Today's schedule

Schedule:

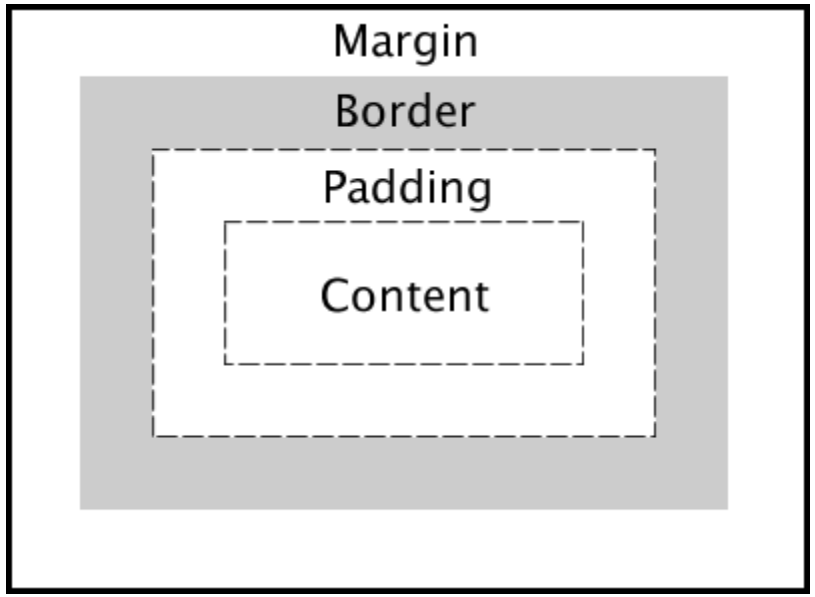
- Box model
- Mobile web

CSS Box Model

The CSS Box Model

Every element is composed of 4 layers:

- the element's content
- the **`border`** around the element's content
- **`padding`** space between the content and border (inside)
- a **`margin`** clears the area around border (outside)



border

HTML

```
<p>
  Welcome to CS193X: Web Programmi
  In this class, you will learn mou
</p>
```

CSS

```
p {
  border: 2px solid black;
}
```

Welcome to CS193X: Web Programming Fundamentals! In this class, you will learn modern full-stack web development techniques.

We've used the shorthand:

`border:` *width style color*`;`

border

Can also specify each border individually:

```
border-top
border-bottom
border-left
border-right
```

And can set each property individually:

```
border-style: dotted;
border-width: 3px;
border-color: purple;
```

(all styles)

border

Can also specify each border individually:

```
border-top
border-bottom
border-left
border-right
```

And can set each property individually:

```
border-style: dotted;
border-width: 3px;
border-color: purple;
```

(all styles)

There are other units besides pixels (px) but we will address them in the next couple lectures.

Rounded border

Can specify the `border-radius` to make rounded corners:

```
border-radius: 10px;
```

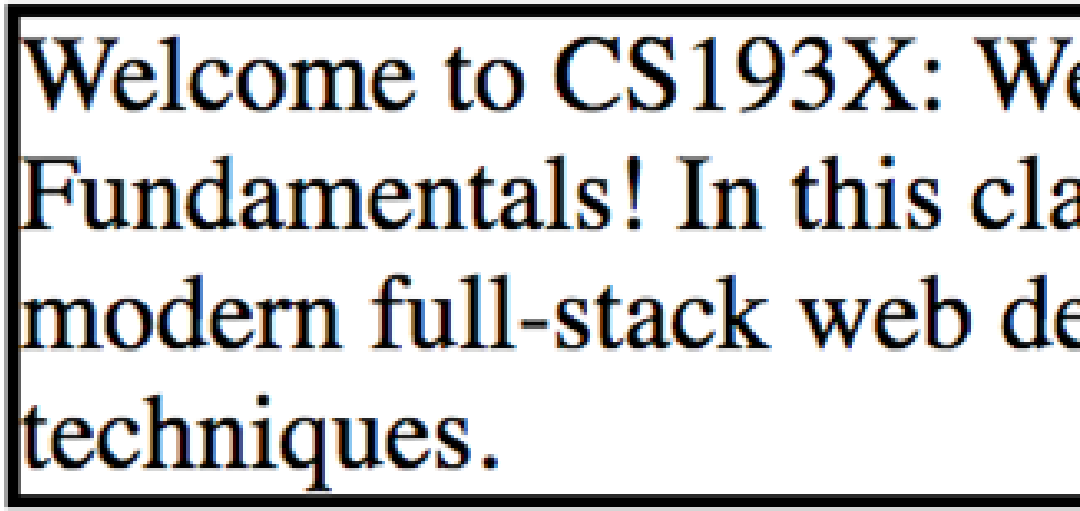
You don't actually need to set a border to use border-radius.

```
p {
  background-color: purple;
  border-radius: 10px;
  color: white;
}
```

Welcome to CS193X: Web Programming Fundamentals! In this class, you will learn modern full-stack web development techniques.

Borders look a little squished

When we add a border to an element, it sits flush against the text:



Q: How do we add space between the border and the content of the element?

padding

```
p {
  border: 2px solid black;
  padding: 10px;
}
```

Welcome to CS193X: Web Programming Fundamentals! In this class, you will learn modern full-stack web development techniques.

padding is the space between the border and the content.

- Can specify `padding-top`, `padding-bottom`, `padding-left`, `padding-right`
- There's also a shorthand:

  `padding: 2px 4px 3px 1px;` <- top|right|bottom|left

  `padding: 10px 2px;` <- top+bottom|left+right

s look a little squished

When we add a border to multiple divs, they sit flush against each other:

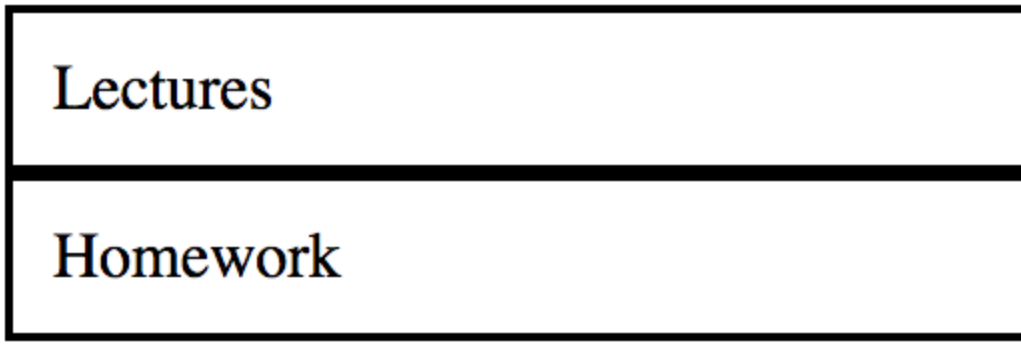
HTML

```
<div>
  Lectures
</div>
<div>
  Homework
</div>
```

CSS

```
div {
  border: 2px solid black;
  padding: 10px;
}
```

Q: How do we add space between multiple elements?



margin

```
div {
  margin: 20px;
  padding: 10px;
  border: 2px solid black;
}
```

Lectures

Homework

`margin` is the space between the border and other elements.

- Can specify `margin-top`, `margin-bottom`, `margin-left`, `margin-right`
- There's also a shorthand:

  `margin: 2px 4px 3px 1px;` <- **top|right|bottom|left**

  `margin: 10px 2px;` <- **top+bottom|left+right**

margin

Actually, why doesn't this:

```
div {
  margin: 20px;
  padding: 10px;
  border: 2px solid black;
}
```

Lectures

Homework

Look more like this?

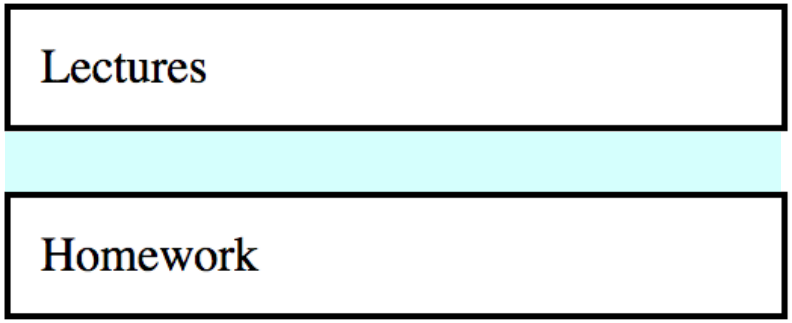
Lectures

Homework

margin

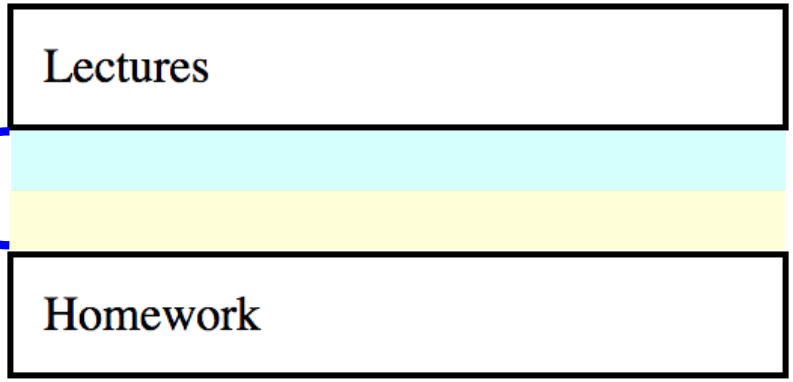
Actually, why doesn't this:

```
div {
  margin: 20px;
  padding: 10px;
  border: 2px solid black;
}
```



...look more like this?

20px margin-bottom +
20px margin top =
40px margin?



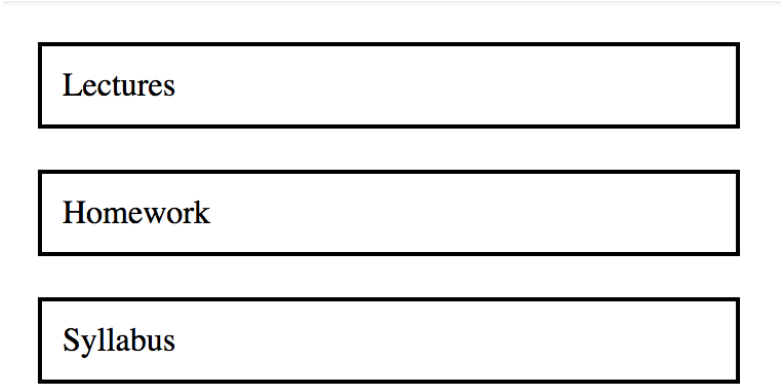
`margin` collapsing

Sometimes the top and bottom margins of block elements are combined ("collapsed") into a single margin.

- This is called **margin collapsing**

Generally if:

- The elements are siblings
- The elements are block-level (***not*** *inline-block*)



then they collapse into **max**(*Bottom Margin*, *Top Margin*).

(There are some exceptions to this, but when in doubt, use the Page Inspector to see what's going on.)

Negative margin

Margins can be negative as well.

- **No negative margin on image:**

HTML

```
<div id="header"></div>
<div id="profile">
  <img src="https://s3-us-west-2.amazonaws.com/
</div>
```

CSS

```
#header {
  background-color: lightblue;
  height: 200px;
}

img {
  margin-left: 50px;
  height: 140px;
  border: 2px solid LIGHTGRAY;
}
```

Negative margin

Margins can be negative as well. (CodePen)

- `img { margin-top: -50px; }`

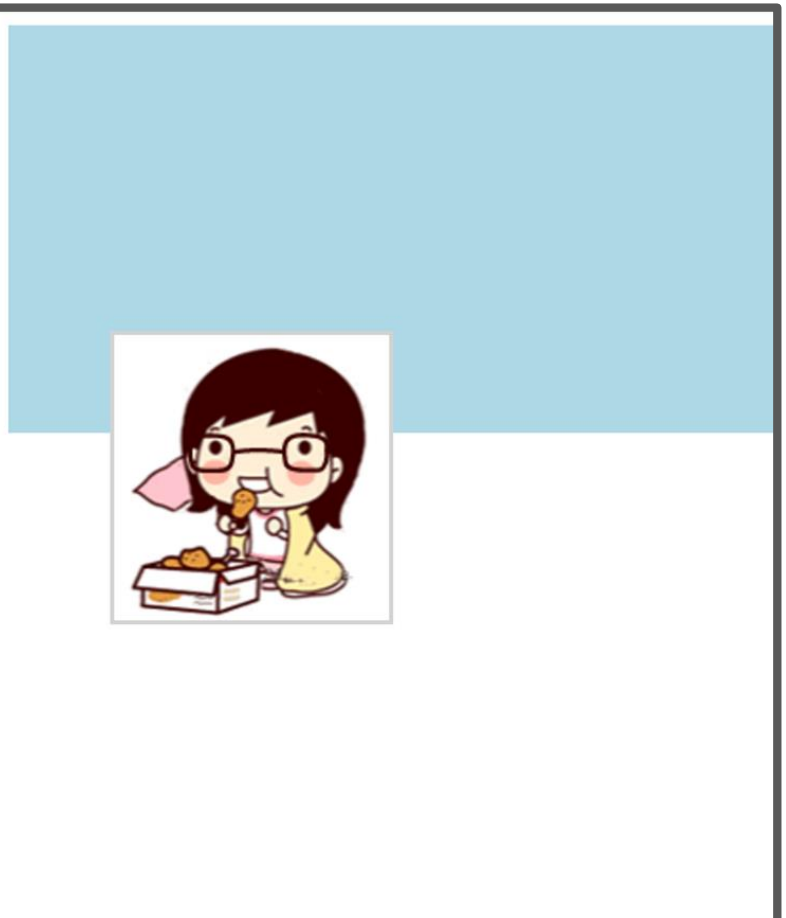
HTML

```
<div id="header"></div>
<div id="profile">
  <img src="https://s3-us-west-2.amazonaws.com/
</div>
```

CSS

```
#header {
  background-color: lightblue;
  height: 200px;
}

img {
  margin-top: -50px;
  margin-left: 50px;
  height: 140px;
  border: 2px solid LIGHTGRAY;
}
```



auto margins

If you set `margin-left` and `margin-right` to `auto`, you can center a block-level element (CodePen):

HTML

```html
<html>
  <head>
    <meta charset="utf-8">
    <title>Auto Margins</title>
  </head>
  <body>
    <div>
      This is a box of text.
    </div>
  </body>
</html>
```

CSS

```css
div {
  margin-left: auto;
  margin-right: auto;

  border: 2px solid black;
  padding: 10px;
  width: 300px;
}
```

This is a box of text.

Box model for inline elements?

Q: Does the box model apply to inline elements as well?

Box model for inline elements?

Q: Does the box model apply to inline elements as well?

A: Yes, but the box is shaped differently.

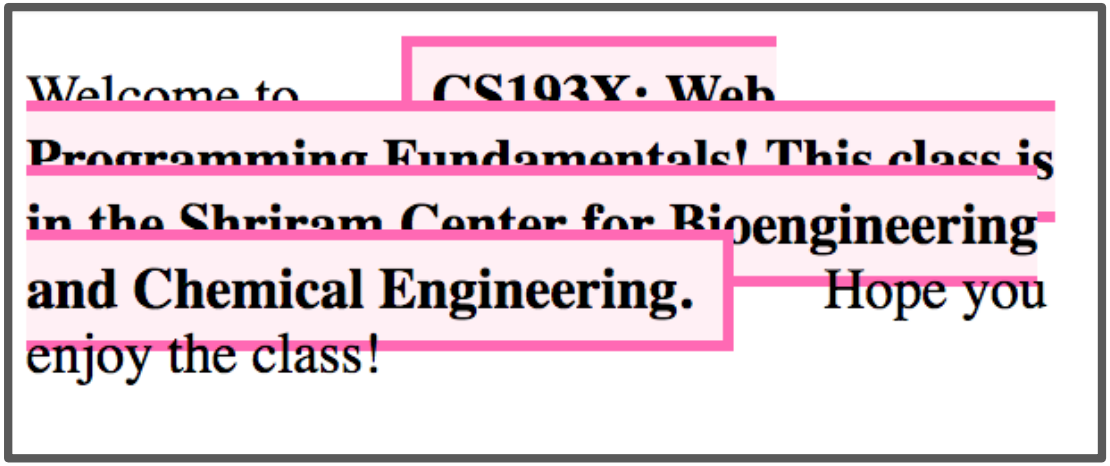
CSS

```
strong {
  border: 3px solid hotpink;
  padding: 5px;
  margin: 25px;

  background-color: lavenderblush;
}
```

HTML

```
<p>
  Welcome to
  <strong>
    CS193X: Web Programming
  </strong>
  Hope you enjoy the class!
</p>
```



Box model for inline elements?

Q: Does the box model apply to inline elements as well?

A: Yes, but the box is shaped differently.

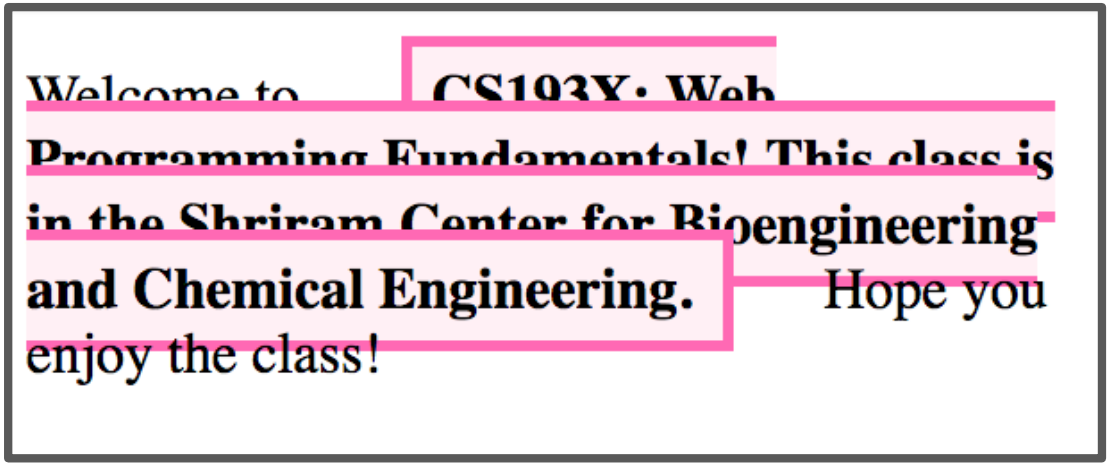
CSS

```
strong {
  border: 3px solid hotpink;
  padding: 5px;
  margin: 25px;

  background-color: lavenderblush;
}
```

HTML

```
<p>
  Welcome to
  <strong>
    CS193X: Web Programming
  </strong>
  Hope you enjoy the class!
</p>
```



Let's change the line height to view this more clearly...

Inline element box model

CSS

```
p {
  width: 300px;
  line-height: 50px;
}

strong {
  border: 3px solid hotpink;
  padding: 5px;
  margin: 25px;
  background-color: lavenderblush;
}
```

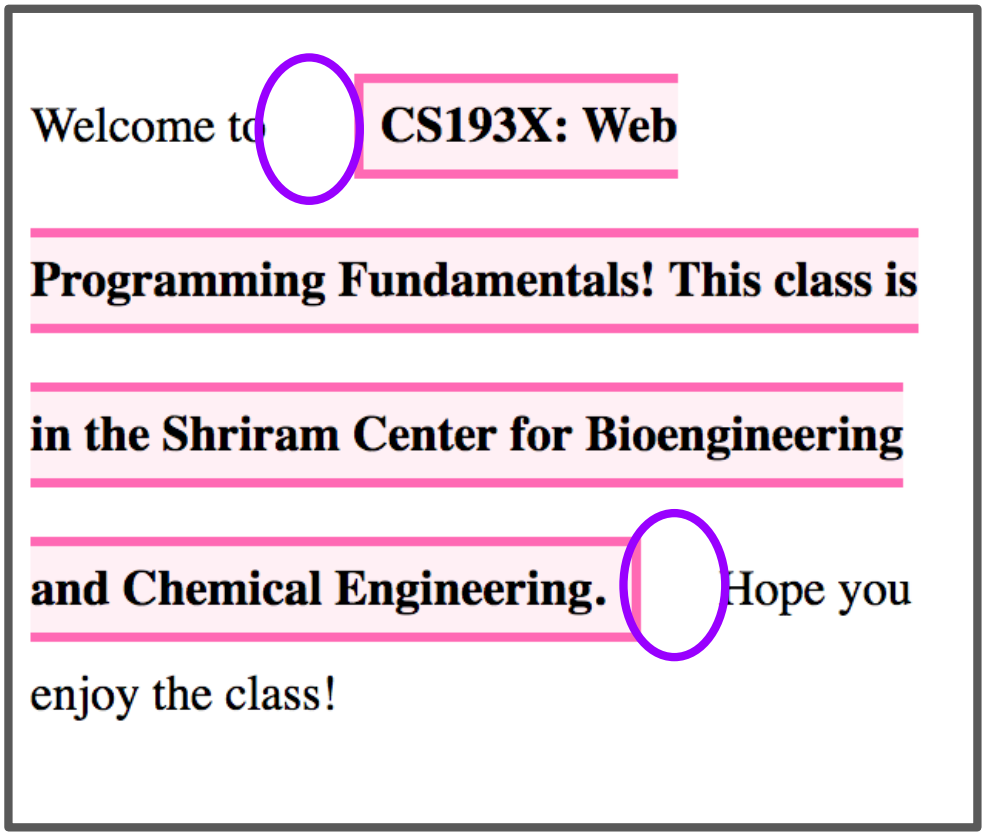
Welcome to **CS193X: Web Programming Fundamentals! This class is in the Shriram Center for Bioengineering and Chemical Engineering.** Hope you enjoy the class!

(Codepen)

Inline element box model

CSS

```
strong {
  border: 3px solid hotpink;
  padding: 5px;
  margin: 25px;
  line-height: 50px;
  background-color: lavenderblush;
}
```

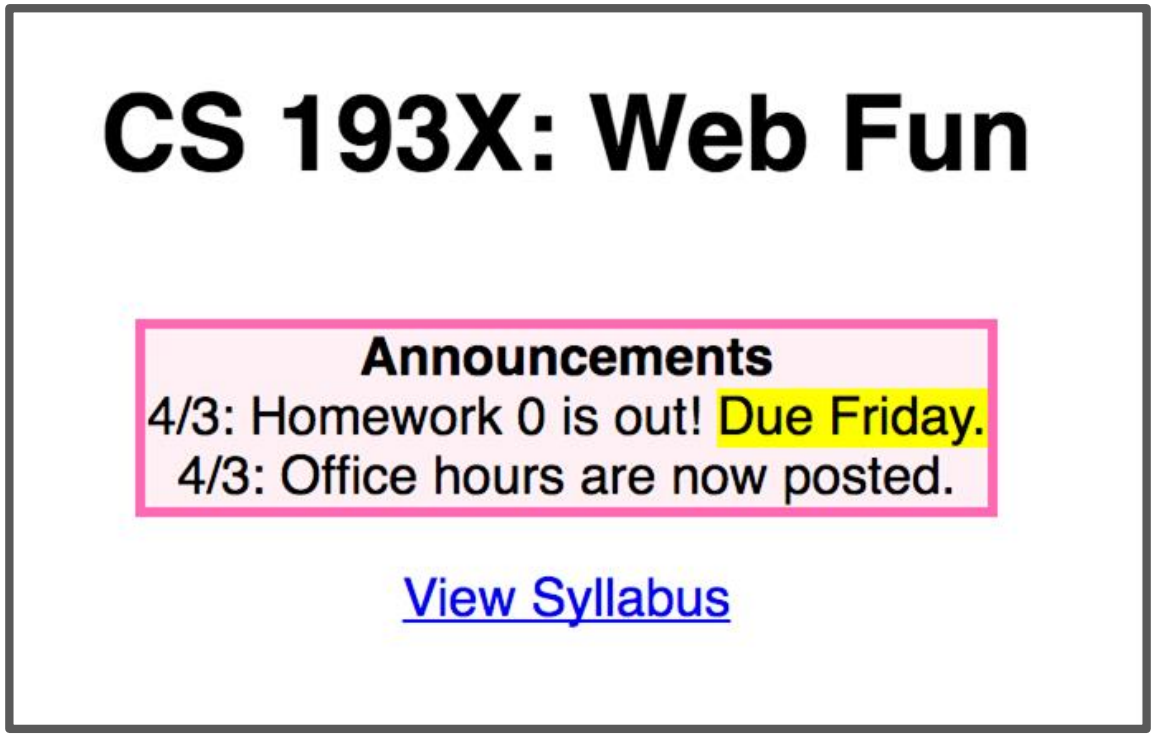


- **margin** is to the left and right of the inline element
  - `margin-top` and `margin-bottom` are **ignored**
- use **`line-height`** to manage space between lines

(Codepen)

The CSS Box Model

Let's revisit our Course web page example:



Q: What does this look like in the browser?

```
div {
  display: inline-block;
  background-color: yellow;
}
```

```
<body>
 <div>
  <p>Make the background color yellow!</p>
  <p>Surrounding these paragraphs</p>
 </div>
</body>
```

Make the background color yellow!

Surrounding these paragraphs

Q: Why is there a white space around the box?

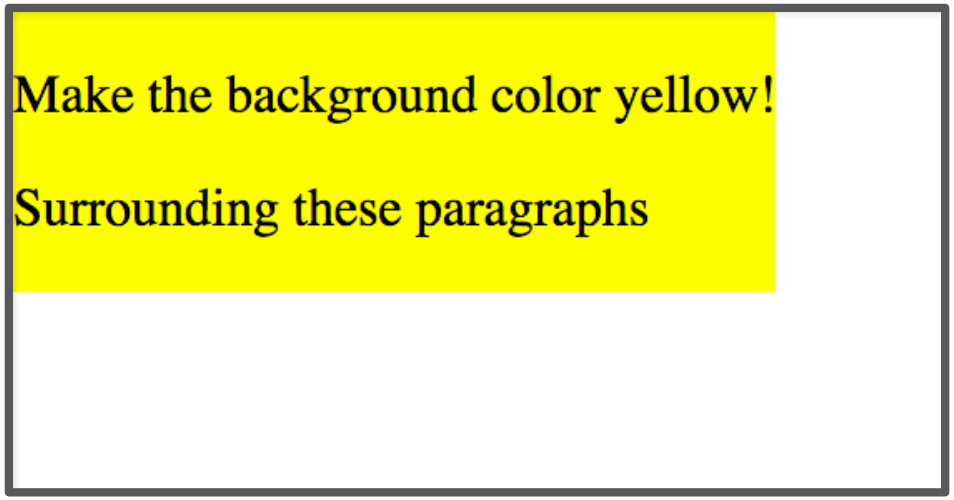
We can use the browser's Page Inspector to help us figure it out!

body has a default margin

Set `body { margin: 0; }` to make your elements lay flush to the page.

```
body {
  margin: 0;
}

div {
  display: inline-block;
  background-color: yellow;
}
```



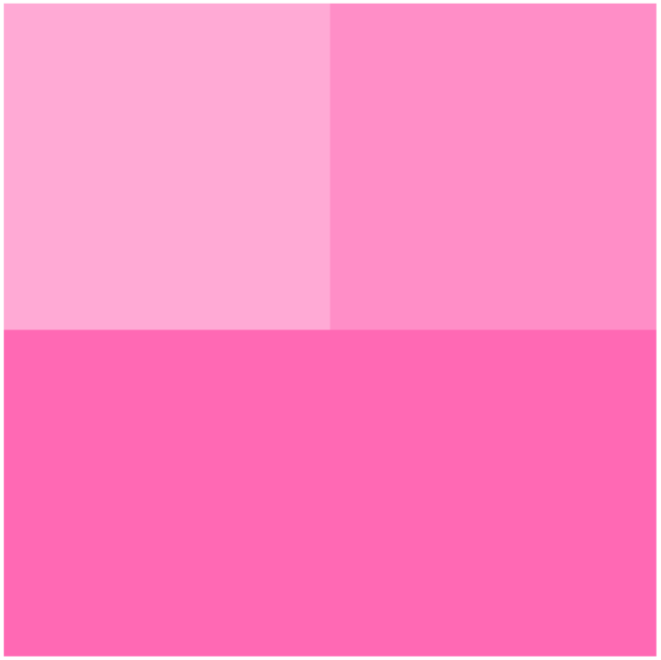
Aside: sizing

Q: What happens if we add a border to `#upper-half`?

```
<div id="box">
  <div id="upper-half">
    <div id="upper-quarter"></div>
  </div>
</div>
```

```
#upper-half {
  height: 50%;
  width: 100%;
}
```

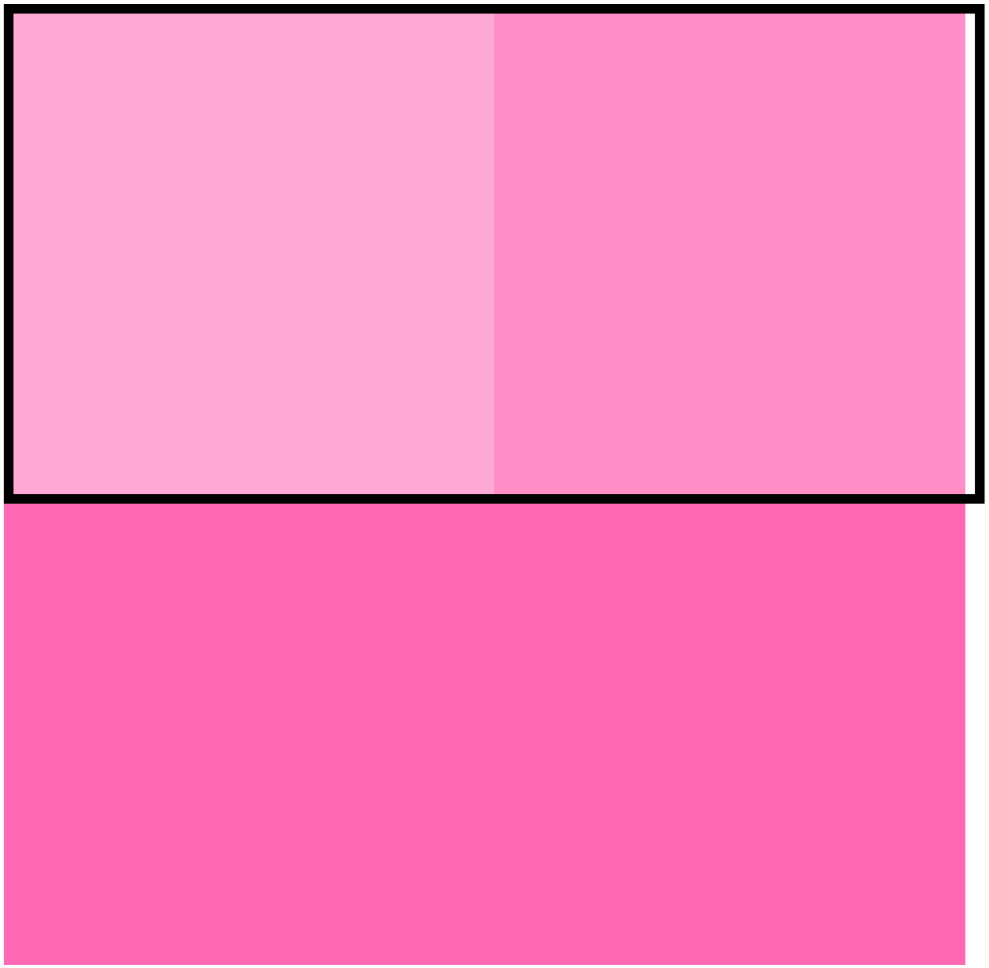
(rest of the css)



??
?

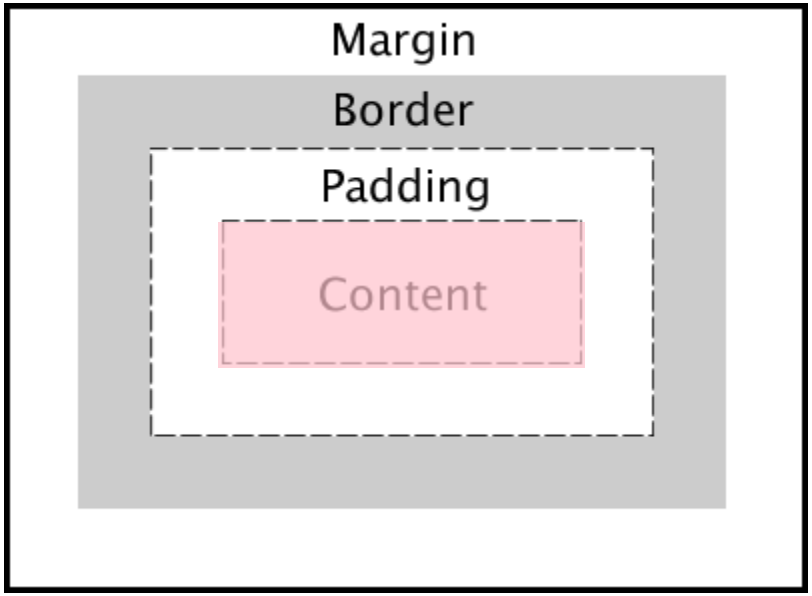
```
#upper-half {
  height: 50%;
  width: 100%;
  border: 5px solid black;
}
```

(rest of the CSS)



CSS box model width and height

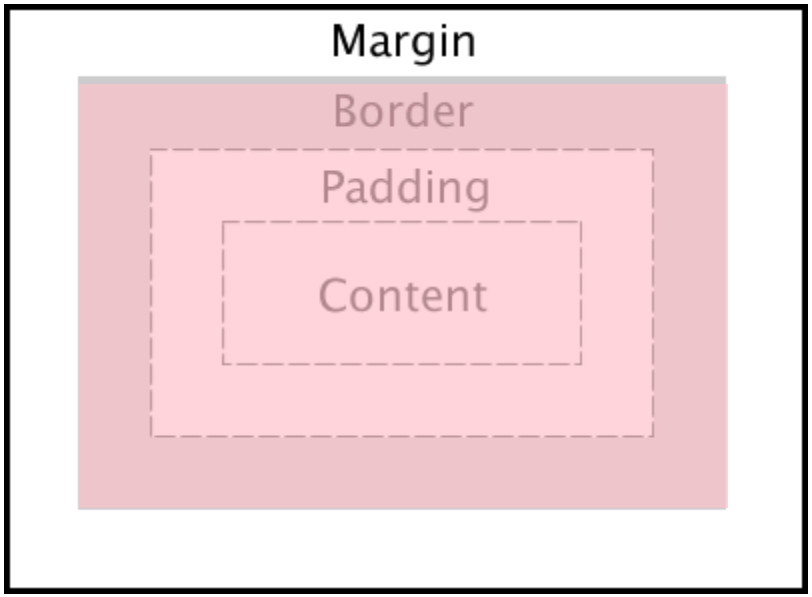
The box model defines CSS `width` and `height` properties to refer to the element's **content** width and height:



box-sizing

If you want to have `width` and `height` refer to the element's **border** width and height, use `box-sizing`:

- `box-sizing: border-box;`

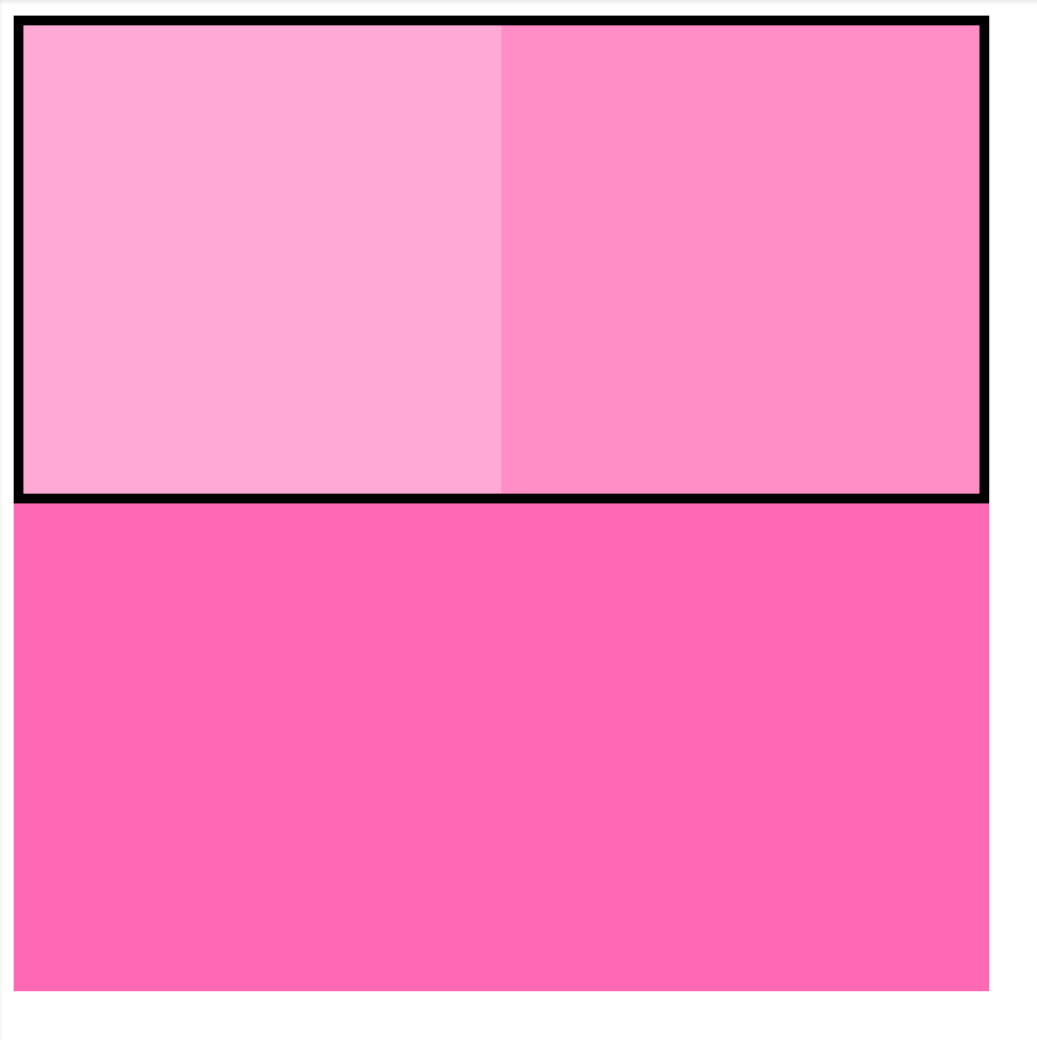


Note: Using `border-box` will include padding in the `width` and `height` as well.
Note: You **cannot** select padding-box or margin-box.

Fixed example

```
#upper-half {
  height: 50%;
  width: 100%;
  border: 5px solid black;
  box-sizing: border-box;
}
```

(rest of the CSS)



Recap so far...

We've talked about:

- **block vs inline** and the "natural" layout of the page, depending on the element type
- **`classes and ids`** and how to specify specific elements and groups of elements
- **`div and span`** and how to create generic elements
- **The CSS box model** and how every element is shaped like a box, with content -> padding -> border -> margin

Let's try making a "real" looking page!

Mobile web

Say you have the following website:

CS 193X: Spring 2017 Victoria Perso...

Secure https://yayinternet.github.io/lecture06/mobile.html

Stanford CS Department

CS 193X: Spring 2017

Welcome to CS193X: Web Programming Fundamentals! This is an opinionated introduction to web programming. In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereqs: CS106B; Unix/command-line proficiency or CS1U
- Lectures: Mon-Wed-Fri, 1:30-2:20 in Shriram Center 104
- Final exam: No final exam.
- Final project: Yes, there will be one. Details TBA.

CS193X is limited enrollment. This is the first offering of CS193X, so enrollment is limited and by application only. It will be offered again sometime in the 2017-2018 school year.

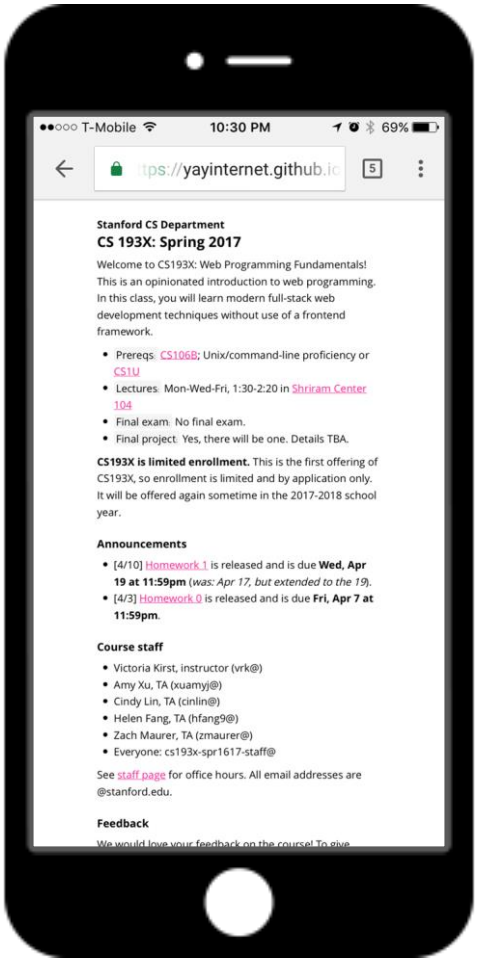
Announcements

- [4/10] Homework 1 is released and is due **Wed, Apr 19 at 11:59pm** (*was: Apr 17, but extended to the 19*).
- [4/3] Homework 0 is released and is due **Fri, Apr 7 at 11:59pm**.

Course staff

- Victoria Kirst, instructor (vrk@)
- Amy Xu, TA (xuamyj@)
- Cindy Lin, TA (cinlin@)
- Helen Fang, TA (hfang9@)
- Zach Maurer, TA (zmaurer@)
- Everyone: cs193x-spr1617-staff@

Q: What does it look like on a phone?



Not terrible… but pretty small and hard to read.

Responsive web design

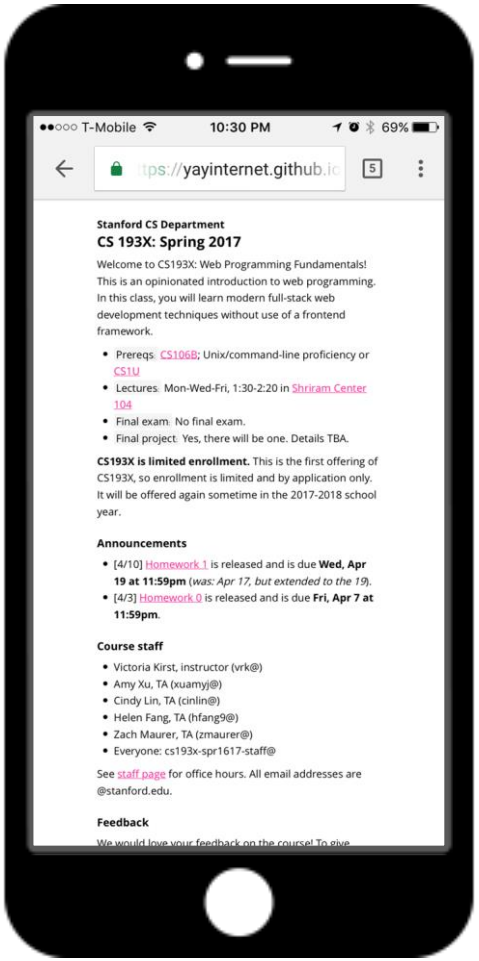
We want to write our CSS in a way that can look nice in a wide range of screen sizes:

- 27" desktop monitor
- Macbook Air
- Samsung Galaxy S7
- iPhone 7
- iPad

Q: How do we do this?

Do we need to write totally different CSS for every screen size?!

Mobile sizing



Unless directed otherwise via HTML or CSS cues, mobile browsers render web pages at a **desktop screen width** (~1000px), then "zooms out" until the entire page fits on screen.

(That's why you sometimes get web pages with teeny-tiny font on your phone: these webpages have not added support for mobile.)

(Read more on how this works)

Meta viewport tag

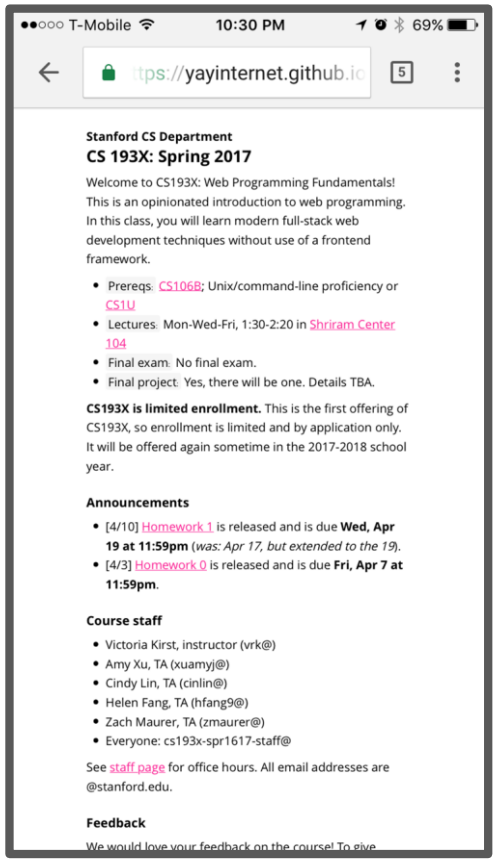
To prevent phone browsers from rendering the page at desktop width and zooming out, use the **meta viewport tag**:

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

This belongs in the <head> section of your HTML.

(Same section as the `<title>`, `<link>`, and other metadata elements.)

Meta viewport tag



Without the meta viewport tag

With the meta viewport tag

Meta viewport tag

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

- **`name=viewport`**: "Browser, I am going to tell you how I want the viewport to look."
- **`width=device-width`**: "The viewport's width should always start at the device's width." (i.e., don't do that thing on mobile where you render the page at the desktop's width)
- **`initial-scale=1`**: "Start at zoom level of 100%."

Meta viewport tag

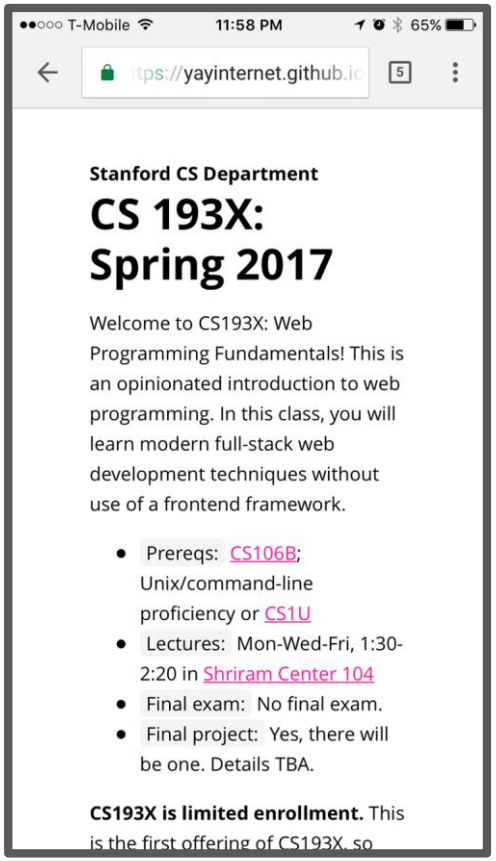
```
<meta name="viewport"
content="width=device-width, initial-scale=1">
```

(You should pretty much always include this tag in your HTML.)

Making adjustments

The meta viewport tag gets us almost all the way there, but we want to make a few adjustments.

For example, the margin seems a tad too big on mobile. Can we set a different margin property for mobile?



CSS media queries

You can define a **CSS media query** in order to change style rules based on the characteristics of the device:

```
@media (max-width: 500px) {
  article {
    margin: 0 2px;
  }
}
```

You can create much more complex media queries as well.

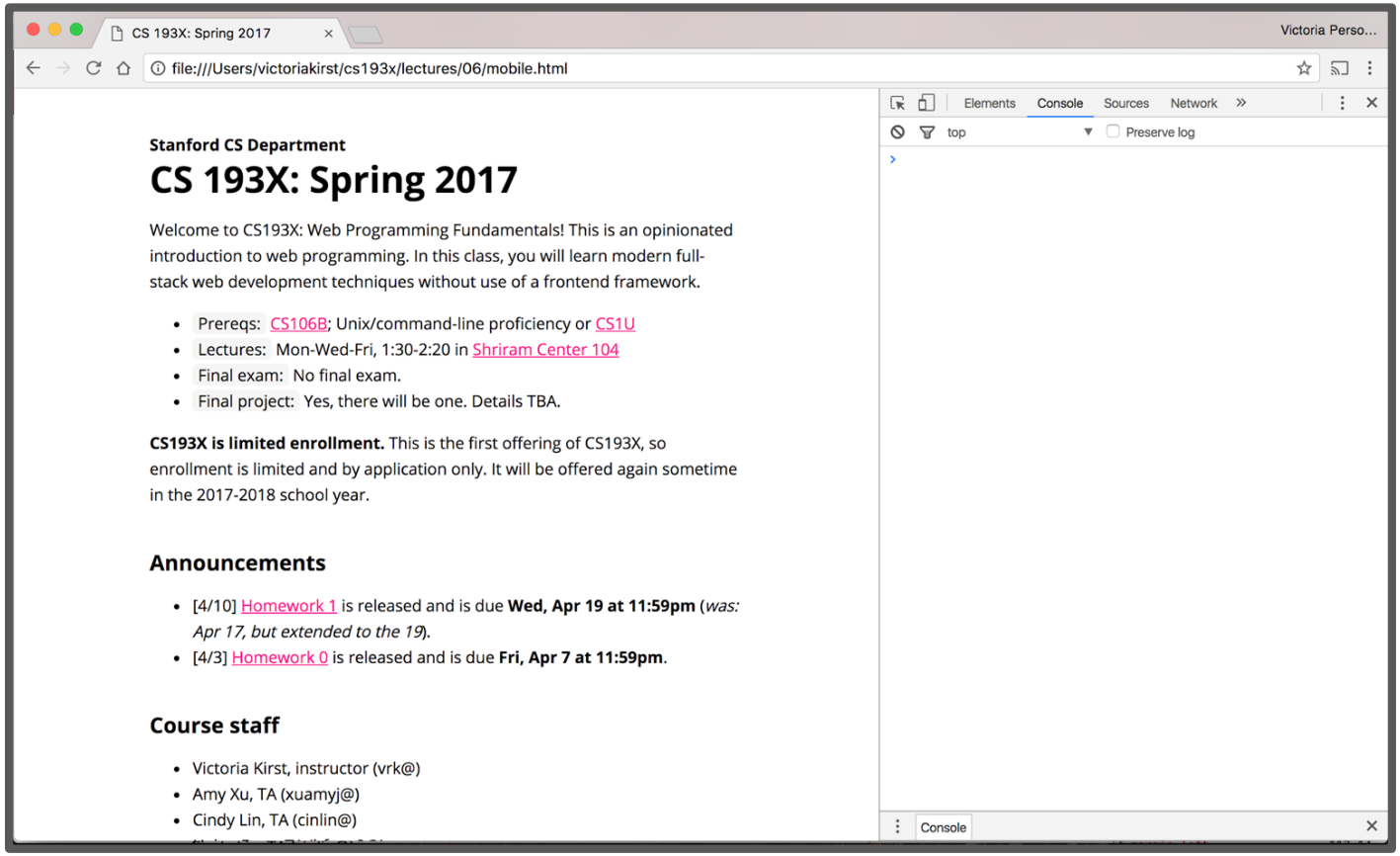
Development strategies

Practical question: **How do you test mobile layouts?**

- Do you upload your HTML+CSS somewhere online and navigate to that URL on your phone?
- Is there a way to connect your phone to your local device?
- Do you run it in an Android/iOS emulator?
- Other?!

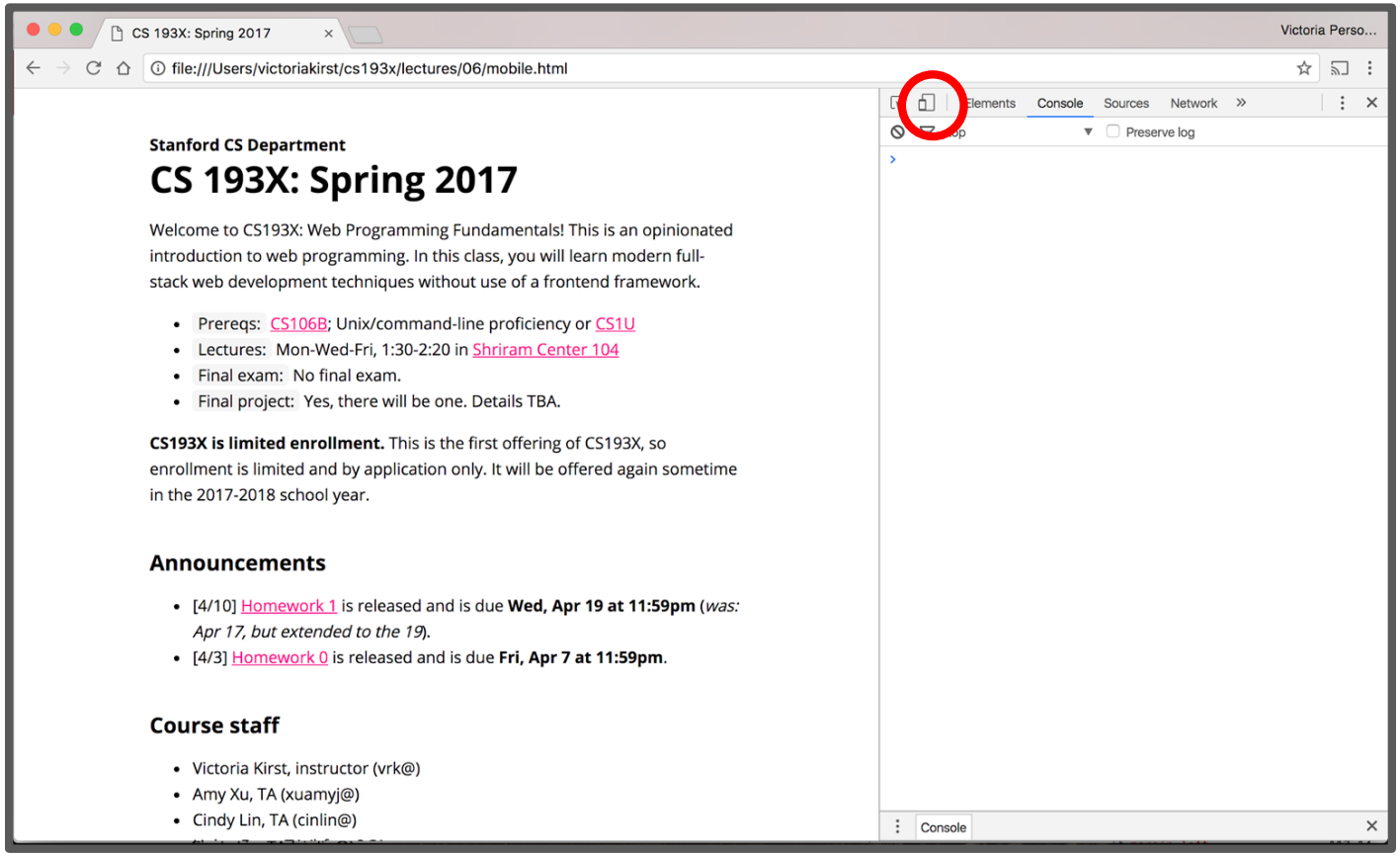
Chrome device mode

You can simulate a web page in a mobile layout via [Chrome device mode]:



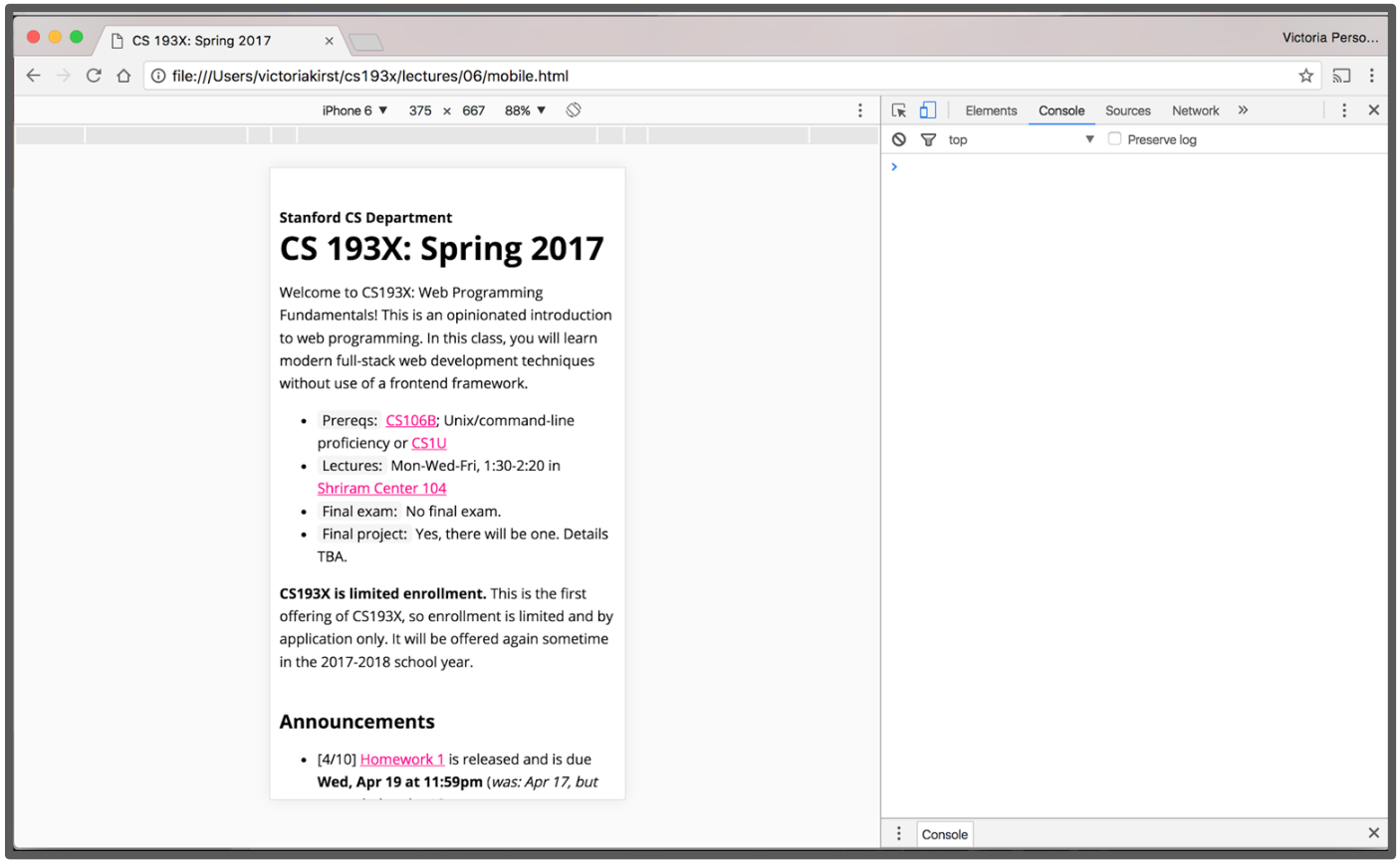
Chrome device mode

You can simulate a web page in a mobile layout via [Chrome device mode]:



Chrome device mode

You can simulate a web page in a mobile layout via [Chrome device mode](#):



Chrome device mode

Advantages of Chrome device mode:

- Super convenient
- Mostly accurate

Disadvantages of Chrome device mode:

- Not always accurate - iPhone particularly an issue
- A little buggy
- Doesn't simulate performance issues

You should always test on real devices, too.

Relative font sizes: percent, `em`, `rem`

Relative units

Whenever possible, it's best to use **relative units** (like percentage) instead of absolute units (like px).

Advantages:

- More likely to work on different screen sizes
- Easier to reason about; fewer magic numbers
  `10% / 80% / 10%` vs `122px / 926px / 122px`

Q: Should we be using relative units on `font-size`?

Relative font sizes: percent

You can define font sizes in terms of percentage:

```html
<body>
  <h1>This is 60px</h1>
  <p>This is 15px</p>
</body>
```

```css
body {
  font-size: 30px;
}

h1 {
  font-size: 200%;
}

p {
  font-size: 50%;
}
```

This is 60px

This is 15px

(CodePen)

Relative font sizes: percent

Percent on font-size behaves exactly like percentage on width and height, in that it's relative to the parent:

```
<div>
  This is 60px
  <p>This is 45px</p>
</div>
```

```
body {
  font-size: 30px;
}

div {
  font-size: 200%;
}

p {
  font-size: 75%;
}
```

This is 60px

This is 45px

(CodePen)

Relative font sizes: percent

Percent on font-size behaves exactly like percentage on width and height, in that it's relative to the parent:

```
<div>
  This is 60px
  <p>This is 45px</p>
</div>
```

```
body {
  font-size: 30px;
}

div {
  font-size: 200%;
}

p {
  font-size: 75%;
}
```

This is 60px

This is 45px

p is 75% of its parent, which is 200% of 30px.

p's size: .75*2*30 = 45px

(CodePen)

Relative font sizes: `em`

But instead of percentages, relative font sizes are usually defined in terms of `em`:

- `em` represents the calculated `font-size` of the element
  - `1em` = the inherited font size
  - `2em` = 2 times the inherited font size

In other words,

`font-size: 1em;` is the same as `font-size: 100%;`

Relative font sizes: em

```html
<body>
  <h1>This is 60px</h1>
  <p>This is 15px</p>
</body>
```

```css
body {
  font-size: 30px;
}

div {
  font-size: 2em;
}

p {
  font-size: .5em;
}
```

This is 60px

This is 15px

(CodePen)

Relative font sizes: em

```html
<div>
  This is 60px
  <p>This is 45px</p>
</div>
```

```css
body {
  font-size: 30px;
}

div {
  font-size: 2em;
}

p {
  font-size: .75em;
}
```

This is 60px

This is 45px

(CodePen)

Relative font sizes: em

```html
<div>
  This is 60px
  <p>This is 45px</p>
</div>
```

```css
body {
  font-size: 30px;
}

div {
  font-size: 2em;
}

p {
  font-size: .75em;
}
```

This is 60px

This is 45px

p's inherited font size is 2em, which is 60px. So 0.75em is 0.75*60 = 45px.

(CodePen)

```html
<body>
  This is
  <h1>
    <strong>120px</strong>
  </h1>
</body>
```

```css
body {
  font-size: 30px;
}

strong {
  font-size: 2em;
}
```

This is

120px

Wait, why is `<strong>` 120px and not 60px?

```html
<body>
  This is
  <h1>
    <strong>120px</strong>
  </h1>
</body>
```

```css
body {
  font-size: 30px;
}

strong {
  font-size: 2em;
}
```

This is

120px

(CodePen)

```
h1 {                                    user agent stylesheet
   display: block;
   font-size: 2em;
   -webkit-margin-before: 0.67em;
   -webkit-margin-after: 0.67em;
   -webkit-margin-start: 0px;
   -webkit-margin-end: 0px;
   font-weight: bold;
}
```

In the Chrome Inspector, we see the default browser `font-size` for h1 is 2em. So it's 30*2*2 = 120px.

Relative font sizes: **`rem`**

If you **do not** want your relative font sizes to compound through inheritance, use `rem`:

- `rem` represents the `font-size` of the root element
  - `1rem` = the root (`html` tag) font size
  - `2rem` = 2 times root font size

Relative font sizes: rem

```html
<body>
  <div>
    This is 60px
    <p>This is 22.5px</p>
  </div>
</body>
```

```css
html {
  font-size: 30px;
}

div {
  font-size: 2rem;
}

p {
  font-size: .75rem;
}
```

This is 60px

This is 22.5px

(CodePen)

Relative font sizes: rem

```html
<body>
  <div>
    This is 60px
    <p>This is 22.5px</p>
  </div>
</body>
```

```css
html {
  font-size: 30px;
}

div {
  font-size: 2rem;
}

p {
  font-size: .75rem;
}
```

This is 60px

This is 22.5px

`font-size` is set on the `html` element, not body (or any other tag)

(CodePen)

Relative font sizes: rem

```html
<body>
  <div>
    This is 60px
    <p>This is 22.5px</p>
  </div>
</body>
```

This is 60px

This is 22.5px

```css
html {
  font-size: 30px;
}

div {
  font-size: 2rem;
}

p {
  font-size: .75rem;
}
```

`.75em` is calculated from the root, which is 30px, so 30*.75 = 22.5px.

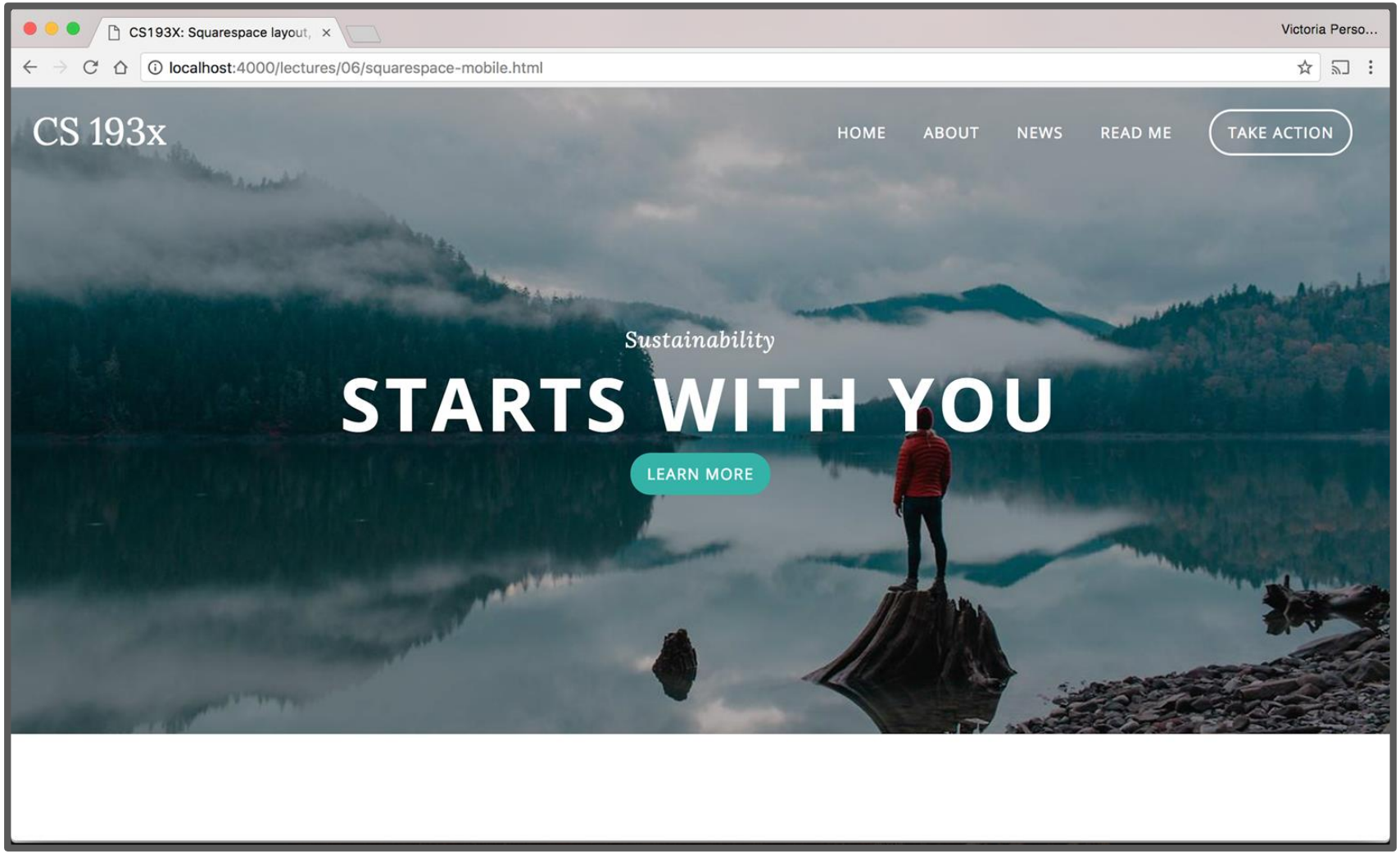
(CodePen)

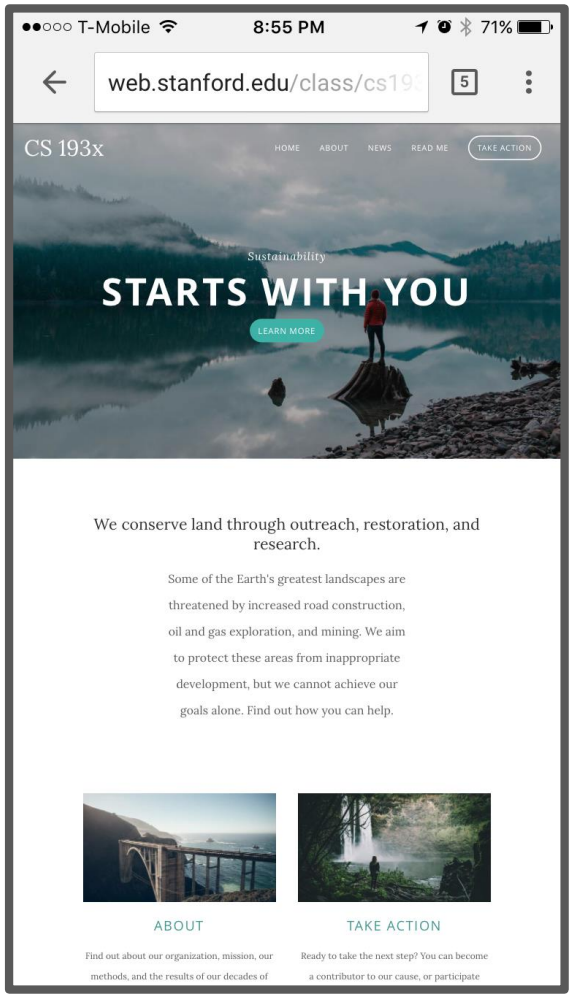
Relative font conclusions

Use relative fonts for the same purpose as using relative heights and widths:

- Prefer `em` and `rem` over percentages
  - Not for any deep reason, but `em` is meant for font so it's semantically cleaner
- Use `rem` to avoid compounding sizes
- In addition to `font-size`, consider `em`/`rem` for:
  - `line-height`
  - `margin-top`
  - `margin-bottom`

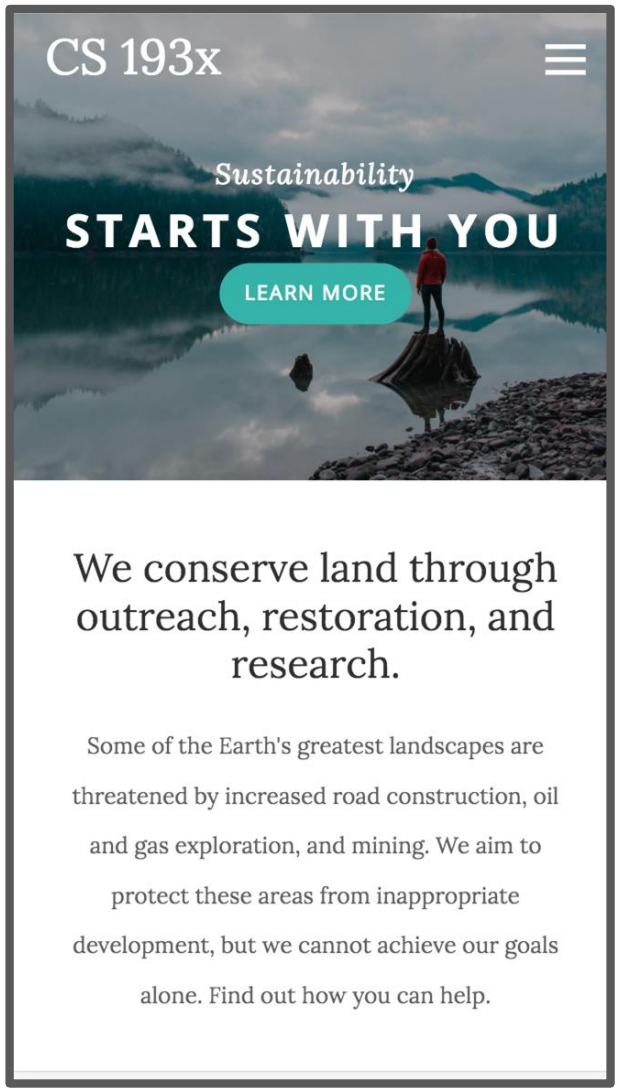
What does our Squarespace layout look like in a phone with the meta viewport tag?

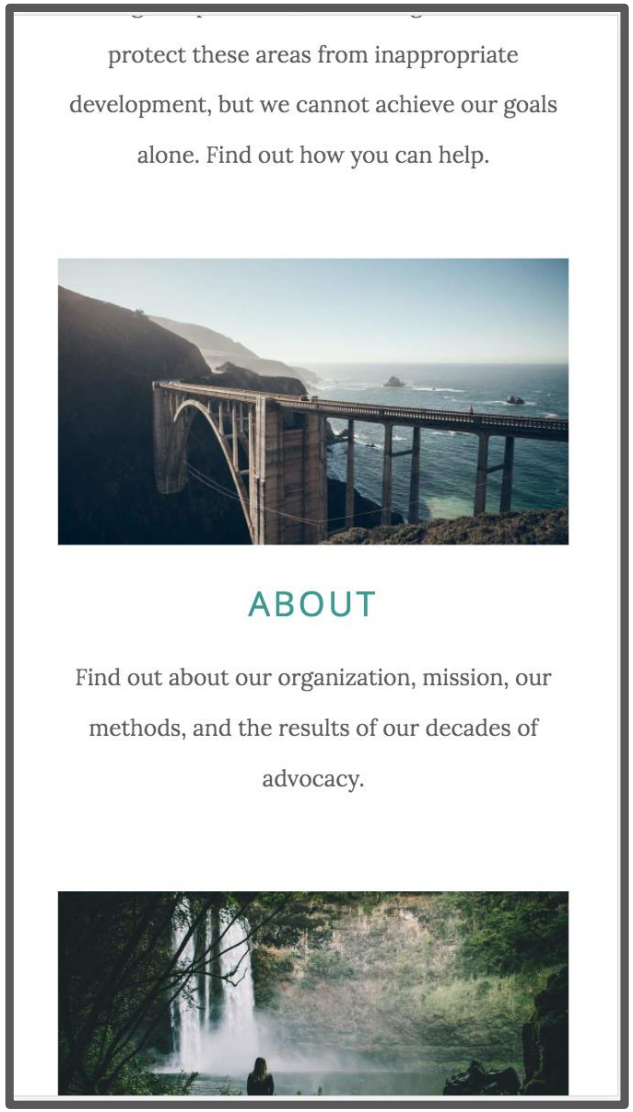




Without the meta viewport tag

With the meta viewport tag





Completed mobile layout

Mobile summary

- Always add the **meta viewport tag**
- Use **`@media` queries** to add styles for devices with certain characteristics, such as screen width
- Use the **Chrome Device Mode** to simulate mobile rendering on desktop
- For `height` and `width`, prefer percentages
- For fonts, prefer `em` and `rem`
- Try to **minimize dependent rules** (Changing the width of one container force you to change 15 other properties to look right)

More on [responsive web design]

CSS wrap-up

Even though we're "done" with CSS, we will be using CSS all quarter in homework and examples.

More?

- *Read more: flexbox*
  - More practice with flexbox / game
- CSS animations
- Possibly `grid`