AQUENT
GYMNASIUM

RESPONSIVE WEB DESIGN:
BUILD A PORTFOLIO FOR ALL DEVICES

LESSON 6

ABOUT THIS DOCUMENT

This handout is an edited transcript of the Responsive Web Design lecture videos. There's nothing in this handout that isn't also in the videos, and vice versa. Some students work better with written material than by watching videos alone, so we're offering this handout to you as an optional, helpful resource.

Some elements of the instruction, like live coding, can't be recreated in a document like this one. We encourage you to use this handout alongside the videos, rather than as a replacement of them.

CHAPTER 1: USING A FLUID GRID

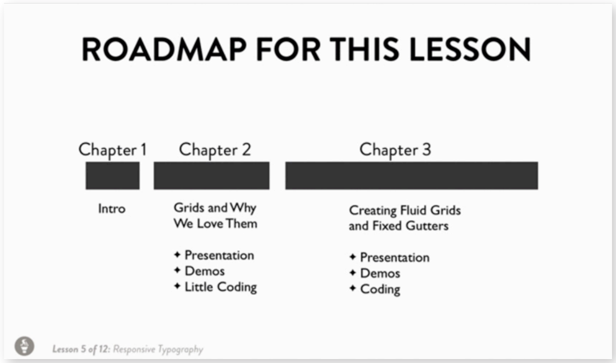
Welcome to Responsive Web Design, a Gymnasium course brought to you by Aquent and Vitamin T. Responsive Web Design or Promote Yourself Responsibly: Build a Portfolio for all Devices.

GRIDS AND WHY WE LOVE THEM

Lesson 6 of 12: Responsive Web Design

This is lesson 6, Using a Fluid Grid. As always, there will be an assignment and a brief quiz at the end of this lesson. Be sure to use the pause button at any given point in order to stop the video, catch up with code, and so forth. If you have questions, be sure to hit the forum. There you'll have TA's and teachers available to answer your questions as well as your other classmates.

Let's just talk a little bit about the road map for this lesson. We're currently in chapter 1, the intro. Chapter 2 will be discussing grids and why we love them. There will be a presentation, some short demos, and a little bit of coding. In chapter 3, Creating Fluid Grids and Fixed Gutters, we'll be getting to the heart of the lesson. There will be a presentation, some demos, and lots of coding.



ROADMAP FOR THIS LESSON

Lesson 5 of 12: Responsive Typography

This is Responsive Web Design.

CHAPTER 2: GRIDS AND WHY WE LOVE THEM

Grids and why we love them. Let's get started and talk about what we're going to be doing. We'll have an introduction to grids. We'll talk about grids on the web. We'll do a small review of a baseline grid. And finally, we'll take a look at creating fluid grids.

LET'S GET STARTED

- An introduction to grids
- Grids on the web
- The baseline grid
- The fluid grid

Lesson 6 of 12: Responsive Web Design

But first, grids. Grids are everywhere, even places you don't see them. Grids for city planning have been traced back to 3000 BC. And New York City, seen here, is famous for it's grid structure. In his book "Delirious New York", Rem Koolhaas called the Manhattan grid a graphic projection.

The uniformity of the city blocks provide a canvas in which a variety idea of architecture can thrive. City grids help us utilize the space we have in a meaningful and recognizable way. And whether these are city grids or grids we use for websites, the grid is an inescapable framework built into the very medium of digital imagery.

Raster graphics, also known as bitmap graphics, use the square pixel as the foundation for digital imagery. And the very screens we rely on are inherently grid-like. Theo H. Ballmer's "New Building" poster from 1928 is a Bauhaus poster with a very obvious grid. Newspapers have been using grids for a long time to display as many stories above the fold as possible. As a fringe benefit, modular grid templates have made it easier for production teams to lay out complex information in reliable and efficient ways.

Now, of course grids have been used on the web as well. And this will be the prime focus of what we'll be talking about later. But let's go back to print for a second.

Here, we can see Romek Marber's design for Penguin book covers. Created in the 1960s, this grid allowed for different cover designs but still had a structure that was recognizable to the brand. Without grids, we'd have a tough time making sense of anything. They help us know where to look for something and help us scan without having to see everything.

Khoi Vinh, a prolific designer who worked on www.newyorktimes.com and wrote a book on grids, said the following, "Design is nothing if not order applied to disorder." Let's take a look at some of the examples that Khoi used to illustrate a redesign of a fictional website called Yeeaahh!.

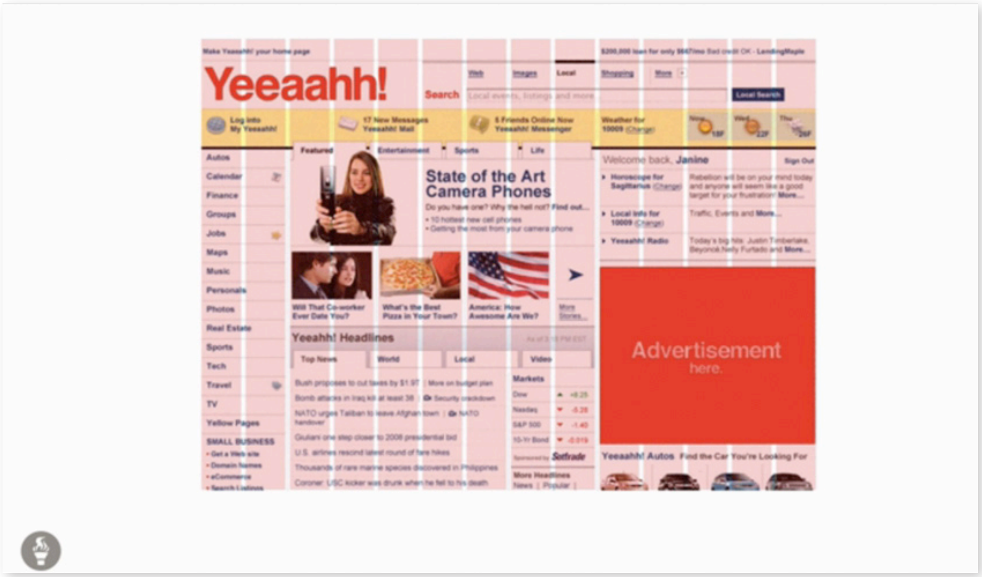
Khoi Vinh

"DESIGN IS NOTHING IF NOT ORDER APPLIED TO DISORDER"

Lesson 6 of 12: Responsive Web Design

In this page, there's a very complex grid taking place behind the scenes. Let's take a quick tour of some of the structure behind this grid. The entire grid itself uses a 14 column structure.

And we could see the header here uses its own horizontal column. On the left-hand side, we have a vertical column for the navigation keeping these firmly aligned. And in the center of the page, we have this grid structure here which uses four columns to align the tabs at the top and the bottom of this subsection.



One of the things that grids do is reinforce hierarchy and therefore the clarity of your content. We can see here this example of Crate & Barrel which actually exposes the grid. We can see the color swatches are also presented in a grid. And callouts are given extra weight whenever they span multiple columns, such as here where it says free local in-home furniture delivery.

As we've mentioned in past lessons, creating grid structure on the web has always been a little easier when we have a fixed width, for example 968 pixels. This allows us to create very common and regular columns, for example, each of these with their own fixed width. However, the challenge increases when we have fluid websites and responsive websites, and we're not just working toward a single screen but multiple screens and multiple devices.

In the last lesson, we talked a little bit about baseline grids. So we're just going to do a quick review of that here. In the last lesson, we laid the groundwork for grids by using relative font sizing. So here we have the HTML style with a font-size of 100% and a line-height of one. And both of these were used to reset our type styles.

Next, we define the ideal size for our paragraph text. In this case, 17 pixels was the original size. And then we converted it to ems.

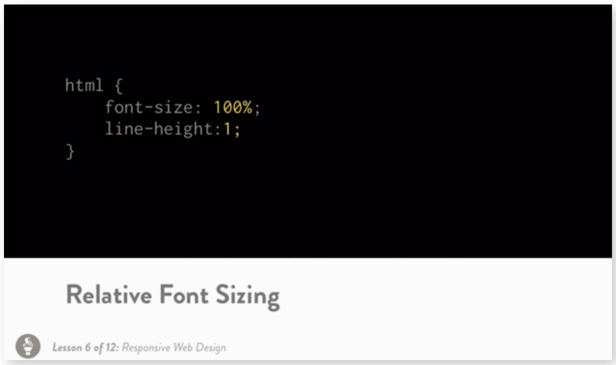
We talked a little bit about why ems are the ideal unit for responsive typography. They are tied to the font-size of the body. And this gives us a fluid, flexible baseline upon which to work.

The other part of the equation this line-height. So font-size and line-height allow us to create a typographic scale. In this case, the typographic scale and the magic number we used is 1.3. We took our scale of 1.3, and we divided that by the font-size of the paragraph, which is 1.063. And we came up with this number for line-height, 1.223.

And finally when it came to adding proportional space using margins, here we can see 0.612 em, which is being used for top and bottom margins. What all this really creates is what we call a vertical rhythm. We have consistent baselines. And this gives us consistent spacing for our paragraphs, lists, and so forth. And what we end up with is an attractive page that engages the reader's eye and also scales wonderfully for different screens.

So before Ethan Marcotte came out with his well-known article on responsive design, he actually published this article on fluid grids. The concepts here are as relevant as always. And in fact, we have used many of these in our lessons.

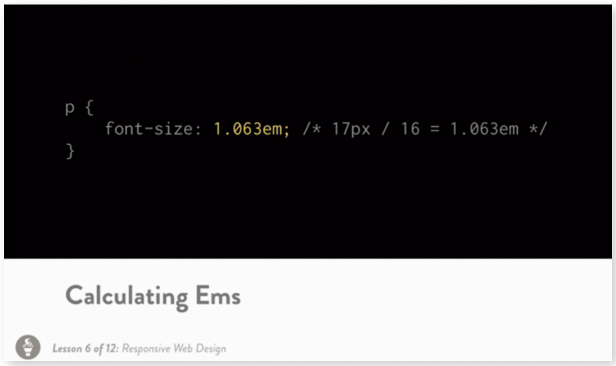
So our grid layout is actually halfway done, perhaps even more than halfway done. And just in the spirit of review, let's go ahead and take a look at how we can implement our vertical rhythm even further with another example. So you'll need to open up your text editor at this point. And open up the file portfolio_start.html. Go ahead and save this as portfolio_work.html. This creates a backup as always.



```
html {
    font-size: 100%;
    line-height:1;
}
```

Relative Font Sizing

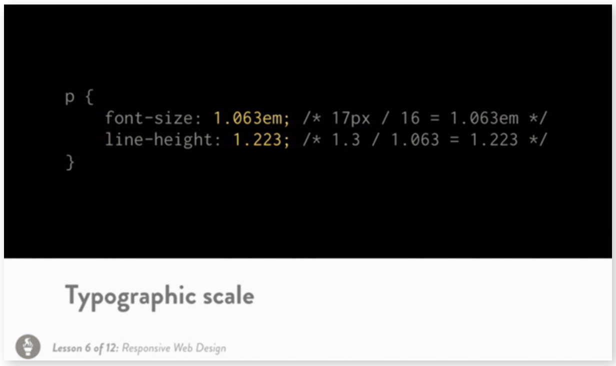
Lesson 6 of 12: Responsive Web Design



```
p {
    font-size: 1.063em; /* 17px / 16 = 1.063em */
}
```

Calculating Ems

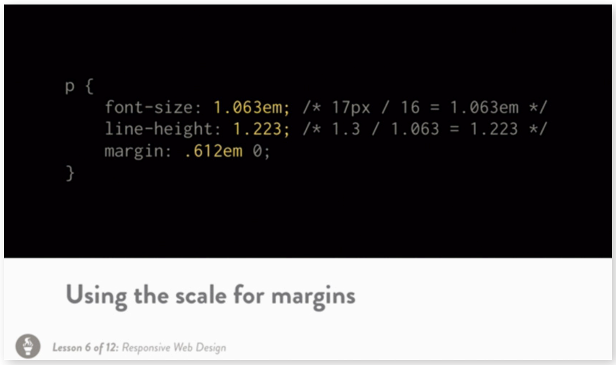
Lesson 6 of 12: Responsive Web Design



```
p {
    font-size: 1.063em; /* 17px / 16 = 1.063em */
    line-height: 1.223; /* 1.3 / 1.063 = 1.223 */
}
```

Typographic scale

Lesson 6 of 12: Responsive Web Design



```
p {
    font-size: 1.063em; /* 17px / 16 = 1.063em */
    line-height: 1.223; /* 1.3 / 1.063 = 1.223 */
    margin: .612em 0;
}
```

Using the scale for margins

Lesson 6 of 12: Responsive Web Design

Now, before diving in too deep, let me just talk a little bit about some of the work I've done on this file since the last lesson. I've moved all of the internal styles to an external stylesheet. I did this from an organizational perspective, but also so we could have a clean slate upon which to work in this lesson.

So let’s just quickly review this file, style.css. And here we can see our headings have now been organized as well as all our class and ID styles. And of course, we have our media queries here. And we have some print styles.

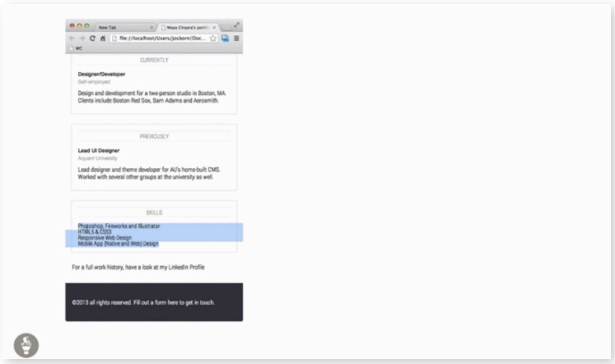
So for this lesson, we’ll largely be adding to our internal stylesheet. Although, we will have to return to our external stylesheet for a few items. Let’s just take a look at what we need to do.

We’ll go ahead and open up this page inside of our browser. And we need to remember that we’re primarily concerned with the single column view first. And then we’re going to take a look at the larger desktop three column view.

So let’s go ahead and go back to that single column view. And the section that we want to focus on is here, the Skills section. So this is actually an unordered list with the bullets removed. And we need to change the spacing here, because it’s looking pretty cramped.

Now, the good news is we don’t really have to create a new style. We can piggyback on our existing styles, and specifically the paragraph style. So all we’re going to do here is we’re going to add a comma and then add `#resume li`. So this targets the list item inside the resume ID, which is what we were just looking at in the skills section.

Let’s go ahead and save that. Take a look at this by opening the page in a new tab. And this is going to allow us to toggle back and forth between the old design and the new one. So there’s the old. There’s the new. Old, new.



CURRENTLY

Designer/Developer
Self-employed
Design and development for a two-person studio in Boston, MA. Clients include Boston Red Sox, Sam Adams and Aerosmith.

PREVIOUSLY

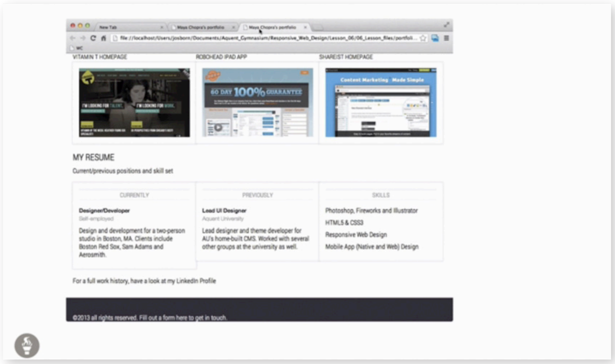
Lead UI Designer
Aquent University
Lead designer and theme developer for AU’s home-built CMS. Worked with several other groups at the university as well.

SKILLS

Photoshop, Fireworks and Illustrator
HTML5 & CSS3
Responsive Web Design
Mobile App (Native and Web) Design

For a full work history, have a look at my LinkedIn Profile

©2013 all rights reserved. Fill out a form here to get in touch.



MY RESUME

Current/previous positions and skill set

CURRENTLY	PREVIOUSLY	SKILLS
Designer/Developer	**Lead UI Designer**	Photoshop, Fireworks and Illustrator
Self-employed	Aquent University	HTML5 & CSS3
Design and development for a two-person studio in Boston, MA. Clients include Boston Red Sox, Sam Adams and Aerosmith.	Lead designer and theme developer for AU’s home-built CMS. Worked with several other groups at the university as well.	Responsive Web Design Mobile App (Native and Web) Design

For a full work history, have a look at my LinkedIn Profile

©2013 all rights reserved. Fill out a form here to get in touch.

So this is looking much better. We’re using the same consistent spacing that we have for the rest of the page. Let’s just take a look at how this works in the larger three column view. We’ll need to expand our browser window.

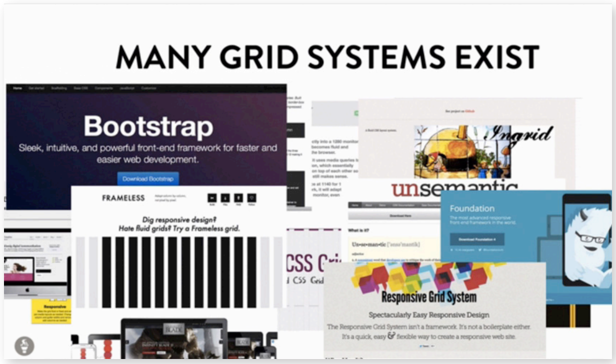
And look at that, we’ve got a nice, proportional spacing that aligns with our paragraphs. And this is exactly what we’re looking for. This is exactly the benefit of using vertical rhythm.

Now, you might be looking at all these styles, and all this math, and be thinking there’s got to be a better way. And what about frameworks? I’ve heard about grid frameworks and grid systems. Well, we talked about this in a previous lesson.

Many grid systems are out there. All of them have different benefits. But they also have the downside of having to learn that system.

And so what we’re really trying to do here is give you the skills that you need from a basic level. Think of it as custom work. You’re creating a custom grid here that you can use for many reasons. In our case, we’re creating a portfolio page.

But you'll be able to take these skills at a later point and be able to jump into any framework, in any grid system out there. So trust us, the skills you've learned here will be applicable to many situations in the future. Next up, we're going to be creating fluid grids with fixed gutters. This is Responsive Web Design.

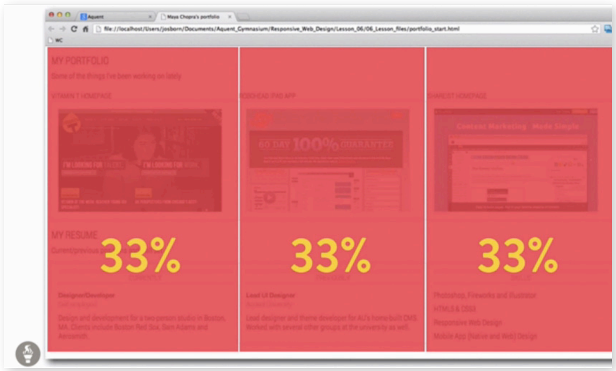

MANY GRID SYSTEMS EXIST
Bootstrap
Sleek, intuitive, and powerful front-end framework for faster and easier web development.
FRAMELESS
Dig responsive design? Hate fluid grids? Try a Frameless grid.
Ingrid
unsemantic
Foundation
CSS Grid
Responsive Grid System
Spectacularly Easy Responsive Design

CHAPTER 3: CREATE FLUID GRIDS WITH FIXED GUTTERS

In this chapter, you'll learn how to create fluid grids with fixed gutters. Specifically, your objective will be to add some left and right padding to your portfolio and resume boxes, while maintaining the typographic scale we discussed. So to make this a little more concrete, let's take a look at the bottom half of our page. And we have our two sections-- My Portfolio and My Resume.

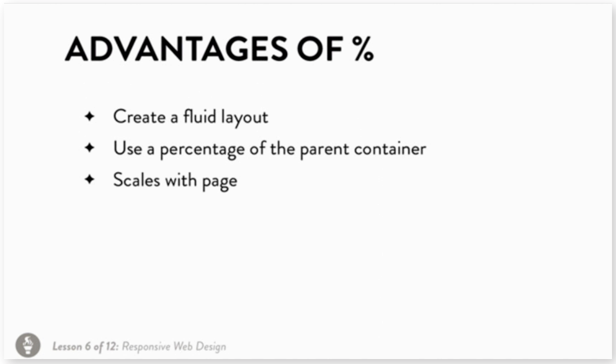
Each of these sections is using a three-column layout. In the top half, we have our image thumbnails, the bottom half, we have our boxes called Currently, Previously, and Skills. Just as a refresher, these are all currently using 33% to define the width of each column. So when we do this, this creates a very rough grade. But it could use some refining.

Specifically, what we don't have are good gutters between each of these columns. So let me just explore that a little further. Here, we can see, as we stretch the width of this page, that we are getting what I call these squishy gutters. So look at the space between each of those thumbnails. It's a percentage-based gutter.



And this means that it's fluid, but it's almost too fluid. What we'd like to do is add some space between each of these boxes to give them a little bit of breathing room. So of course, we're using percentages, and percentages are great because they use a percentage of the parent container and they scale with the page. But we need to do a little more work to make this truly shine.

So it's time to fix those gutters. Again, just to make it really clear what we want to do, this is the after. This is what we're looking to do here. So we have our six boxes here, or our three columns. And what we really want to do is we want to add these two gutters here in the middle. And we can't neglect the two gutters on the outside.



ADVANTAGES OF %

- Create a fluid layout
- Use a percentage of the parent container
- Scales with page

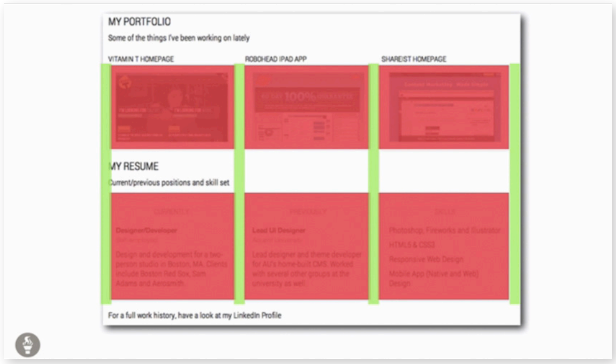
Lesson 6 of 12: Responsive Web Design

So to begin thinking of how we do that, again, let's just review at the very top, we have the before-- so three columns, 33% each. And what we're trying to do, what our target is is down below this after. So before and after-- how do we get from point A to point B?

OK, we have a problem here.

This is Houston, say again, please.

Houston, we've had a problem.

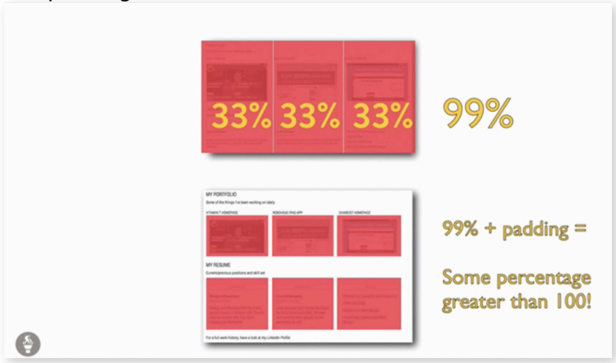

MY PORTFOLIO
Some of the things I've been working on lately
VITAMIN T HOMEPAGE
ROBOHEAD IPAD APP
SHAREIST HOMEPAGE
MY RESUME
Current/previous positions and skill set
For a full work history, have a look at my LinkedIn Profile

All of our column width is used up. If you think about it, we have to find three columns of 33%, so that's almost 100%. And by adding padding, we're asking that to be larger than 100%. So we could do this by going back and doing some new calculations. So instead of 33%, we could do some other math.

The truth is that math starts to get very ugly, very quickly. So we're actually going to look toward another solution called CSS3 box-sizing. And to understand box-sizing, let's just take a step back and talk about the CSS box model. So here's the traditional CSS box model.

I do want to point out that I'm not putting a color value in here, even though this box looks colored, just to keep it simple. But we have a width of 200 and a height of 100 for this box. If we add a border of 4 pixels to all four sides and then we add a padding of 10 pixels for all four sides, in the traditional box model, what does this do? So this is CSS 101.

When you do this, it increases the true size of the box. So borders and padding in the traditional box model will add to that true size. So the true height is 100 pixels plus 8 pixel border plus 20 pixel padding. That's 128. And the width is 200 plus 8 plus 20, equals 228. To wrap your head around this, I want you to think of this as the content box model.



And then we're going to look at another model in just a second. So the CSS box-sizing property allows you to switch the type of box-sizing that's being used. So the first model that we saw was `content-box`. And the other model that we're going to use is called `border-box`. And here's how it works.

If you add this property, box-sizing, and the value, border-box, this changes the rules. Suddenly, what happens here is the width and height will be 200 by 100, no matter what. That border and padding get applied to the inside. So border box is the value that we want to add to the box-sizing property.

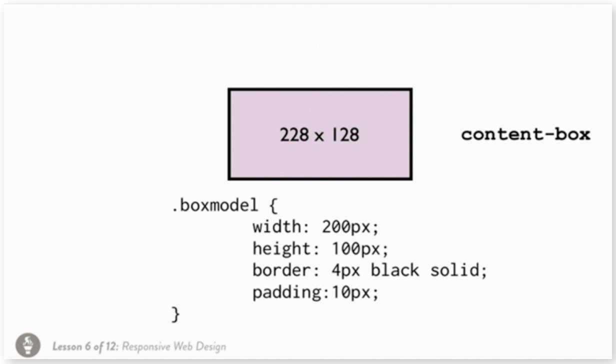
And we can apply this to our columns so we don't have to worry about increasing the width of the column by adding padding. Let's take a look at what we're going to do specifically. We're going to add additional HTML and CSS to our page to add our gutters. We need to add support for the box-sizing property for older versions of IE.

Additionally, we're going to be adding new styles to our media queries in order to align text to our new grid structure. And finally, we're going to be adding new style rules for margins within our media queries in order to prevent our layout from breaking on wide screens. So let's get started.

Make sure you have your text editor open again. And as in previous lessons, I've created a little cheat sheet for you called codetocopy.txt. So this is a text file that we're going to be copying and pasting some styles from. Let's go ahead and open that now.

What I want you to do is to find this class for wrapper. Now notice that we have a padding here--top and bottom of zero, left and right of 0.65em. There's that number again.

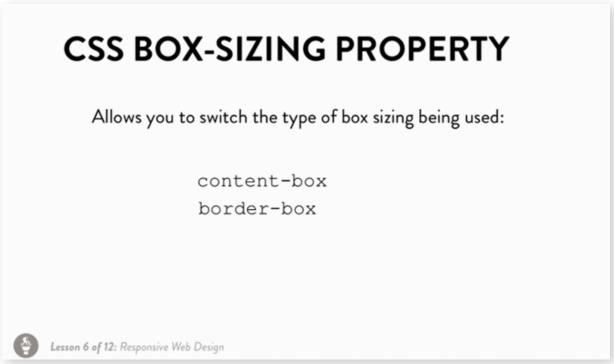
This is related to our scale of 1.3. So this is half. And of course, we have to use browser prefixes for box-sizing. So we're adding support for WebKit and Moz, or Mozilla browsers here. Let's go ahead and copy that entire class, paste it into our internal stylesheet. Now let's do one other thing here.



```
.boxmodel {
        width: 200px;
        height: 100px;
        border: 4px black solid;
        padding:10px;
}
```

Lesson 6 of 12: Responsive Web Design

Because we're using box-sizing, we can now afford to be a little more precise with those columns that we're creating. So open up your style.css stylesheet, locate the media queries section, and instead of a width of 33%, let's go ahead and use a width of 33.333%, because this is going to get us closer to that 100% that we're looking for. Now, our next step is to begin adding that class name to the various sections of our page.



CSS BOX-SIZING PROPERTY

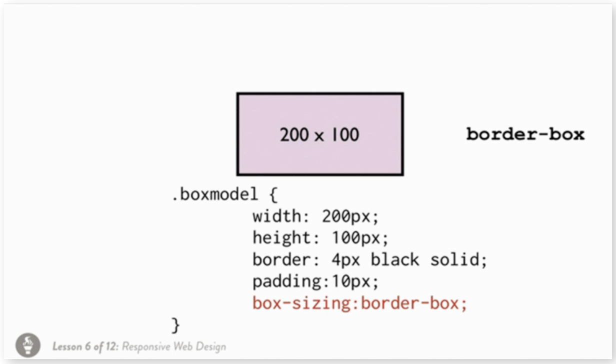
Allows you to switch the type of box sizing being used:

```
content-box
border-box
```

Lesson 6 of 12: Responsive Web Design

So the first one will be for the section ID equals nav, so let's go ahead and add this class equals wrapper. After you add that, you can actually copy that whole attribute and value so we can paste it elsewhere. And the next place we're going to paste it is here for header ID equals masthead. Let's paste that there.

Scroll down a little bit. So let's go ahead and put it here in the header from My Portfolio. We also need to put it in the article for our image thumbnails. Now, we already have a class called group here, so we can simply add wrapper to that by typing wrapper. So we're going to have to do it here for the second article, and then again here for the third.



```
.boxmodel {
        width: 200px;
        height: 100px;
        border: 4px black solid;
        padding:10px;
        box-sizing:border-box;
}
```

Lesson 6 of 12: Responsive Web Design

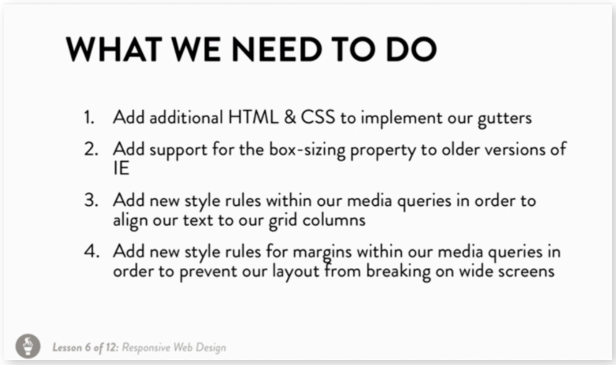
Now, moving on to the My Resume section, let's go ahead and add the wrapper to the header here. And then we're going to add it to the three articles, just like we did previously. So article, class equals wrapper, class equals wrapper for the second one, and finally, class equals wrapper for the third one. A few other details-- let's go ahead and add it to this footer here.

And one more time for the footer at the very bottom of the page. Technically, Footer ID equals colophon can also be class equals wrapper. So let's save our page. Let's take a look at how this works. Let's actually load this in a new tab.

And we're going to align this here. And there's the previous. And here's the new. There's the previous-- no gutters-- new-- gutters.

So we can now see that we have these fixed gutters between each of our columns. And this is looking mighty nice and exactly what we want. Let's go ahead and play around with the size of the page a little bit by changing the browser width and resizing. We can see that those fixed gutters are doing exactly what they're supposed to.

They're staying fixed, and this gives us a solid layout, regardless of the browser window. So box-sizing-- pretty cool, right? It is. There's a small catch, however.

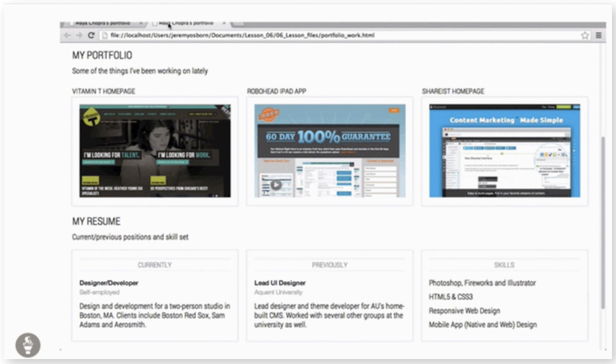


WHAT WE NEED TO DO

1. Add additional HTML & CSS to implement our gutters
2. Add support for the box-sizing property to older versions of IE
3. Add new style rules within our media queries in order to align our text to our grid columns
4. Add new style rules for margins within our media queries in order to prevent our layout from breaking on wide screens

Lesson 6 of 12: Responsive Web Design

Here is one of my favorite pages, caniuse.com. What we're seeing here is that box sizing has pretty good support, actually, in most browsers. But what I like to do is check html5please.com, which is another great website, because this gives me the skinny.


file://localhost/Users/jeremyosborn/Documents/Lesson_06/06_Lesson_files/portfolio_work.html

MY PORTFOLIO

Some of the things I've been working on lately

VITAMIN T HOMEPAGE

ROBOHEAD IPAD APP

SHAREIST HOMEPAGE

MY RESUME

Current/previous positions and skill set

CURRENTLY

Designer/Developer

Self-employed

Design and development for a two-person studio in Boston, MA. Clients include Boston Red Sox, Sam Adams and Aerosmith.

PREVIOUSLY

Lead UI Designer

Aquent University

Lead designer and theme developer for AU's home-built CMS. Worked with several other groups at the university as well.

SKILLS

Photoshop, Fireworks and Illustrator

HTML5 & CSS3

Responsive Web Design

Mobile App (Native and Web) Design

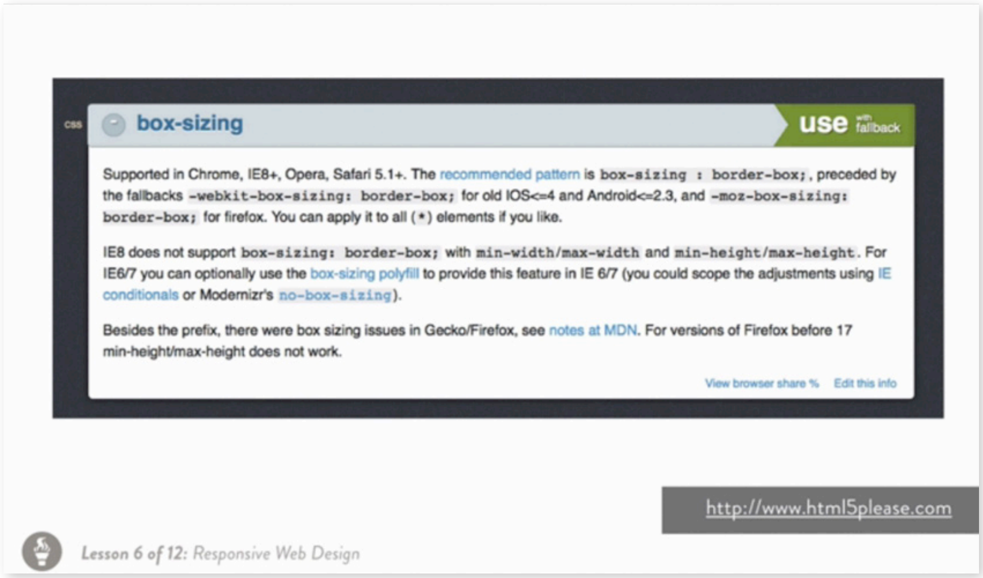
Can I use this, or not? And it says yes, use it with a fallback. Why do we need a fallback? Well, we have IE 8, 7, and 6 that we need to pay attention to. These either do not support box-sizing at all or support it poorly.

So we're going to have to add some code in in order to do this. Again, we want to go back to our codetocopy.txt file. If you want to do a little bit of research about the code that we're using on your own time, here's the address for this on GitHub. This is a box-sizing polyfill.

Again, this will essentially support box-sizing in IE8, 7, and 6, specifically 7 and below. So let's go ahead and copy and paste that code. So everything from here on line 51 down to 63. You want to copy that.

Let's put this in our external stylesheet. And I'll put it right here above our media queries. So we'll paste it here. So one thing I want to point out, this URL is presently a dummy URL. In order to prevent a bug, you'll need to make sure that this path is absolute. You're going to need to put the name of your own website in here to replace absolutepath.com to make sure that this polyfill is working correctly.

So in other words, no relative paths here. With support for IE put into place, let's go ahead and take a look at what we now want to do. So we've got this nice grid structure that's working. Let's pay attention to some of the other elements on our page, particularly I'm interested in this long block of text here where it says a Boston-based designer developer hybrid.


CSS box-sizing — use with fallback

Supported in Chrome, IE8+, Opera, Safari 5.1+. The recommended pattern is `box-sizing : border-box;`, preceded by the fallbacks `-webkit-box-sizing: border-box;` for old IOS<=4 and Android<=2.3, and `-moz-box-sizing: border-box;` for firefox. You can apply it to all (`*`) elements if you like.

IE8 does not support `box-sizing: border-box;` with `min-width/max-width` and `min-height/max-height`. For IE6/7 you can optionally use the box-sizing polyfill to provide this feature in IE 6/7 (you could scope the adjustments using IE conditionals or Modernizr's `no-box-sizing`).

Besides the prefix, there were box sizing issues in Gecko/Firefox, see notes at MDN. For versions of Firefox before 17 min-height/max-height does not work.

View browser share % Edit this info

http://www.html5please.com

Lesson 6 of 12: Responsive Web Design

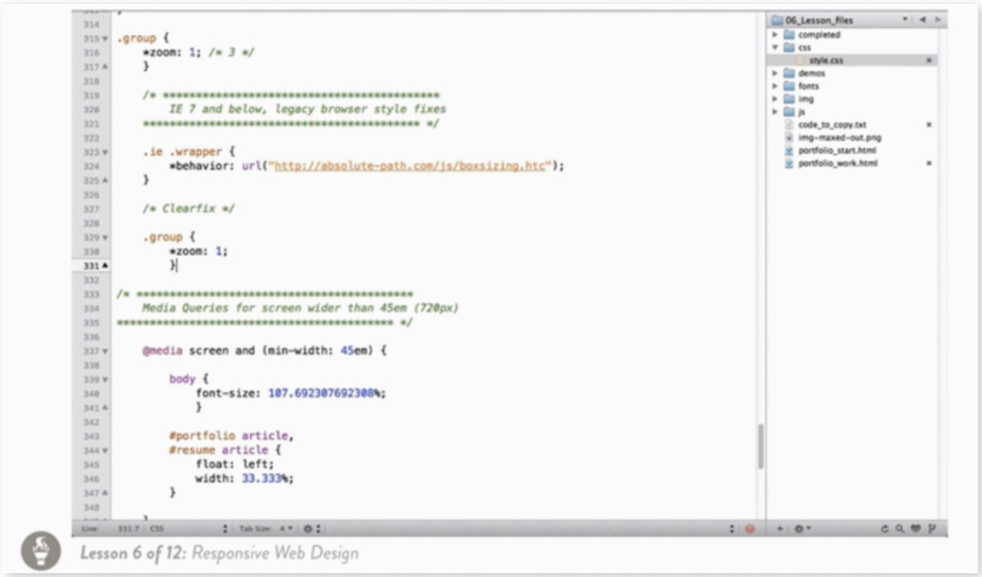
So this long line of text does not work particularly well either in the single column view, but particularly in the three column view. So we're going to do a little bit of work on this by changing the width of that ID, which happens to be called ID masthead. We can do this in the internal stylesheet for now.

Let's go ahead and type the following-- #masthead. And let's just play around with this a little bit. We'll start with a width of 90% and see if that makes any effect in this single column. So save that, open it in your browser, and let's just compare the two-- no, no difference.

So let's go back and shave this off by 10% and make a width of 80. Make sure you save your file. So we'll reload and compare-- a little nicer. So a width of 80% works with the single column view. What about the widescreen view?

And the answer is not so great. The ideal breakpoint for that would be maybe around the second column. So in order to add this, we're going to need to go to our media query. Let's go ahead and open up style.css.

So let's go ahead and locate our media query. And we can use the same selector masthead. But the property we're going to use here is actually max-width instead of width. And instead of a pixel number, we're going to use an em value.

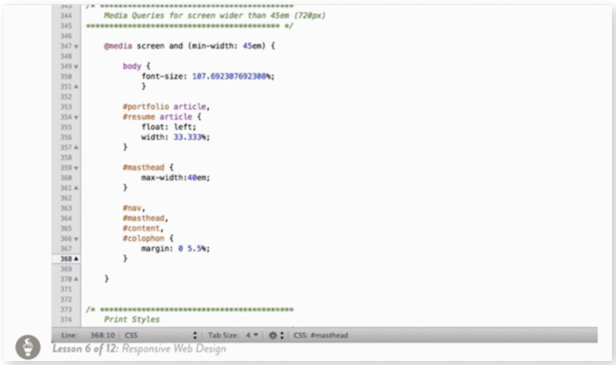


```
.group {
    *zoom: 1; /* 3 */
    }

    /* ********************************************
        IE 7 and below, legacy browser style fixes
    ******************************************** */

    .ie .wrapper {
        *behavior: url("http://absolute-path.com/js/boxsizing.htc");
    }

    /* Clearfix */

    .group {
        *zoom: 1;
        }

/* ********************************************
    Media Queries for screen wider than 45em (720px)
******************************************** */

    @media screen and (min-width: 45em) {

        body {
            font-size: 107.692307692308%;
            }

        #portfolio article,
        #resume article {
            float: left;
            width: 33.333%;
        }
```

One of the reasons we want to do this is we're using a minimum width of 45 em's to begin with inside of this media query.

So we should be consistent. So let's see how this works by loading this in a new browser, and compare the two. Looking much better-- this is now aligning well with the second column on most larger screens. So this is great. We're combining the advantages that we created earlier with our text with the new ones that we just created with our fixed gutters.

Now, we can't neglect the other gutters as well. We talked earlier about how there's four gutters that we have to account for. We just took care of two of them. So for the next step, we need to take care of these outer gutters. And let's do that now.



```
    Media Queries for screen wider than 45em (720px)
******************************************** */

    @media screen and (min-width: 45em) {

        body {
            font-size: 107.692307692308%;
            }

        #portfolio article,
        #resume article {
            float: left;
            width: 33.333%;
        }

        #masthead {
            max-width:40em;
        }

        #nav,
        #masthead,
        #content,
        #colophon {
            margin: 0 5.5%;
        }

    }


/* ********************************************
    Print Styles
```

Before we do this, I just want to point out something I'm doing for this particular video clip. What I've done is recorded this at 1680 by 1050, which is much different than the resolutions that I traditionally record these broadcasts in. The reason for this is so I can demonstrate to you the widest screen resolution versus some of the smaller screens.

So first, I'm going to start in the single column smartphone view. I'm going to expand the width of the browser. And what we want to pay attention to are the outer gutters-- so the ones on the left-hand side of the browser and the ones on the right. These currently are not fixed. These are fluid.

So we want to make them fixed. To do that, let's go into our text file, and let's find this group of rules here for nav, masthead, content, and colophon. Notice that these have a top and bottom margin of zero and a left and a right margin of 3.3%. Let's go ahead and copy those styles.

And in this instance, we actually want to put these styles in our external stylesheet, not our internal stylesheet. So go ahead and open up your external stylesheet. I'm going to put these styles right here after the image style around line 309. Let's go ahead and paste that. Lets save our page.

Let's reload this in a new browser window. And you're going to see that the difference here is relatively subtle. So it does shift when we toggle back and forth. And what we're really interested is making sure that this follows through for our wider screens. And this isn't perfect.

So 3.3% is not going to work on our larger screen. So let's go ahead and add the same styles to our media query section, but let's make the margins a little bigger-- 5.5% instead of 3.3%. So let's copy that from our text file. Locate the media query section, paste.

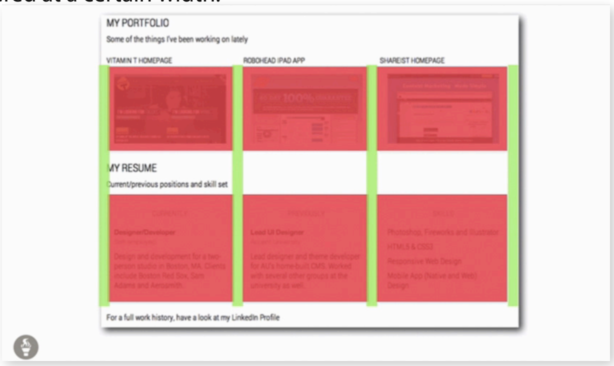
Again, margin's 5.5%. Save your page. Let's open this in a new browser, and then we can see the difference here. Now again, the differences here are relatively subtle, but most apparent on the widest screens. So we can see when we toggle back and forth between browser tabs that there's a slight shift. And if we toggle between the new code and if I toggle all the way back to the original code where there is no margins, we can see this actually creates a dramatic difference.

So what's happening here is that by increasing those margins, we're actually improving the layout of our images themselves. And this is a great side benefit. So that's it. We have a very nice and reliable three-column layout.

It centers relatively well, no matter what the size of the screen is. We still have a little bit more work to do with the image part of that. But we'll be doing that in the next lesson. Speaking of the next lesson, we're going to be taking a look at adding additional responsive images soon, as well as adding responsive navigation. We've got a tiny bit of housekeeping to do before that, and we're going to be adding a little bit of review of all the concepts that we've covered.

But these are the major components of what we need to do. As always, here's some homework for you, a short quiz to help you test your knowledge and reinforce the concepts that we talked about. For assignment number two, there's a great article on box-sizing by Paul Irish, and here's the URL.

So I'd like you to read that. Additionally, inside your lesson files, there is a document called index. html, located within the fluid-fixed folder. What I'd like you do is to open up this file and look at the code. This file shows a fluid grid that becomes fixed and centered at a certain width.


MY PORTFOLIO
Some of the things I've been working on lately
VITAMIN T HOMEPAGE
ROBOHEAD IPAD APP
SHAREIST HOMEPAGE
MY RESUME
Current/previous positions and skill set
For a full work history, have a look at my LinkedIn Profile

And I want to make sure that you take a look at it, understand how it's working, and if you don't, ask questions on the Forum. And of course, assignment number 3-- add some fixed gutters to your portfolio in progress, use box-sizing, add some fixed gutters to your own page, and be sure to experiment with the gutter width and margins, and make sure that the layout is perfect for your content. That's it.

I'll see you in the Forum, and I'll see you in the next session.