

Interactive Web Programming

1st semester of 2021

Murilo Camargos
(**murilo.filho@fgv.br**)

Heavily based on [Victoria Kirst](#) slides

Today's schedule

Schedule:

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

Reminders:

- [HW0](#) is due next Tuesday (09/03)

Announcements:

- The tentative syllabus with more details is out!

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!](#)

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>Programação Web Interativa</h1>  
<strong>Avisos</strong><br />  
01/03: Começaram nossas aulas!<br />
```

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

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;  
}
```

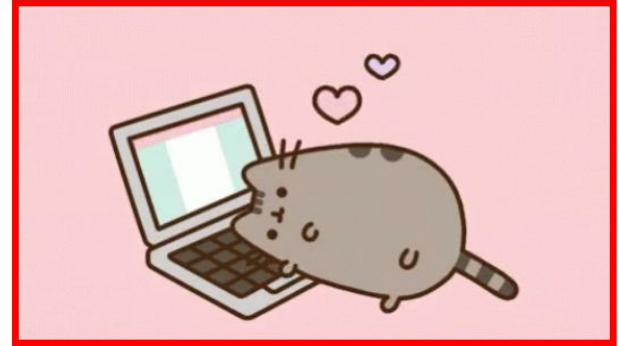
Moby Dick

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:

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](#)

Solution?!

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

Produces:

Programação Web Interativa

Avisos:

01/03: Começaram nossas aulas!

01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)

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 really wide!
- How to center the box?!
- How do you highlight?!

How do we get from this...

Programação Web Interativa

Avisos:
01/03: Começaram nossas aulas!
01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)



... to this?

Programação Web Interativa

Avisos:
01/03: Começaram nossas aulas!
01/03: A tarefa 0 está disponível. **Para Terça.**

[Ver Ementa](#)

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:

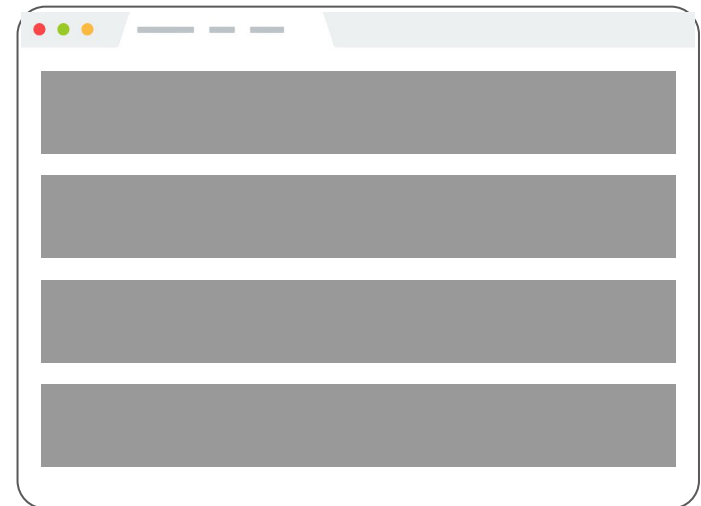
1. **block**: large blocks of content, has height and width
`<p>`, `<h1>`, `<blockquote>`, ``, ``, `<table>`
2. **inline**: small amount of content, no height or width
`<a>`, ``, ``, `
`
 - a. **inline block**: inline content with height and width
``
3. **metadata**: information about the page, usually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

Block elements

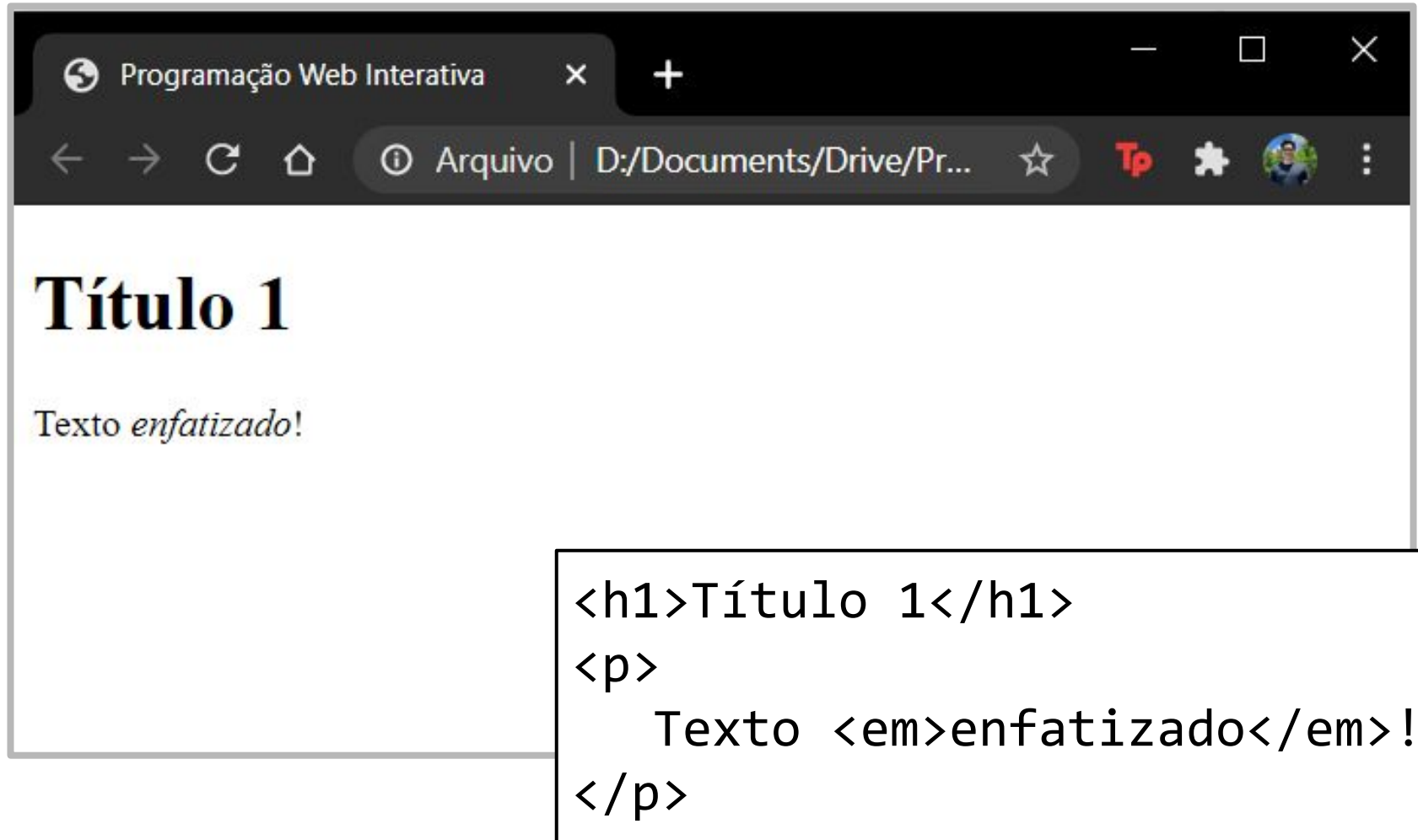
Examples:

`<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 dark theme. The address bar contains the text "Arquivo | D:/Documents/Drive/Pr...". The main content area displays a large, bold, black serif font heading "Título 1" followed by a paragraph of text "Texto *ênfatizado!*". A white box with a black border is positioned at the bottom right, containing the HTML code for the rendered content. A thin grey line connects the bottom left corner of this box to the text "Texto ênfatizado!" in the browser window.

```
<h1>Título 1</h1>  
<p>  
  Texto <em>ênfatizado</em>!  
</p>
```

Q: What does this look like in the browser?

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



```
<h1>Título 1</h1>  
<p>  
  Texto <em>enfatizado</em>!  
</p>
```

Programação Web Interativa



Arquivo | D:/Documents/Drive/Pr...



Título 1

Texto *ênfatisado!*

Block-level:

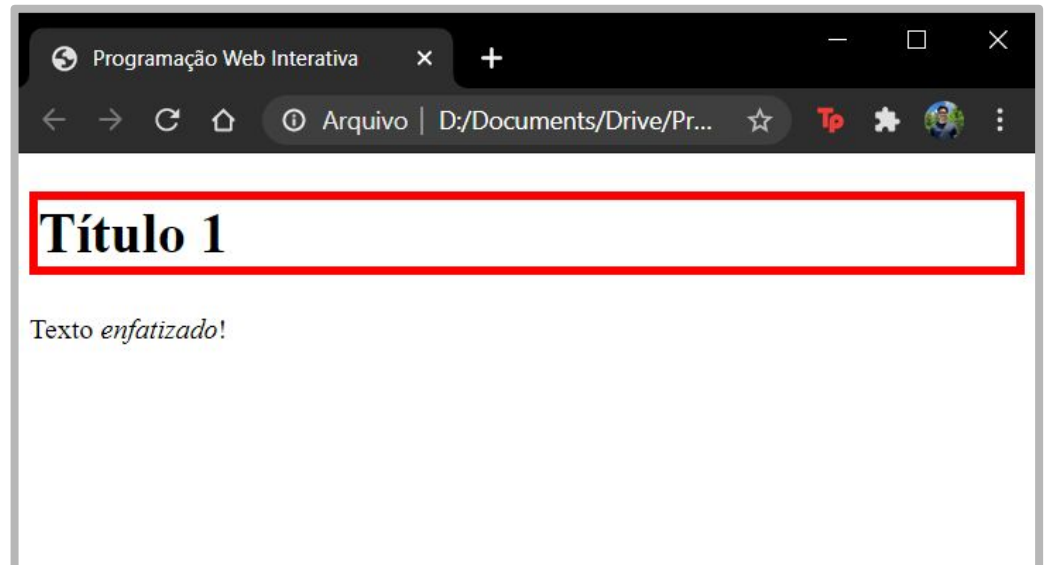
extends the full width of the page

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

```
<h1>Título 1</h1>  
<p>  
  Texto <em>enfaticado</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



Q: What does this look like in the browser?

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



```
<h1>Título 1</h1>  
<p>  
  Texto <em>enfatizado</em>!  
</p>
```

Programação Web Interativa



Arquivo | D:/Documents/Drive/Pr...



Título 1

Texto *ênfatisado!*

Block-level

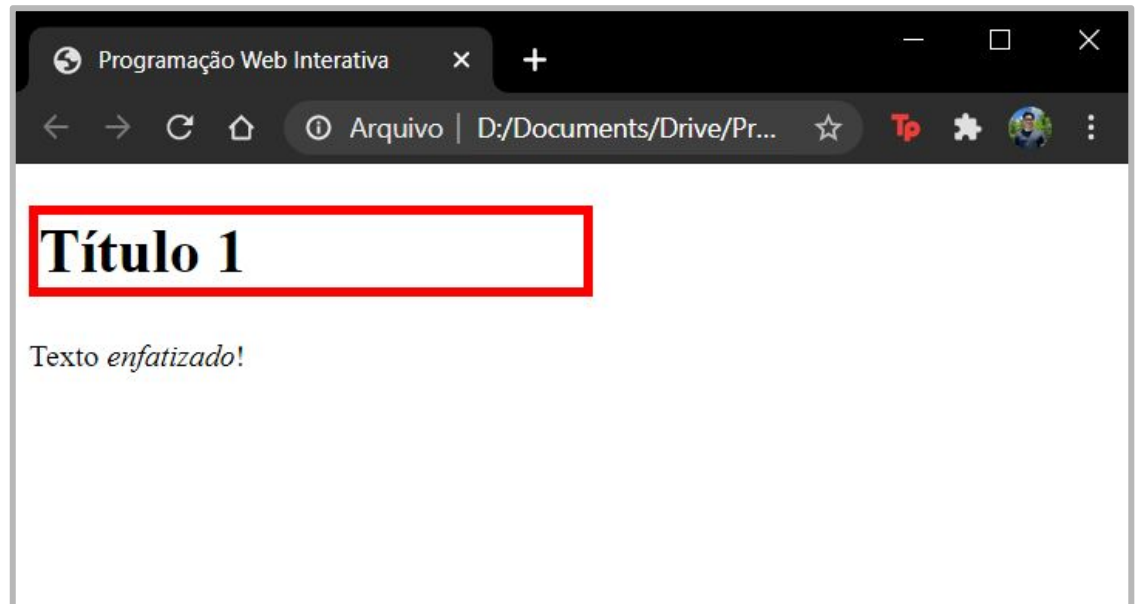
width can be modified

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

```
<h1>Título 1</h1>  
<p>  
  Texto <em>enfaticado</em>!  
</p>
```

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

Block-level elements
still flow top to
bottom

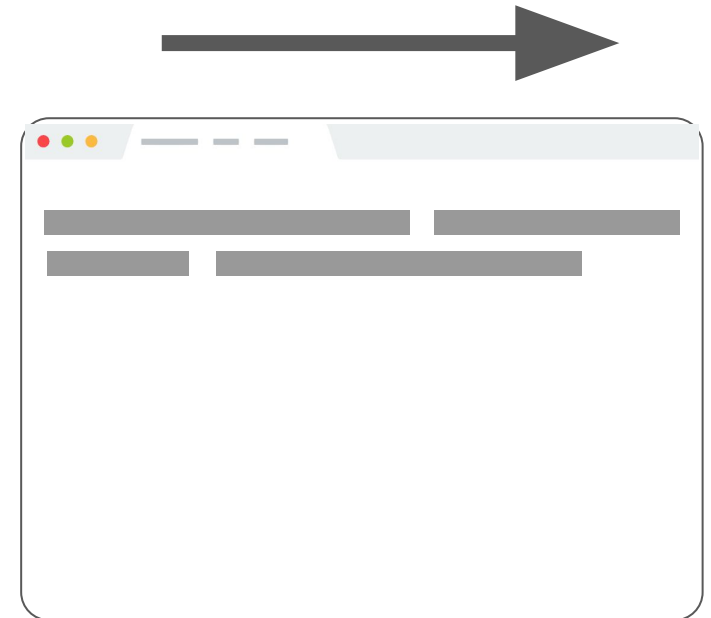


Inline elements

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



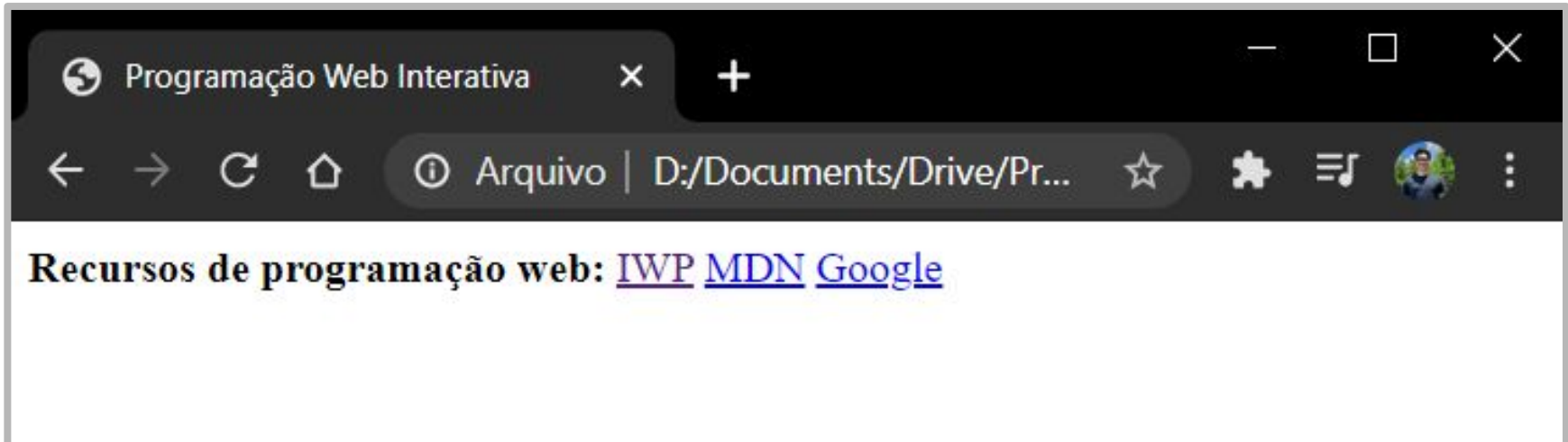
Example: Inline



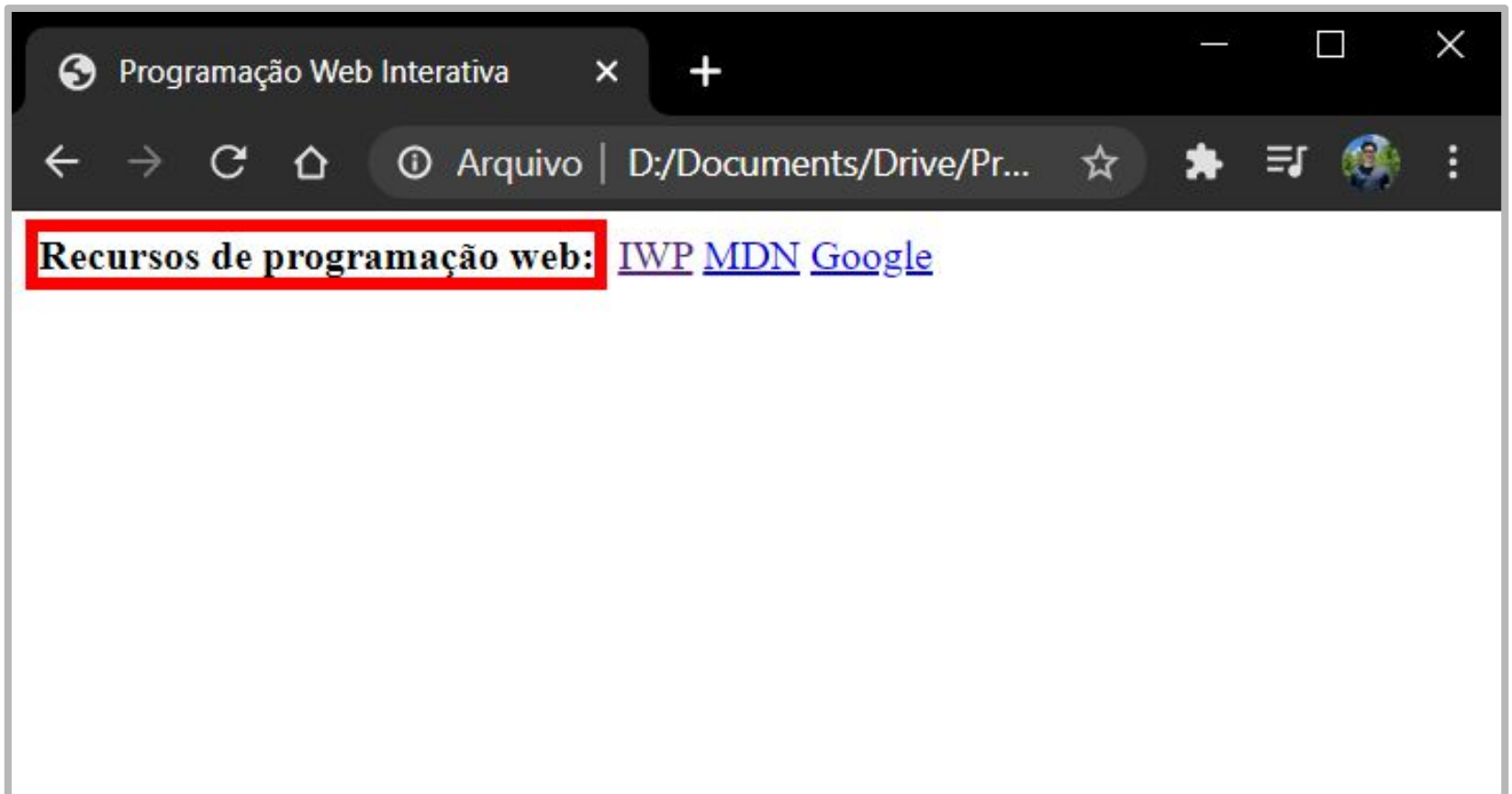
```
<strong>Recursos de programação web:</strong>  
<a href="https://murilocamargos.github.io/iwp">IWP</a>  
<a href="https://developer.mozilla.org/en-US/">MDN</a>  
<a href="http://google.com">Google</a>
```

Q: What does this look like in the browser?

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



```
<strong>Recursos de programação web:</strong>  
<a href="https://murilocamargos.github.io/iwp">IWP</a>  
<a href="https://developer.mozilla.org/en-US/">MDN</a>  
<a href="http://google.com">Google</a>
```

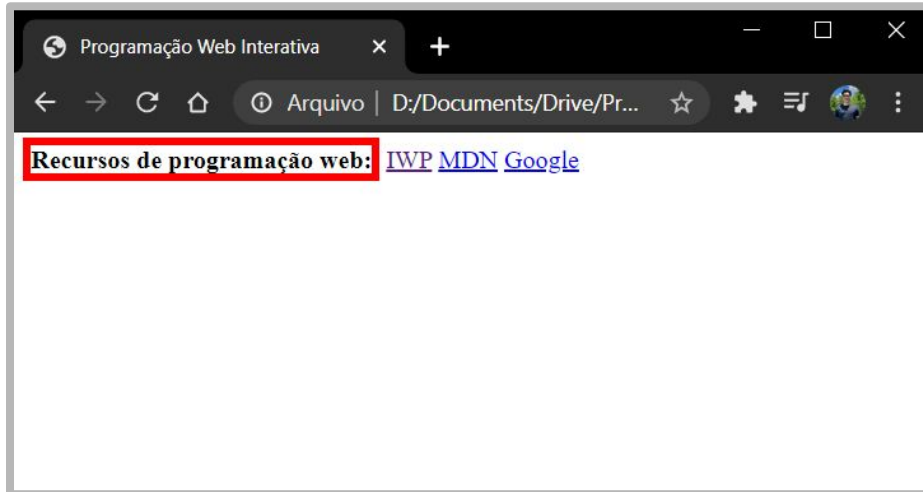


Recursos de programação web: [IWP](#) [MDN](#) [Google](#)

Inline elements ignore width

width cannot be modified

```
<strong>Recursos de programação web:</strong>  
<a href="https://murilocamargos.github.io/iwp">IWP</a>  
<a href="https://developer.mozilla.org/en-US/">MDN</a>  
<a href="http://google.com">Google</a>
```



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

Cannot set **width** on inline element, so it is ignored.

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/a2mAkYQs.jpg> =



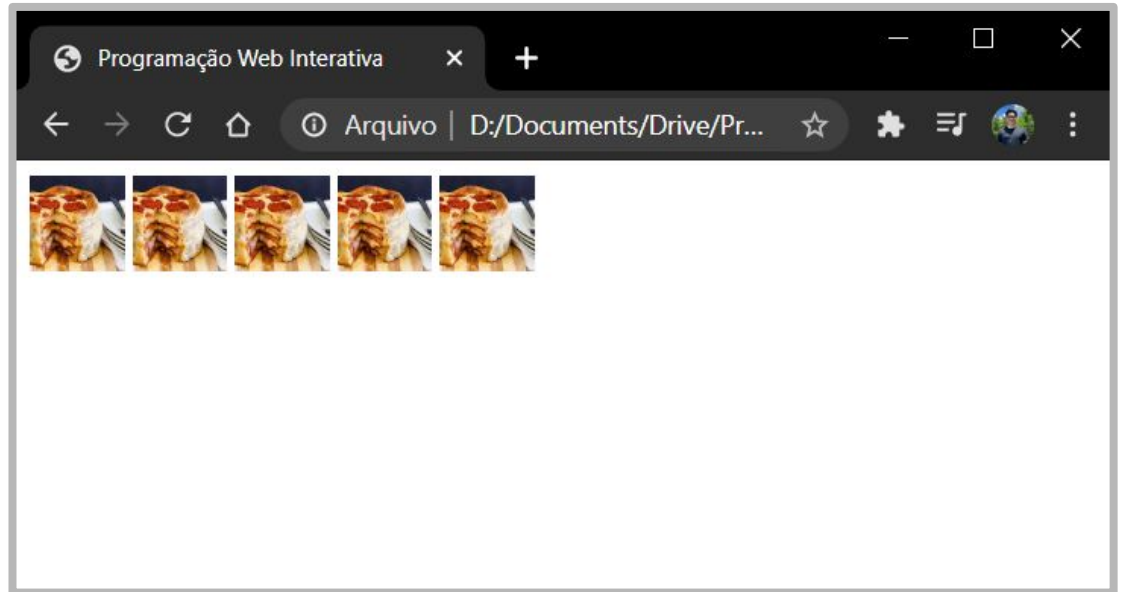


Inline-block

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;  
}
```

```
  
  
  
  

```

Addendum: **paths**

`img src`, `a href`, and `link href` can all take either **relative** or **absolute** paths to the resource:

- `About`
- ``
- `<link rel="stylesheet" href="css/style.css"/>`

If you are unfamiliar with paths, check out the following:

- [Absolute vs relative paths](#)
- [Unix directories and file paths](#)

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>Programação Web Interativa</h1>  
<strong>Avisos:</strong>  
01/03: Começaram nossas aulas!
```

Programação Web Interativa

Avisos: 01/03: Começaram nossas aulas!

```
<h1>Programação Web Interativa</h1>  
<strong>Avisos:</strong><br />  
01/03: Começaram nossas aulas!
```

Programação Web Interativa

Avisos:
01/03: Começaram nossas aulas!

h1 vs strong demystified!

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

```
<h1>Programação Web Interativa</h1>
<strong>Avisos:</strong>
01/03: Começaram nossas aulas!
```

Programação Web Interativa

Avisos: 01/03: Começaram nossas aulas!

```
<h1>Programação Web Interativa</h1>
<strong>Avisos:</strong><br />
01/03: Começaram nossas aulas!
```

Programação Web Interativa

Avisos:
01/03: Começaram nossas aulas!

**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;
}
```

Programação Web Interativa

Avisos:

01/03: Começaram nossas aulas!

01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)

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; }`

```
h1 { /* works */
  text-align: center;
}
a { /* works :D */
  text-align: center;
  display: block;
}
```

Programação Web Interativa

Avisos:

01/03: Começaram nossas aulas!

01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)

HTML

```
<p>
  <a href="url">
    Ver Ementa
  </a>
</p>
```

Programação Web Interativa

Avisos:

01/03: Começaram nossas aulas!

01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)

CSS

```
h1 { /* works */
  text-align: center;
}
p { /* works :D */
  text-align: center;
}
```

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;  
}
```

Programação Web Interativa

Avisos:

01/03: Começaram nossas aulas!

01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)

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 #ff69b4;  
  background-color: #d8bfd8;  
}
```

Programação Web Interativa

Avisos:

01/03: Começaram nossas aulas!

01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)

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;  
}
```

Programação Web Interativa

Avisos:

01/03: Começaram nossas aulas!

01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)

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;  
}
```

Programação Web Interativa

Avisos:

01/03: Começaram nossas aulas!

01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)

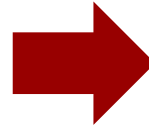
Highlight mystery

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

Programação Web Interativa

Avisos:
01/03: Começaram nossas aulas!
01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)



Programação Web Interativa

Avisos:
01/03: Começaram nossas aulas!
01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)

How do we fix this?

Highlight: demystified!

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

```
<strong>Avisos:</strong><br/>  
01/03: Começaram nossas aulas!<br/>  
01/03: A tarefa 0 está disponível.  
<em>Para Terça</em>.
```

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

Programação Web Interativa

Avisos:
01/03: Começaram nossas aulas!
01/03: A tarefa 0 está disponível. **Para Terça.**

[Ver Ementa](#)

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**:

```
<strong>Avisos:</strong><br/>  
01/03: Começaram nossas aulas!<br/>  
01/03: A tarefa 0 está disponível.  
<em>Para Terça</em>
```

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

Programação Web Interativa

Avisos:
01/03: Começaram nossas aulas!
01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)

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:

```
<strong>Avisos:</strong><br/>  
01/03: Começaram nossas aulas!<br/>  
01/03: A tarefa 0 está disponível.  
<span>Para Terça</span>.
```

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

Programação Web Interativa

Avisos:

01/03: Começaram nossas aulas!
01/03: A tarefa 0 está disponível. Para Terça.

[Ver Ementa](#)

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?

Programação Web Interativa

Avisos:

01/03: **Começaram** nossas aulas!

01/03: A tarefa 0 está disponível. **Para Terça.**

[Ver Ementa](#)

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 Tue.<br/>  
<em class="hw">HW1</em> goes out Thu.<br/>  
<em>All homework due at 11:59pm.</em>
```

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

Homework

HW0 is due Tue.

HW1 goes out Thu.

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

element.className

Syntax	Example	Example described
<i>element.className</i>	<code>p.abc</code>	<code><p></code> elements with abc class

HTML

```
1 <h1 class="hw">Homework 0</h1>
2 <p class="hw">Due Tue</p>
3 <p class="hw">Late cutoff Thu</p>
4 <h1>Lectures</h1>
5 <p>Mar 2: Syllabus</p>
6 <p>Mar 4: HTML+CSS</p>
```

CSS

```
1 p.hw {
2   color: green;
3 }
```

Homework 0

Due Tue

Late cutoff Thu

Lectures

Mar 2: Syllabus

Mar 4: HTML+CSS

Descendent selector

Syntax	Example	Example described
<i>selector selector</i>	<code>div strong</code>	<code></code> elements that are descendants of a <code><div></code>

HTML

```
1 <div class="hw">
2   <h1>Homework 0</h1>
3   <p>Due Tue</p>
4   <p>Late cutoff Thu</p>
5 </div>
```

CSS

```
1 .hw p {
2   color: green;
3 }
```

Homework 0

Due Tue

Late cutoff Thu

Lectures

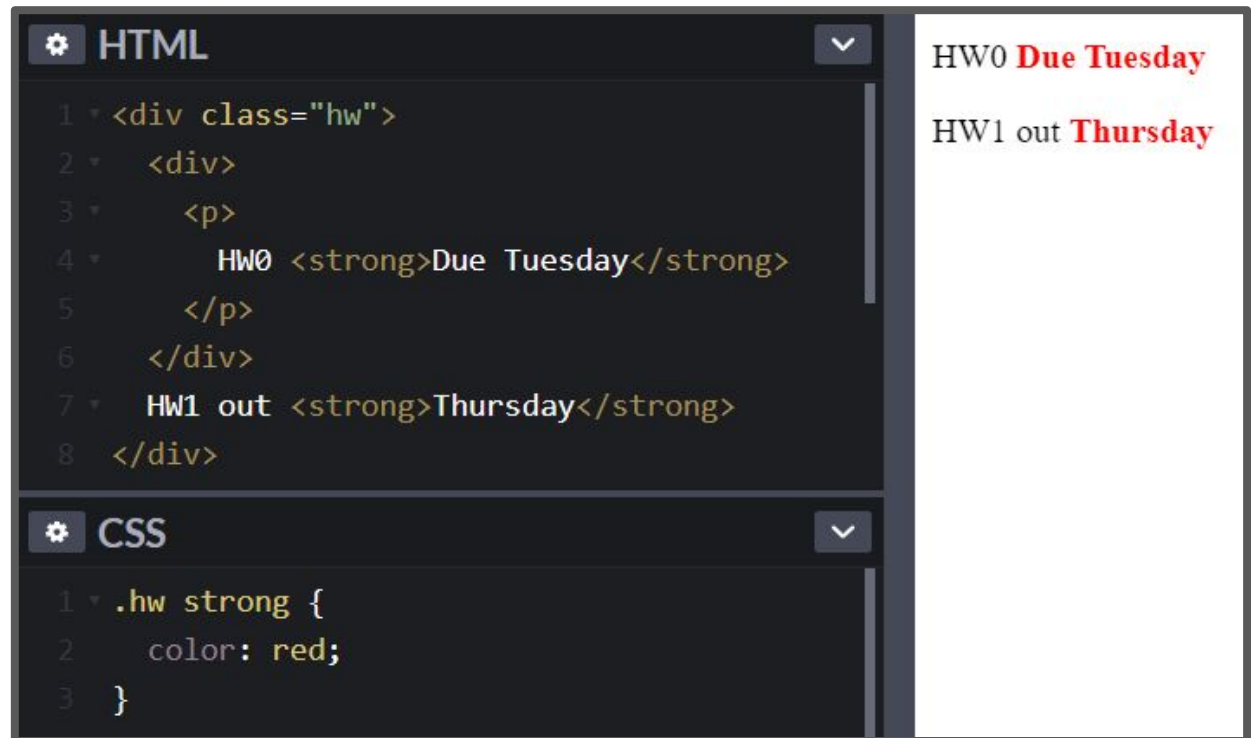
Mar 2: Syllabus

Mar 4: HTML+CSS

Descendent selector

Syntax	Example	Example described
<i>selector selector</i>	div 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.



The screenshot shows a code editor with two panels: HTML and CSS. The HTML panel contains the following code:

```
1 <div class="hw">
2   <div>
3     <p>
4       HW0 <strong>Due Tuesday</strong>
5     </p>
6   </div>
7   HW1 out <strong>Thursday</strong>
8 </div>
```

The CSS panel contains the following code:

```
1 .hw strong {
2   color: red;
3 }
```

The rendered output on the right shows:

HW0 **Due Tuesday**
HW1 out **Thursday**

Descendent selector

Syntax	Example	Example described
<i>selector selector</i>	<code>div strong</code>	<code></code> elements that are descendants of a <code><div></code>

Discouraged:

```
<h1 class="hw">Homework 0</h1>
<p class="hw">Due Tue</p>
<p class="hw">Late cutoff Thu</p>
```

vs

Preferred:

```
<div class="hw">
  <h1>Homework 0</h1>
  <p>Due Tue</p>
  <p>Late cutoff Thu</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
<i>selector, selector</i>	h2, div	<h2> elements and <div>s

The image shows a code editor with two panels on the left and a rendered preview on the right. The top panel is labeled 'HTML' and contains the following code:

```
1 <h1>Course Info</h1>
2 <h2>Lectures</h2>
3 <p>Tue-Thu 14h00-15h30</p>
4 <h2>Honor Code</h2>
5 <p>Do the right thing</p>
```

The bottom panel is labeled 'CSS' and contains the following code:

```
1 h1, h2 {
2   font-family: Arial;
3 }
```

The rendered preview on the right shows the following output:

Course Info

Lectures

Tue-Thu 14h00-15h30

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	 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:

- `` tags that are children of `` 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>
```

 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:

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

```
gear 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?



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](#)

```
<!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!

Before we move on:
A few style notes

Why not `<div>` everywhere?

Technically, you can define your entire web page using `<div>` and the `class` attribute.

- Is this a good idea?
- Why does HTML have `ids` when you have `classes`?
- Why does HTML have `<p>`, `<h1>`, ``, etc. when you have `<div>`, ``, `class`, and `id`?

The box model:
Next time!