Building Faster Drupal Sites

Adventures in optimizing the critical path and intelligent asset management

Who am I?

- Front-end Drupal Developer at Chapter Three
- Life-long HTML addict
- @bollskis

Measure everything, for science

Measure early and often

Web page test is the best tool to measure your performance. Page speed insights will help you fine-tune your site.

- www.webpagetest.org
- https://developers.google.com/speed/pagespeed/insights/

Optimizing the critical path & Drupal

What is the critical path?

tl:dr; The steps from download to render.

The browser turns our HTML, CSS, and JavaScript into a usable page. The critical path is the steps the browser takes to render the code we give it.

The key to optimizing for the critical path? **Stop shoving it full of so much shit.**

Critical Path Blockers

These items will require the browser to download them before rendering.

- Any CSS file
- Any non-async JS file
- External fonts
- External JS libraries
- External Media assets

ASYNC LOAD ALL THE THINGS!

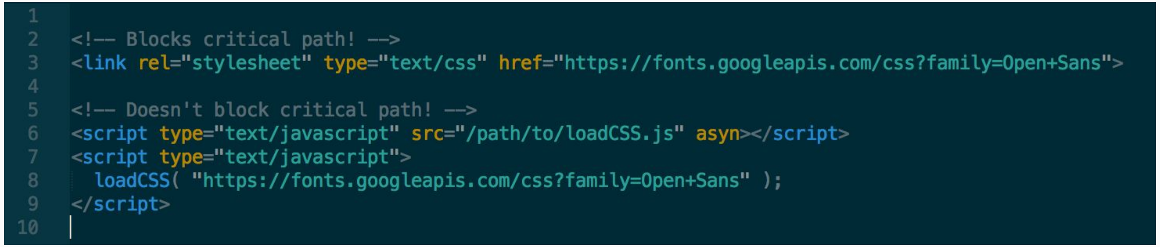
But I *need* [insert external piece of shit]. How do I load it so it doesn’t block the critical path?

Like all problems in our current web environment, let’s solve it with JavaScript!

LoadCSS, loadJS, and the Filament Group to the rescue.

LoadCSS … loads… CSS

But it loads it asynchronously, so the file doesn’t block the critical path.



```

<!-- Blocks critical path! -->
<link rel="stylesheet" type="text/css" href="https://fonts.googleapis.com/css?family=Open+Sans">

<!-- Doesn't block critical path! -->
<script type="text/javascript" src="/path/to/loadCSS.js" asyn></script>
<script type="text/javascript">
  loadCSS( "https://fonts.googleapis.com/css?family=Open+Sans" );
</script>

```

Further reading

- https://developers.google.com/web/fundamentals/performance/critical-rendering-path
- Google “Delivering the goods Paul Irish”
- https://www.filamentgroup.com/lab/performance-rwd.html

Responsive images in core!

Responsive images are here and they’re badass

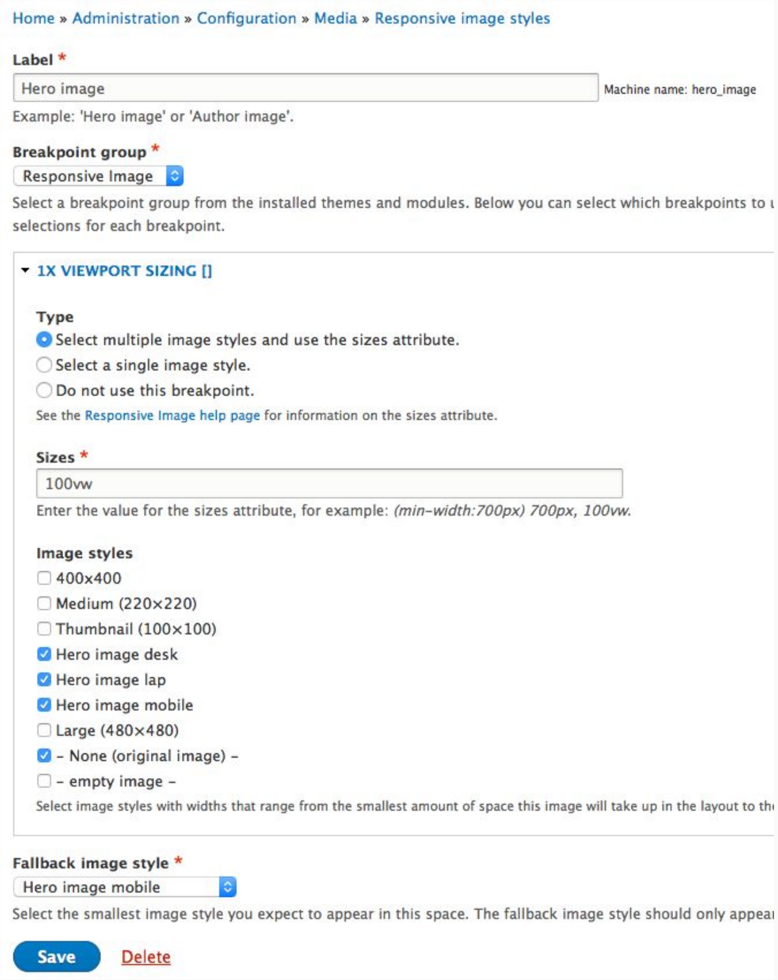
<picture> is supported in most major browsers, and Drupal 8 has responsive images in core.

Utilizing the <picture> element is **key** to a well-built responsive site. And it’s really not that difficult to use.

Hero Banners

A hero banner is usually a large image that resides at the top of the page.

If you’re not changing resolutions, responsive hero banners could not be simpler to set up.



Home » Administration » Configuration » Media » Responsive image styles

Label *

Hero image   Machine name: hero_image

Example: 'Hero image' or 'Author image'.

Breakpoint group *

Responsive Image

Select a breakpoint group from the installed themes and modules. Below you can select which breakpoints to u
selections for each breakpoint.

▾ 1X VIEWPORT SIZING []

Type

● Select multiple image styles and use the sizes attribute.
○ Select a single image style.
○ Do not use this breakpoint.

See the Responsive Image help page for information on the sizes attribute.

Sizes *

100vw

Enter the value for the sizes attribute, for example: *(min-width:700px) 700px, 100vw.*

Image styles

☐ 400x400
☐ Medium (220×220)
☐ Thumbnail (100×100)
☑ Hero image desk
☑ Hero image lap
☑ Hero image mobile
☐ Large (480×480)
☑ – None (original image) –
☐ – empty image –

Select image styles with widths that range from the smallest amount of space this image will take up in the layout to th

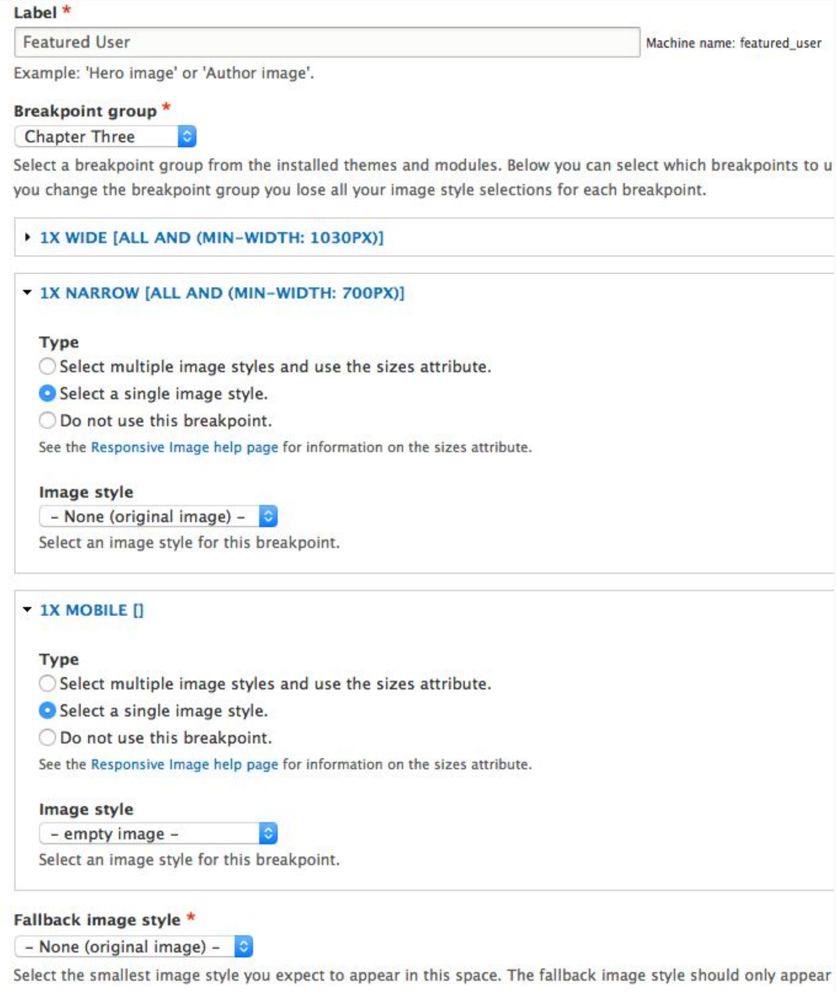
Fallback image style *

Hero image mobile

Select the smallest image style you expect to appear in this space. The fallback image style should only appear

Empty mobile images

Sometimes we want an image for larger display, and not for mobile. We can do this without a nasty display: none;!



Label *

Featured User

Machine name: featured_user

Example: 'Hero image' or 'Author image'.

Breakpoint group *

Chapter Three

Select a breakpoint group from the installed themes and modules. Below you can select which breakpoints to u
you change the breakpoint group you lose all your image style selections for each breakpoint.

▸ 1X WIDE [ALL AND (MIN-WIDTH: 1030PX)]

▾ 1X NARROW [ALL AND (MIN-WIDTH: 700PX)]

Type

○ Select multiple image styles and use the sizes attribute.
● Select a single image style.
○ Do not use this breakpoint.

See the Responsive Image help page for information on the sizes attribute.

Image style

- None (original image) -

Select an image style for this breakpoint.

▾ 1X MOBILE []

Type

○ Select multiple image styles and use the sizes attribute.
● Select a single image style.
○ Do not use this breakpoint.

See the Responsive Image help page for information on the sizes attribute.

Image style

- empty image -

Select an image style for this breakpoint.

Fallback image style *

- None (original image) -

Select the smallest image style you expect to appear in this space. The fallback image style should only appear

SVG icons are better than font icons.

Because, reasons:

- Faster
- Better for screen readers
- Better for i18
- Semantically sound
- Did I mention faster?

More reading: https://css-tricks.com/icon-fonts-vs-svg/

SVGs are just as easy to use (in theory)

Instead of

```
<i class=”fa fa-bars”></i>
```

We use

```
<svg class=”icon icon-bars”>
  <use xlink:href=”#icon-bars”></use>
</svg>
```

Similar syntax, and not an icon system snuck into a font.

Gulp can watch a folder of icons and compile it into a single SVG sprite.

Inline the SVG sprite into html.html.twig for best x-browser support.

SVG Resources

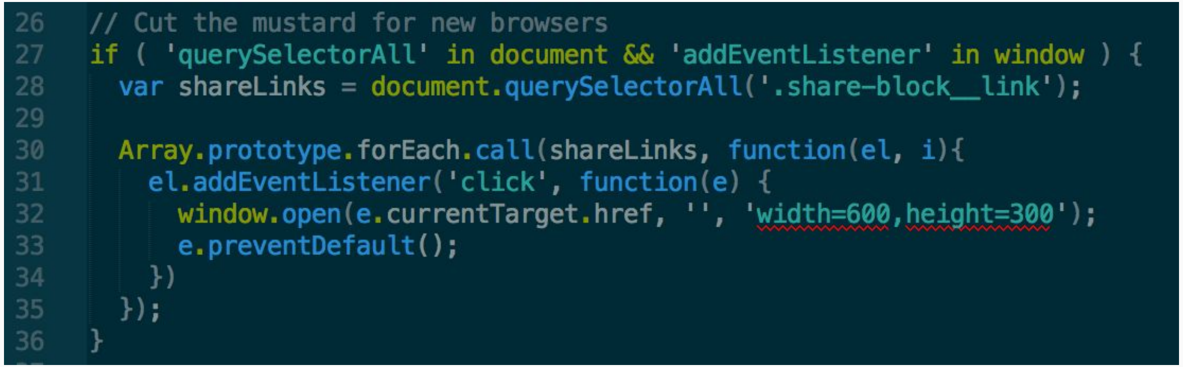
Just a few good resources to learn more about SVG icon systems

- https://css-tricks.com/icon-fonts-vs-svg/
- https://css-tricks.com/svg-sprites-use-better-icon-fonts/
- https://github.com/encharm/Font-Awesome-SVG-PNG/
- https://github.com/google/material-design-icons

Cutting the mustard

Serving content to users that can’t handle it is plain rude.

Cutting the mustard means testing the user's environment before serving up enhanced content.

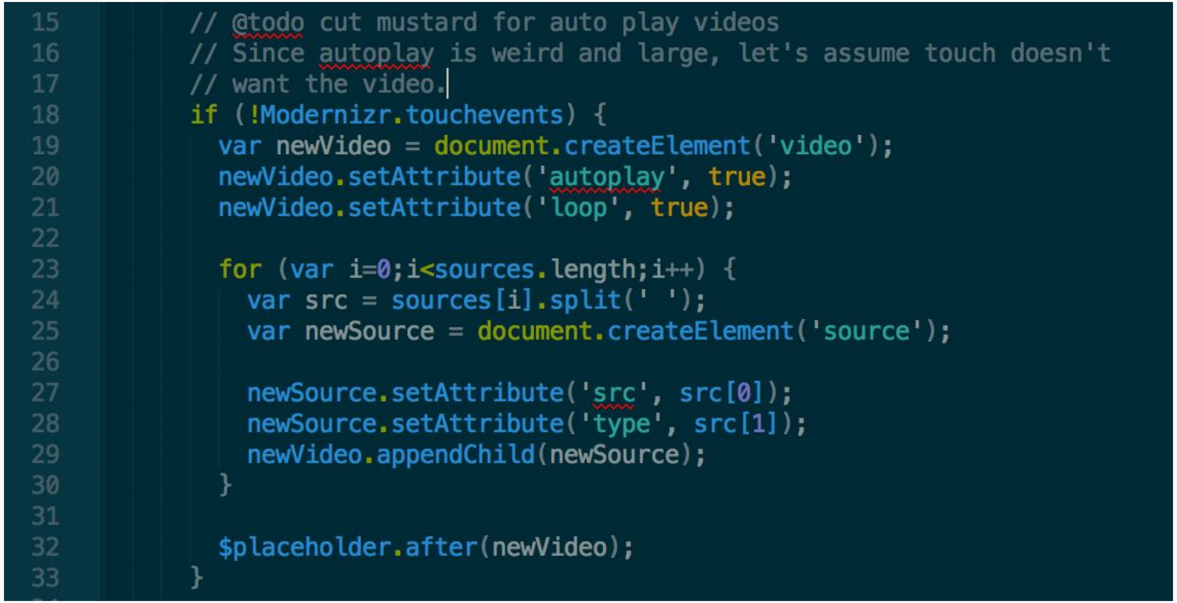


```
// Cut the mustard for new browsers
if ( 'querySelectorAll' in document && 'addEventListener' in window ) {
  var shareLinks = document.querySelectorAll('.share-block__link');

  Array.prototype.forEach.call(shareLinks, function(el, i){
    el.addEventListener('click', function(e) {
      window.open(e.currentTarget.href, '', 'width=600,height=300');
      e.preventDefault();
    })
  });
}
```

Serve the smallest amount of data first

Only enhance your page with certain features once you know they’re supported.

A great example is the chapterthree.com homepage. We autoplay a video, but we **don’t even want a video if you can’t autoplay it.**

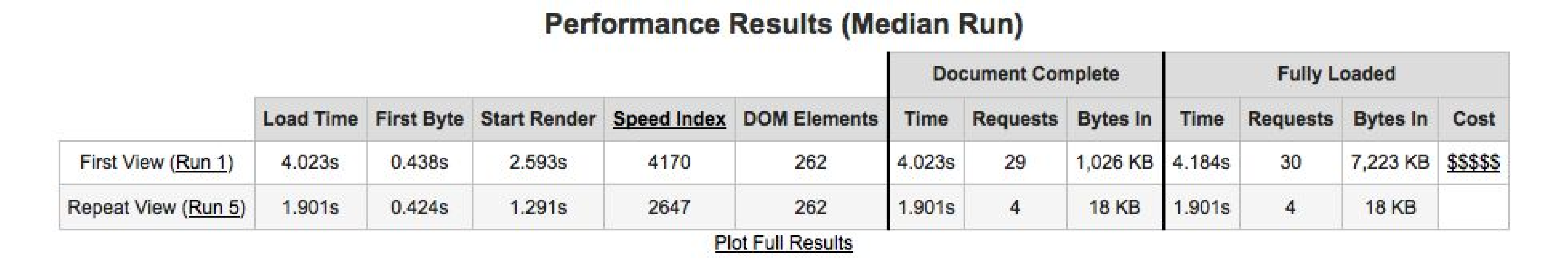


```
// @todo cut mustard for auto play videos
// Since autoplay is weird and large, let's assume touch doesn't
// want the video.
if (!Modernizr.touchevents) {
  var newVideo = document.createElement('video');
  newVideo.setAttribute('autoplay', true);
  newVideo.setAttribute('loop', true);

  for (var i=0;i<sources.length;i++) {
    var src = sources[i].split(' ');
    var newSource = document.createElement('source');

    newSource.setAttribute('src', src[0]);
    newSource.setAttribute('type', src[1]);
    newVideo.appendChild(newSource);
  }

  $placeholder.after(newVideo);
}
```

Put up or shut up

Old vs New Chapter Three homepage

Old

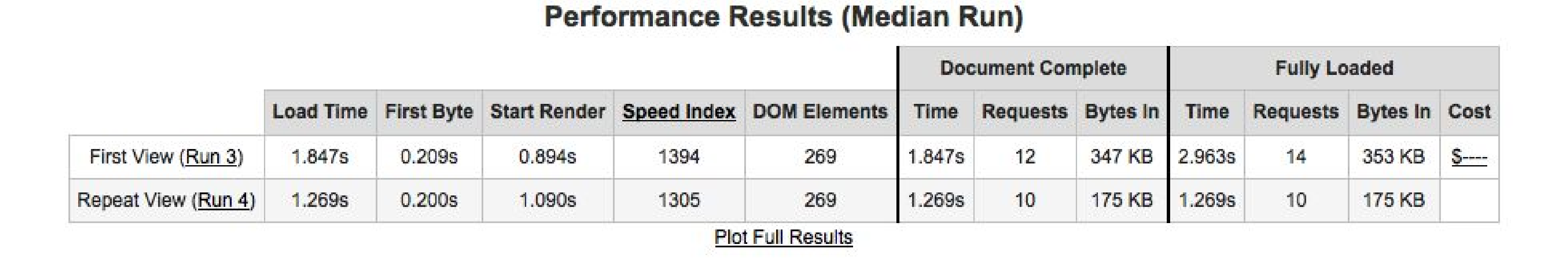


Performance Results (Median Run)

						Document Complete			Fully Loaded			
	Load Time	First Byte	Start Render	Speed Index	DOM Elements	Time	Requests	Bytes In	Time	Requests	Bytes In	Cost
First View (Run 1)	4.023s	0.438s	2.593s	4170	262	4.023s	29	1,026 KB	4.184s	30	7,223 KB	$$$$$
Repeat View (Run 5)	1.901s	0.424s	1.291s	2647	262	1.901s	4	18 KB	1.901s	4	18 KB	

Plot Full Results

New



Performance Results (Median Run)

						Document Complete			Fully Loaded			
	Load Time	First Byte	Start Render	Speed Index	DOM Elements	Time	Requests	Bytes In	Time	Requests	Bytes In	Cost
First View (Run 3)	1.847s	0.209s	0.894s	1394	269	1.847s	12	347 KB	2.963s	14	353 KB	$----
Repeat View (Run 4)	1.269s	0.200s	1.090s	1305	269	1.269s	10	175 KB	1.269s	10	175 KB	

Plot Full Results

Now go and make shit fast :)