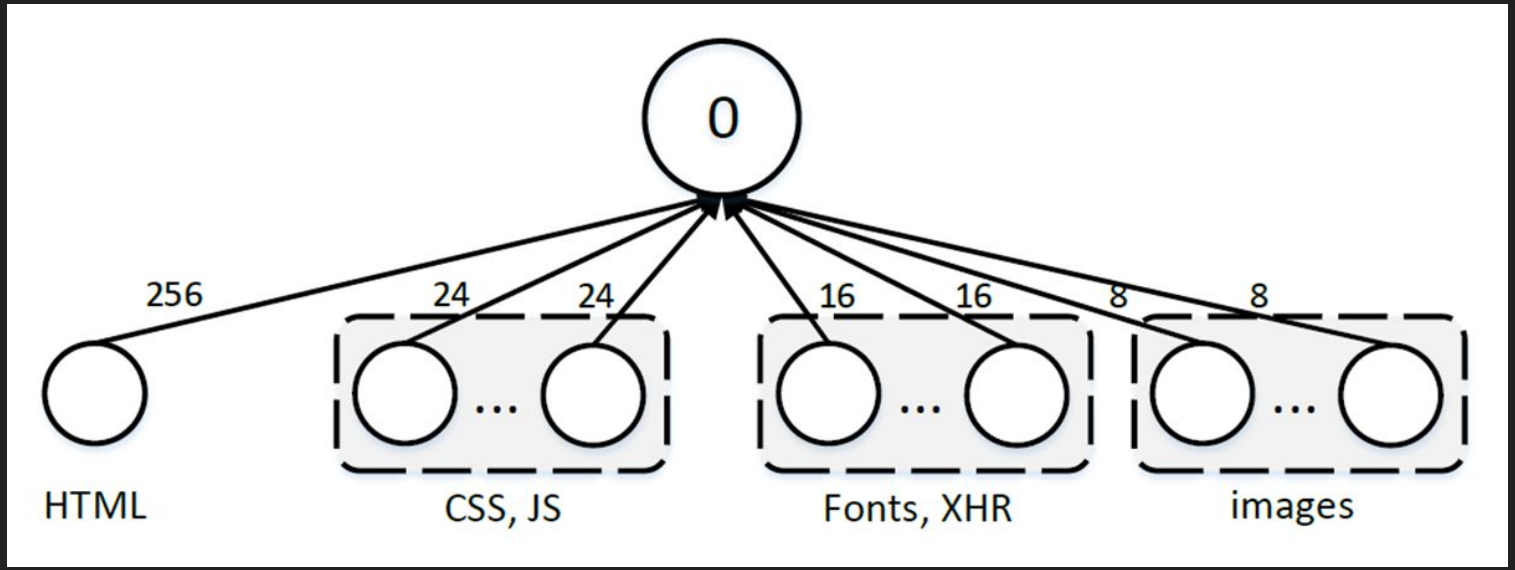
H2 and H3(Chrome) : First-Come-First-Served



HTTP/2 original (Firefox) : “placeholders” avant-la-lettre



HTTP/2 (Safari) : Weighted Round-Robin



Overview of H2, H3, pmeenan*

*Patrick Meenan as well as other WG members

What are H2 priorities really?

Strict prioritization is implicit in the tree structure

Weights to share bandwidth between nodes

What are H2 priorities really?

Strict prioritization is implicit in the tree structure
Weights to share bandwidth between nodes

Conceptually, this is a very clean model, but maybe the toolbox is a bit too large?

H3 Priorities: basically keep/expand the toolbox

Add explicit placeholders

- Though potentially allowing 0 (or “too few”) is an issue (1,2)

Send all PRIORITY frames on the control stream

- Ensure consistency* between client and server trees

‘Orphan placeholder’ to default FIFO

- If a stream or placeholder arrives before anything
- Functionally similar to a ‘0-weight’ proposal (3)

1 https://github.com/quicwg/base-drafts/issues/2734
2 https://github.com/quicwg/base-drafts/issues/2753
3 https://github.com/quicwg/base-drafts/pull/2723

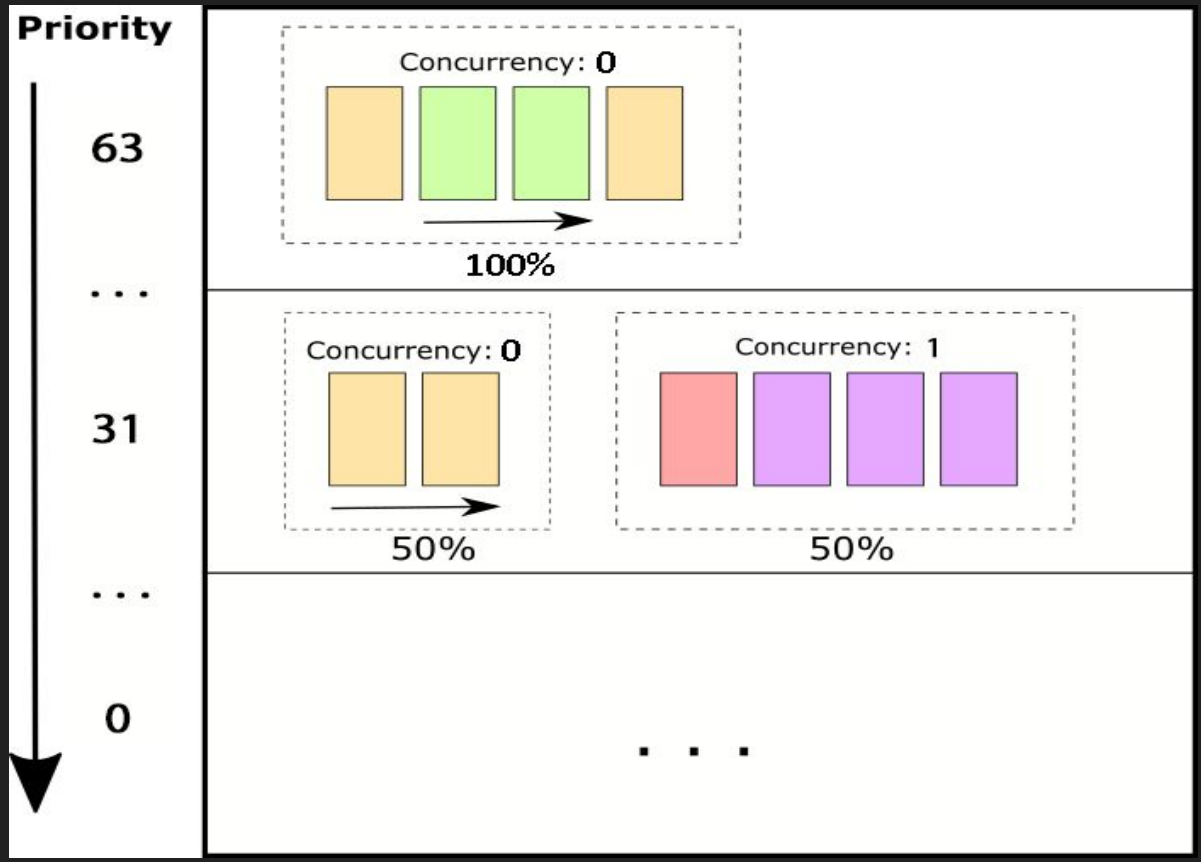
Patrick Meenan's Proposal

SPDY-style numerical priorities with an additional bit for concurrency

0: Sequential

1: Concurrent

(aka Round Robin)



Proposal: http3-prioritization-proposal

Blog post: better-http-2-prioritization-for-a-faster-web

What do we actually need?

Separate research efforts concluded ~same thing

“IMHO, an optimal ordering looks something like this:

Serialize the CSS and blocking JS in the head
Load the visible images in parallel (simple round robin)
Serialize async (and late-body) JS and load them in parallel with non-visible images. The non-visible images might be better to serialize in document order for long-scroll pages instead of loading ALL of them in parallel“

- Patrick Meenan from #2502

Separate research efforts concluded ~same thing

“I think there is a rough consensus that a ideal tree looks like:

Root

  +- JS1 (blocking)

      +- CSS1 (blocking)

          +- CSS2 (blocking)

              +- image1 (weight=X)

              +- image2 (weight=X)

              +- image3 (weight=X)

HTML could be either in between the root and JS1, or between CSS2 and image1.“

- Kazuho from #2753

Separate research efforts concluded ~same thing

“... more sequential schemes generally outperform more Round-Robin-alike schemes.“

“ it is perfectly possible to switch to a simplified prioritization framework while still fully supporting the web browsing use case and without losing performance.”

From Robin Marx Paper at https://h3.edm.uhasselt.be/

What do we really need from the toolbox (browser use case)?

- Most things need to be FIFO link
  - Scripts, CSS, fonts, … need to be fully downloaded to use
  - Yet, H2 makes FIFO difficult (inter-stream dependencies)

- Only some things should be Round-Robin link
  - Progressive images, video, HTML (+-)
  - Yet this is the default in H2

- So: H3 made FIFO easier

H2 priorities can achieve optimal loading, so why change?

Achieving wide adoption

Full compliance to H2 priorities is ~25% of major CDNs link

- Everyone agrees this is a problem, no sign of it changing
- Many implementations think it's too complex(even if they support it)
- Some found it doesn't outperform FIFO

FYI: H2 priorities are off by default at Google

Issue #2739

Allow Server input

Sometimes the server knows more than the browser

- The priority tree is a distributed synchronization problem, similar to QPACK. The server can't change it.
- There are no JS APIs for priorities*, so apps are stuck

Note: Server push REALLY benefits from an initial priority

Issue #2740

* https://www.chromestatus.com/feature/5273474901737472

What if you don’t have enough/any placeholders?

This breaks Firefox…

However, Robin’s paper concluded “it is perfectly possible to switch to a simplified prioritization framework while still fully supporting the web browsing use case and without losing performance. Schemes such as bucket HTML and zeroweight are easy to implement performantly, do not require placeholders”

Issue #2753, #2734

A starting point for other applications

How could Datagrams/WebTransport fit into H2 priorities?

WebRTC wants prioritization, but definitely not a tree

Potential Concerns

What if it doesn’t work?

Research shows most browsers would be better with FIFO

Chrome’s been running a linked list for years

gQUIC has always used SPDY priorities

Cloudflare has demonstrated performance improvements

H2 will be ‘left behind’

We can backport any scheme to H2

- Depends upon TTPHTTP WG interest, but it’s not that much work

Most servers have poor support for H2 priorities today

Most browsers would likely use H3 priorities better than H2’s

Next Steps

A Proposal on how to move forward

Wide interest in moving from tree to N buckets + concurrency

- Can we agree on that now for H3?
- If there are objections, let’s discuss and see what to do

A side meeting is scheduled Wed @ 8:30am in Van Horne

- Discuss framing, what N should be, binary and/or header encodings

Come back Thursday with more complete design(s)

Thanks!

To Robin Marx, for his excellent research and great illustrations!

To Patrick Meenan, for sending out an alternate proposal earlier this year

To Kazuho Oku and Lucas Purdue for their ideas, feedback, and the header-based proposal

To all the other WG members who gave me feedback on ideas, slides, etc