My name is Justin do the RWD Weekly newsletter, Podcast and build the knowledge hub RWD.is site.

Today I want to talk to you about what you need to be thinking about and practicing when approaching your responsive design projects now and in 2018.

The web moves at such a fast pace, it can be difficult to keep up with the constantly growing number of devices, screens, inputs, and Internet users around the globe.

It also means that the things I used to outline my talk in April aren’t necessarily the things that I’m interested in telling you in October…. so I’m going to do both.

- **280 weeks ago**
- A reason to **read lots**, distill the content, put a spin on it and reshape it.
- Highly encourage you to **start something similar,** even if it's just for work because it's a great way to keep everyone informed.
- **How I approached creating the talk**
    - **panic for filling time**
    - **do loads, create a tonne of code examples**
    - **wind up with too much content**
    - **try and compress it all into the time.**
- **There will be code, but I'll step you through it so it's understandable**
- **Take notes, but I'll email everyone with a link this week with full slides and notes with links and live examples.**

1. Use CSS Grid to build in browser wireframes and break the traditional mould of web design

2. Make your website faster without scrimping on design and functionality

3. Progressively enhance and go from website to progressive web app with offline support

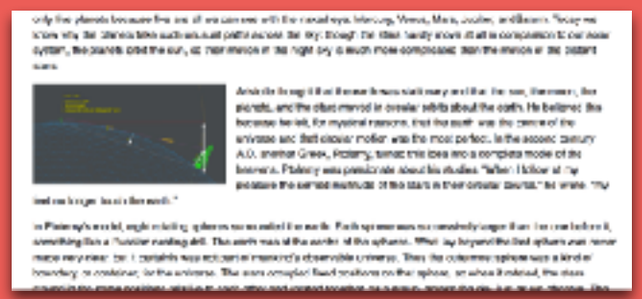
4. Handle more than just the keyboard and mouse — effective ways to deal with the new wave of input methods - and more specifically a range of new out put methods.

CSS Grid Layout

Let’s get to know how to use CSS Grid Layout…. but before then lets see how we got here.

Problem with floats.

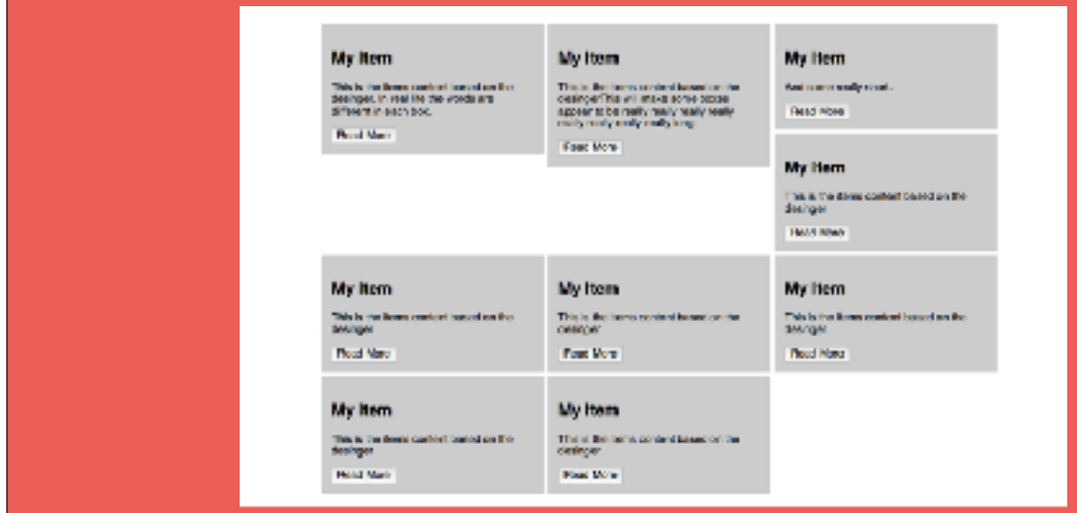
The main problem with floats is that they were never designed for laying out the structure of our website. Someone found way that we could use them to allow for this…. but it never was the intension.



Floats were created to do two fundamentally basic things, very similar to each other.

Float an image to the left. and, float an image to the right.

Separate Content from Design



Then we move into the real world and add real words to the mix and end up with this. Some are really really long and others are really really short.

After many years of using floats for layout we were given a shimmer of hope…

CSS Flex Box

1 Dimensional Layout

Flex box is a one dimensional layout that provided us with the ability to start stepping away from floats.

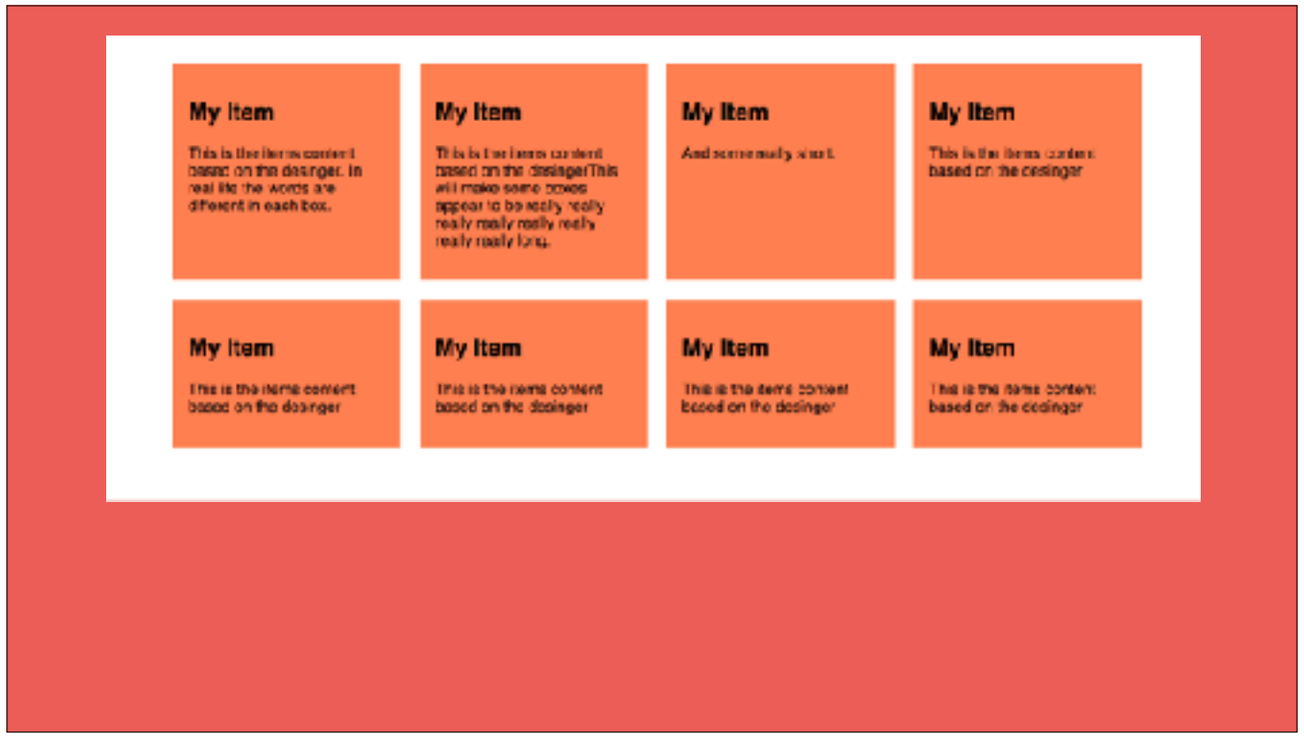
When you apply flex box to a container you tell it to go along the x axis or the y axis.

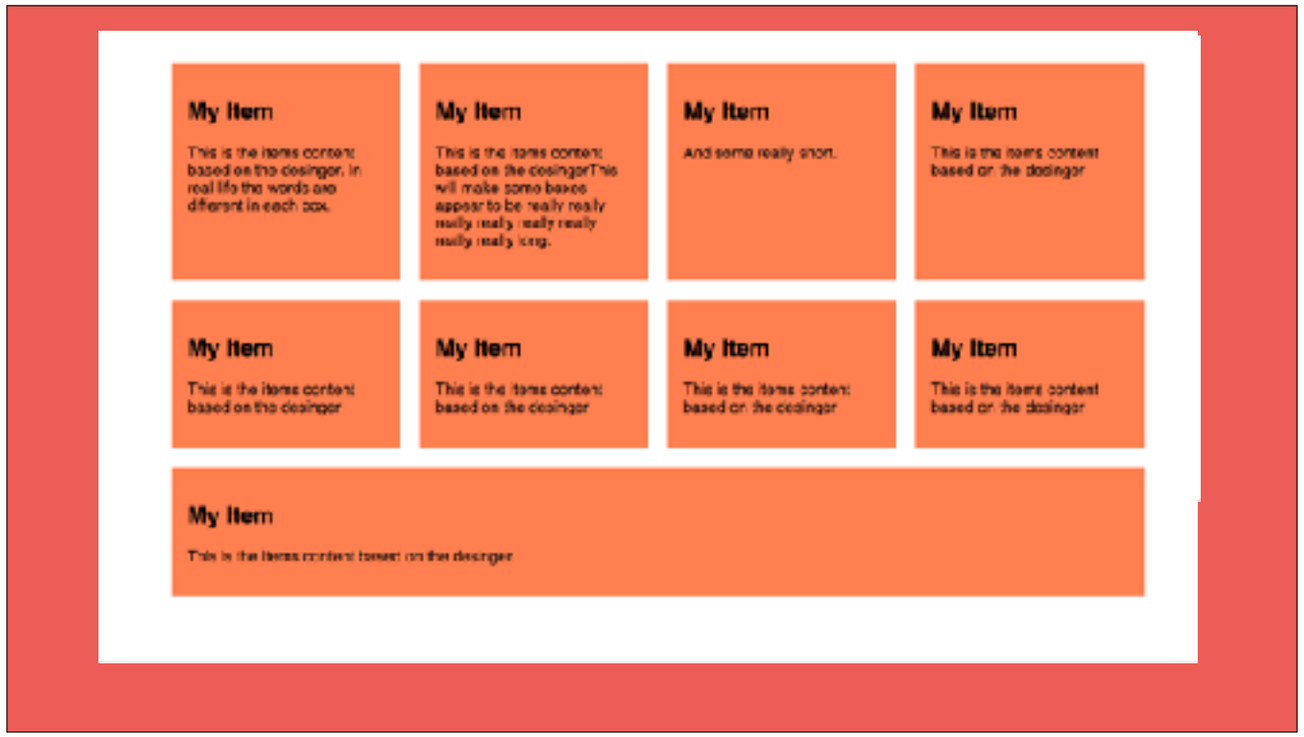
```
<div class=“container”>
    <div class=“item”></div>
    <div class=“item”></div>
    <div class=“item”></div> <!--8 items -->
</div>
```

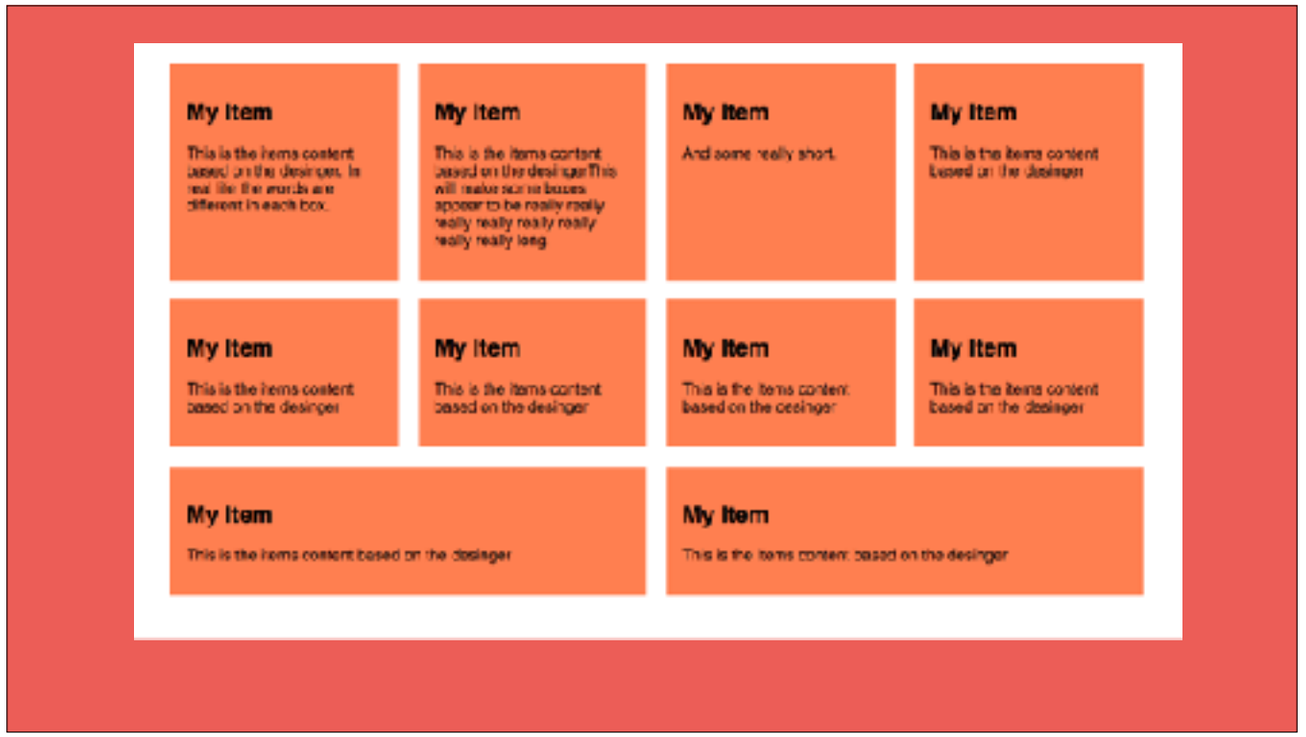
```css
.container {
    width: 95%;
    max-width: 1024px;
    margin: 1em auto;
    display: flex;
    flex-wrap: wrap;
  }
```

```
.item {
    padding: 1em;
    margin: 1%;
    flex-basis: 15em;  /* be 15em wide   */
    flex-grow: 1; /* Grow full width  */
    flex-shrink:0; /* Don't shrink.  */
  }
```


update the comment on the slide


My Item
This is the items content based on the designer. In real life the words are different in each box.
My Item
This is the items content based on the designerThis will make some boxes appear to be really, really, really really really really really long.
My Item
And some really short.
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer


My Item
This is the items content based on the dosingor. In real life the words are different in each box.
My Item
This is the items content based on the dosingorThis will make some boxes appear to be really really really really really really long.
My Item
And some really short.
My Item
This is the items content based on the dosingor
My Item
This is the items content based on the dosingor
My Item
This is the items content based on the dosingor
My Item
This is the items content based on the dosingor
My Item
This is the items content based on the dosingor
My Item
This is the items content based on the dosingor


My Item
This is the items content based on the designer. In real life the words are different in each box.
My Item
This is the items content based on the designer.This will make sure boxes appear to be really really really really really really long
My Item
And some really short.
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer


My Item
This is the items content based on the designer. In real life the words are different in each box.
My Item
This is the items content based on the designerThis will make some boxes appear to be really really really really really really really long.
My Item
And some really short.
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer
My Item
This is the items content based on the designer

** Add in the RWD Breakpoints for the floats back a few slides.

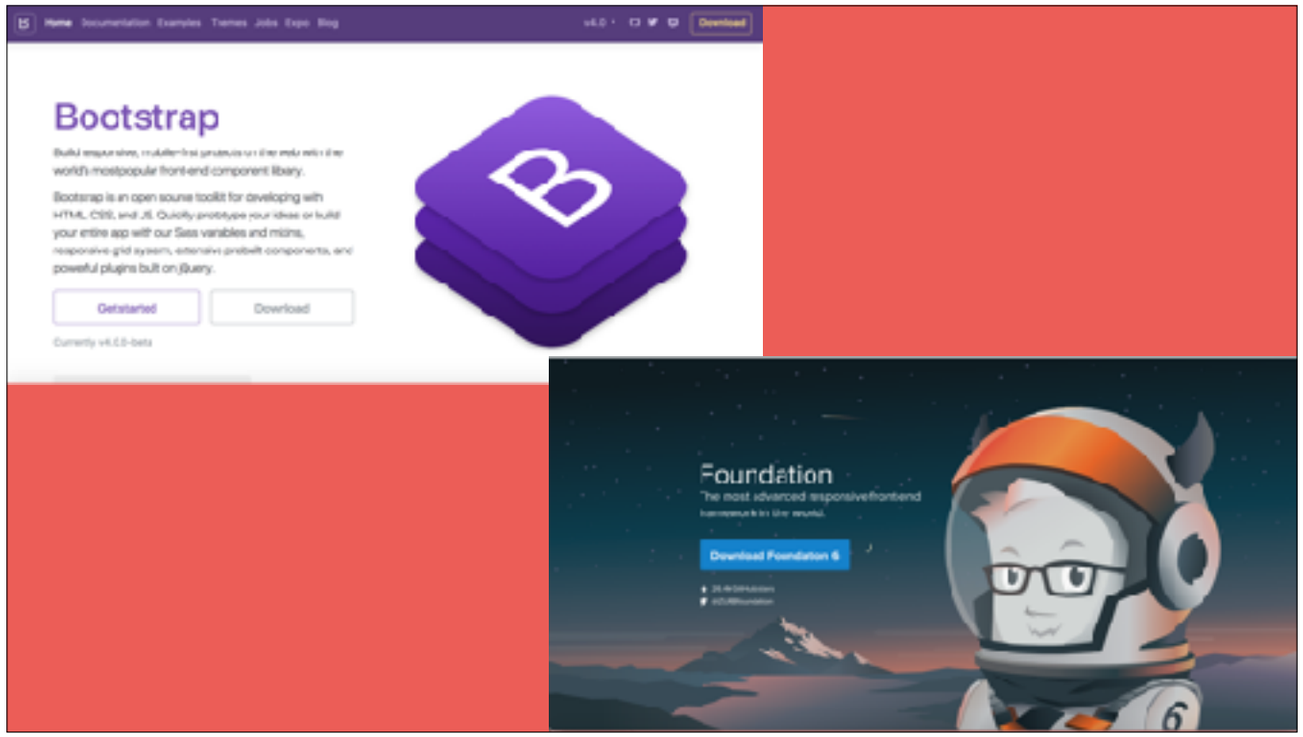
Need breakpoints to change the width of the items because I’ve given int a flexible width, but lets see how this is done with fixed items.

** Flex ignores floats.

** Add in the RWD Breakpoints for the floats back a few slides.

Need breakpoints to change the width of the items because I’ve given int a flexible width, but lets see how this is done with fixed items.

** Flex ignores floats.



There’s a few people that have based their previously really cool FLOAT based layouts on Flex Box because it’s far easier to lay out a page in that way.

BUT this is still a hack way of approaching the issue. FlexBox was not designed for overall layout of the page, but instead to allow us to fill the unknown areas of a page with unknown amounts of content. Not Grid, which inherently means structured, but FLEX which comes from the English phrase known as FLEXIBLE.

Grid is heavily supported now and there is *very* little reason to not be including it on your site instead of floats, or at least as part of some layout areas that you would use floats for traditionally.

There are some amazing teaching resources out there as well, for example flex box froggy is a journey where you try and return the frogs to their lily pads.

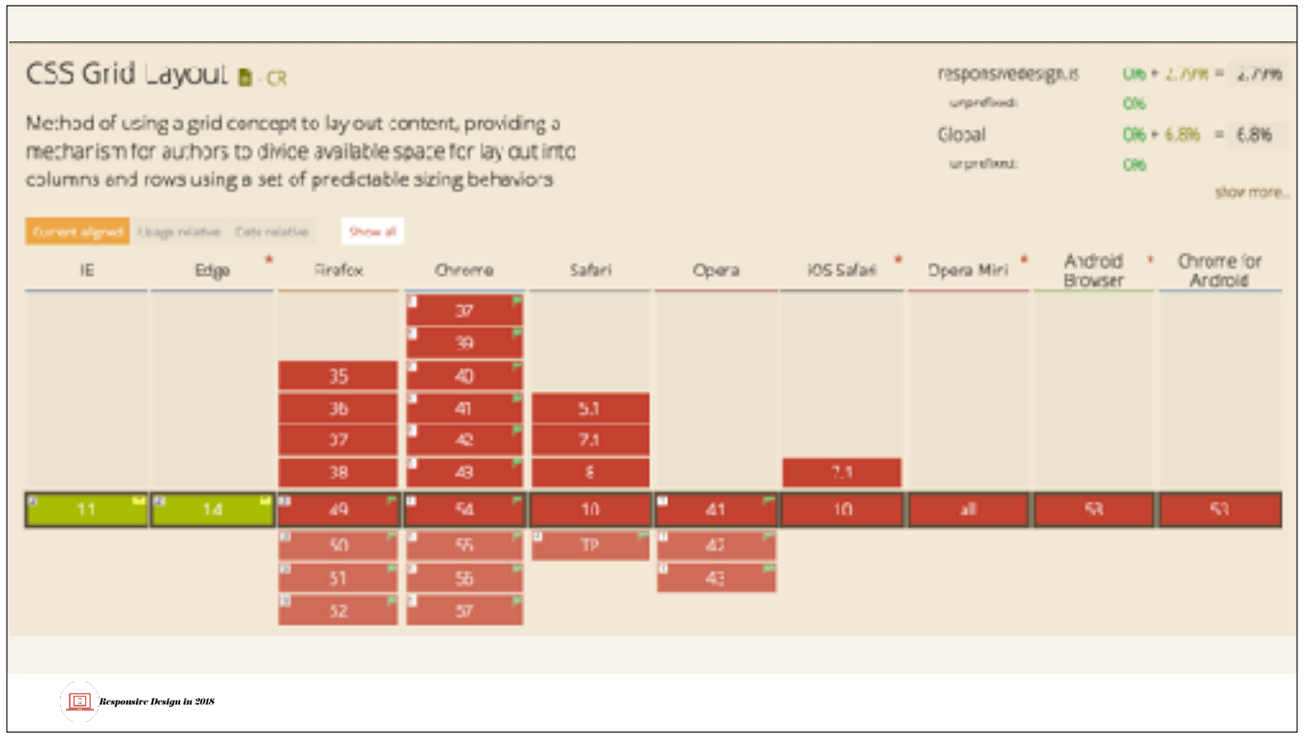
Flex Direction: Row, along the X Axis, **Reverse** because you want them reversed.

Justify Content - I get mixed up with this being vertical all the tme. Think of setting justify content pushes your content horizontally.

Align-items: get the content to the end. Note that it’s not bottom or top, it’s start and end.

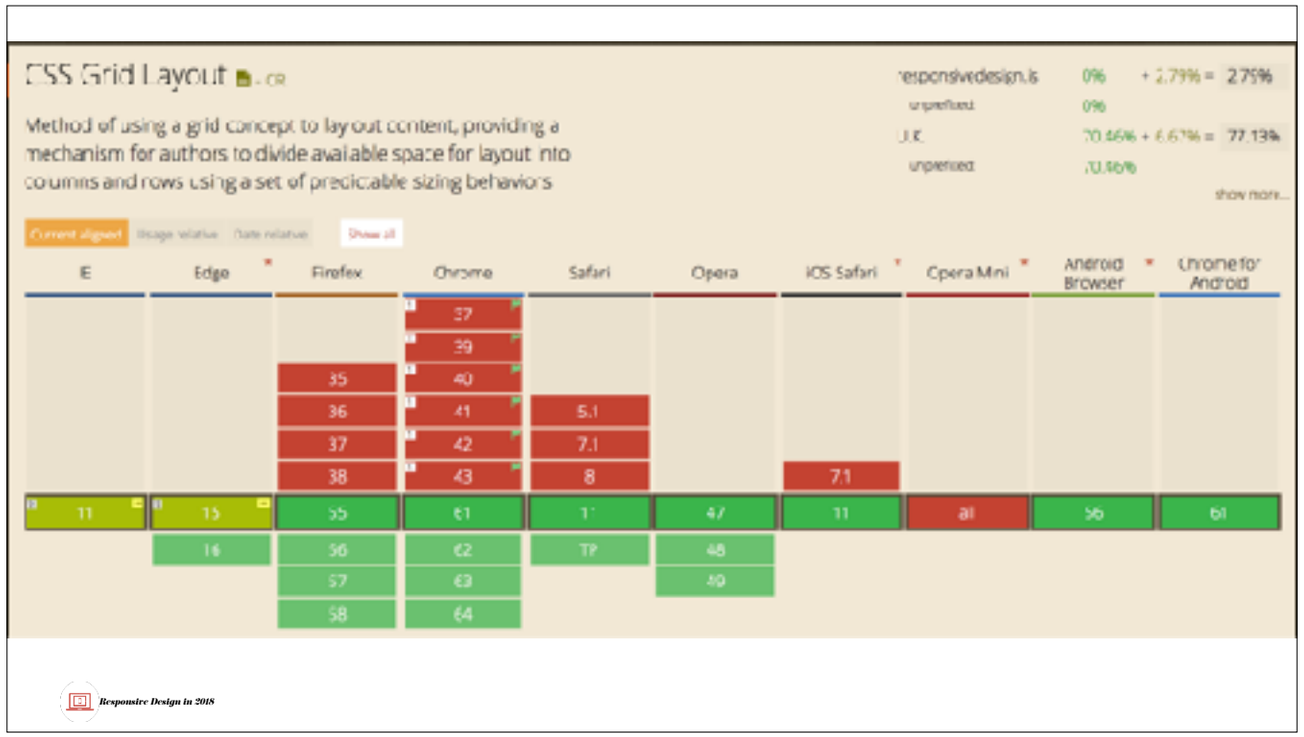
CSS Grid Layout

2 Dimensional Layout



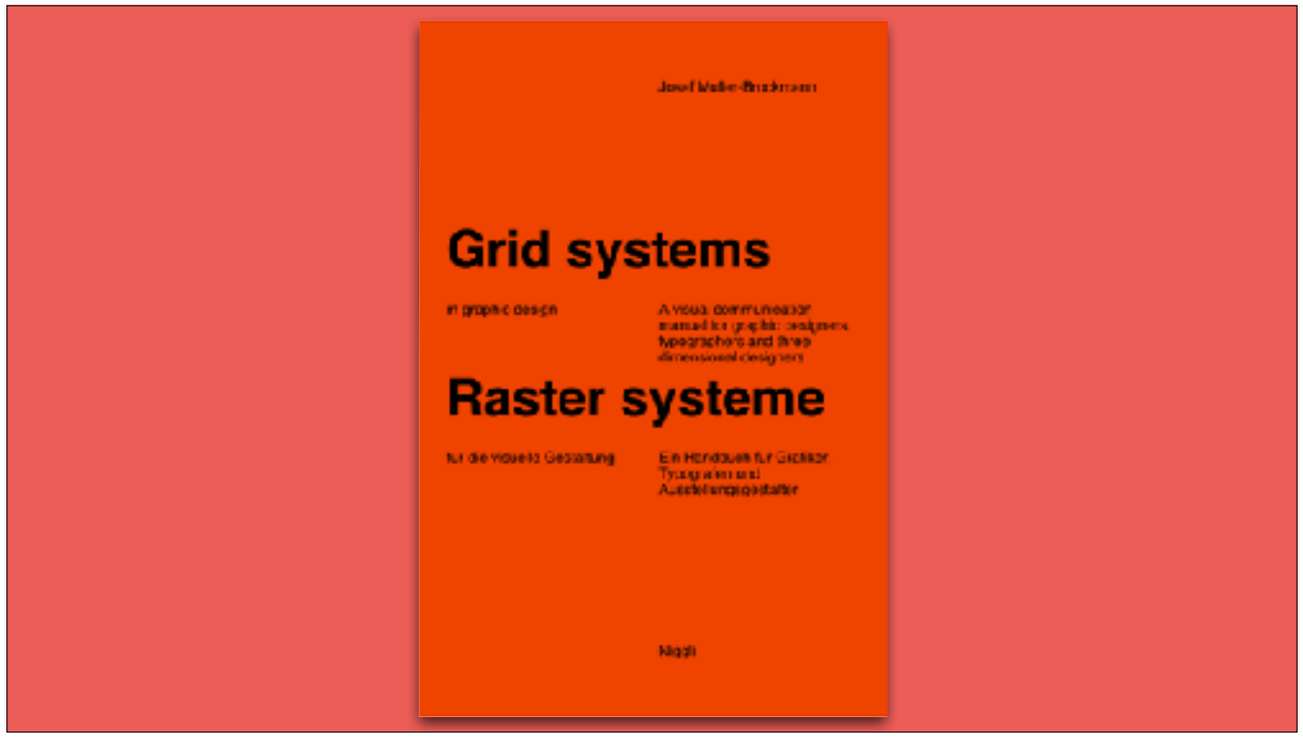
Last year I stood on stage and said that the support for this was horrible, but that Grid isn’t super easy though and takes some time to get your head around,

I recommended going and learning now and when it hits the browsers early next year you are in a good place.

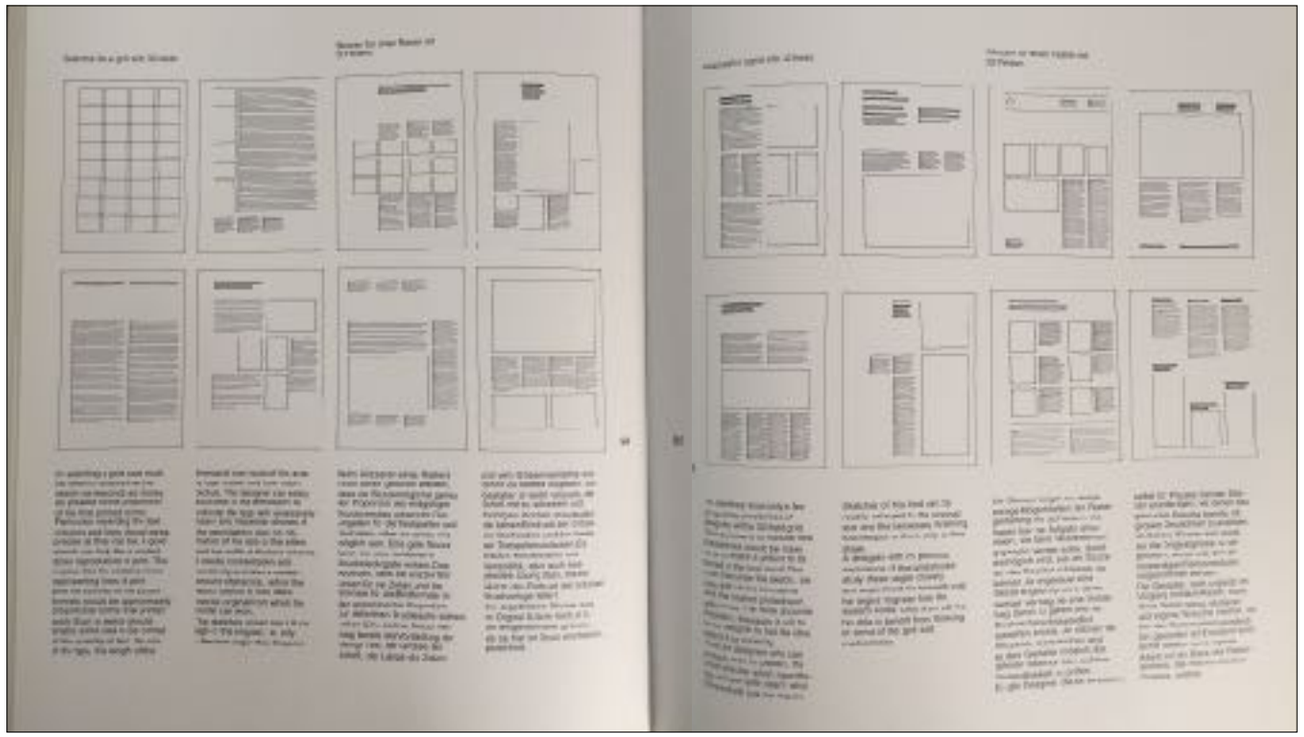


Well now that it’s a year later we can see that it’s lit up green on every major current browser, and only Opera Mini doesn’t offer support (although we’re likely to tube our content on this one.

To get thinking about layouts in a Grid System way I turned to the best possible book to get started.



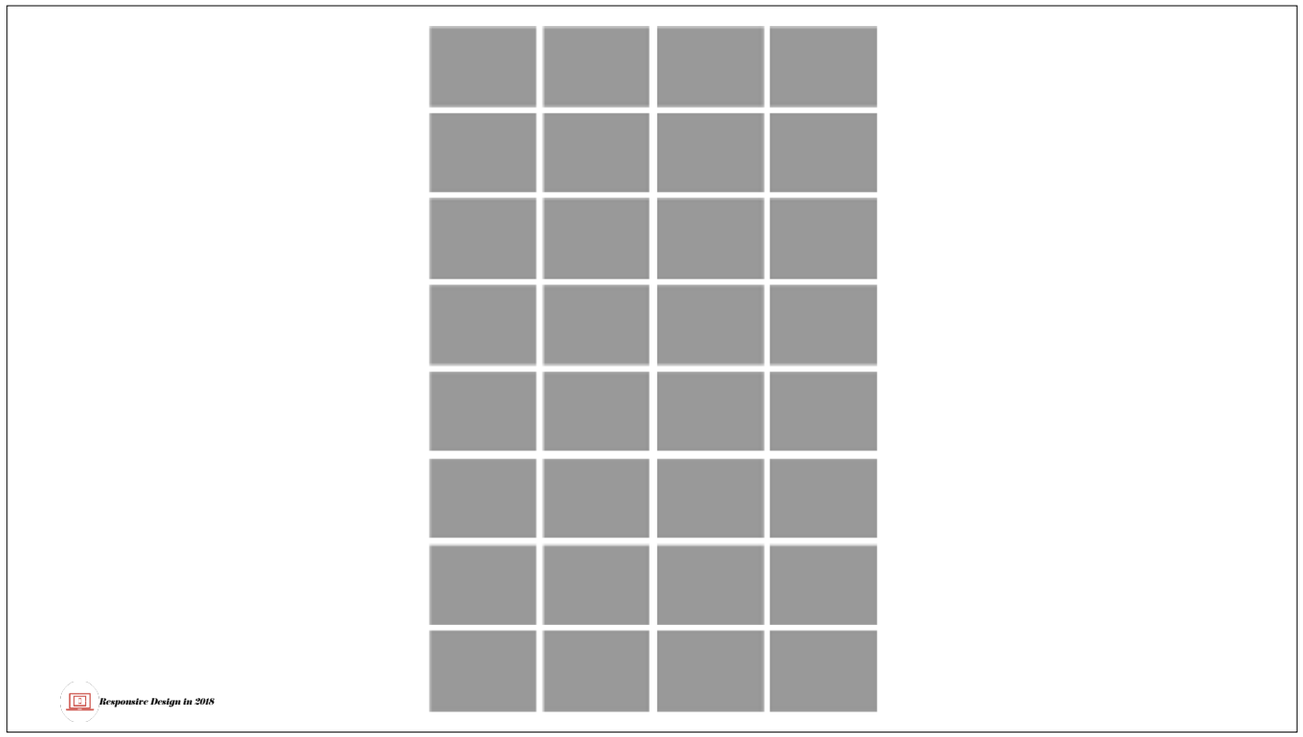
This book by Josef Muller-Brokmann is the default reading for Graphic Design Students and remains the number one recommendation whenever I ask people where I should get more information about Grid layouts.



Inside the book are tonnes of pages of examples of Grid based layout that you can choose from. A really great source of inspiration when you’re trying out the Grid Layout.

These pages in particular were of interest to me because it showed a number of different possibilities for laying out a grid.

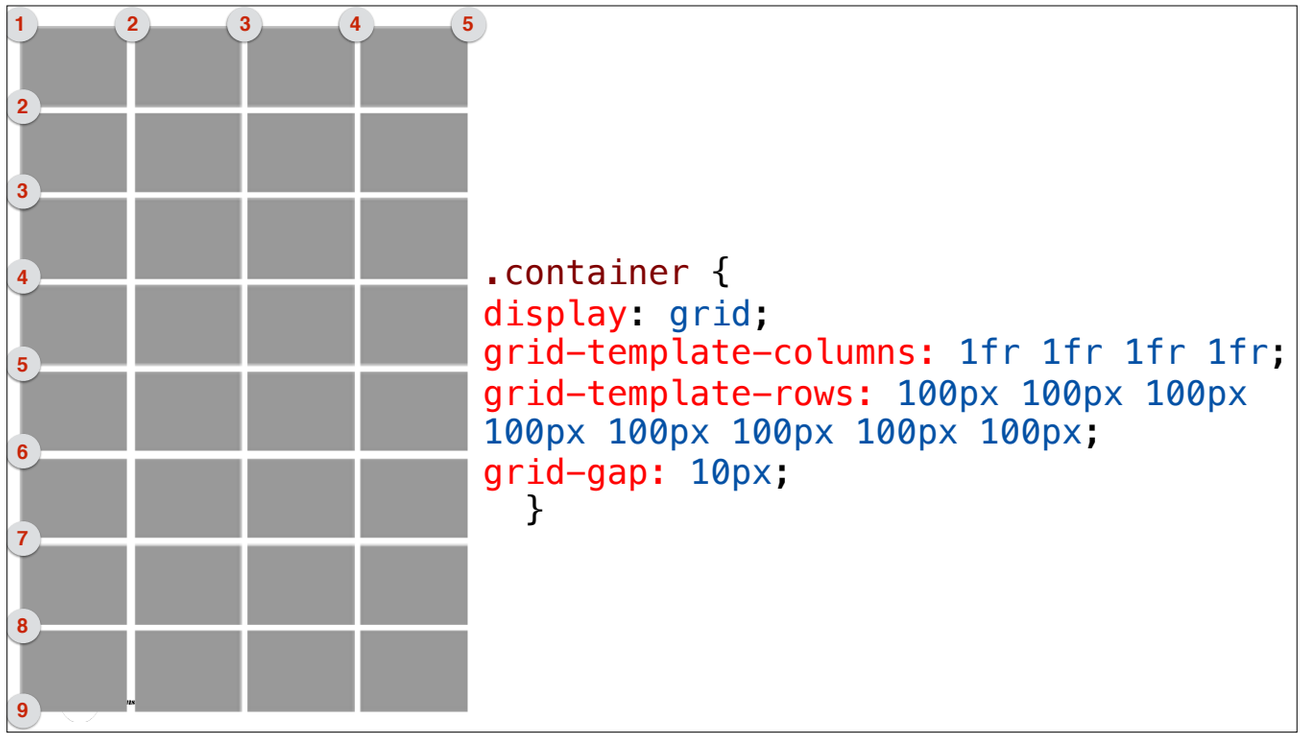
I took what I thought would be my most challenging to start with….



Which of course was the very simplest example on the pages. But simple is where we want to start when learning something new, and from that base we can extend our capabilities.

So here’s a straight forward 4 x 8 grid.

Let’s look at how we can lay this out using CSS Grid Layout.



A ***.container*** which I've literally named container. This is the parent. We apply the **grid** to the **parent** which affects **all direct children**.

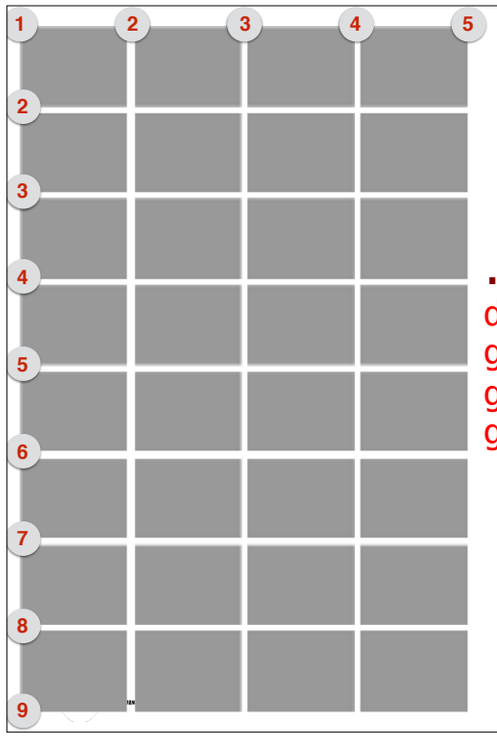
Then we use ***grid-template-columns***: to define how many **columns** and how wide we want them.(fr is a new unit, it equates to 1 fraction of the screen width… so four columns that take up 1 fraction of the width)

Then we use ***grid-template-rows***: to define how many **rows** and how tall we want them. I've put 100px eight times to get 8 * 100px rows. (here I've put in pixels).

grid-gap is used to, and some of you may have already guessed, gives us the gap between the grid items.

Now this is a little verbose, there's a lot of repeated items there, so we can do this….

Don't mention different values yet




```
.container {
display: grid;
grid-template-columns: repeat(4, 1fr);
grid-template-rows: repeat(8, 100px);
grid-gap: 10px;
  }
```

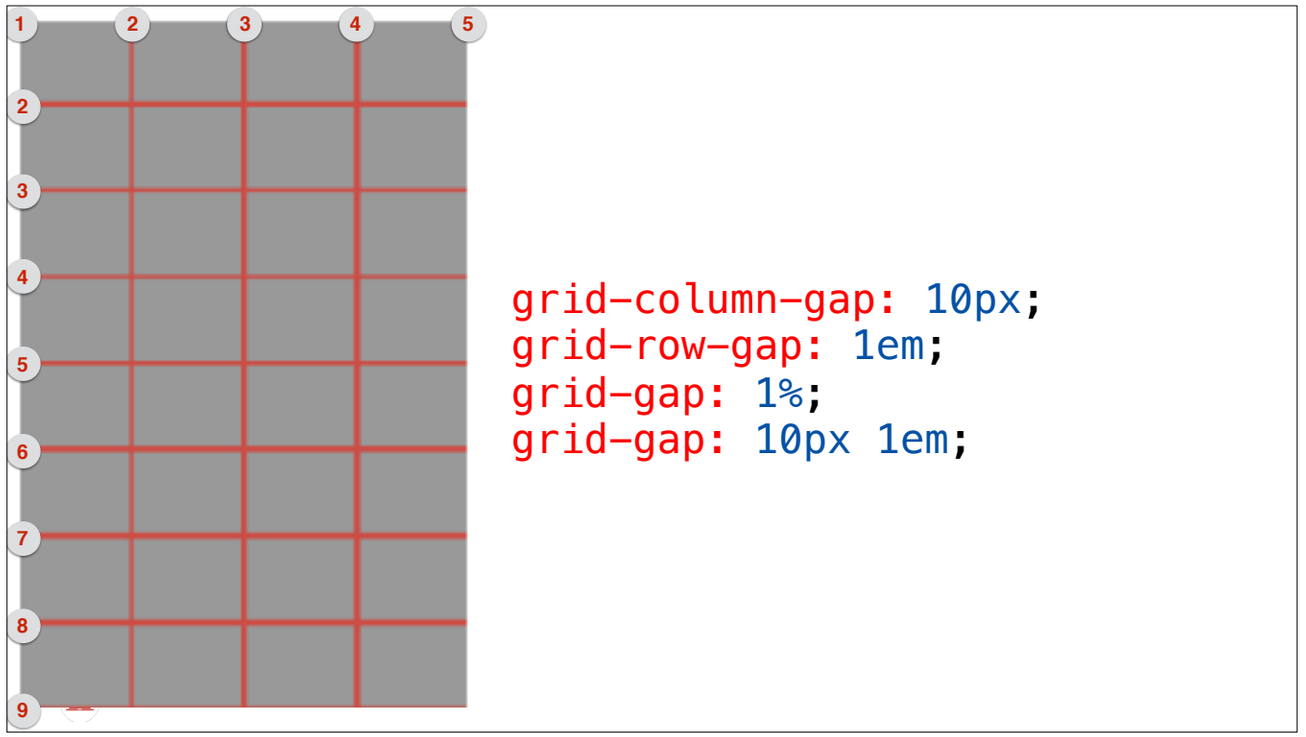
We can use **repeat** which is followed by the number of times you want to repeat it, and then the value you want repeated.

This is great because it makes it easier to write AND easier for someone to look at and understand.

You might have noticed the numbers appear here as well.

These are gridlines.

When defining a grid we define the columns and rows, but a grid item is made up from a **grid-column-start** and **grid-column-end** line, and a **grid-row-start** and **grid-row-end** line.

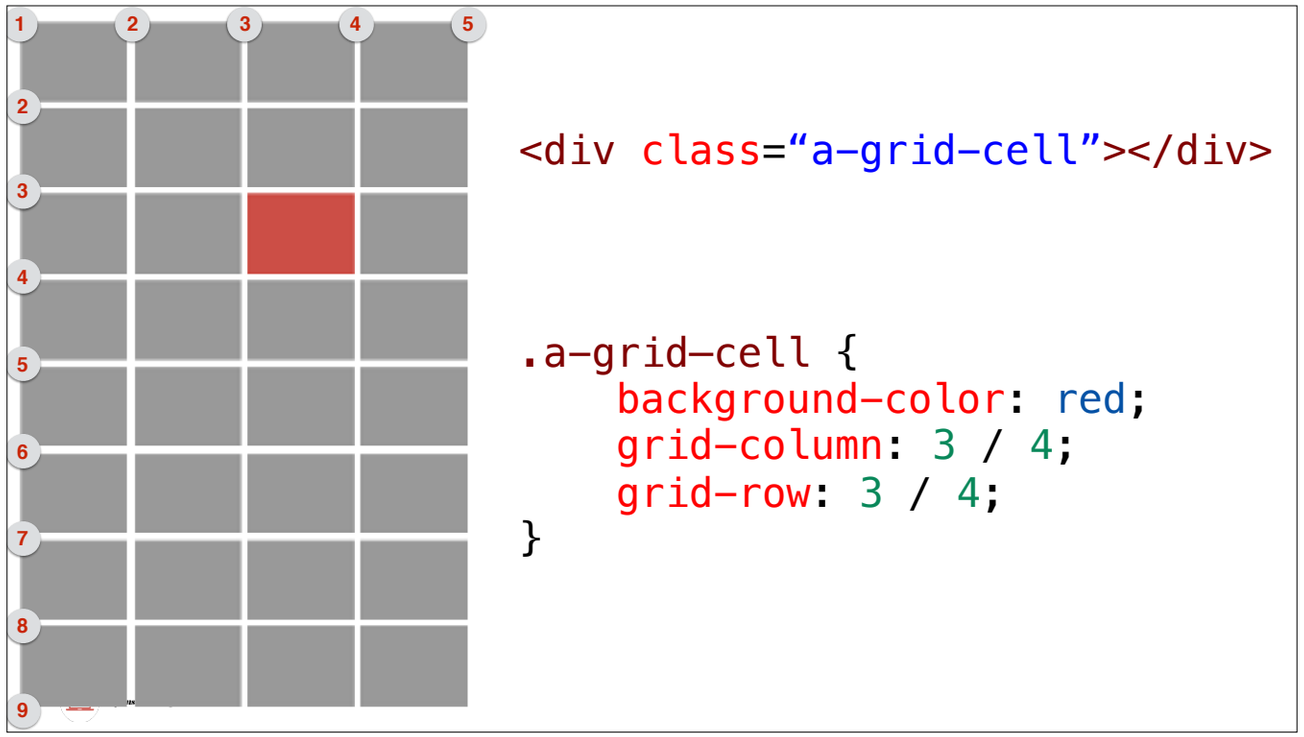


You can set the ***grid-row-column*** and ***grid-row-gap*** individually, you can set it as a single value that applies to everything, or you can set two values on '***grid-gap***' for **column** and then **row**.

As you can see from the markup here **you can apply any value** to the gap, and **even mix the values as well**.

This is **true** for **all size values** in CSS Grid Layout, you can have a **mixture** as we'll **see in later examples**.

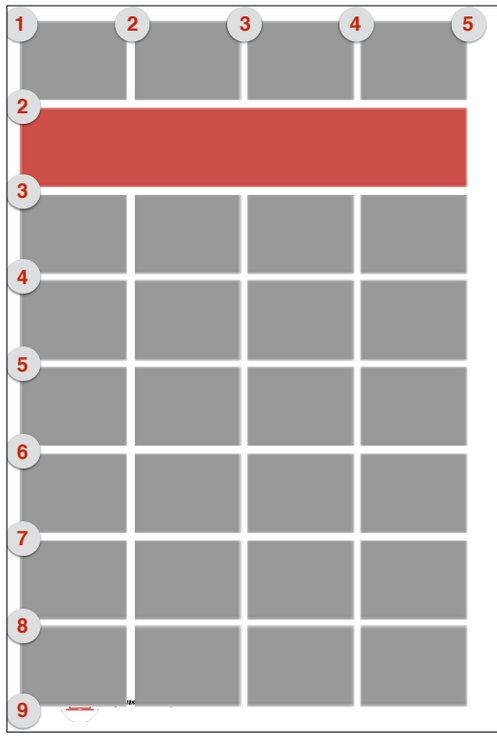
You should note that although you can see the **gridlines** here in red, it's **not possible to style gridlines**. If you set a ***background-color*** to the container and occupy all of the grid spaces then you will be left with the gaps as you can see here.



In these examples we have the HTML at the top and the CSS at the bottom.

The **grid-column** starts at grid column line 3 and goes to line 4.

The **grid-row** starts at grid row line 3 and goes to grid line 4.



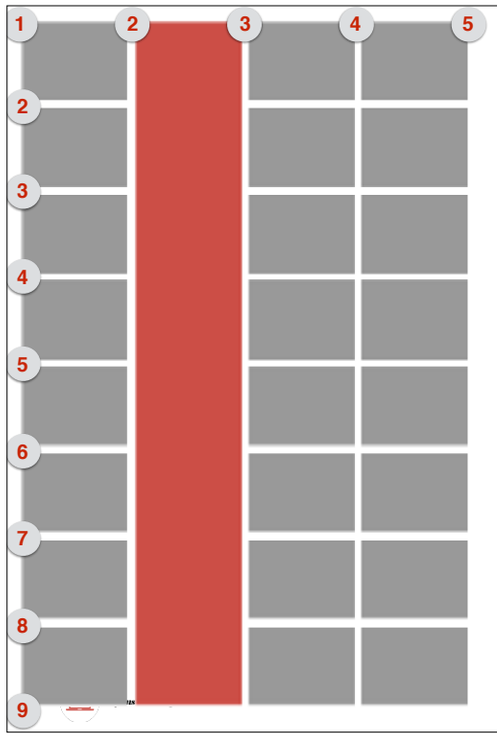

```
<div class="a-grid-row"></div>


.a-grid-row {
    background-color: red;
    grid-column: 1 / 5;
    grid-column: 1 / span 4;
    grid-column: 1 / -1;
    grid-row: 2 / 3;
}
```

Here we want this to occupy an entire row, i.e. span across all of the columns.

Here I've included the different ways you can accomplish this as well. With all great things on the web we offer several solutions to the same problem just to keep everyone guessing.

1 / 5 starts and grid line one and goes to grid line 5.
1 / span 4 starts at grid line 1 and spans 4 lines (which actually works out to be the number of columns you want to span, I like this one but it can be a little harder to think about what it's doing).
1 / -1 starts a grid line 1 and continues along until the end (or until it reaches another element placed on that row).

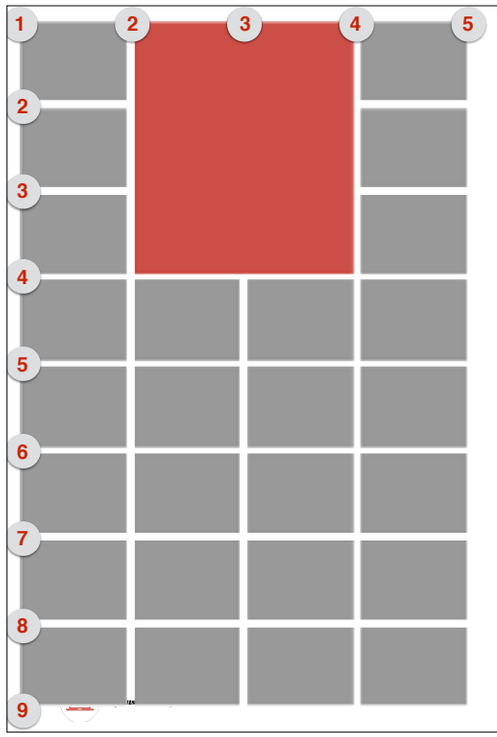
Finally, we say which row it is occupying.




```
<div class="a-grid-col"></div>


.a-grid-col {
    background-color: red;
    grid-column: 2 / 3;
    grid-row: 1 / 9;
    grid-row: 1 / span 8;
    grid-row: 1 / -1;
}
```

This is the same as the previous example, except across the other axis.

grid-column is specifying that it will go from 2 / 3, however if I just left 2 there it would still do the same thing because by default the item will span a single grid track in either direction.




```
<div class="a-grid-area"></div>
```

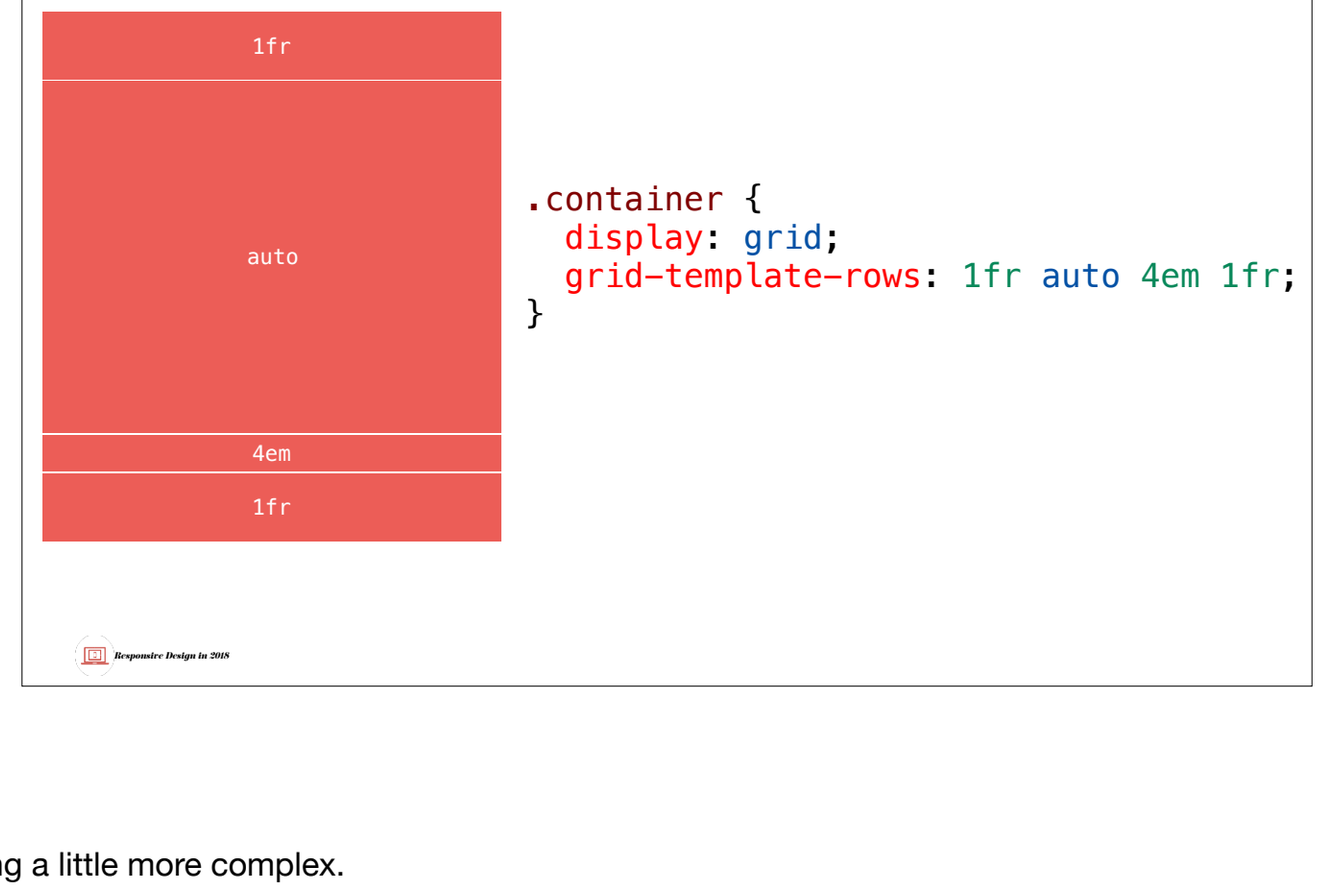
```
.a-grid-area {
    background-color: red;
    grid-column: 2 / 4;
    grid-row: 1 / 4;
}
```

You can also span multiple ***grid-columns*** and ***grid-rows*** .

You can overlap items and do a whole bunch more that I won't have time to delve into today.

I have said grid-area here as a way of describing a larger space, but there is also ***grid-template-areas*** that allow you to set up named regions on your layout.

Lets look at an example.



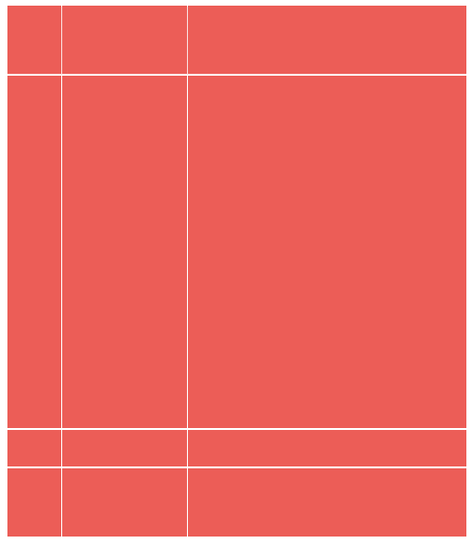
Lets create a new grid layout, something a little more complex.

Here i’m setting the .container to display: grid;, and setting up some rows to be 1fr, auto, 4em and 1fr.

Note that you can have different values? Then accept
FR, **VW**, **VH**, **PX**, **EM**, **REM**, and **auto**.

Then I declare the **grid-template-columns**, and I'm visually hiding the rows for now, or 6em, 1fr and 3fr.

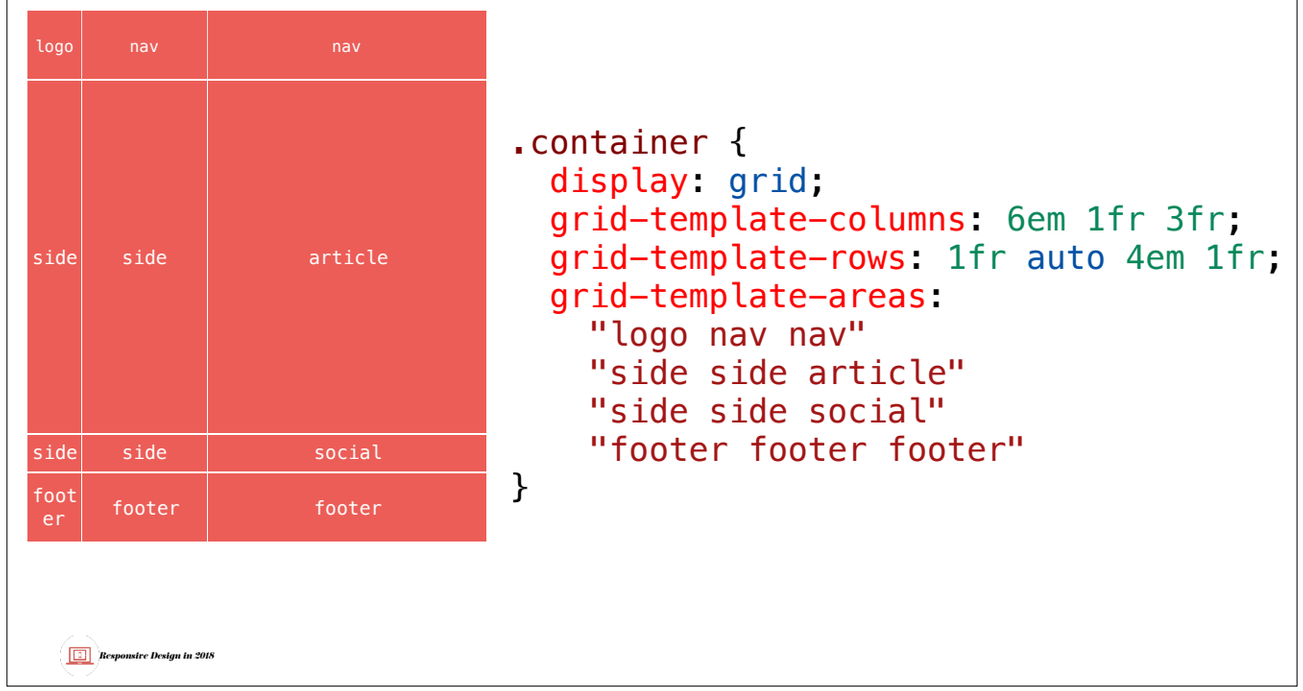
This is what it looks like with them both together.



```
.container {
  display: grid;
  grid-template-rows: 1fr auto 4em 1fr;
  grid-template-columns: 6em 1fr 3fr;
}
```


So now we can assign names to each of these areas, or a group of these areas.

Unfortunately we can only create two kinds of tetris pieces: the square and the rectangle, but it’s still pretty cool.

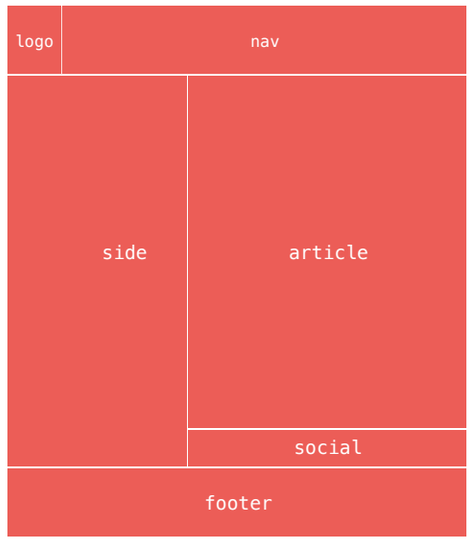


to do this we include ***grid-template-areas*** and define each of them by naming them in the CSS.

Here we've got a grid of 3 x 4, so we need to name three across and four down. We say we want

- logo in the first column on the first row
- nav to then go across the next two columns on the first row
- side should occupy the first two columns AND the first two rows
- article should occupy the third column in the second row
- social should go in the third column of the third row
- footer should span across the entire grid container

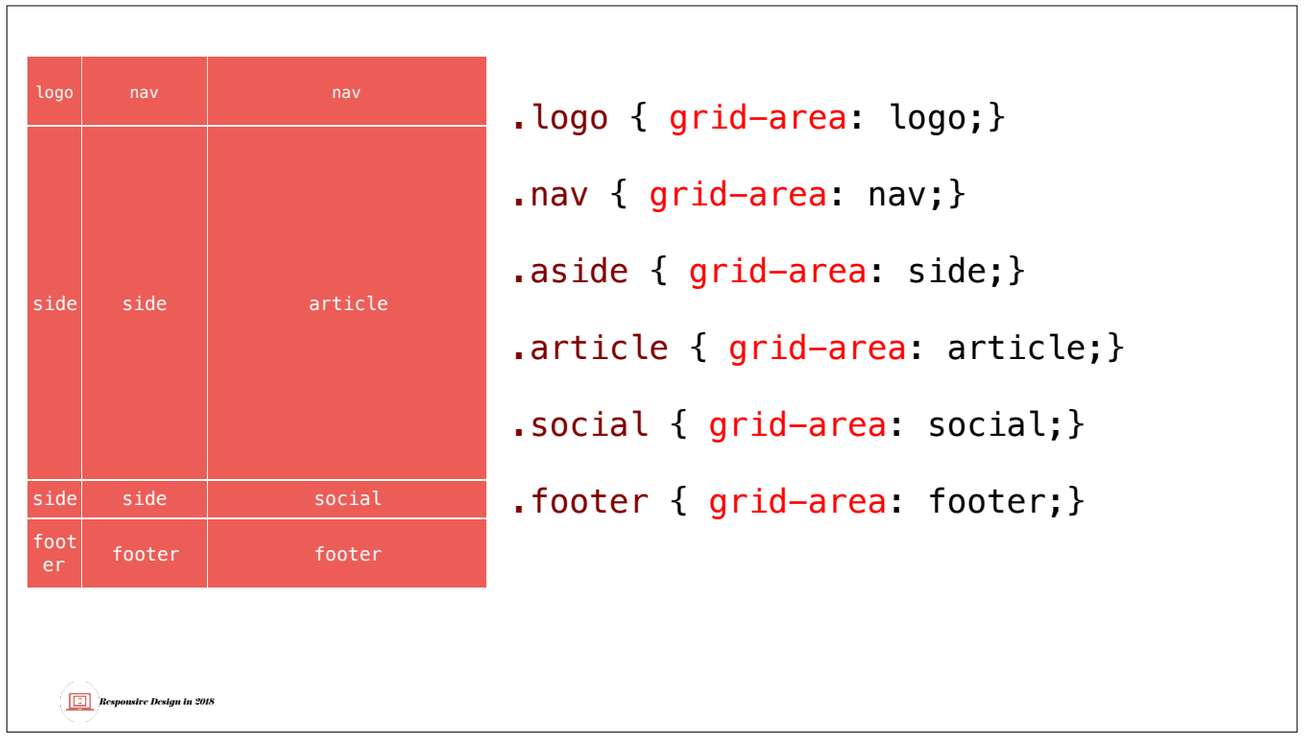
This essentially means that we have the following areas defined for our content….




```
.container {
  display: grid;
  grid-template-columns: 6em 1fr 3fr;
  grid-template-rows: 1fr auto 4em 1fr;
  grid-template-areas:
    "logo nav nav"
    "side side article"
    "side side social"
    "footer footer footer"
}
```

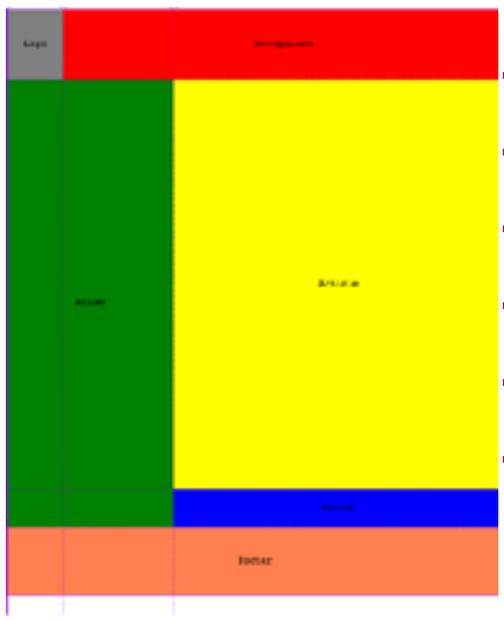

Those grid lines are still there, we’re still working on a 3 x 4 grid, but our grid areas are spanning more than one grid place.

Let’s go back to the other grid before we place our content.



Now for each of the bits of content on the site we can easily place them in their spot by using the ***grid-area: name***

logo - grid-area: logo;
nav = grid-area: nav;
aside = grid-area: side;

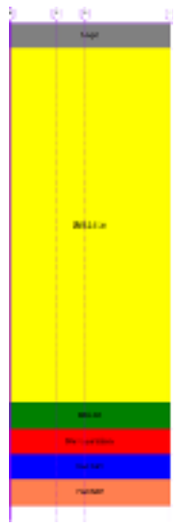



```
.logo { grid-area: logo;}

.nav { grid-area: nav;}

.aside { grid-area: side;}

.article { grid-area: article;}

.social { grid-area: social;}

.footer { grid-area: footer;}
```

Responsive Design in 2018

This is what it looks like inside of an actual browser.

This is great for the desktop, but what about if we wanted to set this up mobile first and update at the tablet and desktop?



```
.container {
    display: grid;
    grid-template-columns: 6em 1fr 3fr;
    grid-template-rows: auto;
    grid-template-areas:
      "logo logo logo"
      "article article article"
      "side side side"
      "nav nav nav"
      "social social social"
      "footer footer footer";
}
```

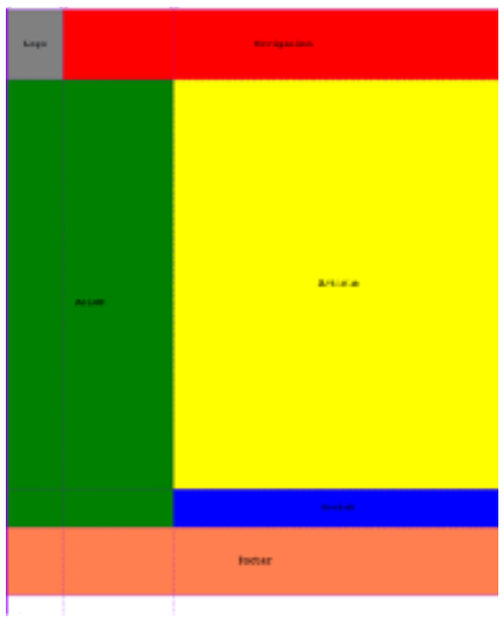

This would be our Mobile Layout.

I’m setting the ***grid-template-rows*** to auto because we need lots of them and I don’t want to restrict their height. This means that it will keep adding rows for each new grid item (it does that by default if you leave it out as well, but I included it to be clear).

Now we just set the order across the 3 columns again.


```
@media (min-width: 500px) {
 .container {
      grid-template-areas:
      "logo nav nav"
      "article article article"
      "side side side"
      "social social social"
      "footer footer footer";
      }
 }
```

At 500px and above I want to move the nav up into the first row, so I just update the ***grid-template-areas*** and it automagically moves to where it is told.

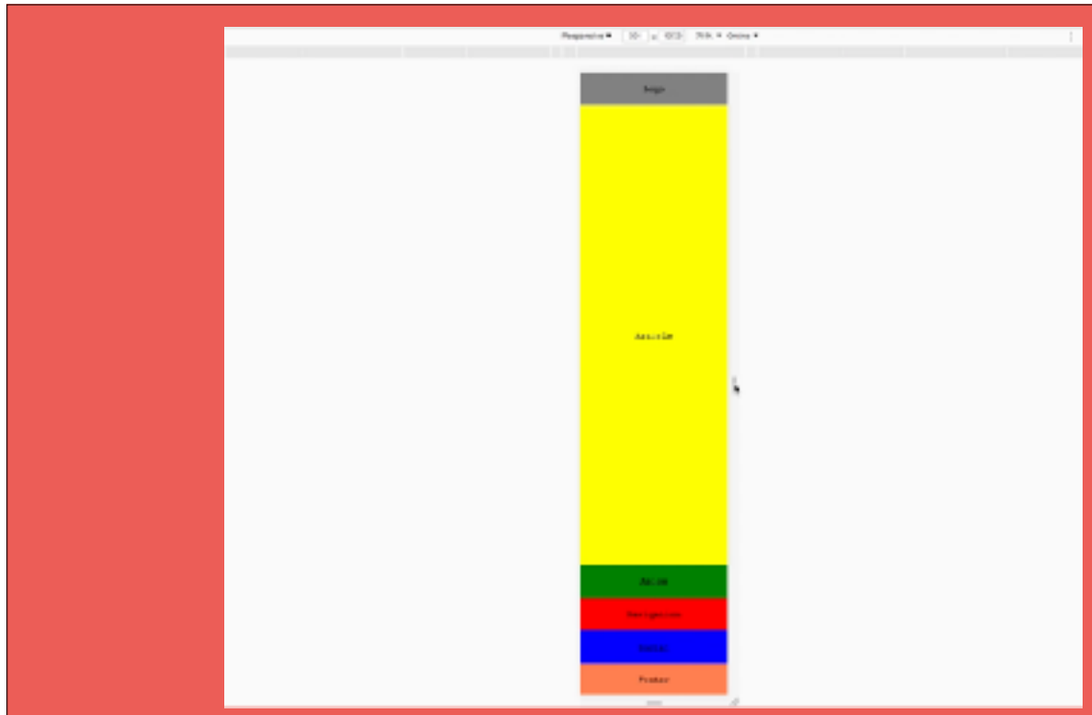



```
@media (min-width: 800px) {
 .container {
  grid-template-rows: 1fr auto 4em 1fr;
  grid-template-areas:
      "logo nav nav"
      "side side article"
      "side side social"
     "footer footer footer";
      }
 }
```

Responsive Design in 2018

Then at 800px I put in my final media query.

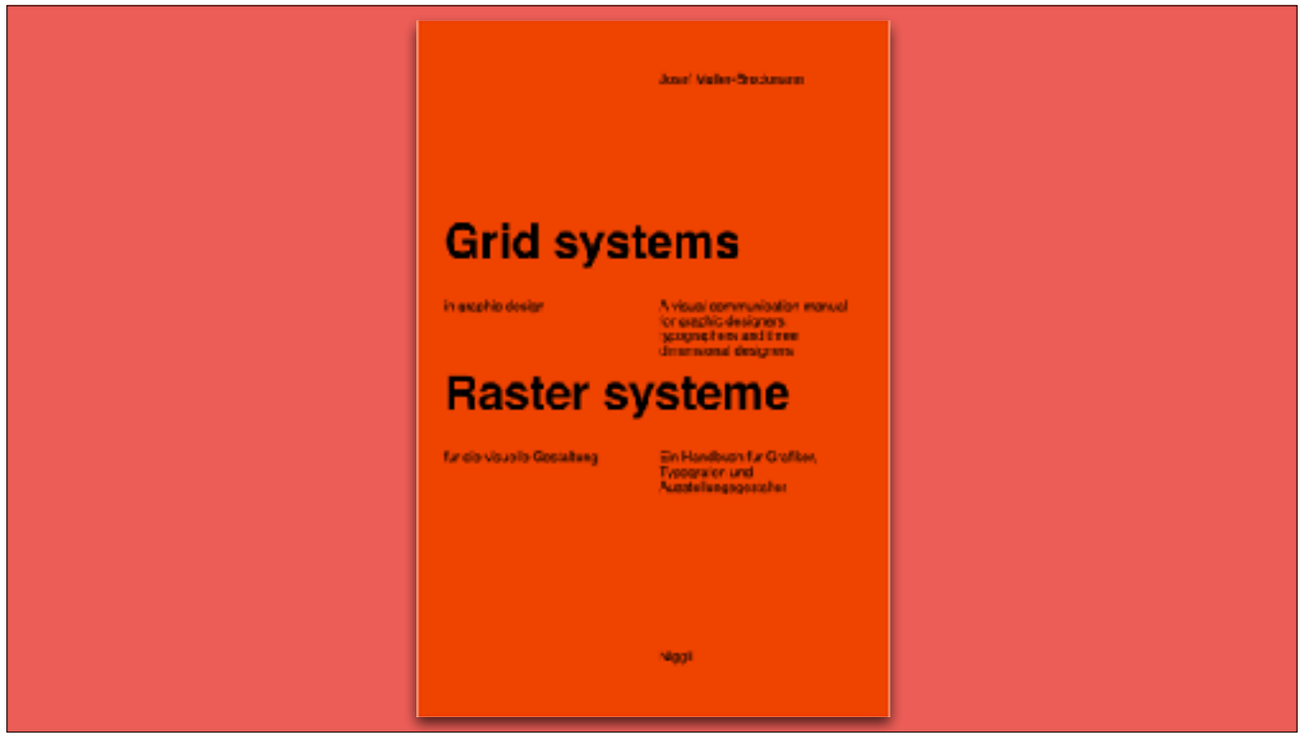
Here I set the ***grid-template-rows*** to the heights I want on desktop, and once again I change change the ***grid-template-areas*** to position the elements where I want them.



Here it is inside the browser.

A couple of things to point out.

1. This makes source order insignificant for layout. Because we are explicitly placing the content items onto the grid the HTML can be in any order you want, as long as it is a direct child of the grid container. So you could have
   LOGO ARTICLE ASIDE SOCIAL NAVIGATION FOOTER
   ARTICLE LOGO ASIDE SOCIAL NAVIGATION FOOTER
   I would suggest this is better for SEO to get your main content higher up the page.
2. You are not able to transition between grid positions. SUCKS. The positive is that it's just you and I who sit dragging our browser windows in and out, but still….

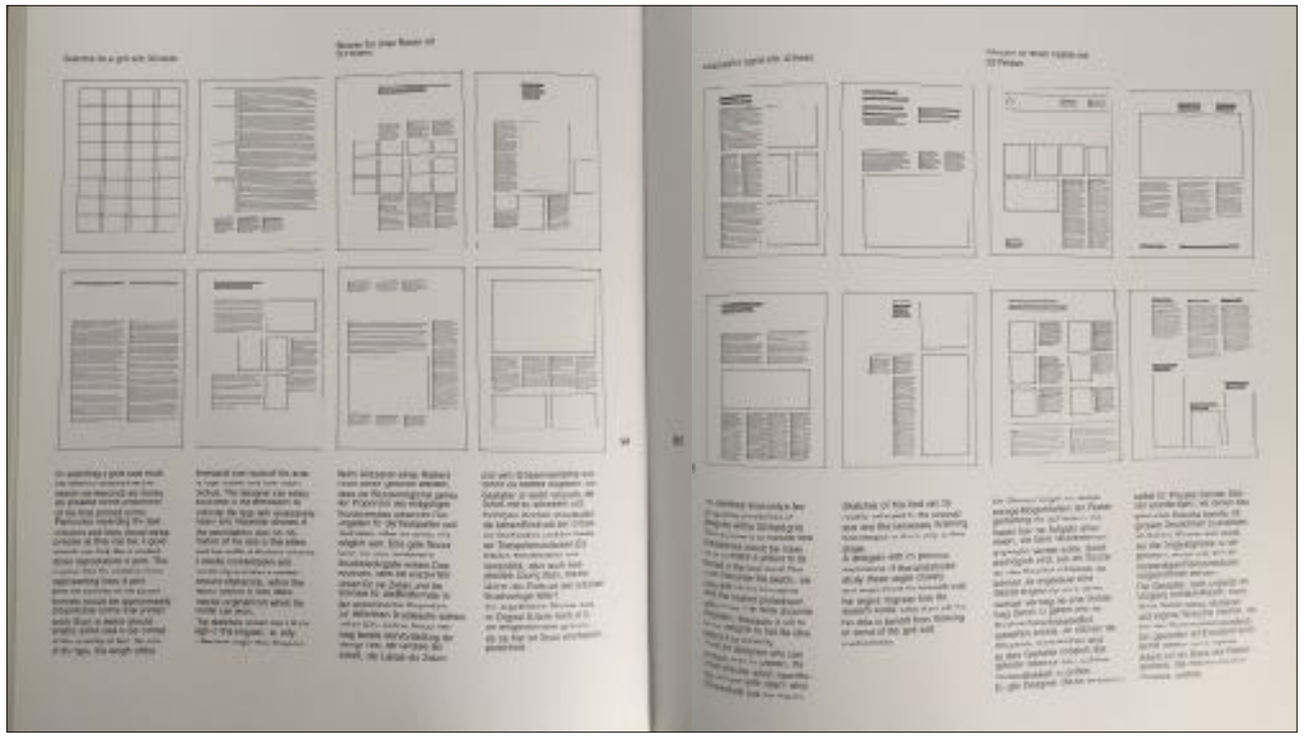


So if we go back to the Grid Systems book once again we can start looking through at different examples that we can try and create based on that 4 x 8 grid.

In fact, if you look at the cover of the book itself….

It is a 4 x 8 grid as well.

This isn’t the grid actually, but instead a code pen layout that I created when learning about how to use Grid Layout options.

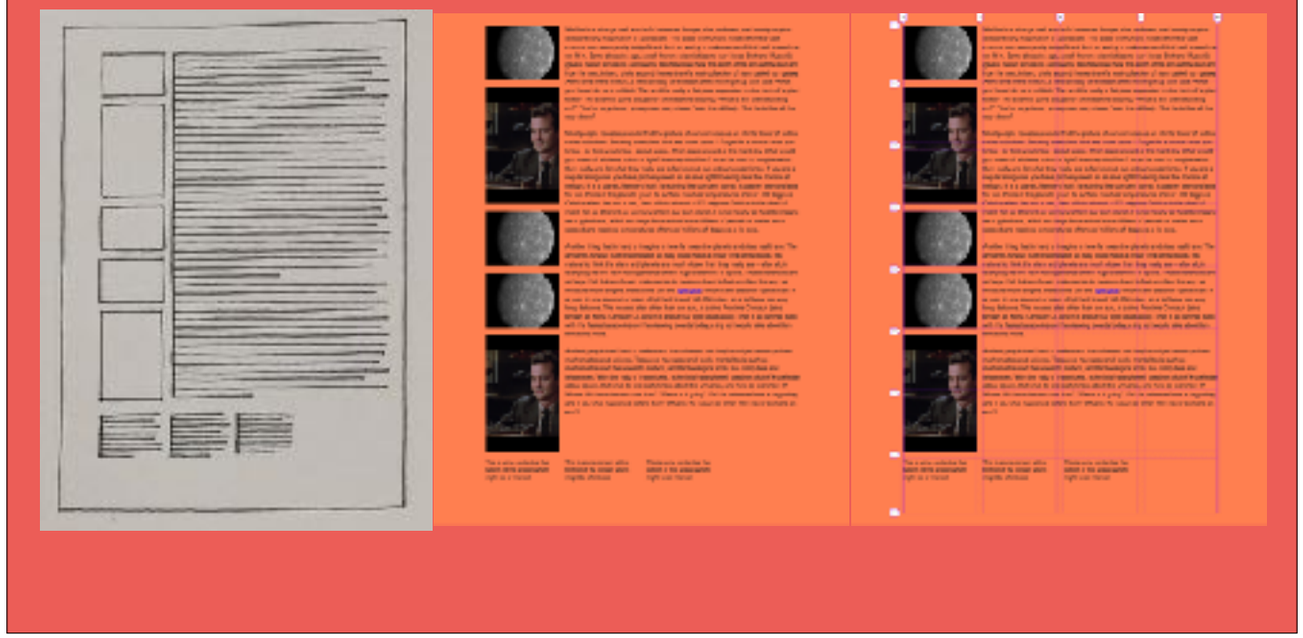


Lets look back at our page again and take some examples and how they might be created on the grid.

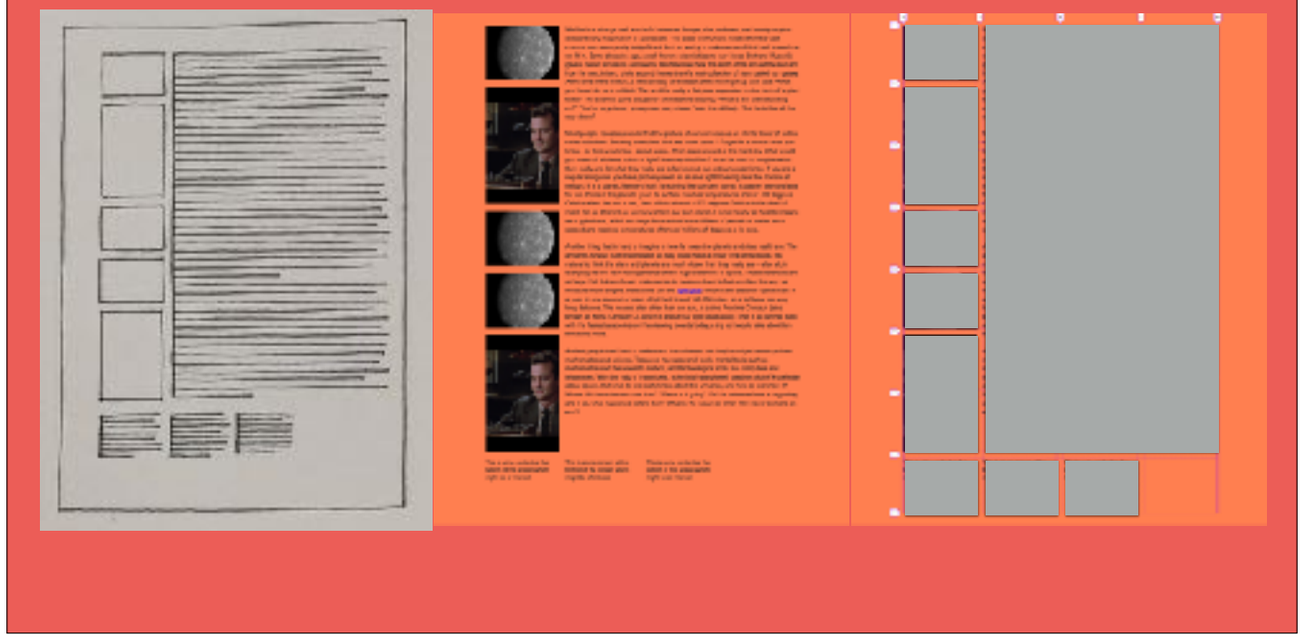
So what can we really do with Grid?

Well I was at a loss for a long time when a few weeks ago I had an idea. What would

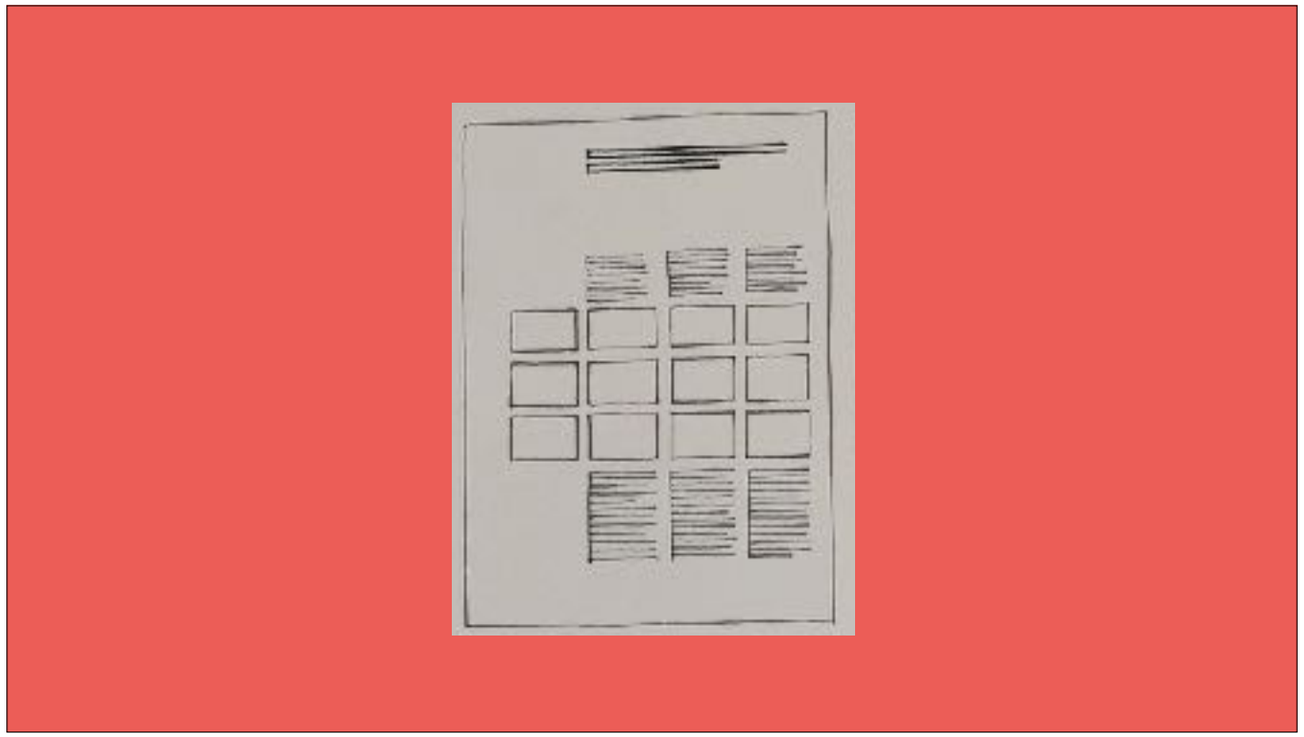
Aside from cleaning up the markup, and using true source order which is vital for screen readers and the future of Voice



This one has some images down the left and side and the content on the right.

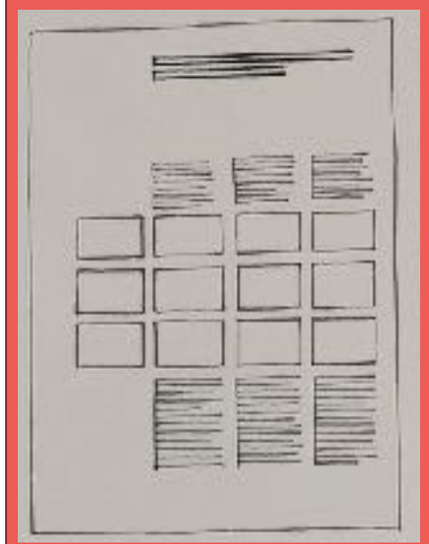


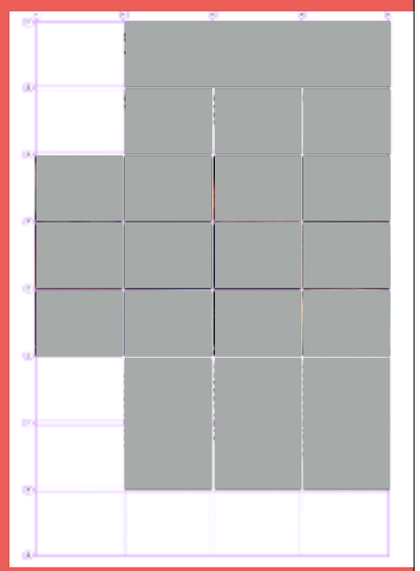
Here’s the grid areas blocked out for you to see.



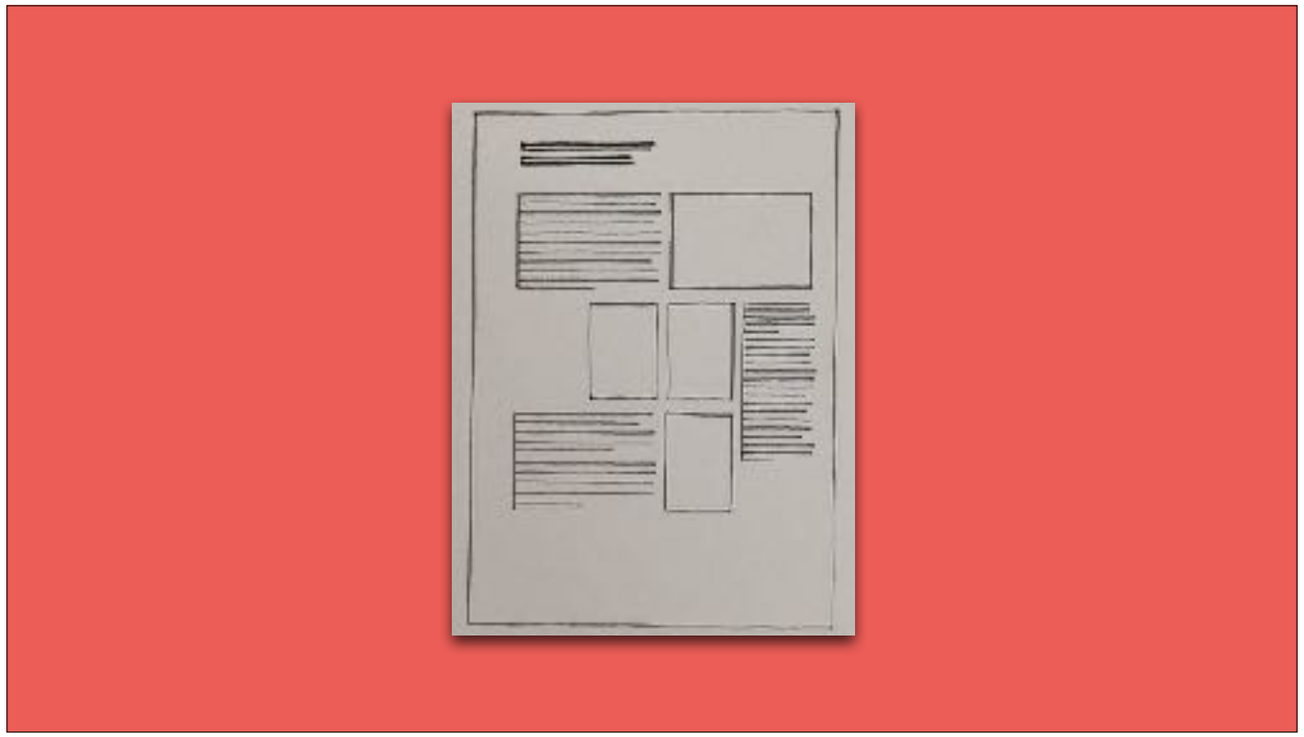
This is more of a Photo Grid layout.

Here’s the grid layout….

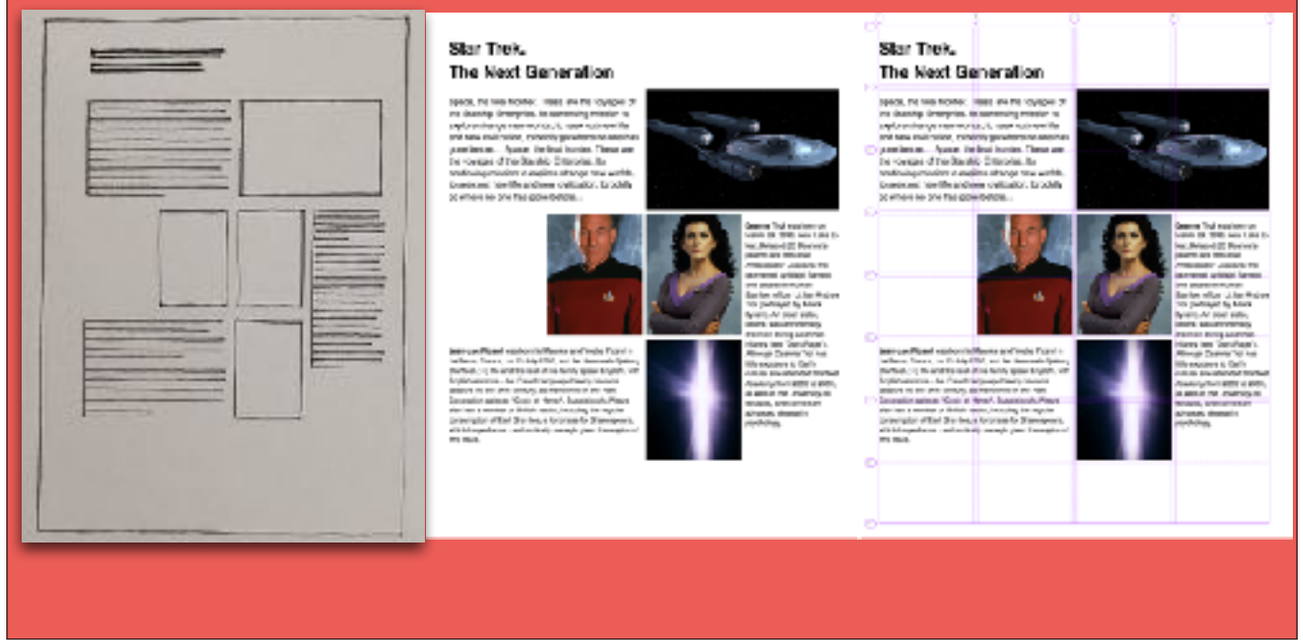




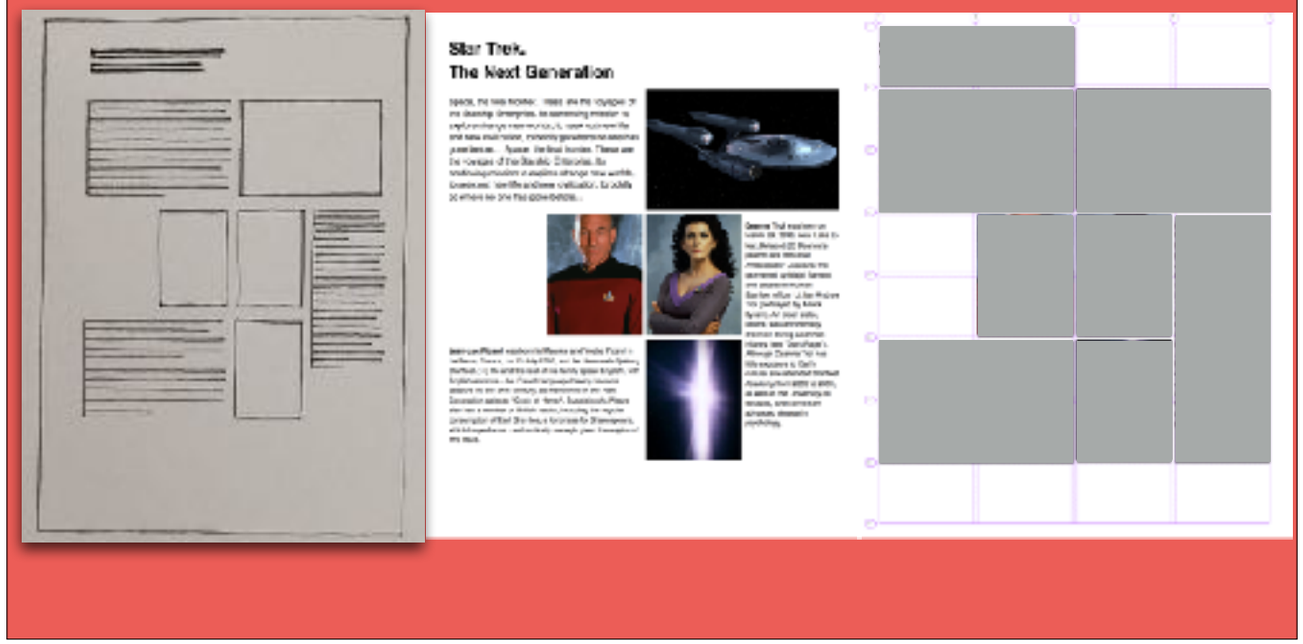
And the grid areas that I used to create them.



This was probably the most interesting layout on that page.

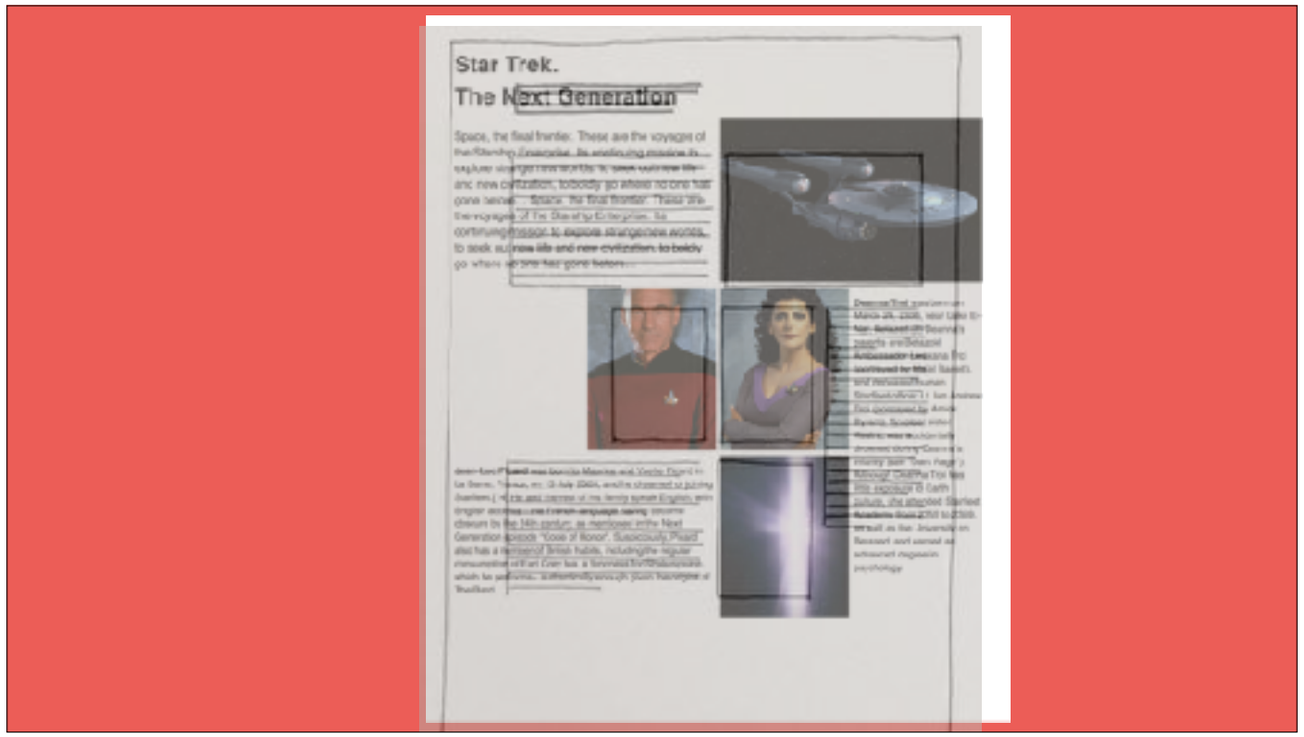


I have a bit of a **space theme** going on but…..

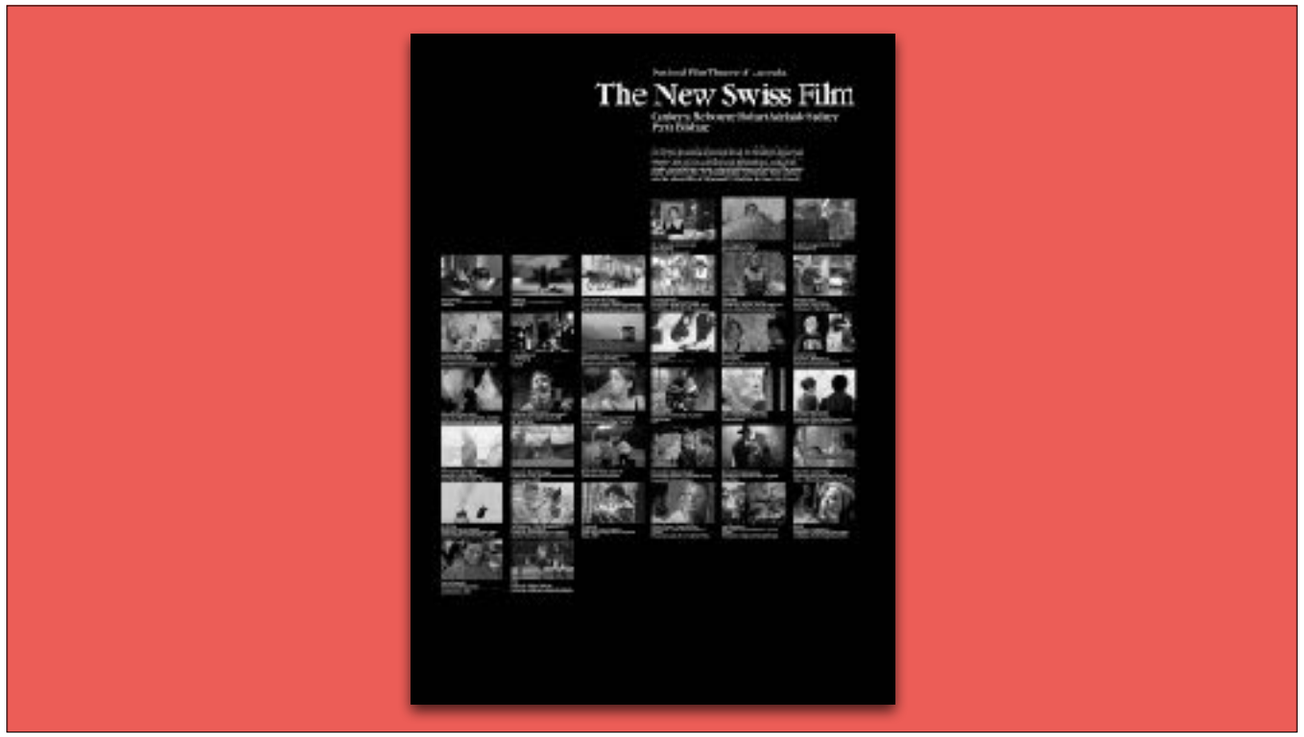


….but you can see how many **different kinds of layouts** that we can start to achieve with something as **simple** as a **4x8 grid** that took us a **few lines** to create.

This is what I’ve started doing with our digital designer at work.

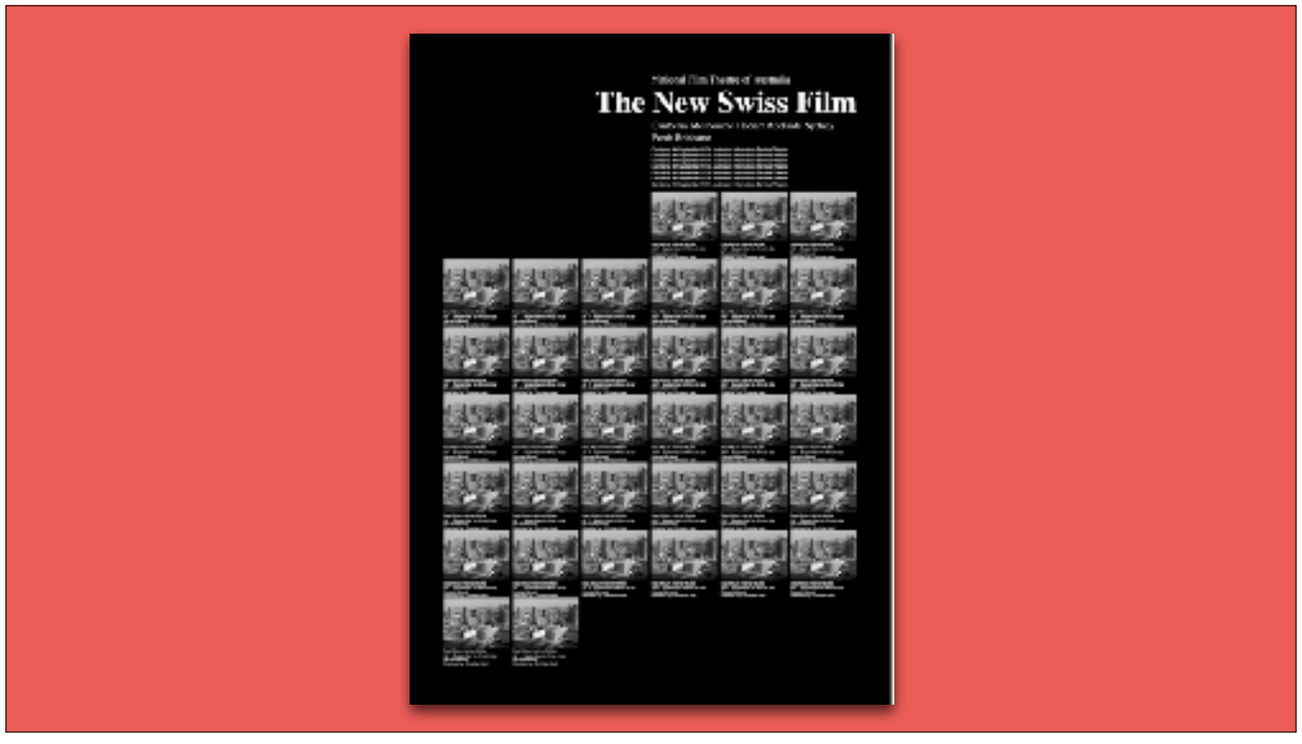


I ask her to find me some cool layouts that she’s keen to use on our next web project and she sketches them out for me, and then I use Grid Layout to do the positioning quickly to see not just how it lays out on desktop, but to then work out how it’s going to behave across all devices.



This was the last one we worked on. It was a poster for the Swiss film festival.

This is slightly more complex than our 4x8 grid, but the same rules apply.



This is what I initially came up with….

And lets look at how they look side by side.

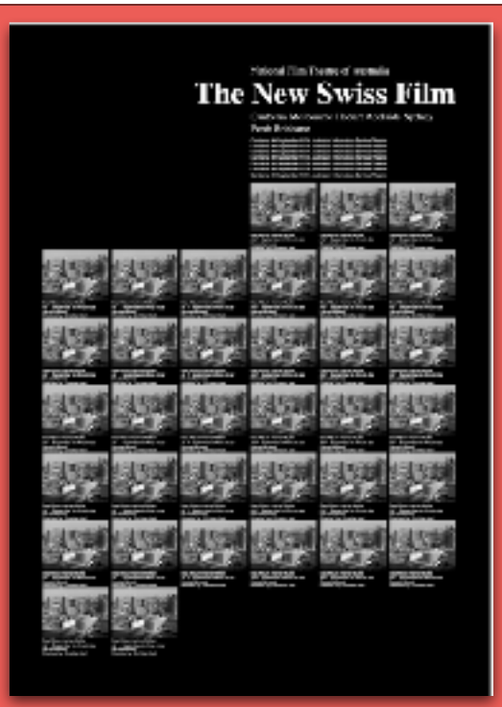
Fonts are slightly out.

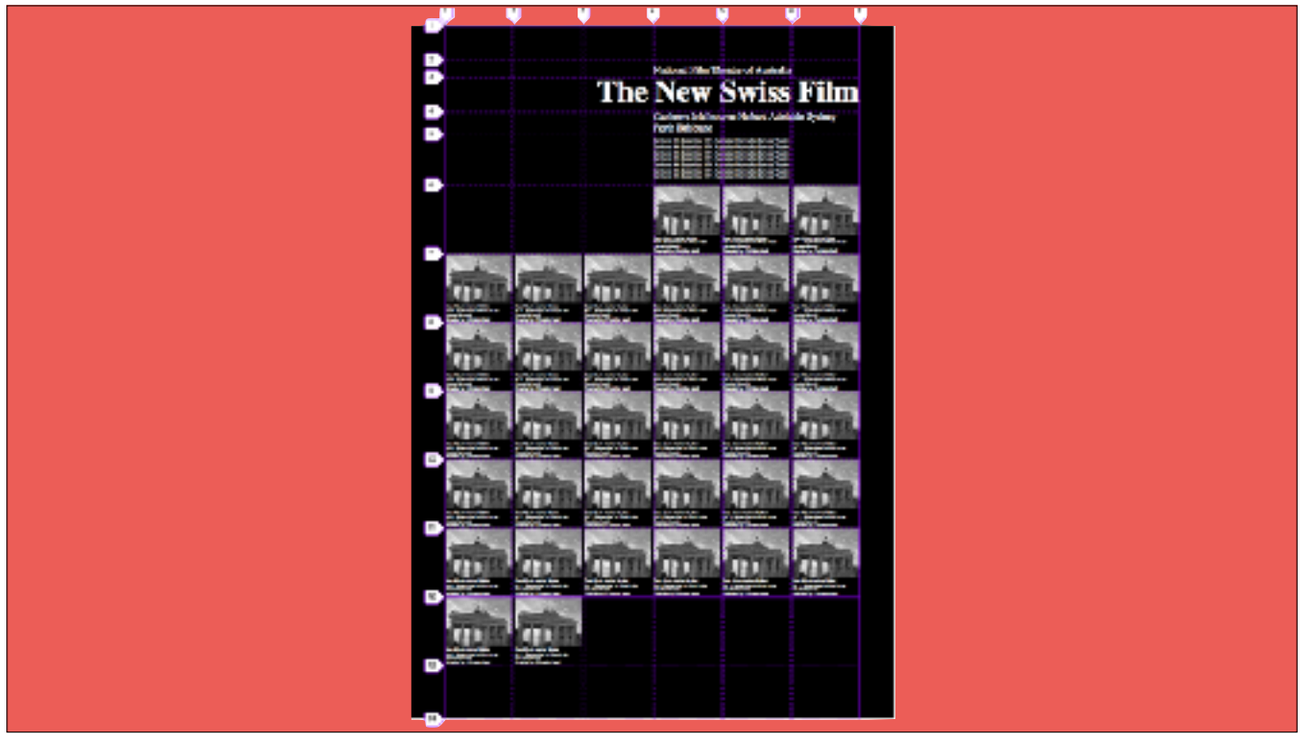
list of events was another poster I looked at, while the one of the right has a paragraph of content.

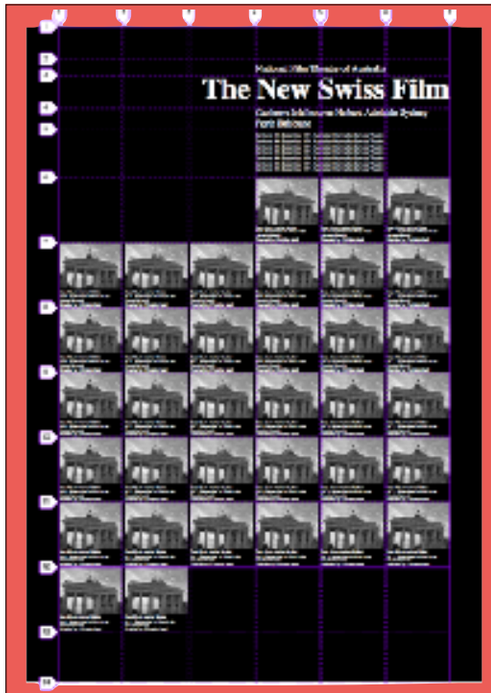
The images I used are a little bit taller, they were from a lorem ipsum style image site.

The descriptions are all the same, so it looks a little more rigid I think.

Overall though, not too far off.


The New Swiss Film


The New Swiss Film



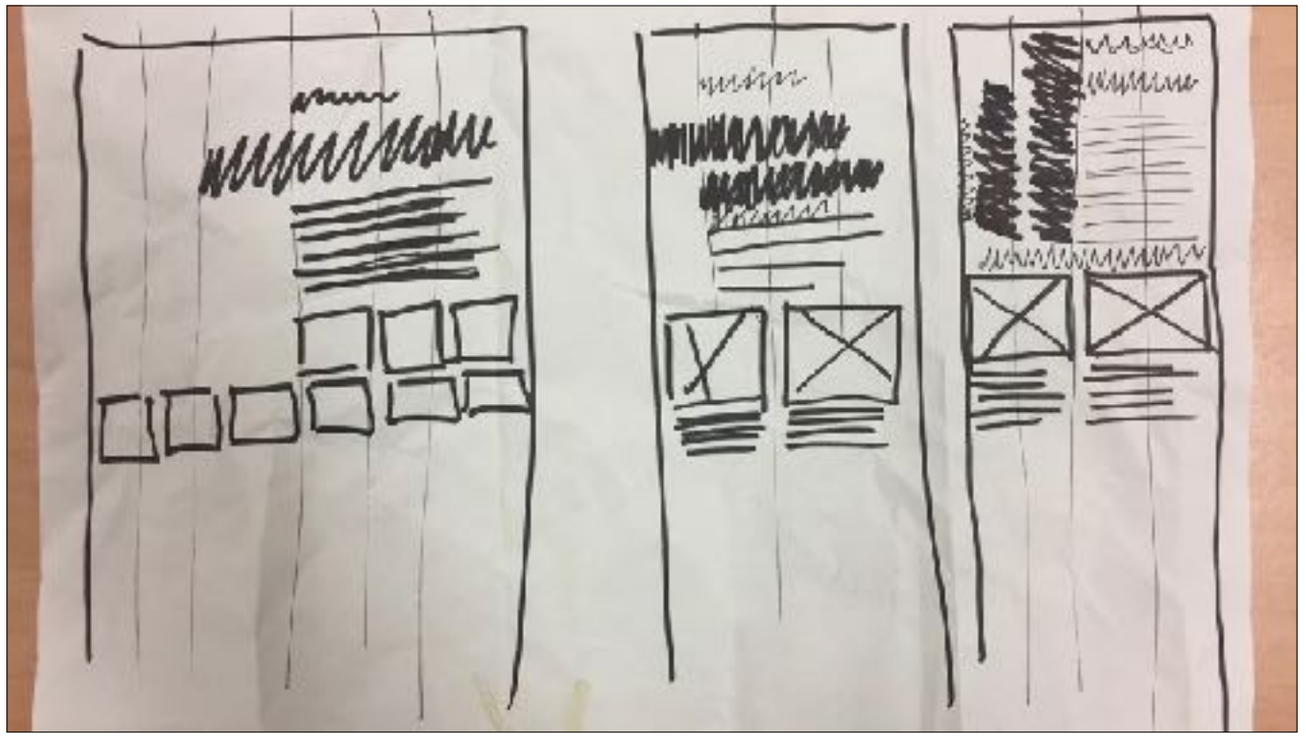

```
.wrapper {
  background: black;
  padding: 0 100px;
  display: grid;
  grid-template-columns: repeat(6, 1fr);
  grid-column-gap: 10px;
  grid-template-rows: minmax(100px, auto)
minmax(50px, auto) minmax(100px, auto)
minmax(50px, auto) minmax(100px, auto)
minmax(50px, auto) minmax(150px, auto)
minmax(150px, auto) minmax(150px, auto)
minmax(150px, auto) minmax(150px, auto) ;
  margin: 50px auto;
  max-width: 1400px;
}
```


```
.wrapper {
  background: black;
  padding: 0 100px;
  display: grid;
  grid-template-columns: repeat(6, 1fr);
  grid-column-gap: 10px;
  grid-template-rows: minmax(100px, auto)
minmax(50px, auto) minmax(100px, auto)
minmax(50px, auto) minmax(100px, auto)
minmax(50px, auto) minmax(150px, auto)
minmax(150px, auto) minmax(150px, auto)
minmax(150px, auto) minmax(150px, auto) ;
  margin: 50px auto;
  max-width: 1400px;
}
```

minmax() sizing…. the minimum height the row can be, but it can expand to auto to fit the content.

Avoids overflow issues.

This was okay, but what about the other viewports?

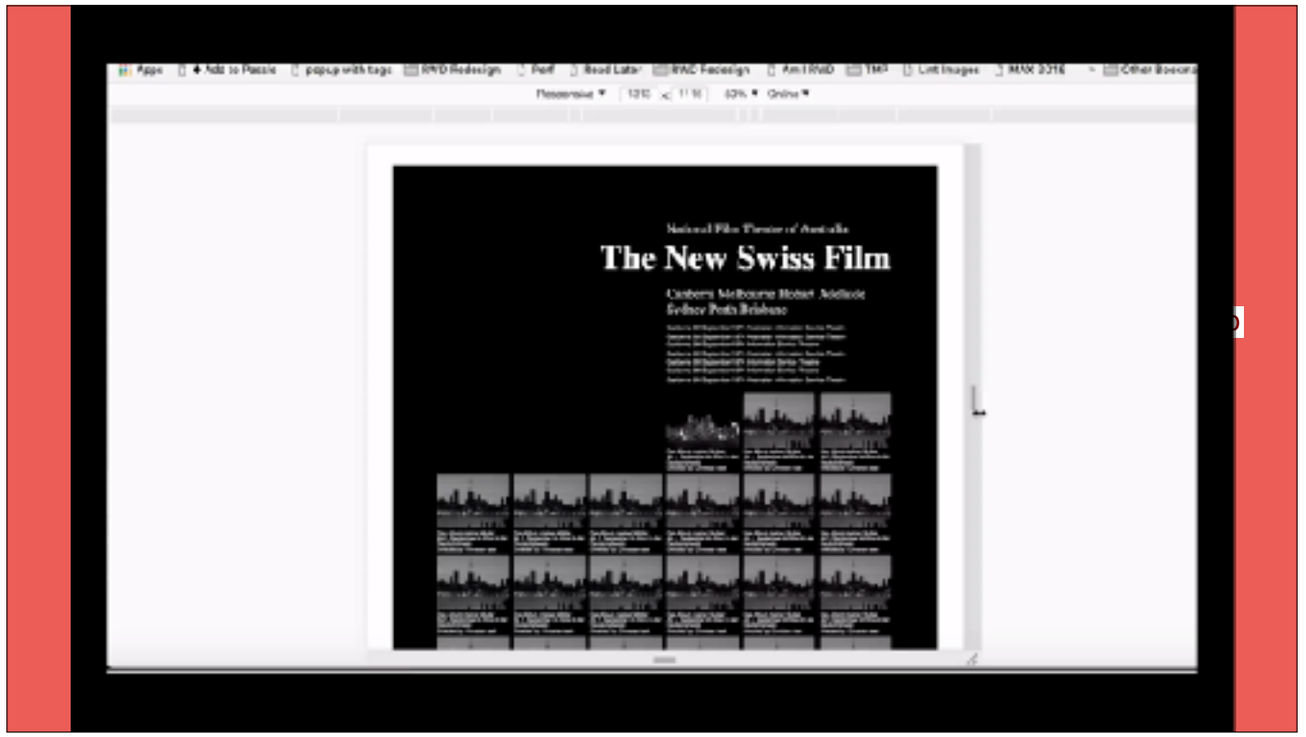


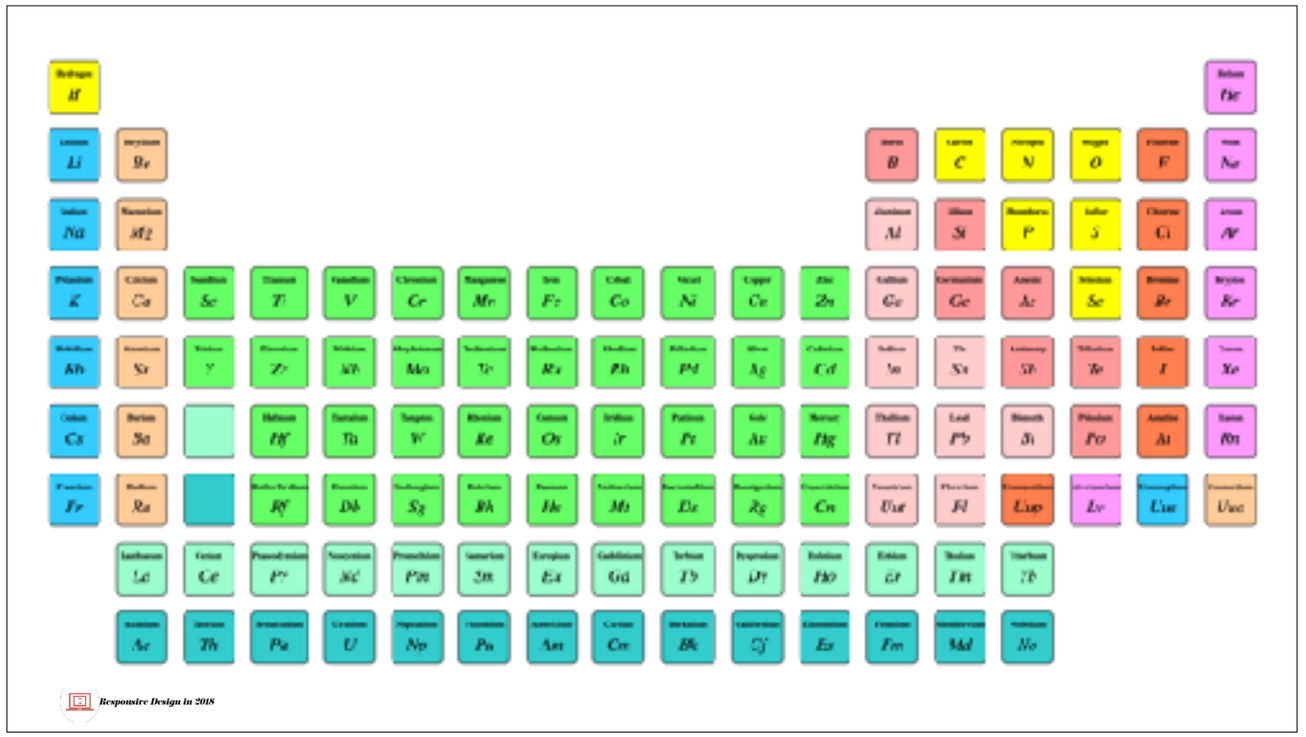
Well sitting down with Raquel she sketched out her idea.

She wondered about the the tablet layout of the left, thinking that it could probably handle the same as the desktop.

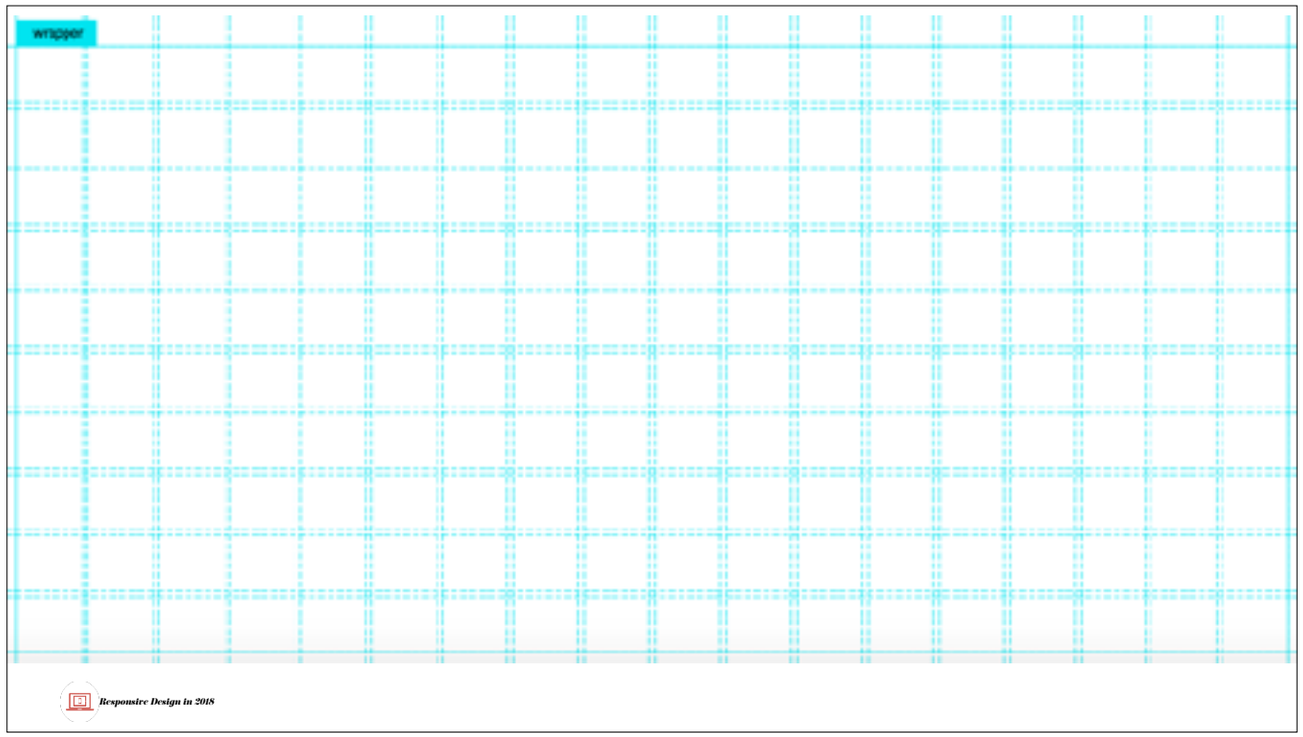
For mobile she wanted the elements to go 2 across, and either offsetting the title by one column to the left still or turning the content around so the text ran vertically instead so I got a chance to explore writing modes.


National Film
Theatre of
Australia
The New
Swiss Film
Canberra 9th September 1974
Australian Information Service
Theatre
Canberra 9th September 1974
Australian Information Service
Theatre
Canberra 9th September 1974
Information Service Theatre
Canberra 9th September 1974
Australian Information Service
Theatre
Canberra 9th September 1974
Information Service Theatre
Canberra 9th September 1974
Information Service Theatre
Canberra 9th September 1974
Australian Information Service
Theatre


The New Swiss Film



And it doesn’t stop there, you can create anything you can think of with CSS Grid.

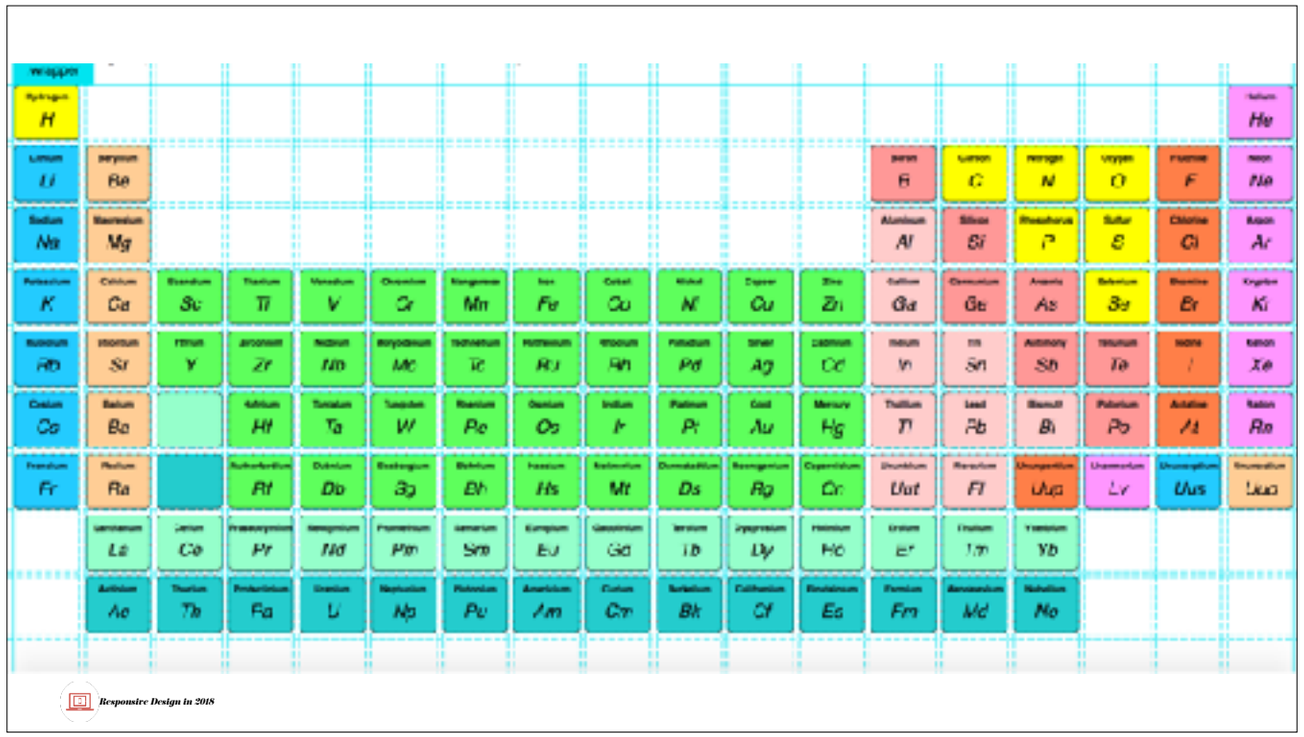


We set the row to auto which is the same as leaving that off, it evenly divides the elements into the available columns.

Instead of letting the row be dictated by source order though we specify the location in the same way that we specified the columns earlier.

```
.h  {grid-column:1;  grid-row:1;}
.he {grid-column:18; grid-row:1;}
.li {grid-column:1;  grid-row:2;}
.be {grid-column:2;  grid-row:2;}
.b  {grid-column:13; grid-row:2;}
.c  {grid-column:14; grid-row:2;}
.n  {grid-column:15; grid-row:2;}
.o  {grid-column:16; grid-row:2;}
```


Then I set each of those in their position on the grid.
You can see here there’s a bit of short hand as well. because each grid item is only spanning a single track I don’t need to give it a start and end.


Responsive Design in 2018


GRID GARDEN

Grid or Flex

Well it’s really up to you.

Last year I said Grid was for known layouts and flex was for unknown layouts but that’s not true. You can layout using flex when it’s known, and you can use Grid to layout unknown items.

I think it’s best to think of it as….

2 Dimensions vs 1 Dimension

Grid vs Flex is simply 2 dimensional layouts vs 1. And they can work together.

To get a bit more experience on Grid you can play a similar game to froggy.

Rachel Andrews	https://gridbyexample.com
Jen Simmons	http://labs.jensimmons.com/
Melanie Richards	https://summit.microsoftedge.com/
Jen Kramer	http://jenkramer.org/

Grid vs Flex is simply 2 dimensional layouts vs 1. And they can work together.

To get a bit more experience on Grid you can play a similar game to froggy.

~~CSS Grid Layout~~	Faster Websites
Offline / PWAs	Cool Sh*tuff

Faster Websites

None of us set out to build slow websites, but what is the impact of building a slow website? Or better yet, what is the impact on our customers when we make our current website faster?

Amazon sees a 1% decrease in revenue for every 100ms increase in load time.

http://radar.oreilly.com/2008/08/radar-theme-web-ops.html

Amazon sees a 1% decrease in revenue for every 100ms increase in load time.

Daily Revenue: $93,709,589.04

For every 100ms delay it costs them **$937,095.89**

Netflix saw a 43% decrease in their bandwidth bill after turning on GZip.

http://cdn.oreillystatic.com/en/assets/1/event/7/Improving%20Netflix%20Performance%20Presentation.pdf

Netflix saw a 43% decrease in their bandwidth bill after turning on GZip.

Watching Netflix uses about 1 GB of data per hour for each stream of standard definition video, and up to 3 GB per hour for each stream of HD video.

The Trainline reduced latency by 0.3 seconds across their funnel and customers spent an extra £8 million (~$11.5 million) a year.

https://youtu.be/ai-6qwT6ES8?t=462

The Trainline, **a UK based company**, reduced latency by 0.3 seconds across their funnel and customers spent an extra £8 million (~$11.5 million) a year.

Tests of the new, faster FT.com showed users were up to 30% more engaged—meaning more visits and more content being consumed.

https://www.wsj.com/articles/financial-times-hopes-speedy-new-website-will-boost-subscribers-1475553602

Tests of the new, faster FT.com showed users were up to 30% more engaged—meaning more visits and more content being consumed.

All of those stats come from WPO stats. You can get a tonne of different statistics that will help back up your business case about why you ***need*** to spend time and effort making your site more performant. Each statistic is linked to research or an article backing the claim as well.

- Ads
- Conversions
- Revenue
- Expenses
- Traffic
- Bounces
- 2006-2017

But aside from these reasons there’s also another reason to build faster sites.

The next billion users

At this conference a year ago Ethan Marcotte stood on stage and talked to you about creating sites for the next billion users.

Last year one of the creative speakers that talk on the Thursday morning was Zach Posen and he talked about reaching billions of people around the world with his designs.

Well, ***you*** have that capability. As content editors, web designers, art directors, web developers, product owners…. as ***people(individuals and teams) who craft web experiences,*** you all have the power to reach billions of people with your work.

WORLD INTERNET USAGE AND POPULATION STATISTICS
JUNE 30, 2016 - Update

World Regions	Population (2016 Est.)	Population % of World	Internet Users 30 June 2016	Penetration Rate (% Pop.)	Growth 2000-2016	Table % Users
Asia	4,052,652,889	55.2 %	1,846,212,654	45.6 %	1,515.2%	50.2 %
Europe	832,073,224	11.3 %	614,979,903	73.9 %	485.2%	16.7 %
Latin America / Caribbean	626,054,392	8.5 %	384,751,302	61.5 %	2,029.4%	10.5 %
Africa	1,185,529,578	16.2 %	340,783,342	28.7 %	7,448.8%	9.3 %
North America	359,492,293	4.9 %	320,067,193	89.0 %	196.1%	8.7 %
Middle East	246,700,900	3.4 %	141,489,765	57.4 %	4,207.4%	3.8 %
Oceania / Australia	37,590,820	0.5 %	27,640,564	73.5 %	281.4%	0.8 %
WORLD TOTAL	7,340,094,096	100.0 %	3,675,624,813	50.1 %	918.3%	100.0 %

NOTES. (1) Internet Usage and World Population Statistics updated as of June 30, 2016. (2) CLICK on each world region name for detailed regional usage information. (3) Demographic (Population) numbers are based on data from the US Census Bureau, Eurostats and from local census agencies. (4) Internet usage information comes from data published by Nielsen Online, by the International Telecommunications Union, by GfK, by local ICT Regulators and other reliable sources. (5) For definitions, disclaimers, navigation help and methodology, please refer to the Site Surfing Guide. (6) Information in this site may be cited, giving the due credit and placing a link to www.internetworldstats.com.

Last year we tipped the balance at 50.1% of the worlds population that was online.

World Internet Usage and Population Statistics June 30, 2017

World Regions	Population (2017 Est.)	Population % of World	Internet Users 30 June 2017	Penetration Rate (% Pop.)	Growth 2000-2017	Internet Users %
Africa	1,246,504,865	16.6%	**388,376,491**	31.2%	8,503.1%	10.0%
Asia	4,148,177,672	55.2%	**1,938,075,631**	46.7%	1,595.5%	49.7%
Europe	822,710,362	10.9%	**659,634,487**	80.2%	527.6%	17.0%
Latin America / Caribbean	647,604,645	8.6%	**404,269,163**	62.4%	2,137.4%	10.4%
Middle East	250,327,574	3.3%	**146,972,123**	58.7%	4,374.3%	3.8%
North America	363,224,006	4.8%	**320,059,368**	88.1%	196.1%	8.2%
Oceania / Australia	40,479,846	0.5%	**28,180,356**	69.6%	269.8%	0.7%
WORLD TOTAL	**7,519,028,970**	**100.0%**	**3,885,567,619**	**51.7%**	**976.4%**	**100.0%**

Responsive Design in 2018

+1.6% **+209,742,806**

This year we've increased by another **1.6%** to **51.7%,** which has added a more than **200 million people** that we can reach.

“Our” Users
“Our” Target Market

We talk a lot about “*our*” users and “*our*” target market, but when you look at it from a global stand point the number of people that we can reach across the UK, Australia or the US are dwarfed in comparison with the number of users that are accessing our sites across the rest of the world.

Anecdote about the OUR target market with a client.

This is one of my favourite maps. If you haven’t seen it before it shows a white circle, and there are more people living within that circle than are living outside of it. These are the next billion people that will be using the web and accessing the websites that we craft for them today.

http://brilliantmaps.com/population-circle/

World Internet Usage and Population Statistics June 30, 2017

World Regions	Population (2017 Est.)	Population % of World	Internet Users 30 June 2017	Penetration Rate (% Pop.)	Growth 2000-2017	Internet Users %
Africa	1,246,504,865	16.6%	**388,376,491**	31.2%	8,503.1%	10.0%
Asia	4,148,177,672	55.2%	**1,938,075,631**	46.7%	1,595.5%	49.7%
Europe	822,710,362	10.9%	**659,634,487**	80.2%	527.6%	17.0%
Latin America / Caribbean	647,604,645	8.6%	**404,269,163**	62.4%	2,137.4%	10.4%
Middle East	250,327,574	3.3%	**146,972,123**	58.7%	4,374.3%	3.8%
North America	363,224,006	4.8%	**320,059,368**	88.1%	196.1%	8.2%
Oceania / Australia	40,479,846	0.5%	**28,180,356**	69.6%	269.8%	0.7%
WORLD TOTAL	**7,519,028,970**	**100.0%**	**3,885,567,619**	**51.7%**	**976.4%**	**100.0%**

Responsive Design in 2018

If we look at **Africa** and **Asia** and the number of current internet users it makes up almost **60%** of **all users on the internet**.

If you think what we would consider to be “**our**” target market in our respective countries…. the **combination** of the **USA** and **Australia** don’t even reach **9%**.

World Internet Usage and Population Statistics June 30, 2017

World Regions	Population (2017 Est.)	Population % of World	Internet Users 30 June 2017	Penetration Rate (% Pop.)	Growth 2000-2017	Internet Users %
Africa	1,246,504,865	16.6%	**388,376,491**	31.2%	8,503.1%	10.0%
Asia	4,148,177,672	55.2%	**1,938,075,631**	46.7%	1,595.5%	49.7%
Europe	822,710,362	10.9%	**659,634,487**	80.2%	527.6%	17.0%
Latin America / Caribbean	647,604,645	8.6%	**404,269,163**	62.4%	2,137.4%	10.4%
Middle East	250,327,574	3.3%	**146,972,123**	58.7%	4,374.3%	3.8%
North America	363,224,006	4.8%	**320,059,368**	88.1%	196.1%	8.2%
Oceania / Australia	40,479,846	0.5%	**28,180,356**	69.6%	269.8%	0.7%
WORLD TOTAL	**7,519,028,970**	**100.0%**	**3,885,567,619**	**51.7%**	**976.4%**	**100.0%**

Responsive Design in 2018

If you look at how many people are coming online…. Between **Africa** and **Asia** they make up **more than 71%** of the worlds population.

This is where the next billion will be coming from.

And the next billion users have completely different challenges from us.

Speaking broadly here, when most of us in the room access the internet on our phones we do it out of convenience.

Emerging markets use mobile phones to access the web out of necessity.

Power is massive problem for emerging markets. There are many charging stations like this one, ones plugged into generators, and a number of solar powered stations.

They also have the same issue around the cost of access to the web. They’re not on Gigabyte plans, but megabyte plans and it can be incredibly expensive.

In fact, if we ignore emerging markets for a second, there are a large number of US and UK families who’s only access to the web is via mobile devices as well, and they share the issue that the cost for data can be expensive.

“Our” Users
“Our” Target Market

If I go back to ‘our’.

Disney want to **sell** products.
Universities want to generate **student fees**.

Disney also wants to make **children happy**.
Universities want to make the **world** a **more educated** place - **research** etc.

If we build fast performant sites not only do we cater for the next billion users, but if we go back to **‘OUR’** target market then they have far better experiences and from WPO Stats we can see how that translates to improved engagement.

No one sets out to build a slow site that costs a fortune to see

I know that none of us set out to build the slowest and most expensive to see website, but unfortunately through twists and turns of a project and stakeholders we can sometimes end up there.

For this section, and **putting philanthropic Justin to one side,** I want to **step** through a **site** that we should all be **pretty familiar** with… it's one site I can guarantee you've all visited.

How many people visited this site?

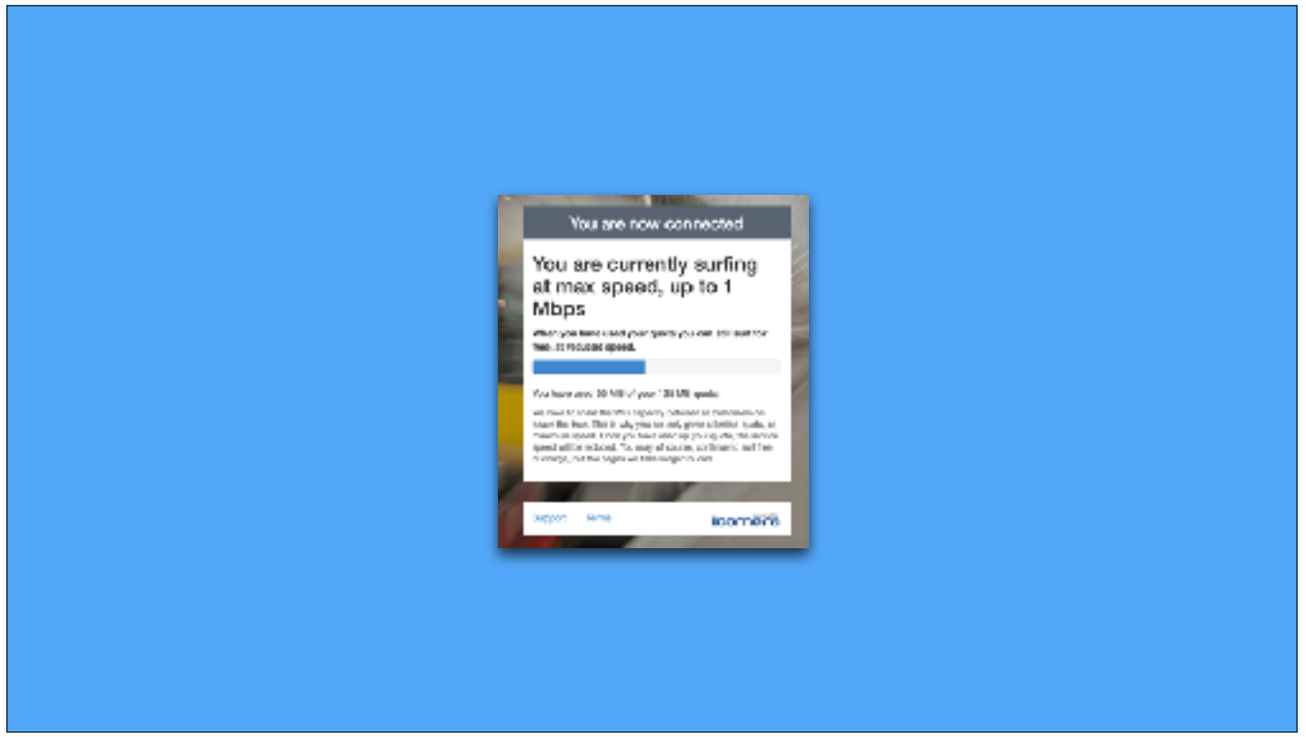
If you can’t remember it, lets take a closer look at it.

I took this site and saved it down locally.

I created a new site for it at adobemax.simplethin.gs
For 1 week on my morning train commute of about 45 minutes I would tinker.

This is my version.

Did you notice any changes? You should have noticed a couple but we'll get into those in a little bit.



I’m extremely lucky because I get 125mb every morning on my commute to work, but based on the size of the Adobe Max homepage I can only visit it 5 times before I’ve run out of data.

$1.25 per day is the poverty line, **5.5 million** people in the US are on this.

$2.05 **USA**
$1.92 **Switzerland**
$1.84 **Ireland**

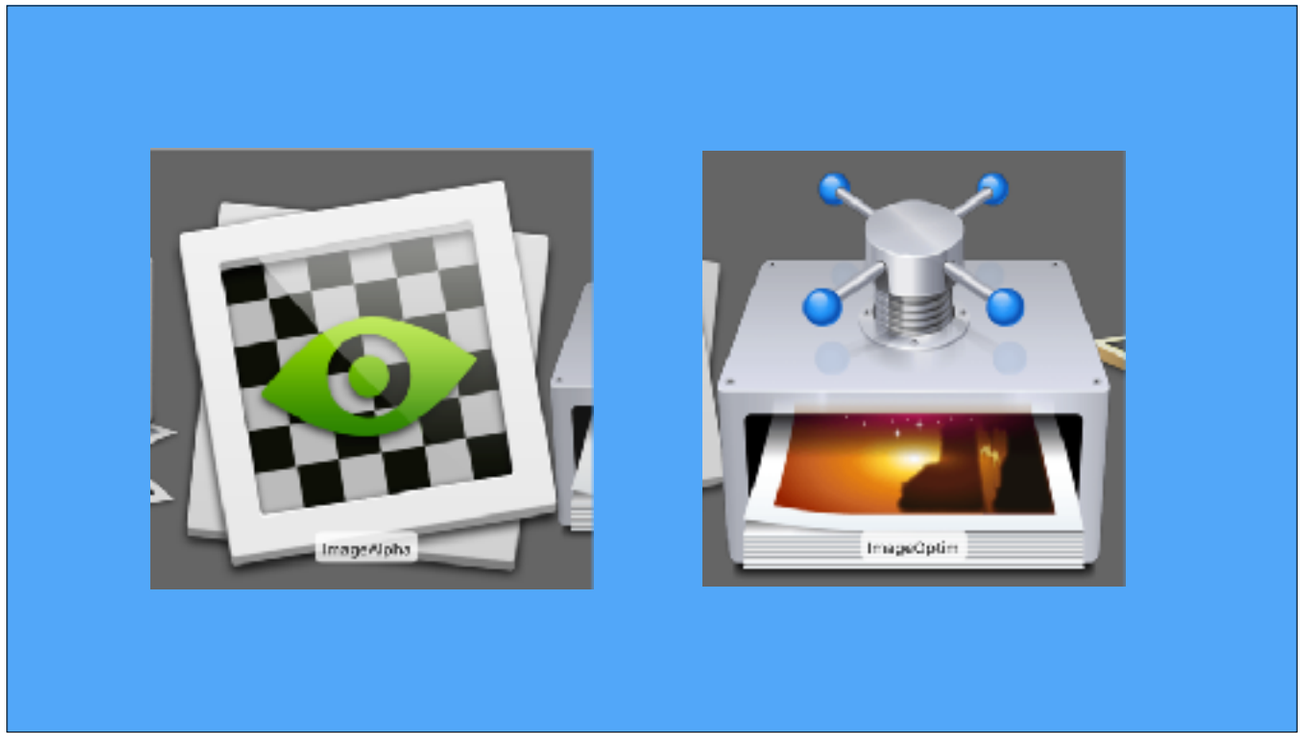
The GNI per capita is the dollar value of a country’s final income in a year, divided by its population. It reflects the **average income of a country’s citizens**.

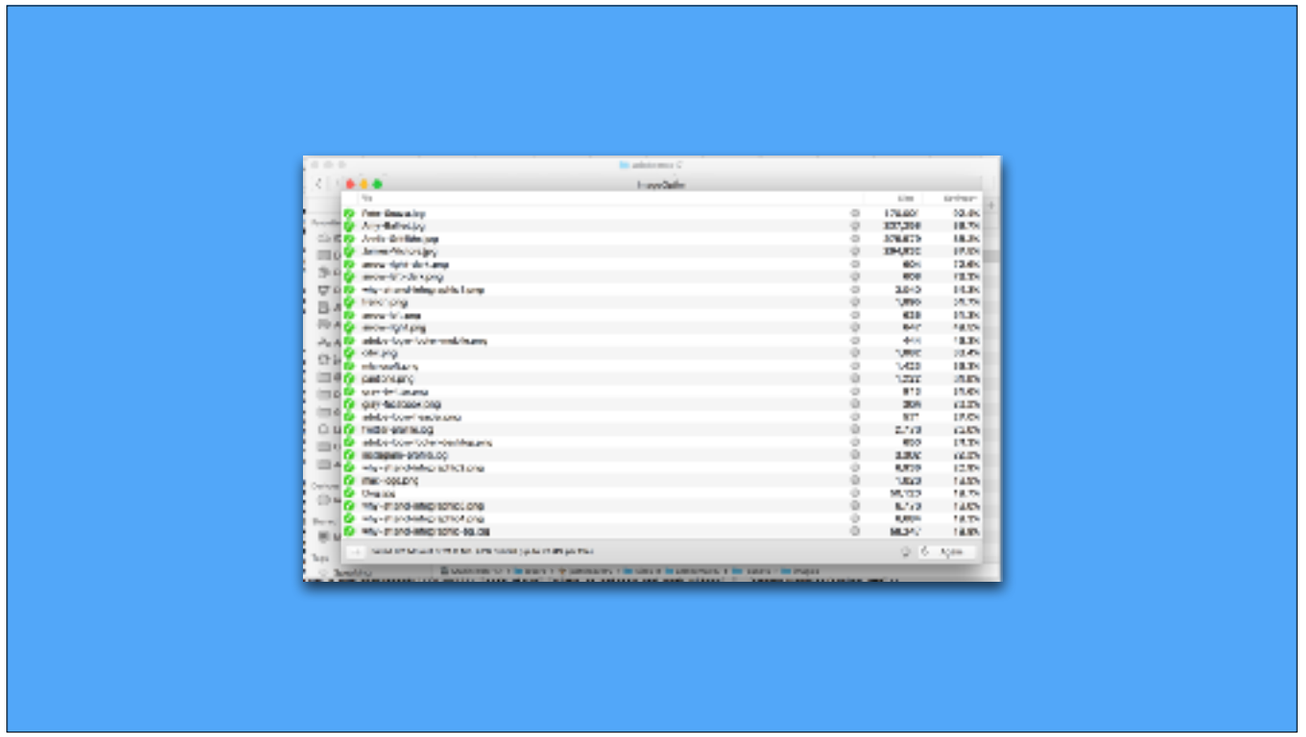
15% of the average income of someone in Vanuatu to visit the adobe web site.?

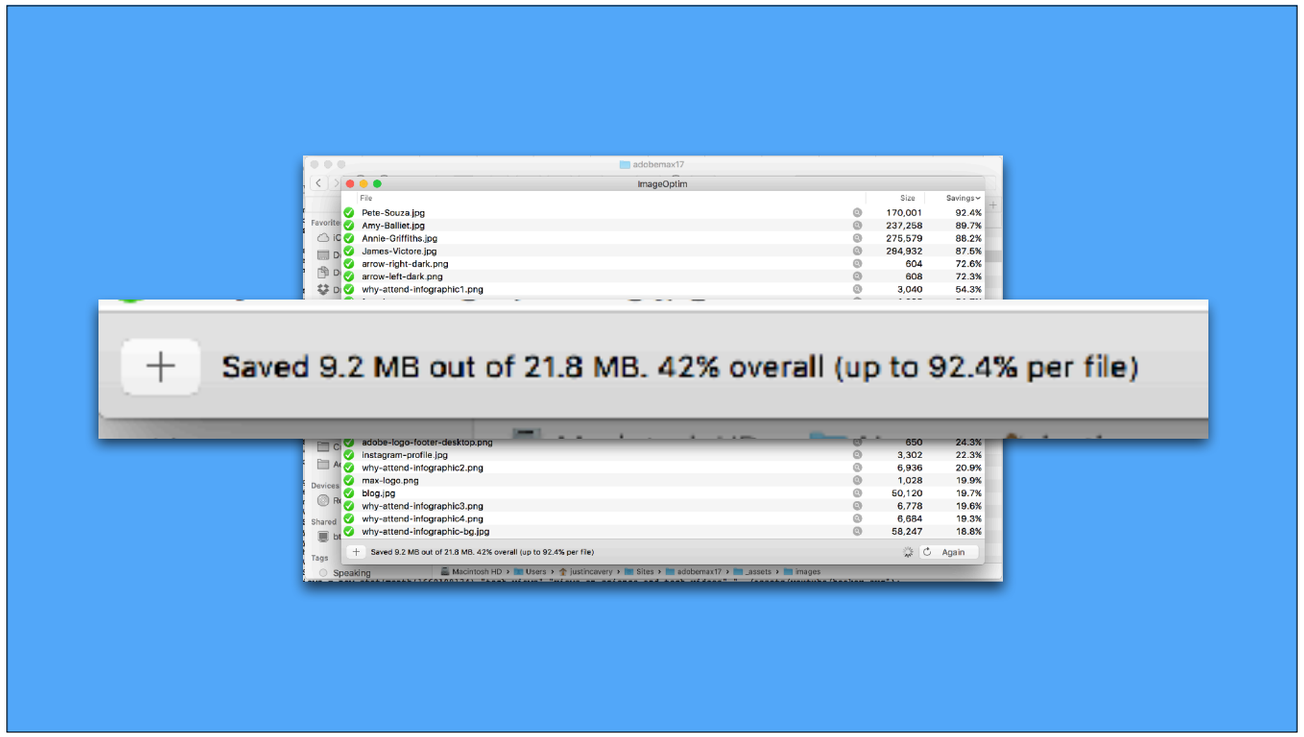
Step 1

Optimise the images

Let’s look at some really basic approaches


ImageAlpha
ImageOptim





9.2MB - 42% overall savings

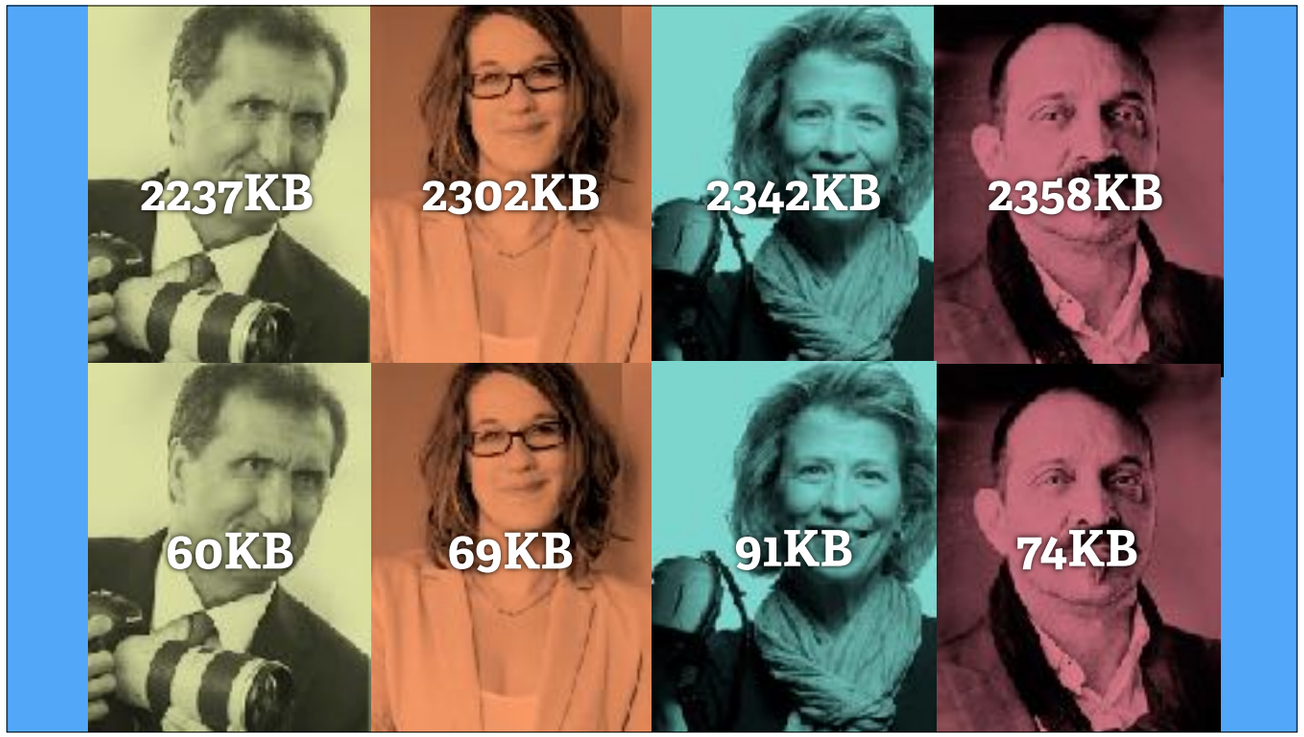
Step 2

Resize the images

Who can tell the difference between these two images?

That’s right, you can’t. The difference was that the one of the left was the image being served, the one of the right is the one I’ve dropped in.

Linsey image also went from 2000px to 800px saving



More than **9MB** on the homepage with these four images.

Replaced with **300kb**.

https://images.guide/

Addy Osmani has written an online resource at images.guide for you to check out.
You might scratch your head and wonder why I’m banging on about responsive images and image weight in a title looking to 2018, but the truth of the matter is that we haven’t fixed the problem.

Fastly

Cloundinary

ImgIX

ImageEngine

Step 3

Apply GZIP & Browser caching

Apply GZIP & Browser caching

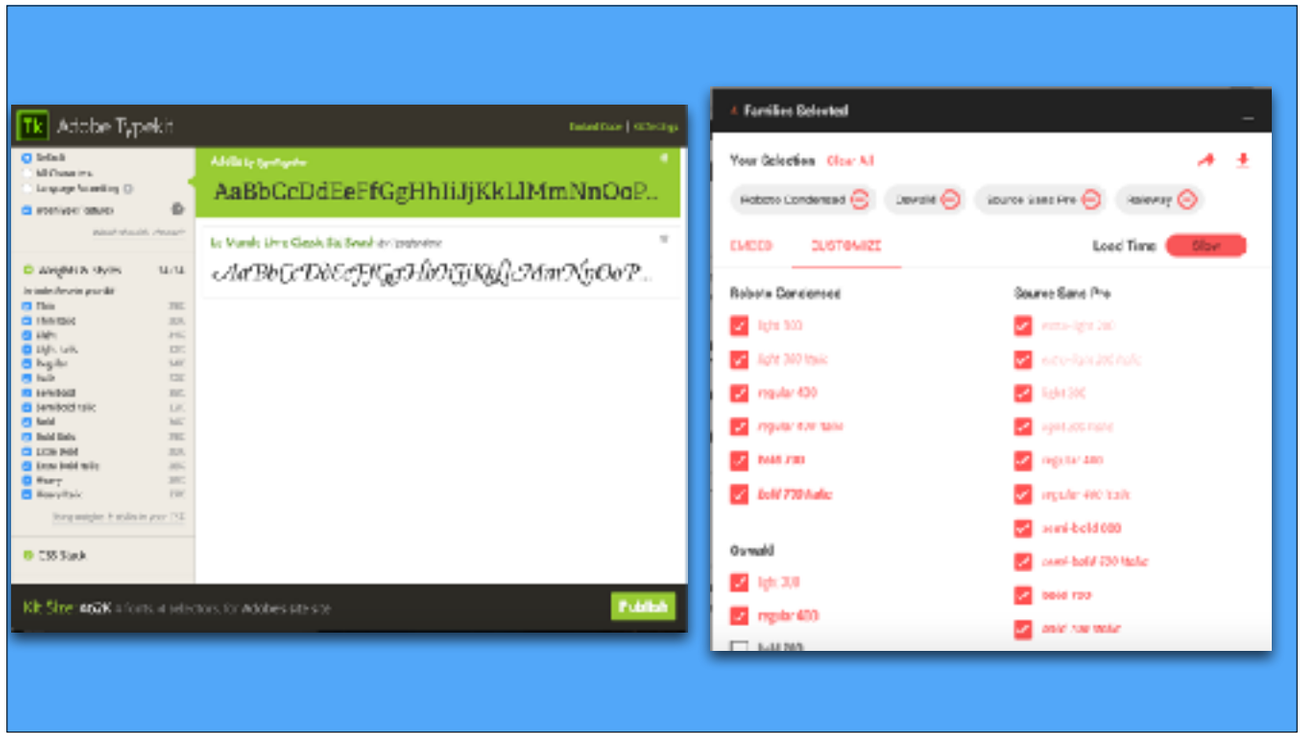
HTML5….

https://html5boilerplate.com/

The **HTML5 Boilerplate v 6.0** has a wonderful **.htaccess** file that provides you with out of the box best practice gzip and caching options.

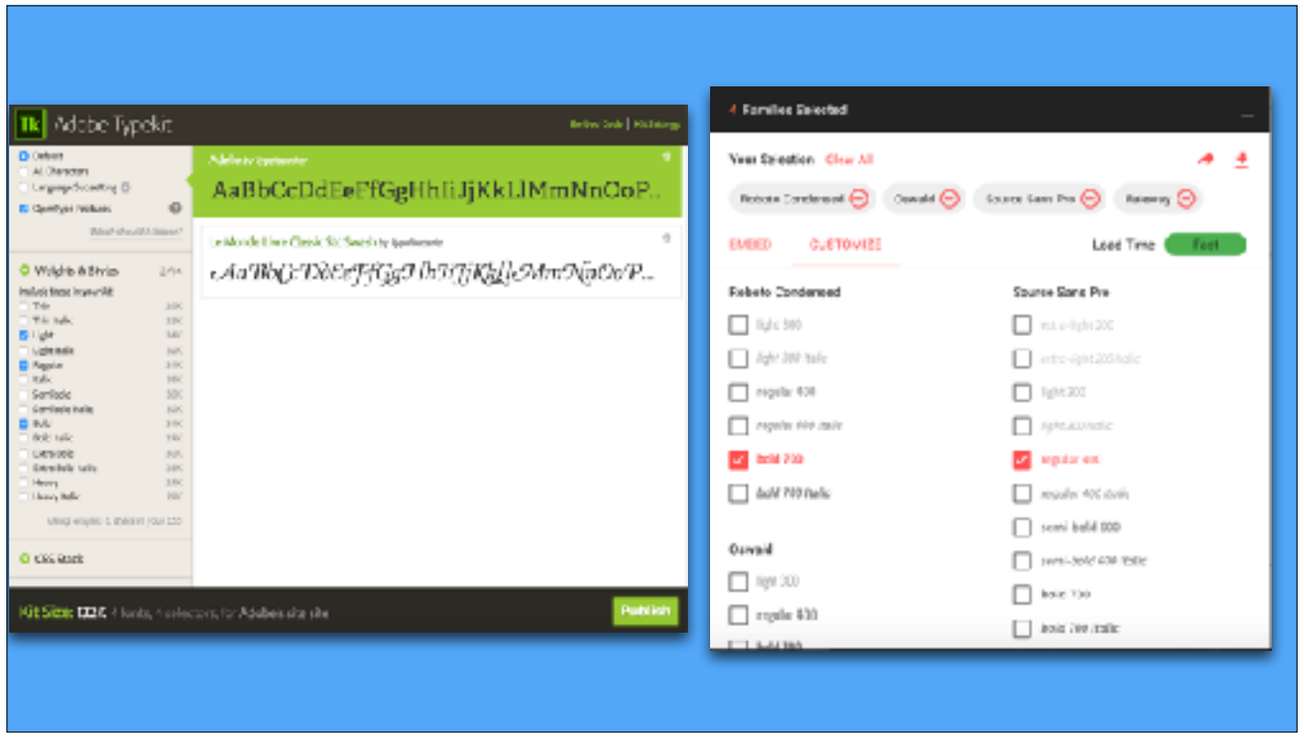
Step 4

Optimise Typography



The designers I work with tend to want all the font with all the font weights.

Even if it’s only used once. POINT OUT THE SIZE AND SLOW.



I encourage, or demand, that the designers use type as a constraint rather than a free for all.

I usually set a limit of 150kb worth of fonts.

```
body {
    font-family: -apple-system,
                 BlinkMacSystemFont,
                 "Segoe UI",
                 Roboto,
                 Oxygen-Sans,
                 Ubuntu,
                 Cantarell,
                 "Helvetica Neue",
                 sans-serif;
}
```

In fact why not do away with web fonts all together.

In a world where everyone is using the same 10 web font pairings are we really providing much difference to our websites?

I'm not saying outright not to use them, but I would question whether their use is justified on the project that you're currently working on. I'm eating my own dog food here because I'm about to push live an update to RWD.is which takes this approach.

Systems Fonts Saved 574KB

For the Adobe site removing the web fonts dropped the overall page weight by 574KB…. so about half a megabyte.

It also helps get our content rendered faster because we don’t need to download the fonts to be able to display the content as well.

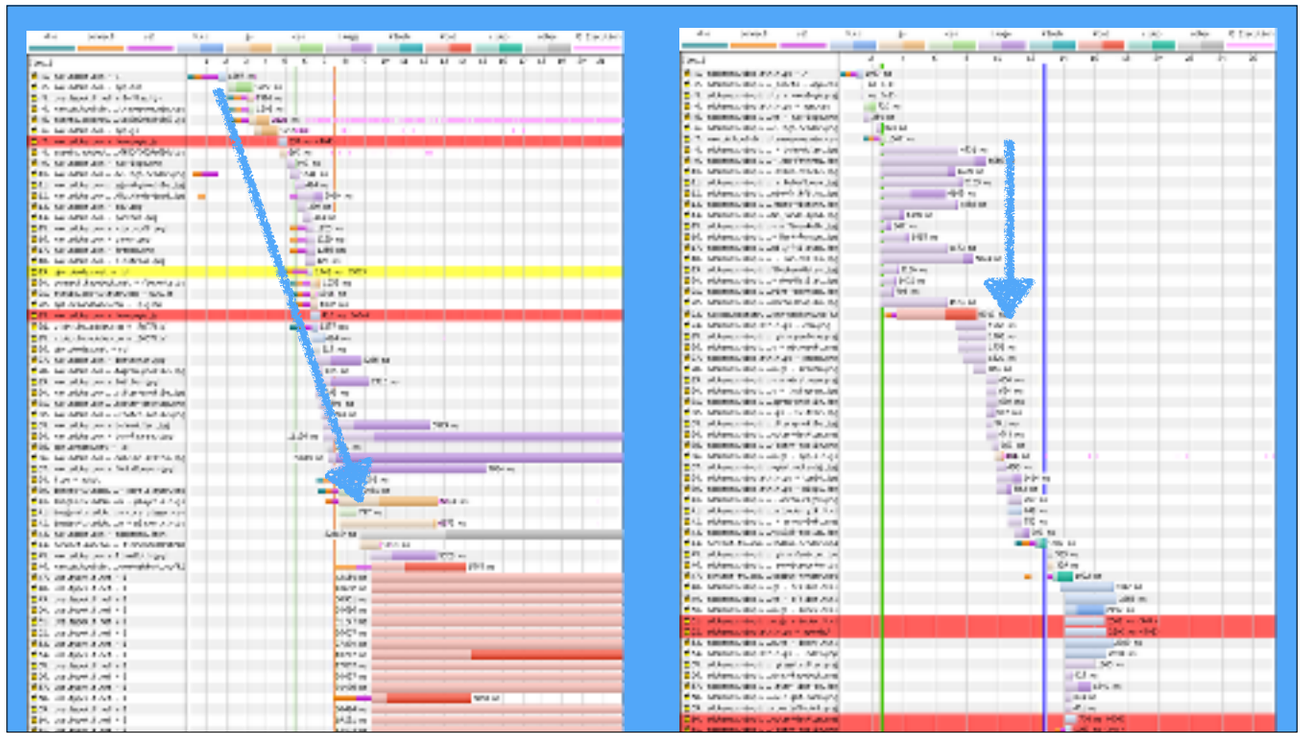
Users that are used to their system fonts are also used to the ones that will be shown, so your site will be windows on windows, Mac on Mac, Android on android etc.

Lets have a closer look at the difference between the two.

For me there is a visible change and I actually prefer the roundness of the webfoot in this case…. but because we're talking about performance here the 500kb was not worth the tradeoff for me.

This is most certainly coming at you in 2017 and it may have a significant affect on the way in which we're building our websites today…. but that's okay.

In fact if you're running on a secure connection HTTPS, and
They haven't said you have to run https for it, but currently no-one is supporting it unless you are running HTTPS and that doesn't look as though it will change.



Step 6

HTTP2 PUSH

Mars Journey.

How traditional web page requests work.

This site “If the moon was 1 pixel”

182 seconds

People are going to be living on Mars and need to build a civilisation from scratch. They will need to build refineries to mine the planet to be able to produce fuel to get back to Earh, but also to power the other things they need to build. They need to build greenhouses and start growing food, they going to need to terraform the planet to make it more hospitable for the future generations to visit or grow up there.

When ever I need help with building something there is only one place that I turn to and that’s the Internet. That might be a good reason not to send me as one of the early first parties, but I think you get what I mean. They cant take a box of the internet with them because it will have changed by the time they arrive there. And they’re going to want to add to it while they’re there…. How is snap chat going to work from Mars? What about Instagram? How are the first colonists going to Tweet?

Okay so this might be thinking a little too far into the future…. But it’s the type of thing that from now on I want you to consider when you are building your next website, or when you’re making redesign changes to your current site, or how you’re going to approach your website updates when you go back to work on Monday.

Facebook have this practice called 2G Tuesdays, where an office will slow their internet connection down to 2G so they can seen how half of their audience experience their product.

Forget 2G Tuesdays…. I want you to start thinking about Mars Mondays.

HTTP2 PUSH

```
header("Link: <_assets/app.css>; rel=preload;
as=style", false);
header("Link: <_assets/images/global/max-
logo.png>; rel=preload; as=image", false);
echo '<div id="headers-sent"></div>';
```

The false at the end of each line tells it not to override an existing Link header. If that was omitted then the last one would be the only one sent.

I’m also including an ‘echo’ on here so I could see in the HTML easily that this was working.

HTTP2 PUSH

```
header("Link: <_assets/app.css>; rel=preload;
as=style", false);
header("Link: <_assets/images/global/max-
logo.png>; rel=preload; as=image", false);
echo '<div id="headers-sent"></div>';
```

The false at the end of each line tells it not to override an existing Link header. If that was omitted then the last one would be the only one sent.

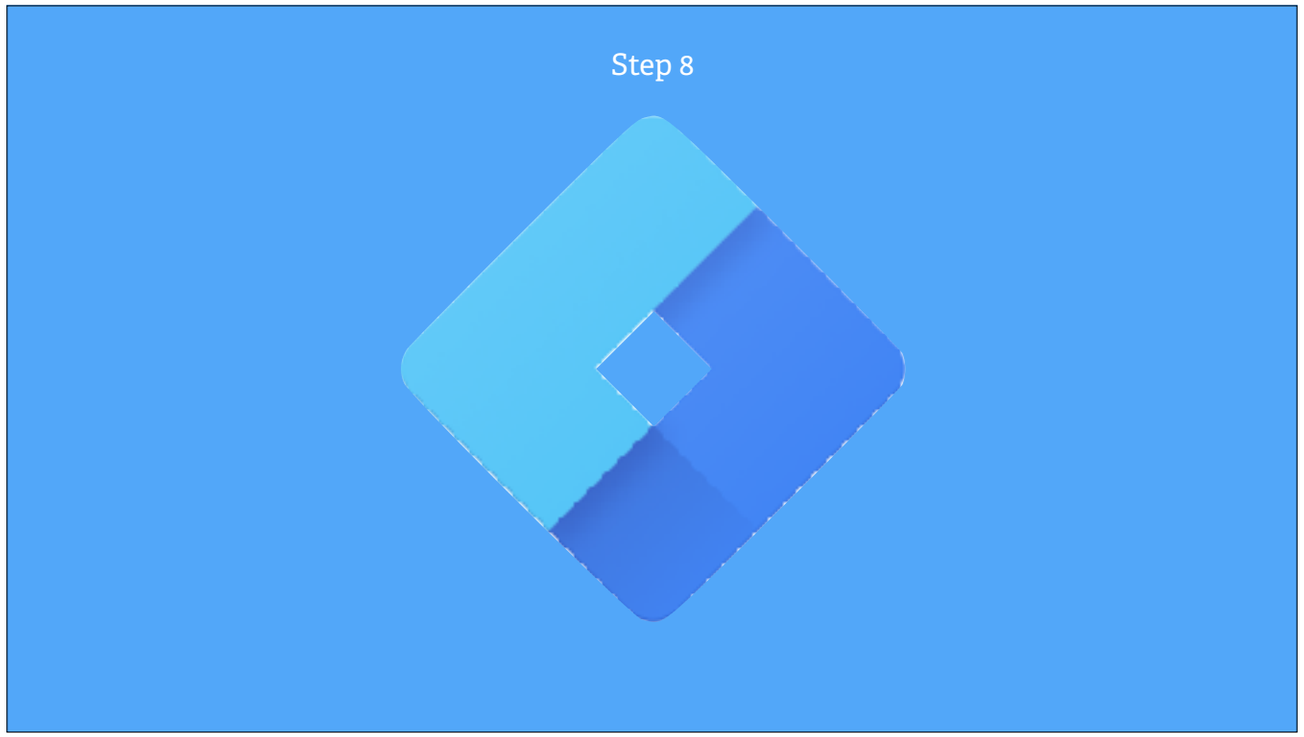
I’m also including an ‘echo’ on here so I could see in the HTML easily that this was working.

Step 7

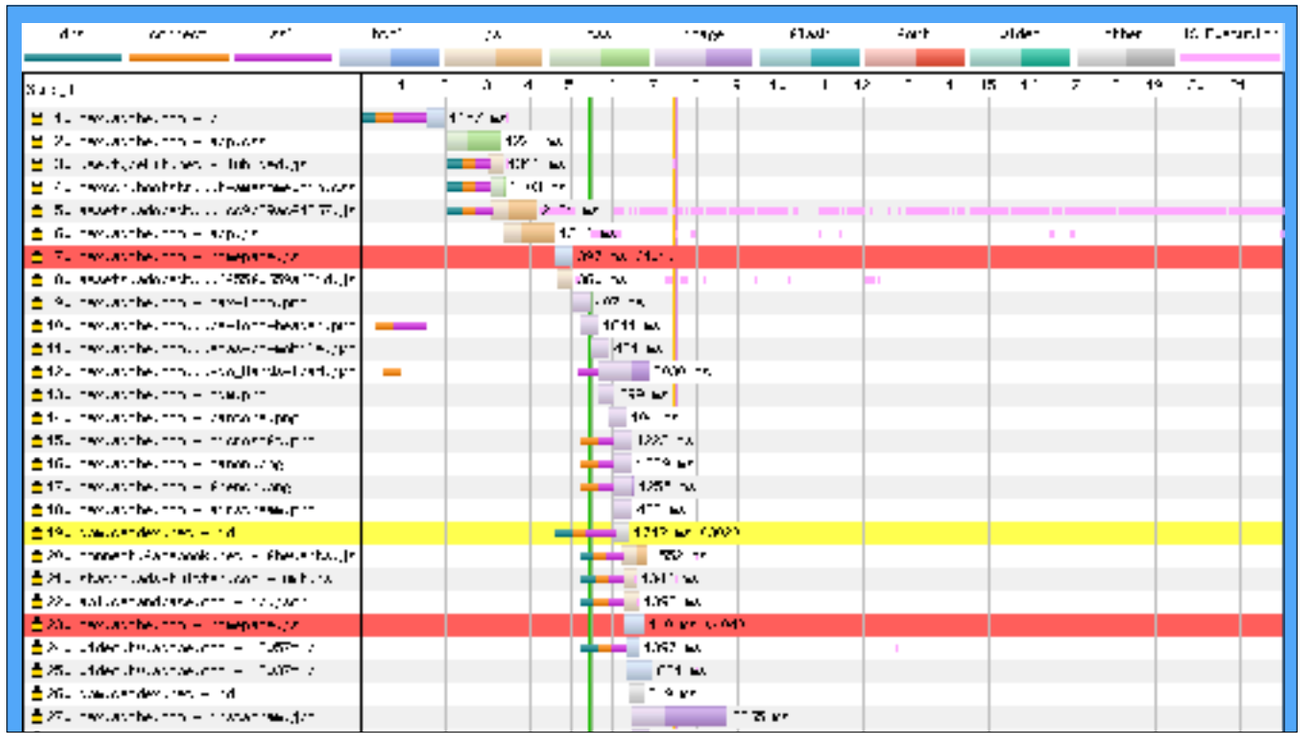
What is a video worth?

Responsive Design in 2018

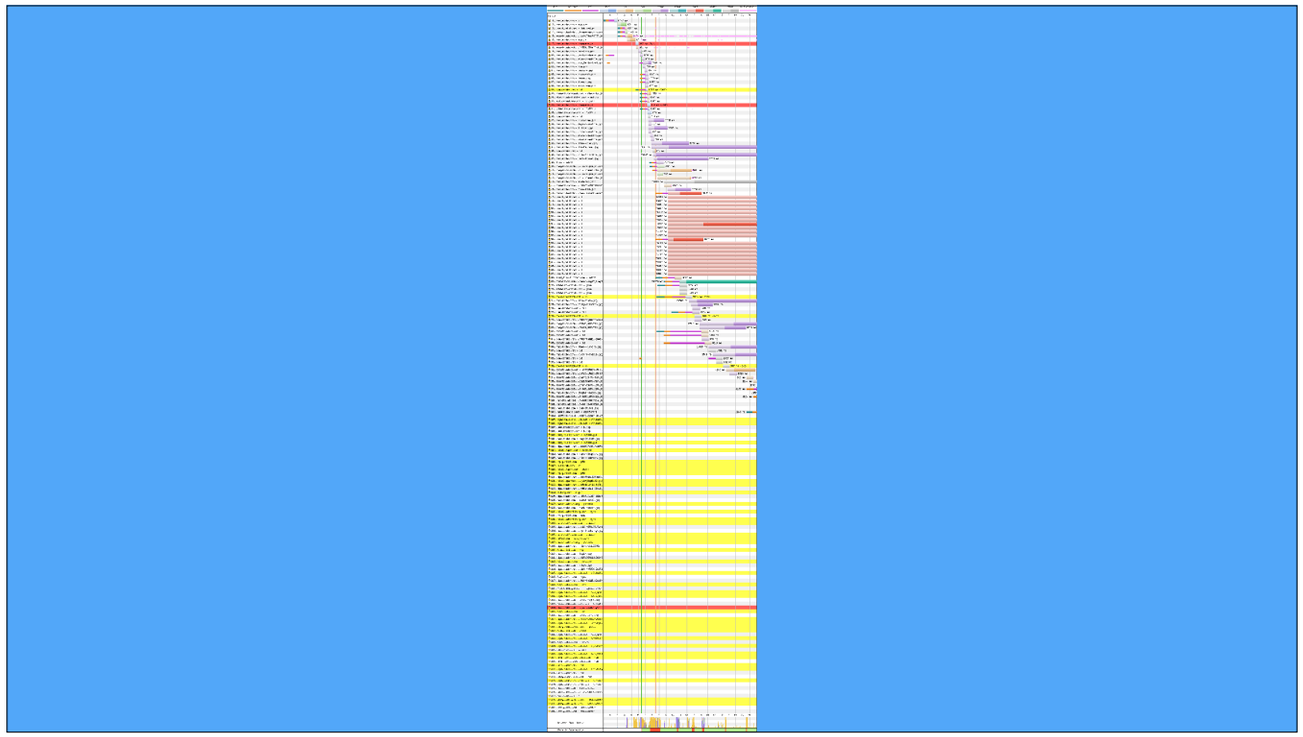
https://tympanus.net/Development/ShapeMorphIdeas/index3.html



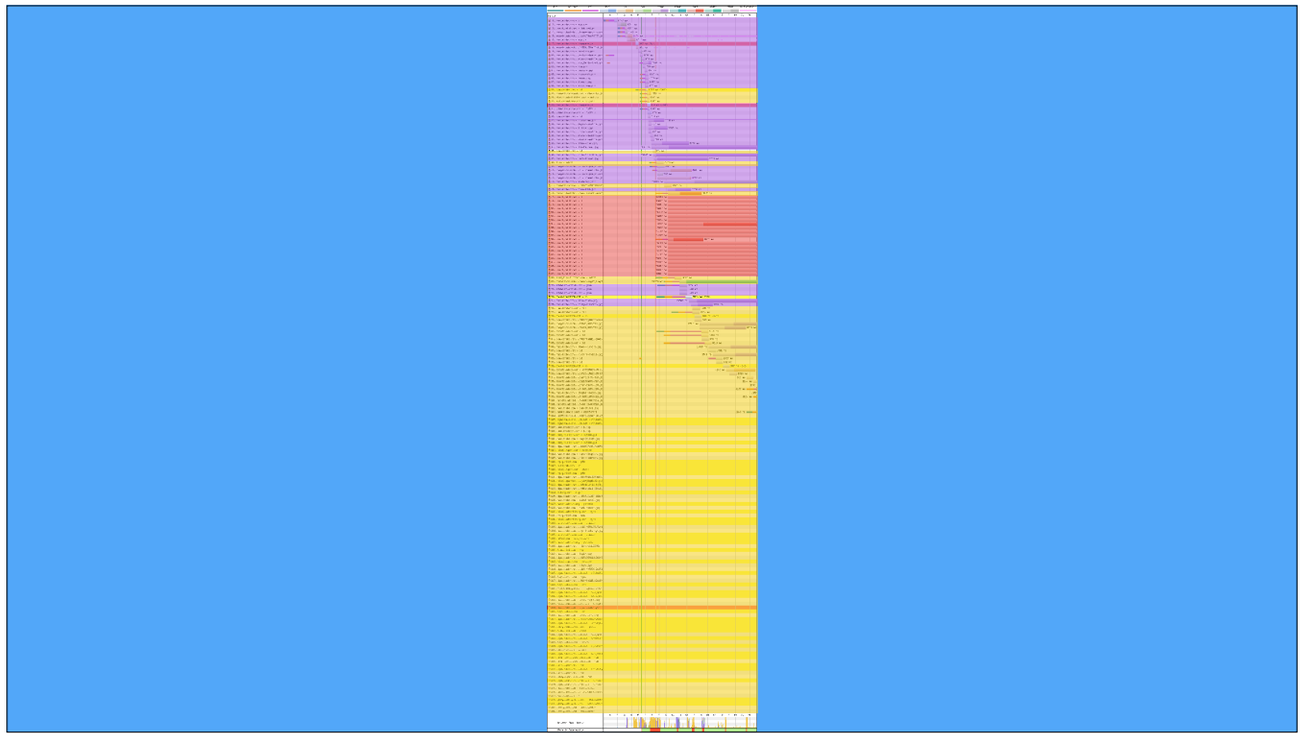
We drop tracking code after tracking code on to our site and for what? A bit of data? But what happens when those third parties fail? Does our site still load? Did you even test it?



Remove all of the tracking scripts on the page. Illustrate how much this improves the load time, and link that to the increase in sales seen with savings in load time from WPOStats.



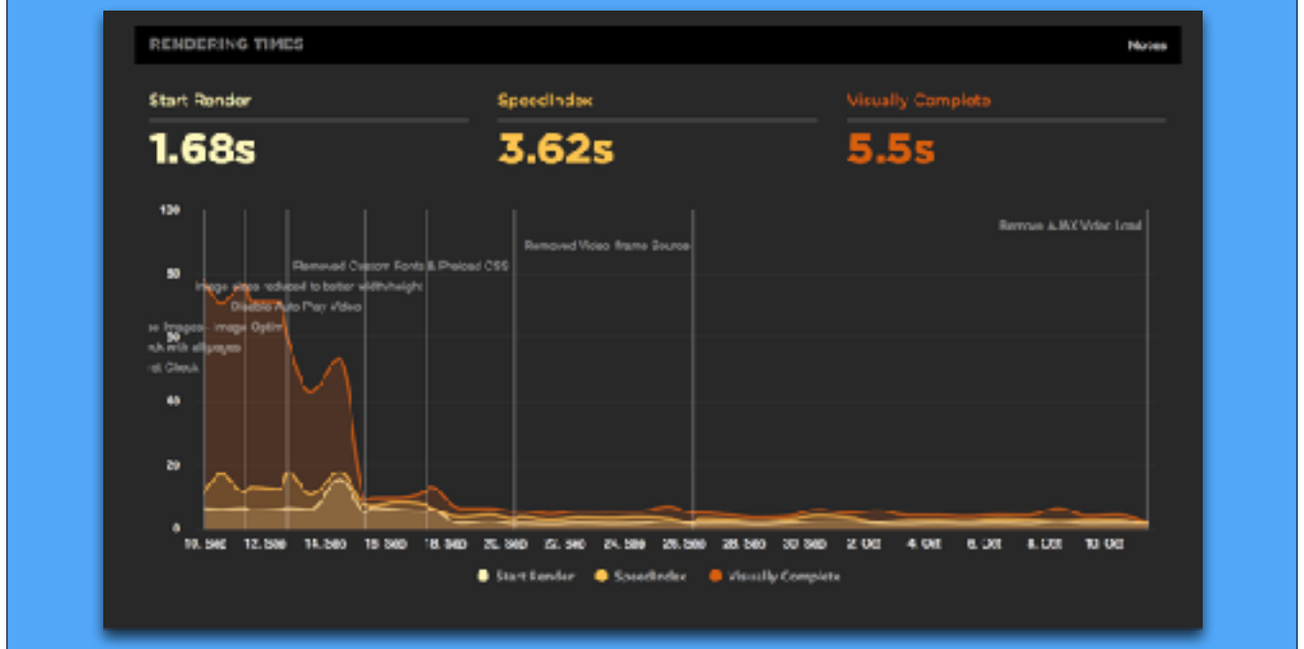
Remove all of the tracking scripts on the page. Illustrate how much this improves the load time, and link that to the increase in sales seen with savings in load time from WPOStats.



Remove all of the tracking scripts on the page. Illustrate how much this improves the load time, and link that to the increase in sales seen with savings in load time from WPOStats.

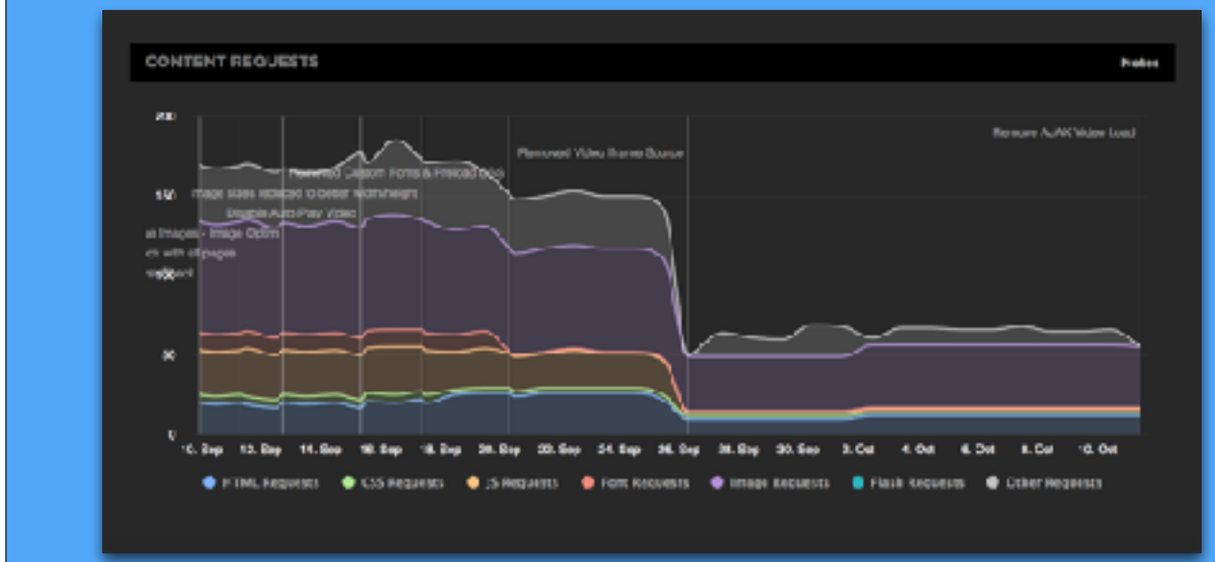
Step 9

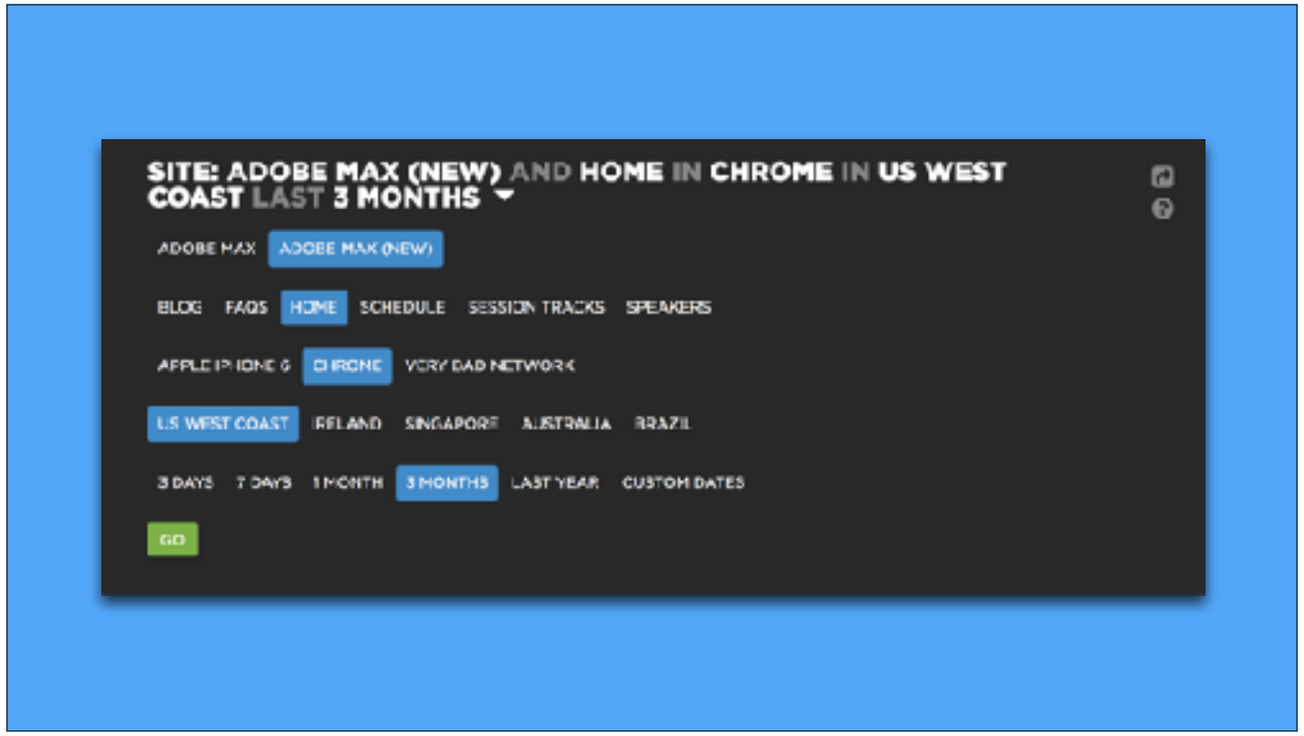
Monitoring

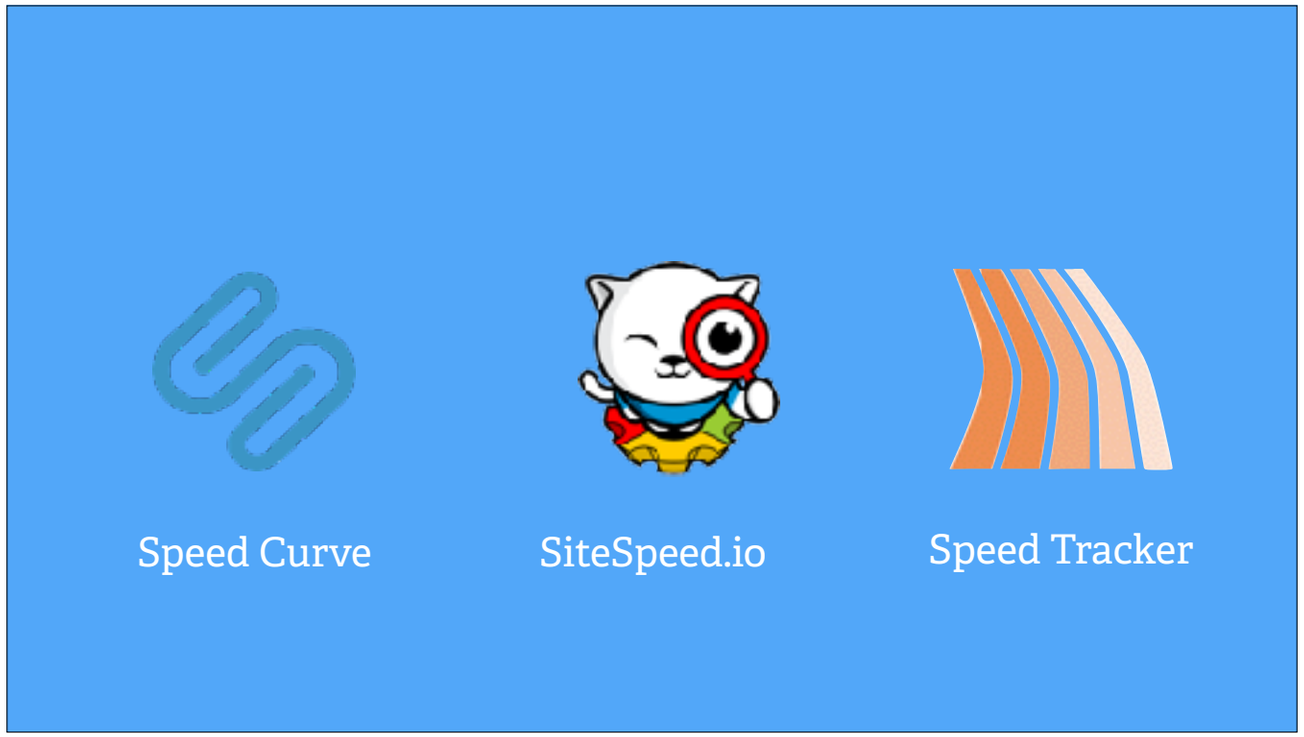

RENDERING TIMES
Notes
Start Render
1.68s
Speedindex
3.62s
Visually Complete
5.5s
Removed Video frame Source
Start Render
Visually Complete

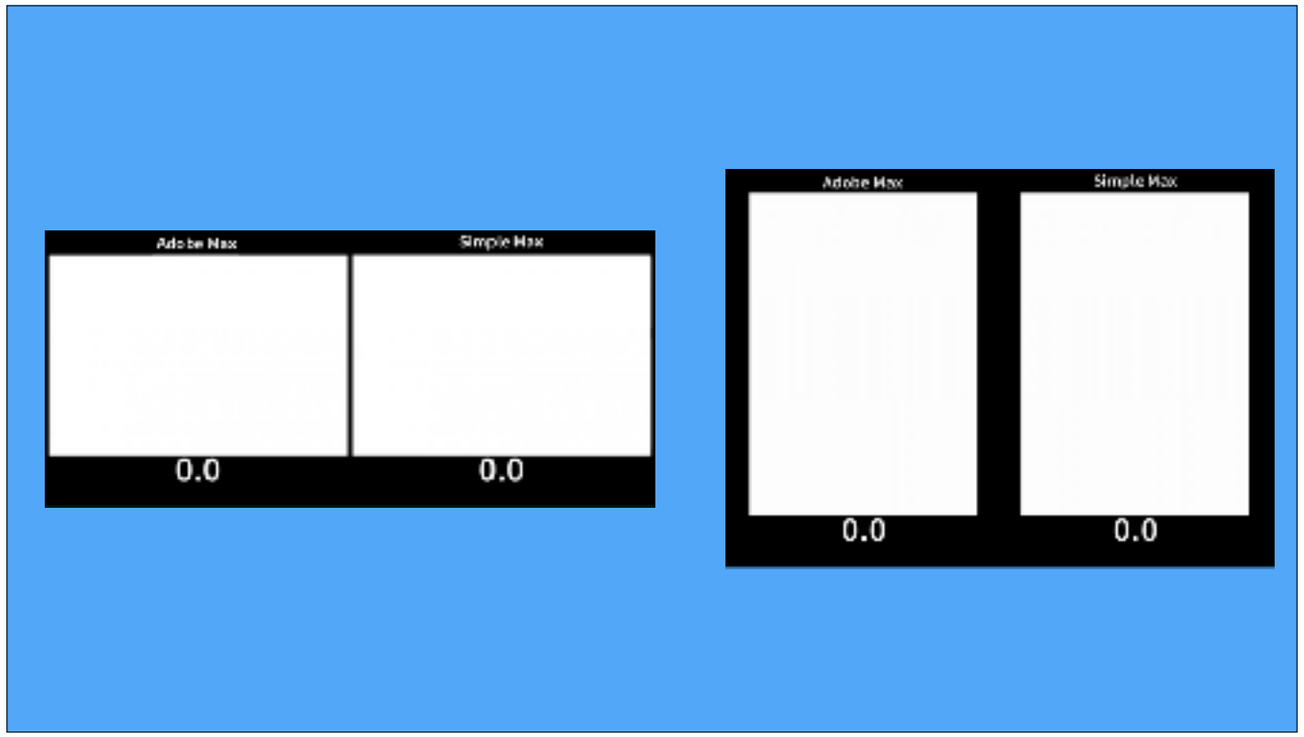

START RENDER TIME BUDGET
Notes
Current Start Render Time
1.57s
Start Render Time Budget
2s
Decrease over 90 days
(-75%)
-4.71s
Remaining budget (25%)
0.43s
Removed Video Iframe Source
Remove AJAX Video Load


FONTS SIZE BUDGET
Current Fonts Size
76KB
Fonts Size Budget
200KB
Decrease over 90 days
(-97%)
-503KB
Remaining budget (62%)
124KB
Removed Video Iframe source
Disable Auto Play Video
15. Sep
12. Sep
14. Sep
18. Sep
16. Sep
20. Sep
22. Sep
24. Sep
26. Sep
28. Sep
30. Sep
2. Oct
4. Oct
6. Oct
8. Oct


CONTENT REQUESTS
Remove AJAX Video Load
Permanent Video Iframe Source
Image Auto Play Video
Sep
Sep
Sep
Sep
Sep
Sep
Sep
Sep
Sep
Sep
Sep
Oct
Oct
Oct
Oct
Oct
HTML Requests
CSS Requests
JS Requests
Font Requests
Image Requests
Flash Requests
Other Requests


SITE: ADOBE MAX (NEW) AND HOME IN CHROME IN US WEST COAST LAST 3 MONTHS
ADOBE MAX
ADOBE MAX (NEW)
BLOG
FAQS
HOME
SCHEDULE
SESSION TRACKS
SPEAKERS
APPLE IPHONE 6
CHROME
VERY BAD NETWORK
US WEST COAST
IRELAND
SINGAPORE
AUSTRALIA
BRAZIL
3 DAYS
7 DAYS
1 MONTH
3 MONTHS
LAST YEAR
CUSTOM DATES
GO


Speed Curve
SiteSpeed.io
Speed Tracker


Adobe Max
Simple Max
0.0
0.0
Adobe Max
Simple Max
0.0
0.0

~~CSS Grid Layout~~

~~Faster Websites~~

Offline / PWAs

Cool Sh*tuff

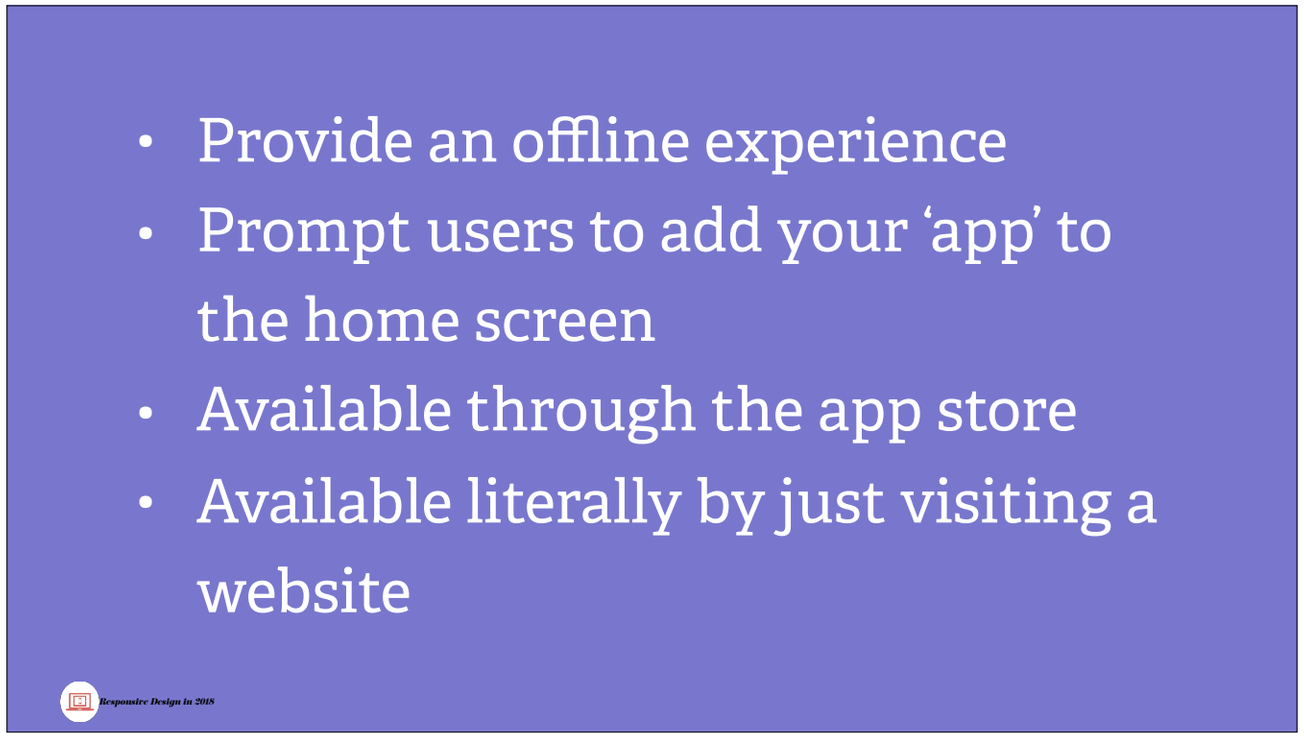
Offline / PWAs

Now I've talked a lot today about how we need to build a faster web. We need to be more thoughtful about how we approach the build of webpages to ensure that we can delivery amazing experiences for our clients, but also allow everyone one of their clients to still be able to access the content regardless of their situation.

The world of the web is about to go gangbusters.

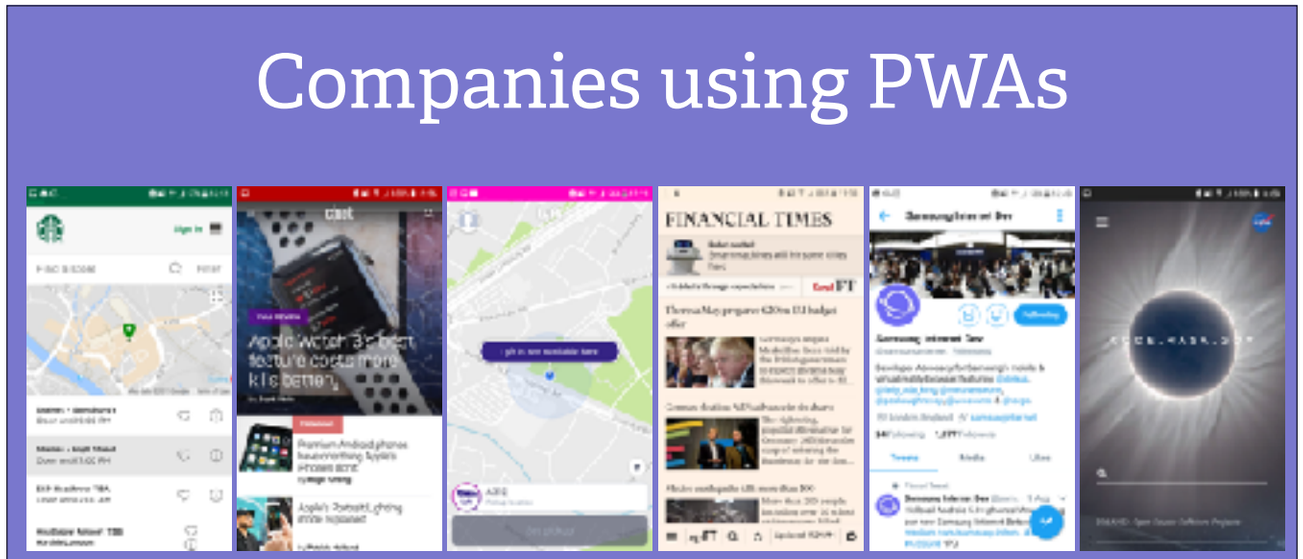
We are at the point where the web is very much catching up with the native world, as we saw with our offline web app.

PWA's are the next bit thing on the web and they're really what is doing the most to bridge the gap between the web and native.



- Provide an offline experience
- Prompt users to add your ‘app’ to the home screen
- Available through the app store
- Available literally by just visiting a website
- Push notifications when you don’t have the site open.
- BRIDGING THE GAP BETWEEN NATIVE AND WEB


https://www.theguardian.com/info#1-across
free become a member
sign in
search
jobs
more
UK edition
theguardian
home › home
UK
world
politics
sport
football
opinion
culture
all
Unable to connect to the Internet
In the meantime, how about a quick crossword?
Quick crossword No 14,188
Across
1 Poetic name for Britain (6)
4 Unpowered aircraft (6)



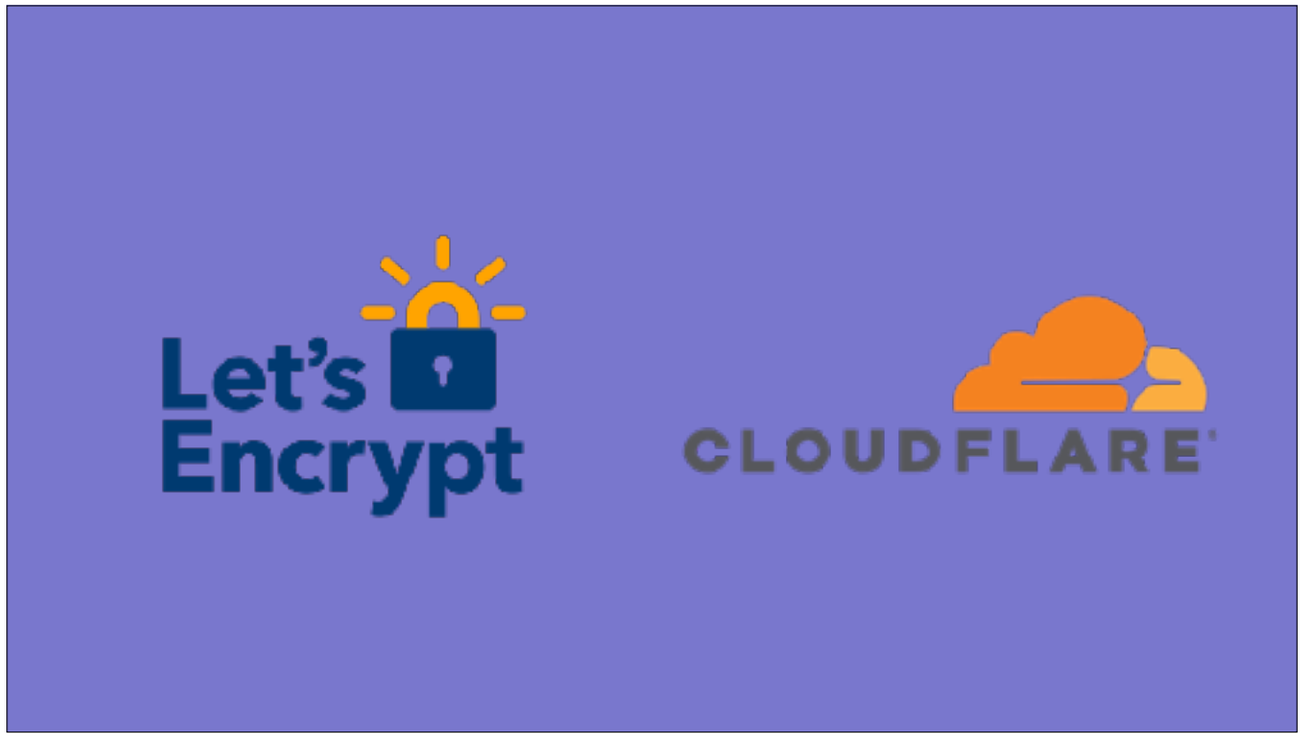
Thanks to **Peter O'Shaugnessy** for pulling together this slide.

Starbucks, CNET, Lyft, FT, Twitter, NASA

Starbucks were one of the first big brands to go responsive, now they're doing the same with PWA.

So how do we build one?

Step 1: HTTPS



Let’s Encrypt - a new free way to get SSL

Cloudflare - a CDN that also provides FREE SSL.

Step 2: manifest.json

Step 2: manifest.json

```
{
  "short_name": "AdobeMax",
  "name": "Adobe Max 2017 Conference",
  "icons": [
    {
      "src": "launcher-icon-1x.png",
      "type": "image/png",
      "sizes": "48x48"
    },
    {
      "src": "launcher-icon-2x.png",
      "type": "image/png",
      "sizes": "96x96"
    },
    {
      "src": "launcher-icon-4x.png",
      "type": "image/png",
      "sizes": "192x192"
    }
  ],
  "start_url": "index.html?launcher=true"
}
```

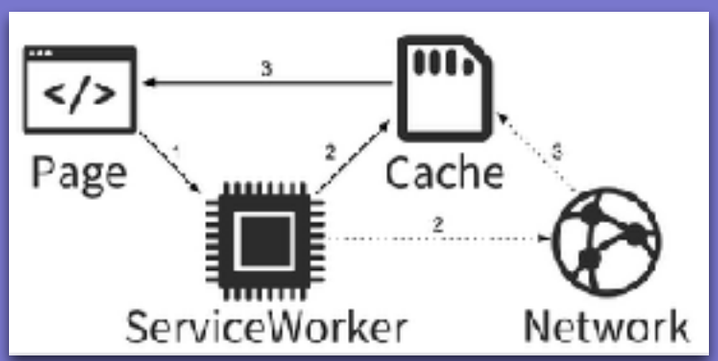
Everything you need.

- Short Name, for home screen
- Name, for the title bar in the web app
- Different icons for different screens
- A start URL.

Step 3: Add a serviceworker.js

A service worker does……..

Show Can I Use
Show browsers that are working on it.
Show an example of how it works



Any traffic for the site goes through the service worker first.

It checks if the request is already in cache, and if not it goes to the network.

If the network doesn’t exist, i.e. you’re offline, it returns an offline page.

brieferhistoryoftime.com

adobemax.simplethin.gs

IE has it under active development.
Webkit is under consideration (fingers crossed).
This is a progressive enhancement anyway.

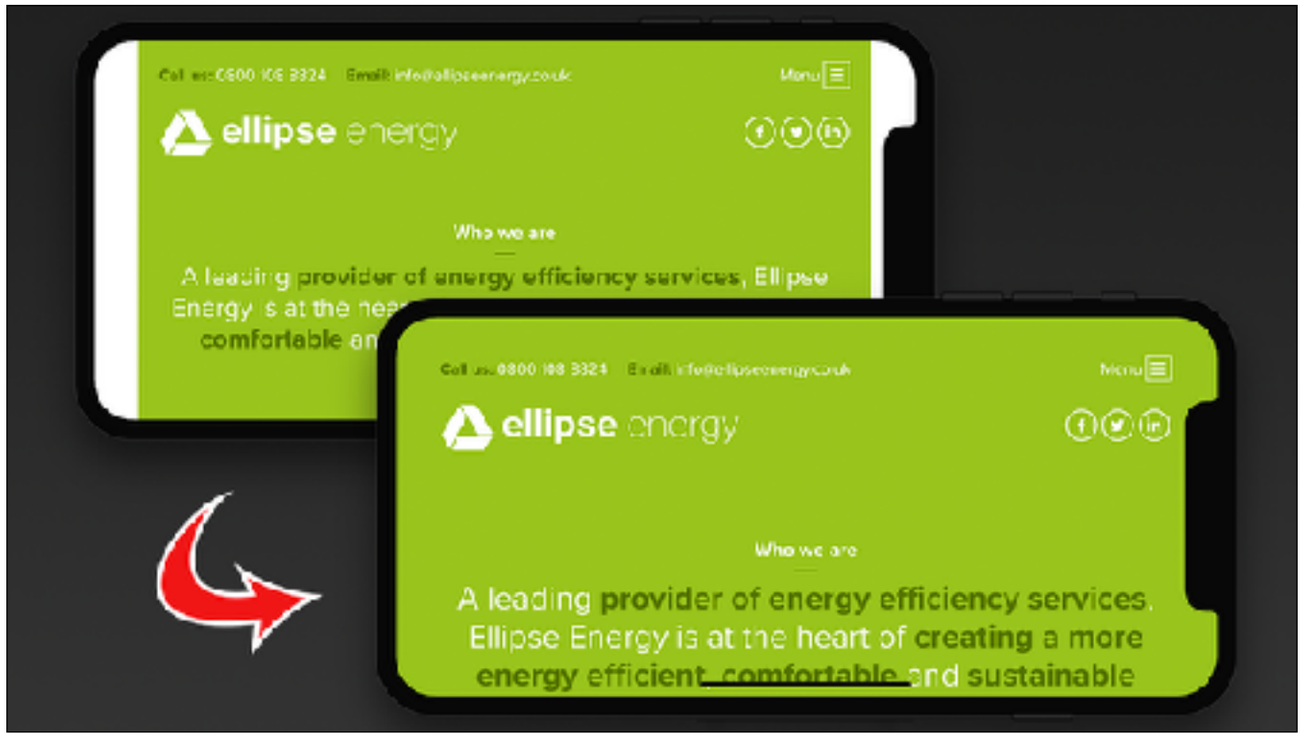
~~CSS Grid Layout~~

~~Faster Websites~~

~~Offline / PWAs~~

Cool Sh*tuff

Cool Sh*tuff



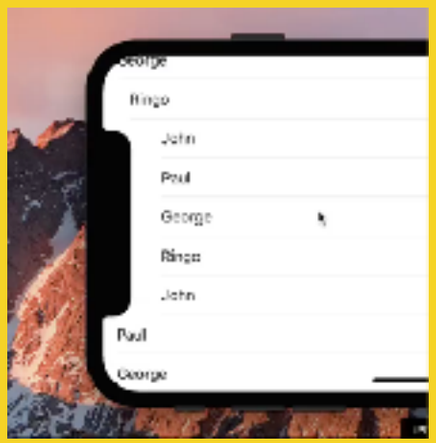
http://stephenradford.me/removing-the-white-bars-in-safari-on-iphone-x/

```
<meta name="viewport"
content="width=device-
width, initial-scale=1.0,
viewport-fit=cover">
```

Now we have it expanding across the full viewport, but we lose content behind the notch!

```
padding: constant(safe-area-inset-top)
constant(safe-area-inset-right) constant(safe-
area-inset-bottom) constant(safe-area-inset-
left);
```

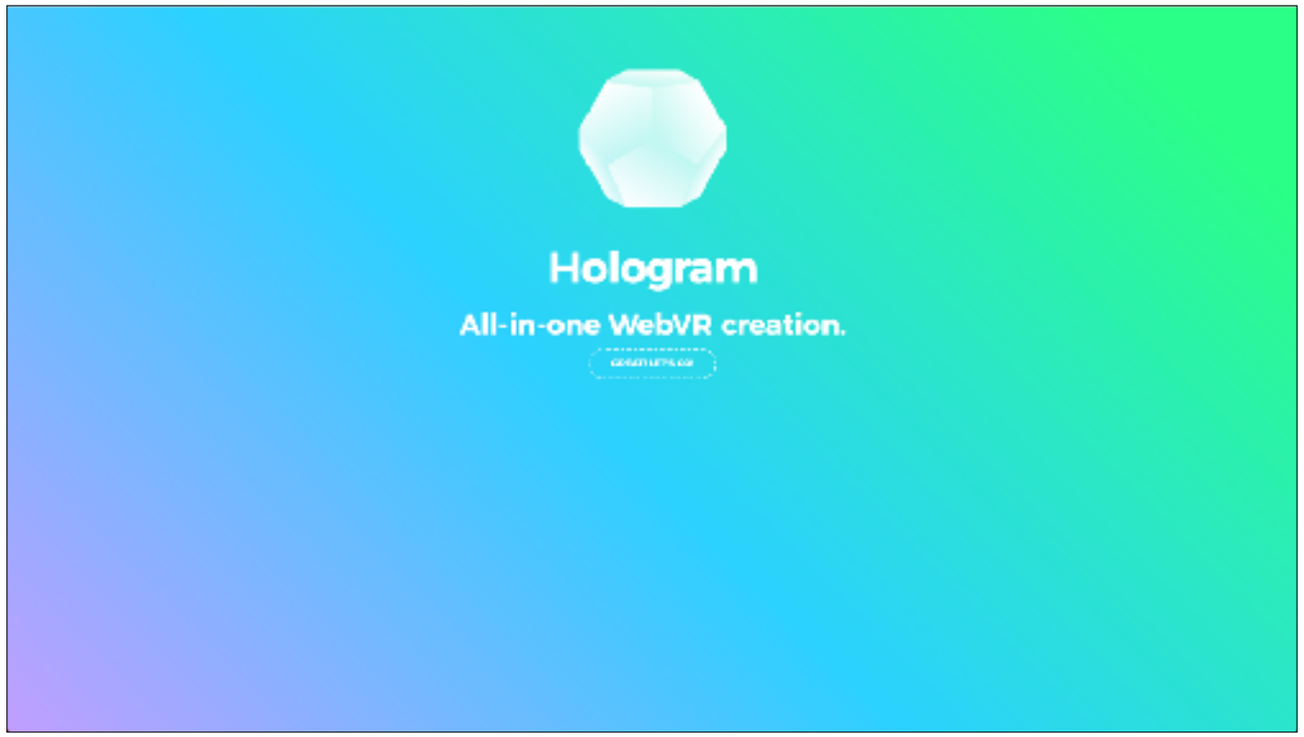
https://drafts.csswg.org/css-round-display/#viewport-fit-descriptor

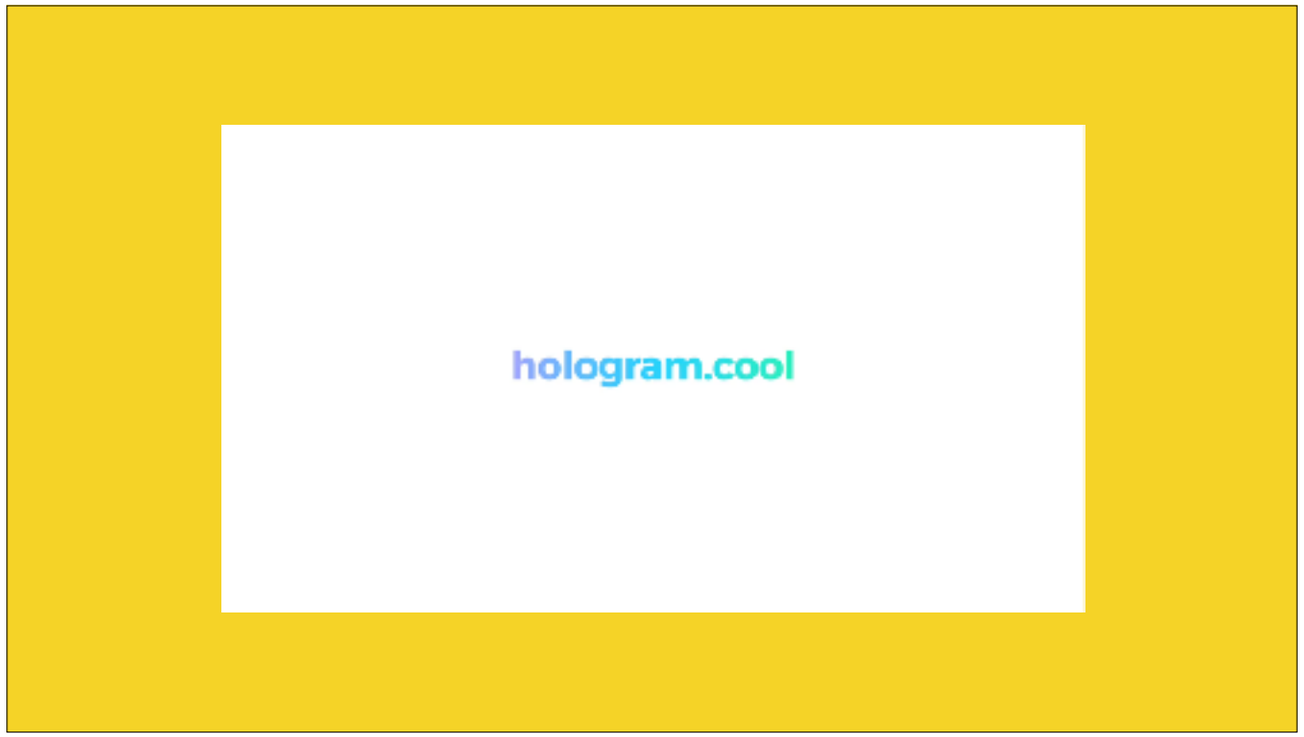


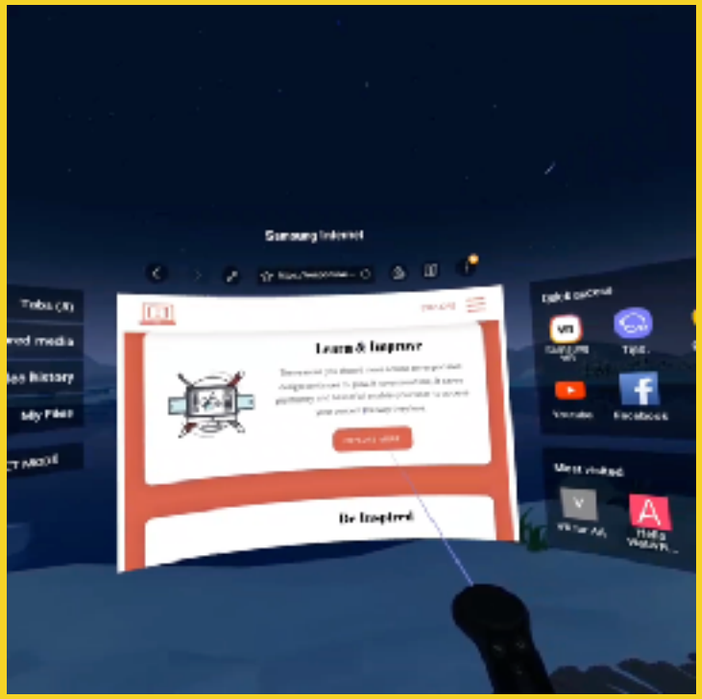
https://twitter.com/vojtastavik/status/907911237983449088?ref_src=twsrc%5Etfw&ref_url=https%3A%2F%2Fcss-tricks.com%2Fthe-notch-and-css%2F

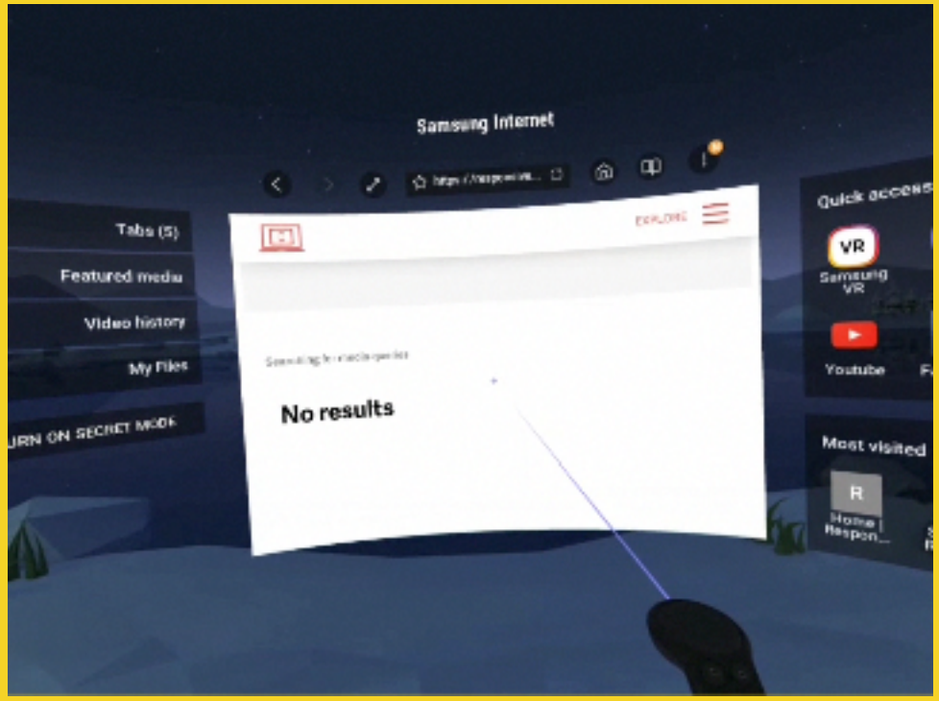
https://twitter.com/vojtastavik/status/907911237983449088?ref_src=twsrc%5Etfw&ref_url=https%3A%2F%2Fcss-tricks.com%2Fthe-notch-and-css%2F

Web VR


Hologram
All-in-one WebVR creation.


hologram.cool


Samsung Internet
Learn & Improve
Be Inspired


Samsung Internet
Tabs (5)
Featured media
Video history
My files
No results
Quick access
Samsung VR
Youtube
Most visited

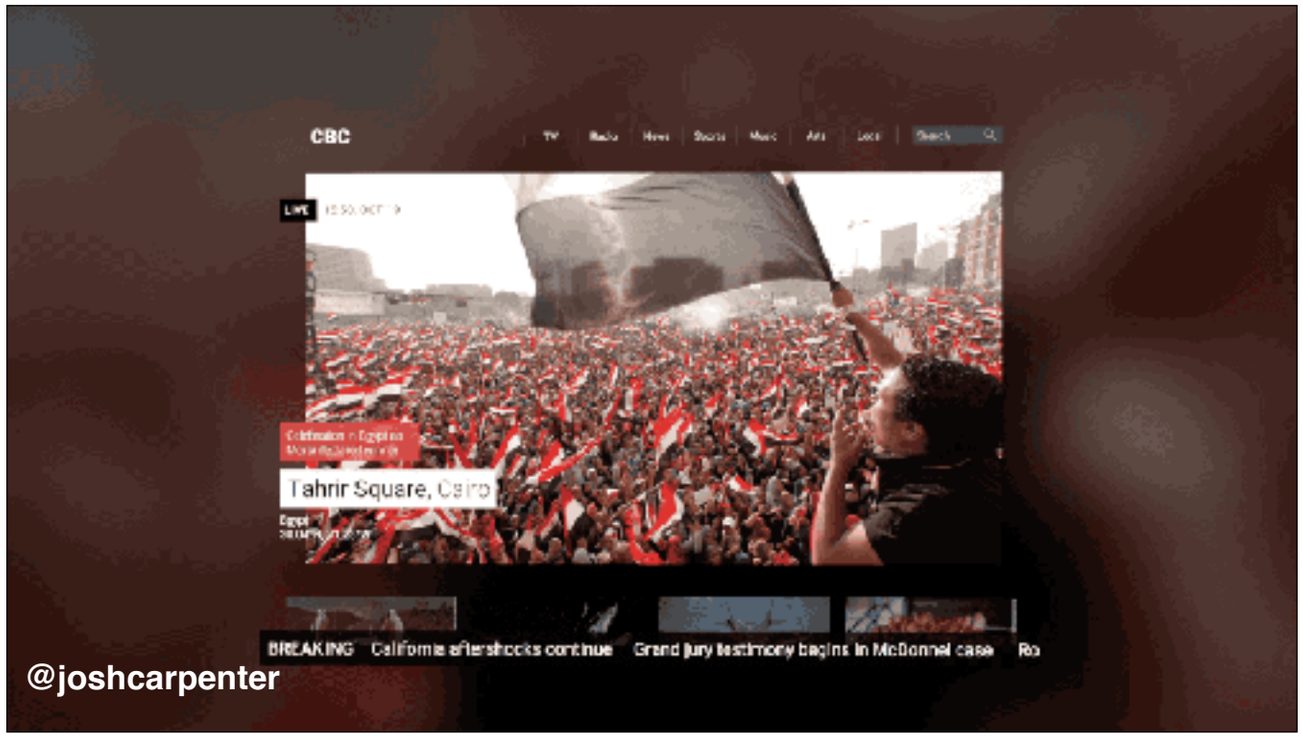
https://archilogic-com.github.io/aframe-gblock/

https://archilogic-com.github.io/aframe-gblock/

https://archilogic-com.github.io/aframe-gblock/

Josh Carpenter has done some experiments into WebVR and was the lead UX at Mozilla before moving to Google to do a similar role.

Here we are breaking out of the confines of the browser that we saw in the earlier example.

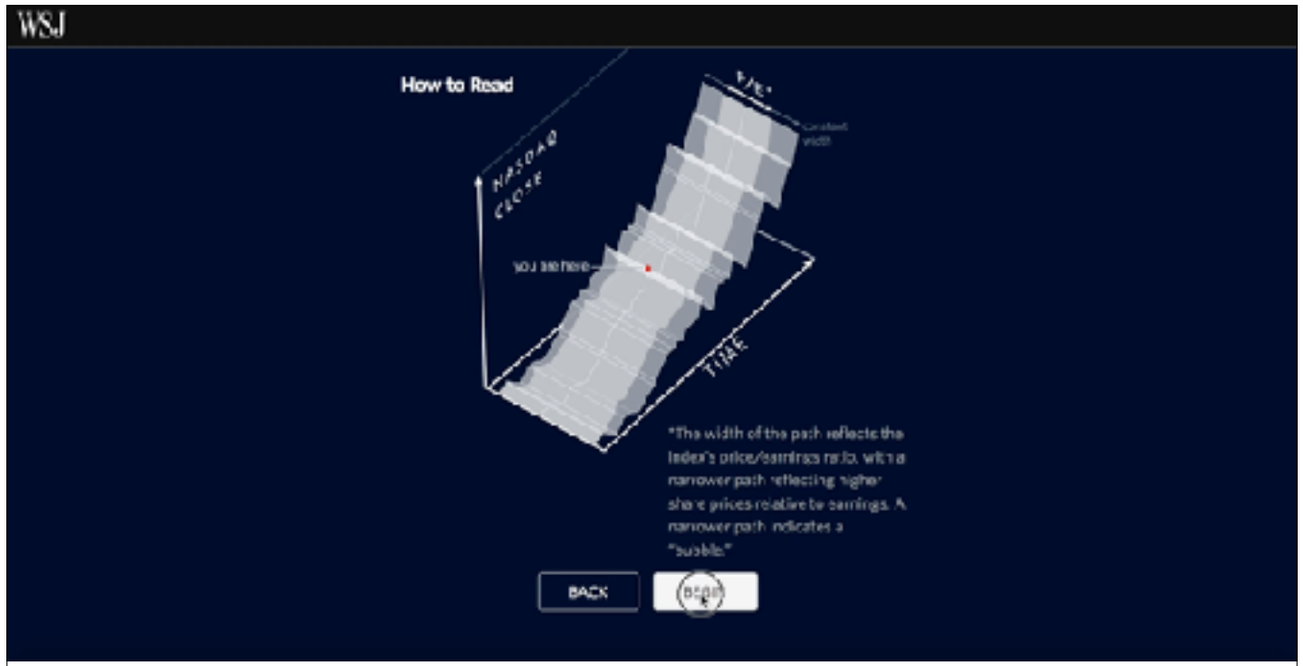


@joshcarpenter

We are now able to look at taking more advantage over the z-index.

And then if the user agrees to allowing the website to take over more of the screen it we can do it….

The content now spreads out across the entire viewport.



Responsive Design in 2018

http://graphics.wsj.com/3d-nasdaq/

This was an experiment from the Wall Street Journal.

Empathy

So how do you create responsive content that spans all devices?

http://graphics.wsj.com/3d-nasdaq/

- Desktop (2D): CLICK(HOLD)/DRAG + CLICK.

(Supported on Windows(Chrome/Firefox/Edge/IE) & Mac(Chrome/Firefox/Safari))

- Tablet (2D): TOUCH/SLIDE + TAP

(Supported on Android(Chrome) & iOS(Safari))

- Mobile (2D): ROTATE + TAP

(Supported on Android(Chrome) & iOS(Safari))

- Cardboard (3D/VR Polyfilled): ROTATE + TAP

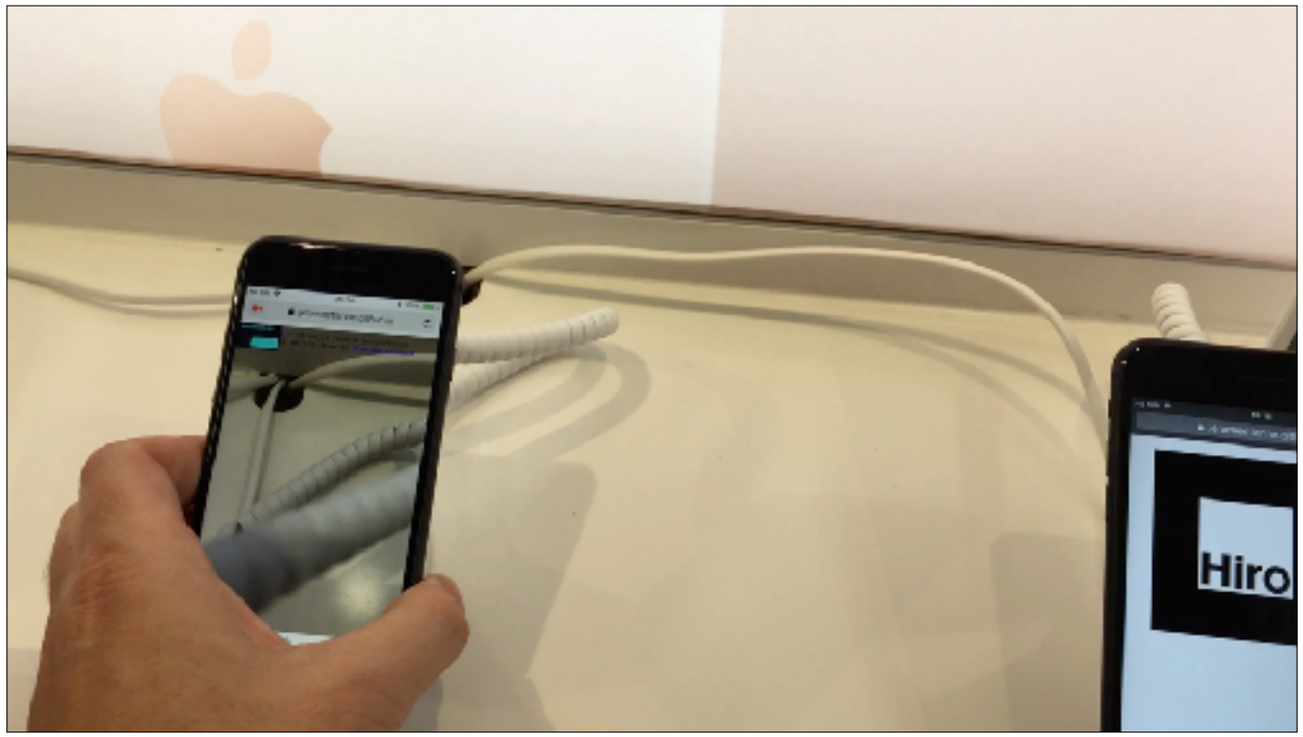
(Supported on Android(Chrome) & iOS(Safari))

- Daydream (3D/WebVR API): ROTATE + TAP(CONTROLLER)

(Supported on Android(Chrome) on Daydream ready phones)

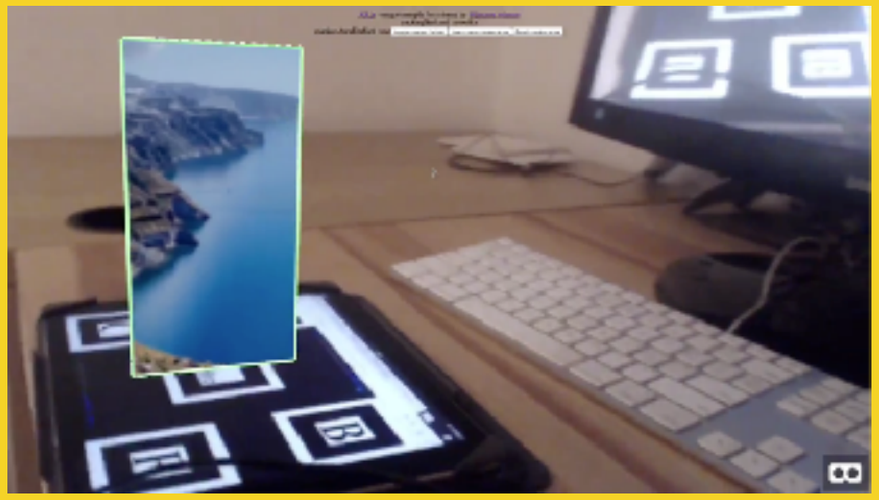
Web AR

With the release of the Google Pixel and new Iphone they have camera setups and onboard processing power designed to handle augmented reality.



At the moment we just see a spinning object, but imagine using AI tools like sense’ to pass through what you’re viewing and return in real time valuable information.

Speed signs = current speed.
Food = nutritional value



https://twitter.com/jerome_etienne/status/893217730517749760

https://twitter.com/jerome_etienne/status/893217730517749760

CSS Media Queries 5

@media (light-level: *variable*)

normal | *dim* | *washed*

@media (light-level: *variable*)

```
@media (light-level: normal) {
  p { background: url("texture.jpg"); color:
#333 }
}
@media (light-level: dim) {
  p { background: #222; color: #ccc }
}
@media (light-level: washed) {
  p { background: white; color: black; font-
size: 2em; }
}
```

@custom-media: name [media-query]

@custom-media: name [media-query]

```
@custom-media --big-screens(min-width: 100em);
@custom-media --dark(light-level: dim);

@media (--big-screens) {
  /* big screen styles here */
}
@media (--big-screens) and (--dark) {
  /* special styles for when your on a big
screen in the dark */
}
```

Voice

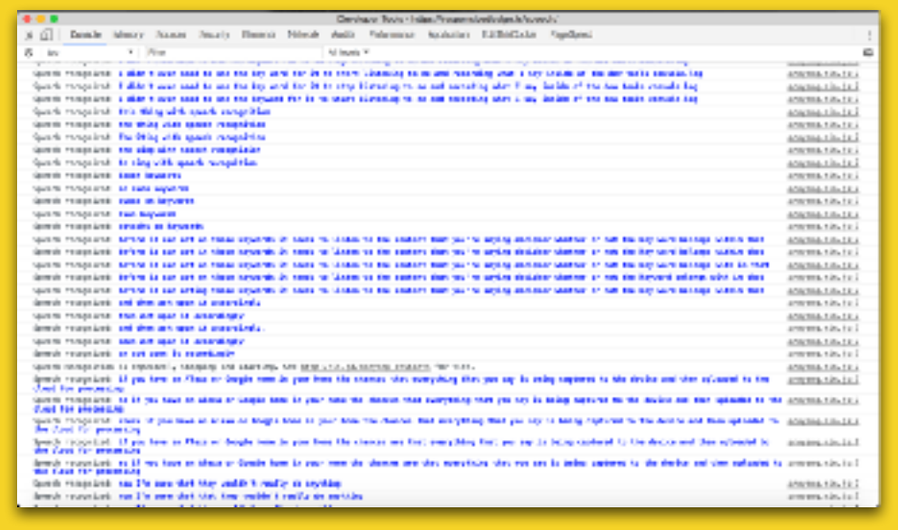
As we saw this morning with **Adobe Sense’**

Voice Search = **natural**

VR headset = voice control

Wanna see something scary as sh?t

Wanna see something scary as hell?



Building Progressively

Progressive Enhancement isn’t about building a crappier version of your site for people that might decide to turn off javascript. It is not about saying you shouldn’t use Javascript at all. It’s about building the site in a way that you can use as much javascript as you could possibly want…. layer that website with 35 different flavours of JS if that’s what you want to do… but building with progressive enhancement means that in the off chance that something goes wrong with the JS the user isn’t starring at a blank screen.

1. Identifying what your user needs are
2. Building a site using the most simplest of methods to deliver the tasks/information your users need on the site.
3. Improving upon how those items work if the user happens to be using the latest devices

Before we get in we’re going to look at a bit of history, take a step back into yesteryear to get the lay of the land.

> “Web design must mature and accept the developments of the past several years”

Web design must mature and accept the developments of the past several years.

We need to embrace the **new ways** in which we can **connect** and **experience** content on the web.

It is our responsibility to explore and create those experiences…. yet when we do it we need to make sure we do it in a **sensible all inclusive way**.

> *“The goal of web design is not merely to dazzle, but to deliver information to the widest audience.”*

At the end of the day we are delivering content to users in one way shape or form.

While it is great to make that experience as impressive as we can do, **the goal of web design is not merely to dazzle but to deliver information to the widest audience.**

I encourage you to take up building out layouts with
CSS Grid Layout
Making your **websites** as **fast** as they can be from the **outset**
Progressively enhancing them into **progressive web apps**
Taking advantage of the new inputs and outputs that are available today.

3.8 billion users

What will you create?

3.8 billion users right now with another 1 billion on the way.

What will you choose to create for them?

Thank you

- https://responsivedesign.is
- @justinavery / @reswebdes
- https://maxtalk.simplethin.gs

- Ladies and Gentleman, thank you.

- Is there time for questions?

Two more things….

1. On your Max App you are encouraged to provide **feedback on each session** you attend. You go into a draw to win prizes, we also get feedback on how we went as speakers which is super valuable.
2. I have some **RWD notebooks available**, I'm sorry I don't have one each but please come and pick one up.

Removed from 2017/18

Introduction

(no time for stories :( )

To look at breaking the traditional moulds of the web I think it’s best if we take a little stroll back down memory lane to find out where we came from so that we know which direction we need to be going.

http://www.writeofthemiddle.com/wp-content/uploads/2017/04/img_4823.jpeg

6th August 1991

Pages were light - there wasn’t any images, css, or js to slow our pages down.

Didn’t worry about font choices, again nothing to slow down our site

Everything was responsive, we just happened to all have the same monitor

We focussed on making it a web of hypertext links…. at the end of the day I think we forget about that. The web is indeed that, a web of information linked together to make it easy to discover and consume knowledge.

Bryan Adams - Everything I do, I do it for you. -44s
Hot Shots, and Terminator 2: Judgement Day

I’m sure there’s many people sitting in the room today who hadn’t been born when this page went online.

In 1993 images arrived. This is an email from Marc Andreessen recommending the img tag>

I will Always Love You - Whitney Houston - 42s
Groundhog Day & Falling down

The arrival of CSS

On October 10 1994 we had CSS hit the scene and the trusty float was around to help us put images to the left and the right and have text wrap around them.

Floats, designed to be really good at doing one thing right (and page layout was not one of those as we will learn later in this talk).

The arrival of CSS

On October 10 1994

I’ll make Love to You - Boyz II Men. - 23s

Pulp Fiction is released

1996

On the 16th September 2014 the W3C moved HTML5 On 16 September 2014, W3C moved HTML5 to Proposed Recommendation but browsers had been supporting the tags for a long time and the best thing about the HTML language and writing in it is that if the browser doesn't understand something you've included, like <date> for example, then it will just parse the contents as if it existing in a with no harm, no foul.

Although this made the transition at this point, we were using for a long time before this as HTML5 wasn't just about a set of new tags to use, but really around a set of Browser API's that would help us win the race against the Native apps.

Shake It Off - Tylor Swift, just before (although confusingly it’s off the album titled 1989.

All about that bass - Meghan Trainor

Then we had the grand arrival of Responsive Design!

I remember back in 2011 just as I was about to leave the UK and move back to Australia with my now wife I was working with University in London, a reasonably major university at that. We had previously quoted £5k to design and create a mobile version of their website to run off a m dot version of the site.

We had one meeting where I described what we would be doing as part of the project and why it would be more advantageous to create a responsive version rather than a mobile specific version which was rightfully signed off.

In a subsequent meeting to discuss the next steps they brought along an external consultant that had been working closely with the Marketing team in the past who managed to convince the budget holder that people would rather double tap to zoom in on the content rather than have it render to the size of the screen.

Now days I can sit back and smugly smile and say “how’s your pinch and zooming going now huh”…. But the thing is I’m not smug about it, I’m annoyed at myself for not being able to take the client on the journey to realise what a fundamental error they were making by taking that particular approach to building the website.

It is our responsibility and digital designers, strategists, UI or UX designers, content strategists, native app developers or Sales folks to persuade our clients to build their sites in a way that is future proof and will stand the test of time. We don’t want to build things based upon a fad or a short term buzz word, but instead build it in the right way from the foundation.

"Not Afraid” - Eminem
“OMG" - Usher featuring will.i.am

Shrek Forever After
Toy Story 3

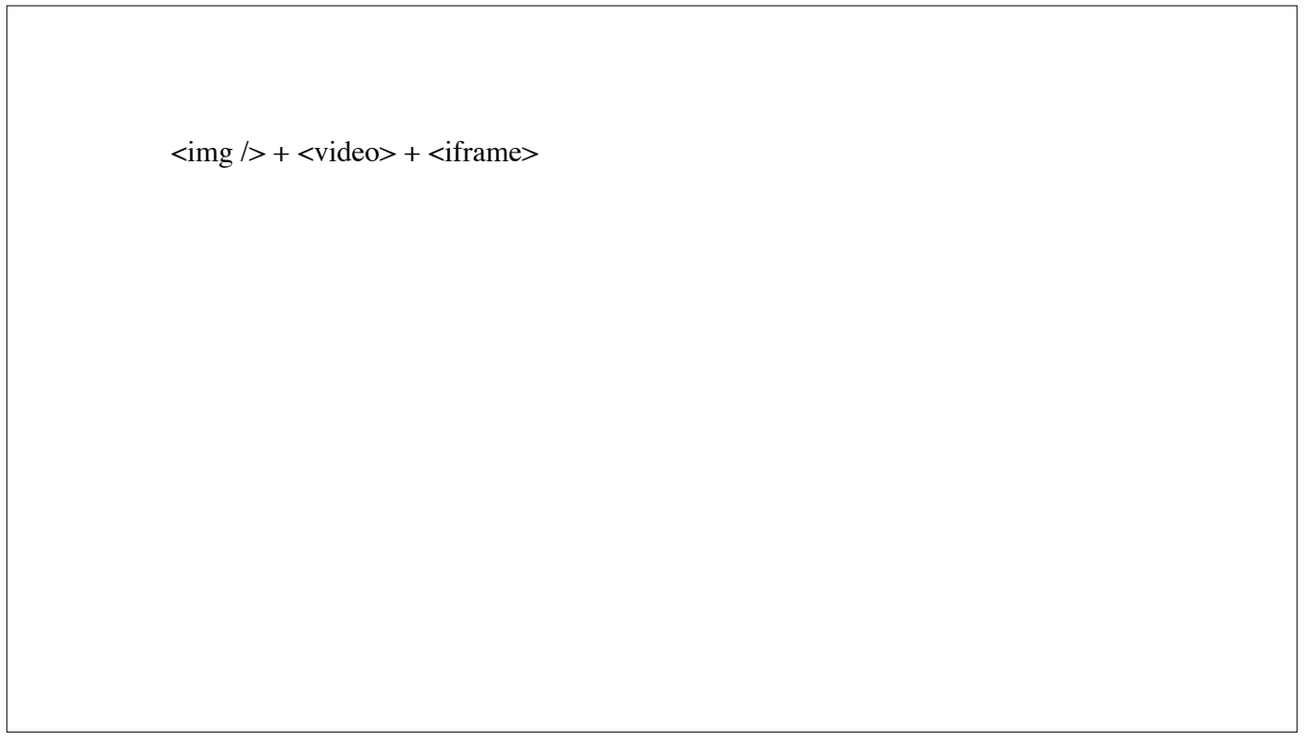
Content
This is what really matters. It’s the information that every single person goes to your website to get. Maps. Email. Facebook

Content First. It’s the most important thing.

HTML

Mark up
<html><head><title><body> <b><i><p>

Originally this was all we had for markup. Back when Bryan Adams was singing his song.



Then once Whitney joined the scene…. we have IMG at least.

Semantics
<header> <article> <section> <aside> <date> <button> <input>
<quote> <blockquote> <code> <pre> <figure> <figcaption>

By the time Taylor Swift was shaking things up we had our semantics.

This is our first ingredient of the web, the HTML. Without HTML, the web would not exist. By the

Cascading Style Sheets (CSS)
Making our content look more appealing

This is our second ingredient, when boyz 2 men we’re making love to us.

It’s designed to make our sites look different, more appealing.

Type
Adding Type (font-family), personality, brand.

Adding Type (font-family), personality, brand.

Font Size/Weight

Font size and font weight will allow you to provide visual hierarchy to improve readability of your content.

Margin & Padding

Margin and padding allows you to create space around your message to improve the readability.

Color

color: #222; and background-color:#eee;

Content

This is what **really** matters. It's the information that *every single person* goes to your website to get.

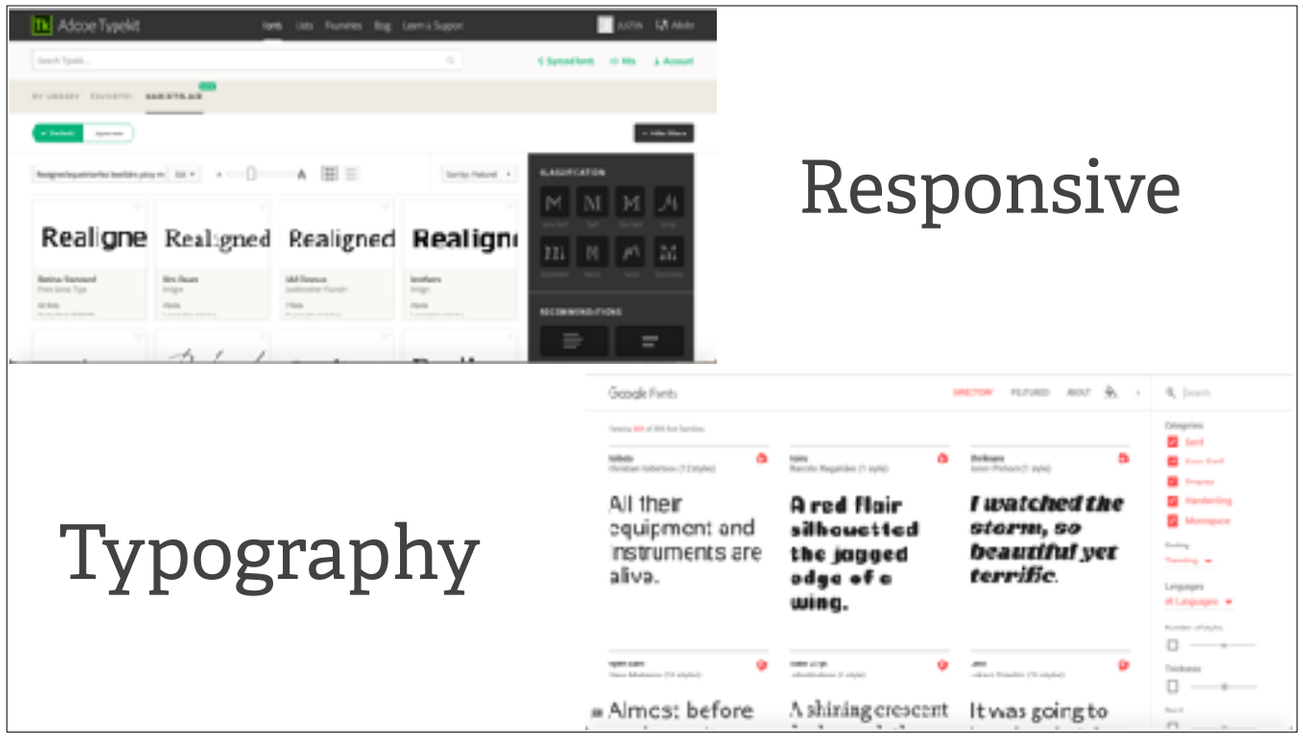
Maps. Email. Facebook

CSS Grid Layout

That was a transition that you might expect on a flashy website. It probably uses CSS3 transitions as much as possible, but the real hero with manipulating the DOM is **Javascript** and is often the shunned third ingredient of the web.

That was a transition that you might expect on a flashy website. It probably uses CSS3 transitions as much as possible, but the real hero with manipulating the DOM is Javascript and is often the shunned third ingredient of the web.

This talk won’t have a lot to do with Javascript. Not because it’s not a wonderful ingredient and the web but because I feel that every job that I see these days requires 10 years experience in React even though it’s only been around for a couple of years and I want to show that there’s more than enough you need to know without ever getting into javascript…. although we we start looking at some solutions we will dabble into it ever so slightly.



Responsive

Typography

With more than 1000 fonts to choose from now on Adobe Typekit, and a further XXX on fonts.google.com


MIKE'S
BBQ
JOINT
LEO'S
PIZZA
NGRATULATIONS DIG G
START-UP WORLD:
PRODUCT
DESIGN
77th
60
ONLINE SALE
9A-DMM
THREE
THINGS
TO LEA
ON THE BEING of
AFFECTION
334 N 6th Pl. – Elizabeth Johnstone – (717) 420-5921
ACCOUNTING & TAXES
BUSINESS ASSESSMENT - STRATEGY IMPLEMENTATION - REFINING AND ADAPTING
Issue №2
BAT
"The core business of the plaintiff
ads to its visitors. Journalistic co
vehicle to get readers to view the a
22 23
Posterfes
Zagreb • Croati
free admission for everyon

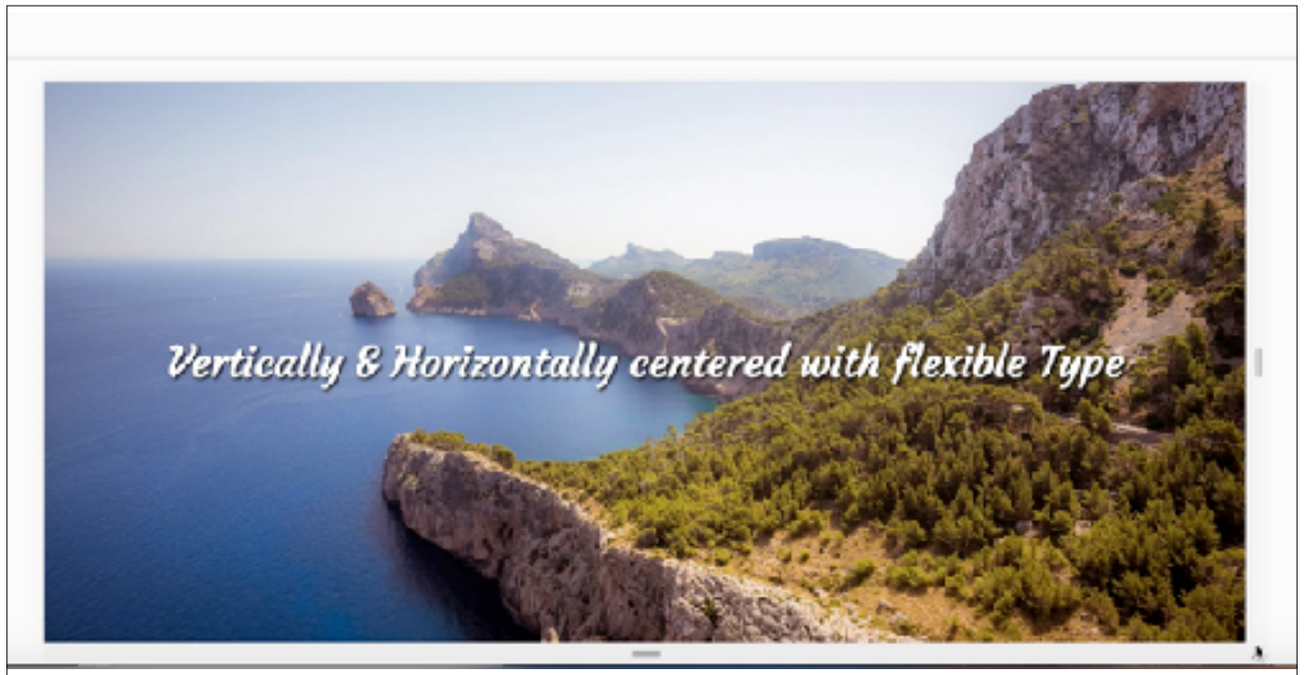
http://www.axis-praxis.org/specimens/__DEFAULT__

```
body {
    font-size: 1em;
    font-size: calc(1em + 1vw);
}
```


Super flexible font sizes across the board.

you get a standard font size, 1em based on your body font size (100%) and then you're adding 1vw width to it. VW is the viewport width of the current window, so if it's a really small window then the text will increase in size a very small amount, and if it's a really big window then the text will increase a larger amount.

The idea is that you get a fluid font across any viewport without having to add in any breakpoints to increase your font size.



Here’s an example of a few of these things working.

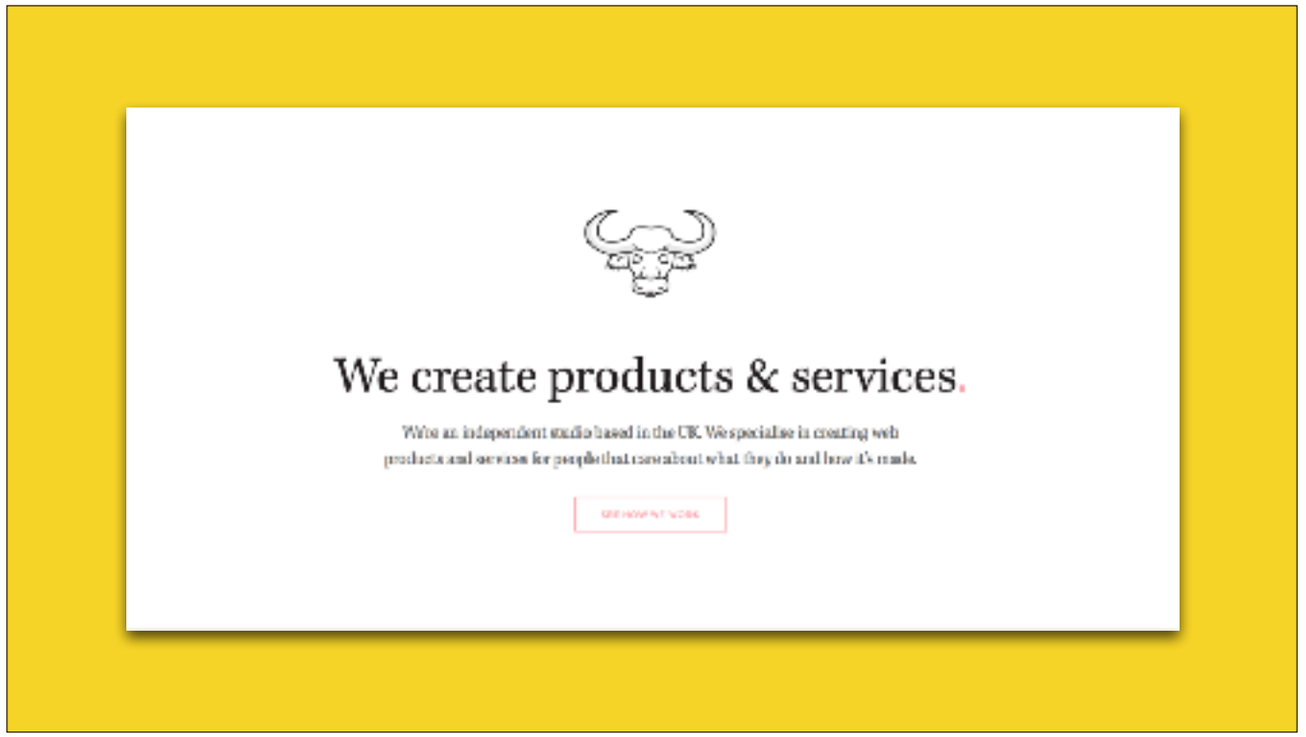
1. Fluid Text
2. Setting the image to object-fit
3. Setting the text position to be centre centre with flex box
4. Setting the height of the first panel on the page to 100vh/vw

Viewport Height/Width is a measurement that can be used anywhere that you’ve included % or EM. The difference between VW and VH

DESIGN/DEVELOPMENT PROCESS

(not included)

There are many



Designing in the browser

Sketch and Invision wouldn’t exist if everything was developed in the designer.

Not thinking about reusable code if you’re designing in the browser.

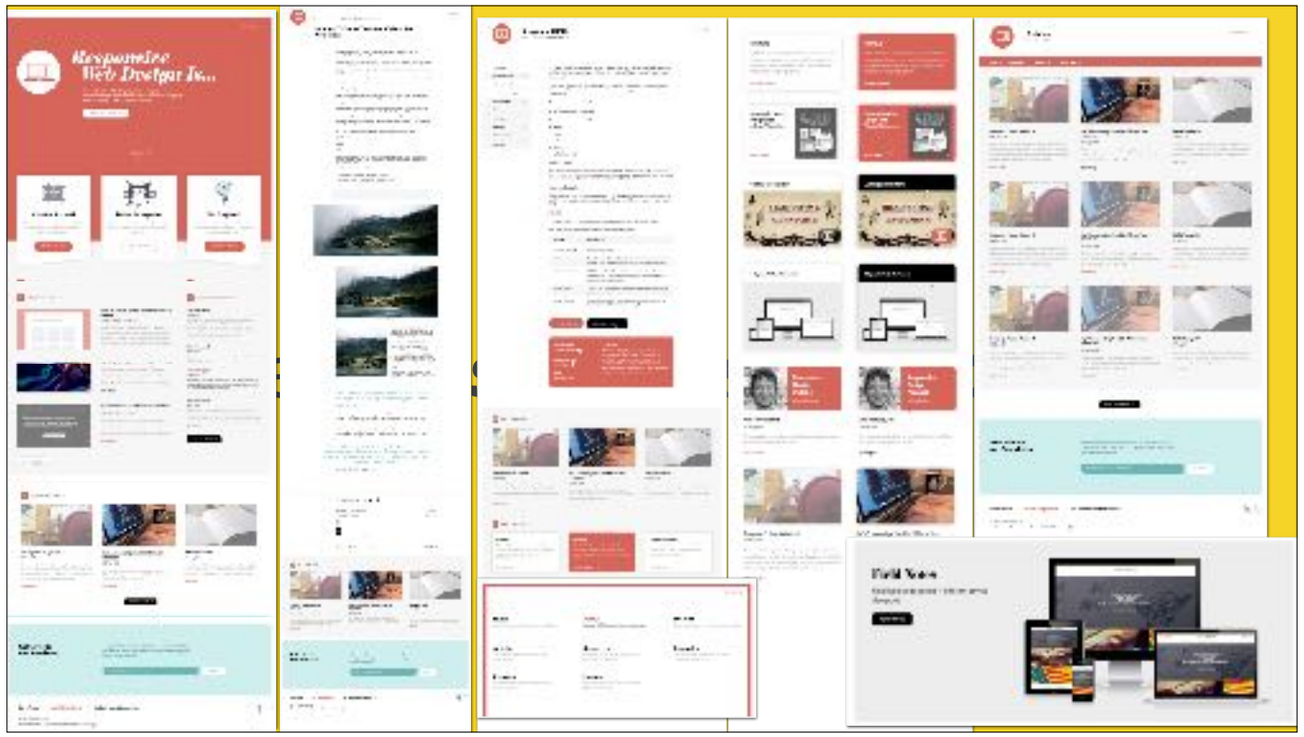
Adobe After Effects for interactive design. You can do this in Invision as well.

Designing in the browser is a great end goal, but don’t worry because it’s probably only about 5% of the world. We’ll see people on stage saying that this is the best way to go, but the complexity of it.

======
What ever works for you!
======

Pattern Libraries

Pattern Libraries/Style Guides

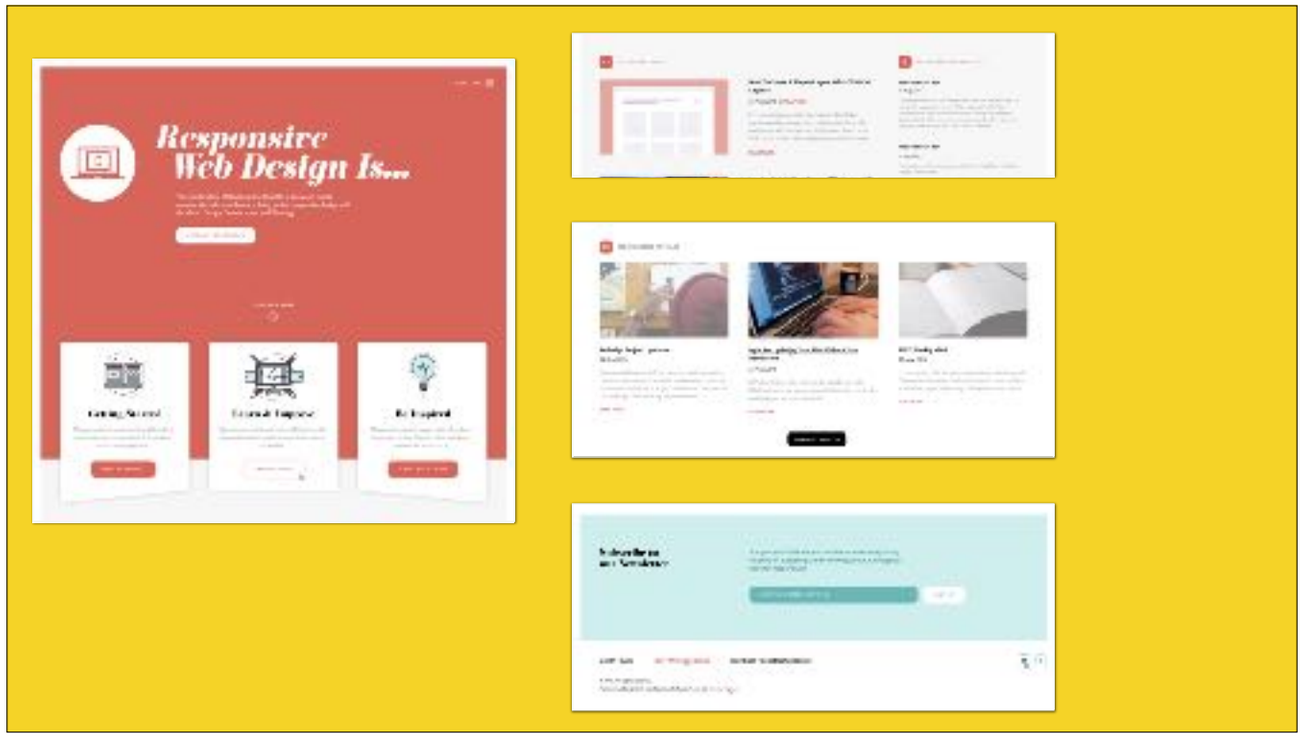


Ethan Marcotte talked a lot about getting patterns from designs so I wanted to also share the process that I'm currently going through with the team that is redesigning responsivedesign.is

We took all the designs that were done, dropped them into Trello and began creating tasks for each of those designs.

First step was to go through each of the designs, highlighting the elements that were consistent and then numbering them as modules.

These are then created as sections with a series of todo items which are worked through programatically.

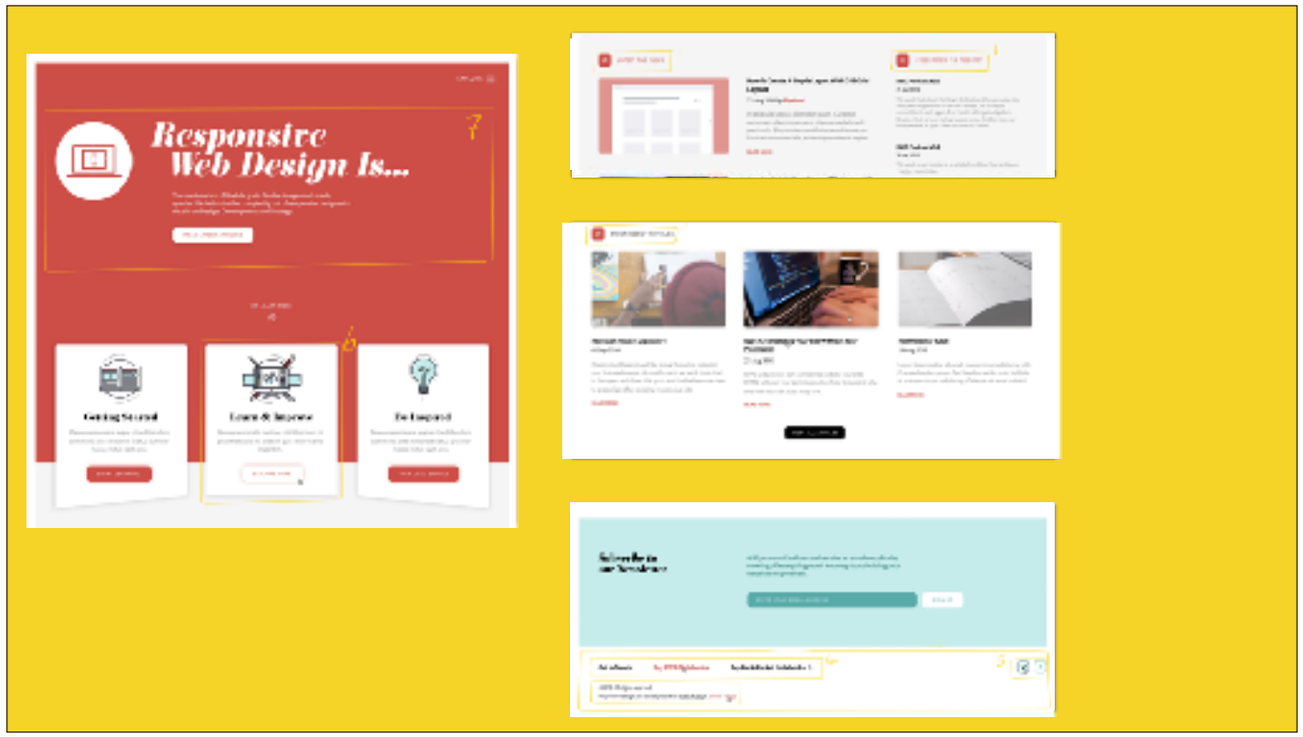


Ethan Marcotte talked a lot about getting patterns from designs so I wanted to also share the process that I'm currently going through with the team that is redesigning responsivedesign.is

We took all the designs that were done, dropped them into Trello and began creating tasks for each of those designs.

First step was to go through each of the designs, highlighting the elements that were consistent and then numbering them as modules.

These are then created as sections with a series of todo items which are worked through programatically.

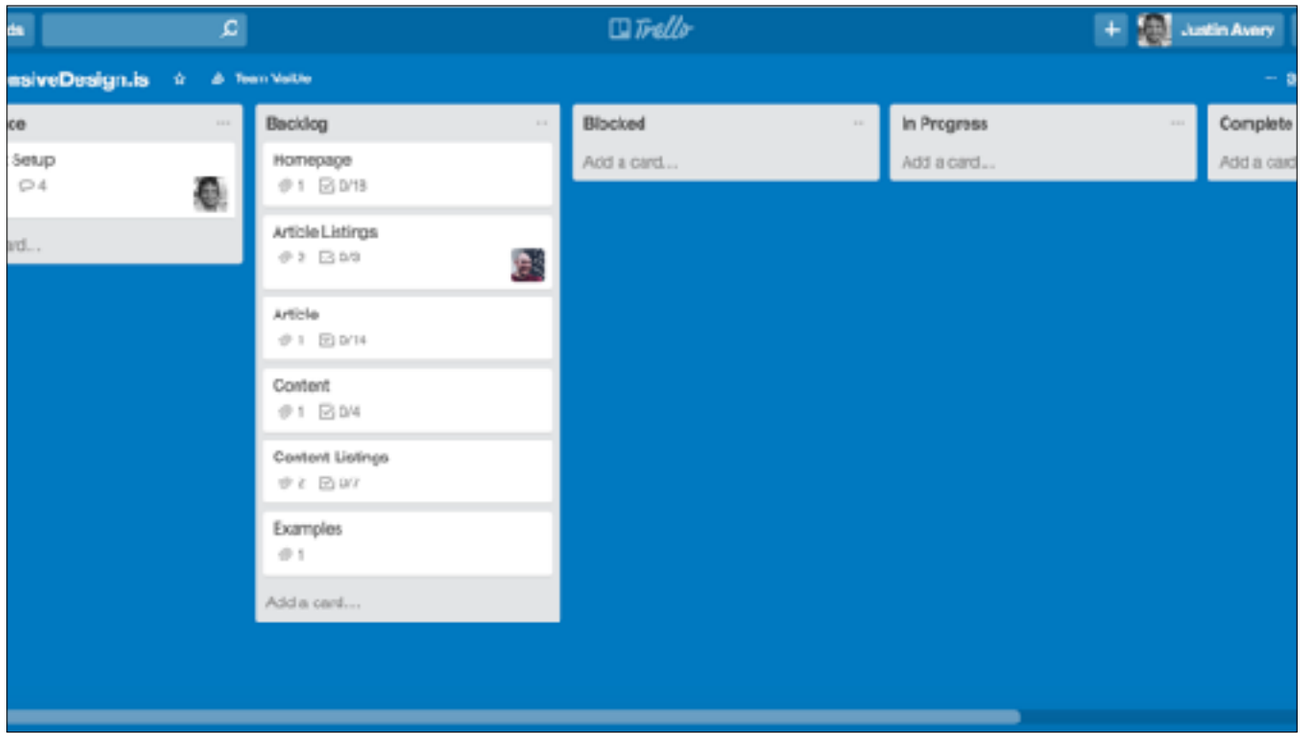


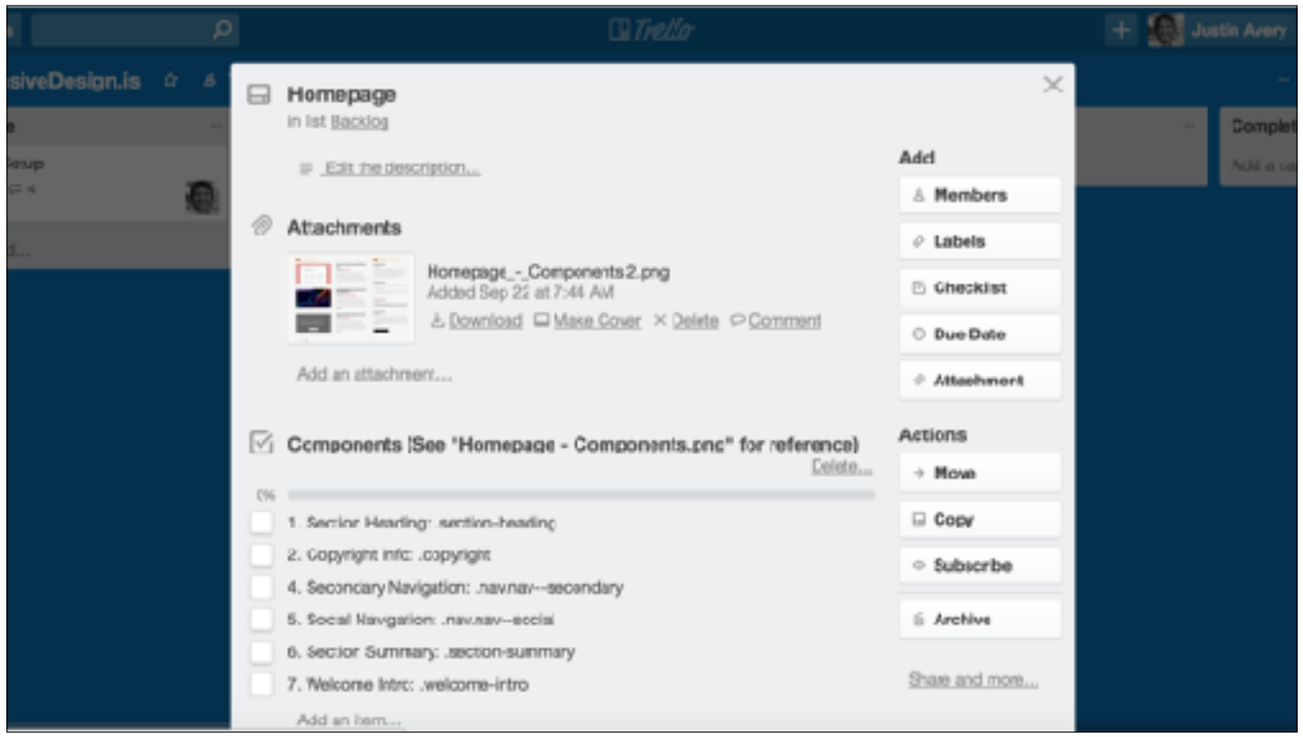
Ethan Marcotte talked a lot about getting patterns from designs so I wanted to also share the process that I'm currently going through with the team that is redesigning responsivedesign.is

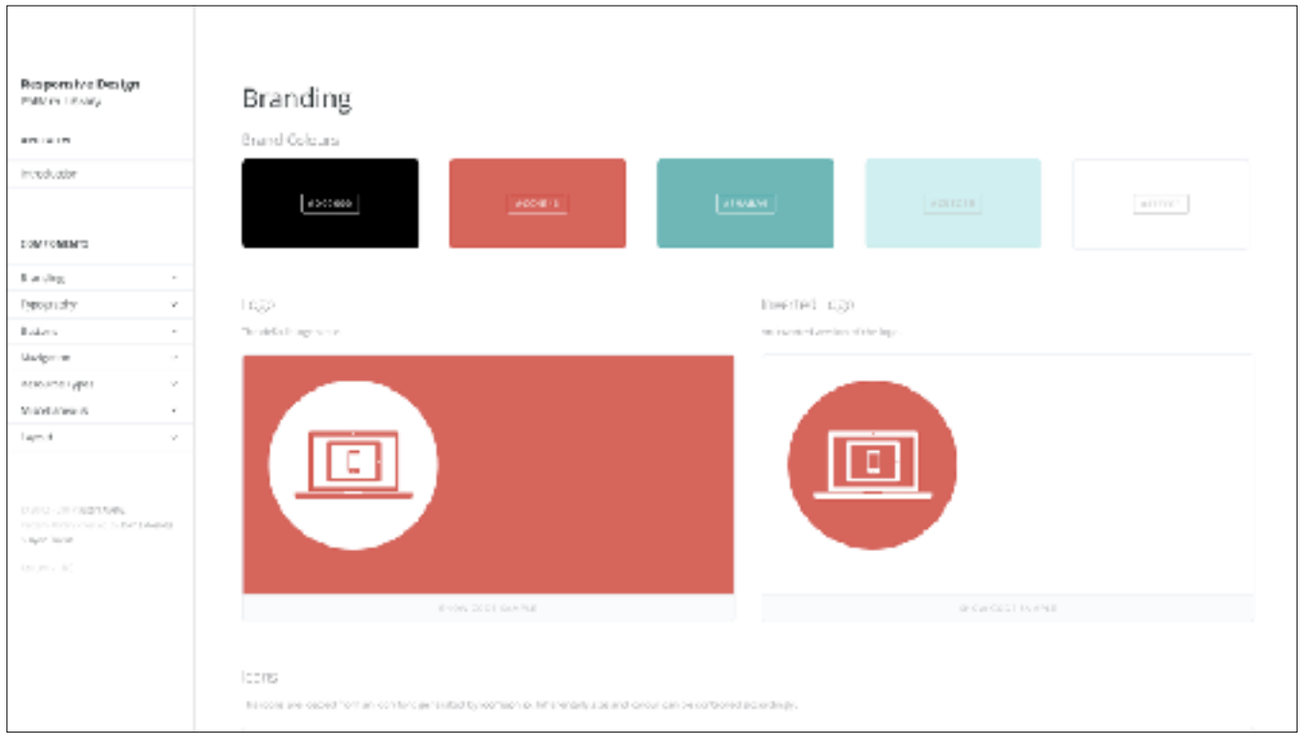
We took all the designs that were done, dropped them into Trello and began creating tasks for each of those designs.

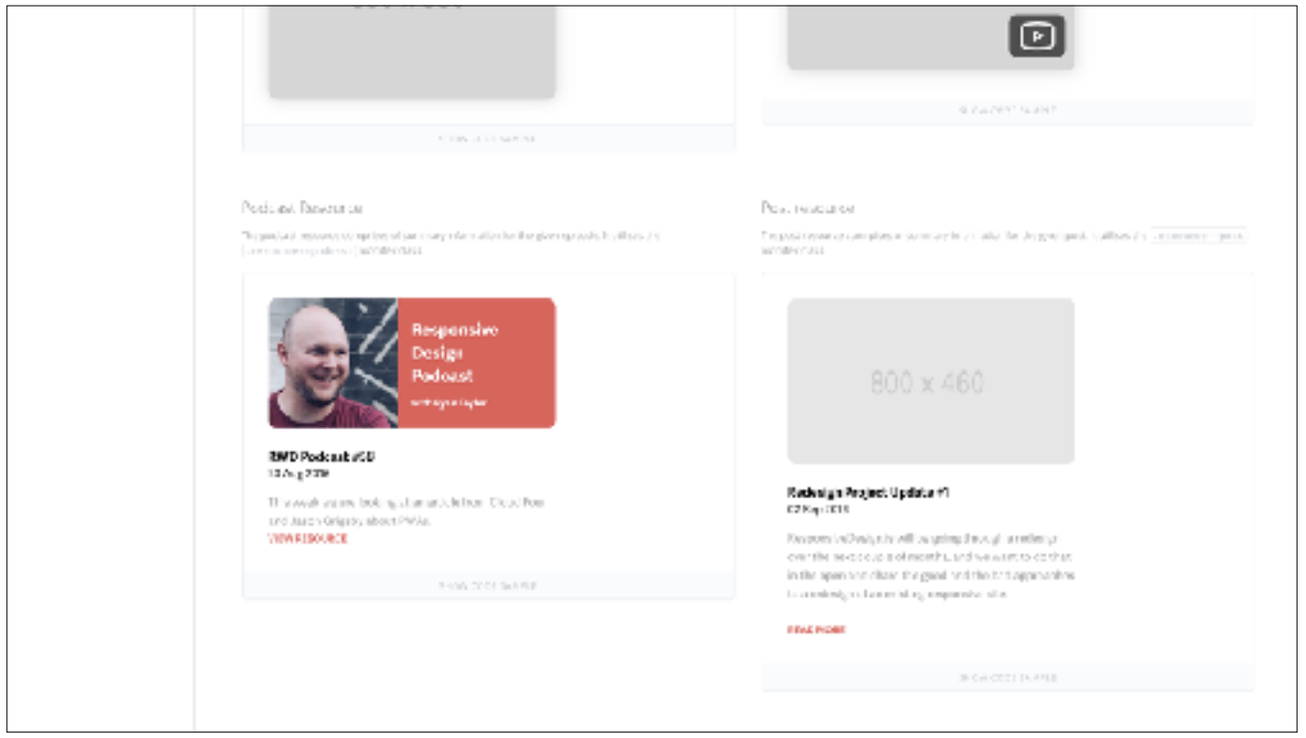
First step was to go through each of the designs, highlighting the elements that were consistent and then numbering them as modules.

These are then created as sections with a series of todo items which are worked through programatically.

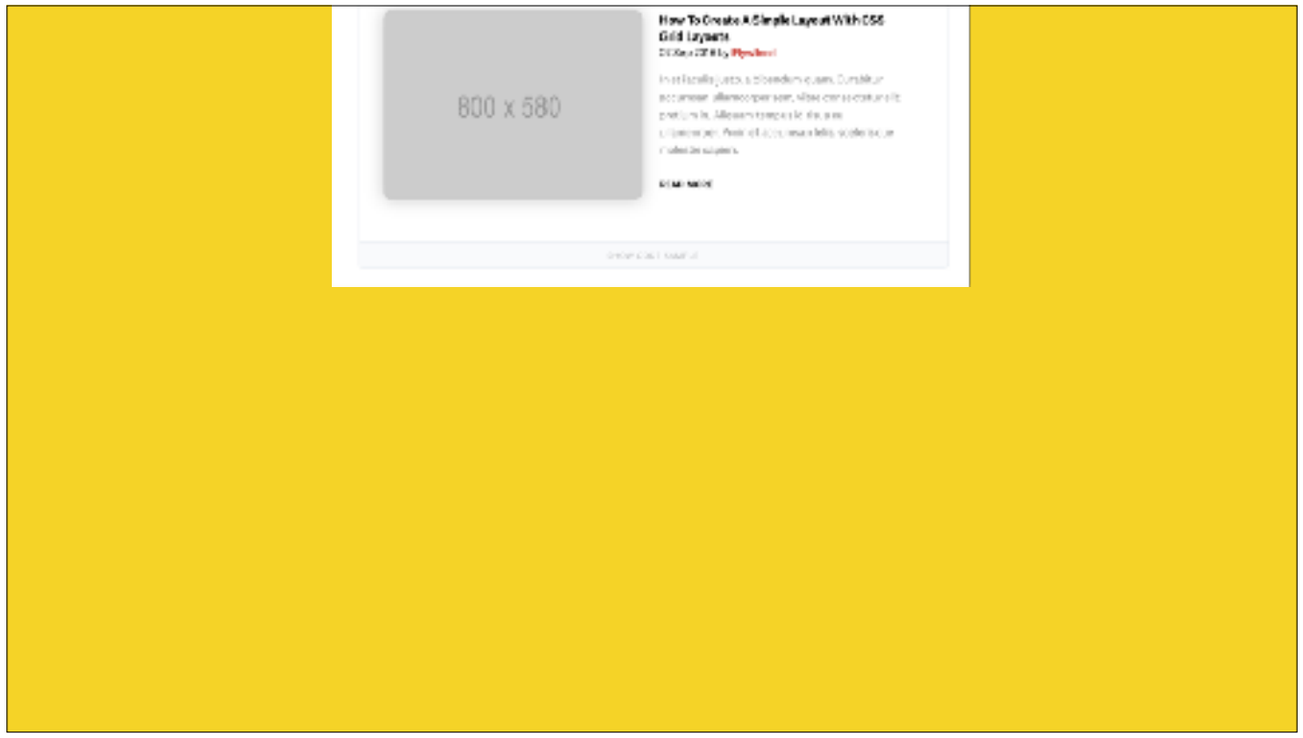

Trello
Justin Avery
Backlog
Homepage
Article Listings
Article
Content
Content Listings
Examples
Add a card...
Blocked
Add a card...
In Progress
Add a card...
Complete


Trello
Justin Avery
Homepage
in list Backlog
Edit the description...
Attachments
Homepage_-_Components2.png
Added Sep 22 at 7:44 AM
Download
Make Cover
Delete
Comment
Add an attachment...
Components (See "Homepage - Components.png" for reference)
Delete...
0%
1. Section Heading: .section-heading
2. Copyright Info: .copyright
4. Secondary Navigation: .nav.nav--secondary
5. Social Navigation: .nav.nav--social
6. Section Summary: .section-summary
7. Welcome Intro: .welcome-intro
Add an item...
Add
Members
Labels
Checklist
Due Date
Attachment
Actions
Move
Copy
Subscribe
Archive
Share and more...


Responsive Design
Introduction
Branding
Typography
Buttons
Badges
Alerts
Tables
Navigation
Branding

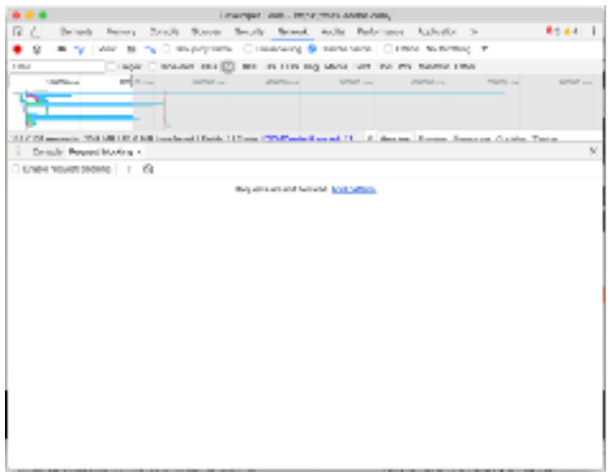

Responsive
Design
Podcast
800 x 460


Wide Post Resource
800 x 580
How To Create A Simple Layout With CSS Grid Layouts


800 x 580


800 x 580
How To Create A Simple Layout With CSS Grid Layouts

Performance Omissions



Chrome Dev tools has the ability to block certain resources from a page so that you can see how it works when your third parties fail.

Most importantly you can work on how to progressively enhance your site so that it doesn’t happen again.

What Does My Site Cost?

Find out how much it costs for someone to see your site on mobile networks around the world.

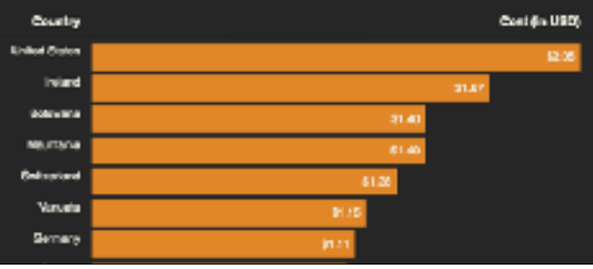
https://max.adobe.com weighs 20.96MB. Here's what that costs around the globe.

Want to make things better? Check out the full results and recommendations at WebPageTest.org.

Cost in USD | Cost in USD, PPP | Cost as % of GNI, PPP | Postpaid | Prepaid

Cost in USD (PREPAID DATA)

This is the cost of the site based on data from the ITU, without any adjustment for purchasing power or relative affordability. Prices were collected from the operator with the largest market share in the country using the least expensive plan with a (minimum) data allowance of 500 MB over (a minimum of) 30 days. Prices include taxes. Because these numbers are based on the least expensive plan, they are best case scenarios.



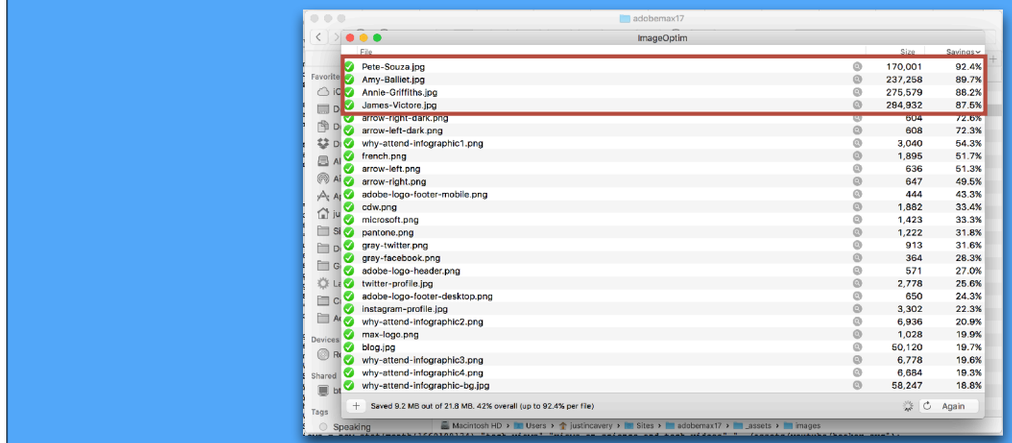
Rebuilding Pinterest pages for performance resulted in a 40% decrease in wait time, a 15% increase in SEO traffic and a 15% increase in conversion rate to signup.

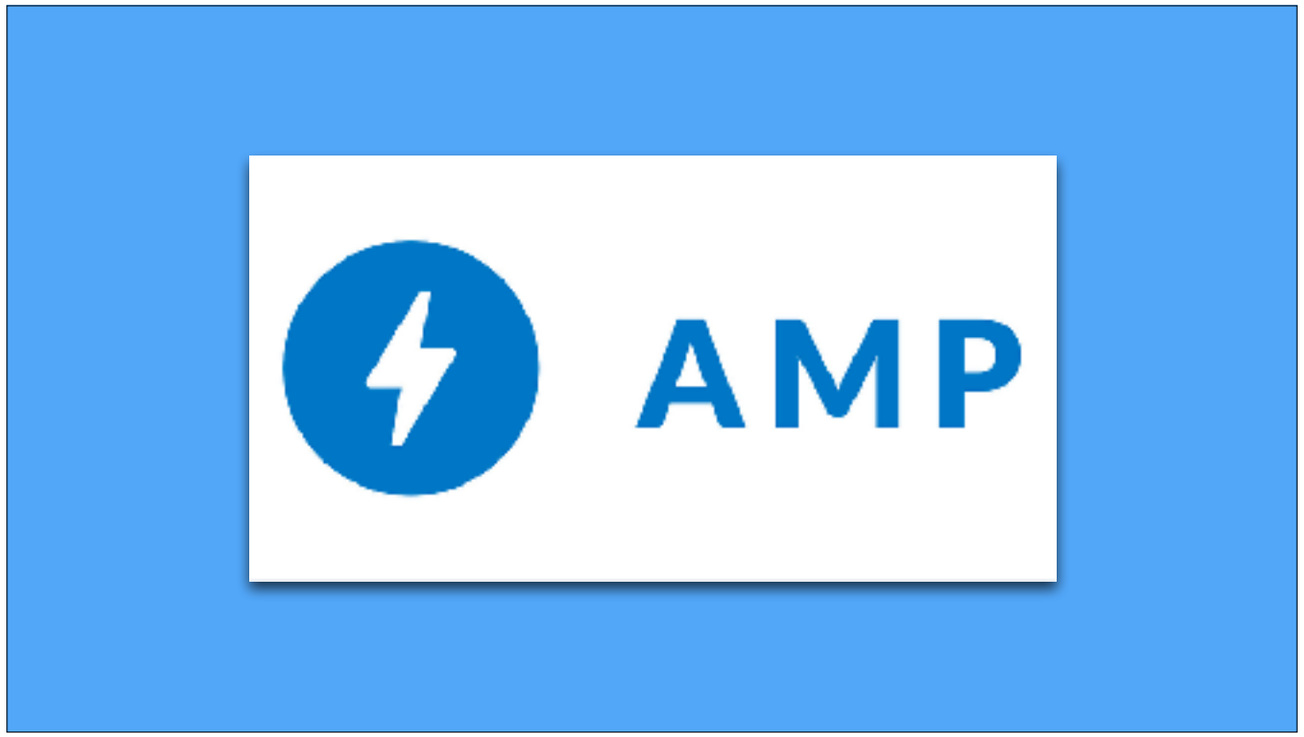
https://wpostats.com/2017/03/10/pinterest-seo.html

Rebuilding Pinterest pages for performance resulted in a 40% decrease in wait time, a 15% increase in SEO traffic and a 15% increase in conversion rate to signup.

https://twitter.com/addyosmani/status/916391453076602880

```
<!-- if possible: -->
<!-- the decode for this image may be deferred 🌕 -->
<img async=on src="space-cats.jpg">

<!-- the decode for this image should not be deferred ✋ -->
<img async=off src="space-dogs.jpg">

<!-- the browser is free to do what it feels is best for the user 🤔 -->
<img src="space-pizza.jpg">
```


adobemax17
ImageOptim
File
Size
Savings
Pete-Souza.jpg 170,001 92.4%
Amy-Balliet.jpg 237,258 89.7%
Annie-Griffiths.jpg 275,579 88.2%
James-Victore.jpg 284,932 87.5%
arrow-right-dark.png 604 72.6%
arrow-left-dark.png 608 72.3%
why-attend-infographic1.png 3,040 54.3%
french.png 1,895 51.7%
arrow-left.png 636 51.3%
arrow-right.png 647 49.5%
adobe-logo-footer-mobile.png 444 43.3%
cdw.png 1,882 33.4%
microsoft.png 1,423 33.3%
pantone.png 1,222 31.8%
gray-twitter.png 913 31.6%
gray-facebook.png 364 28.3%
adobe-logo-header.png 571 27.0%
twitter-profile.jpg 2,778 25.6%
adobe-logo-footer-desktop.png 650 24.3%
instagram-profile.jpg 3,302 22.3%
why-attend-infographic2.png 6,936 20.9%
max-logo.png 1,028 19.9%
blog.jpg 50,120 19.7%
why-attend-infographic3.png 6,778 19.6%
why-attend-infographic4.png 6,684 19.3%
why-attend-infographic-bg.jpg 58,247 18.8%
Saved 9.2 MB out of 21.8 MB. 42% overall (up to 92.4% per file)
Again
Macintosh HD › Users › justincavery › Sites › adobemax17 › _assets › images
Speaking



I haven’t mentioned AMP because it’s a quick fix that I fundamentally disagree with.

It’s a throwback to two different codebases for desktop and mobile and Google dangle the enormous carrot of being in the top carousel on search results as a way to move publishers in that direction.

I would encourage you to take ownership of your own site and make it fast on it’s own. People won’t always get the AMP version of the site if you have it, but they can always get the web version…. so make THAT fast and everyone wins.

1. TypeKit Code - the async version (or Font-Face Observer with <link>
2. Added **`preload=none`** on the video elements because they were downloading for some reason.
3. Add Critical CSS
4. Applied responsive images solution (automate it)
5. Lazy load images below the fold….
6. Put on Cloudflare with https = HTTP2
7. Added a SVG image for crisp images across all viewports
8. Add a service worker with offline page (we'll get to that soon)

TypeKit Code - the async version
Added Preload=None on the video elements because they were downloading for some reason.
Add Critical CSS
Applied responsive images solution
Lazy load images below the fold….
Put on Cloudflare with https = HTTP2
Added a SVG image for crisp images across all viewports
Add a service worker with offline page (we'll get to that soon)

- https://twitter.com/aweil/status/788783790554427392

Performance is really interesting because we often align it with a monetary figure.

We increased the sales of widgets by 200% for every 0.2s we shaved off the load time.

These are great because we understand those metrics. We can also use them to convince company's to spend money with us to deliver them a more successful business through faster websites.

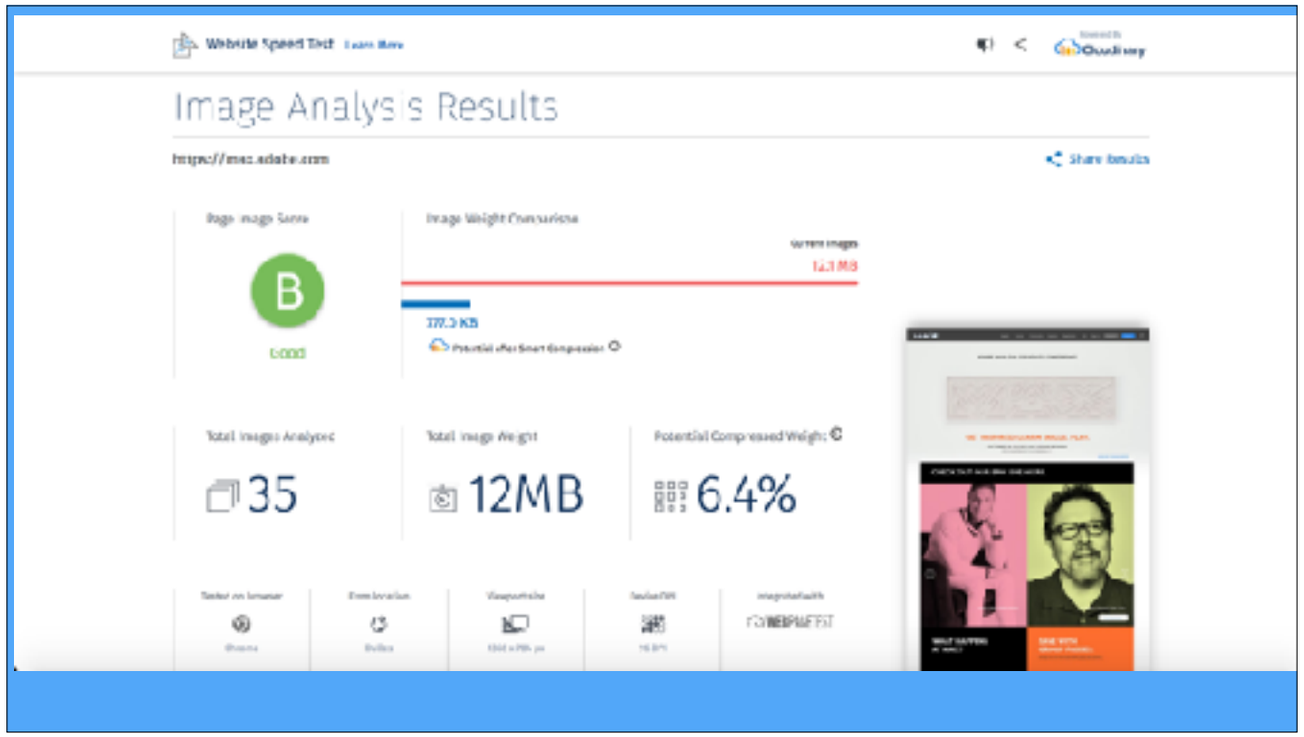
YouTube player.
Dozens of requests and 1.2mb on it's own
98kb
2 minutes to download instead of 20.
http://blog.chriszacharias.com/page-weight-matters

Load time

vs

Perceived Load Time

Story of the baggage coming off the plane and making people walk to make it seem as though it was coming off faster.



https://webspeedtest.cloudinary.com/results/170928_EQ_322a9dce79dd855639a079ea59d0a829

Accelerated Mobile Pages (AMP)

I’ve had a lot to say about AMP since it was first released by Google in <insert date here>.

For those of you that are unfamiliar with AMP lets take a look at what it is.

- AMP results appear first in the top stories
- Content can be swiped fast.
- Watch videos, has advertising
- closing returns to the results
- If you have two pages, a responsive and an AMP version, the AMP version will be shown to mobile users
- Your content is on Googles servers, not your own.

AMP Success Stories

- **Washington Post** — **23% increase** in mobile search users who return within 7 days
- **Slate** — **44% increase** in monthly unique visitors and a **73% increase** in visits per monthly unique visitor
- **Gizmodo** — **80%** of Gizmodo's traffic from AMP pages is **new traffic**, **50% increase** in **impressions**
- **Wired — 25%** increase in **click through rates** from search results, with **CTR** on **ads** in **AMP stories up by 63%**.
- **Relay Media** — in the last 30 days alone has **converted over 2.5 million AMP** pages for publishers like The Daily Dot, Hearst Television and The Miami Herald which says **mobile users** who start with an AMP article **spend 10% more time** than those who land on regular mobile pages.

What AMP should be used for

Now I wanted to tell you about this because it’s certainly something which will be more prevalent come 2017, and something that if you aren’t already taking advantage of you certainly should be.

My advice is two fold. I think you should use it now because there are so many free wins once the template is set up.
The second part of my advice is that you should use the AMP project as a check list for best practice performance that you should follow for your regular build.

Remember that not everyone uses Google for searches and the only time people will get your AMP page is going to be through those results. Arguably this is part of Googles plan for world domination to get more people using the search engine for faster results and pages, but then also push people through their ad model.

Implementing AMP

- Lazy Loading
- Extensive use of Preconnect
- Perfecting of lazy loaded resources
- All async Javascript
- Inline style sheets
- Zero http requests block font downloads
- Instant loading through pre rendering
- Prerendering only resources above the fold
- Not prerendering things that might be expensive in terms of CPU
- Intelligent resource prioritisation
- Remove document layout from resource downloads
- Maximum size for a style sheet

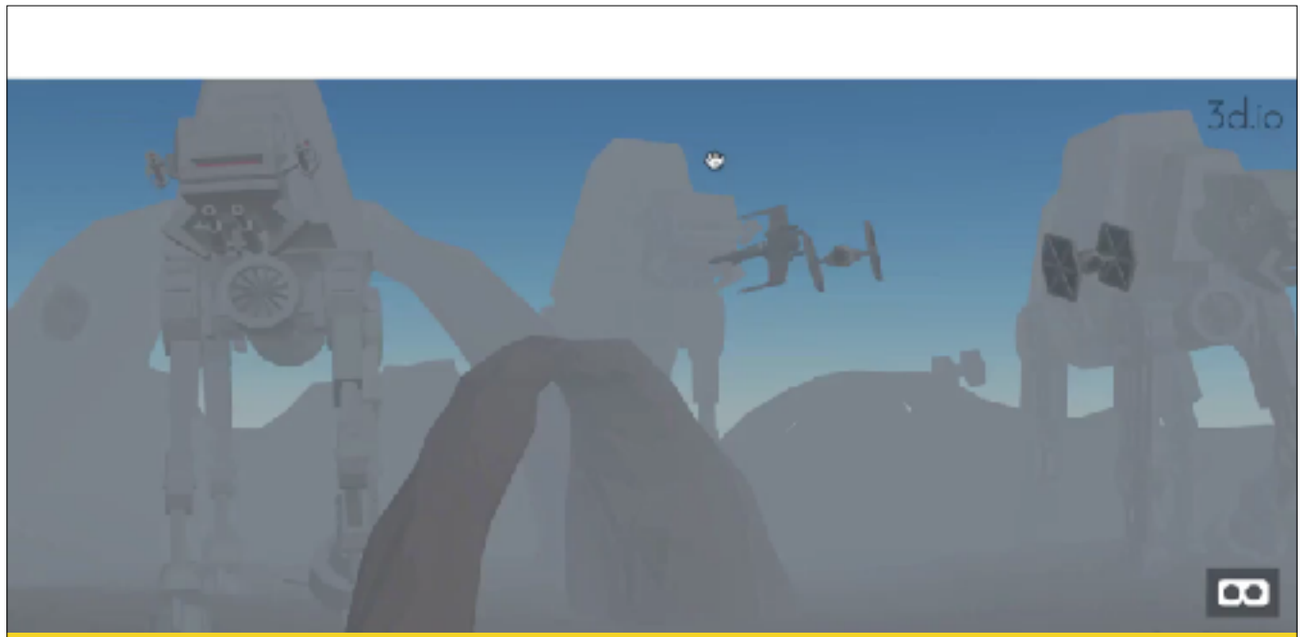
Add lazy loading to images below the fold.

But also add that you can do picture to load WEBP images when they are supported

Run Lighthouse test on it and start working on those elements

If we're going to be talking about responsive design for 2018 it would be remiss of me to mention the £1000 elephant in the room.

New Stuff Omissions



https://twitter.com/TomasPolach/status/905966569238523905

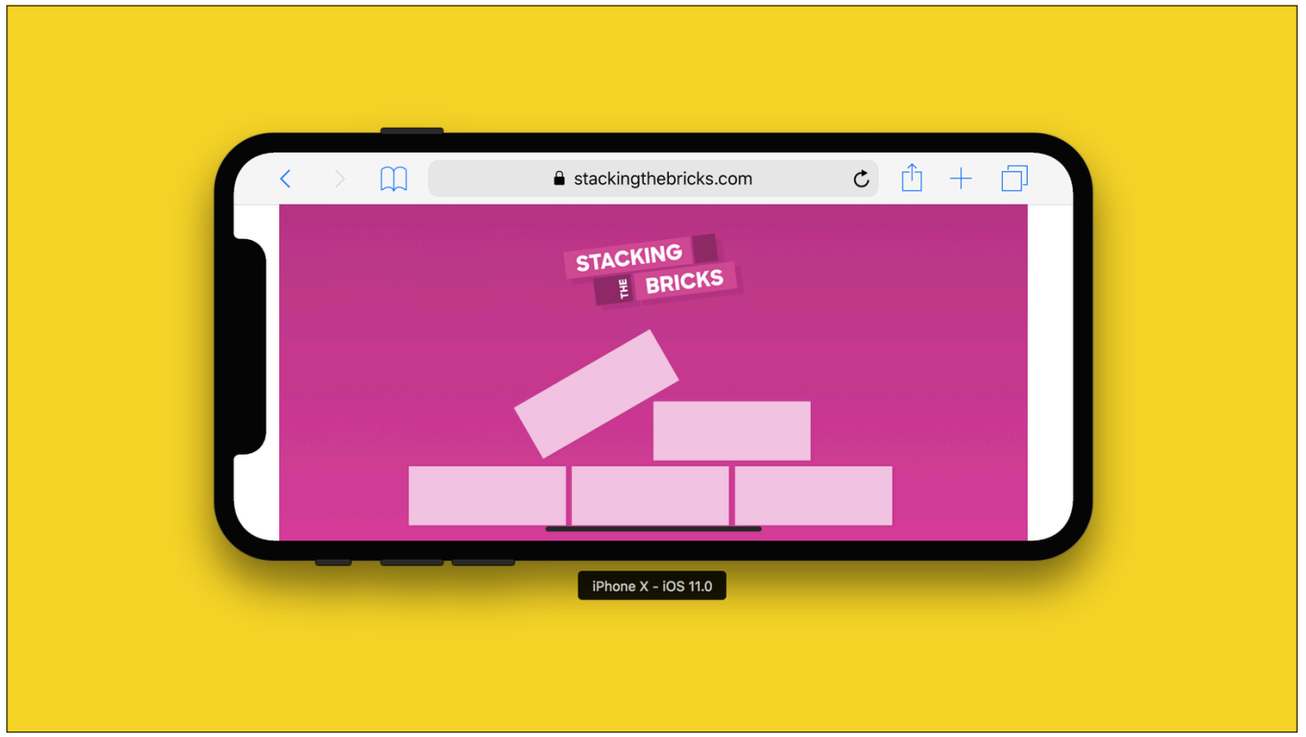
https://twitter.com/TomasPolach/status/905966569238523905

GitHub -

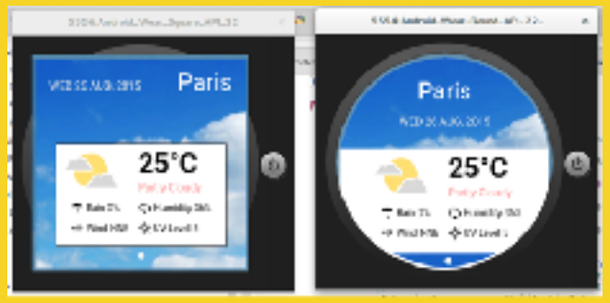

Mountain

https://twitter.com/jerome_etienne/status/893217730517749760

So how do you create a content for VR but in a progressively enhanced way?


stackingthebricks.com
STACKING
THE
BRICKS
iPhone X - iOS 11.0

This is how to get **around** the issue of the notch



LG Watches.

@prefers-reduced-motion:

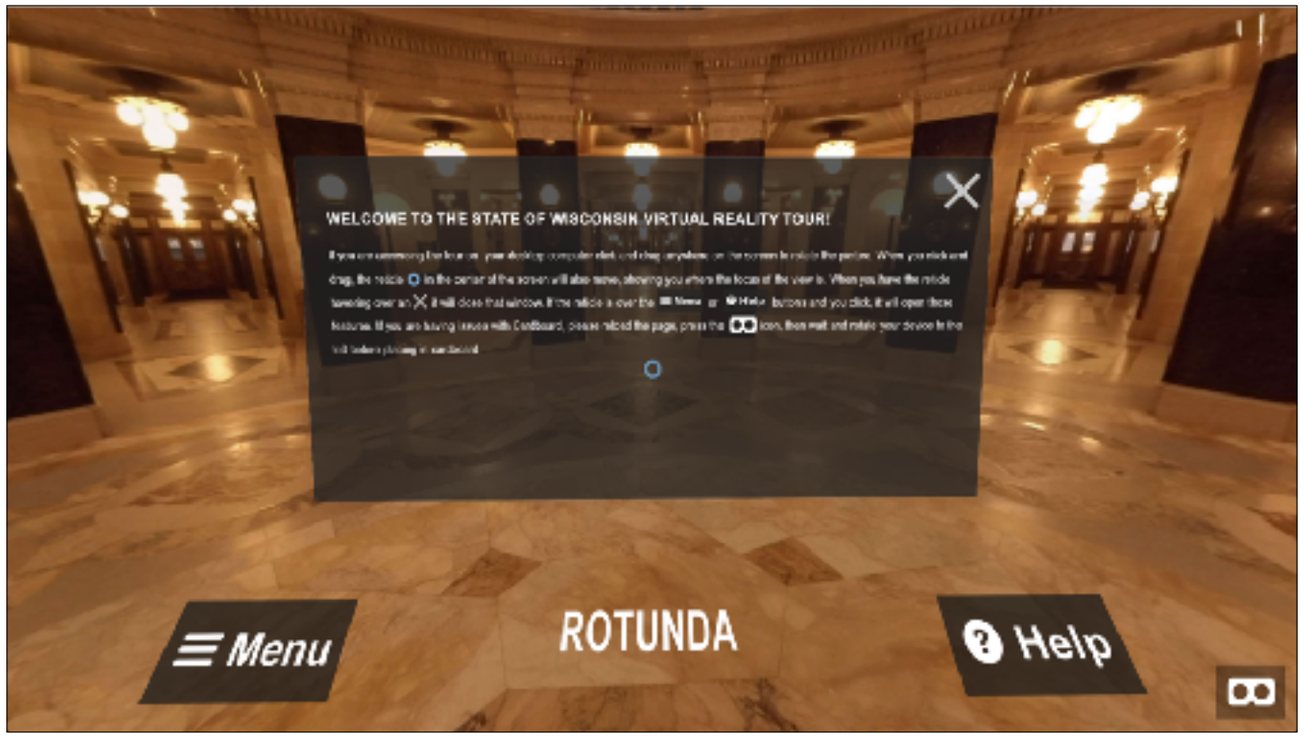
reduce | no-preference

Prefers-Reduced Motion

@prefers-reduced-motion: reduce | no-preference

```
@media (@prefers-reduced-motion: reduce) {
  * { transition: unset; }
}
```

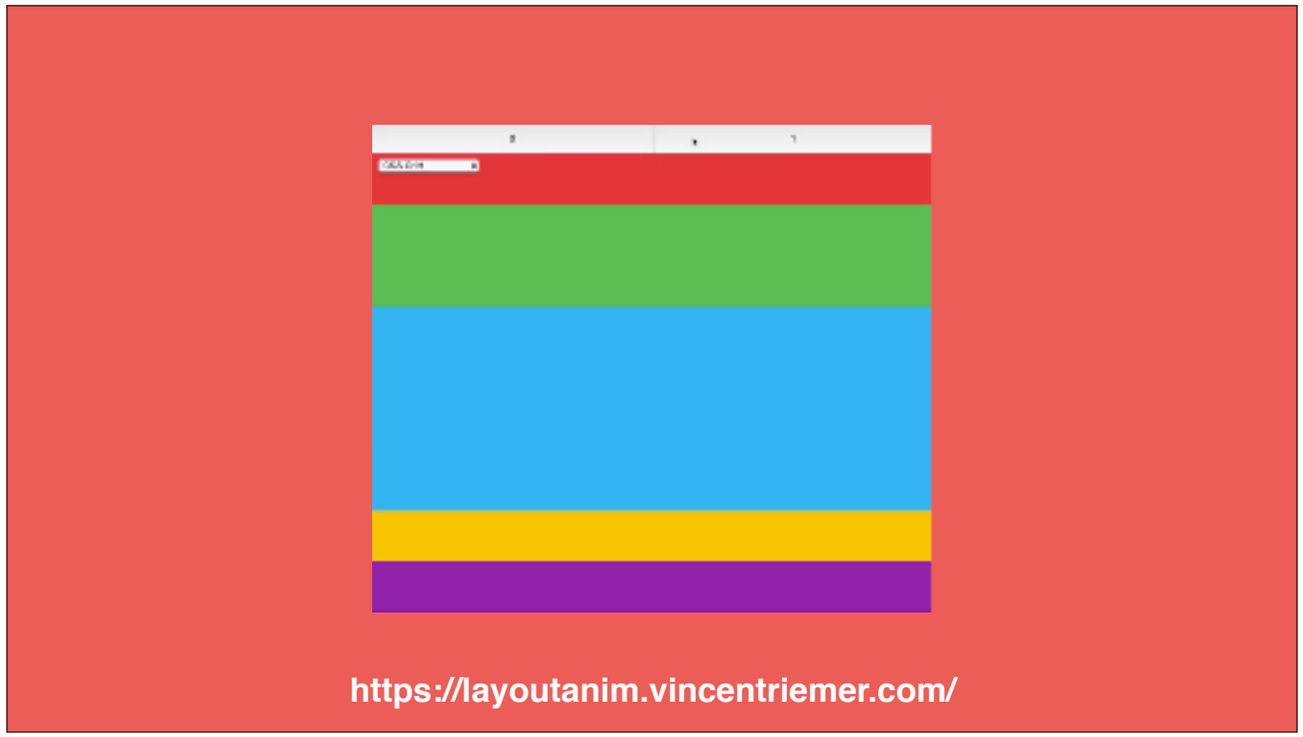
If someone prefers reduced motion, a setting that they can configure on their device, then you can



CHANGE THIS TO A LIVE VIDEO

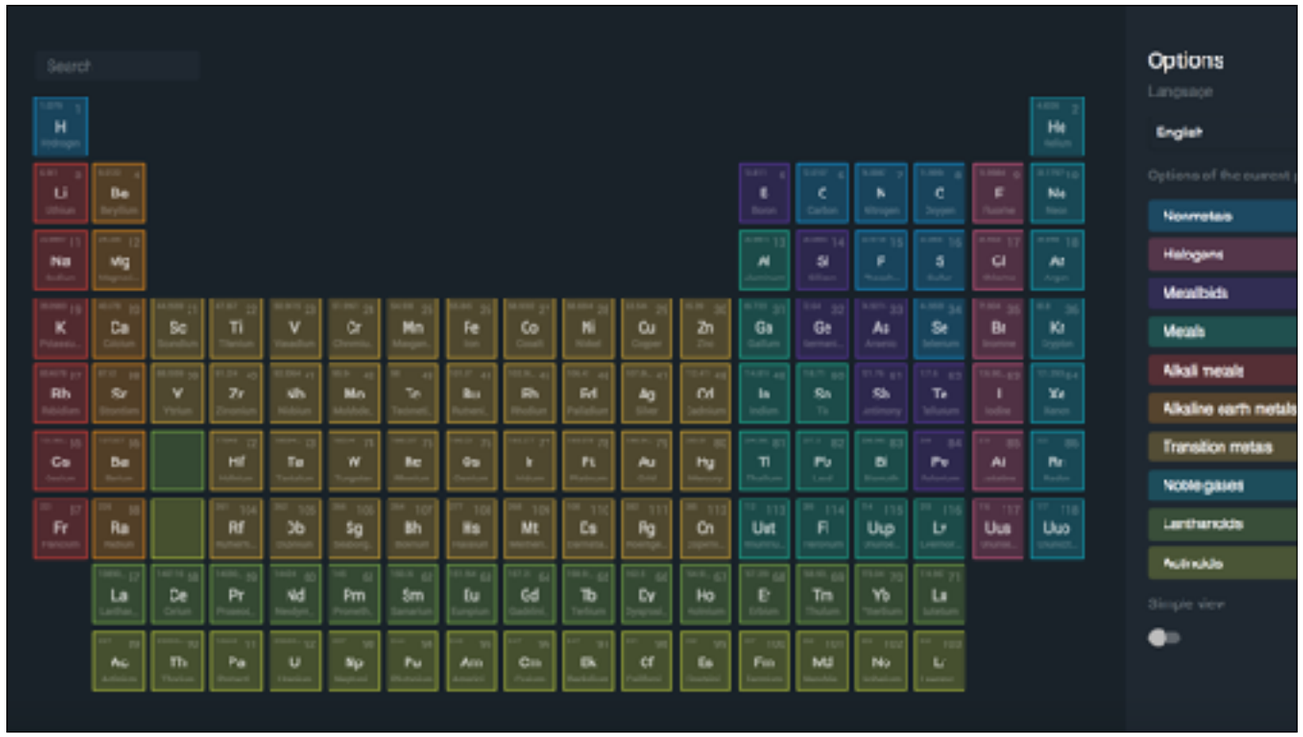
https://capitol.wisconsin.gov/

CSS Grid Layout Omissions



This sadly is not possible.

https://twitter.com/vincentriemer/status/916334249770397696


Search
Options
Language
English
Nonmetals
Halogens
Metalloids
Metals
Alkali metals
Alkaline earth metals
Transition metals
Noble gases
Lanthanoids
Actinoids


THE EXPERIMENTAL LAYOUT LAB
OF JEN SIMMONS
WORKSHOP EXAMPLES
LAYOUT LAB

THE EXPERIMENTAL LAYOUT LAB OF JEN SIMMONS

Check back often. This will keep changing.

Includes examples for
Revolutionize Your Page: Real Art Direction on the Web
Fragmenting Our Layouts
Modern Layouts: Getting Out of Our Ruts

Follow @jensimmons on Twitter for more as it happens.
Sign up for Layout Land, a place for us to show off new ideas for layout.

Workings Examples

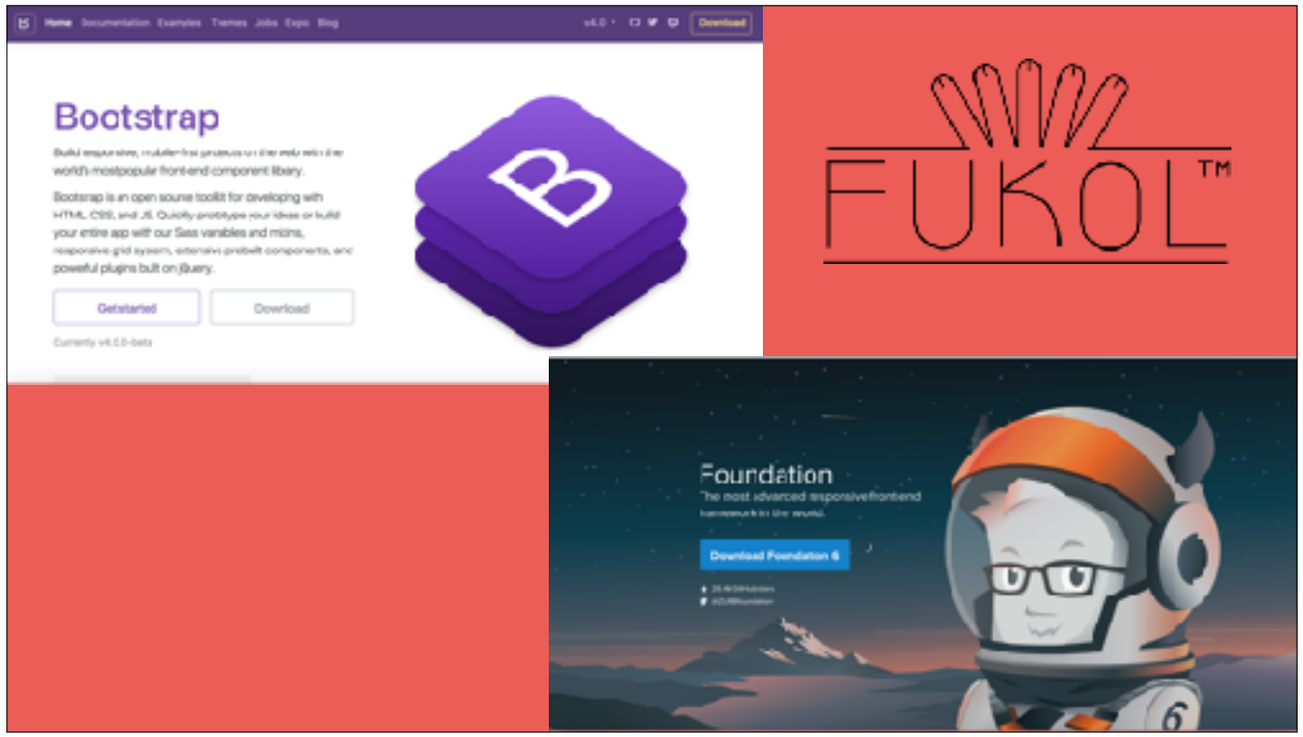
The header of this page is much more interesting in a browser that supports CSS Grid. Try Firefox Nightly, Safari Technical Preview, or Chrome Canary (where you need to turn Grid on manually). (It doesn't quite work in Chrome 58 because not enough of Grid is implemented.)

```css
* { box-sizing: border-box; }

.grid > div {
  float: left;
  width: 33.333%;
}

@supports (display: grid) {
  .grid > div {
    width: auto;
  }
}

.grid {
  display: grid;
  grid-gap: 10px;
  grid-template-columns: repeat(3, 1fr);
  width: 500px;
}
```

Well we could still set float properties on some elements (see the top)

Then when we **set display:grid** to the container the child elements lose their float properties as if it wasn't there.

One issue is that if grid is supported it will set the **DIV to be 33%** the **width** of the **grid item**, so you include **@supports (display: grid)** which is saying to the browser, hey if you know what **display: grid** means, then do the stuff here, otherwise just ignore it.



There was a swagger of flex box based grids that came out but my favourite BY FAR was the grid layout that fit it a tweet, developed by the ever tongue in cheek Heydon Pickering called Fukol grids.

```css
.flex-grid {
    display: flex; /* 1 */
    flex-wrap: wrap; /* 2 */
    margin: -0.5em; /* 5 (edit me!) */
  }

  .flex-grid > * {
    flex: 1 0 5em; /* 3 (edit me!) */
    margin: 0.5em; /* 4 (edit me!) */
  }
```


1. Fukol™ is a Flexbox based grid system. Even Opera Mini supports Flexbox. Older user agents that don't support Flexbox ignore the display: flex declaration, degrading to a single column layout. No harm done.
2. This line determines how items are handled. The wrap value means items will start a new row if there's not enough room on the current one.
3. This is the 'element query' part. Instead of setting an arbitrary number of columns and using breakpoints, we decide roughly how wide we want the item to be (5em in the example — the flex basis) and make sure items can grow to use the available space (1) but not shrink (0). So only change the 5em value and leave 1 0 as it is.
4. This is for gutters. A 0.5em margin here means gutters of 1em (the margins double up).
5. This should always be a negative version of 4. It compensates for the margins created by the items. It makes sure the outside of the .fukol-grid container remains flush horizontally and no additional margin is added to the vertical flow.

Well we could still set float properties on some elements (see the top)

Then when we **set display:grid** to the container the child elements lose their float properties as if it wasn't there.

One issue is that if grid is supported it will set the **DIV to be 33%** the **width** of the **grid item**, so you include **@supports (display: grid)** which is saying to the browser, hey if you know what **display: grid** means, then do the stuff here, otherwise just ignore it.

Responsive Design in 2018


```
@supports (display: flex) {
  .flex {
    display: flex;
    flex-direction: row;
  }
  .flex > div {
    flex: 1 0 15em;
  }
}
```

Well we could still set float properties on some elements (see the top)

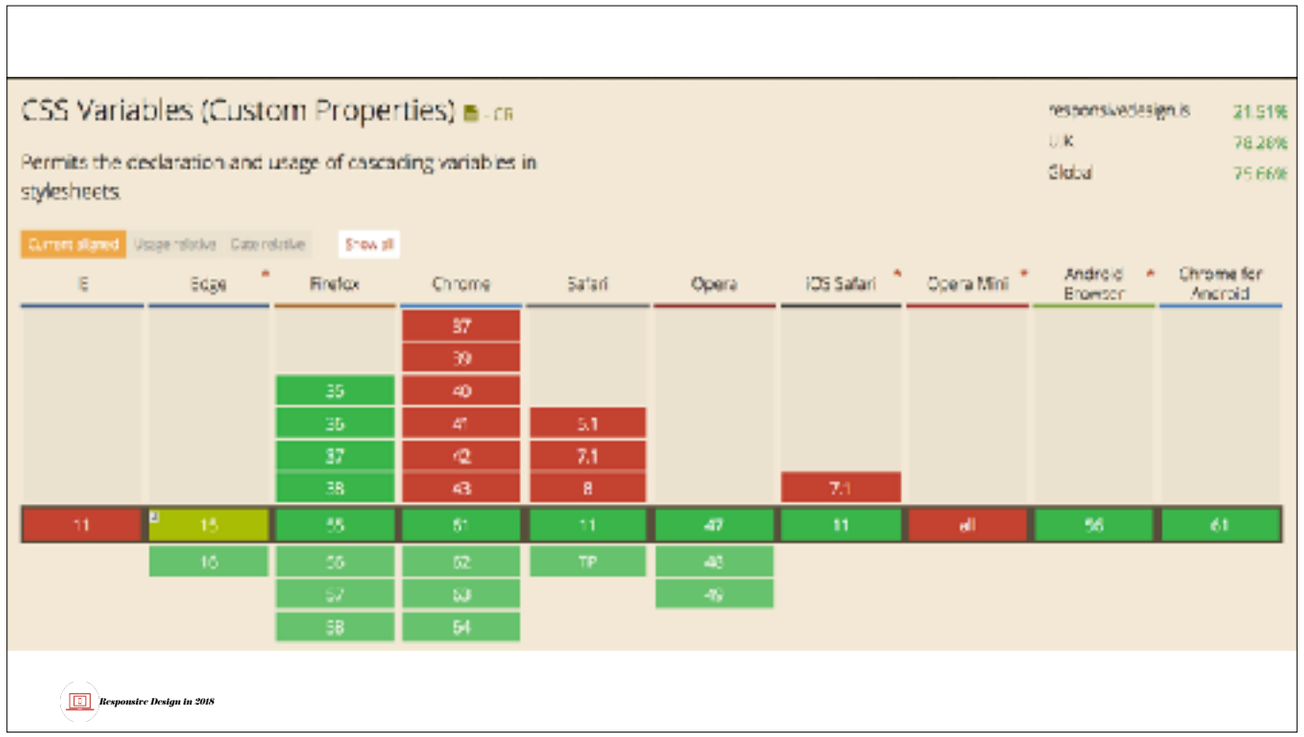
Then when we **set display:grid** to the container the child elements lose their float properties as if it wasn't there.

One issue is that if grid is supported it will set the **DIV to be 33%** the **width** of the **grid item**, so you include **@supports (display: grid)** which is saying to the browser, hey if you know what **display: grid** means, then do the stuff here, otherwise just ignore it.

https://twitter.com/wesbos/status/916313815175847937

CSS Custom Variables

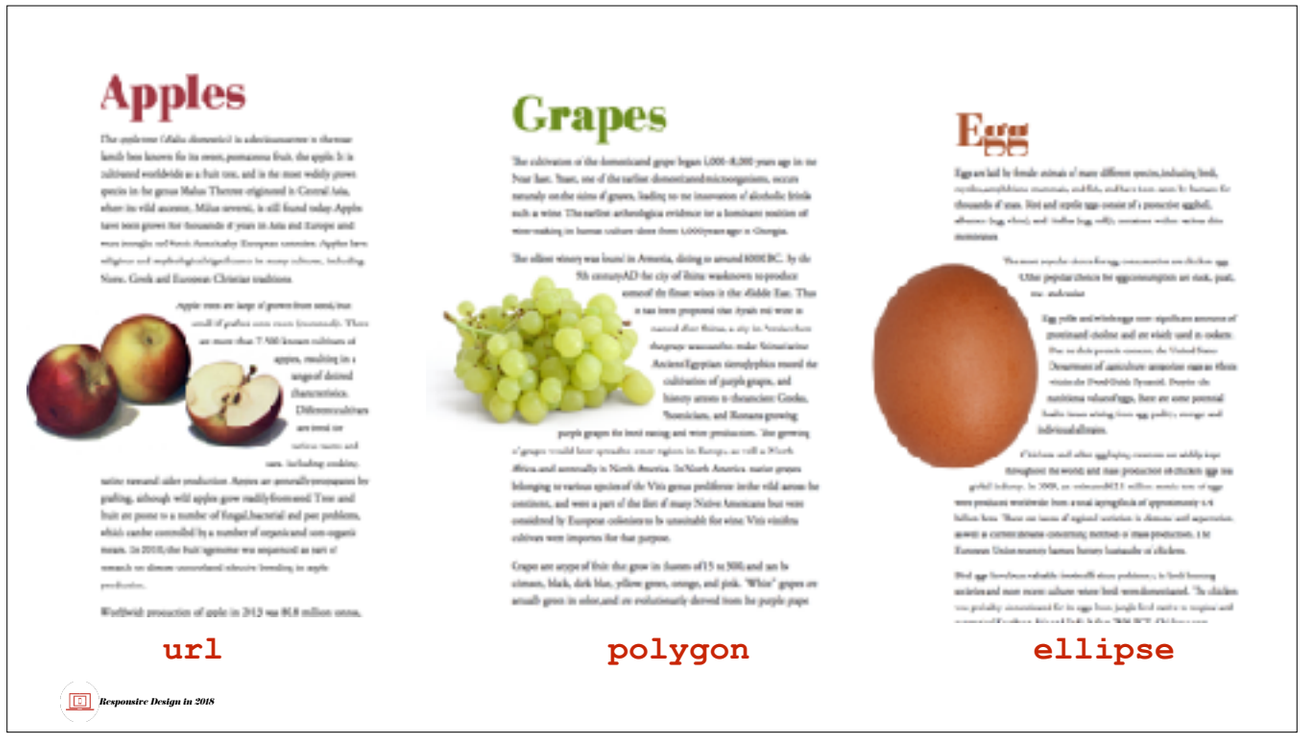

CSS Variables (Custom Properties) - CR
Permits the declaration and usage of cascading variables in stylesheets.
responsivedesign.is 21.51%
Global 66.39%
U.K. 75.19%
IE
Edge
Firefox
Chrome
Safari
Opera
iOS Safari
Opera Mini
Android Browser
Chrome for Android
37
39
35 40
36 41 5.1
37 42 7.1
38 43 8 7.1
11 14 49 54 10 41 10 all 53 53
60 55 TP 42
57 56 43
52 57
Responsive Design in 2018


CSS Variables (Custom Properties)
Permits the declaration and usage of cascading variables in stylesheets
Current aligned
Show all
IE
Edge
Firefox
Chrome
Safari
Opera
iOS Safari
Opera Mini
Android Browser
Chrome for Android
11
15
55
61
11
47
11
all
56
61
Responsive Design in 2018

CSS Image Shapes

Now we want out type to look amazing, but also interact with the imagery on our site in the ways that we've been perfecting for years on print.

CSS Image Shapes was originally introduced by Adobe and it is a wonderful way to start bringing some of the bedded in layout ideas of the past few hundred years of printing onto the web.



URL traces the text around the transparent PNG and matches the image.

Polygon allows you to draw the path yourself.

Ellipse and circle do similar things. LETS SEE HOW WE DO IT.

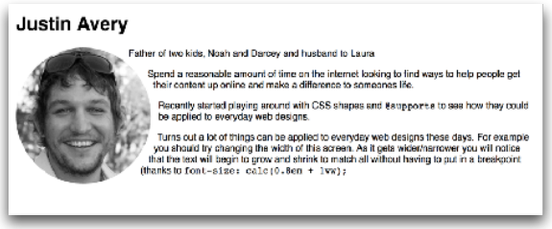
Justin Avery

Father of two kids, Noah and Darcey and husband to Laura

Spend a reasonable amount of time on the internet looking to find ways to help people get their content up online and make a difference to someones life.

Recently started playing around with CSS shapes and `@supports` to see how they could be applied to everyday web designs.

Turns out a lot of things can be applied to everyday web designs these days. For example you should try changing the width of this screen. As it gets wider/narrower you will notice that the text will begin to grow and shrink to match all without having to put in a breakpoint (thanks to `font-size: calc(0.8em + 1vw);`

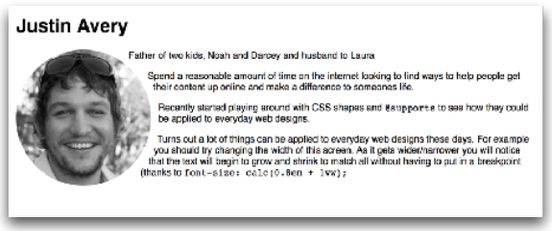



```
<div class="wrapper">
 <h1>Justin Avery</h1>
 <img src="me.jpg" alt="me">
 <p>blah blah</p>
 <p>more blah blah</p>
</div>
```


Standard HTML that you would normally expect

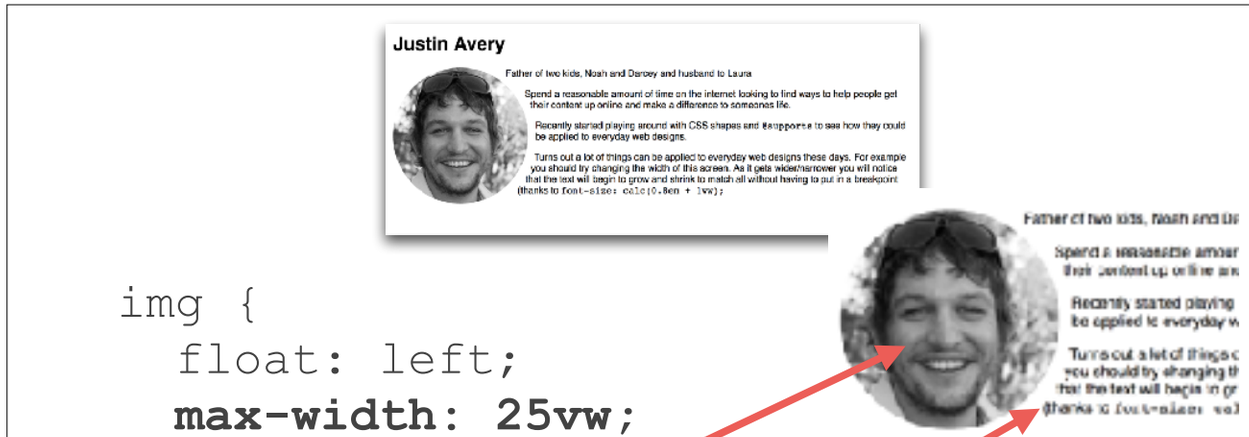

```
<div class="wrapper">
 <h1>Justin Avery</h1>
 <img src="me.jpg" alt="me">
 <p>blah blah</p>
 <p>more blah blah</p>
</div>
```

Note that me.jpg is actually a square image, nothing fancy was done in the making of this image in photoshop. The reason for this will become clear shortly.



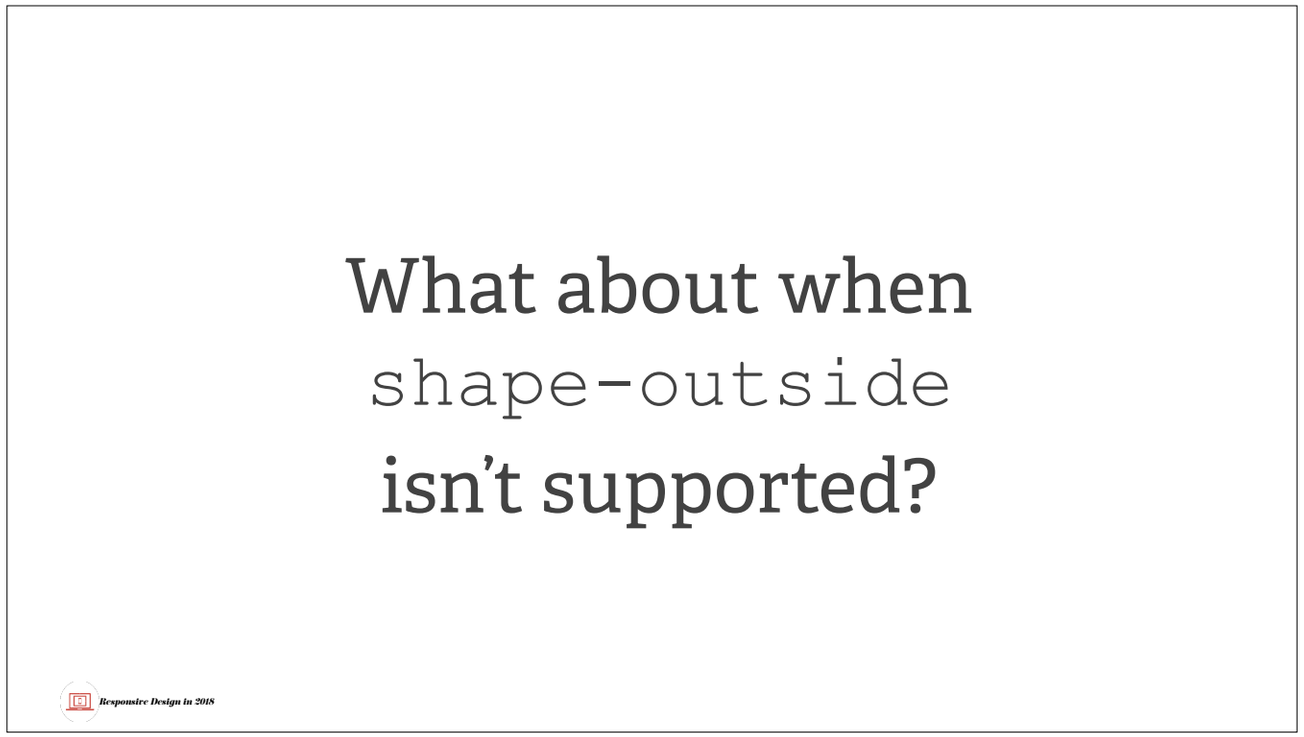

```
img {
   float: left;
   max-width: 25vw;
   border-radius: 50%;
   shape-outside: circle(50%);
}
```


The CSS for this…
float the image to the left, set the image to 100%, but here I’m restricting the width to only 25% of the viewport.




```
img {
   float: left;
   max-width: 25vw;
   border-radius: 50%;
   shape-outside: circle(50%);
}
```


I’m setting the border radius to 50% to give us the circle on the image, and then the shape outside to deliver the trace around the image.



But Justin, I can hear you at the back saying.

What about when things like Grid and Shape-Outside isn't supported?

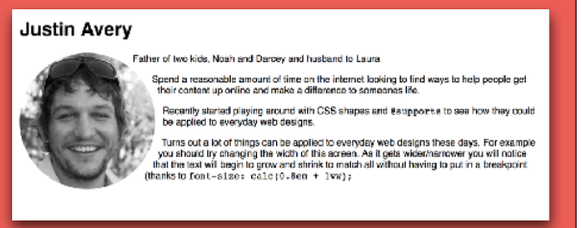
Well I'm glad you asked because we'll bring back out little friend, Progressive-Enhancement

@supports

`@supports` gives CSS the ability to perform a feature query

@Supports gives CSS the ability to perform a feature query.

This is great because you can do this within the CSS itself rather in the Javascript like we currently do, either with your own JS or something like Modernizr.




```
img {
float: left;
max-width: 25vw;
border-radius: 50%;
shape-outside: circle(50%);
}
```

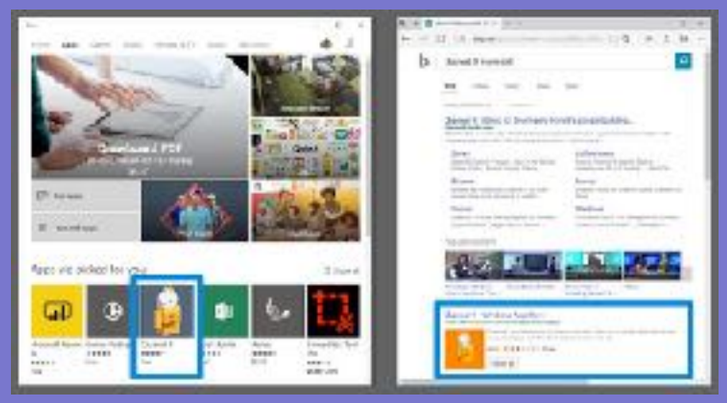
```
img {
  float: left;
  max-width: 25vw;
}
@supports(shape-outside:circle(50%)){
  img {
    border-radius: 50%;
    shape-outside: circle(50%);
  }}
```

Add in can I use @supports

Square image without support, and the regular image with the support.

http://2016.uxlondon.com/speakers

PWA Omissions



Lighthouse

Lighthouse is a tool developed by Google to allow us to test the performance of a site from a mobile app point of view.

Let’s see how the Adobe Max site pans out…

The original site had a lighthouse score of....

It gives us a break down of it’s PWA status, how it ranks on performance, Accessibility, and best practices.
Let’s see how the Adobe Max site pans out…

So the good news is that we're 45% of the way towards becoming a progressive web apps.

It gives us a break down of it's PWA status, how it ranks on performance, Accessibility, and best practices.
Let's see how the Adobe Max site pans out…

Grid Elements Removed

Responsive Design in 2018

This is a super simple example of website layout we might want.

On smaller viewports we stack the content, and on larger viewports we have the header and footer and full width and the main content and the said content next to each other.

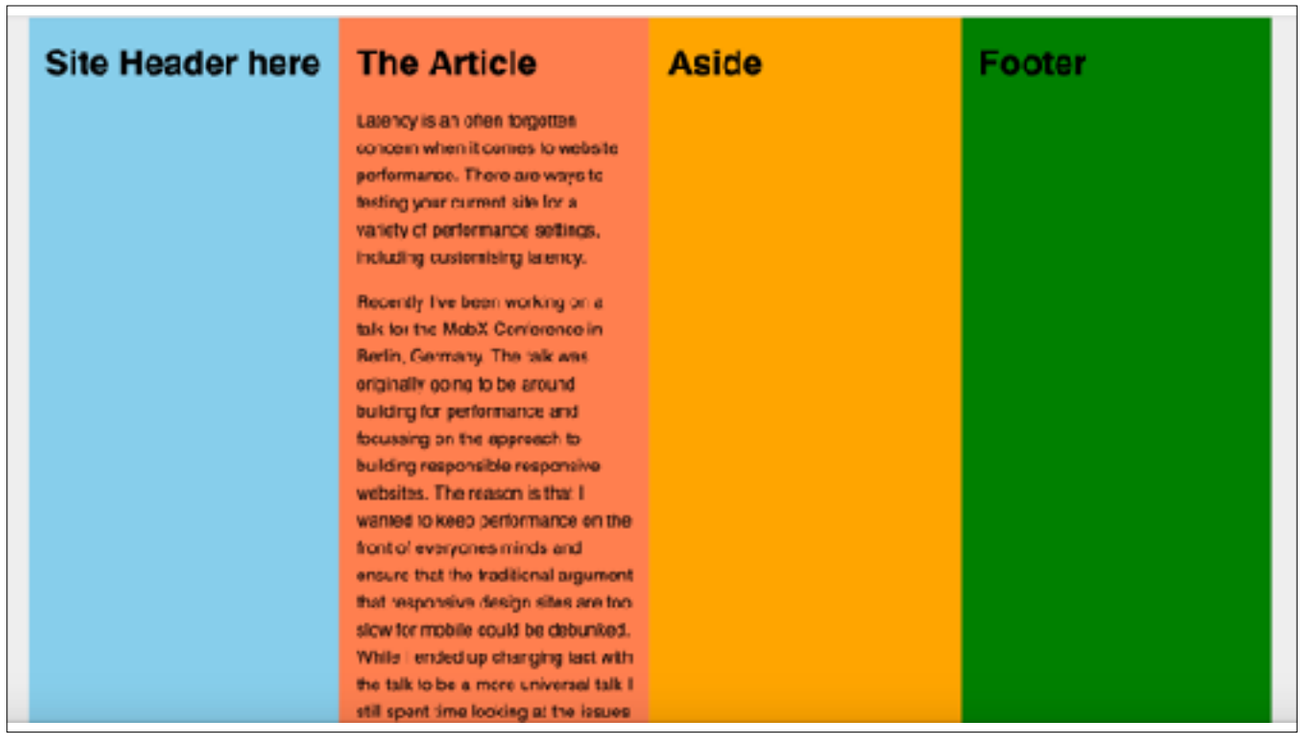
```
<div class="container">
    <header class="header"></header>
    <article class="article"></article>
    <aside class="aside"></aside>
    <footer class="footer"></footer>
</div>
```

```
.container {
    display: grid;
  }
```

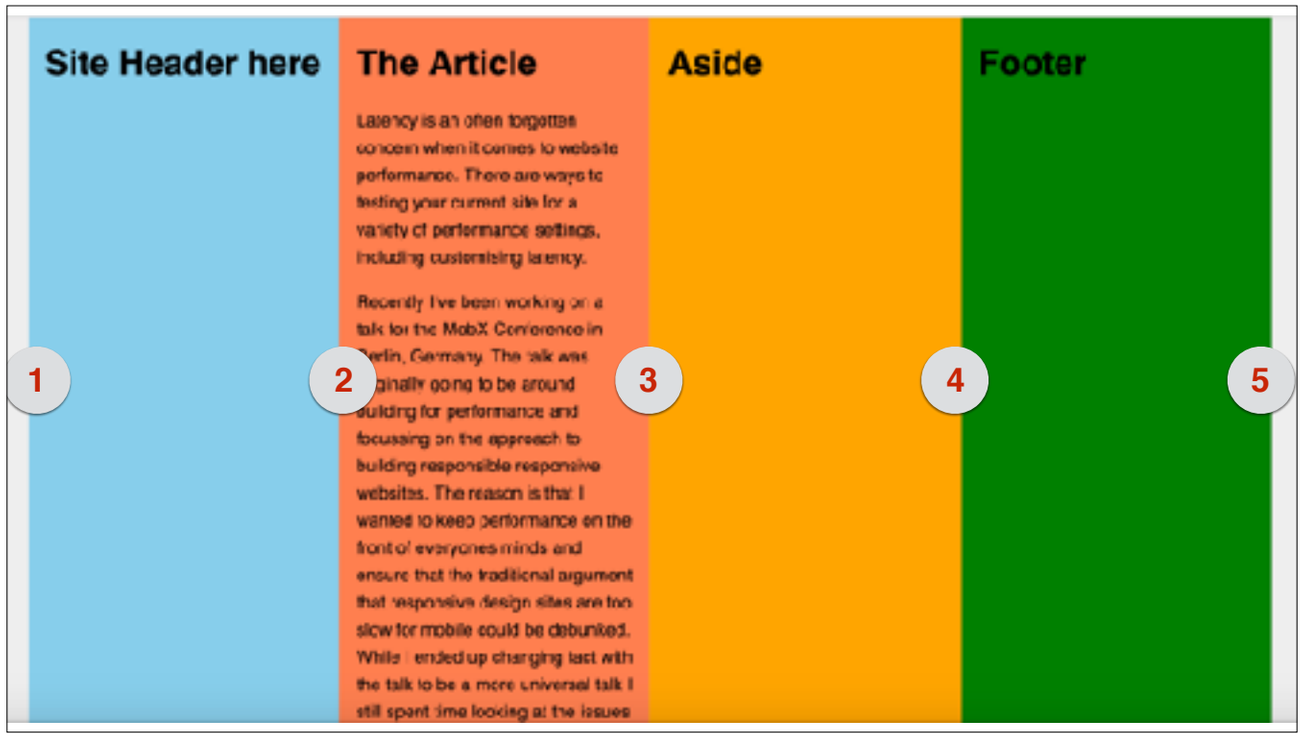
Just setting the grid will give us a stacked version of the page.

Incidentally any browser that do not support Grid will get this as well.

```
@media (min-width: 50em) {
  .container {
   grid-template-rows: auto;
   grid-template-columns: repeat(4,
1fr);
  }
```



don’t forget to mention the equal height columns


Site Header here
The Article
Latency is an often forgotten concern when it comes to website performance. There are ways to testing your current site for a variety of performance settings, including customising latency.
Recently I've been working on a talk for the MobX Conference in Berlin, Germany. The talk was originally going to be around building for performance and focussing on the approach to building responsible responsive websites. The reason is that I wanted to keep performance on the front of everyones minds and ensure that the traditional argument that responsive design sites are too slow for mobile could be debunked. While I ended up changing last with the talk to be a more universal talk I still spent time looking at the issues
Aside
Footer
1
2
3
4
5

```
.header, .footer {
  grid-column:1/5;
}
.article {
  grid-column:1/4;
}
.sidebar {
  grid-column: 4/5;
}
```

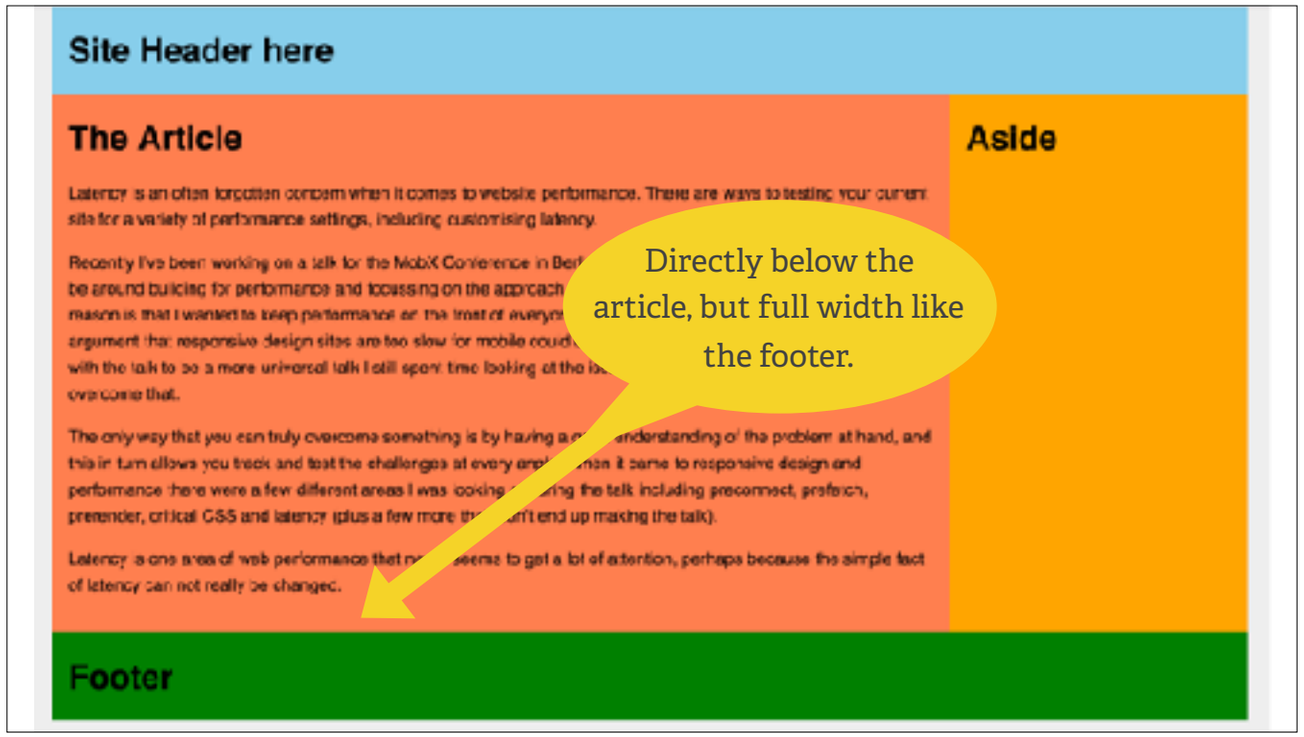

1
Site Header here
5
The Article
Aside
1
4
4
5
1
Footer
5

make an object fill its grid container.

http://caniuse.com/#feat=object-fit

Bringing Grid & Flex together


Site Header here
The Article
Aside
Add some related articles after the main article
Footer


Site Header here
The Article
Aside
Directly below the article, but full width like the footer.
Footer

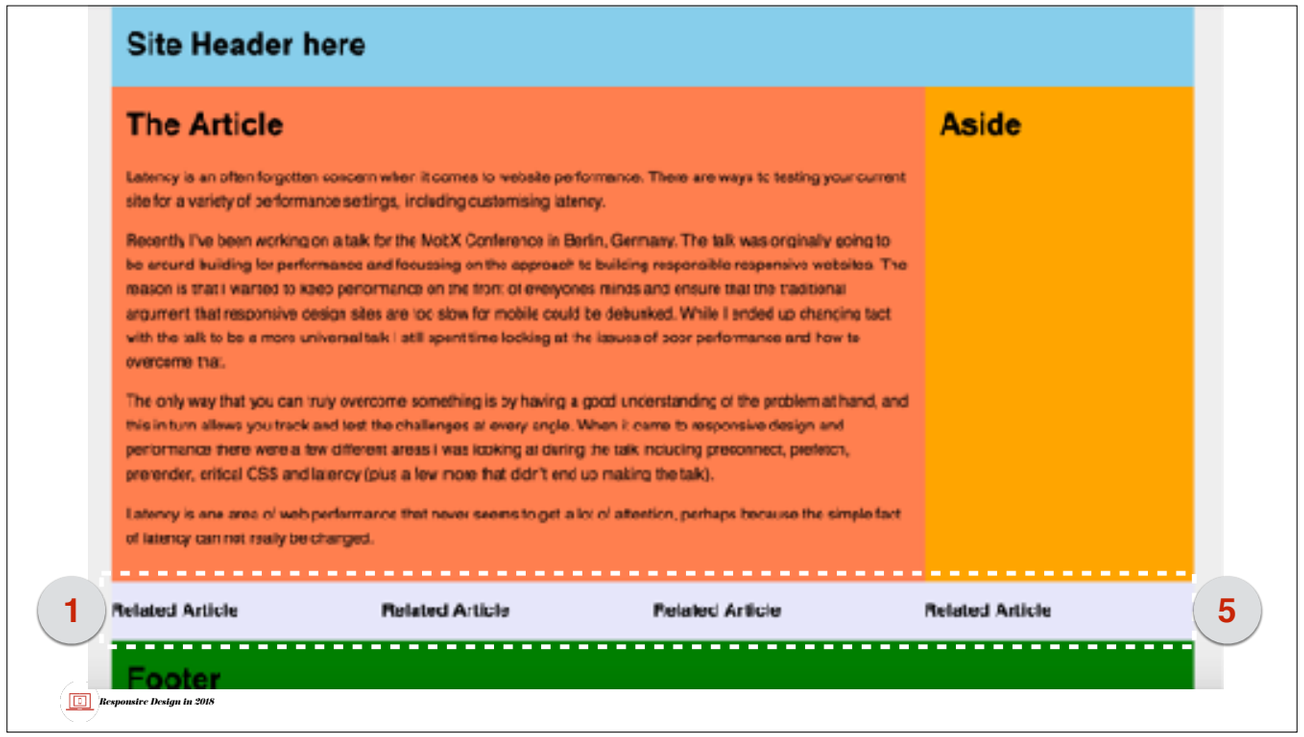
```
<div class="container">
    <header class="header"></header>
    <article class="article"></article>
    <aside class="related"></aside>
    <aside class="aside"></aside>
    <footer class="footer"></footer>
</div>
```

```
.related {
    grid-column: 1/5;
    grid-row: 3;
  }
```


We set the row to auto which is the same as leaving that off, it evenly divides the elements into the available columns.

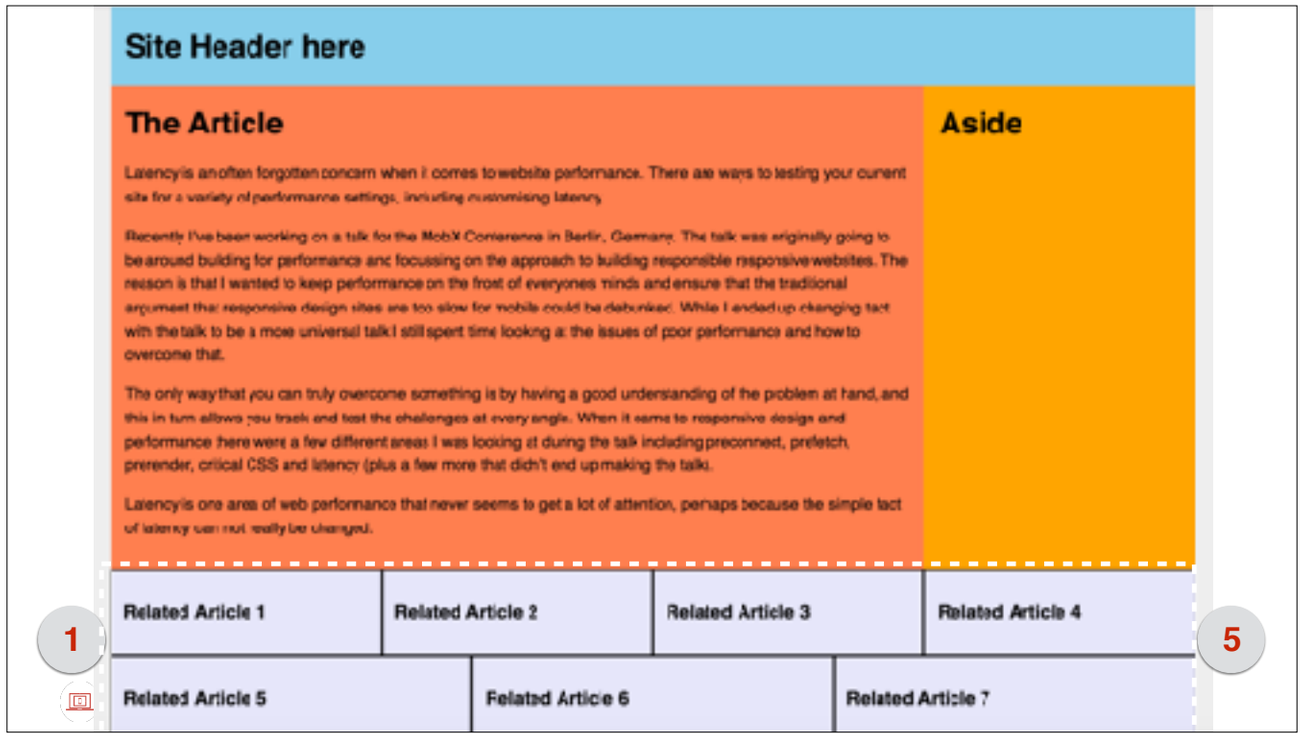
Instead of letting the row be dictated by source order though we specify the location in the same way that we specified the columns earlier.


Site Header here
1
The Article
Aside
Latency is an often forgotten concern when it comes to website performance. There are ways to testing your current site for a variety of performance settings, including customising latency.
Recently I've been working on a talk for the NotX Conference in Berlin, Germany. The talk was originally going to be around building for performance and focussing on the approach to building responsible responsive websites. The reason is that I wanted to keep performance on the front of everyones minds and ensure that the traditional argument that responsive design sites are too slow for mobile could be debunked. While I ended up changing tact with the talk to be a more universal talk I still spent time looking at the issues of poor performance and how to overcome that.
The only way that you can truly overcome something is by having a good understanding of the problem at hand, and this in turn allows you track and test the challenges at every angle. When it came to responsive design and performance there were a few different areas I was looking at during the talk including preconnect, prefetch, prerender, critical CSS and latency (plus a few more that didn't end up making the talk).
Latency is one area of web performance that never seems to get a lot of attention, perhaps because the simple fact of latency can not really be changed.
2
Related Article
Related Article
Related Article
Related Article
3
Footer
4



We set the row to auto which is the same as leaving that off, it evenly divides the elements into the available columns.

Instead of letting the row be dictated by source order though we specify the location in the same way that we specified the columns earlier.



We set the row to auto which is the same as leaving that off, it evenly divides the elements into the available columns.

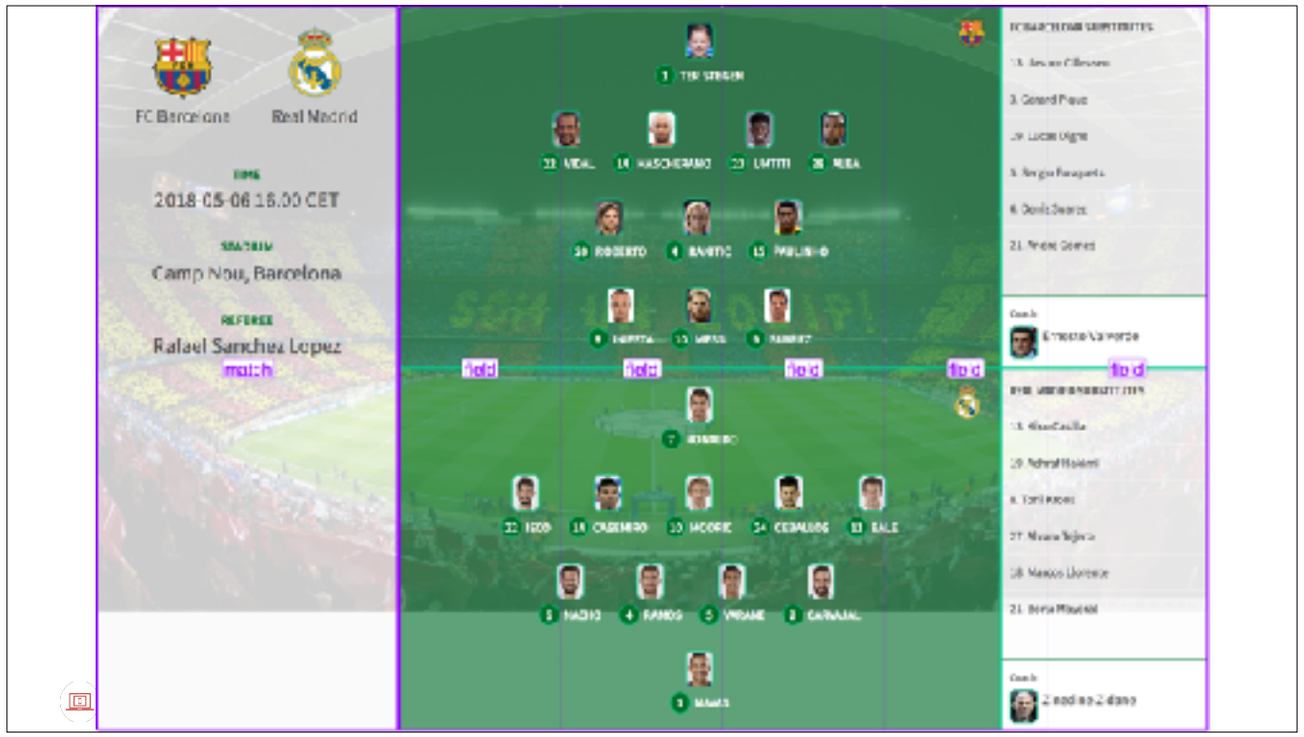
Instead of letting the row be dictated by source order though we specify the location in the same way that we specified the columns earlier.

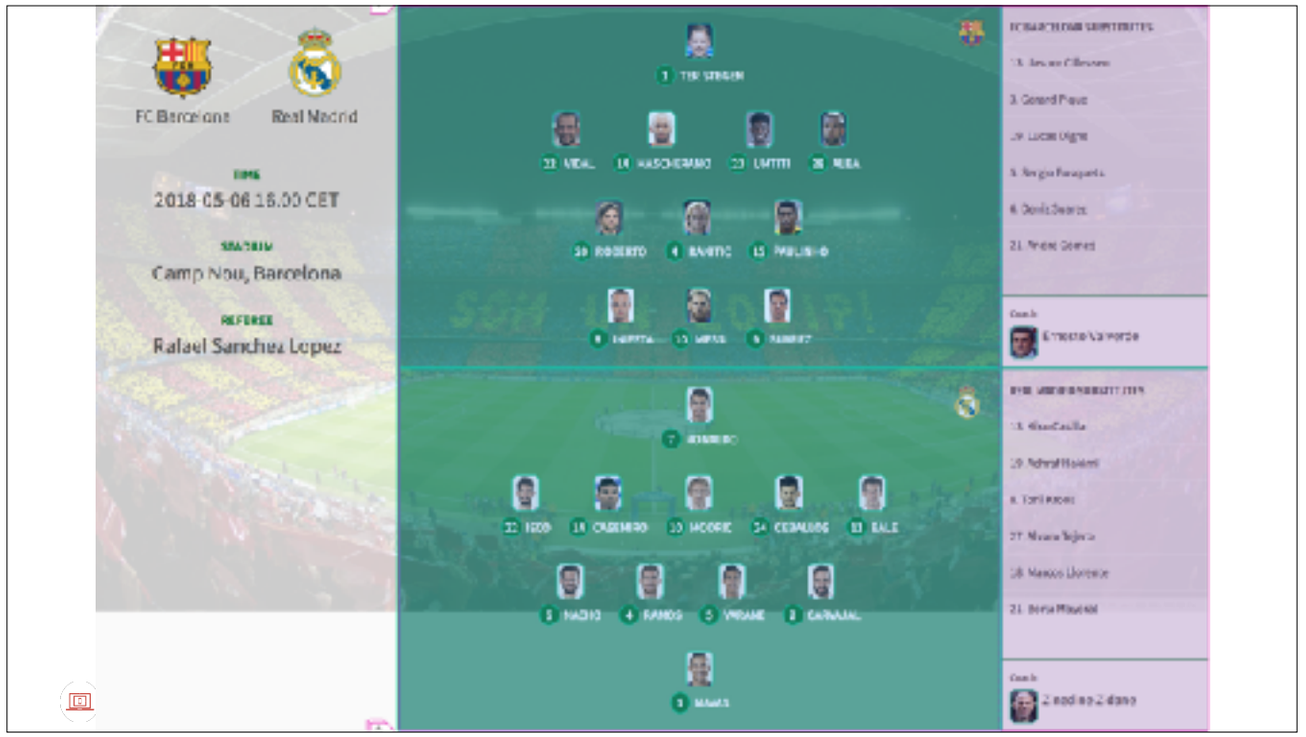
We set the row to auto which is the same as leaving that off, it evenly divides the elements into the available columns.

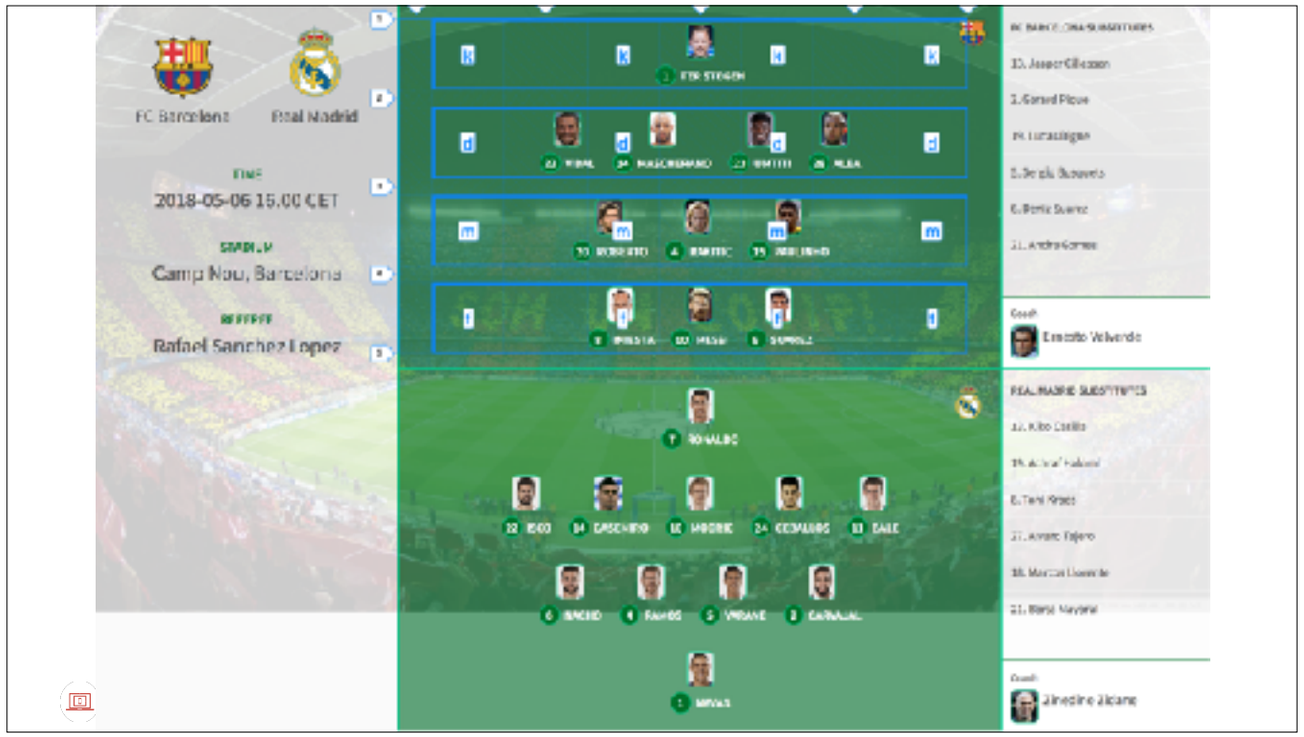
Instead of letting the row be dictated by source order though we specify the location in the same way that we specified the columns earlier.

make an object fill its grid container.

http://caniuse.com/#feat=object-fit


FC Barcelona
Real Madrid
2018-05-06 16:00 CET
Camp Nou, Barcelona
Rafael Sanchez Lopez
match
field
field
field
field
field


FC Barcelona
Real Madrid
2018-05-06 16:00 CET
Camp Nou, Barcelona
Rafael Sanchez Lopez


FC Barcelona
Real Madrid
2018-05-06 15.00 CET
Camp Nou, Barcelona
Rafael Sanchez Lopez
RONALDO
Ernesto Valverde

Responsive Design in 2018

https://erikdson.github.io/smashing-grid-challenge/

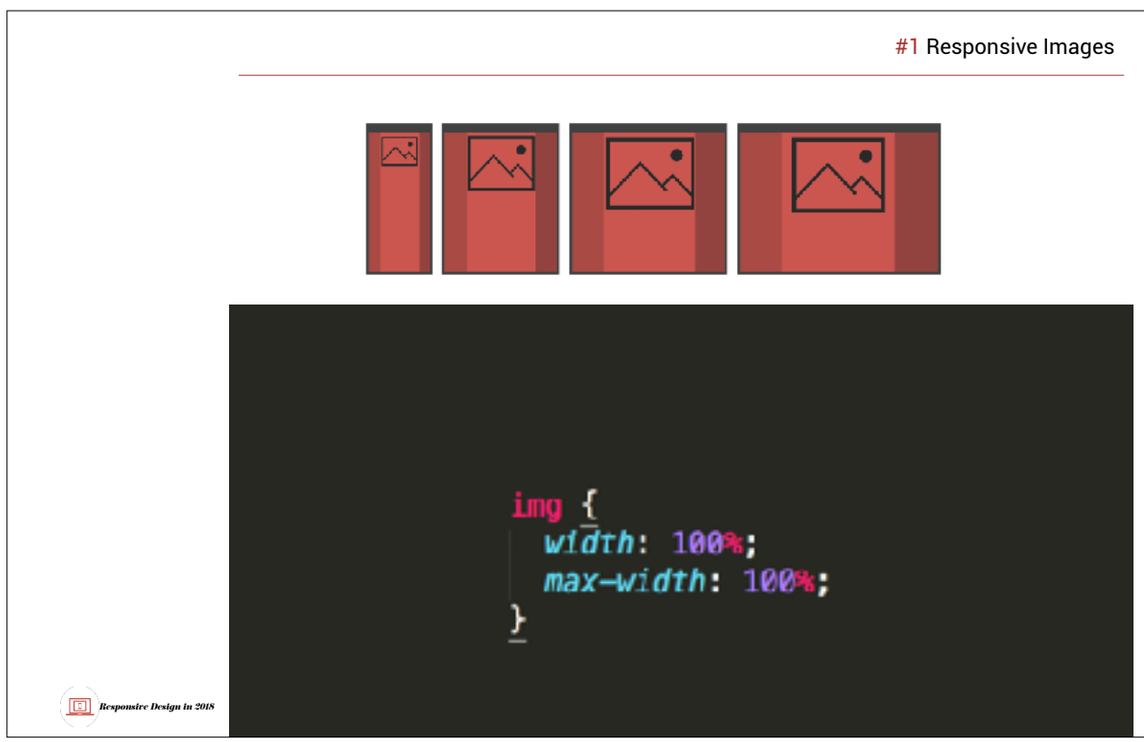
Responsive Images

#1 Responsive Images

An image is fixed widths, that’s just how raster images work.

We needed a way to make these flexible.

This image is how we want to achieve that. An image gradually expanding to fit within it’s container as the viewport increased or decreased.



Make the image width 100% of the container that it lives in.

Max-width stops the image from going outside of the constraints of the container if the images natural size is larger.

Height Auto?
max maybe remove

~~Responsive Images~~

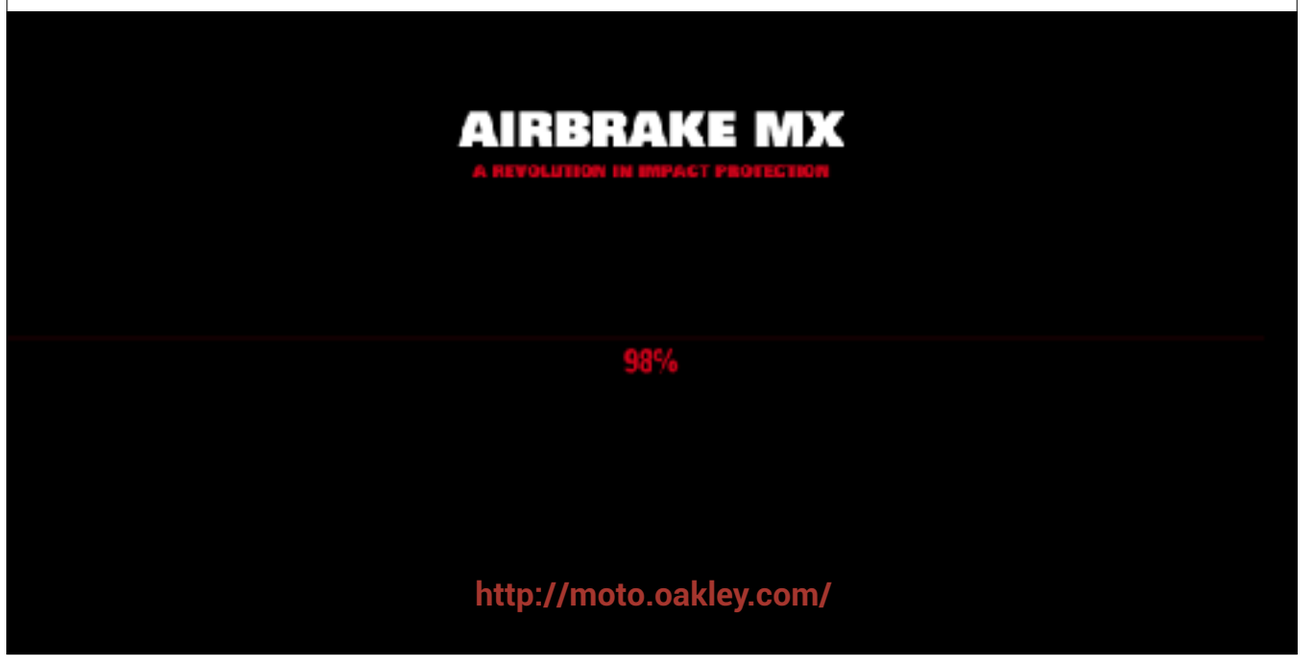
History of Responsive images:

Started with RICG which is now known as RICG.. responsive issues. The great thing about changing their name in this way meant no logo, website, bread redevelopment ;)

There were some competing standards, but the important ones are the ones that won. These are
- srcset
- srcset & sizes
- <picture>

Here’s how we might hand code these and the use cases where we might need to use them.

SRSET - Only focusses upon the pixel density. When I say pixel density I’m talking about the 2x screens of Apple (otherwise known as Retina) and the 3x screens of <insert android device here>



Moto.Oakley site when released showed us a few things.

1) You could design outrageous RWD sites
2) There could be some performance issues with RWD implementations (84MB)
3) We needed a solution for images in responsive design

…. This was a beautiful website and it did a lot to break the mould of what was quickly been framed as a “typical responsive layout” with “All responsive design sites look the same”.

Unfortunately, it also led to the stigma that all responsive websites are slow… or slower than their mobile/desktop dedicated alternatives. We will see why this isn’t so a little bit later and I’ll give you an overview of the steps that you can take to ensure that you make a mobile first responsible responsive design. But – once someone has planted the thought of something being slow then it’s always going to be in the back on their mind.

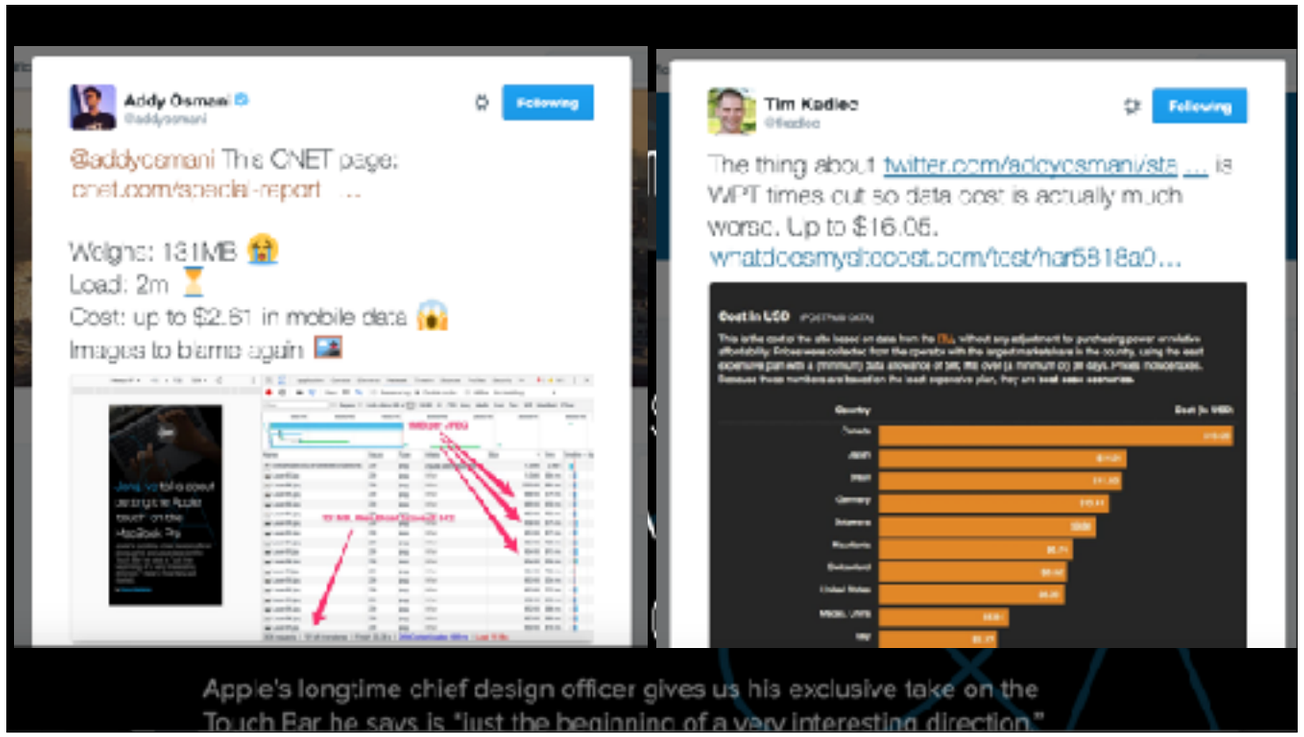
Responsive Design in 2018

Milwaukee police was another one that took an artistic license a little too far in the creation of their website. As creative as the site became and it has reduced the size greatly since the first version was released, it still harbours a 7.5MB homepage with 6.5MB of that coming by way of image. Now if I'm looking for something on a police website I'd much rather have the phone number for the station rather than an image of the Chief of Police who'd file size is representative of that phone number.

You might say “oh well those were ages ago when we were just getting started, we know better now”….

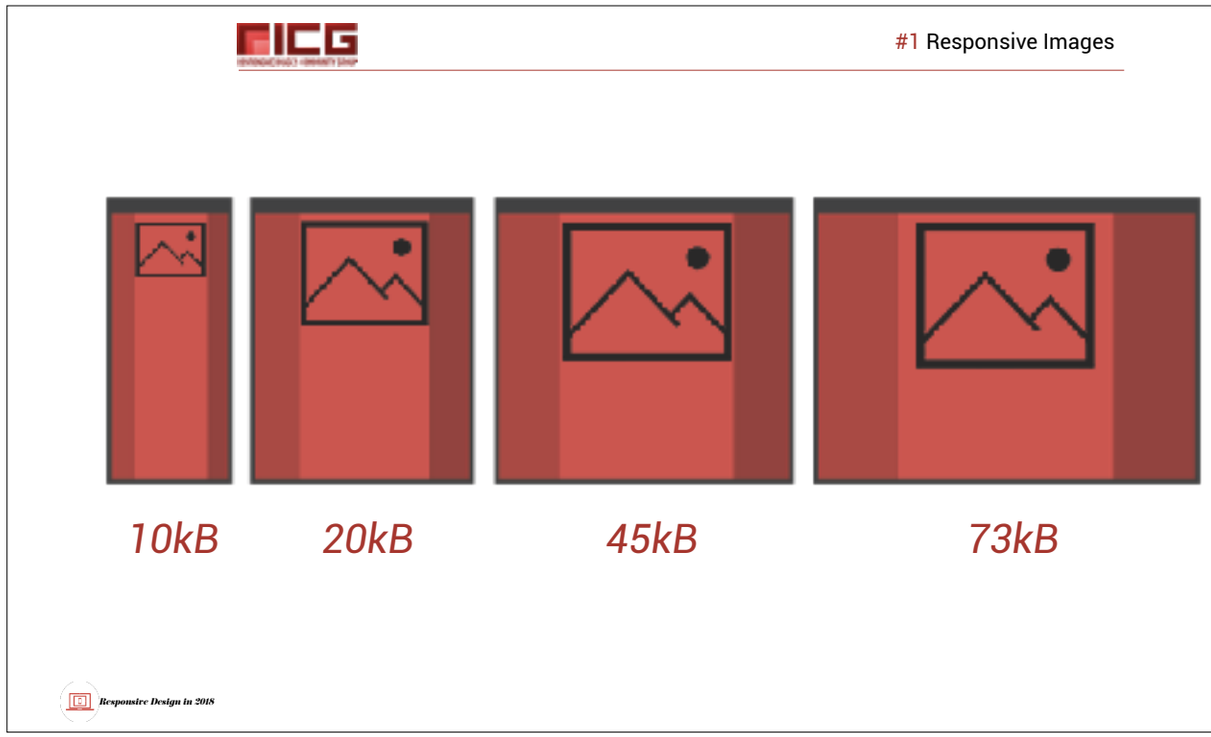
Sadly no.

131MB in size
2.2 minutes load
320 http requests



You might say “oh well those were ages ago when we were just getting started, we know better now”….

Up to $16.05
USA - $8.35



We wanted to send small sized images to

200px 400px 800px 1600px

~~200px~~ ~~400px~~ ~~800px~~ ~~1600px~~

400px 800px 1600px 3200px

http://www.html5rocks.com/en/tutorials/responsive/picture-element/

http://www.html5rocks.com/en/tutorials/responsive/picture-element/

picture/srcset

They ended up going with two solutions to cover off the three problems, a SRCSET and a PICTURE Solution. Today I won't go through both of them in any detail because there are other people that will spend a whole session dedicated to the approach, but lets take a look at the format.

```
<img
      src="horse-350.jpg"
      srcset="horse-350.jpg 350w,
      horse-500.jpg 500w,
      horse-1024.jpg 1024w,
      horse.jpg 2000w"
      sizes="(min-width: 64em) 70vw,
      (min-width: 37.5em) 95vw,
      100vw"
      alt="A horse" />
```

```
<img
        src="horse-350.jpg"
        srcset="horse-350.jpg 350w,
        horse-500.jpg 500w,
        horse-1024.jpg 1024w,
        horse.jpg 2000w"
        sizes="(min-width: 64em) 70vw,
        (min-width: 37.5em) 95vw,
        100vw"
        alt="A horse" />
```

```
<img
      src="horse-350.jpg"
      srcset="horse-350.jpg 350w,
      horse-500.jpg 500w,
      horse-1024.jpg 1024w,
      horse.jpg 2000w"
      sizes="(min-width: 64em) 70vw,
      (min-width: 37.5em) 95vw,
      100vw"
      alt="A horse" />
```


Declare the image of the horse, a horse is a horse of course of course.

Backwards compatibility.

Opera MIni and IE that don’t support the other markup simple ignore it.

```
<img
    src="horse-350.jpg"
    srcset="horse-350.jpg 350w,
    horse-500.jpg 500w,
    horse-1024.jpg 1024w,
    horse.jpg 2000w"
    sizes="(min-width: 64em) 70vw,
    (min-width: 37.5em) 95vw,
    100vw"
    alt="A horse" />
```


Declare all of the image sizes in srcset. Include the image width using the w descriptor so the browser knows the width of each image.

```
<img
      src="horse-350.jpg"
      srcset="horse-350.jpg 350w,
      horse-500.jpg 500w,
      horse-1024.jpg 1024w,
      horse.jpg 2000w"
      sizes="(min-width: 64em) 70vw,
      (min-width: 37.5em) 95vw,
      100vw"
      alt="A horse" />
```


Sizes are what tells the browser our site layout. Here we're staying for any viewport that is 64ems and bigger use an image that will occupy 70% of the viewport

```
<img
      src="horse-350.jpg"
      srcset="horse-350.jpg 350w,
      horse-500.jpg 500w,
      horse-1024.jpg 1024w,
      horse.jpg 2000w"
      sizes="(min-width: 64em) 70vw,
      (min-width: 37.5em) 95vw,
      100vw"
      alt="A horse" />
```


If the viewport isn't that big, then for any viewport that is 37.5ems and bigger use an image that occupies 95% of the viewport

```
<img
      src="horse-350.jpg"
      srcset="horse-350.jpg 350w,
      horse-500.jpg 500w,
      horse-1024.jpg 1024w,
      horse.jpg 2000w"
      sizes="(min-width: 64em) 70vw,
      (min-width: 37.5em) 95vw,
      100vw"
      alt="A horse" />
```


…and if the viewport is smaller than that then use an image that occupies 100% of the viewport

From a performance point of view it doesn't matter if you use min-width or max-width in the sizes attribute, but the source order does matter. The browser will always use the first matching size.

We’re using min-width here rather than max-width. If you do min width you should declare the larger breakpoints first because the browser will implement the first matching sizes query as the right one. If you choose to use max-width then reverse the process with the smallest values first.

```
<img
      src="horse-350.jpg"
      srcset="horse-350.jpg 350w,
      horse-500.jpg 500w,
      horse-1024.jpg 1024w,
      horse.jpg 2000w"
      sizes="(min-width: 64em) 70vw,
      (min-width: 37.5em) 95vw,
      100vw"
      alt="A horse" />
```


always have alt text

```
<img
      src="horse-350.jpg"
      srcset="horse-350.jpg 350w,
      horse-500.jpg 500w,
      horse-1024.jpg 1024w,
      horse.jpg 2000w"
      sizes="(min-width: 64em) 70vw,
      (min-width: 37.5em) 95vw,
      100vw"
      alt="A horse" />
```

#1 Responsive Images

Responsive Design in 2018

Support for SRCSET is great.

Ships with wordpress

```
<img
      src=""
      srcset="horse-350.jpg 350w,
      horse-500.jpg 500w,
      horse-1024.jpg 1024w,
      horse.jpg 2000w"
      sizes="(min-width: 64em) 70vw,
      (min-width: 37.5em) 95vw,
      100vw"
      alt="A horse" />
```


Browser that require picture fill to work will still download the first SRC image thanks to pre-rendering, but there’s a recommendation from Filament Group to remove the “SRC”. If the plugin doesn’t fire for whatever reason you’re left with the alt text. I’m not sure how I feel about deliberately leaving that out but it will depend of what you need for your site.

Currently the SRCSET approach has some pretty good coverage on browsers as you can see from this Can I Use snapshot.

If you need to support browsers that do not have support then there’s a nice little plugin called picturefill.js created by Scott Jehl…. but…..

darcey-avery.jpg

darcey-avery-cropped.jpg

darcey-avery.jpg

darcey-avery-cropped.jpg

```
<picture>
  <source
    media="(min-width:50em)"
    srcset="darcey-avery.jpg ">
  <source
   srcset="darcey-avery-cropped.jpg ">
  <img src="darcey-avery.jpg">
</picture>
```

Describe the settings on this page before you press again to start the video

explain the video, at the end mention the code to copy and paste in and that you should run this through an optimiser

Testing Tools

At the beginning I talked about CERN being the centre of our areas to understand the Universe, well at the centre of CERN, or more around the entire outer ring is the Large Hadron Collider. The test bed for the Universe.

I run an application on my laptop called BOINC. It should really receive the award for the most ingenuitive app with the silliest name.

Responsive test tools like the google material design one, which incidentally you can totally do with Am I Responsive design, are not proper test tools. These are tools that allow you to visualise how your design might look in particular devices. The thing is we don't know what devices are going to be around tomorrow and the whole idea of building a responsive web site is that you have it forever, and that it can adapt to any screen size.

I like Brad Frosts tool that comes as part of PatternLab called ISH. It give's you tabletISH widths, desktopISH widths, mobileISH widths… you get the idea. It even has a disco mode.

The tools that can really help you are devices, you need to test on real devices. Given the amount of money we charge to build websites it's not out of the realm of possibility that for every project you do you bake in another $500 to buy a new device to add to your list of testing devices. Over time you will build up a great suite. If you're just starting out then there are places that will let you come in a use their facilities, in fact my friend Jay Messinger is the co-founder of the Open Device Lab which provides a single location for you to find out a testing place near you.

Once you've built up your own test suite I urge you to also add yourself to the list.

Right, now that we have some devices to do our testing we have the problem that we need to test on every single device. So for each additional device we add that's

Performance

- Images - optimising images, using picture/srcset,
- HTTP2 (https required)
    - This could fundamentally change how we build the backend of websites. Things like sprites, concatenating all of our files together all came about due to the fact that each http request for a new file would add to the load time. That’s not a problem with HTTP2 so will it be better to just keep everything separate?
- Service Workers
- meta DNS prefetch, prerender, preconnect (https://www.w3.org/TR/resource-hints/)
-

add this to the header in the same way that you might do with the Meta Viewport tag.

This is the Accept-Client Hints, and we want to share the Device Pixel Ration, Viewport Width, and the Width.

```
GET /_assets/images/content/band-alabama-
shakes_v9kics_c_scale,w_1062.jpg
Viewport-Width: 1280
DPR: 2
Width: 1080
Accept: image/webp,image/*,*/*;q=0.8
```


Now we get

- The Device Pixel Ratio of the devices
- The width of the image that would best fill the position
- The type of image that the browser accepts

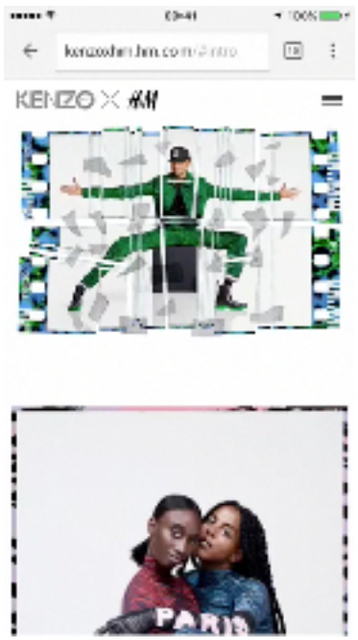

<picture>
<!-- serve WebP to Chrome and Opera -->
<source
media="(min-width: 50em)"
sizes="50vw"
srcset="/image/thing-200.webp 200w, /image/thing-400.webp 400w,
/image/thing-800.webp 800w, /image/thing-1200.webp 1200w,
/image/thing-1600.webp 1600w, /image/thing-2000.webp 2000w"
type="image/webp">
<source
sizes="(min-width: 30em) 100vw"
srcset="/image/thing-crop-200.webp 200w, /image/thing-crop-400.webp 400w,
/image/thing-crop-800.webp 800w, /image/thing-crop-1200.webp 1200w,
/image/thing-crop-1600.webp 1600w, /image/thing-crop-2000.webp 2000w"
type="image/webp">
<!-- serve JPEG-XR to Edge -->
<source
media="(min-width: 50em)"
sizes="50vw"
srcset="/image/thing-200.jpgxr 200w, /image/thing-400.jpgxr 400w,
/image/thing-800.jpgxr 800w, /image/thing-1200.jpgxr 1200w,
/image/thing-1600.jpgxr 1600w, /image/thing-2000.jpgxr 2000w"
type="image/vnd.ms-photo">
<source
sizes="(min-width: 30em) 100vw"
srcset="/image/thing-crop-200.jpgxr 200w, /image/thing-crop-400.jpgxr 400w,
/image/thing-crop-800.jpgxr 800w, /image/thing-crop-1200.jpgxr 1200w,
/image/thing-crop-1600.jpgxr 1600w, /image/thing-crop-2000.jpgxr 2000w"
type="image/vnd.ms-photo">
<!-- serve JPEG to others -->
<source
media="(min-width: 50em)"
sizes="50vw"
srcset="/image/thing-200.jpg 200w, /image/thing-400.jpg 400w,
/image/thing-800.jpg 800w, /image/thing-1200.jpg 1200w,
/image/thing-1600.jpg 1600w, /image/thing-2000.jpg 2000w">
Responsive Design in 2018


```
<picture>
  <source media="(min-width: 50em)"
          sizes="50vw"
          srcset="/image/thing">
  <img sizes="100vw" src="/image/thing-crop">
</picture>
```

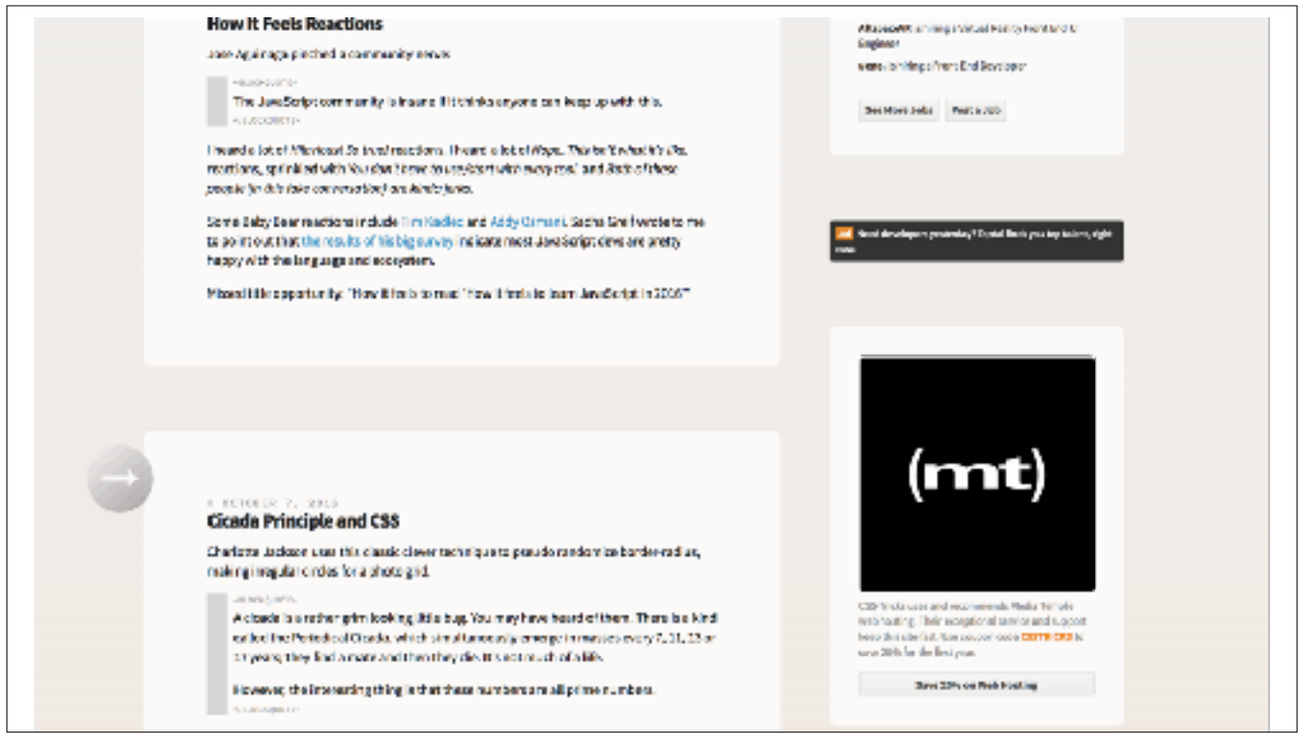
You might notice that we don’t even specify the image type here.

Well that would result in the following request

SVG & Animation


KENZO × H&M
PARIS

beta.rallyinteractive.com



When you're trying to advertise making the advertisement more like part of the page design is great for the users experience for both your own readers as well as the advertiser as well.

This is a great hover effect that shows what Media Temple actually provide.

Image techniques

We can go even further to make our images a lot better. We can work on the images before we upload them onto the site to become a lot lighter from a download point of view, and something which actually helps our focal point.

https://responsivedesign.is/articles/reducing-image-sizes

https://www.flickr.com/photos/zeldman/11298416615/

Optimising	Original Size (bytes)	New Size (bytes)	Saving (bytes)
Cropping/ImageDimensions	1,005,480 bytes	967,471	38,009
Reduce image quality	967,471	211,095	756,376
Image Blurring	211,095	149,792	61,303
Optimising Images	149,792	147,772	2,020
Total Saving	**857,706 bytes**		

- Crop it 1600px
- Save out at 60% quality
- Blur the background as it’s not important and brings out ethans beautiful features a little more
- Run it through Image Optim