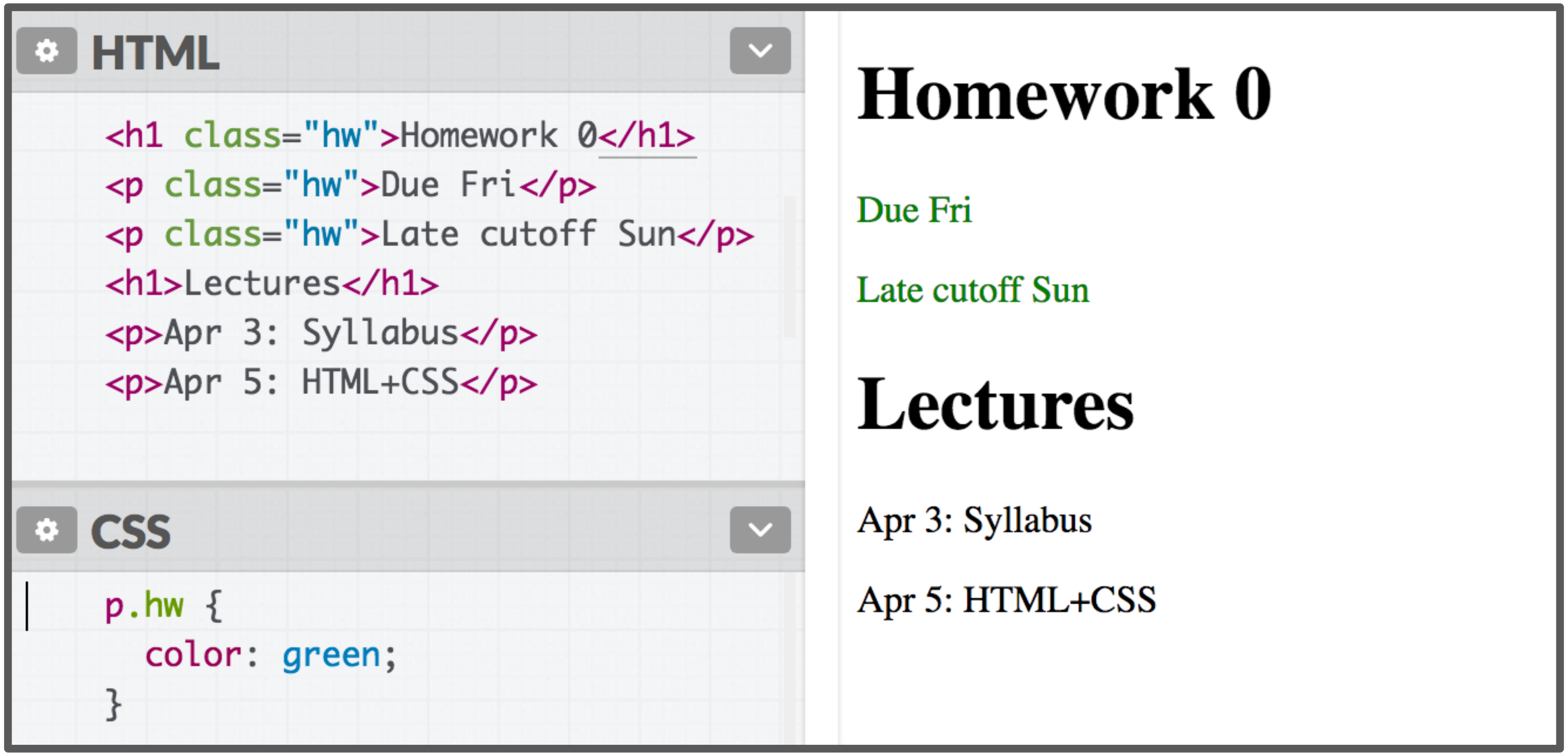
Other selectors

- Complex selectors
- Colliding styles
- Inheritance
- Pseudo classes

element.className

Syntax	Example	Example described
element.className	**p.abc**	**<p>** elements with abc class



```html
<h1 class="hw">Homework 0</h1>
<p class="hw">Due Fri</p>
<p class="hw">Late cutoff Sun</p>
<h1>Lectures</h1>
<p>Apr 3: Syllabus</p>
<p>Apr 5: HTML+CSS</p>
```

```css
p.hw {
  color: green;
}
```

Homework 0

Due Fri

Late cutoff Sun

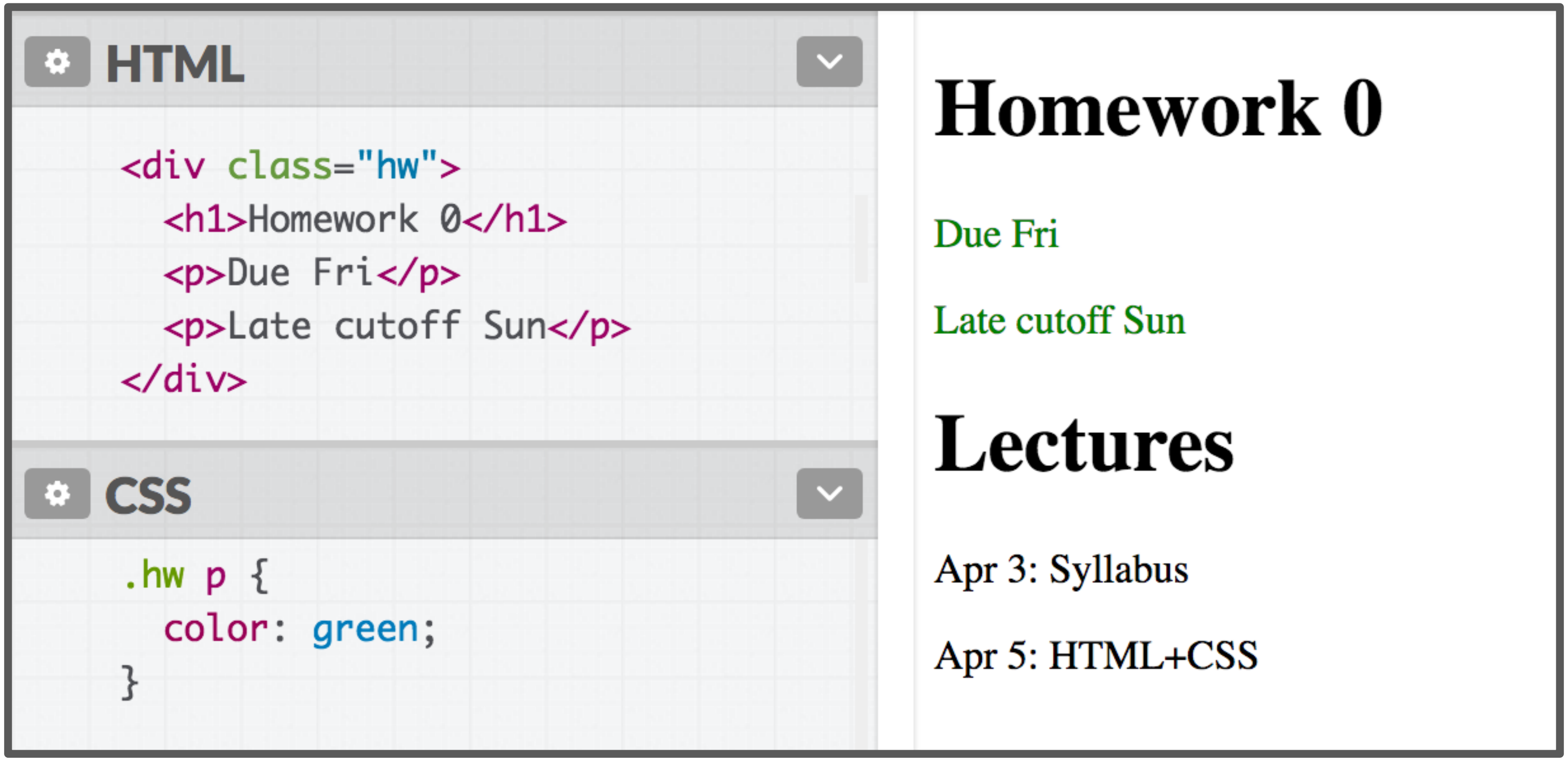
Lectures

Apr 3: Syllabus

Apr 5: HTML+CSS

Descendent selector

Syntax	Example	Example described
selector selector	**`div strong`**	**`<strong>`** elements that are descendants of a **`<div>`**



HTML

```
<div class="hw">
  <h1>Homework 0</h1>
  <p>Due Fri</p>
  <p>Late cutoff Sun</p>
</div>
```

CSS

```
.hw p {
  color: green;
}
```

Homework 0

Due Fri

Late cutoff Sun

Lectures

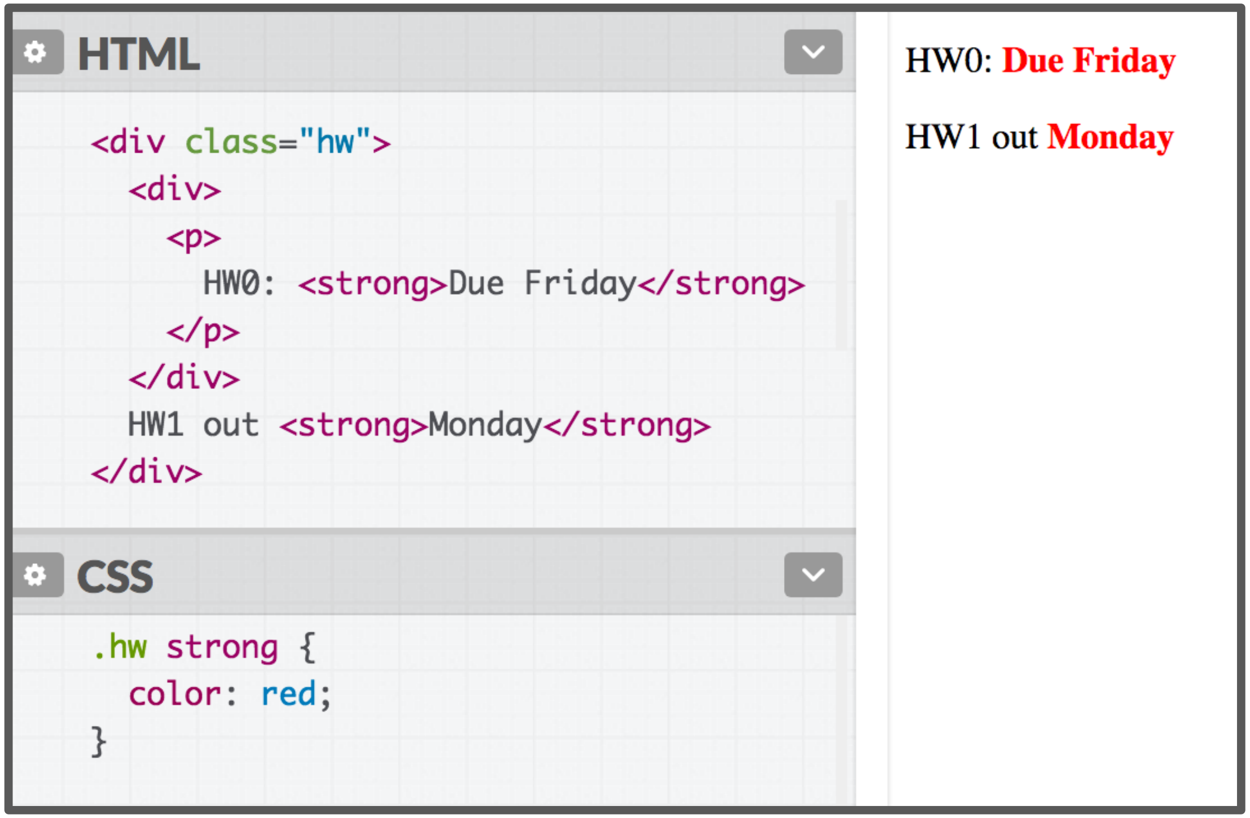
Apr 3: Syllabus

Apr 5: HTML+CSS

Descendent selector

Syntax	Example	Example described
selector selector	**`div strong`**	**`<strong>`** elements that are descendants of a **`<div>`**

Note: The element does not have to be a direct child. The descendent may be nested many layers in.



```html
<div class="hw">
  <div>
    <p>
      HW0: <strong>Due Friday</strong>
    </p>
  </div>
  HW1 out <strong>Monday</strong>
</div>
```

```css
.hw strong {
  color: red;
}
```

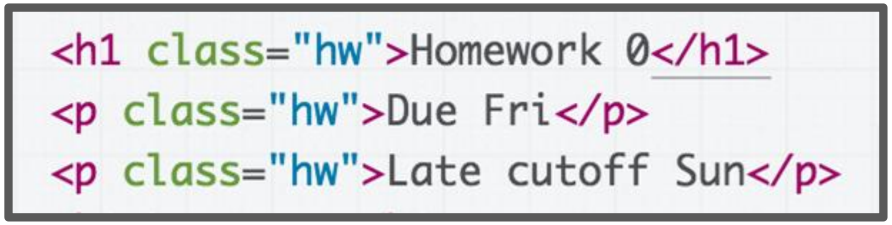
HW0: **Due Friday**

HW1 out **Monday**

Descendent selector

Syntax	Example	Example described
selector selector	`div strong`	**`<strong>`** elements that are descendants of a **`<div>`**

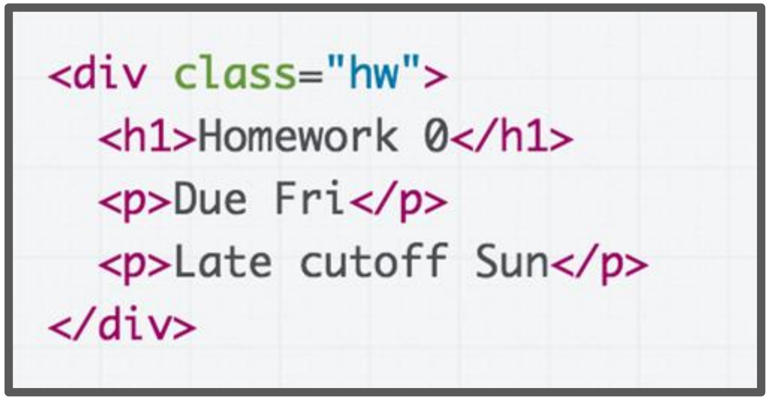
Discouraged:



```
<h1 class="hw">Homework 0</h1>
<p class="hw">Due Fri</p>
<p class="hw">Late cutoff Sun</p>
```

vs

Preferred:

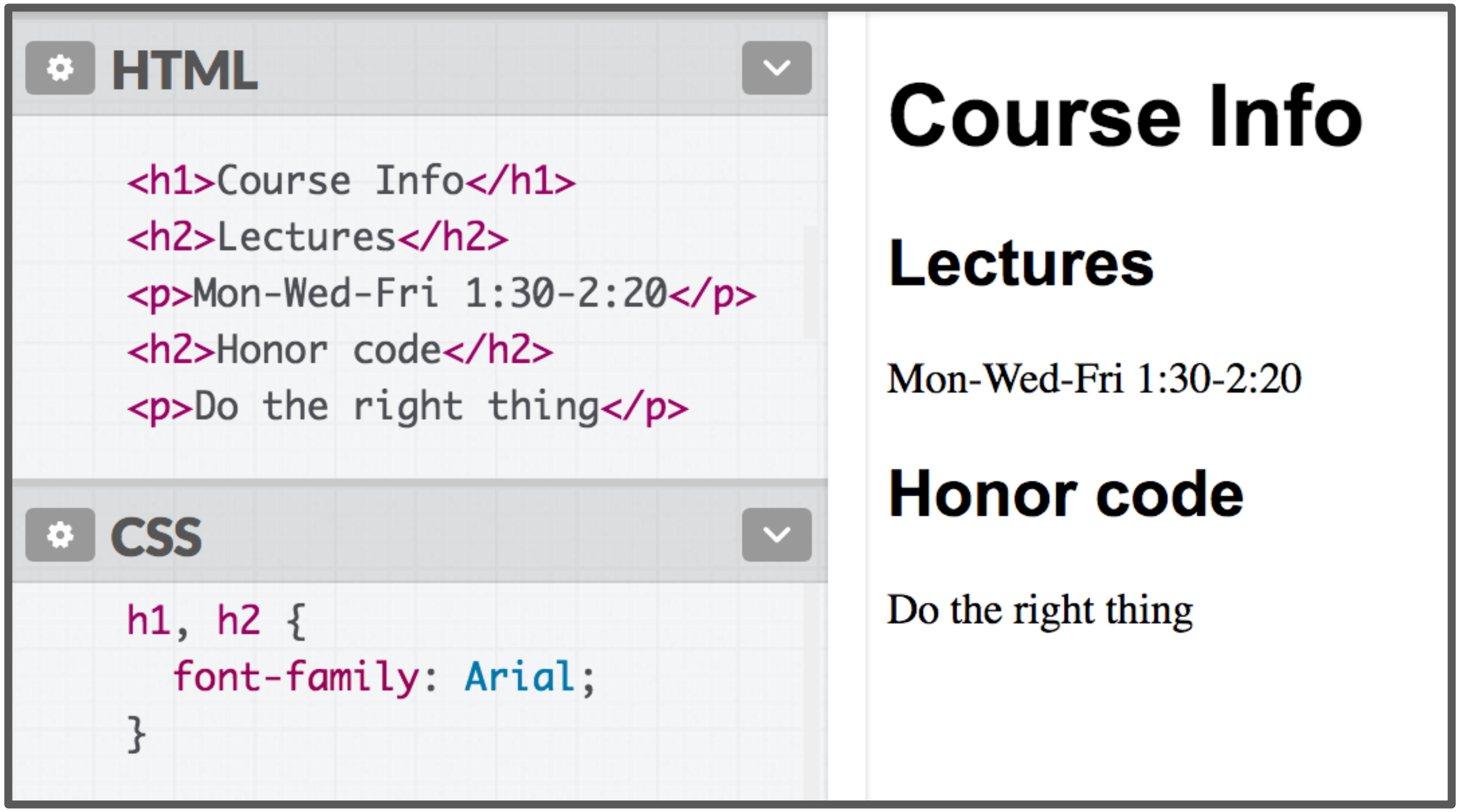


```
<div class="hw">
  <h1>Homework 0</h1>
  <p>Due Fri</p>
  <p>Late cutoff Sun</p>
</div>
```

Instead of applying a class to several adjacent elements, wrap the group in a `<div>` container and style the contents via descendent selectors.

selector, selector (comma)

Syntax	Example	Example described
selector*, *selector	**`h2, div`**	**`<h2>`** elements and **`<div>`**s



```html
<h1>Course Info</h1>
<h2>Lectures</h2>
<p>Mon-Wed-Fri 1:30-2:20</p>
<h2>Honor code</h2>
<p>Do the right thing</p>
```

```css
h1, h2 {
  font-family: Arial;
}
```

Course Info

Lectures

Mon-Wed-Fri 1:30-2:20

Honor code

Do the right thing

Selector summary

Example	Description
`p`	All **`<p>`** elements
`.abc`	All elements with the **`abc class`**, i.e. **`class="abc"`**
`#abc`	Element with the **`abc id`**, i.e. **`id="abc"`**
`p.abc`	**`<p>`** elements with **`abc class`**
`p#abc`	**`<p>`** element with **`abc id`** (**`p`** is redundant)
`div strong`	**`<strong>`** elements that are descendants of a **`<div>`**
`h2, div`	**`<h2>`** elements and **`<div>`**s

Grouping selectors

2 Common bugs:

`p.abc` **vs** `p .abc`

`p .abc` **vs** `p, .abc`

- A <p> element with the **abc** class **vs**
  An element with the **abc** class that descends from <**p**>

- An element with the **abc** class that descends from <**p**> **vs**
  All **<p>** elements *and* all elements with the **abc** class

Combining selectors

You can combine selectors:

```
#main li.important strong {
  color: red;
}
```

Q: What does this select?

Grouping selectors

Q: What does this select?

```
#main li.important strong {
  color: red;
}
```

A: Read from right to left:

- `<strong>` tags that are children of `<li>` tags that have an "`important`" class that are children of the element with the "`main`" id.

Colliding styles

When styles collide, the most specific rule wins (specificity)

```
div strong { color: red; }
strong { color: blue; }

<div>
     <strong>What color am I?</strong>
</div>
```

Colliding styles

When styles collide, the most specific rule wins (specificity)

```
div strong { color: red; }
strong { color: blue; }

<div>
      <strong>What color am I?</strong>
</div>
```

Colliding styles

Specificity precedence rules (details):

- `ids` are more specific than `classes`
- `classes` are more specific than element names
- Style rules that directly target elements are more specific than style rules that are inherited

Colliding styles

- If elements have the same specificity, the later rule wins.

```
strong { color: red; }
   strong { color: blue; }

   <div>
         <strong>What color am I?</strong>
   </div>
```

Aside: The process of figuring out what rule applies to a given element is called the cascade. This is where the "C" in *Cascading* Style Sheets comes from.

Inheritance

We saw earlier that CSS styles are inherited from parent to child.

Instead of selecting all elements individually:

```
a, h1, p, strong {
  font-family: Helvetica;
}
```

You can style the parent and the children will inherit the styles.

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

You can override this style via specificity:

```
h1, h2 {
  font-family: Consolas;
}
```

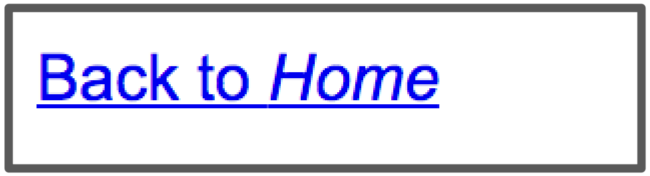
Inheritance

While many CSS styles are inherited from parent to child, **not all CSS properties are inherited**.

```
a {
  display: block;
  font-family: Arial;
}
```

```
<a href="/home">
  Back to <em>Home</em>
</a>
```

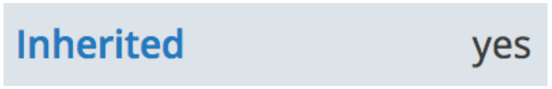
`<em>` inherits the `font-family` property, but not `display`:



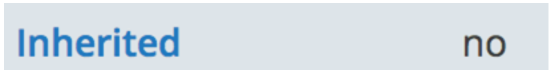
Back to *Home*

Inheritance

While many CSS styles are inherited from parent to child, **not all CSS properties are inherited**.

- There's no rule for what properties are inherited or not; the inheritance behavior defined in the CSS spec.
- You can look it up via MDN, e.g.
  `font-family`: 

Inherited	yes

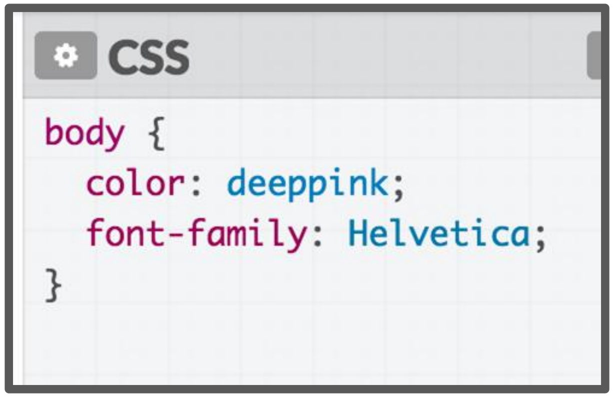
  `display`: 

Inherited	no

- Generally text-related properties are inherited and layout-related properties are not.
- (You can also change this via the `inherit` CSS property, which is somewhat esoteric and not often use)

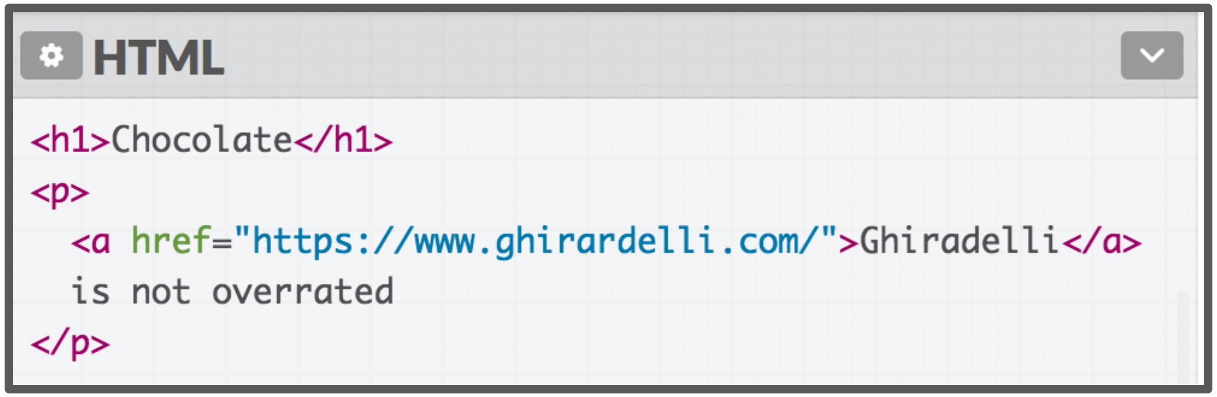
<a> colors?

Hmm, MDN says color is inherited... but if I set the body `color` to deeppink, links don't change color:



CSS

```css
body {
  color: deeppink;
  font-family: Helvetica;
}
```



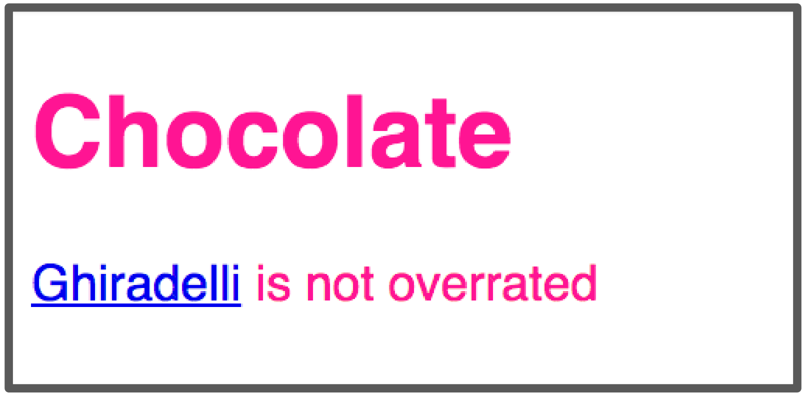
HTML

```html
<h1>Chocolate</h1>
<p>
  <a href="https://www.ghirardelli.com/">Ghiradelli</a>
  is not overrated
</p>
```

`<a>` inherits `font-family`...

Why doesn't `<a>` inherit color?

(Codepen)



Chocolate

Ghiradelli is not overrated

User agent styles

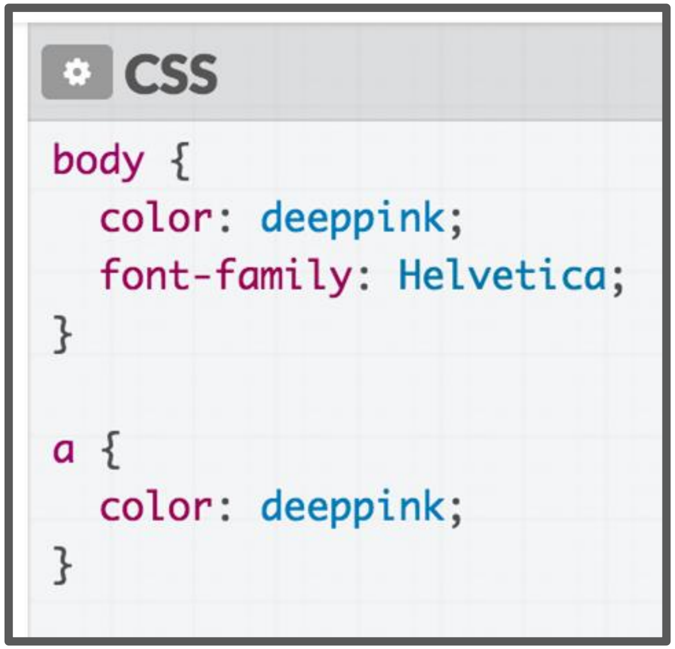
This is because the browser has its own default styles:

- Browser loads its own default stylesheet on every webpage
- Not governed by spec, but there are recommendations

```html
<!DOCTYPE html>
<html>
  <head>
    <title>CS 193X</title>
    <!--
      NOT TOTALLY ACCURATE: This isn't actually injected
      in the HTML, but it is loaded silently!
    -->
    <link rel="stylesheet" href="user-agent-style.css" />
  </head>
```

<a> colors?

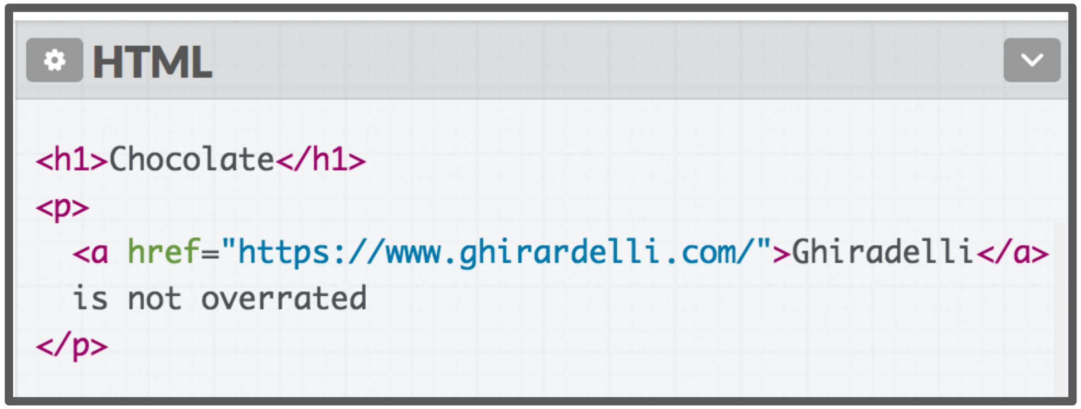
So to style <a> links, we have to override the browser default link style by explicitly setting a color:



CSS

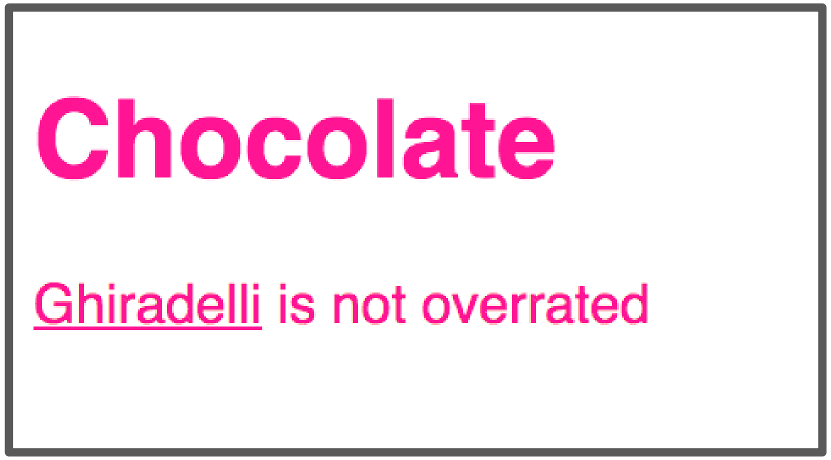
```
body {
  color: deeppink;
  font-family: Helvetica;
}

a {
  color: deeppink;
}
```



HTML

```
<h1>Chocolate</h1>
<p>
  <a href="https://www.ghirardelli.com/">Ghiradelli</a>
  is not overrated
</p>
```



Chocolate

Ghiradelli is not overrated

Link-related CSS

Since we're on the topic of links:

- How do we style **visited** links differently from **unvisited**?

CSS pseudo-classes

pseudo-classes: special keywords you can append to selectors, specifying a *state* or *property* of the selector

Syntax	Explanation
`a`	All anchor tags (links) in all states
`a:visited`	A visited link
`a:link`	An unvisited link
`a:hover`	The style when you hover over a link
`a:active`	The style when you have "activated" a link (downclick)

There are more pseudo-classes than this; have a look!