

CS193X: Web Programming Fundamentals

Spring 2017

Victoria Kirst
(vrk@stanford.edu)

Today's schedule

Announcements:

- Office Hours [now posted](#)

Schedule:

- HTML and CSS
- Inline vs block
- Classes and Ids

HW0 Reminders

[HW0](#) still due this Friday!

A few tips:

- Please don't make your repository public
 - If you do, I will just make it private again
- Don't forget to submit your homework via the [Google Form](#) linked at the bottom of the HW0 spec
- You can update your HW0 GitHub repository/published page without submitting the Google Form again 😊 (but multiple submissions are OK)

Waitlist??

- If you have an access code and have not enrolled:
Please do so ASAP
- If you do not have an access code yet:
Please email me!

Suggestion: Bring your laptop!

- Bring your laptop to lecture so you can follow along with the lecture slides and check out the live examples
- I will be using CodePen in lecture, which lets you livestream the code I write, which might be hard to see on the projector screen



(But, y'know, don't look ahead for the answers to lecture questions and then pretend like you knew them all along.)

HTML and CSS

Quick review

Recall: HTML

HTML (Hypertext Markup Language)

- Describes the **content** and **structure** of a web page; not a programming language.
- Made up of building blocks called **elements**.

<p>

HTML is awesome!!!

</p>

Some HTML elements

Top-level heading: **h1**, **h2**, ... **h6**

```
<h1>Moby Dick</h1>  
<h2>Or, the Whale</h2>
```

Moby Dick

Or, the Whale

Paragraph: **p**

```
<p>Call me Ishmael.</p>
```

Call me Ishmael.

Line break: **br**

```
since feeling is first<br/>  
who pays any attention<br/>  
to the syntax of things
```

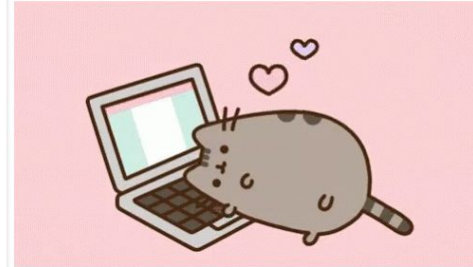
since feeling is first
who pays any attention
to the syntax of things

Some HTML elements

Image: **img**

```

```



Link: **a** (note: not **link**)

```
<a href="google.com">click here!</a>
```

[click here!](https://www.google.com)

Strong (bold): **strong** (note: don't use **b**)

```
<strong>Be BOLD</strong>
```

Be BOLD

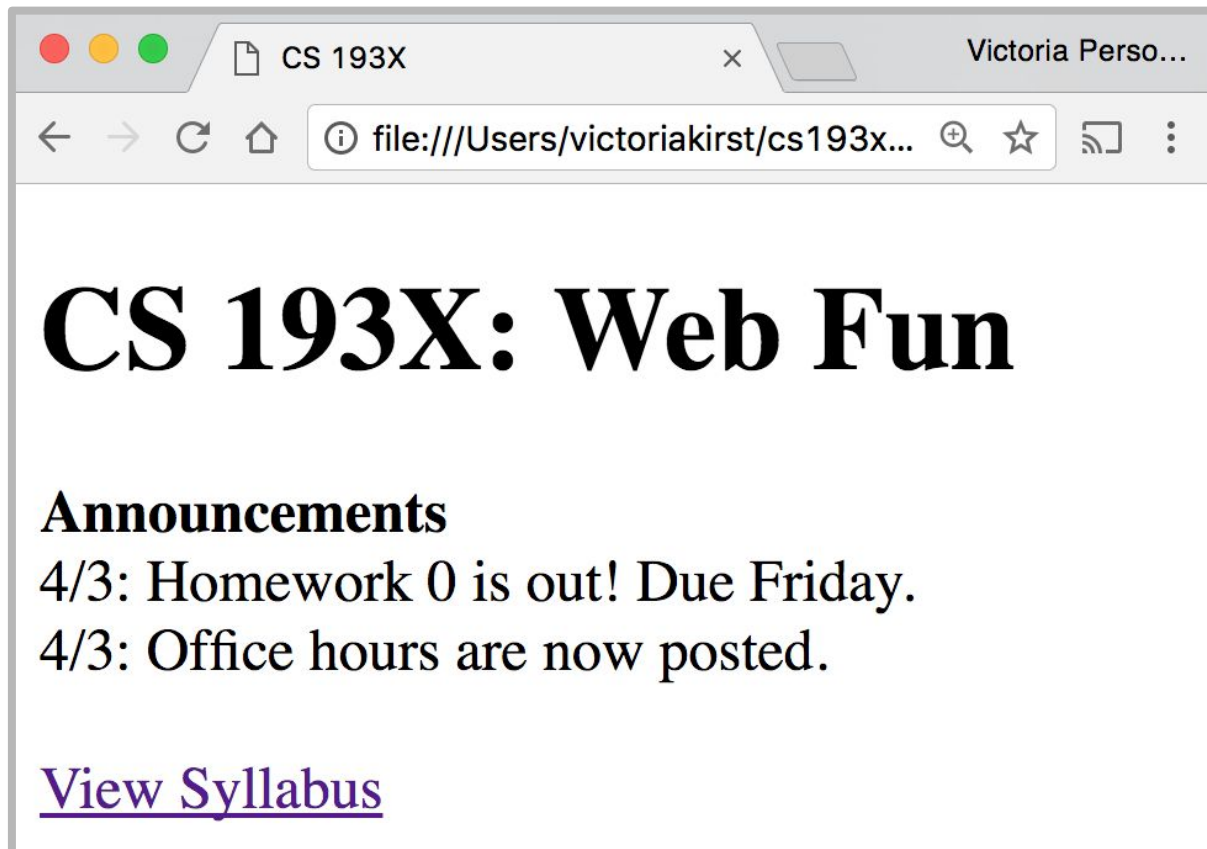
Emphasis (italic): **em** (note: don't use **i**)

```
He's my <em>brother</em> and all
```

He's my *brother* and all

Recall: Course web page

We wrote [some HTML](#) to make the following page:



That was weird

- We saw that HTML whitespace collapses into one space...

```
<h1>CS 193X: Web Fun</h1>  
<strong>Announcements</strong><br />  
4/3: Homework 0 is out!<br />
```

- Except weirdly the `<h1>` heading was on a line of its own, and `` was not.

Hmmm... strange...

Oh well, it works! Let's move on!!!

CSS

Recall: CSS

CSS: Cascading Style Sheets

- Describes the **appearance** and **layout** of a web page
- Composed of CSS **rules**, which define sets of styles

```
selector {  
    property: value;  
}
```

Some CSS properties

Font face: **font-family**

```
h1 {  
  font-family: Helvetica;  
}
```

Moby Dick

Font color: **color**

```
h1 {  
  color: green;  
}
```

Moby Dick

Note that `color` always refers to **font** color, and there's no way to make it mean anything other than font color.

Background color: **background-color**

```
body {  
  background-color: pink;  
}
```

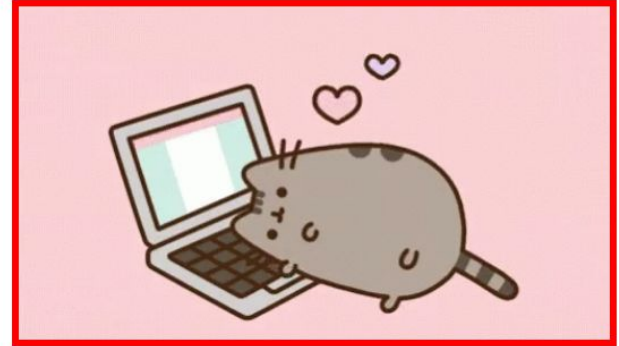
Moby Dick

Assign a `background-color` to `body` to make the page a different color.

Some CSS properties

Border: **border** ([border shorthand syntax](#))

```
img {  
  border: 3px solid red;  
}
```



Text alignment: **text-align** (note: don't use <center>)

```
p {  
  text-align: center;  
}
```

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

CSS colors

140 predefined names ([list](#))

```
color: black;
```

Hex values

```
color: #00ff00;
```

```
color: #0f0;
```

```
color: #00ff0080;
```

rgb() and rgba()

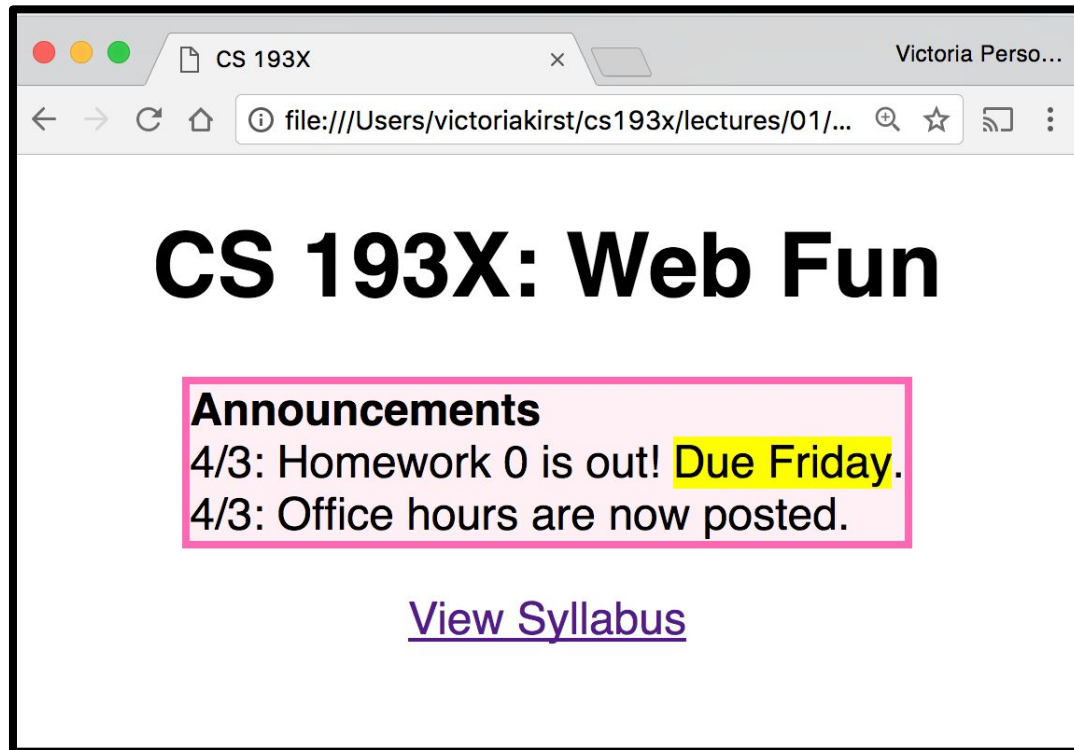
```
color: rgb(34, 12, 64);
```

```
color: rgba(0, 0, 0, 0.5);
```

- The "a" in rgba stands for alpha channel and is a transparency value
- Prefer more descriptive:
 1. Predefined name
 2. rgb / rgba
 3. Hex

Exercise: Course web page

Let's write some CSS to style our page:



✨ [CodePen link: Follow along!](#) ✨

Exercise: Course web page

Let's write some CSS to style our page:

Font face: Helvetica

Border: hotpink 3px

Background color:

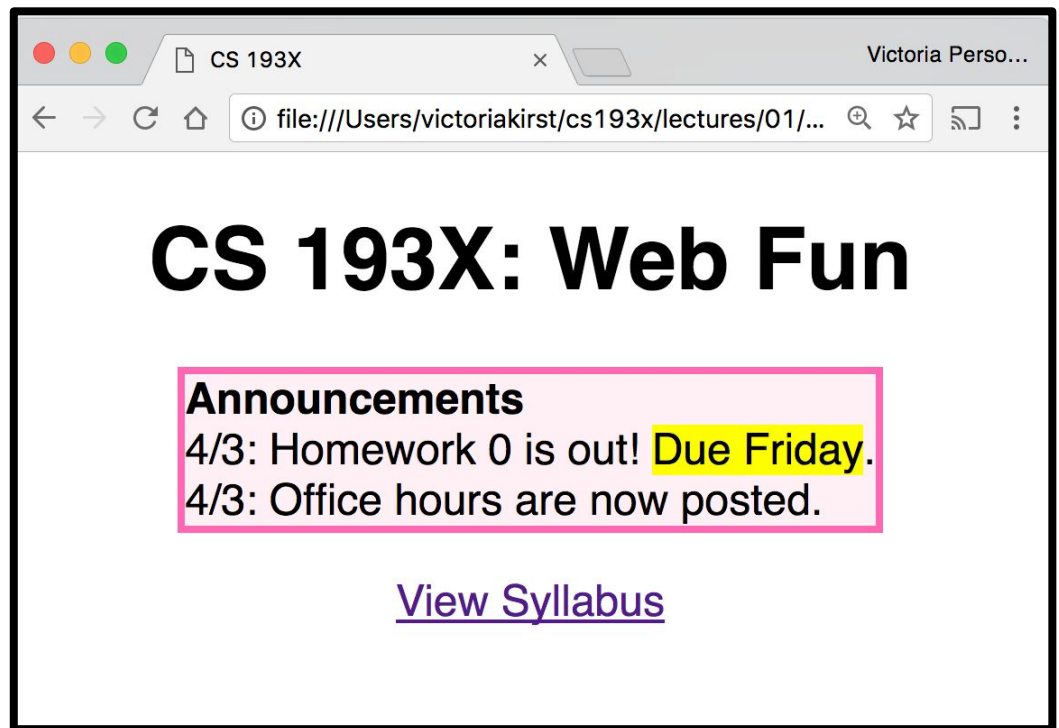
lavenderblush

Highlight: yellow

- Box is **centered**

- Header and link are **centered**

- Box contents are **left-aligned**



✨ [CodePen link: Follow along!](#) ✨

Solution?!

```
body {  
  font-family: Helvetica;  
}  
h1 {  
  text-align: center;  
}  
a {  
  text-align: center;  
}  
p {  
  border: 3px solid hotpink;  
  background-color: lavenderblush;  
}
```

Produces:

CS 193X: Web Fun

Announcements

4/3: Homework 0 is out! Due Friday.

4/3: Office hours are now posted.

[View Syllabus](#)

CSS exercise debrief

We used some **key techniques**:

- Add invisible containers in HTML to select groups of elements in CSS.
- Apply styles to parent / ancestor element to style parent and all its children. (Will talk more about this later.)

CSS exercise debrief

But we encountered **more weirdness...**

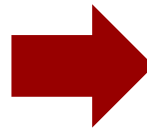
- `text-align: center;` didn't work on the `<a>` tag
- The box was reaaaaaally wide!
- How to center the box?!
- How do you highlight?!

How do we get from this...

CS 193X: Web Fun

Announcements
4/3: Homework 0 is out! Due Friday.
4/3: Office hours are now posted.

[View Syllabus](#)



... to this?

CS 193X: Web Fun

Announcements
4/3: Homework 0 is out! **Due Friday.**
4/3: Office hours are now posted.

[View Syllabus](#)

Q: Why is HTML/CSS so bizarre??

A: There is one crucial set of rules we haven't learned yet...

block vs **inline** display

What is HTML?

HTML (Hypertext Markup Language)

- Describes the **content** and **structure** of a web page
- Made up of building blocks called **elements**.

<p>

HTML is awesome!!!

</p>

And there are 3 basic types.

Types of HTML elements

Each HTML element is categorized by the HTML spec into one of three-ish categories:

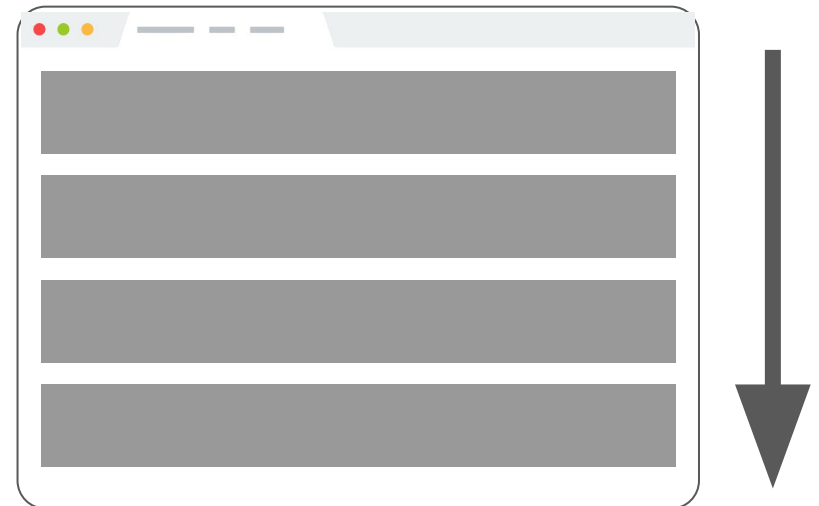
1. **block**: large blocks of content, has height and width
`<p>`, `<h1>`, `<blockquote>`, ``, ``, `<table>`
2. **inline**: small amount of content, no height or width
`<a>`, ``, ``, `
`
 - a. **inline block**: inline content with height and width
``
3. **metadata**: information about the page, usually not visible
`<title>`, `<meta>`

Block elements

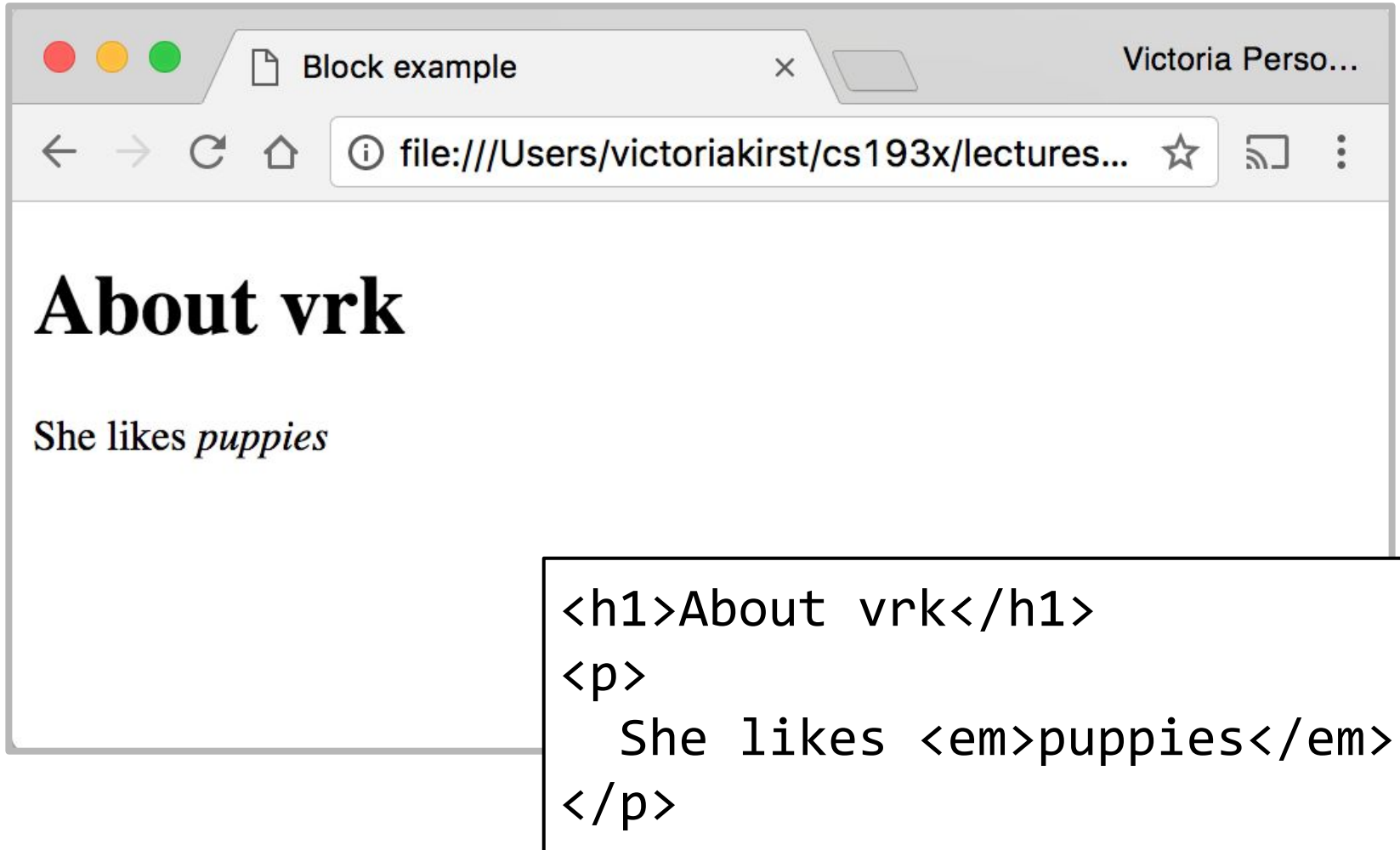
Examples:

`<p>`, `<h1>`, `<blockquote>`, ``, ``, `<table>`

- Take up the full width of the page (**flows top to bottom**)
- Have a height and width
- Can have block or inline elements as children



Example: Block



The image shows a browser window with a single tab titled "Block example". The address bar contains the file path "file:///Users/victoriakirst/cs193x/lectures...". The page content consists of a large heading "About vrk" and a paragraph "She likes *puppies*". A callout box with a black border points to the paragraph and contains the following HTML code:

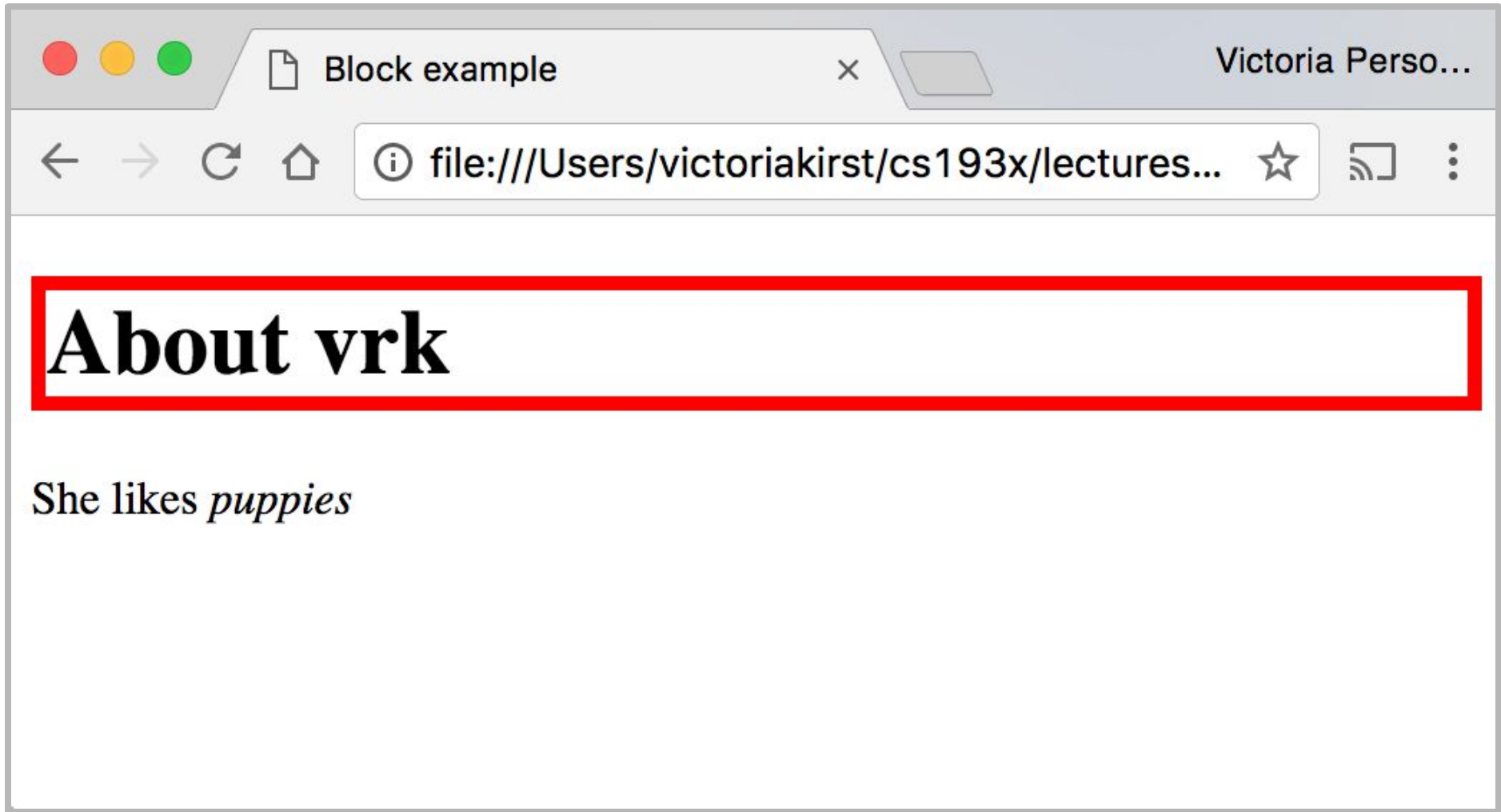
```
<h1>About vrk</h1>
<p>
  She likes <em>puppies</em>
</p>
```

Q: What does this look like in the browser?

```
h1 {  
  border: 5px solid red;  
}
```



```
<h1>About vrk</h1>  
<p>  
  She likes <em>puppies</em>  
</p>
```



([Codepen](#))

Block-level:

extends the full width of the page

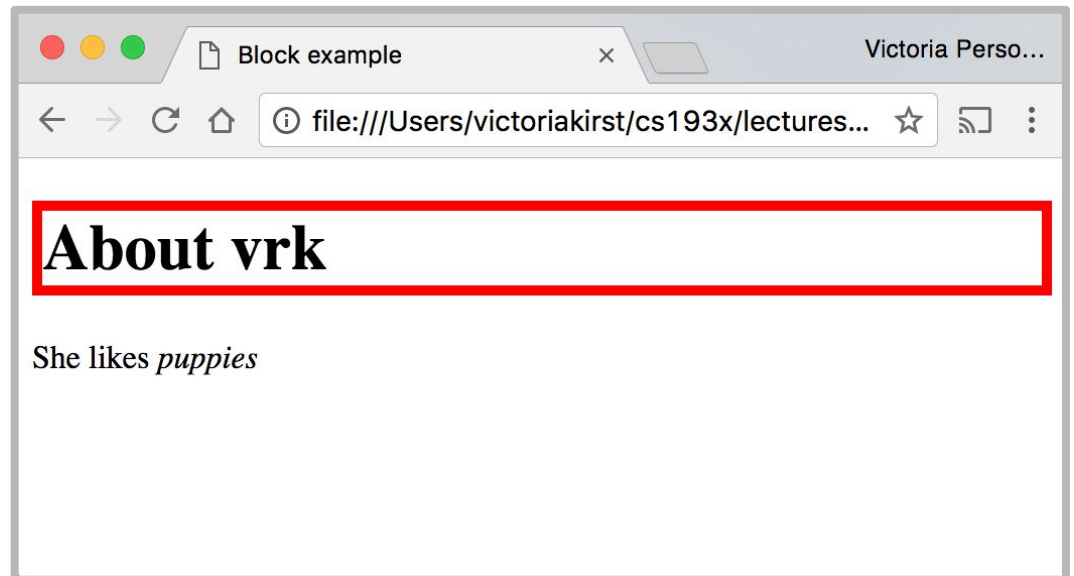
```
h1 {  
  border: 5px solid red;  
}
```

```
<h1>About vrk</h1>  
<p>  
  She likes <em>puppies</em>  
</p>
```

<h1> is block-level, so it extends the full width of the page by default

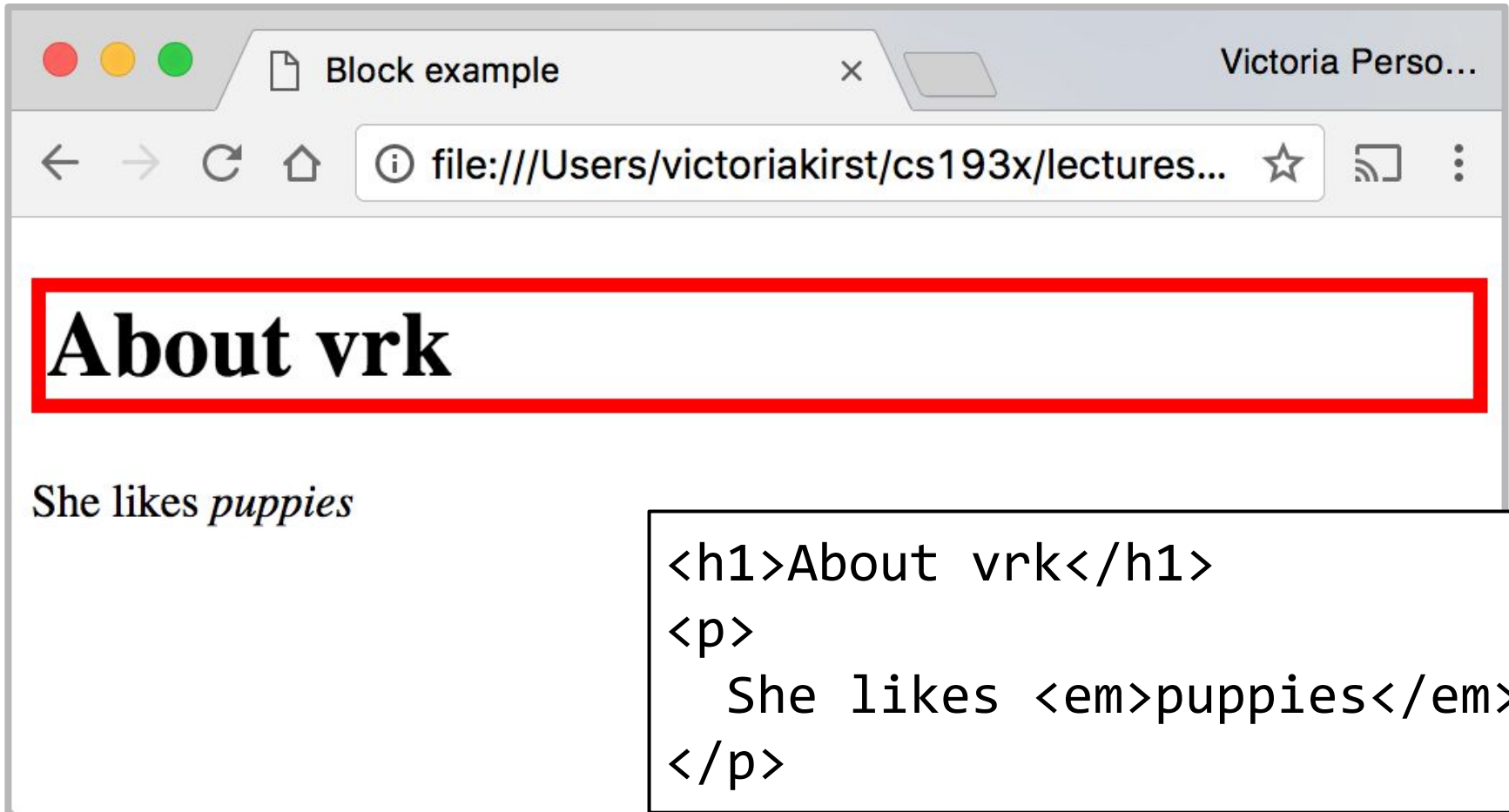
Note how block-level elements (**h1**, **p**) flow top to bottom

See: [Codepen](#)

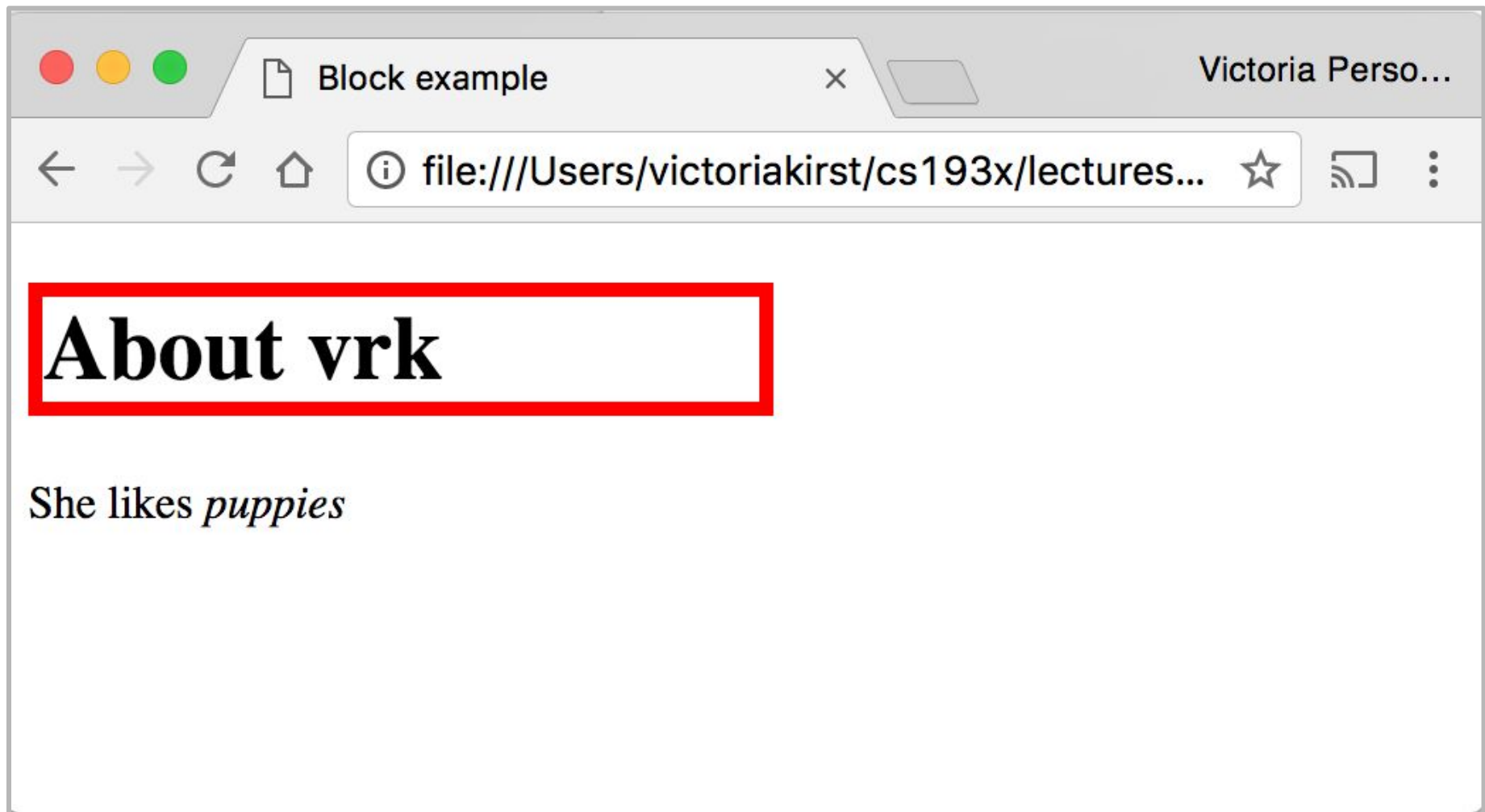


Q: What does this look like in the browser?

```
h1 {  
  border: 5px solid red;  
  width: 50%;  
}
```



```
<h1>About vrk</h1>  
<p>  
  She likes <em>puppies</em>  
</p>
```

([Codepen](#))

Block-level

width can be modified

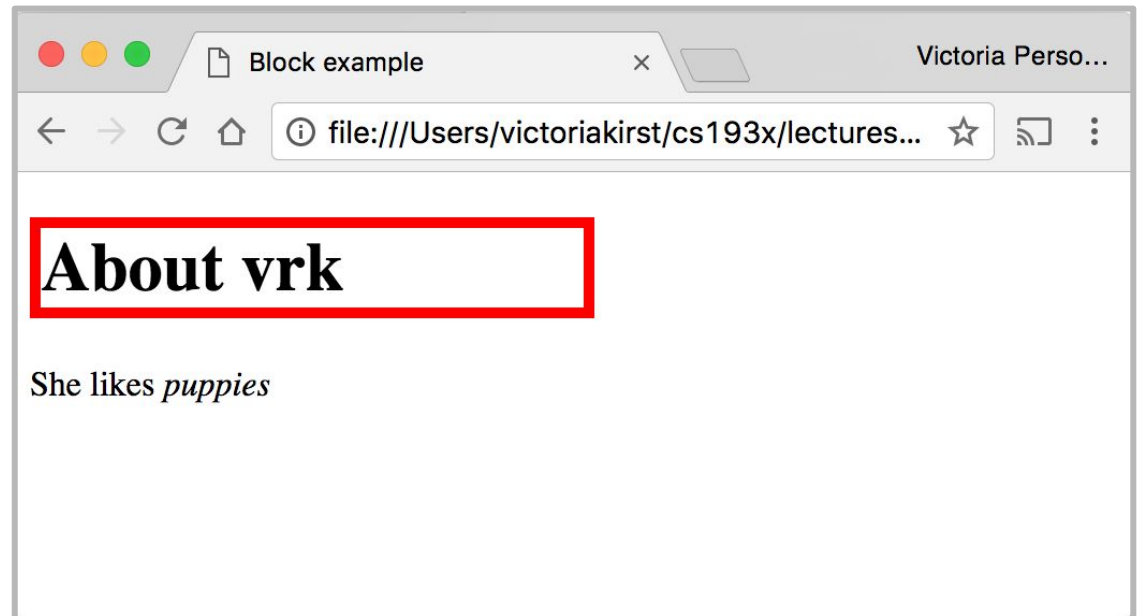
```
h1 {  
  border: 5px solid red;  
  width: 50%;  
}
```

```
<h1>About vrk</h1>  
<p>  
  She likes <em>puppies</em>  
</p>
```

`<h1>` is block-level,
so its **width** can be
modified

Block-level elements
still flow top to
bottom

See: [Codepen](#)

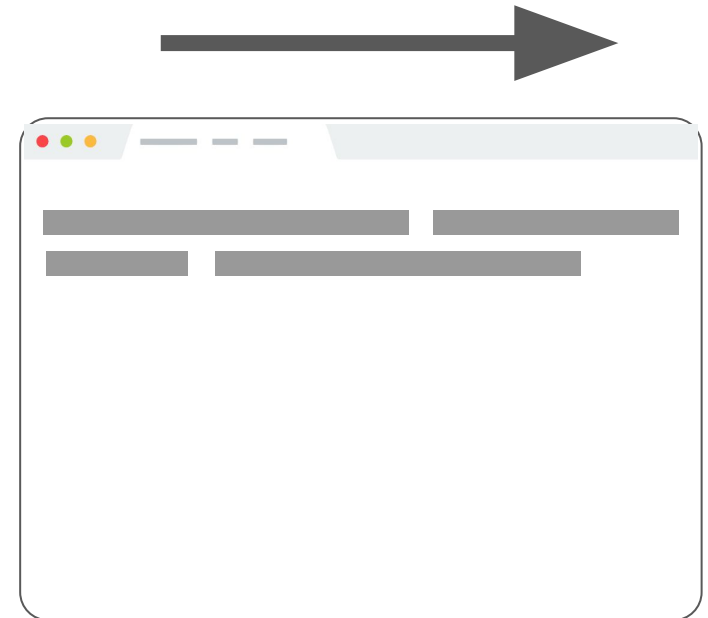


Inline elements

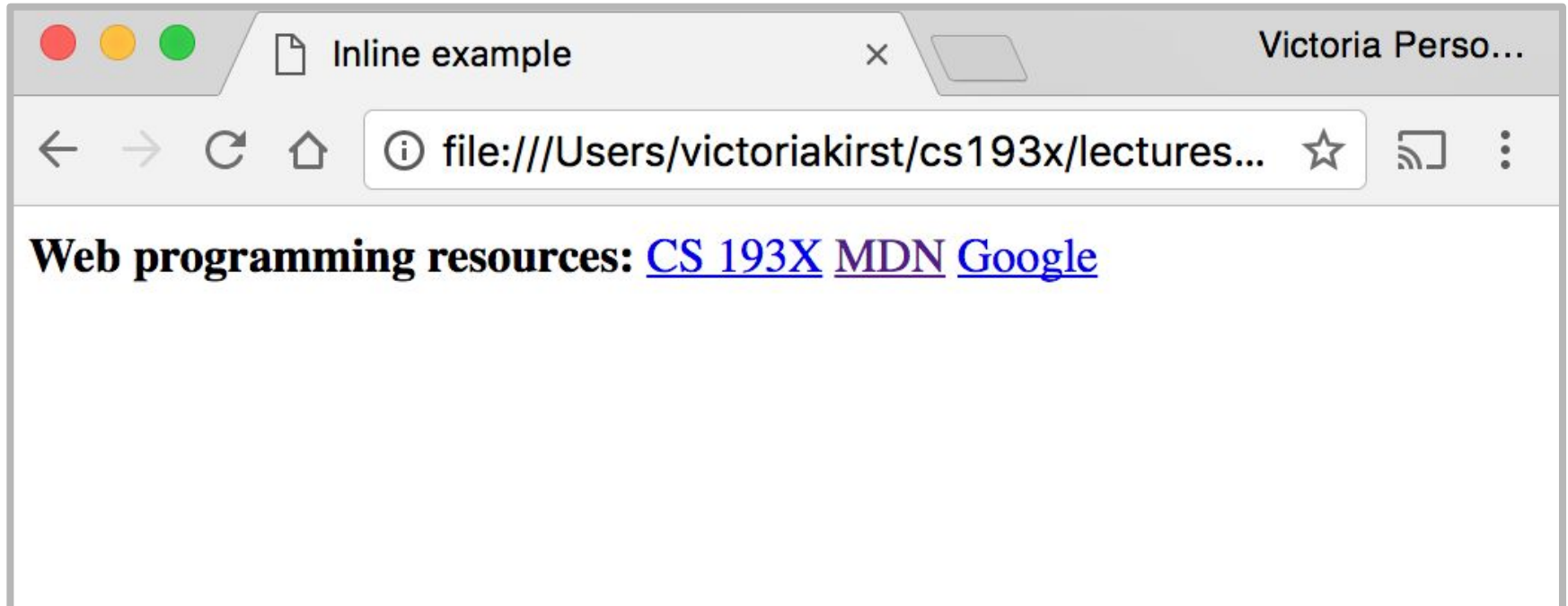
Examples:

`<a>`, ``, ``, `
`

- Take up only as much width as needed (flows left to right)
- **Cannot** have height and width
- **Cannot** have a block element child
- **Cannot** be positioned (i.e. CSS properties like `float` and `position` do not apply to inline elements)
 - Must position **its containing block element** instead



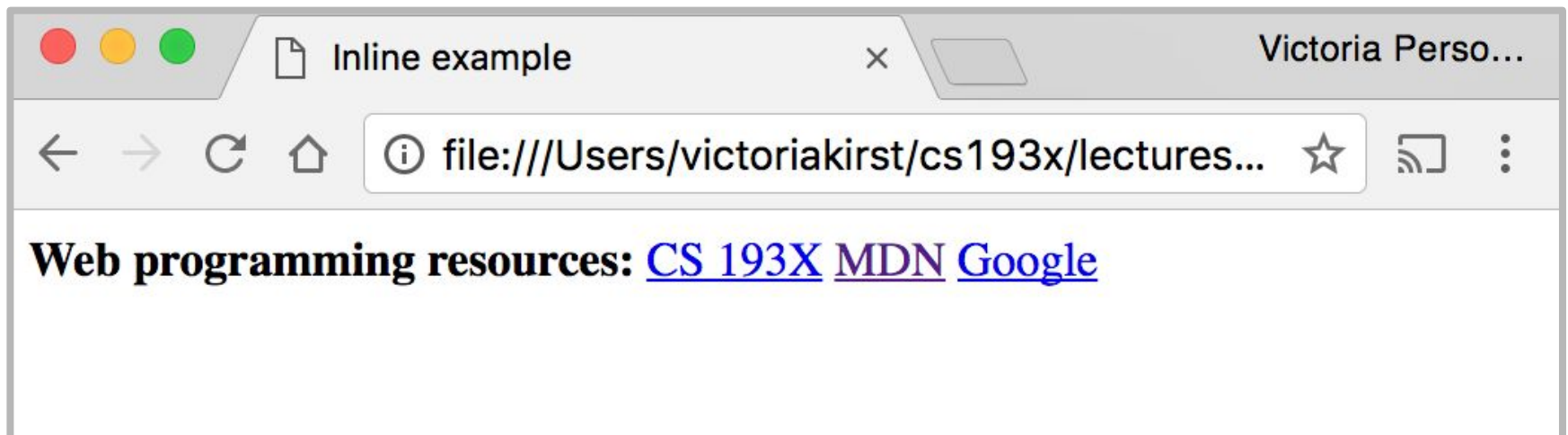
Example: Inline



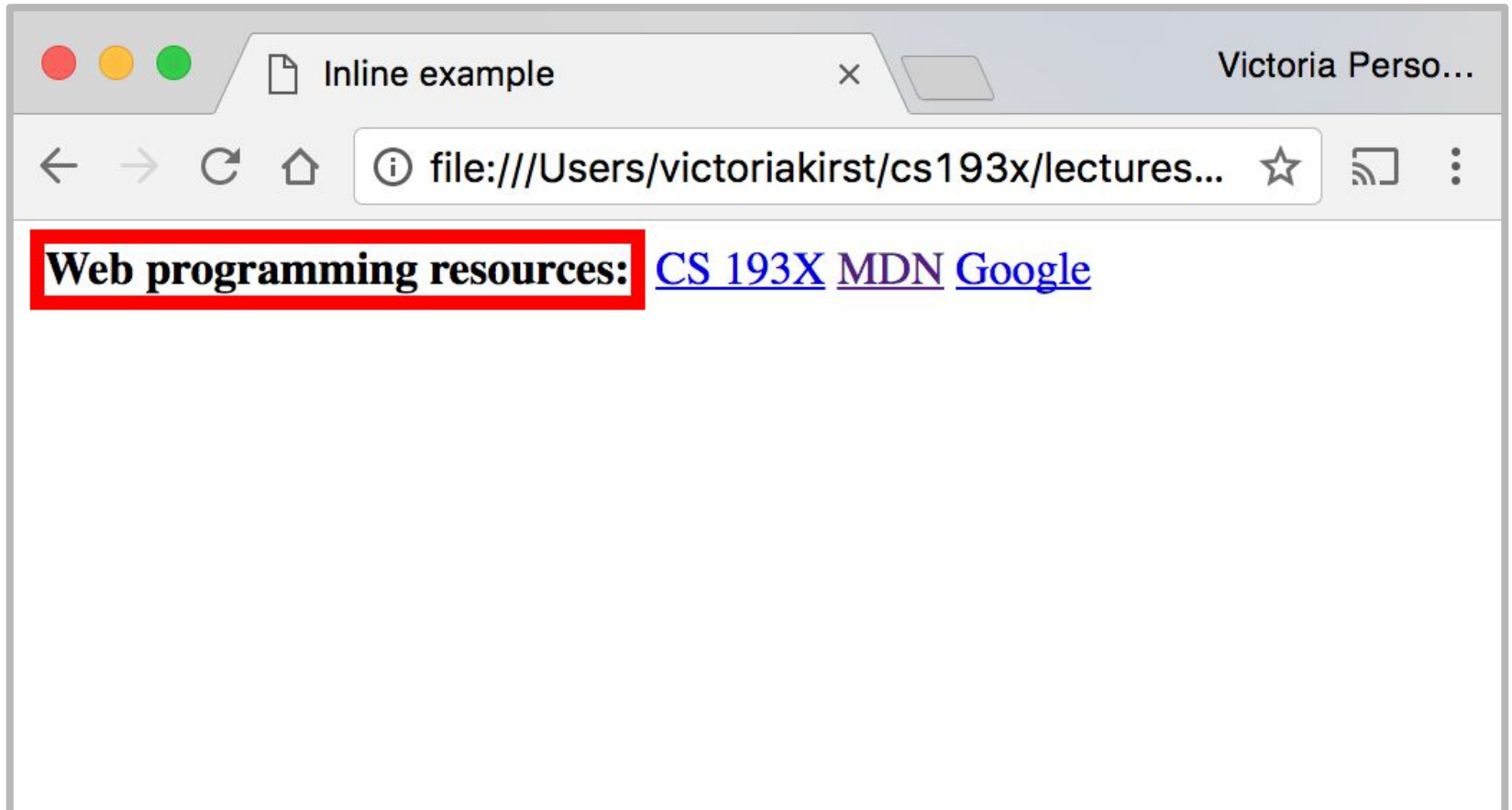
```
<strong>Web programming resources:</strong>  
<a href="http://cs193x.stanford.edu">CS 193X</a>  
<a href="https://developer.mozilla.org/en-US/">MDN</a>  
<a href="http://google.com">Google</a>
```

Q: What does this look like in the browser?

```
strong {  
  border: 5px solid red;  
  width: 1000px;  
}
```



```
<strong>Web programming resources:</strong>  
<a href="http://cs193x.stanford.edu">CS 193X</a>  
<a href="https://developer.mozilla.org/en-US/">MDN</a>  
<a href="http://google.com">Google</a>
```



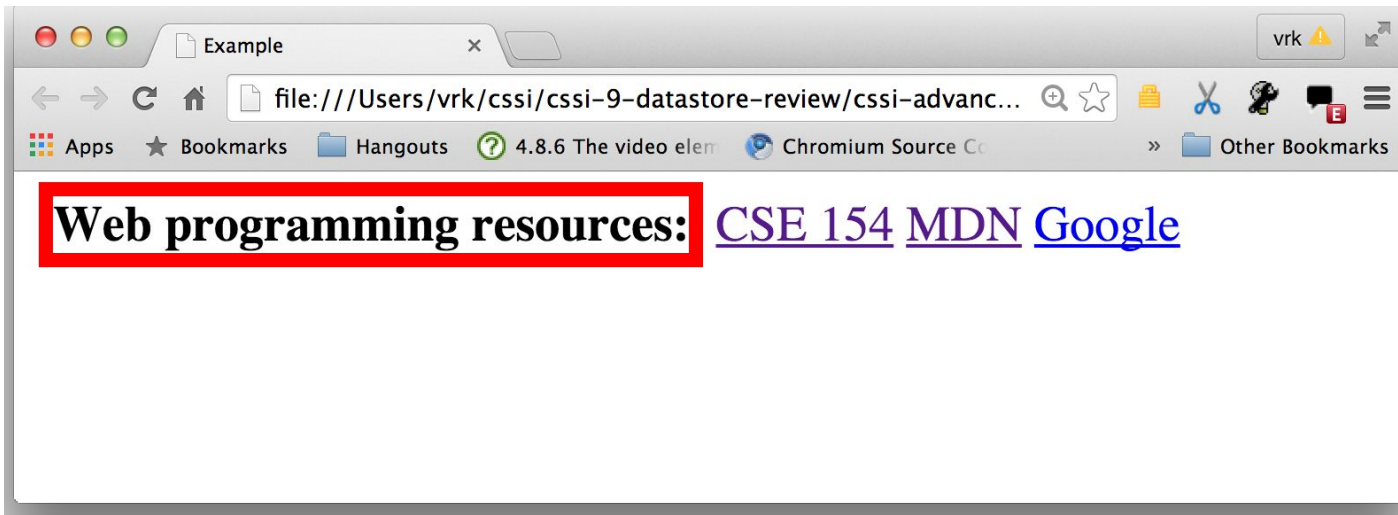
([Codepen](#))

Inline elements ignore width

width cannot be modified

```
strong {  
  border: 5px solid red;  
  width: 1000px;  
  /* Will not work; strong is  
    inline! */  
}
```

```
<strong>Web programming reso  
<a href="http://cs193x.stanf  
<a href="https://developer.m  
<a href="http://google.com">
```

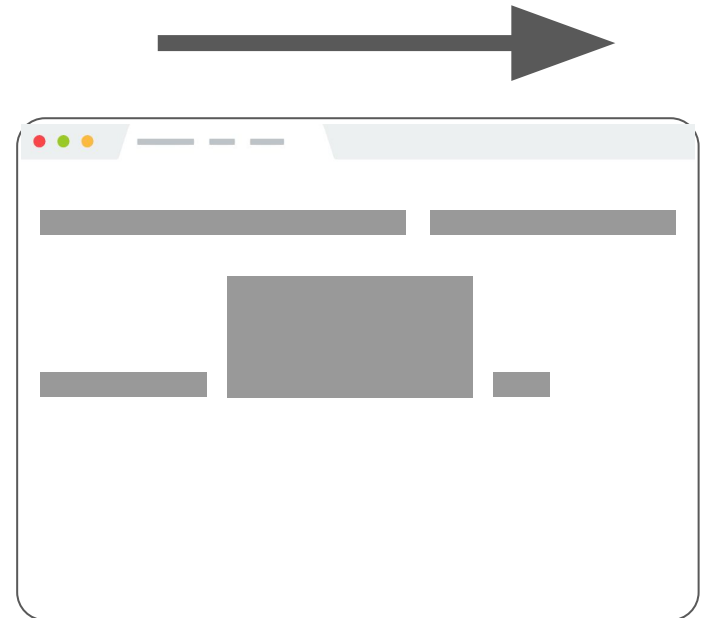


Cannot set **width** on inline element, so it is ignored ([Codepen](#))

inline-block

Examples: ``, any element with `display: inline-block;`

- Width is the size of the content, i.e. it takes only as much space as needed (flows left to right)
- **Can** have height and width
- **Can** have a block element as a child
- **Can** be positioned (i.e. CSS properties like `float` and `position` apply)



Example: Inline-block

```
img {  
  width: 50px;  
}
```

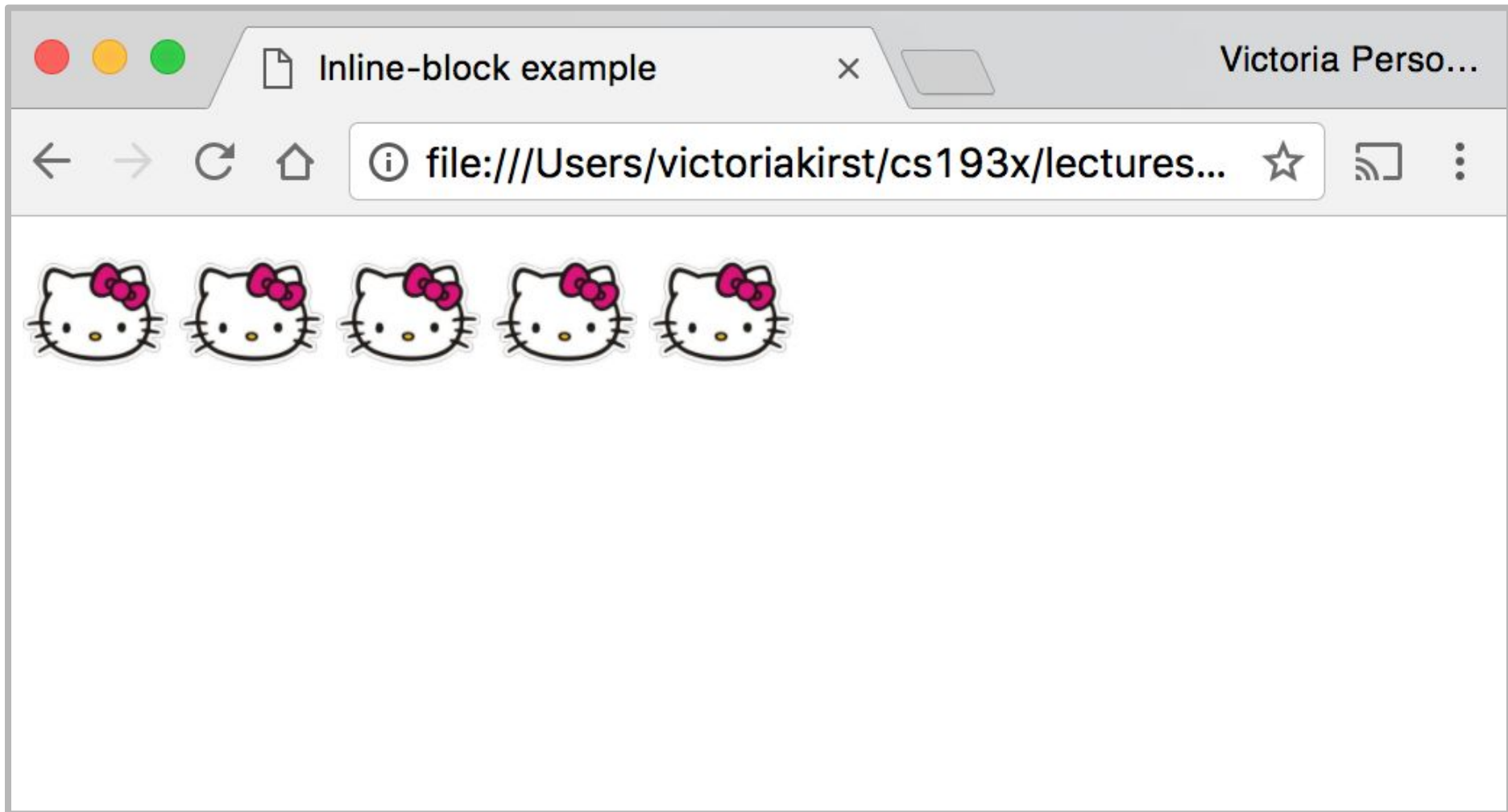
Q: What does this look like in the browser?

```
  
  
  
  

```

<http://i.imgur.com/WJToVGv.jpg> =





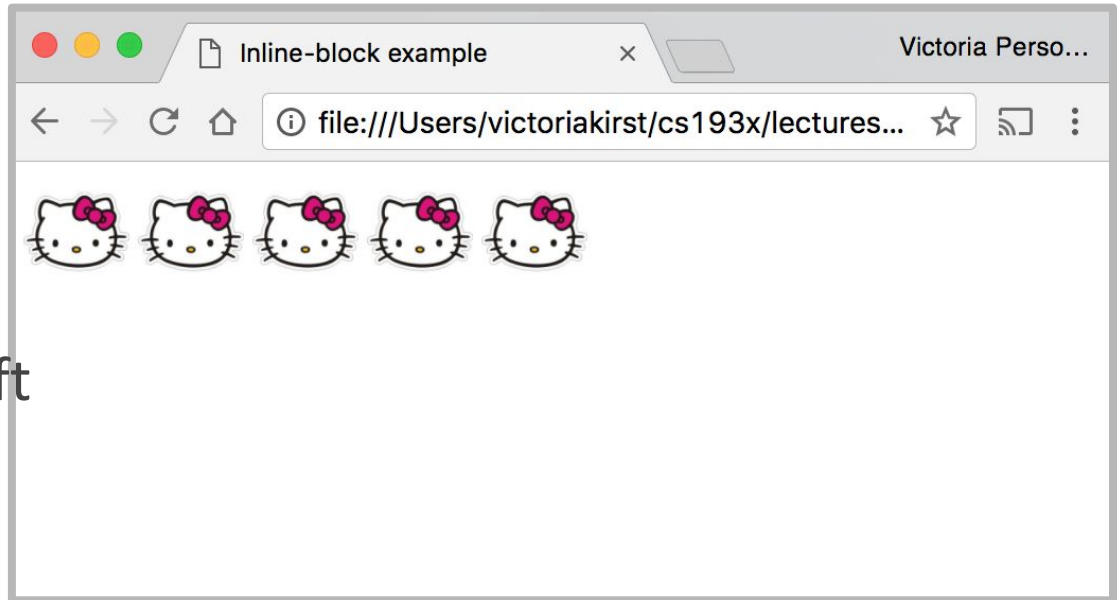
([Codepen](#))

Inline-block

Has width and height; flows left to right

Can set **width** on inline-block element, so image width is set to 50px. ([Codepen](#))

inline-block flows left to right, so images are right next to each other.



```
img {  
  width: 50px;  
}
```

```
  
  
  
  

```

The display CSS property

You can change an element's default rendering type by changing the **display** property. Examples:

```
p {  
  display: inline;  
}
```

```
a {  
  display: block;  
}
```

Possible values for `display`:

- `block`
- `inline`
- `inline-block`
- some others: [link](#)

Review

1. **block**: flows **top-to-bottom**; **has** height and width
<p>, <h1>, <blockquote>, , , <table>
2. **inline**: flows **left-to-right**; **does not have** height and width
<a>, , ,

 - a. **inline block**: flows **left-to-right**; **has** height and width
equal to size of the content

Questions?

Moral of the story:

If your CSS isn't working, see if you're trying to apply block-level properties to inline elements

h1 vs strong mystery

Recall: Weirdly the `<h1>` heading was on a line of its own, and `` was not. -- **Why?**

```
<h1>CS 193X: Web Fun</h1>
<strong>Announcements</strong>
4/3: Homework 0 is out!
```

CS 193X: Web Fun

Announcements 4/3: Homework 0 is out!

```
<h1>CS 193X: Web Fun</h1>
<strong>Announcements</strong><br/>
4/3: Homework 0 is out!
```

CS 193X: Web Fun

Announcements
4/3: Homework 0 is out!

h1 vs strong demystified!

Recall: Weirdly the `<h1>` heading was on a line of its own, and `` was not. -- **Why?**

```
<h1>CS 193X: Web Fun</h1>
<strong>Announcements</strong>
4/3: Homework 0 is out!
```

CS 193X: Web Fun

Announcements 4/3: Homework 0 is out!

```
<h1>CS 193X: Web Fun</h1>
<strong>Announcements</strong><br/>
4/3: Homework 0 is out!
```

CS 193X: Web Fun

Announcements
4/3: Homework 0 is out!

**Because h1 is a block-level element,
and strong is an inline-level element**

text-align mystery

Recall: We couldn't set `text-align: center;` on the `<a>` tag directly, but we could center `<h1>`. **Why?**

```
h1 { /* works! */  
  text-align: center;  
}  
  
a { /* fails :( */  
  text-align: center;  
}
```

CS 193X: Web Fun

Announcements

4/3: Homework 0 is out! Due Friday.

4/3: Office hours are now posted.

[View Syllabus](#)

Let's try looking at the [MDN description of text-align](#)...

text-align mystery

Summary

The **text-align** CSS property describes how inline content like text is aligned in its parent block element. **text-align** does not control the alignment of block elements, only their inline content.

Initial value

start, or a nameless value that acts as left if **direction** is ltr, right if **direction** is rtl if start is not supported by the browser.

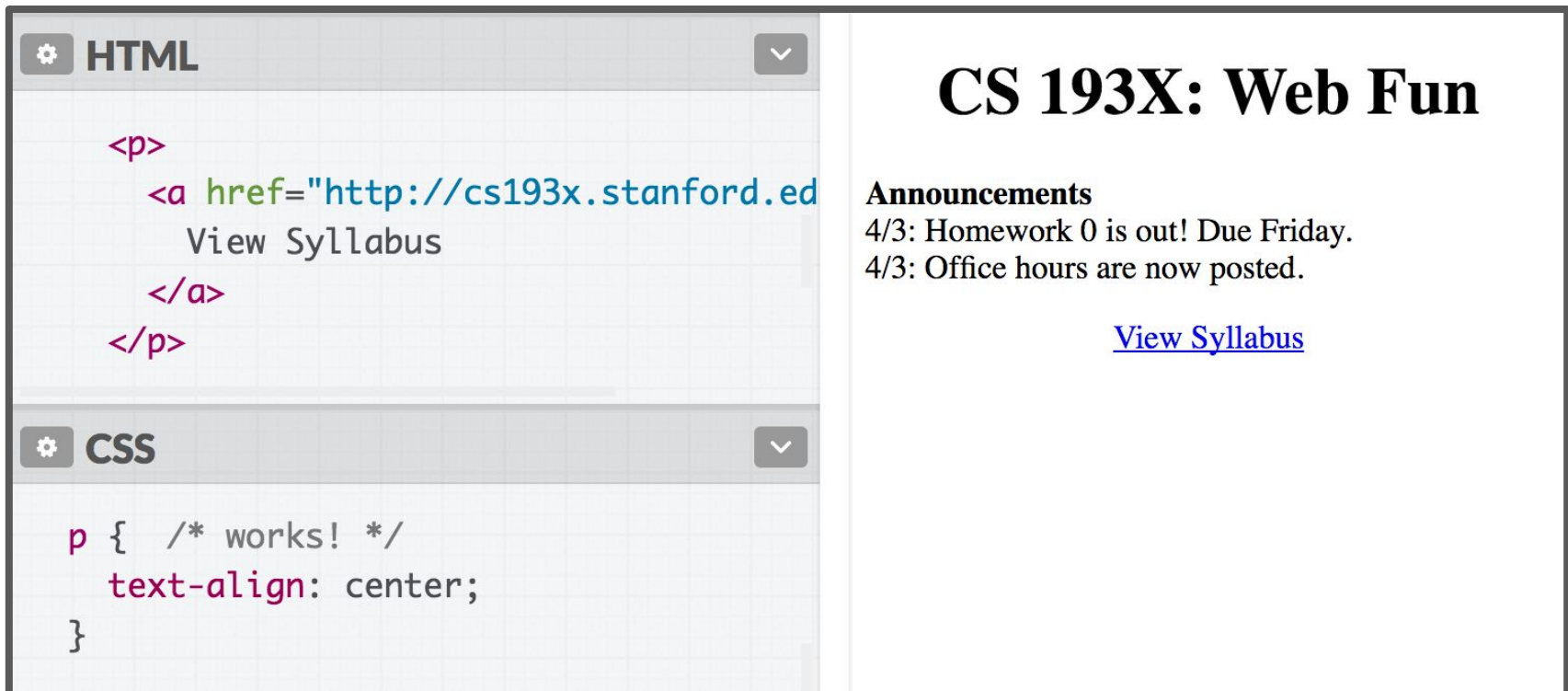
Applies to

block containers

[\(source\)](#)

text-align demystified!

Why? From the [spec](#), can't apply text-align to an inline element; must apply text-align to its block container, or set `a { display : block; }`



The screenshot shows a web browser interface. On the left, there are two panels: 'HTML' and 'CSS'. The HTML panel contains the following code:

```
<p>
  <a href="http://cs193x.stanford.ed
    View Syllabus
  </a>
</p>
```

The CSS panel contains the following code:

```
p { /* works! */
  text-align: center;
}
```

On the right, the rendered output is shown. It features a large heading 'CS 193X: Web Fun'. Below the heading, there is a section titled 'Announcements' with two lines of text: '4/3: Homework 0 is out! Due Friday.' and '4/3: Office hours are now posted.'. At the bottom of the announcements, there is a blue, underlined link that says 'View Syllabus'.

Box size mystery

Recall: The pink box we put around the announcements extended the entirety of the page.

```
p {  
  border: 3px solid hotpink;  
  background-color: lavenderblush;  
}
```

CS 193X: Web Fun

Announcements

4/3: Homework 0 is out! Due Friday.
4/3: Office hours are now posted.

[View Syllabus](#)

Why?

How do we fix this?

Box size mystery

Recall: The pink box we put around the announcements extended the entirety of the page.

```
p {  
  border: 3px solid hotpink;  
  background-color: lavenderblush;  
}
```

CS 193X: Web Fun

Announcements

4/3: Homework 0 is out! Due Friday.
4/3: Office hours are now posted.

[View Syllabus](#)

Why? Because `p` is block-level, so `width == width of the page`

How do we fix this?

Box size mystery: demystified!

Recall: The pink box we put around the announcements extended the entirety of the page.

```
p {  
  border: 3px solid hotpink;  
  background-color: lavenderblush;  
  display: inline-block;  
}
```

CS 193X: Web Fun

Announcements

4/3: Homework 0 is out! Due Friday.
4/3: Office hours are now posted.

[View Syllabus](#)

Why? Because `p` is block-level, so `width == width of the page`

How do we fix this? Change `display` to `inline-block` (though now the space above the box has increased... will address later!)

Centering the box

We can also center the box by centering the body tag, since p is now inline-block.

```
body {  
  text-align: center;  
}  
  
p {  
  border: 3px solid hotpink;  
  background-color: lavenderblush;  
  display: inline-block;  
}
```

CS 193X: Web Fun

Announcements

4/3: Homework 0 is out! Due Friday.
4/3: Office hours are now posted.

[View Syllabus](#)

Highlight mystery

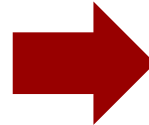
Recall: We didn't know how to select a random snippet of text to change its background.

CS 193X: Web Fun

Announcements

4/3: Homework 0 is out! Due Friday.
4/3: Office hours are now posted.

[View Syllabus](#)



CS 193X: Web Fun

Announcements

4/3: Homework 0 is out! **Due Friday.**
4/3: Office hours are now posted.

[View Syllabus](#)

How do we fix this?

Highlight: demystified!

We can select a random segment of text by wrapping it in an **inline element**:



The image shows a side-by-side comparison of HTML code and its rendered output. On the left, the 'HTML' pane of developer tools displays the following code:

```
<strong>Announcements</strong><br/>
4/3: Homework 0 is out!
<em>Due Friday.</em><br/>
4/3: Office hours are now posted.
```

Below it, the 'CSS' pane shows a rule for the `em` element:

```
em {
  font-style: normal; /* undoes italics */
  background-color: yellow;
}
```

On the right, the rendered page snippet shows the result of this code. The page title is 'CS 193X: Web Fun'. Below it, the word 'Announcements' is bolded. The text '4/3: Homework 0 is out! Due Friday.' is wrapped in a pink box, and the words 'Due Friday.' are highlighted in yellow. The text '4/3: Office hours are now posted.' is below it. A blue link 'View Syllabus' is at the bottom.

Hmmm... but wouldn't it be better to have a "highlight" element?

Highlight: demystified!

We can select a random segment of text by wrapping it in an inline element:

The image shows a browser's developer tools interface. On the left, the 'HTML' panel displays the following code:

```
<strong>Announcements</strong><br/>
4/3: Homework 0 is out!
<em>Due Friday.</em><br/>
4/3: Office hours are now posted.
```

The code for the `Due Friday.` tag is circled in red. Below it, the 'CSS' panel shows a rule for the `em` element:

```
em {
  font-style: normal; /* undoes italics */
  background-color: yellow;
}
```

On the right, the rendered page content is shown. The text 'CS 193X: Web Fun' is at the top. Below it, the text 'Announcements' is bolded. The text '4/3: Homework 0 is out! Due Friday.' is highlighted in yellow, with 'Due Friday.' being the part that was highlighted in the HTML code. Below that, the text '4/3: Office hours are now posted.' is visible. At the bottom, there is a blue link that says 'View Syllabus'.

Hmmm... but wouldn't it be better to have a "highlight" element?
How do we make a generic HTML element?

Have you heard of `<div>` and ``?

What are they?

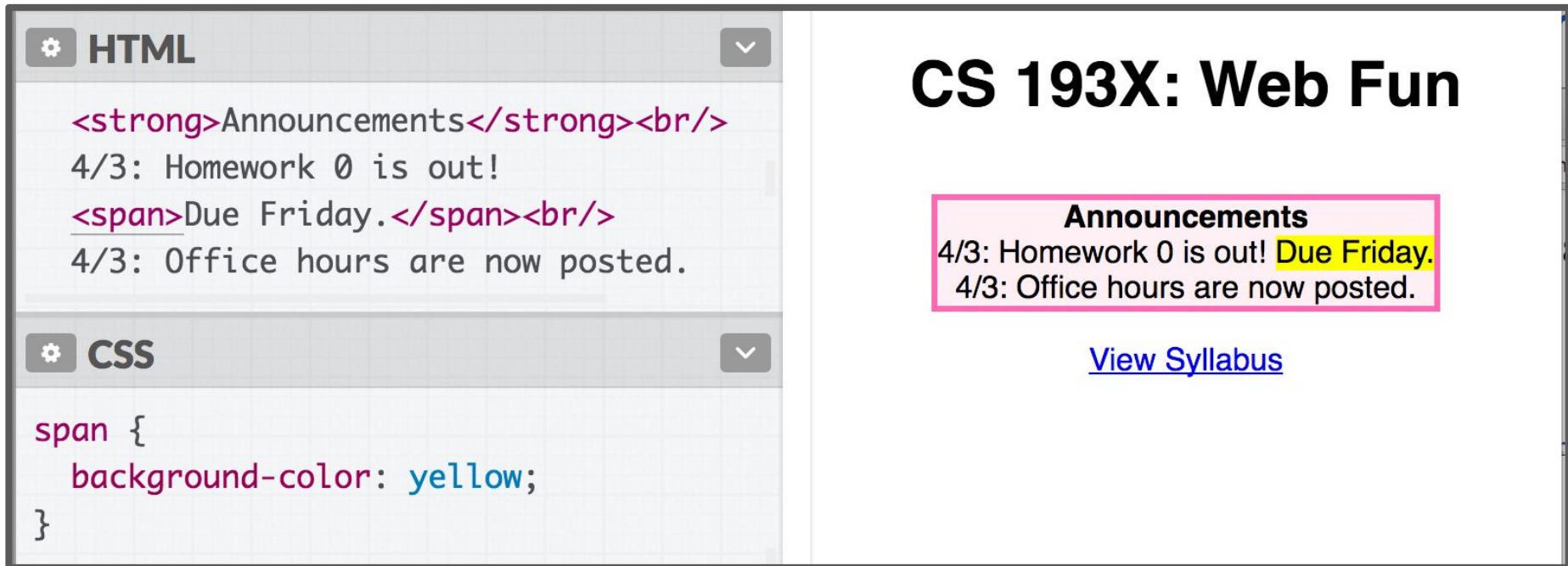
<div> and

Two generic tags with no intended purpose or style:

- <div>: a generic **block** element
- : a generic **inline** element

 in action

We can use as a generic inline HTML container:



```
HTML
<strong>Announcements</strong><br/>
4/3: Homework 0 is out!
<span>Due Friday.</span><br/>
4/3: Office hours are now posted.

CSS
span {
  background-color: yellow;
}
```

CS 193X: Web Fun

Announcements
4/3: Homework 0 is out! **Due Friday.**
4/3: Office hours are now posted.

[View Syllabus](#)

Multiple generic containers?

But won't we often want multiple generic containers?

How do we distinguish two generic containers?

In other words, how do we select a subset of elements instead of **all** elements on the page?

CS 193X: Web Fun

Announcements

4/3: Homework 0 is out! Due Friday.

4/3: Office hours are now posted.

[View Syllabus](#)

CSS Selectors: Classes and Ids

Classes and ids

There are 3 basic types of CSS selectors:

Element selector (this is the one we've been using)	p	All <p> elements
✨ ID selector ✨	#abc	element with id="abc"
✨ Class selector ✨	.abc	elements with class="abc"

```
<h1 id="title">Homework</h1>  
<em class="hw">HW0</em> is due Friday.<br/>  
<em class="hw">HW1</em> goes out Monday.<br/>  
<em>All homework due at 11:59pm.</em>
```


Classes and ids

```
<h1 id="title">Homework</h1>  
<em class="hw">HW0</em> is due Friday.<br/>  
<em class="hw">HW1</em> goes out Monday.<br/>  
<em>All homework due at 11:59pm.</em>
```

```
.hw {  
  color: hotpink;  
}  
  
#title {  
  color: purple;  
}
```

Homework

HW0 is due Friday.

HW1 goes out Monday.

All homework due at 11:59pm.

More on `class` and `id`

- `class` and `id` are special HTML attributes that can be used on any HTML element
 - **class**: Used on 1 or more elements; identifies a **collection** of elements
 - **id**: Used on exactly 1 element per page; identifies **one unique** element
- Can apply multiple classes by space-separating them:
`HW1`
- Often used with `span` and `div` to create generic elements: e.g. `` is like creating a "highlight" element

Other selectors:
Next time!