

RESPONSIVE BASICS

Eric Boyer

LEARNING OBJECTIVES

- Describe responsive design.
- Know the difference between fluid, fixed and responsive layouts
- Apply media queries to achieve a responsive layout.

AGENDA

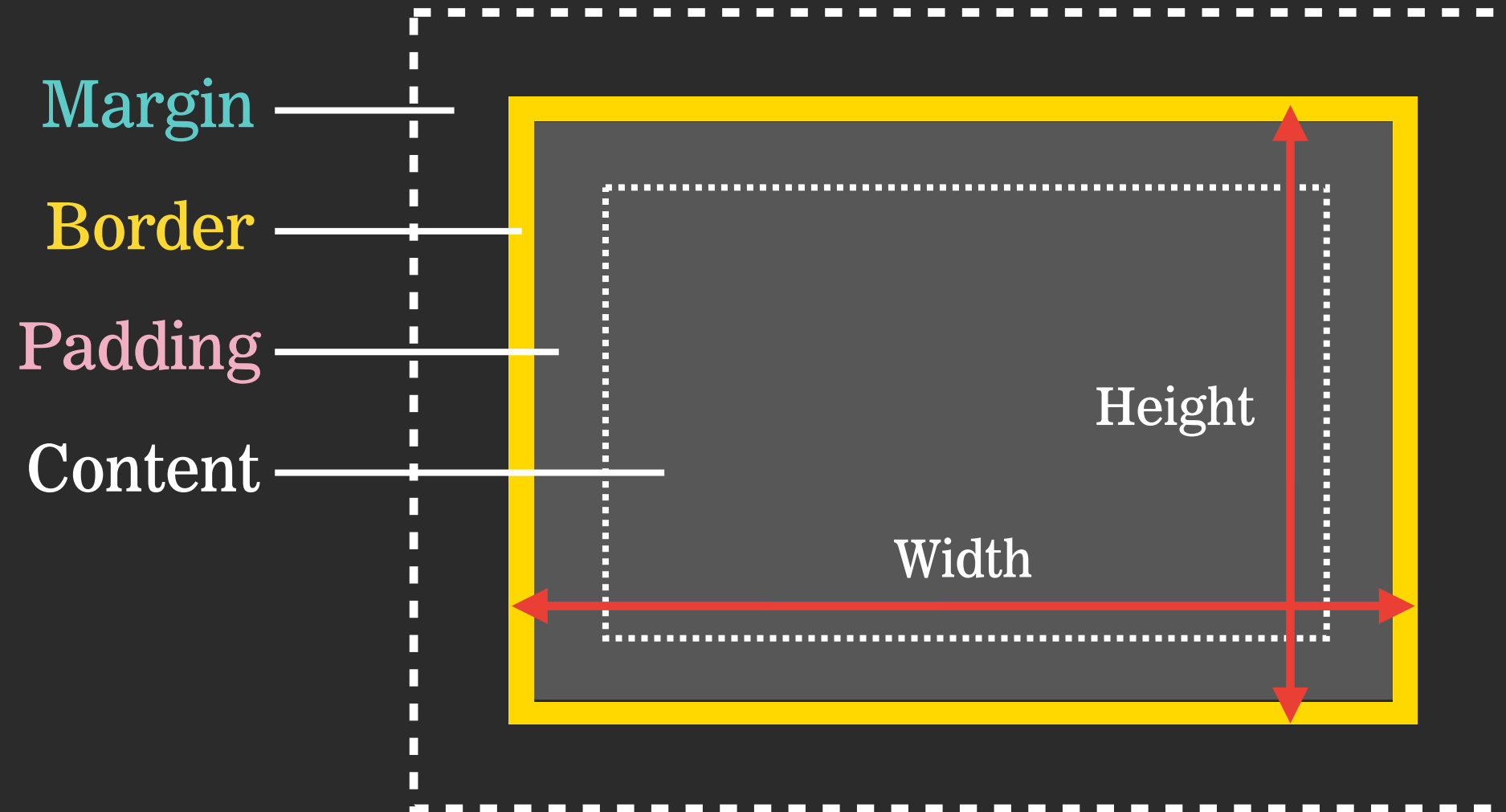


- Review
- Responsive — Layout Design
- Responsive — Media Queries
- Responsive — REM/EM

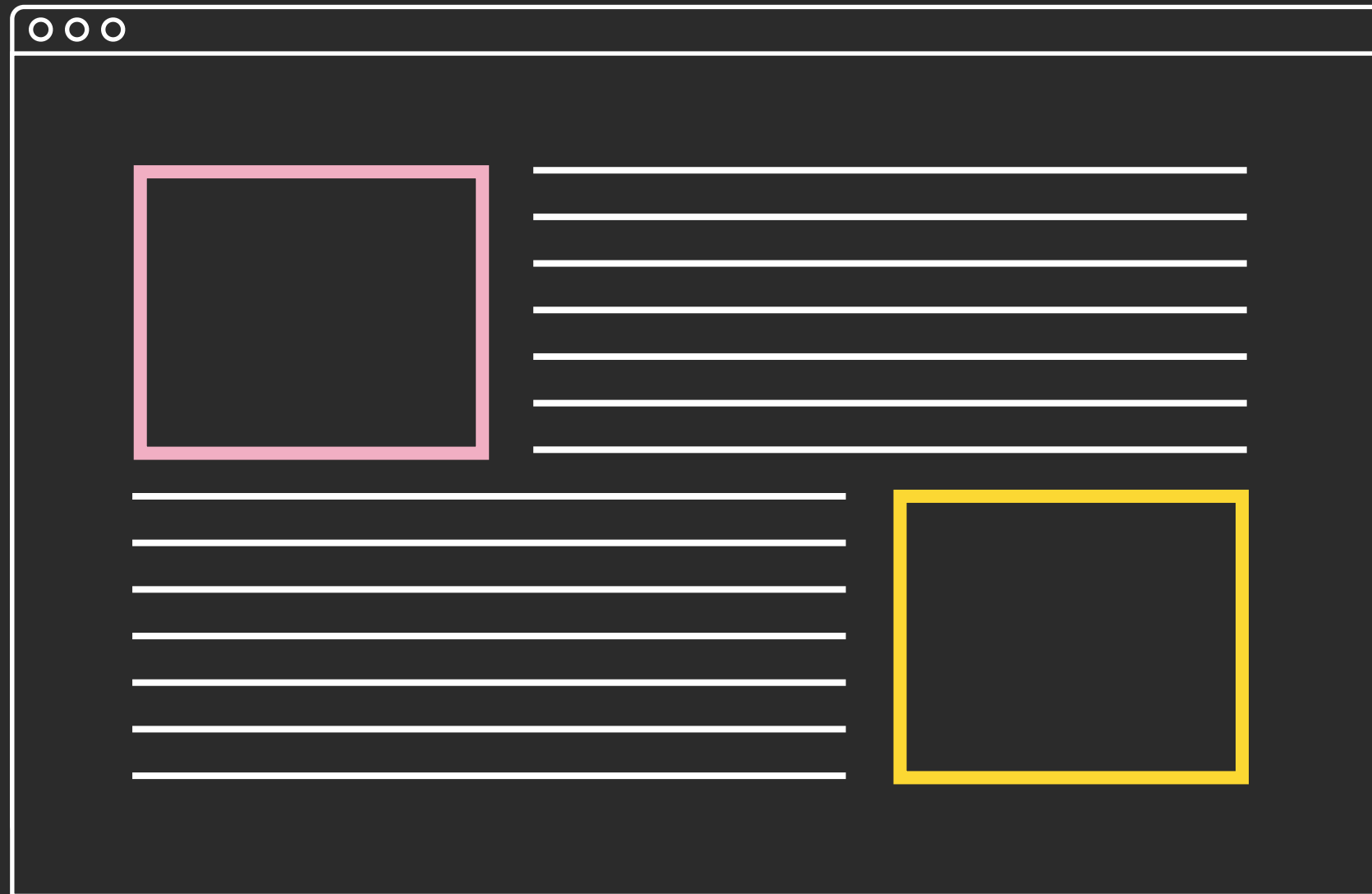
RESPONSIVE BASICS

REVIEW

BOX-SIZING: BORDER-BOX



CSS — FLOATS



PARENTS OF FLOATED ELEMENTS

- ▶ If a containing element **only contains floated elements**, some browsers will treat it as if it is zero pixels tall.

PROBLEM:

1	2	3	4
5	6	7	8



Collapsed parent!

SOLUTION:

1	2	3	4	
5	6	7	8	

PT. 1 — ADD CSS CLASS:

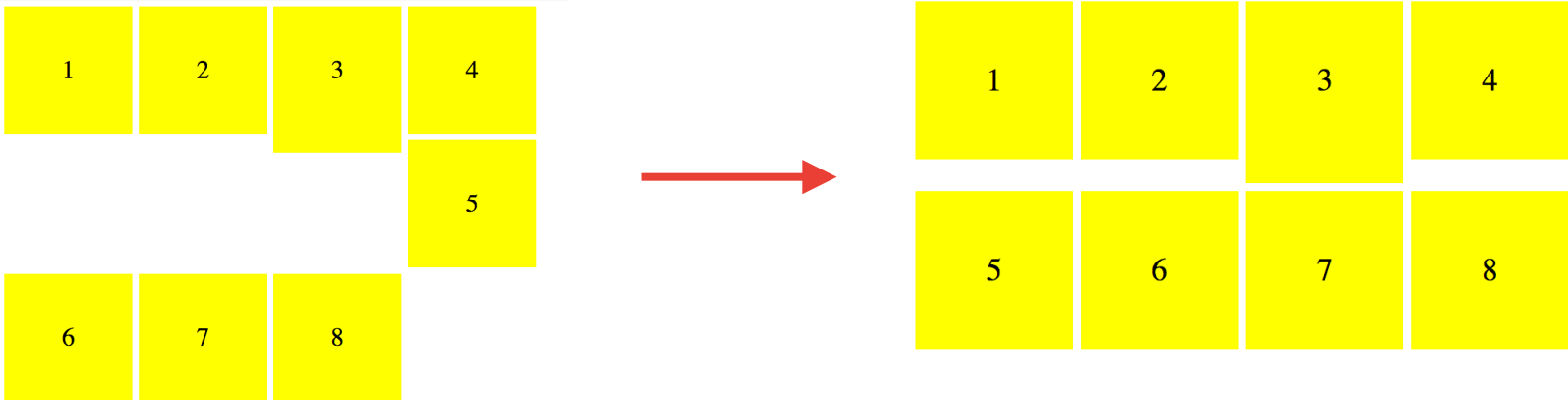
```
.clearfix:after {  
  content: "";  
  display: table;  
  clear: both;  
}
```

PT. 2 — ADD CLASS TO HTML:

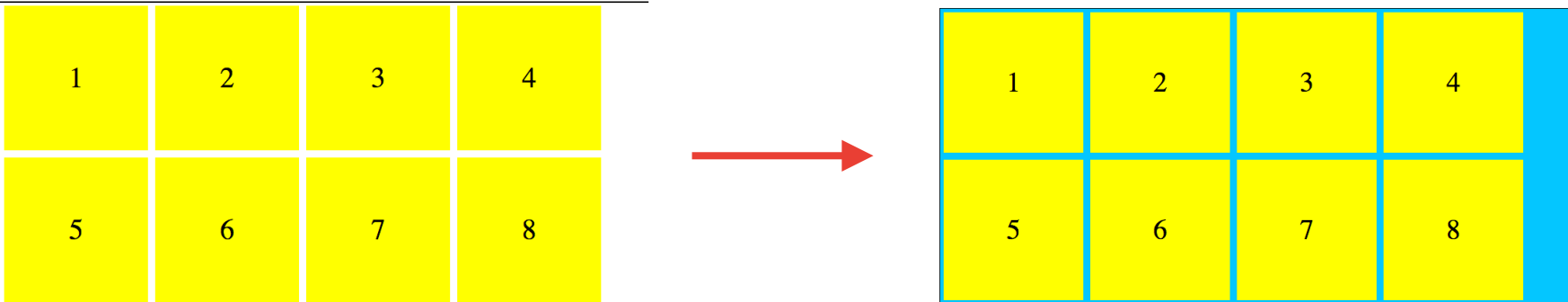
```
<div class="clearfix">  
  <p>1</p> <!-- float: left -->  
  <p>2</p> <!-- float: left -->  
  <p>3</p> <!-- float: left -->  
</div>
```

CONFUSING NAMES — KEEPING THINGS STRAIGHT

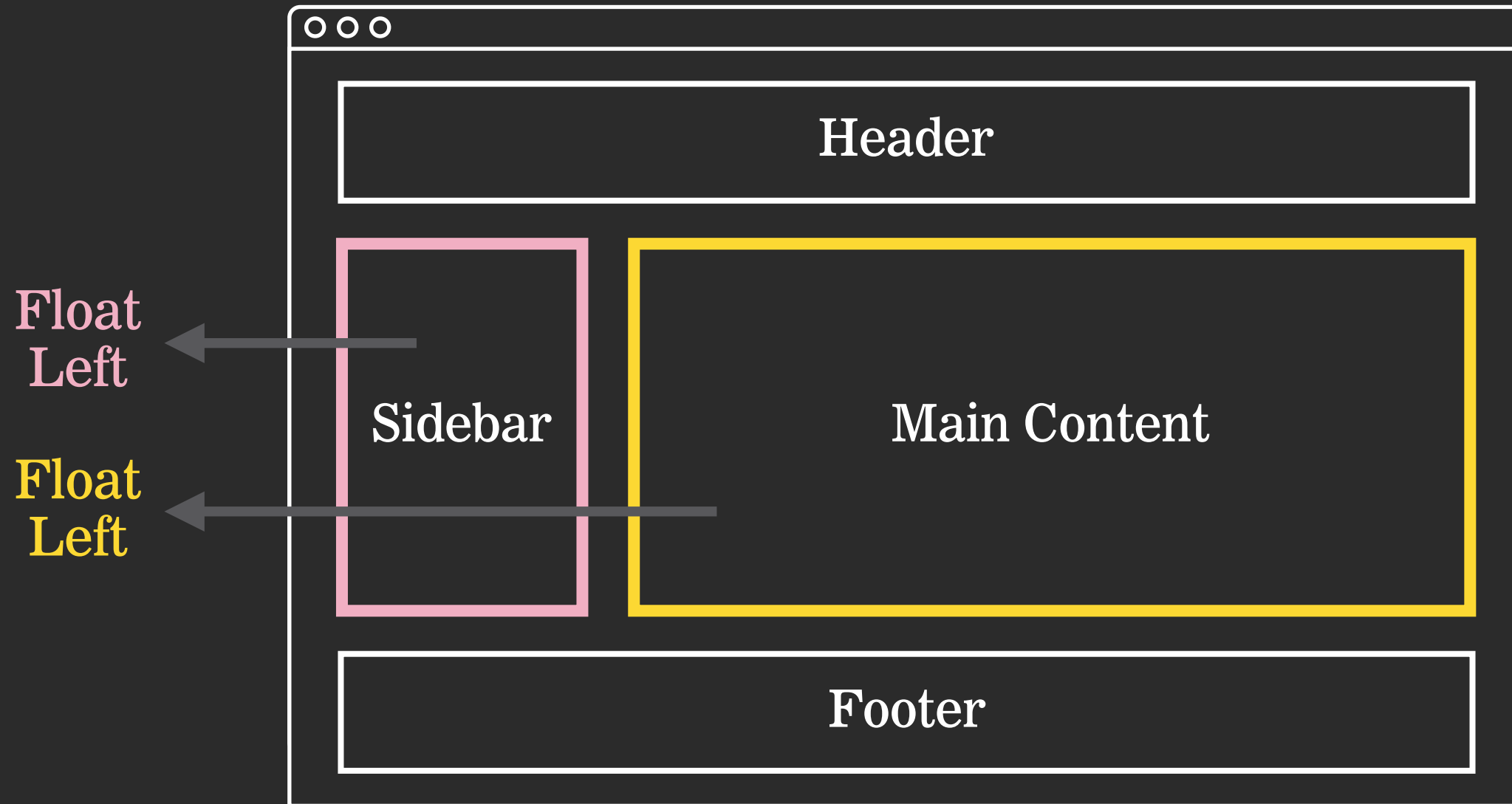
CLEAR: BOTH;
Make sure an element starts on a new line



CLEARFIX:
Fixes collapsed parent



CSS — MULTI-COLUMN LAYOUT



LAB



ACTIVITY



EXERCISE

KEY OBJECTIVE

- ▶ Review HTML/CSS Layouts

TYPE OF EXERCISE

- ▶ Code along w/ Eric to build a layout of boxes

TIMING

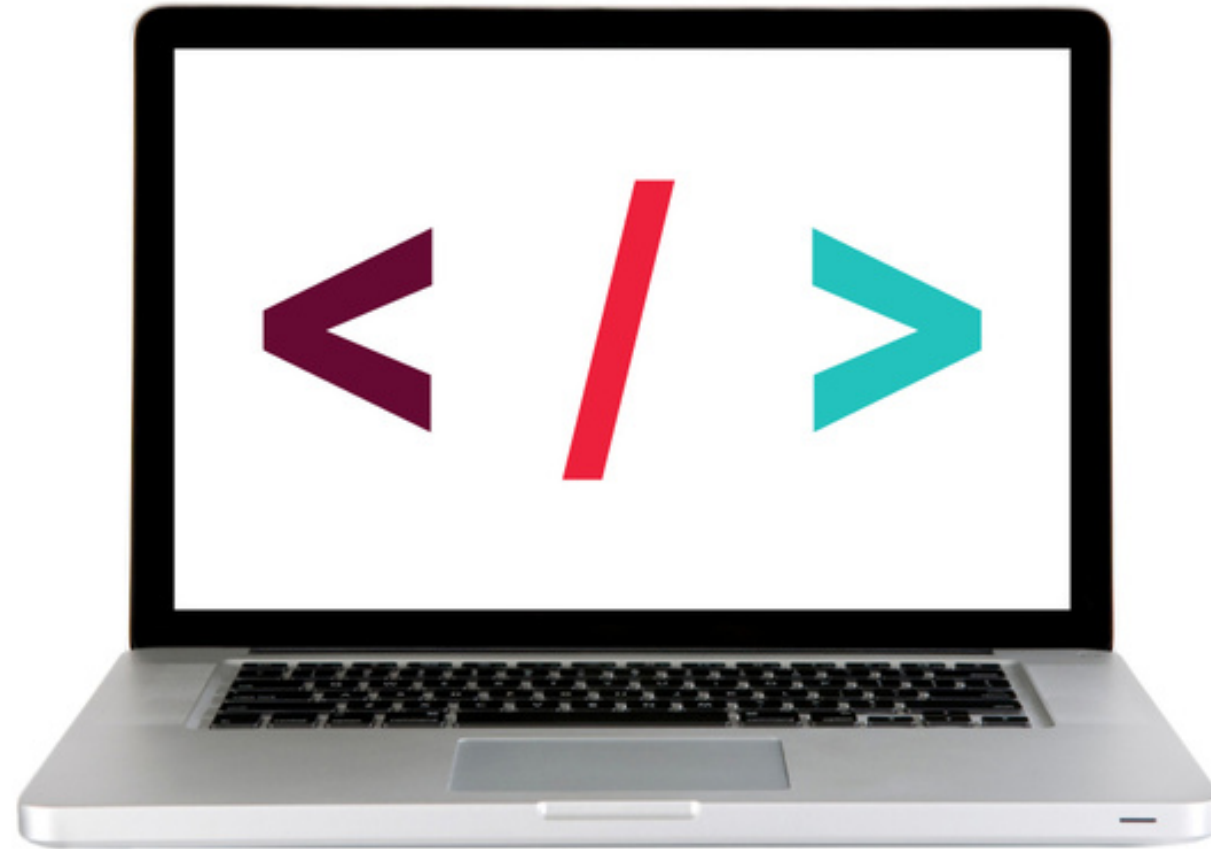
20 min

1. Expand upon the code samples provided to build a layout of boxes

RESPONSIVE BASICS

RESPONSIVE — LAYOUT DESIGN

LET'S TAKE A CLOSER LOOK



<http://stephencaver.com/>

RESPONSIVE DESIGN

“Day by day, the number of devices, platforms, and browsers that need to work with your site grows. Responsive web design represents a fundamental shift in how we’ll build websites for the decade to come.”

- Jeffrey Veen

RESPONSIVE DESIGN



RESPONSIVE DESIGN



LAB



ACTIVITY



EXERCISE

KEY OBJECTIVE

- ▶ Use HTML/CSS to create a mobile layout

TYPE OF EXERCISE

- ▶ Code along w/ Eric to overwrite the layout to be 100% width

TIMING

20 min

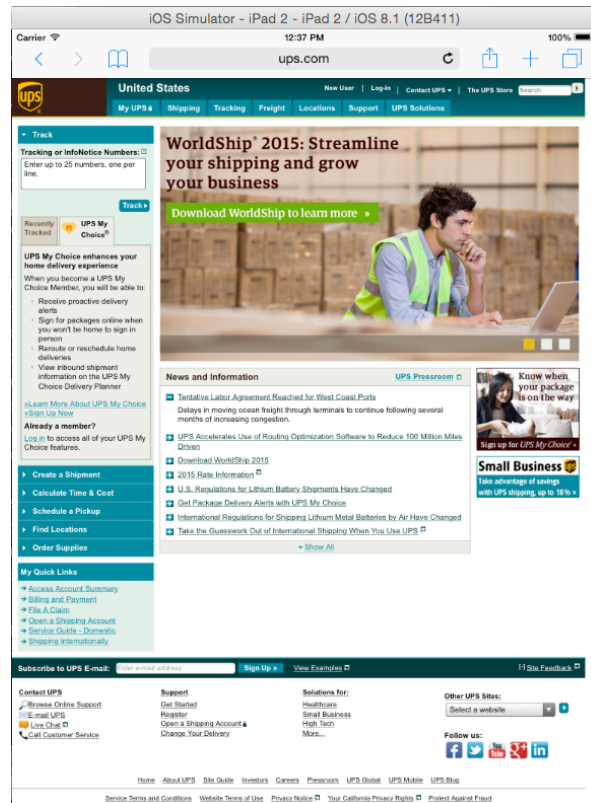
1. Open the style.css file from the first exercise and begin writing CSS to overwrite the previous layout at the **bottom**

RESPONSIVE — TYPES OF LAYOUTS

FIXED VS. RESPONSIVE

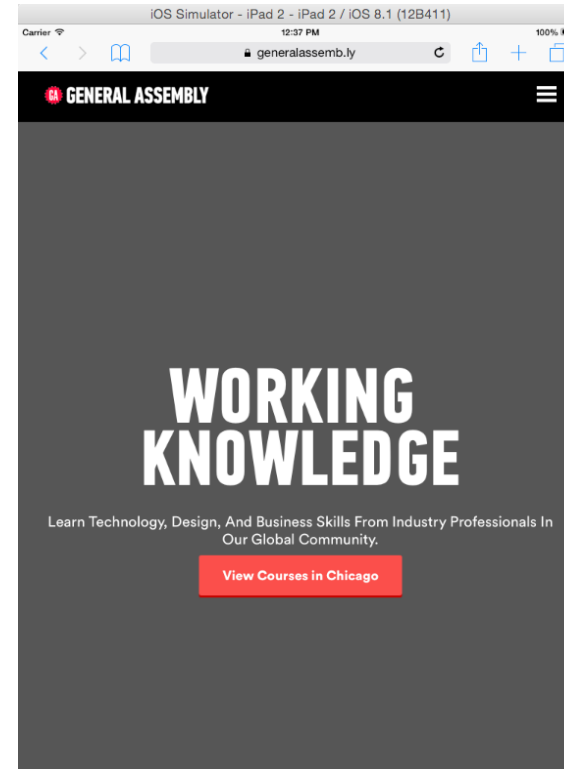
CHECK OUT THESE FIXED SITES:

- ▶ ups.com
- ▶ colourpixel.com



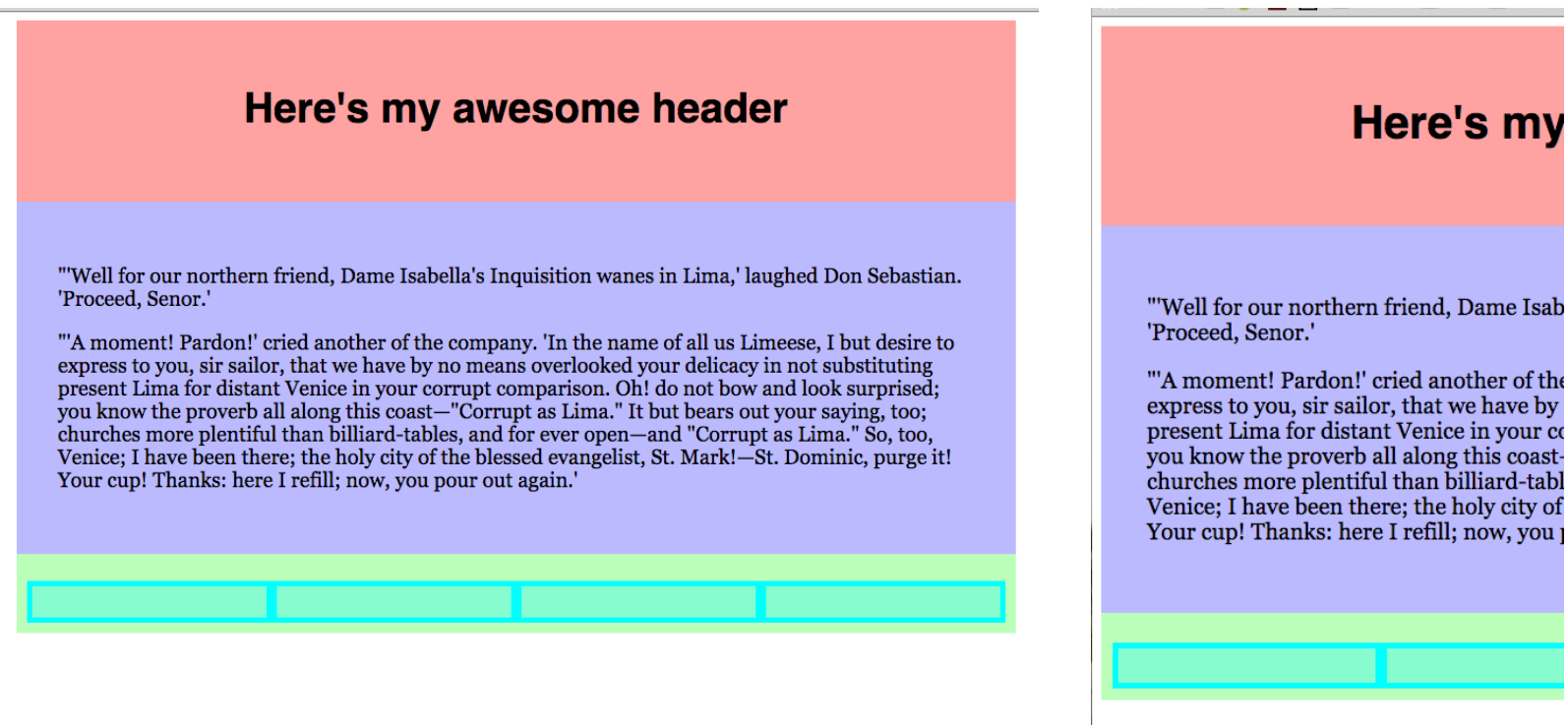
CHECK OUT THESE RESPONSIVE SITES:

- ▶ GeneralAssemb.ly
- ▶ KinHR.com



FIXED LAYOUT

- Relies on a container of a fixed width (uses static units)
- Resizing the browser/viewing it on a different device won't have an effect on the page

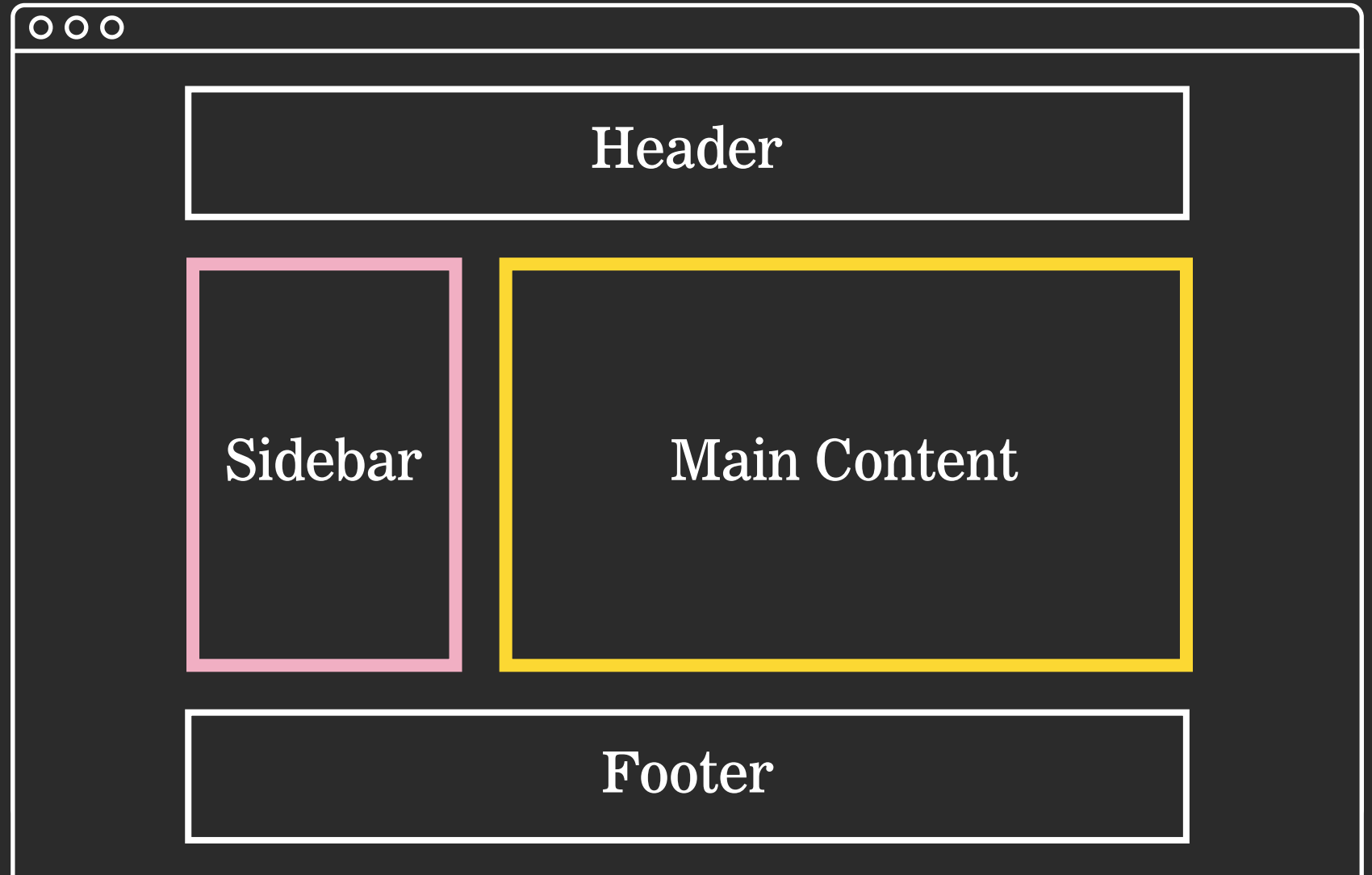


FIXED WIDTH LAYOUT

Fixed width layouts do not change size as the user increases/decreases width of browser window

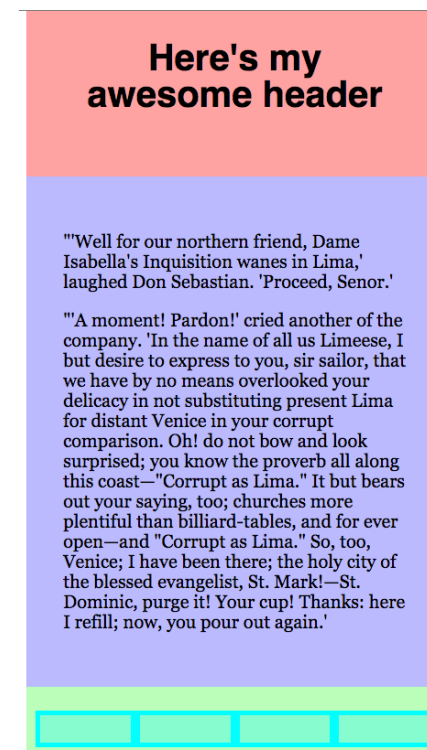
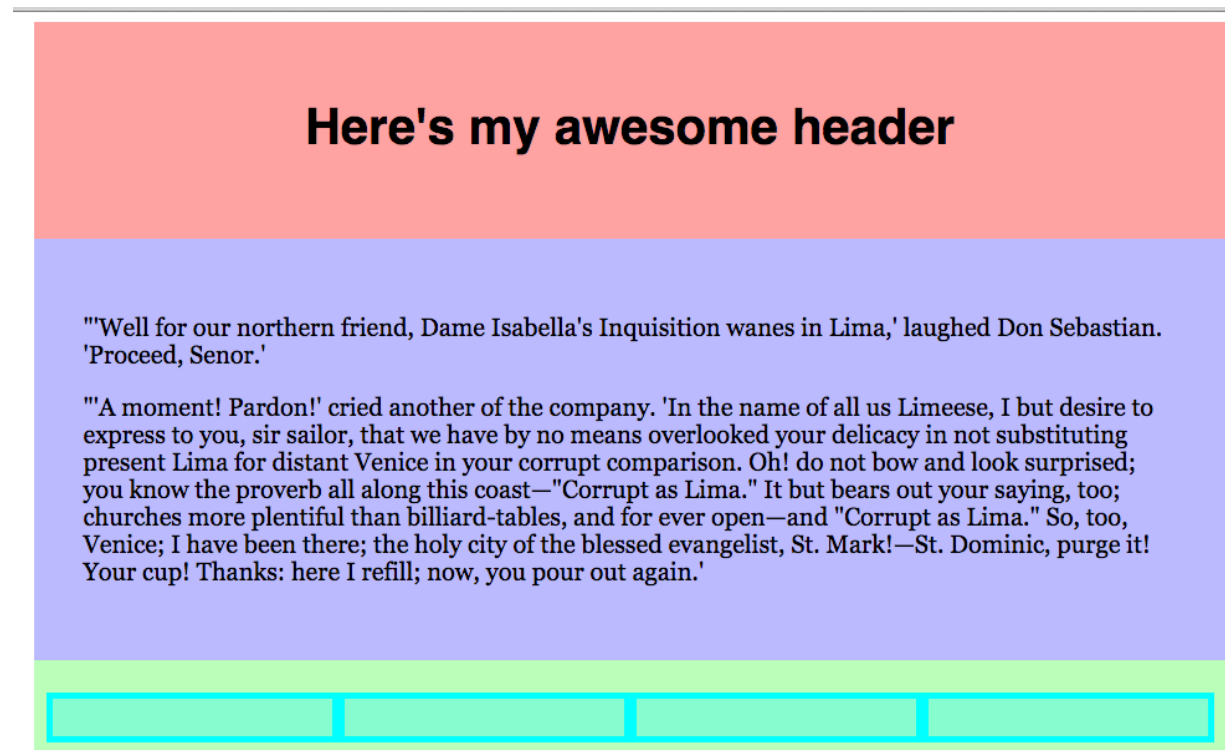
To create:

- Width of any main boxes is set in pixels
- Layout can be centered by setting the value of the left and right margins to auto



FLUID LAYOUT

- Uses relative widths (percentages)
- No media queries

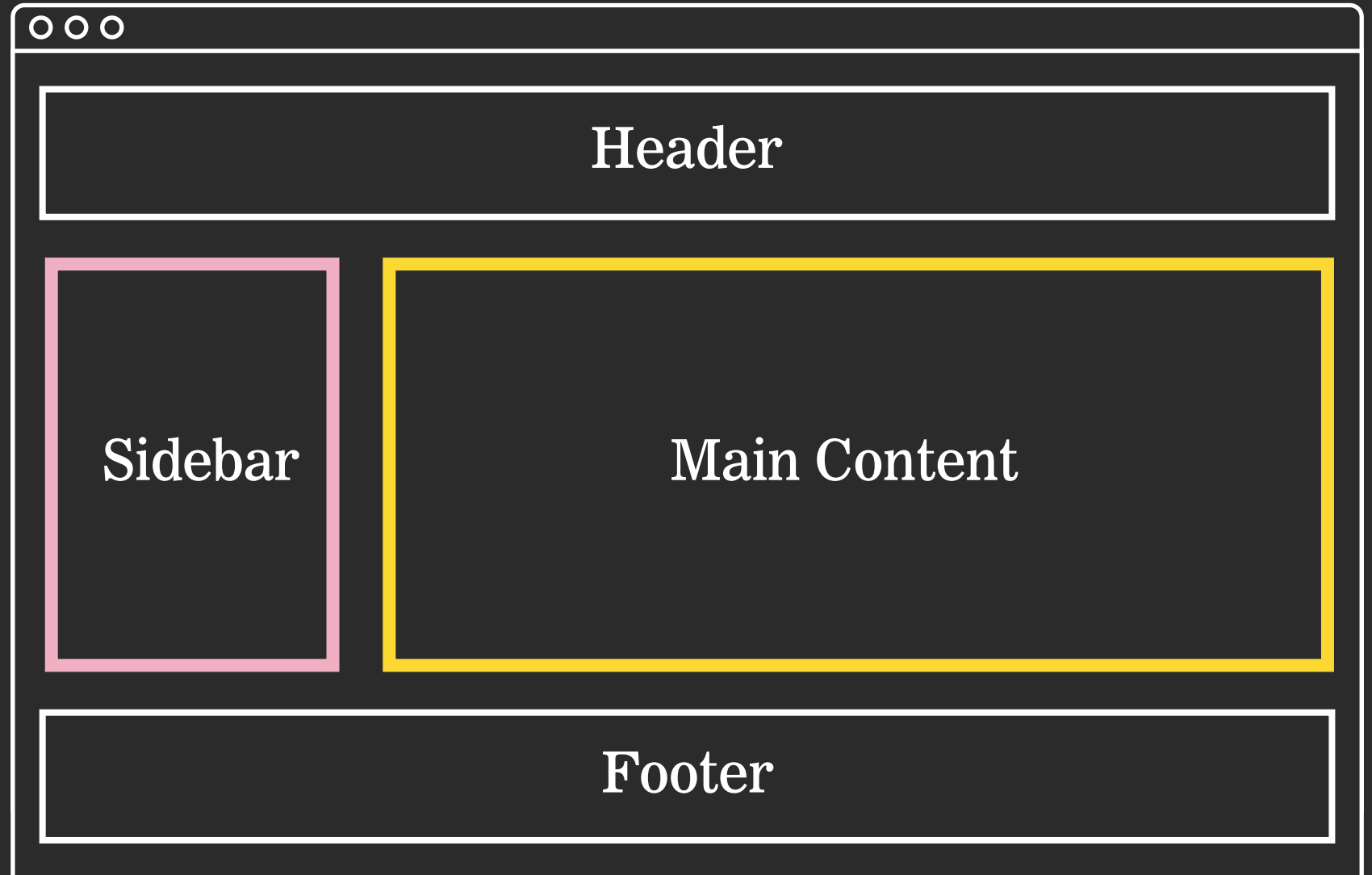


FLUID LAYOUT

Fluid layouts stretch and contract as the user increases/decreases the size of their browser window

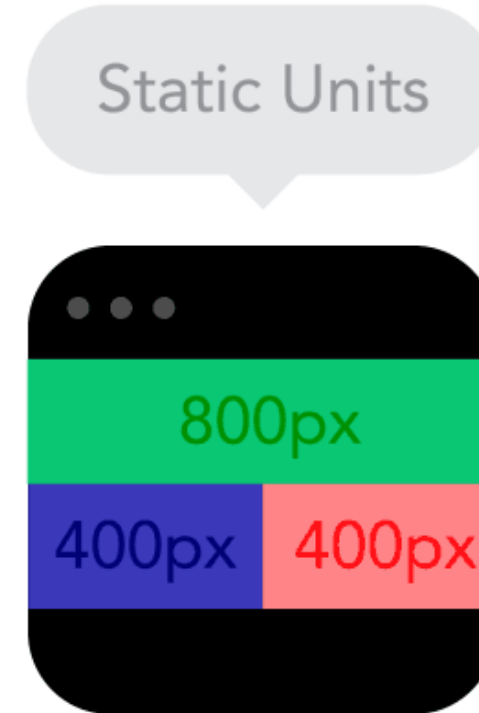
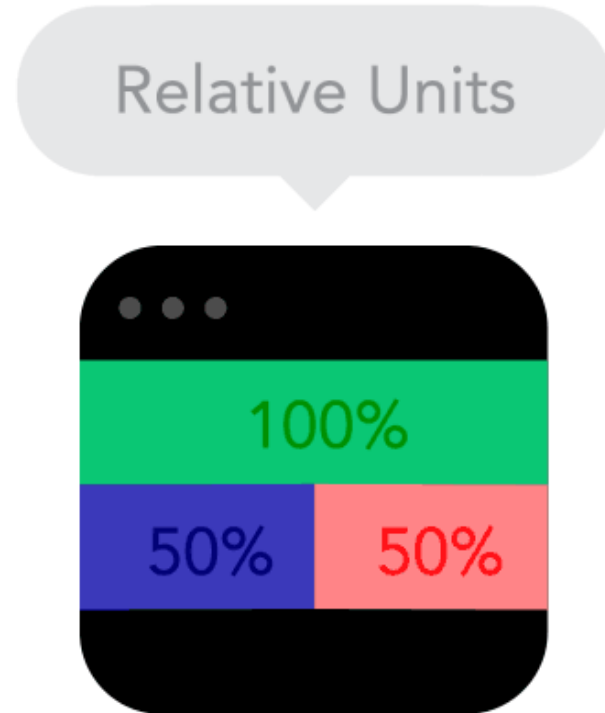
To create:

- Uses percentages to set the width of each box so that the design will stretch to fit the size of the screen



FIXED VS. FLUID

Fluid layout

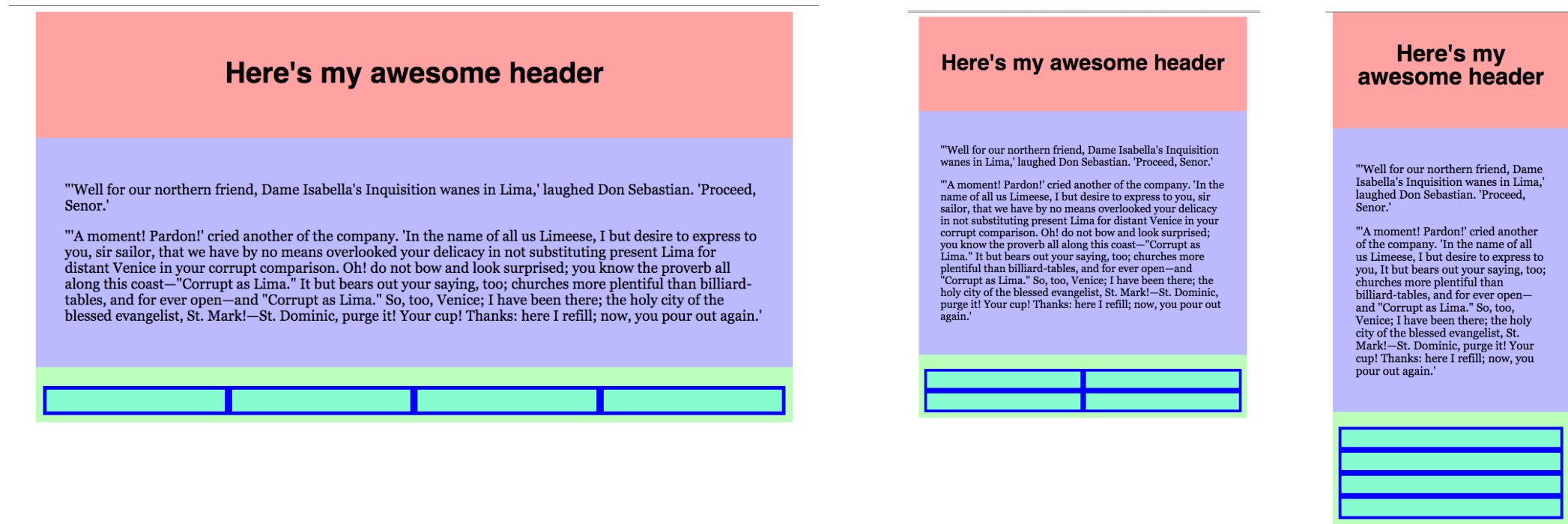


Fixed

Gif credit: [Fast Company](#)

RESPONSIVE LAYOUT

- Uses relative widths (built on a fluid grid)
- Use media queries to control design and content as it scales down or up with the browser or device



WITH BREAKPOINTS VS. WITHOUT BREAKPOINTS

With Breakpoints

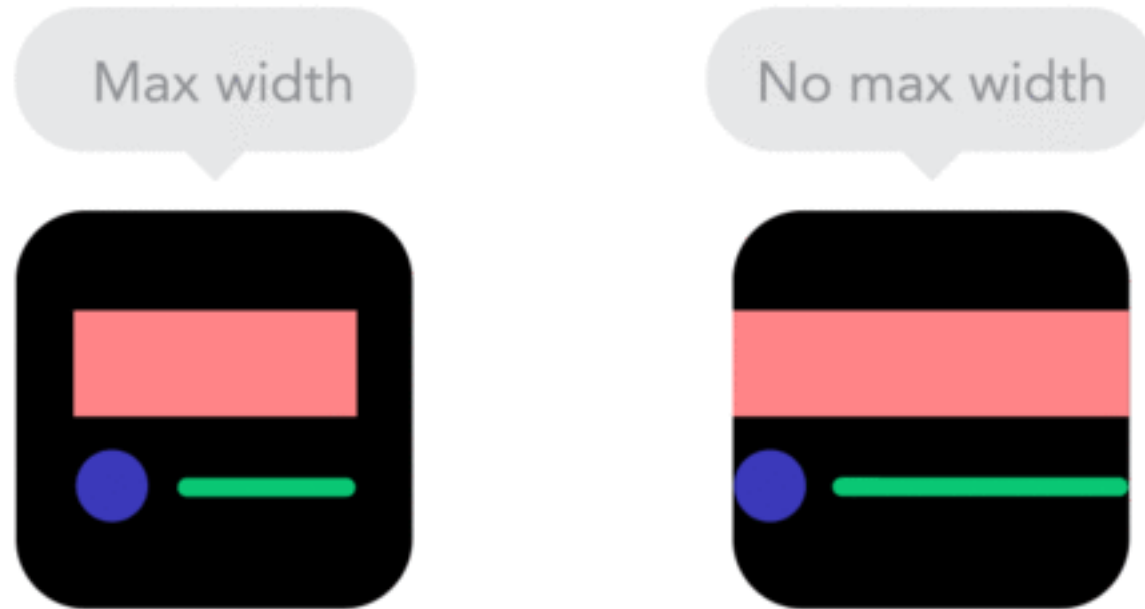


Without Breakpoints



Gif credit: [Fast Company](#)

MAX-WIDTH — A HELPFUL TOOL FOR LAYOUT



Gif credit: [Fast Company](#)

ACTIVITY



EXERCISE

KEY OBJECTIVE

Describe the difference between fixed, fluid and responsive layouts.

TIMING

1 min

1. Turn to partner and discuss

2 min

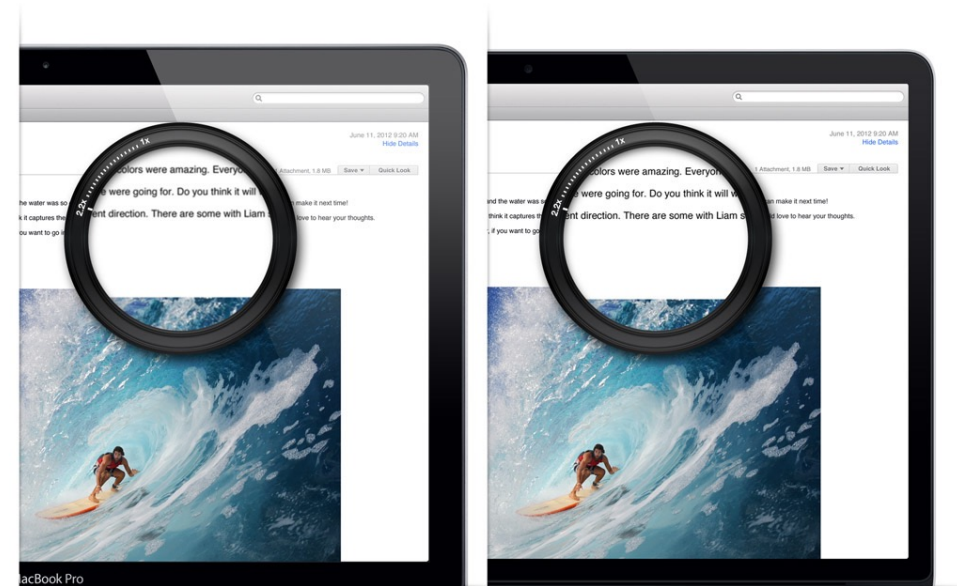
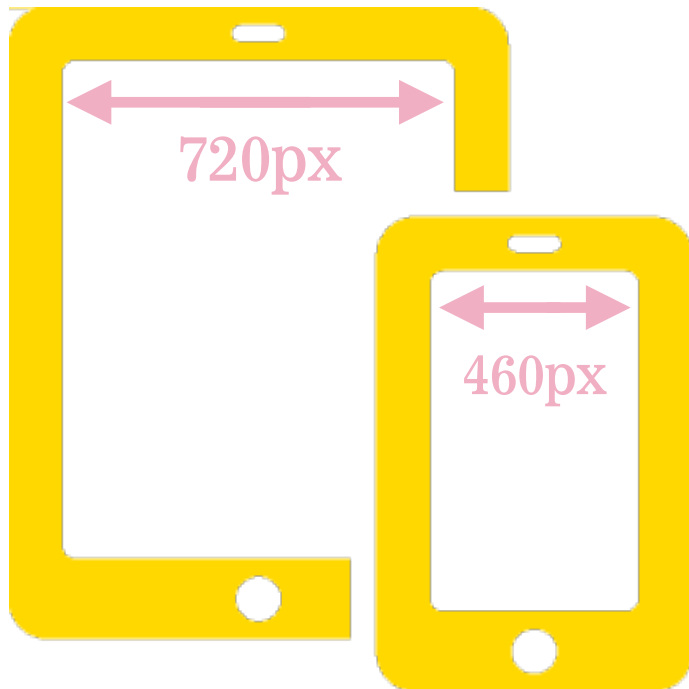
2. Share with rest of class

RESPONSIVE BASICS

RESPONSIVE — MEDIA QUERIES

MEDIA QUERIES

- Media queries allow us to target CSS rules based on screen size, device orientation, display density, etc.



MEDIA QUERIES

- ▶ We can use media queries to allow certain rules to apply for an iPad or iPhone, to add styles for a printer, or to create a responsive site.
- ▶ With media queries, we can allow most of our styles to remain the same, while we make **small tweaks for specific formats**.



MEDIA QUERIES — FIRST METHOD

Create separate stylesheets for different devices

For example:

- Have one main stylesheet as the default stylesheet
- If the screen becomes too narrow, short, tall, wide, etc. we can detect that and load in an additional stylesheet

```
<link rel="stylesheet" media="screen and (max-width: 460px)" href="css/iphone.css" />
```

MEDIA QUERIES — SECOND METHOD

Use media queries directly in your CSS

```
@media screen and (max-width: 600px) {  
  .box {  
    width: 100%;  
  }  
}
```

**Usually goes at the end of stylesheet.*

MEDIA QUERIES — SYNTAX

MEDIA TYPES

- **screen**: color computer screen
- **print**: print preview mode
- **all**: suitable for all devices

```
@media screen {  
  /* Styles for color computer screen */  
}
```

```
@media print {  
  /* All your print styles go here */  
  #header, #footer, #nav { display: none !important; }  
}
```

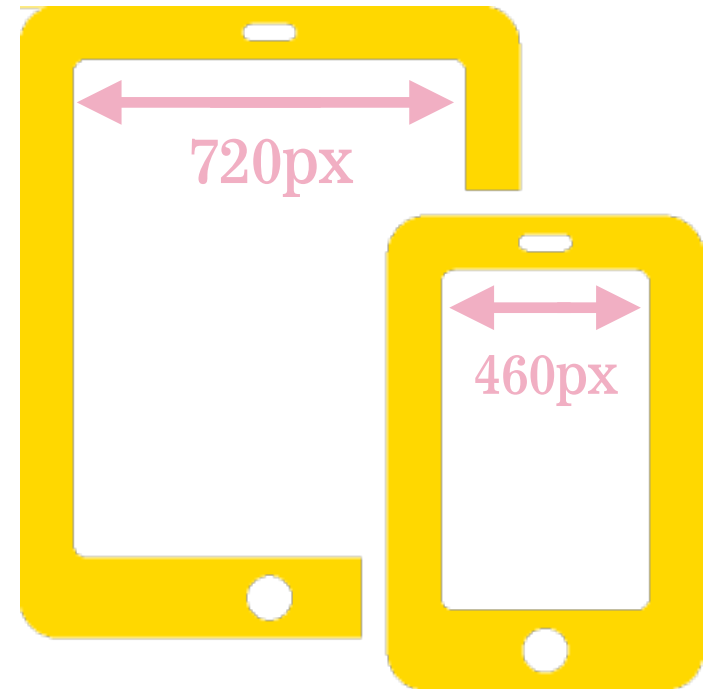
MEDIA QUERIES — SYNTAX

MEDIA FEATURES

- **width:** viewport width
- **height:** viewport height

```
@media screen and (max-width: 600px){  
  /* Styles for screens with a maximum width of 600px */  
}
```

```
@media screen and (min-width: 600px){  
  /* Styles for screens with a minimum width of 600px */  
}
```



***See a full list of features [here](#)

MEDIA QUERIES — SYNTAX

MEDIA FEATURES

- **orientation**: orientation of the viewport

```
@media screen and (orientation: portrait){  
  /* Styles for screens with a maximum width of 600px */  
}
```

```
@media screen and (orientation: landscape){  
  /* Styles for screens with a minimum width of 600px */  
}
```



***See a full list of features [here](#)

MEDIA QUERIES — SYNTAX

LOGICAL OPERATORS

- **and:** can be used to combine multiple media features together, as well as combining media features with media types.

```
@media (min-width: 700px) and (orientation: landscape) { ... }
```

- **comma-separated lists:** behave like the logical operator *or*

```
@media (min-width: 700px), handheld and (orientation: landscape) { ... }
```

- **not:** applies to the whole media query and returns true if the media query would otherwise return false

```
@media not print { ... }
```

- **only:** prevents older browsers that do not support media queries with media features from applying the given styles

```
@media only screen and (min-width: 400px) { ... }
```



VIEWPORT META TAG — AN IMPORTANT NOTE!!

- The viewport meta tag controls how a webpage is displayed on a mobile device.
- Without the tag, mobile devices will assume you want the full desktop experience and will set the viewport width at 980px (iOS)

DEVICE-WIDTH

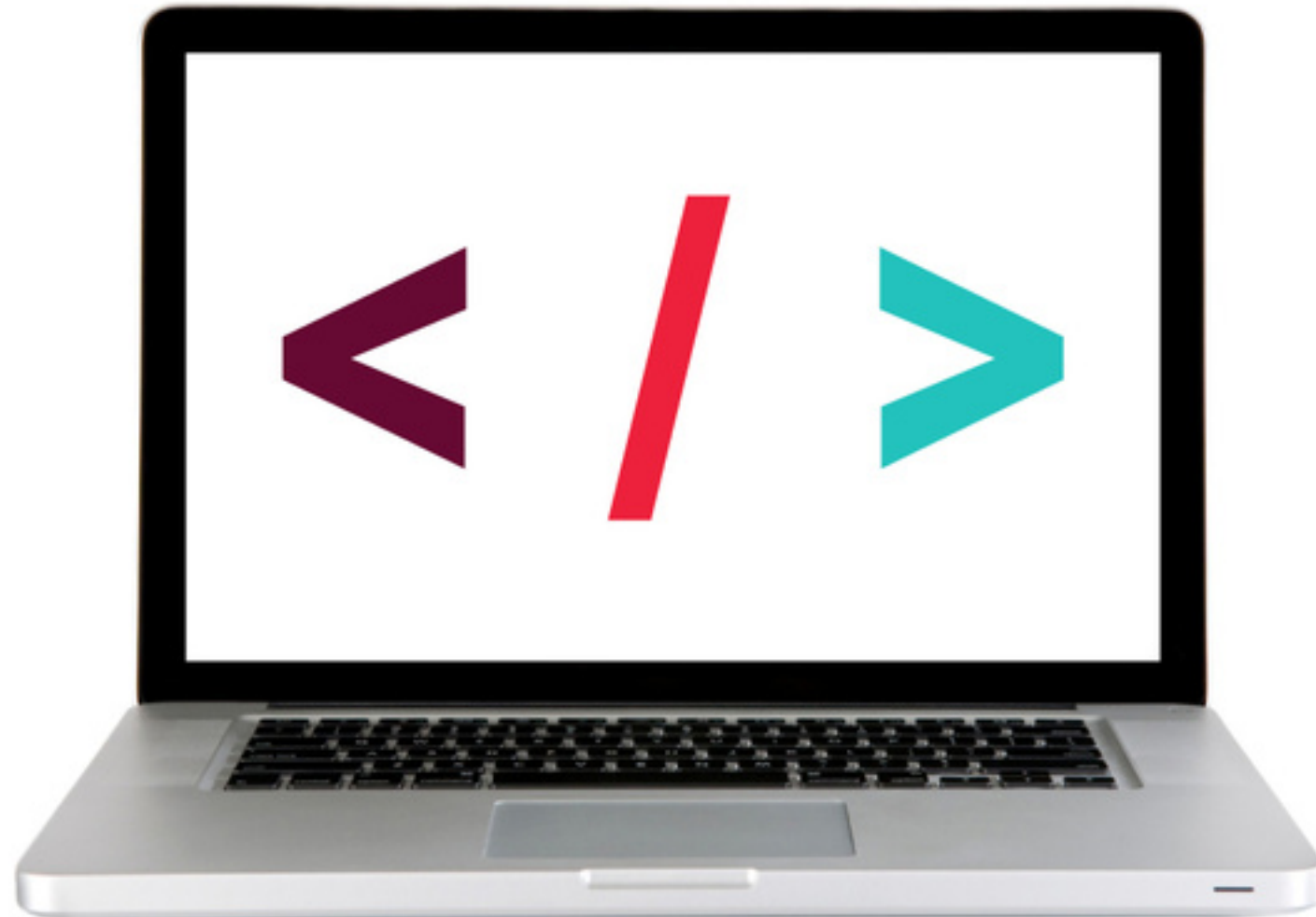
- This tells the browser “My Website adapts to your width”

INITIAL-SCALE

- Sets the initial zoom level and prevents default zooming

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

LET'S TAKE A CLOSER LOOK



LAB



ACTIVITY



EXERCISE

KEY OBJECTIVE

- Apply media queries to achieve a responsive layout.

TYPE OF EXERCISE

Code along w/ Eric to add media queries to bring our 2 projects today together into one.

TIMING

25 min

1. Add media queries to make Boxes exercise responsive.

RESPONSIVE BASICS

RESPONSIVE — REM/EM

EM

- *Relative* unit
- Sized based on the width of the letter “m”
- 1em = 100% font-size
- .5em = 50% font-size
- **Based on parent**

Parent { font-size:16px;}

Child {font-size:2em;}  Child's font size is 32px (200% x 16px)

REM

- "Root" em
- Same as em **except** based on the font-size of the html element

PIXELS AND EMS AND REMS, OH MY!!

	RELATIVE?	BASED ON
PX	absolute	
EM	relative	parent
REM	relative	html element

THE BENEFIT OF USING RELATIVE UNITS

```
html { font-size: 16px; }
h1 { font-size: 33px; }
h2 { font-size: 28px; }
h3 { font-size: 23px; }
h4 { font-size: 19px; }
small { font-size: 13px; }
.box { padding: 20px; }

@media screen and (min-width: 1400px) {
  html { font-size: 20px; }
  h1 { font-size: 41px; }
  h2 { font-size: 35px; }
  h3 { font-size: 29px; }
  h4 { font-size: 24px; }
  small { font-size: 17px; }
  .box { padding: 25px; }
}
```

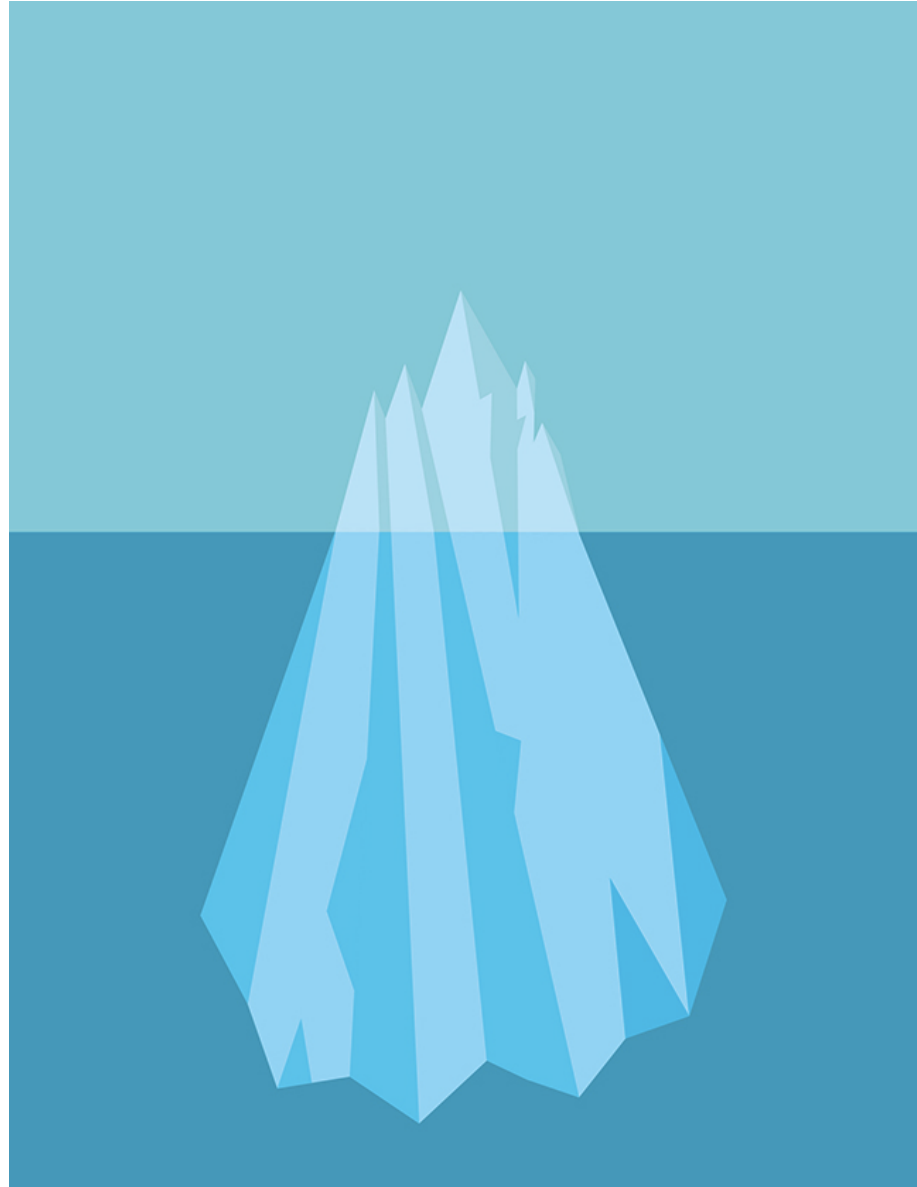
```
html { font-size: 1em; }
h1 { font-size: 2.074em; }
h2 { font-size: 1.728em; }
h3 { font-size: 1.44em; }
h4 { font-size: 1.2em; }
small { font-size: 0.833em; }
.box { padding: 1.25em; }

@media screen and (min-width: 1400px) {
  html { font-size: 1.25em; }
}
```

RESPONSIVE BASICS

MORE RESOURCES

MORE RESOURCES



MORE RESOURCES — THIS IS RESPONSIVE



This Is Responsive.

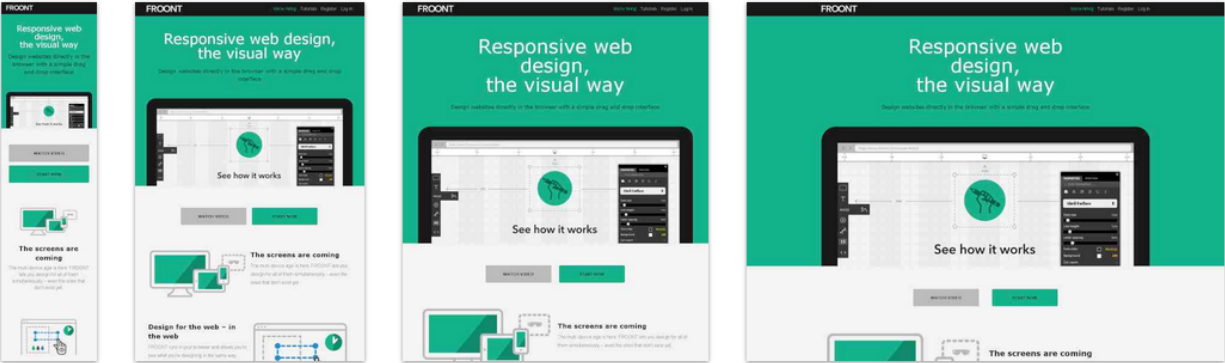
Patterns, resources and news for creating responsive web experiences.

MORE RESOURCES — MEDIA QUERIES



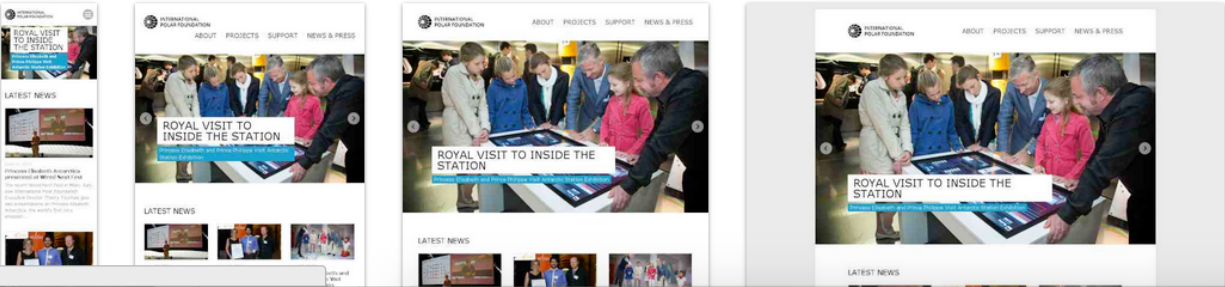
FROONT

13




International Polar Foundation

1



MORE RESOURCES — REMS/EMS

Jeremy Church

[/ index](#) [/ about](#) [/ contact](#) 

February 24, 2014

Confused About REM and EM?

REM can be confusing, especially without a solid understanding of its partner EM and their archvillain, the PX.

Relative Units

Both *rem* and *em* are relative units, *px* is not. Before considering *rem*, it's important to understand the relationship between *em*, markup and inheritance.

Below, the example demonstrates how each nested child assumes the parent is 1em(100%). Thus children inherit size by scaling in relation to the parent font size.

EM values inherit from their parent

HAML	Sass	Result	Edit
<code>html { font-size: 1.375em; }</code>			
		100% (22px)	
<code>.font_small { font-size: 0.773em; }</code>			
		77.3% (17px)	
<code>.font_small { font-size: 0.773em; }</code>			
		77.3% (13px)	

PX values do not inherit

HAML	Sass	Result	Edit
<code>html { font-size: 22px; }</code>			
<code>.font_small { font-size: 17px; }</code>			
<code>.font_small { font-size: 17px; }</code>			

MORE RESOURCES — MEDIA QUERIES

CSS-TRICKS

[Blog](#) [Videos](#) [Almanac](#) [Snippets](#) [Forums](#) [Jobs](#) [Lodge](#)



CSS Media Queries & Using Available Space

Published July 6, 2010 by Chris Coyier

We've covered using CSS media queries to assign [different stylesheets depending on browser window size](#). In that example, we changed the layout of the entire page based on the space available. It isn't required that we make such drastic changes with this technique though, so in this tutorial we'll go over a

The [Lodge](#) is a [member login](#) only area with access to video training on how to build websites from scratch using the best modern tools.



LEARNING OBJECTIVES

- Describe responsive design.
- Know the difference between fluid, fixed and responsive layouts
- Apply media queries to achieve a responsive layout.

RESPONSIVE BASICS

EXIT TICKETS