Interactive Web Programming

1st semester of 2021

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

Heavily based on **Victoria Kirst** slides

Today's schedule

Today

- More flexbox
- `vh / vw / box-sizing`
- `position`
- Mobile layouts
- Random helpful CSS
- CSS wrap-up

Next Tuesday:

- Intro to JavaScript

Simplicity above all else

Always prefer **simplicity.**

Other tips:

- **Separation of concerns**: HTML should contain content NOT style, CSS should contain style NOT content
- **Descriptive HTML tags**: Make your HTML more readable by using e.g. `<header>` instead of `<div>` when appropriate
- **Reduce redundancy**: Try grouping styles, using descendant selectors to reduce redundancy (see past slides for details)

Font-related CSS review

Name	Description
`font-family`	Font face (mdn)
`color`	Font color (and always font color) (mdn)
`font-size`	Font size (mdn)
`line-height`	Line height (mdn)
`text-align`	Alignment of text (mdn)

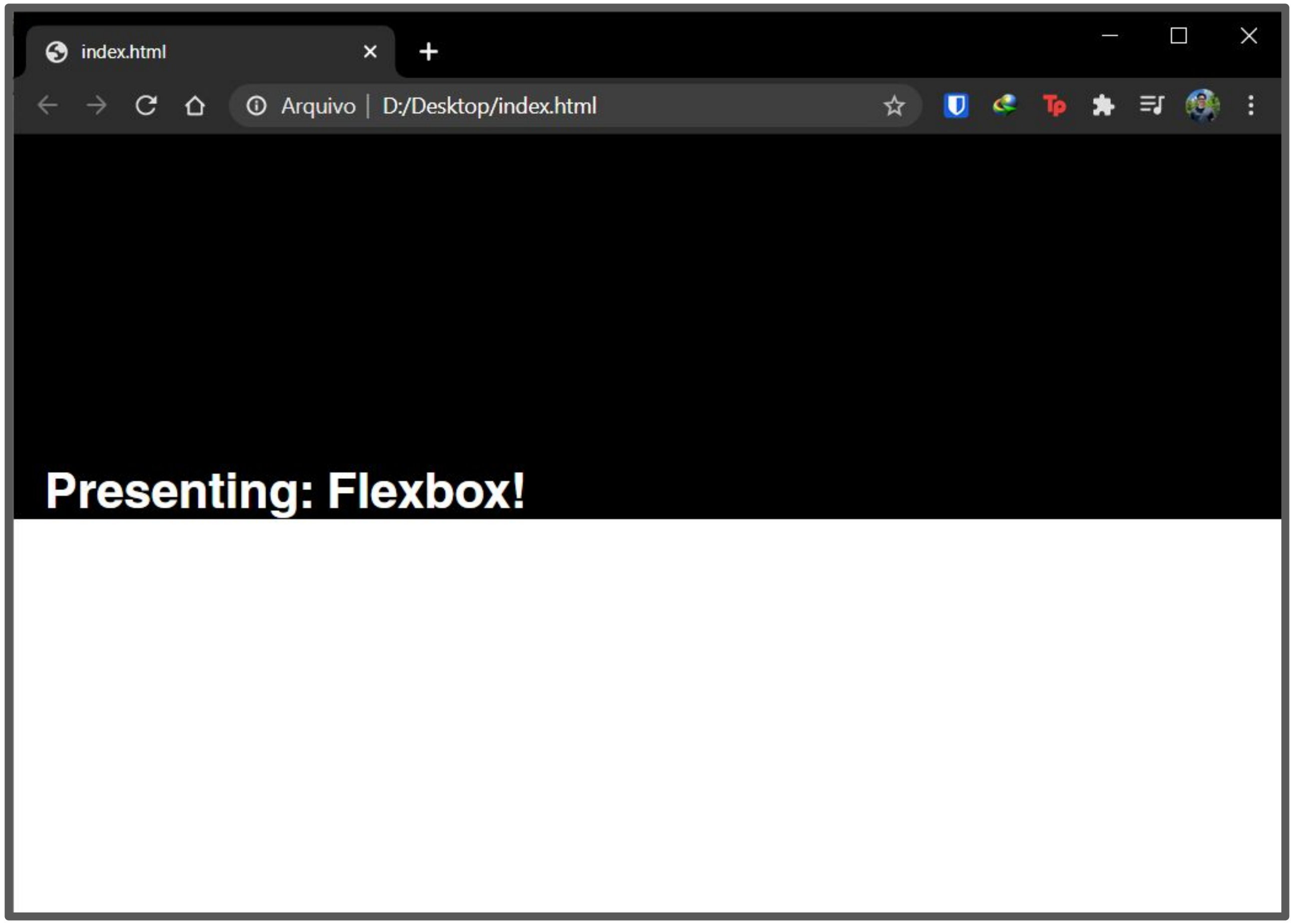
More font-related CSS

Name	Description
`text-decoration`	Can set `underline`, `line-through` (strikethrough) or none (e.g. to unset underline on hyperlinks) (mdn)
`text-transform`	Can change font **case**, i.e. `uppercase`, `lowercase`, `capitalize`, none (mdn)
`font-style`	Can set to `italic` or `normal` (e.g. to unset italic on `<em>`) (mdn)
`font-weight`	Can set to `bold` or `normal` (e.g. to unset bold on h1 - h6) (mdn)
`letter-spacing`	Controls the space between letters (mdn)

Flexbox

Review: Flexbox

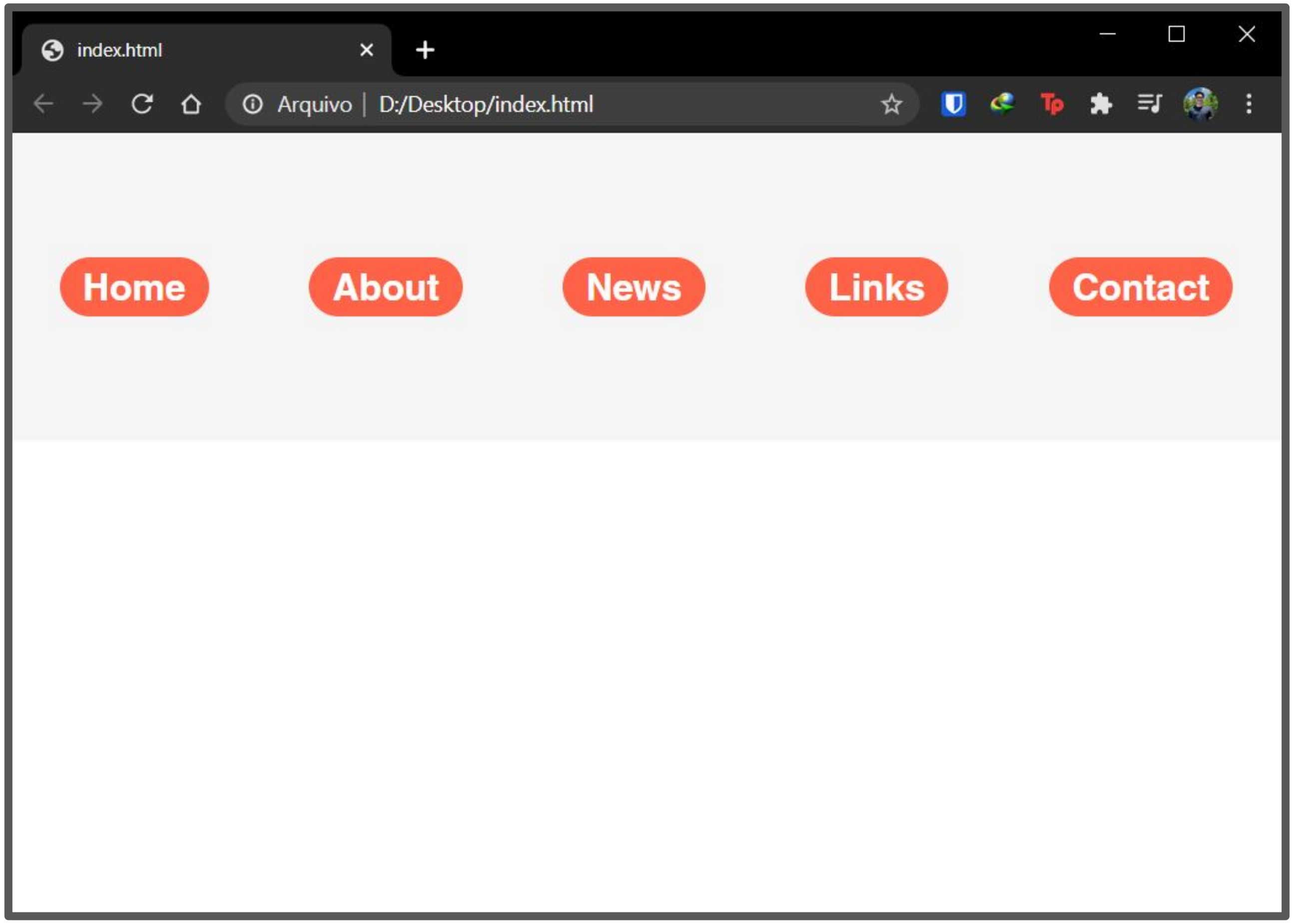
How do we create this look? (Codepen)



Presenting: Flexbox!

Review: Flexbox

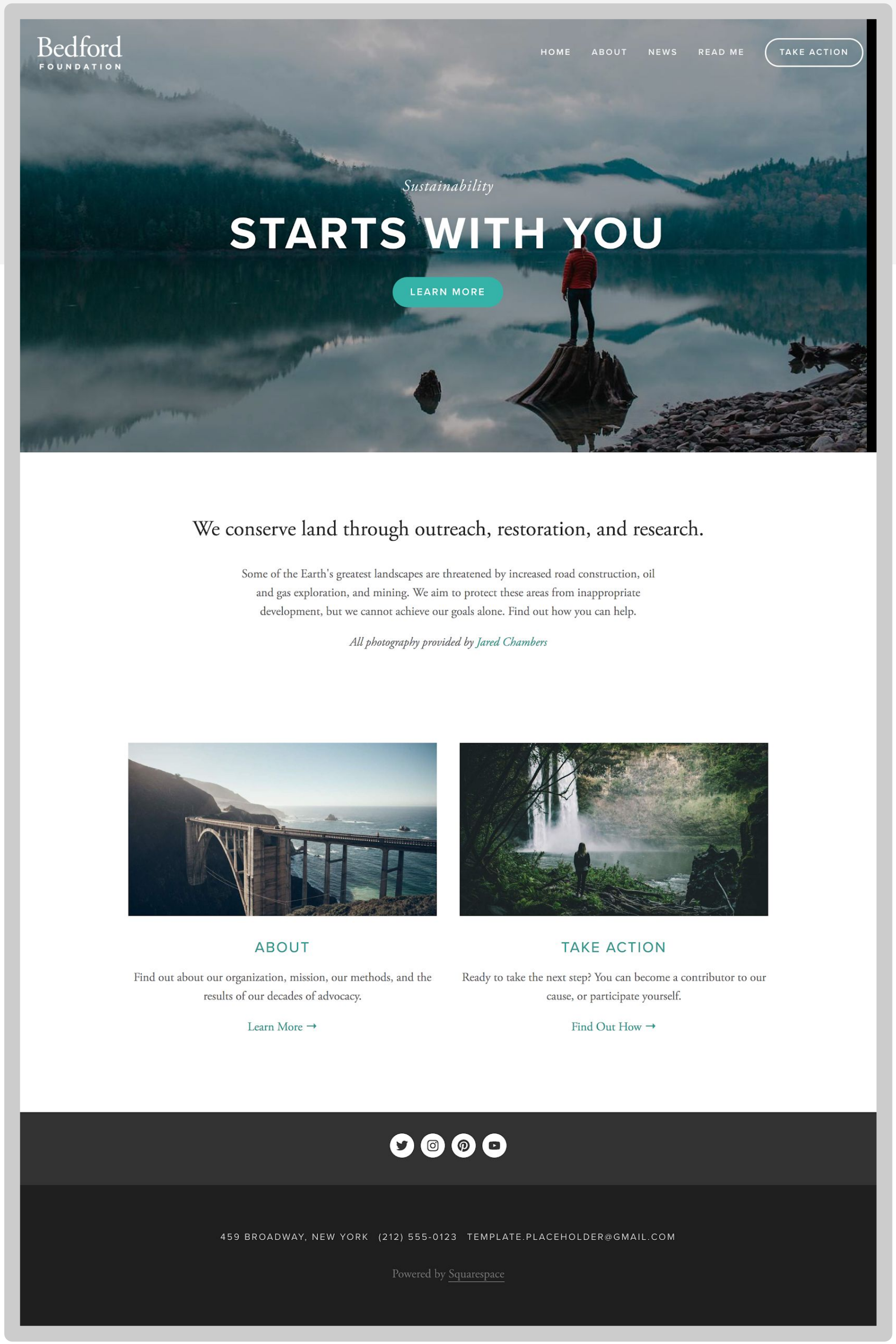
How do we create this look? (Codepen)



Continuing where we left off!

Goal

We were trying to create a layout that looks sort of like this:



Bedford
FOUNDATION

HOME ABOUT NEWS READ ME TAKE ACTION

Sustainability

STARTS WITH YOU

LEARN MORE

We conserve land through outreach, restoration, and research.

Some of the Earth's greatest landscapes are threatened by increased road construction, oil and gas exploration, and mining. We aim to protect these areas from inappropriate development, but we cannot achieve our goals alone. Find out how you can help.

All photography provided by Jared Chambers

ABOUT

Find out about our organization, mission, our methods, and the results of our decades of advocacy.

Learn More →

TAKE ACTION

Ready to take the next step? You can become a contributor to our cause, or participate yourself.

Find Out How →

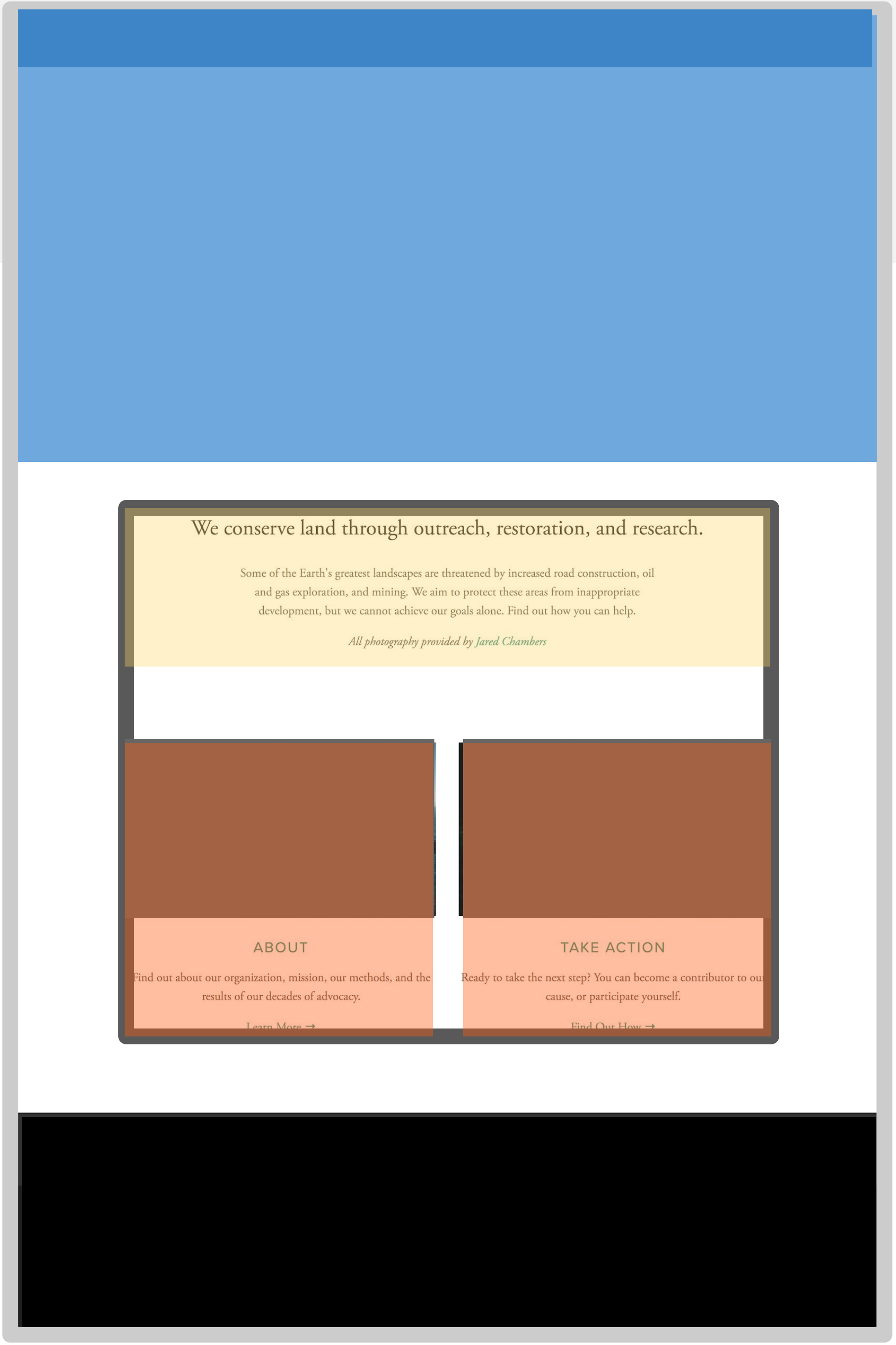
459 BROADWAY, NEW YORK (212) 555-0123 TEMPLATE.PLACEHOLDER@GMAIL.COM

Powered by Squarespace

Status

We broke up the layout into a bunch of colored boxes:

And we got kind of stuck trying to position the orange boxes.



We conserve land through outreach, restoration, and research.

Some of the Earth's greatest landscapes are threatened by increased road construction, oil and gas exploration, and mining. We aim to protect these areas from inappropriate development, but we cannot achieve our goals alone. Find out how you can help.

All photography provided by Jared Chambers

ABOUT

Find out about our organization, mission, our methods, and the results of our decades of advocacy.

Learn More →

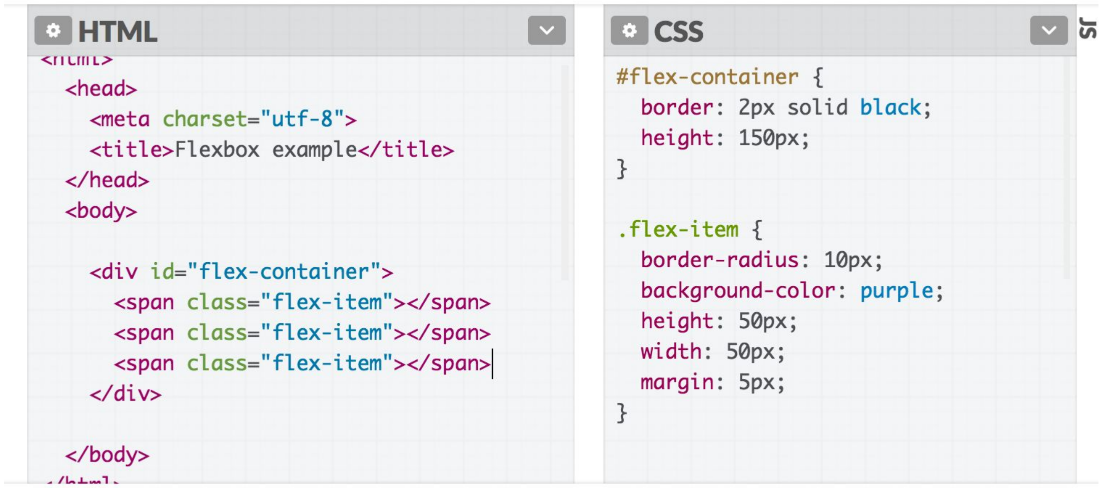
TAKE ACTION

Ready to take the next step? You can become a contributor to our cause, or participate yourself.

Find Out How →

Recall: block layouts

If `#flex-container` was **not** `display: flex`:

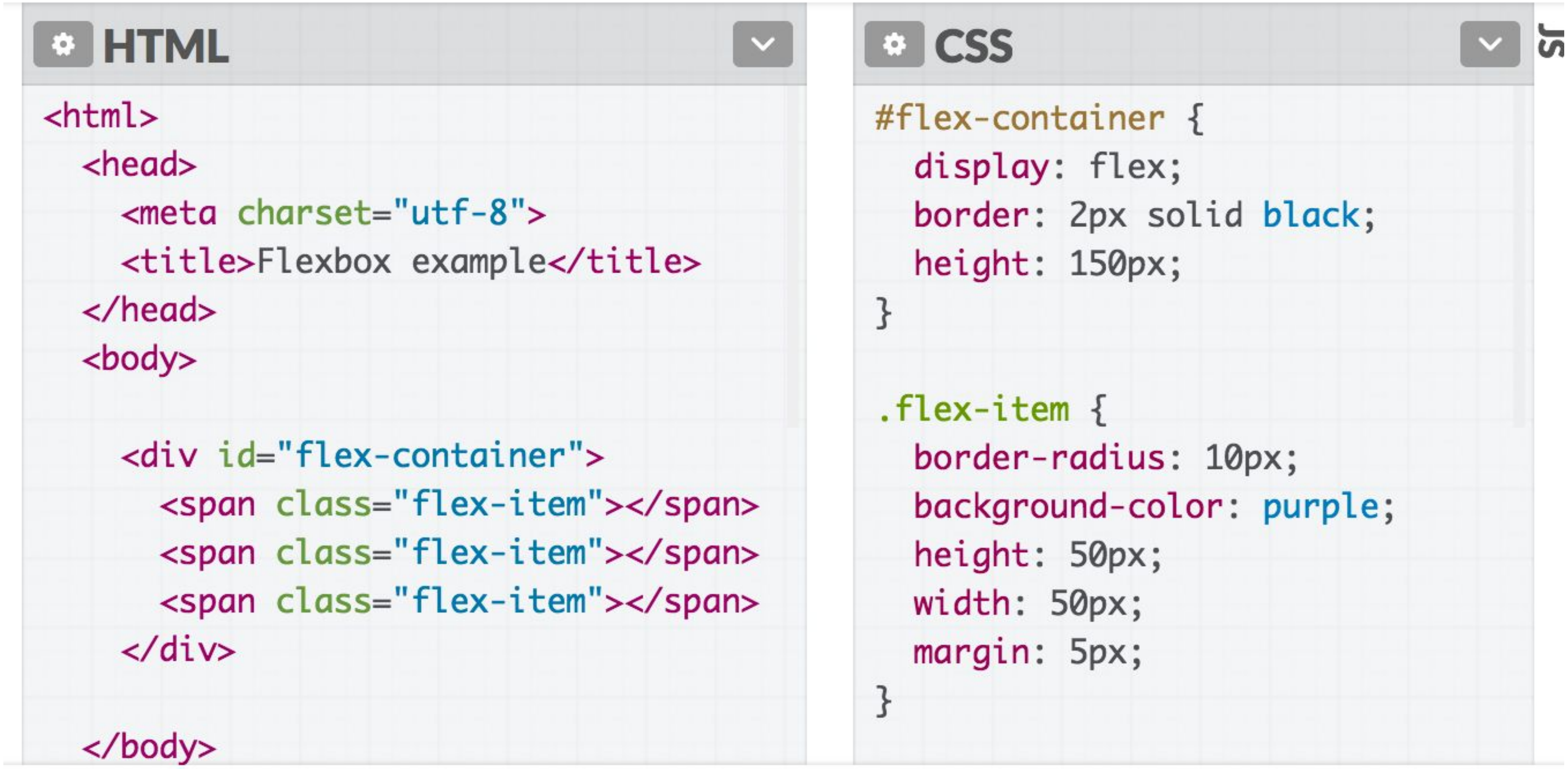


```html
<html>
  <head>
    <meta charset="utf-8">
    <title>Flexbox example</title>
  </head>
  <body>

    <div id="flex-container">
      <span class="flex-item"></span>
      <span class="flex-item"></span>
      <span class="flex-item"></span>
    </div>

  </body>
</html>
```

```css
#flex-container {
  border: 2px solid black;
  height: 150px;
}

.flex-item {
  border-radius: 10px;
  background-color: purple;
  height: 50px;
  width: 50px;
  margin: 5px;
}
```

Then the span `flex-items` would not show up because span elements are inline, which don't have a height and width

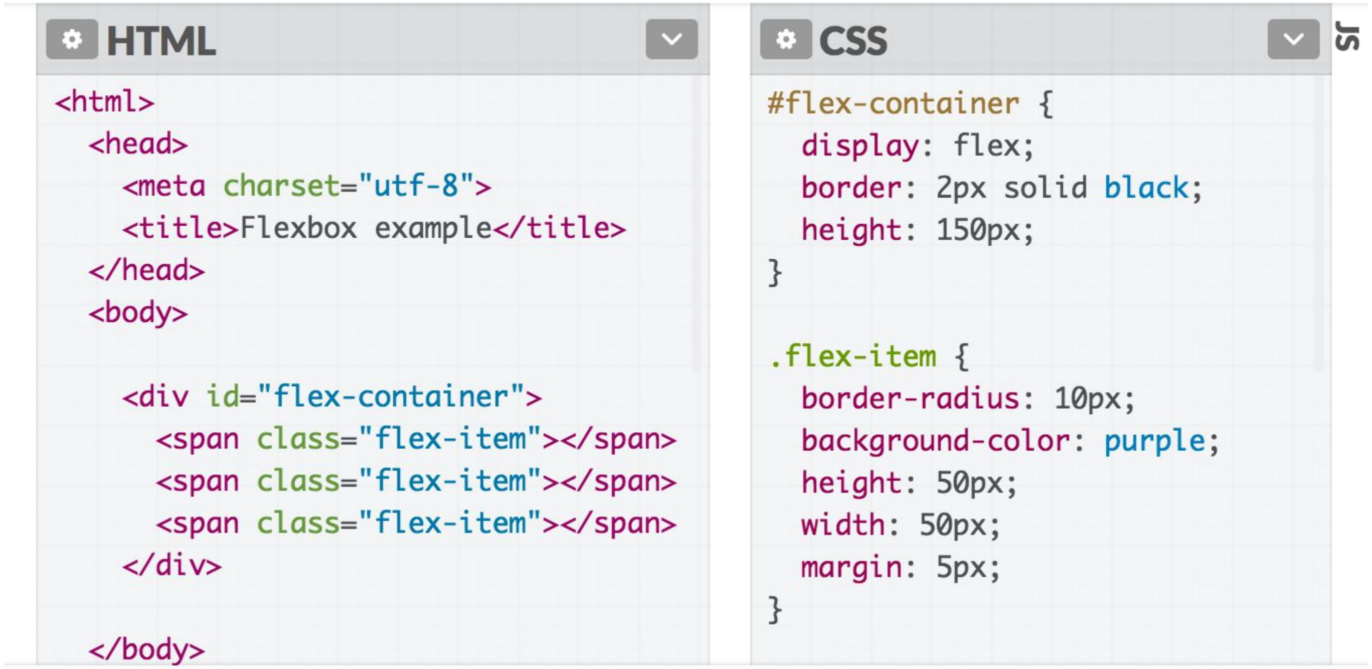
What happens if the flex item is an inline element?



```html
<html>
  <head>
    <meta charset="utf-8">
    <title>Flexbox example</title>
  </head>
  <body>

    <div id="flex-container">
      <span class="flex-item"></span>
      <span class="flex-item"></span>
      <span class="flex-item"></span>
    </div>

  </body>
```

```css
#flex-container {
  display: flex;
  border: 2px solid black;
  height: 150px;
}

.flex-item {
  border-radius: 10px;
  background-color: purple;
  height: 50px;
  width: 50px;
  margin: 5px;
}
```

???

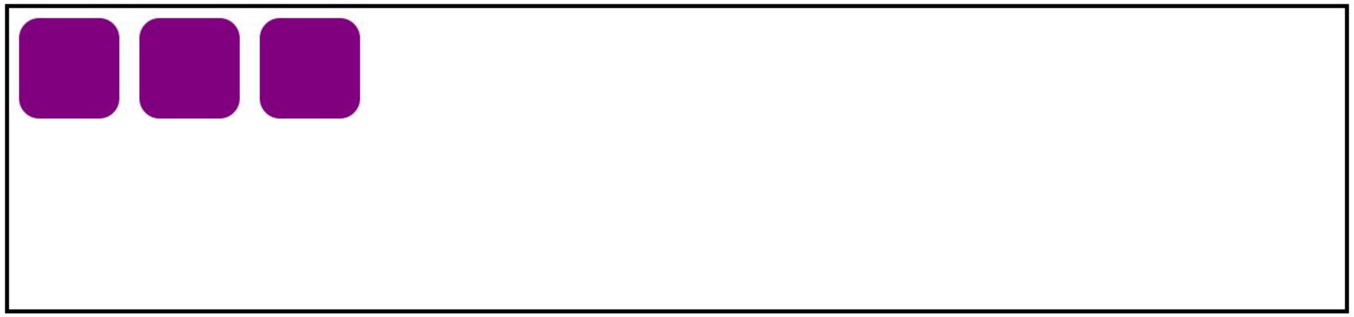


HTML

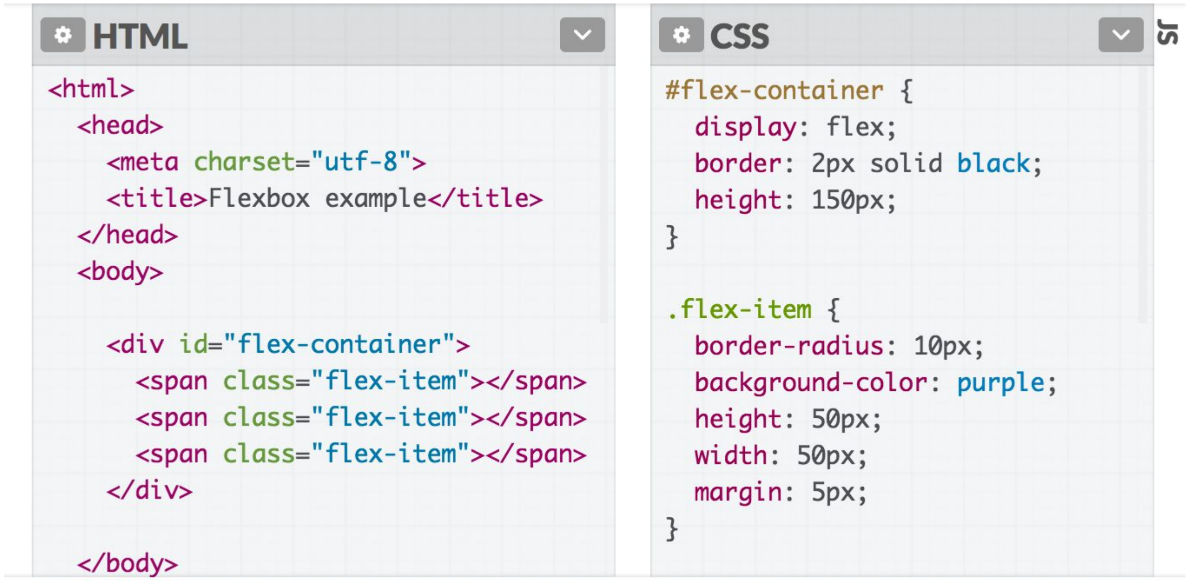
```
<html>
  <head>
    <meta charset="utf-8">
    <title>Flexbox example</title>
  </head>
  <body>

    <div id="flex-container">
      <span class="flex-item"></span>
      <span class="flex-item"></span>
      <span class="flex-item"></span>
    </div>

  </body>
```

CSS

```
#flex-container {
  display: flex;
  border: 2px solid black;
  height: 150px;
}

.flex-item {
  border-radius: 10px;
  background-color: purple;
  height: 50px;
  width: 50px;
  margin: 5px;
}
```

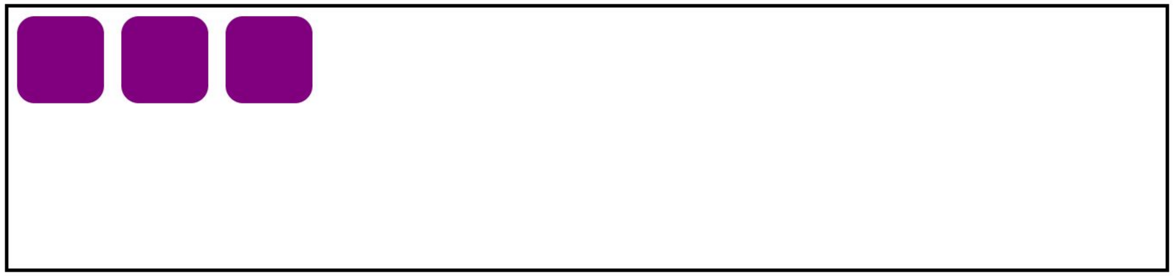


Flex layouts



```html
<html>
  <head>
    <meta charset="utf-8">
    <title>Flexbox example</title>
  </head>
  <body>

    <div id="flex-container">
      <span class="flex-item"></span>
      <span class="flex-item"></span>
      <span class="flex-item"></span>
    </div>

  </body>
```

```css
#flex-container {
  display: flex;
  border: 2px solid black;
  height: 150px;
}

.flex-item {
  border-radius: 10px;
  background-color: purple;
  height: 50px;
  width: 50px;
  margin: 5px;
}
```



Why does this change when `display: flex`?

Why do inline elements suddenly seem to have height and width?

Flex: A different rendering mode

- When you set a container to `display: flex`, the direct children in that container are **flex items** and follow a new set of rules.
- **Flex items are not block or inline;** they have different rules for their height, width, and layout.
  - The *contents* of a flex item follow the usual block/inline rules, relative to the flex item's boundary.
- The **height** and **width** of flex items are… complicated.

Flex item sizing

Flex basis

Flex items have an initial width*, which, by default is either:

- The content width, or
- The explicitly set **`width`** property of the element, or
- The explicitly set **`flex-basis`** property of the element

This initial width* of the flex item is called the **flex basis**.

*width in the case of rows; height in the case of columns

Flex basis

Flex items have an initial width*, which, by default is either:

- The content width, or
- The explicitly set **`width`** property of the element, or
- The explicitly set **`flex-basis`** property of the element

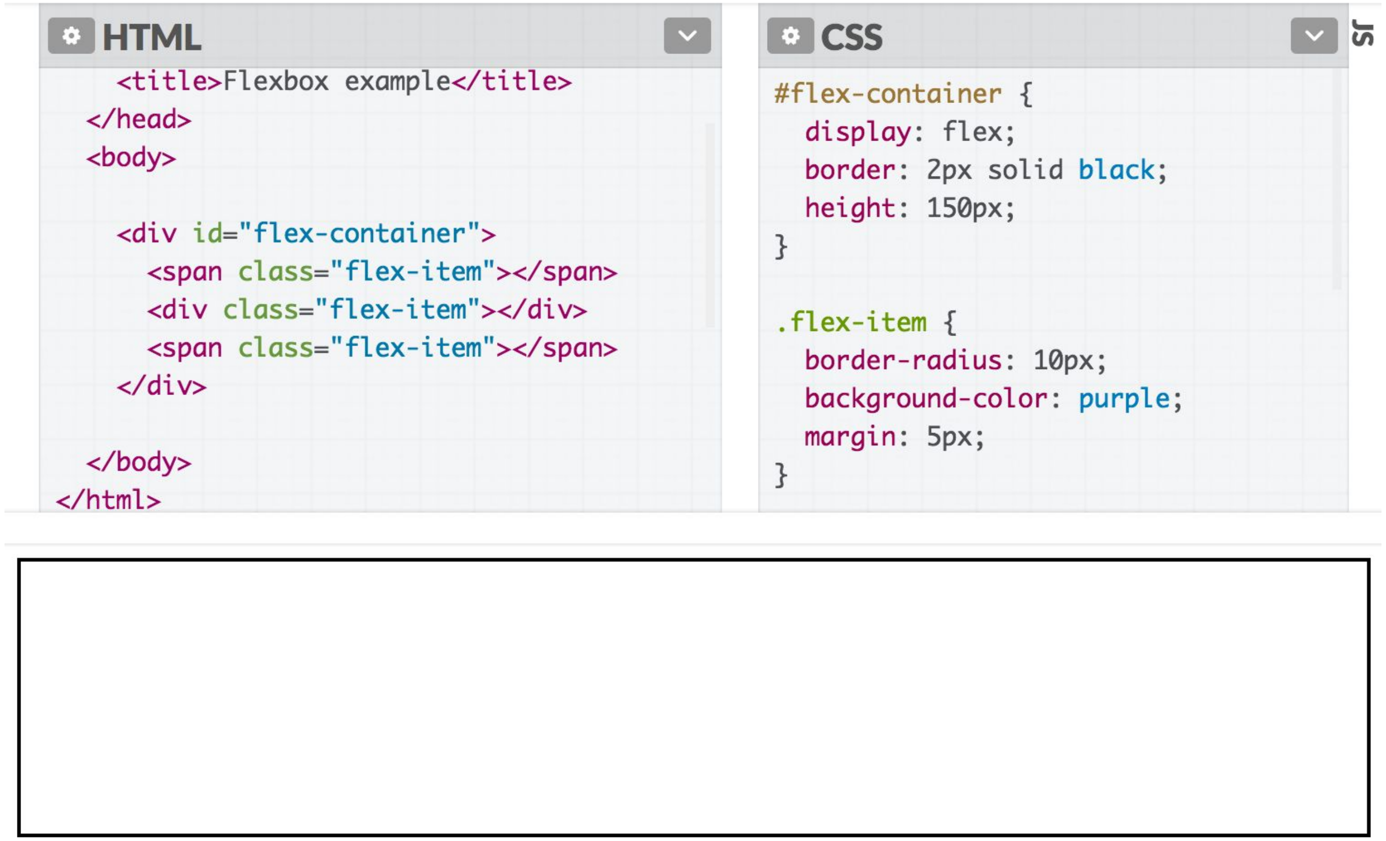
This initial width* of the flex item is called the **flex basis**.

The explicit width* of a flex item is respected *for all flex items*, regardless of whether the flex item is inline, block, or inline-block.

*width in the case of rows; height in the case of columns

Flex basis

If we unset the `height` and `width`, our flex items disappears, because the **flex basis** is now the content size, which is empty:



```html
    <title>Flexbox example</title>
  </head>
  <body>

    <div id="flex-container">
      <span class="flex-item"></span>
      <div class="flex-item"></div>
      <span class="flex-item"></span>
    </div>

  </body>
</html>
```

```css
#flex-container {
  display: flex;
  border: 2px solid black;
  height: 150px;
}

.flex-item {
  border-radius: 10px;
  background-color: purple;
  margin: 5px;
}
```

`flex-shrink`

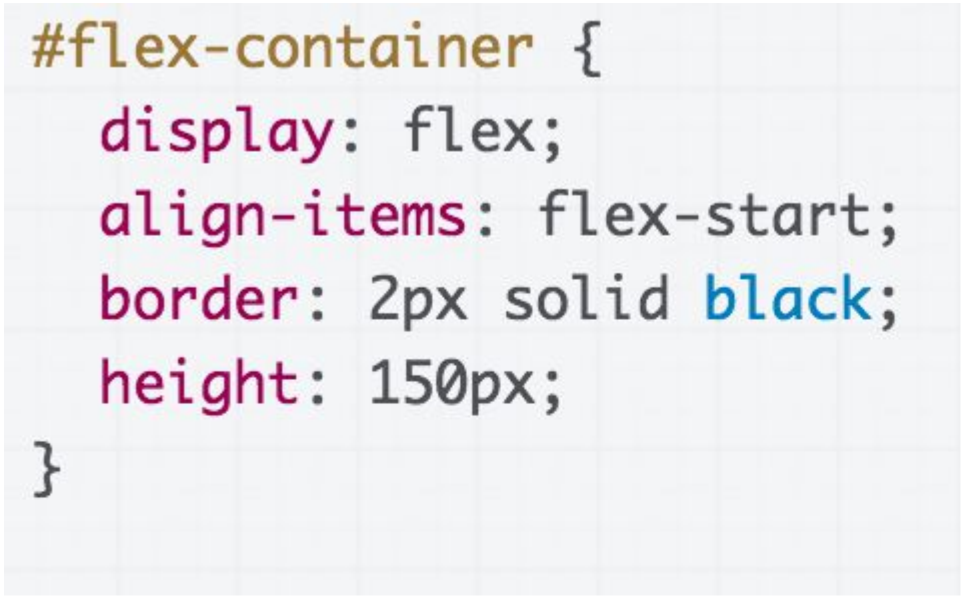
The width* of the flex item can automatically shrink **smaller than the flex basis** via the `flex-shrink` property:

`flex-shrink`:

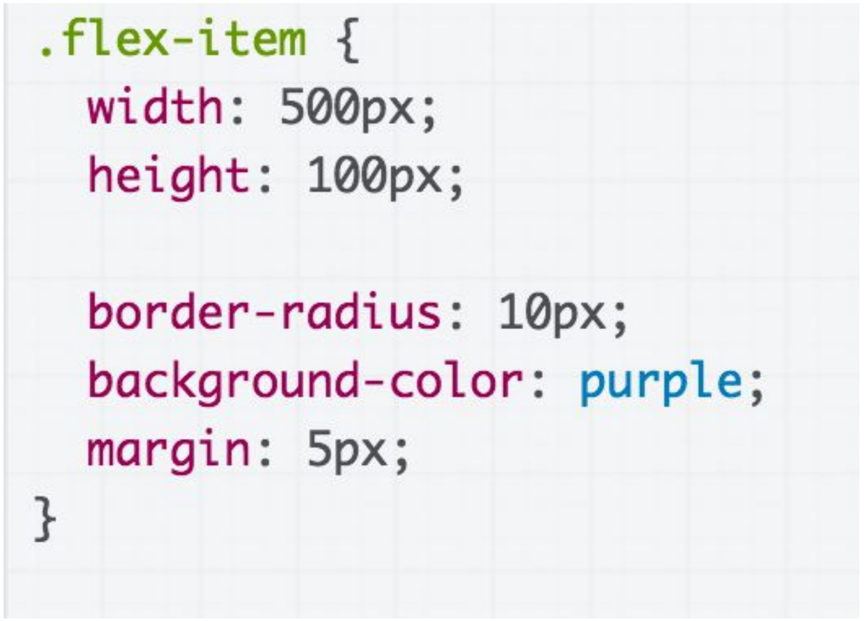
- If set to 1, the flex item shrinks itself as small as it can in the space available.
- If set to 0, the flex item does not shrink.

Flex items have `flex-shrink: 1` by default.

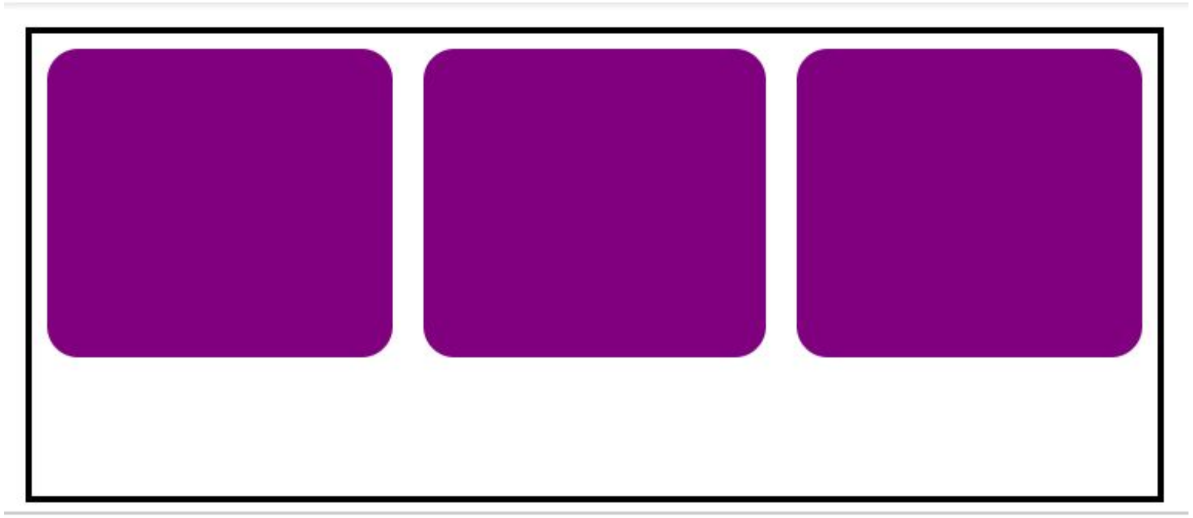
*width in the case of rows; height in the case of columns



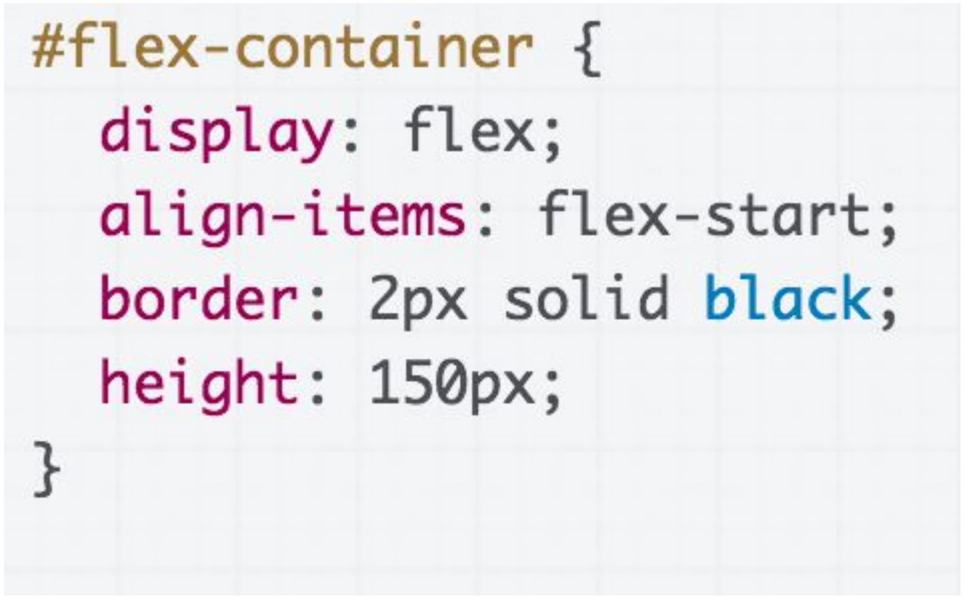
```
#flex-container {
  display: flex;
  align-items: flex-start;
  border: 2px solid black;
  height: 150px;
}
```



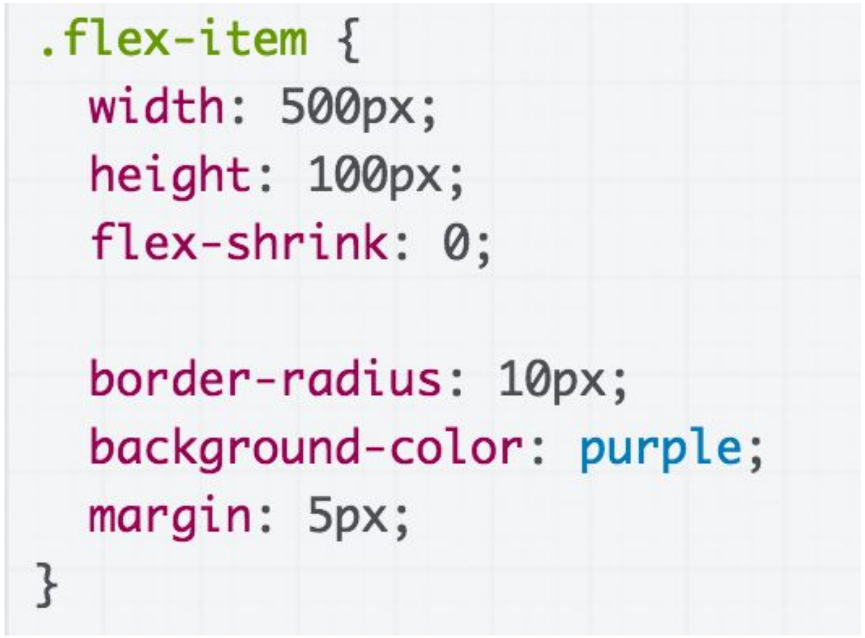
```
.flex-item {
  width: 500px;
  height: 100px;

  border-radius: 10px;
  background-color: purple;
  margin: 5px;
}
```



The flex items' widths all shrink to fit within the container.

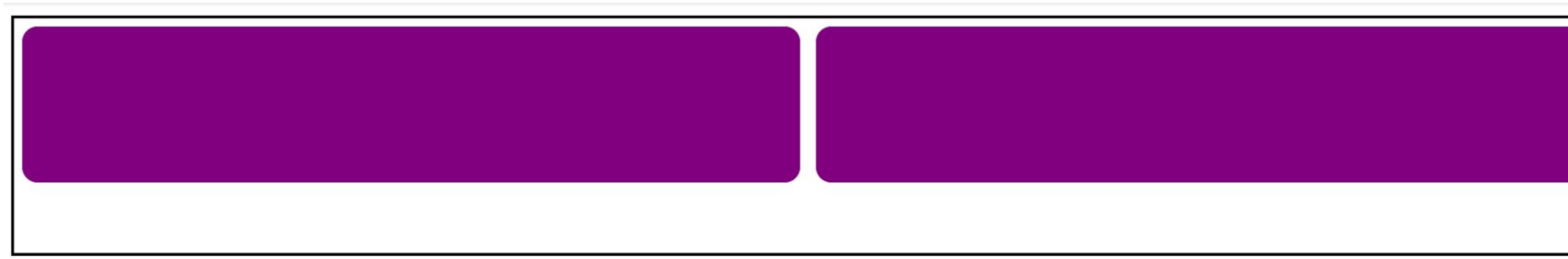


```css
#flex-container {
  display: flex;
  align-items: flex-start;
  border: 2px solid black;
  height: 150px;
}
```



```css
.flex-item {
  width: 500px;
  height: 100px;
  flex-shrink: 0;

  border-radius: 10px;
  background-color: purple;
  margin: 5px;
}
```

Setting `flex-shrink: 0;` undoes the shrinking behavior, and the flex items do not shrink in any circumstance:



`flex-grow`

The width* of the flex item can automatically **grow larger than the flex basis** via the `flex-grow` property:

`flex-grow`:

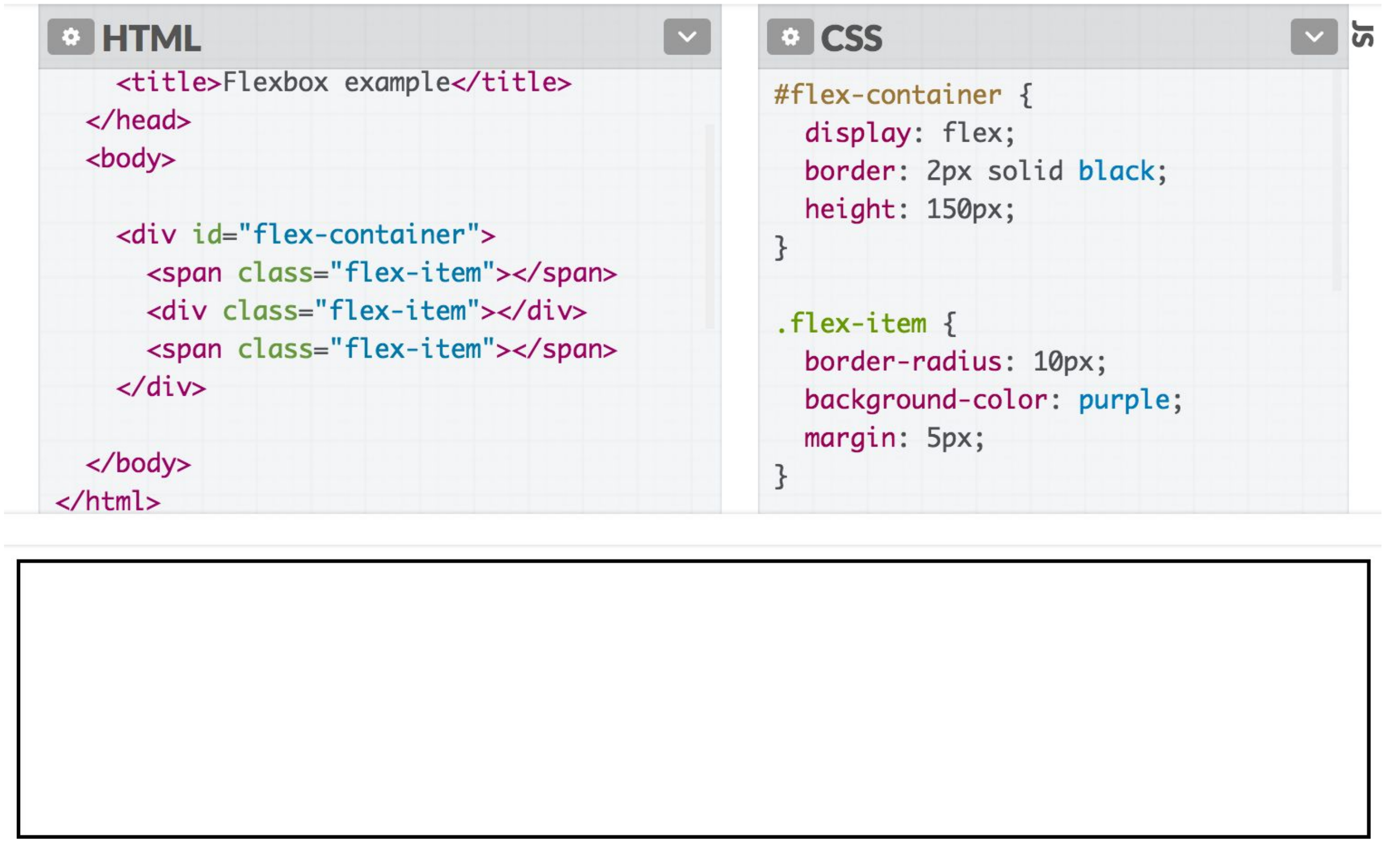
- If set to 1, the flex item grows itself as large as it can in the space remaining.
- If set to 0, the flex-item does not grow.

Flex items have `flex-grow: 0` by default.

*width in the case of rows; height in the case of columns

`flex-grow` example

Let's unset the height and width of our flex items again:

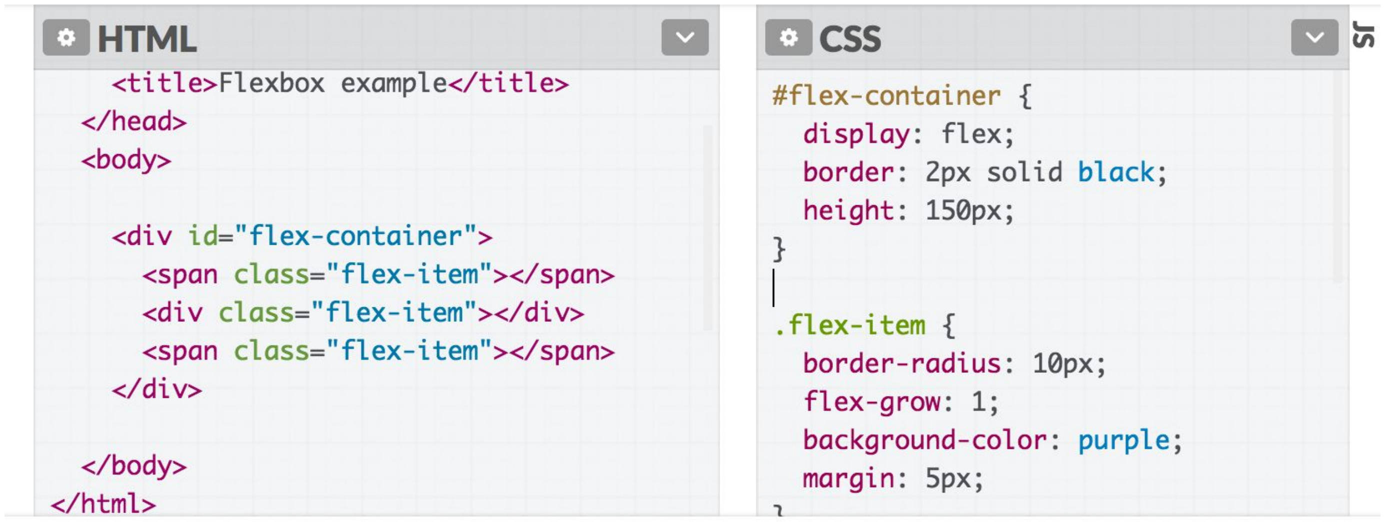


```html
    <title>Flexbox example</title>
  </head>
  <body>

    <div id="flex-container">
      <span class="flex-item"></span>
      <div class="flex-item"></div>
      <span class="flex-item"></span>
    </div>

  </body>
</html>
```

```css
#flex-container {
  display: flex;
  border: 2px solid black;
  height: 150px;
}

.flex-item {
  border-radius: 10px;
  background-color: purple;
  margin: 5px;
}
```

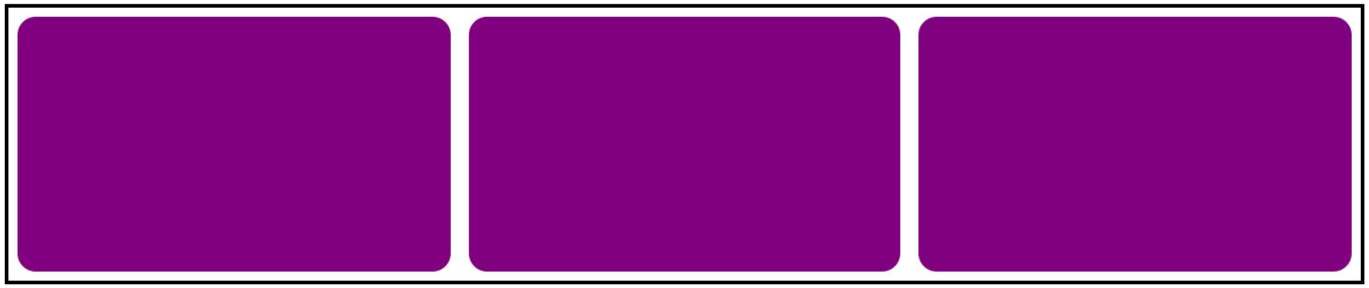
flex-grow example

If we set `flex-grow: 1`, the flex items fill the empty space:



```html
    <title>Flexbox example</title>
  </head>
  <body>

    <div id="flex-container">
      <span class="flex-item"></span>
      <div class="flex-item"></div>
      <span class="flex-item"></span>
    </div>

  </body>
</html>
```

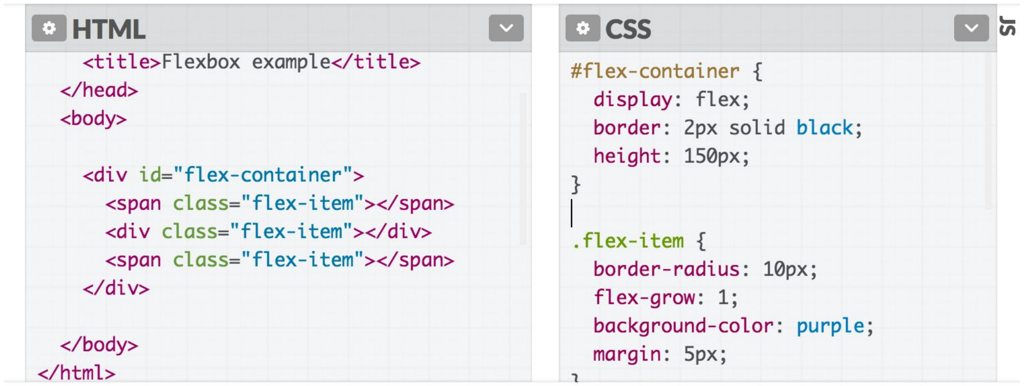
```css
#flex-container {
  display: flex;
  border: 2px solid black;
  height: 150px;
}

.flex-item {
  border-radius: 10px;
  flex-grow: 1;
  background-color: purple;
  margin: 5px;
```



Flex item height**?!

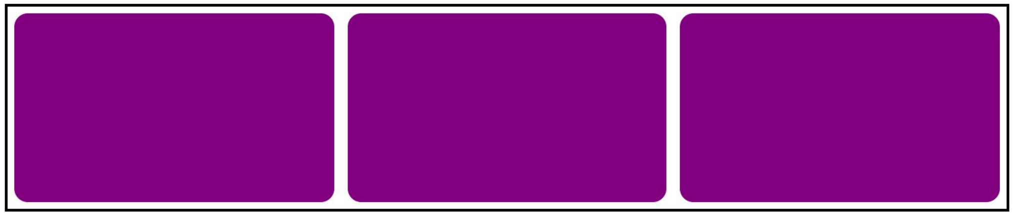
Note that `flex-grow` only controls width*.

So why does the height** of the flex items seem to "grow" as well?



```html
    <title>Flexbox example</title>
  </head>
  <body>

    <div id="flex-container">
      <span class="flex-item"></span>
      <div class="flex-item"></div>
      <span class="flex-item"></span>
    </div>

  </body>
</html>
```

```css
#flex-container {
  display: flex;
  border: 2px solid black;
  height: 150px;
}

.flex-item {
  border-radius: 10px;
  flex-grow: 1;
  background-color: purple;
  margin: 5px;
```



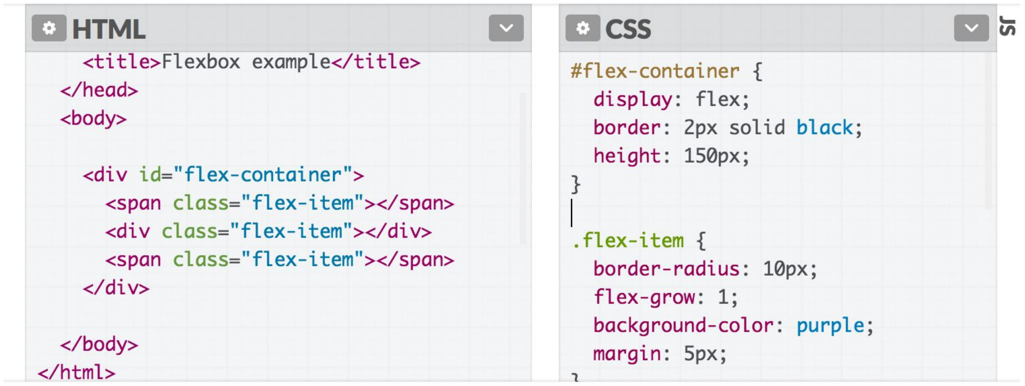
*width in the case of rows; height in the case of columns

**height in the case of rows; width in the case of columns

`align-items: stretch;`

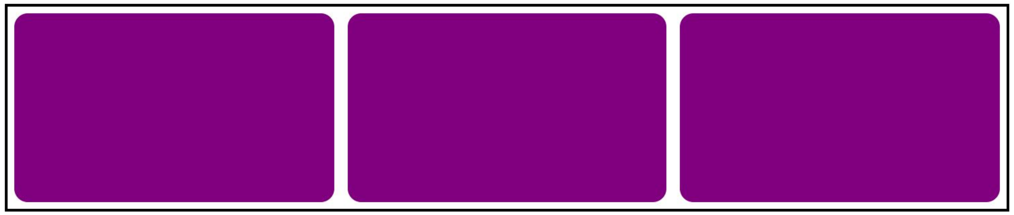
The default value of `align-items` is `stretch`, which means every flex item grows vertically* to fill the container by default.

(This will not happen if the `height` on the flex item is set)



```html
    <title>Flexbox example</title>
  </head>
  <body>

    <div id="flex-container">
      <span class="flex-item"></span>
      <div class="flex-item"></div>
      <span class="flex-item"></span>
    </div>

  </body>
</html>
```

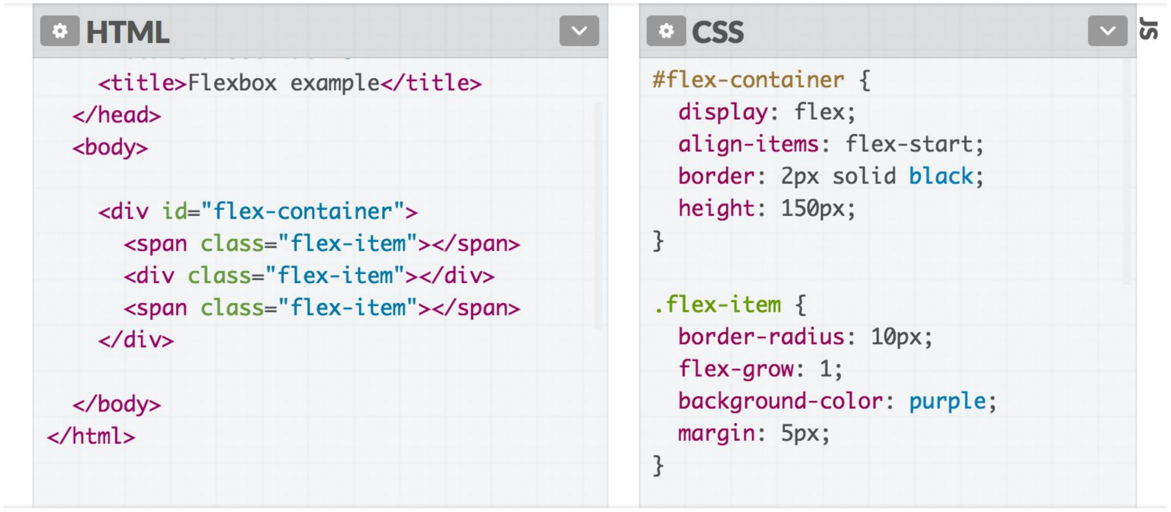
```css
#flex-container {
  display: flex;
  border: 2px solid black;
  height: 150px;
}

.flex-item {
  border-radius: 10px;
  flex-grow: 1;
  background-color: purple;
  margin: 5px;
```



*vertically in the case of rows; horizontally in the case of columns

align-items: stretch;

If we set another value for `align-items`, the flex items disappear again because the height is now content height, which is 0:

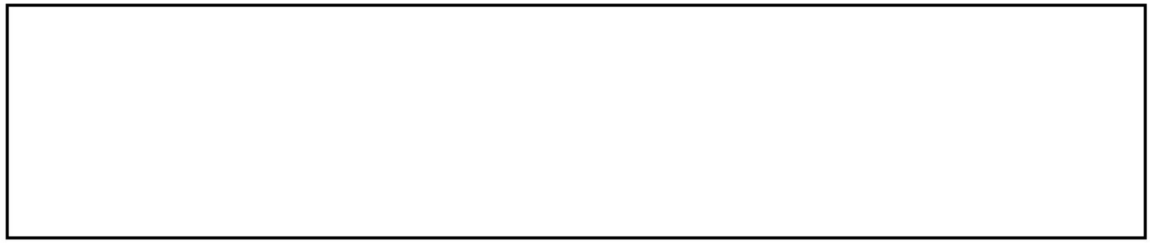


HTML

```
    <title>Flexbox example</title>
  </head>
  <body>

    <div id="flex-container">
      <span class="flex-item"></span>
      <div class="flex-item"></div>
      <span class="flex-item"></span>
    </div>

  </body>
</html>
```

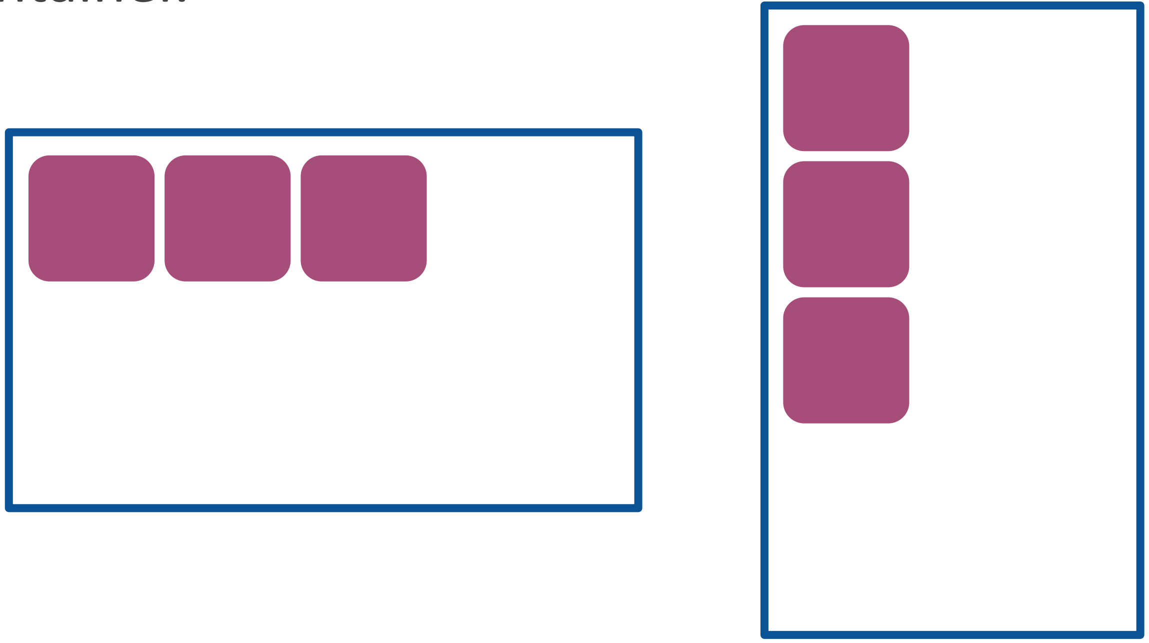
CSS

```
#flex-container {
  display: flex;
  align-items: flex-start;
  border: 2px solid black;
  height: 150px;
}

.flex-item {
  border-radius: 10px;
  flex-grow: 1;
  background-color: purple;
  margin: 5px;
}
```



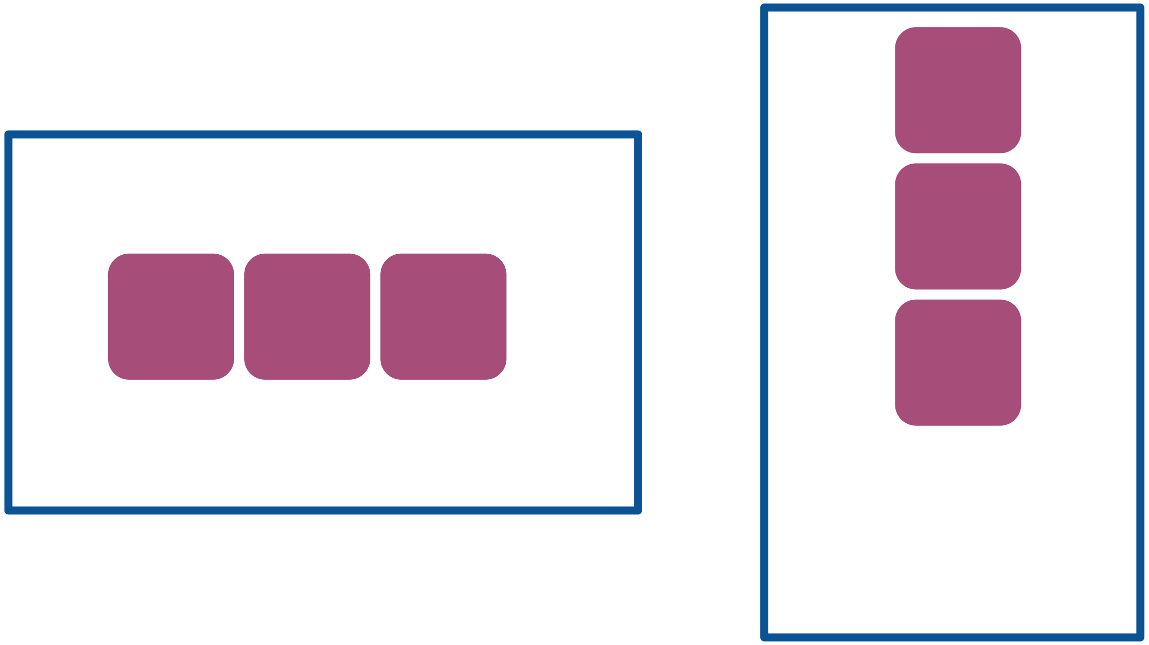
Flex layout recap

- If you set `display: flex`, the element is now a **flex container** and its direct children are **flex items**.
- The items in a flex container will layout in a `row` or `column` depending on the `flex-direction` of the container.



Flex layout recap

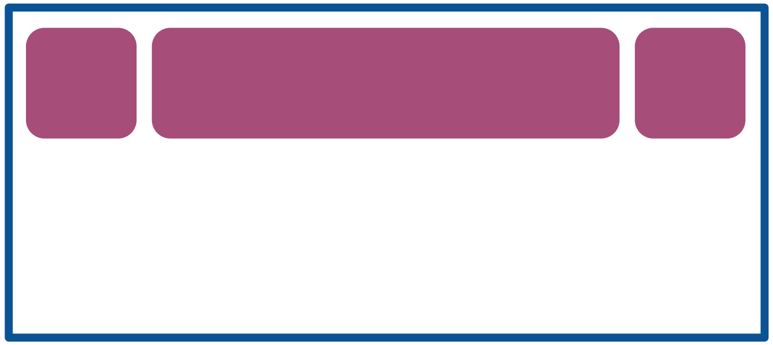
- **`justify-contents`** distributes the items horizontally for `flex-direction: row`, vertically for `column`
- **`align-items`** distributes the items vertically for `flex-direction: row`, horizontally for `column`



Flex layout recap

For `flex-direction: row`:

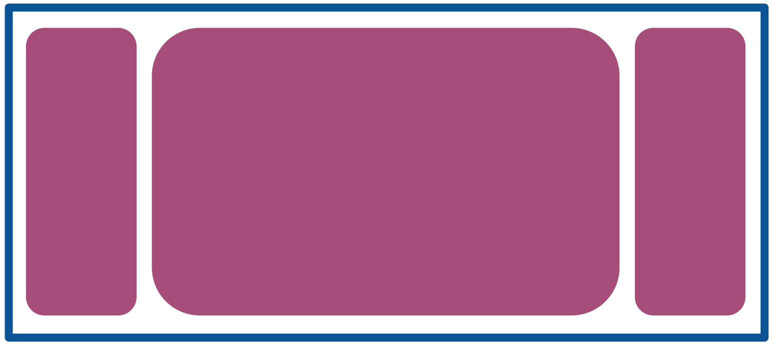
- The **flex basis** is the initial width of a flex item
  - This is either the explicitly set `width`, the explicitly set `flex-basis`, or the content width
- The width of a flex item will **shrink** to fit the container if `flex-shrink` is set to 1 (disabled if 0)
- The width of a flex item will **grow** to fit the remaining space if `flex-grow` is set to 1 (disabled if 0)



Flex layout recap

For `flex-direction: row`:

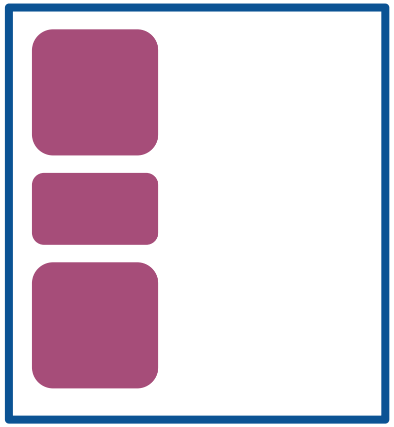
- The height of a flex item is either:
  - the explicitly set `height` on the item, or
  - the content height on the item, or
  - the height of the container if the container's `align-items: stretch;`



Flex layout recap

For `flex-direction: column`:

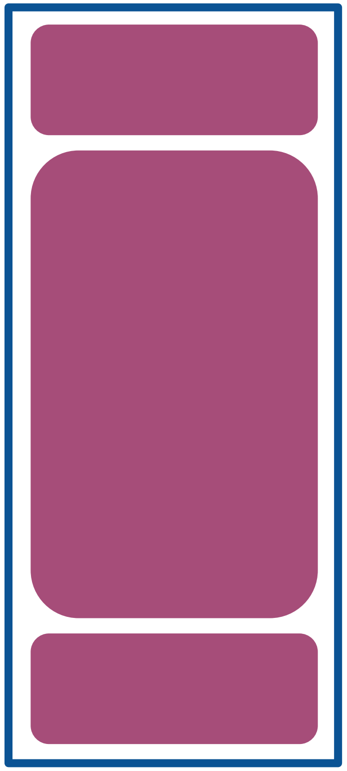
- The **flex basis** is the initial height of a flex item
  - This is either the explicitly set `height`, the explicitly set `flex-basis`, or the content height
- The height of a flex item will **shrink** to fit the container if `flex-shrink` is set to `1` (disabled if 0)
- The height of a flex item will **grow** to fit the remaining space if `flex-grow` is set to `1` (disabled if 0)



Flex layout recap

For `flex-direction: column`:

- The width of a flex item is either:
  - the explicitly set `width` on the item, or
  - the content width on the item, or
  - the width of the container if the container's `align-items: stretch;`



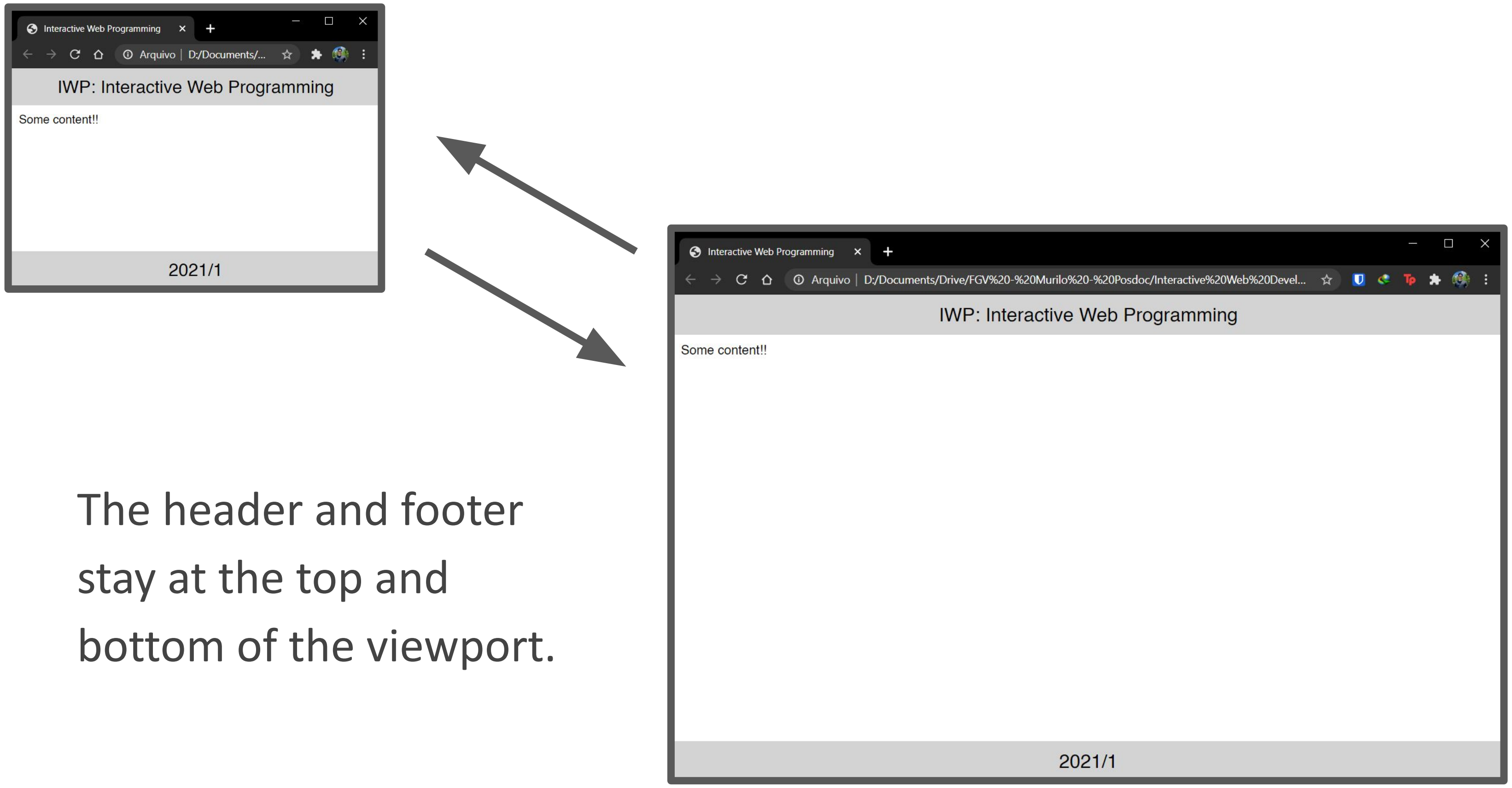
That's still just scratching the surface of flex box...

Questions?

Height and width quirks: `vh`, `vw`, `box-sizing`

Flexbox example

How do we make a layout that looks like this?


Interactive Web Programming
Arquivo | D:/Documents/...
Arquivo | D:/Documents/Drive/FGV%20-%20Murilo%20-%20Posdoc/Interactive%20Web%20Devel...

IWP: Interactive Web Programming

Some content!!

2021/1

The header and footer stay at the top and bottom of the viewport.

`height` and `width` percentages

When `width` is defined as a percentage:

- `width` is specified as a percentage of the **containing block's** `width`.

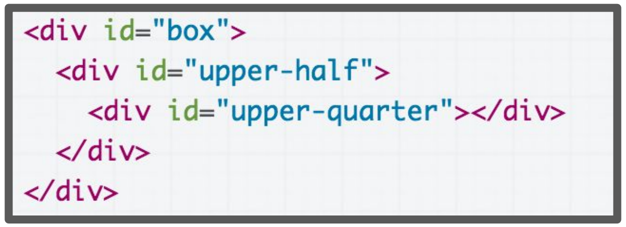
When `height` is defined as a percentage:

- `height` is specified as a percentage of the **containing block's** `height`.

In other words, `height` and `width` are defined **relative to their parent element** when defined as a percentage.

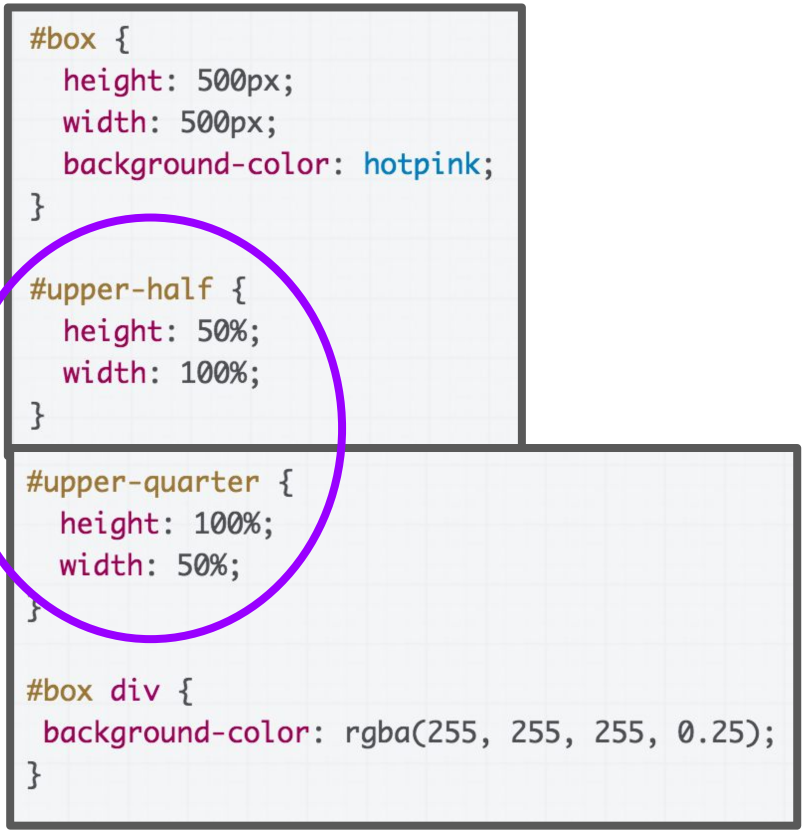
height and width percentages

HTML



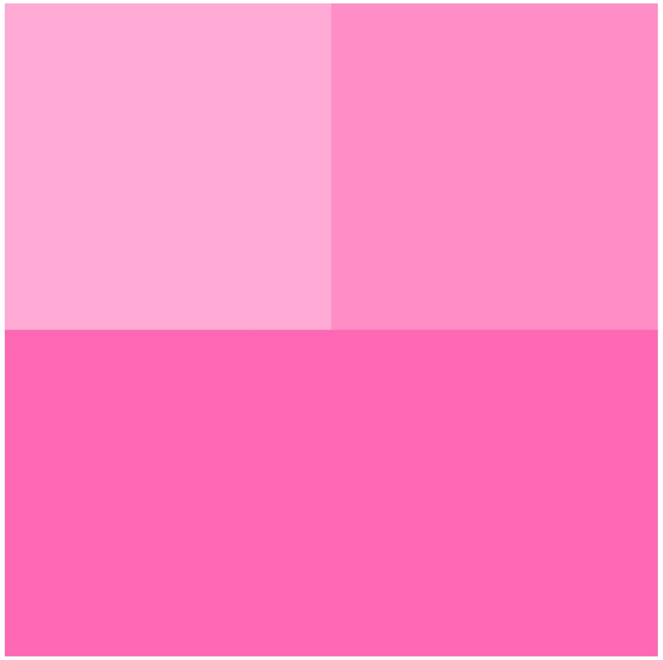
```
<div id="box">
  <div id="upper-half">
    <div id="upper-quarter"></div>
  </div>
</div>
```

CSS



```
#box {
  height: 500px;
  width: 500px;
  background-color: hotpink;
}

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

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

#box div {
 background-color: rgba(255, 255, 255, 0.25);
}
```

OUTPUT



(CodePen)

vh and vw

You can define `height` and `width` in terms of the viewport

- Use units vh and vw to set `height` and `width` to the percentage of the viewport's height and width, respectively (mdn)
- 1vh = 1/100th of the viewport height
- 1vw = 1/100th of the viewport width

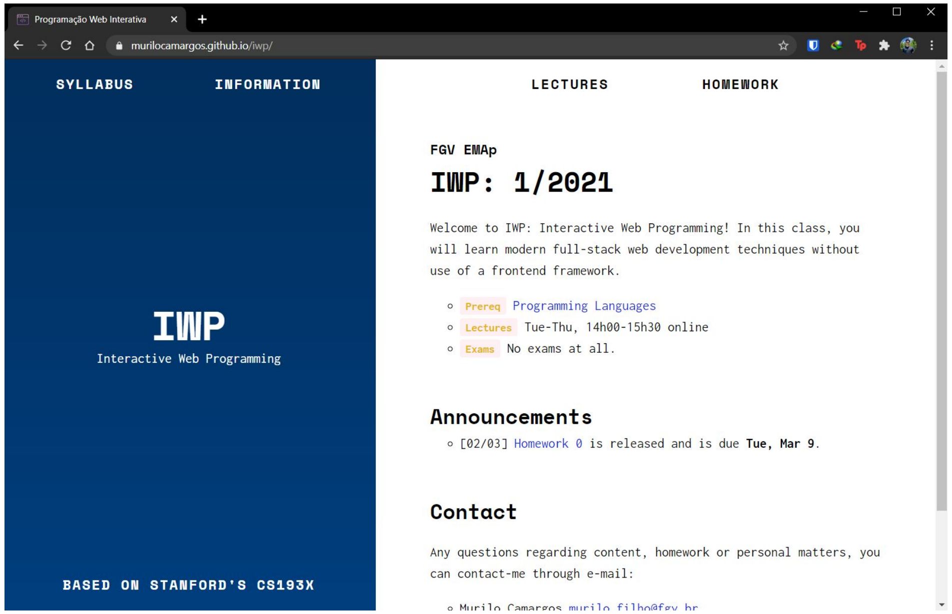
Example:

- `height: 100vh;`
- `width: 100vw;`

Viewport?

Browser vocabulary:

- **viewport:** the rectangle where the webpage shows up, scrollable via a scrollbar
- **chrome:** all the UI that's *not* the webpage, i.e. everything but the viewport


Programação Web Interativa

murilocamargos.github.io/iwp/

SYLLABUS INFORMATION LECTURES HOMEWORK

IWP

Interactive Web Programming

BASED ON STANFORD'S CS193X

FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

Announcements

- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

Contact

Any questions regarding content, homework or personal matters, you can contact-me through e-mail:

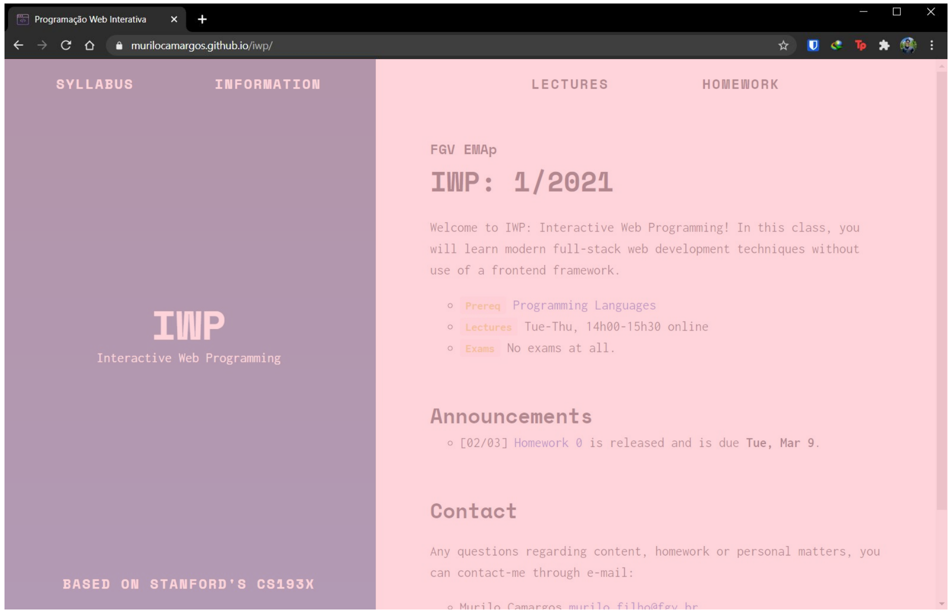
- Murilo Camargos murilo.filho@fgv.br

Viewport?

Browser vocabulary:

- **viewport:** the rectangle where the webpage shows up, scrollable via a scrollbar
- **chrome:** all the UI that's *not* the webpage, i.e. everything but the viewport

The viewport



Programação Web Interativa
murilocamargos.github.io/iwp/
SYLLABUS
INFORMATION
LECTURES
HOMEWORK
IWP
Interactive Web Programming
BASED ON STANFORD'S CS193X

FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

Announcements

- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

Contact

Any questions regarding content, homework or personal matters, you can contact-me through e-mail:

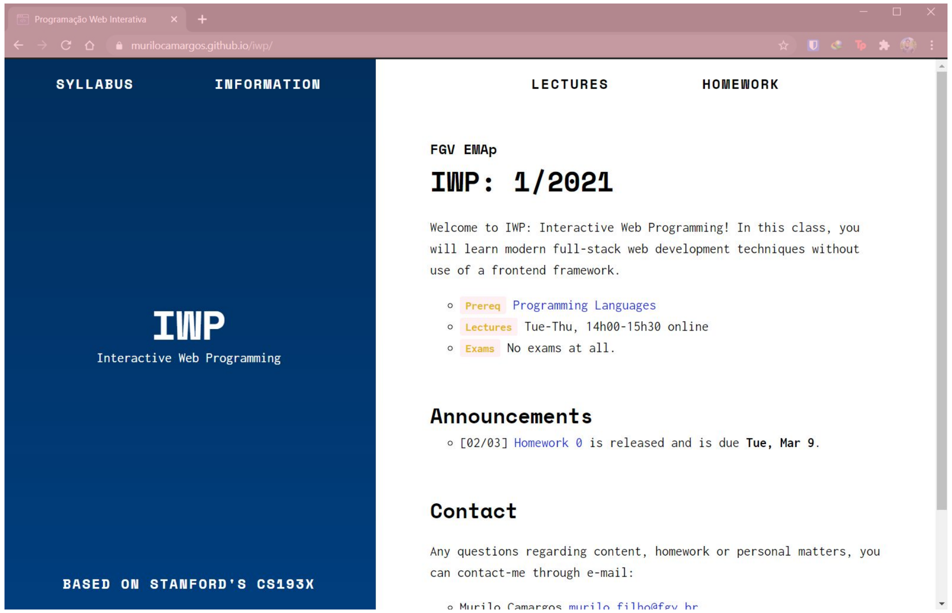
- Murilo Camargos murilo.filho@fgv.br

Viewport?

Browser vocabulary:

- **viewport:** the rectangle where the webpage shows up, scrollable via a scrollbar
- **chrome:** all the UI that's *not* the webpage, i.e. everything but the viewport

The Chrome



SYLLABUS INFORMATION LECTURES HOMEWORK

IWP
Interactive Web Programming

BASED ON STANFORD'S CS193X

FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

Announcements

- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

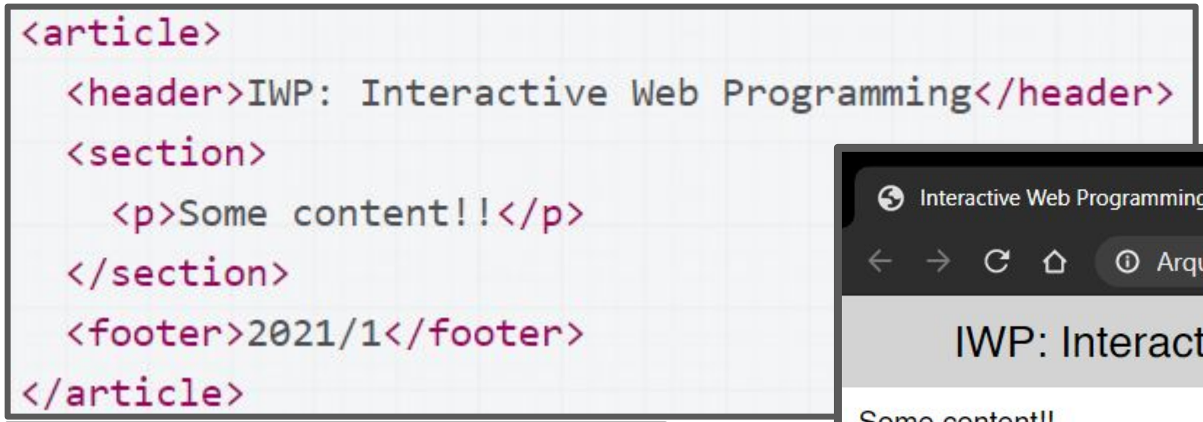
Contact

Any questions regarding content, homework or personal matters, you can contact-me through e-mail:

- Murilo Camargos murilo.filho@fgv.br

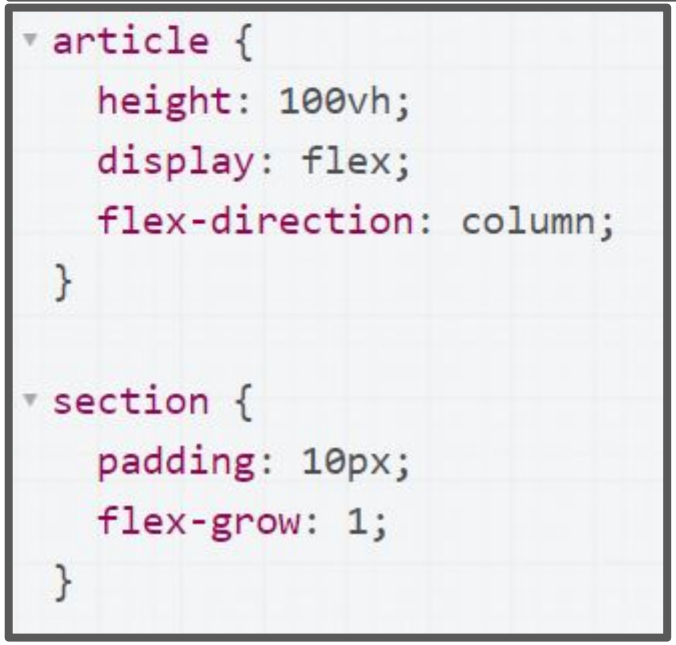
Flexbox example, solved

HTML

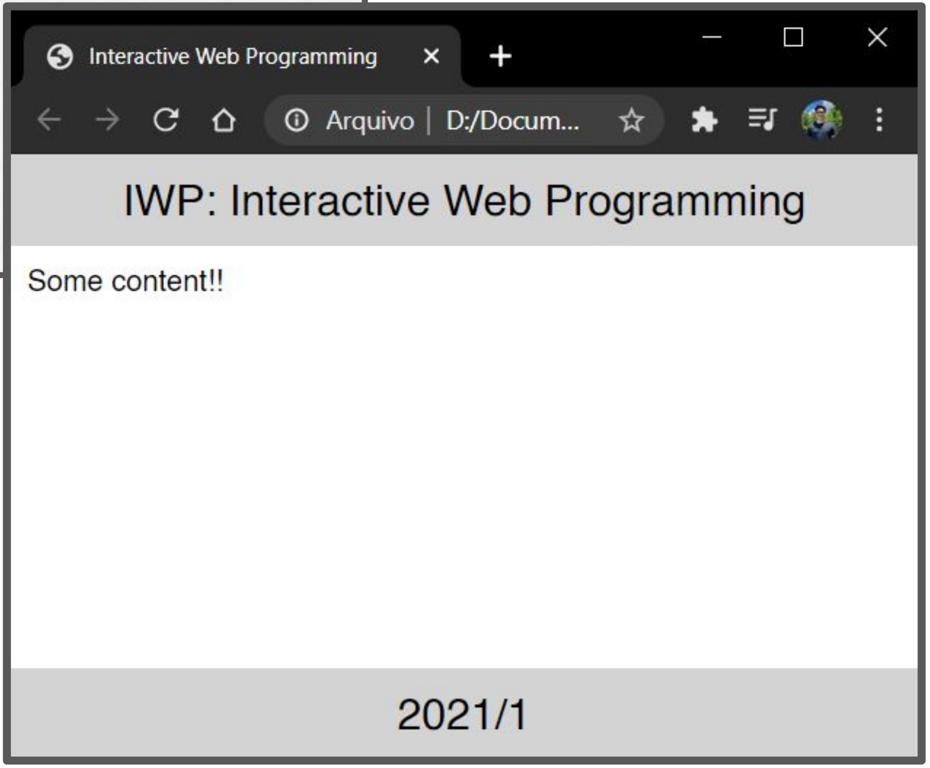


```html
<article>
  <header>IWP: Interactive Web Programming</header>
  <section>
    <p>Some content!!</p>
  </section>
  <footer>2021/1</footer>
</article>
```

CSS



```css
article {
  height: 100vh;
  display: flex;
  flex-direction: column;
}

section {
  padding: 10px;
  flex-grow: 1;
}
```



IWP: Interactive Web Programming

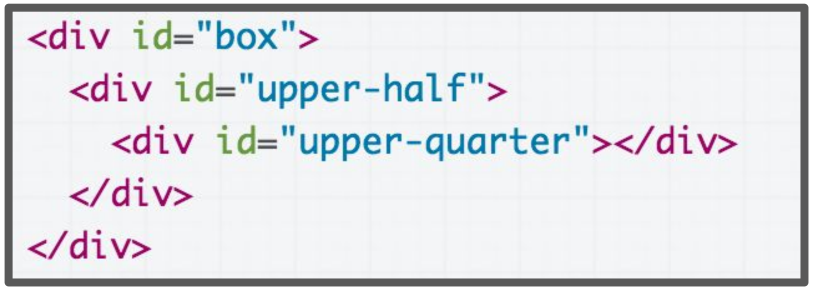
Some content!!

2021/1

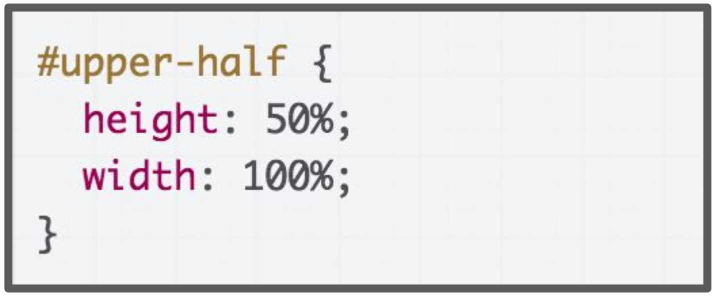
(CodePen)

Aside: sizing

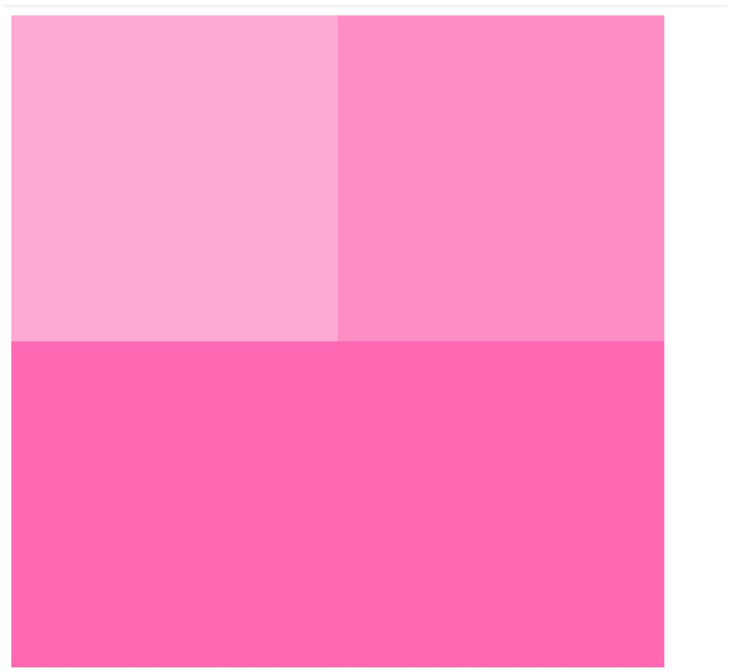
Q: What happens if we add a border to #upper-half?



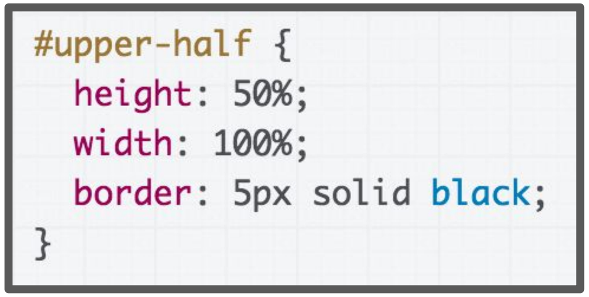
```
<div id="box">
  <div id="upper-half">
    <div id="upper-quarter"></div>
  </div>
</div>
```



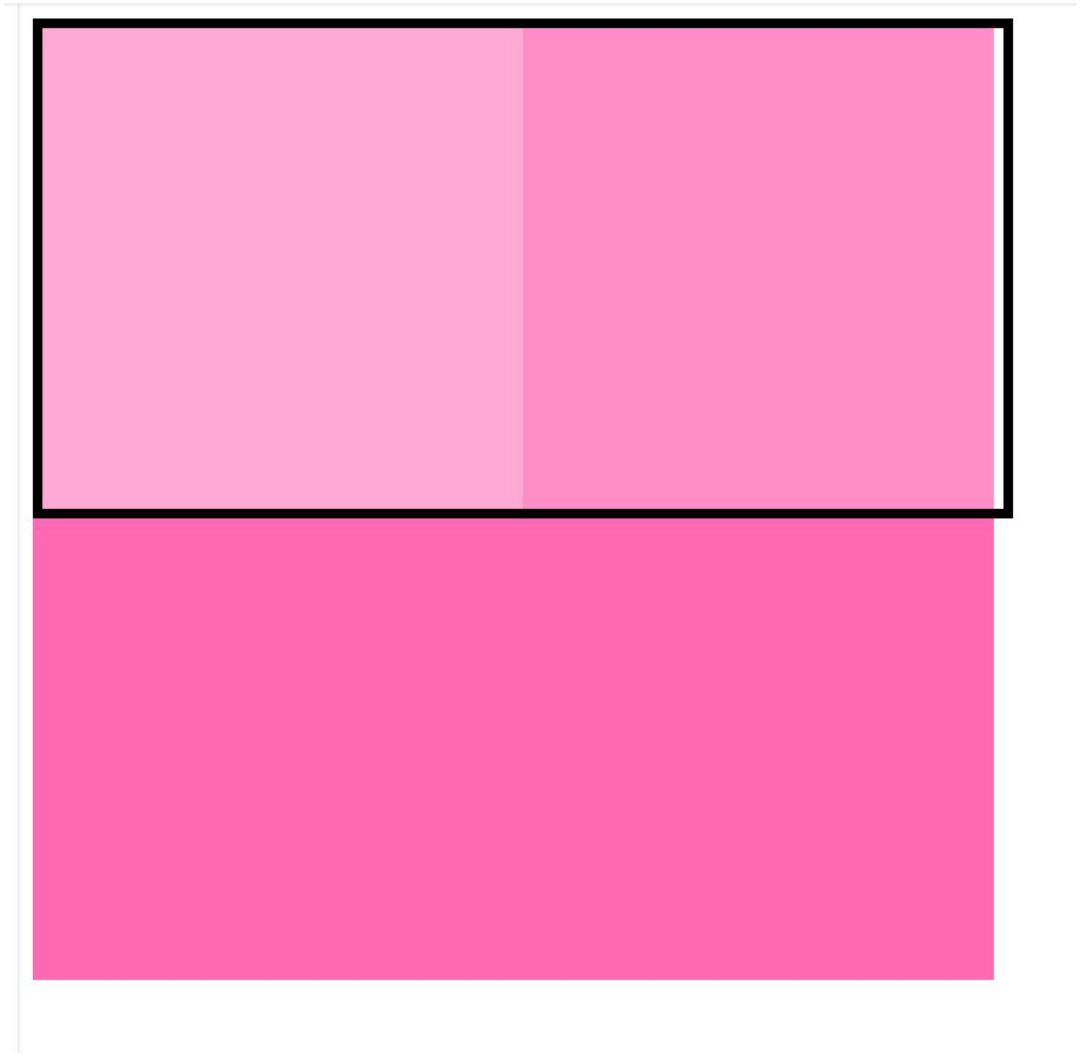
```
#upper-half {
  height: 50%;
  width: 100%;
}
```



???

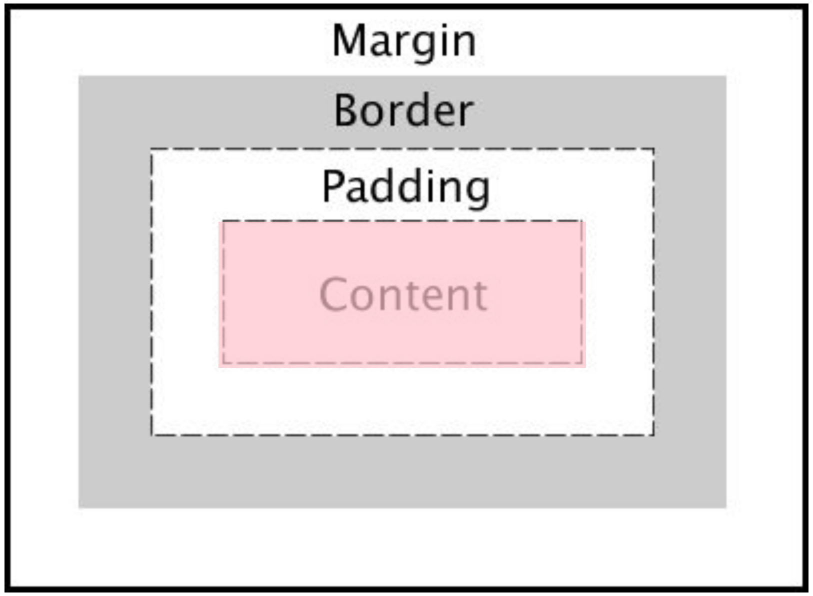


```
#upper-half {
  height: 50%;
  width: 100%;
  border: 5px solid black;
}
```



CSS box model width and height

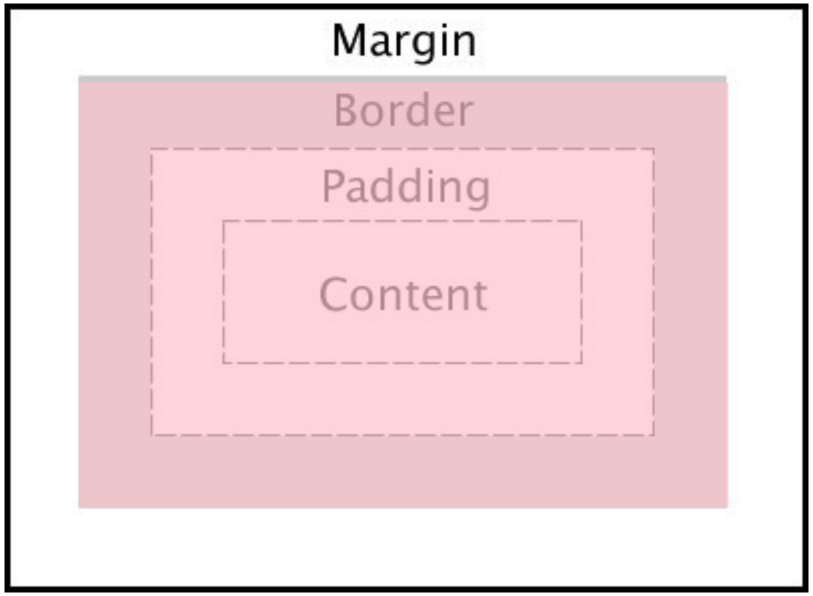
The box model defines CSS `width` and `height` properties to refer to the element's **content** width and height:



`box-sizing`

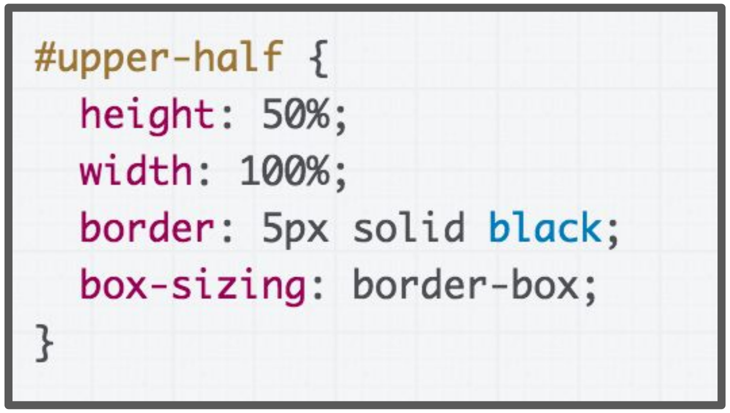
If you want to have `width` and `height` refer to the element's **border** width and height, use `box-sizing`:

- `box-sizing: border-box;`

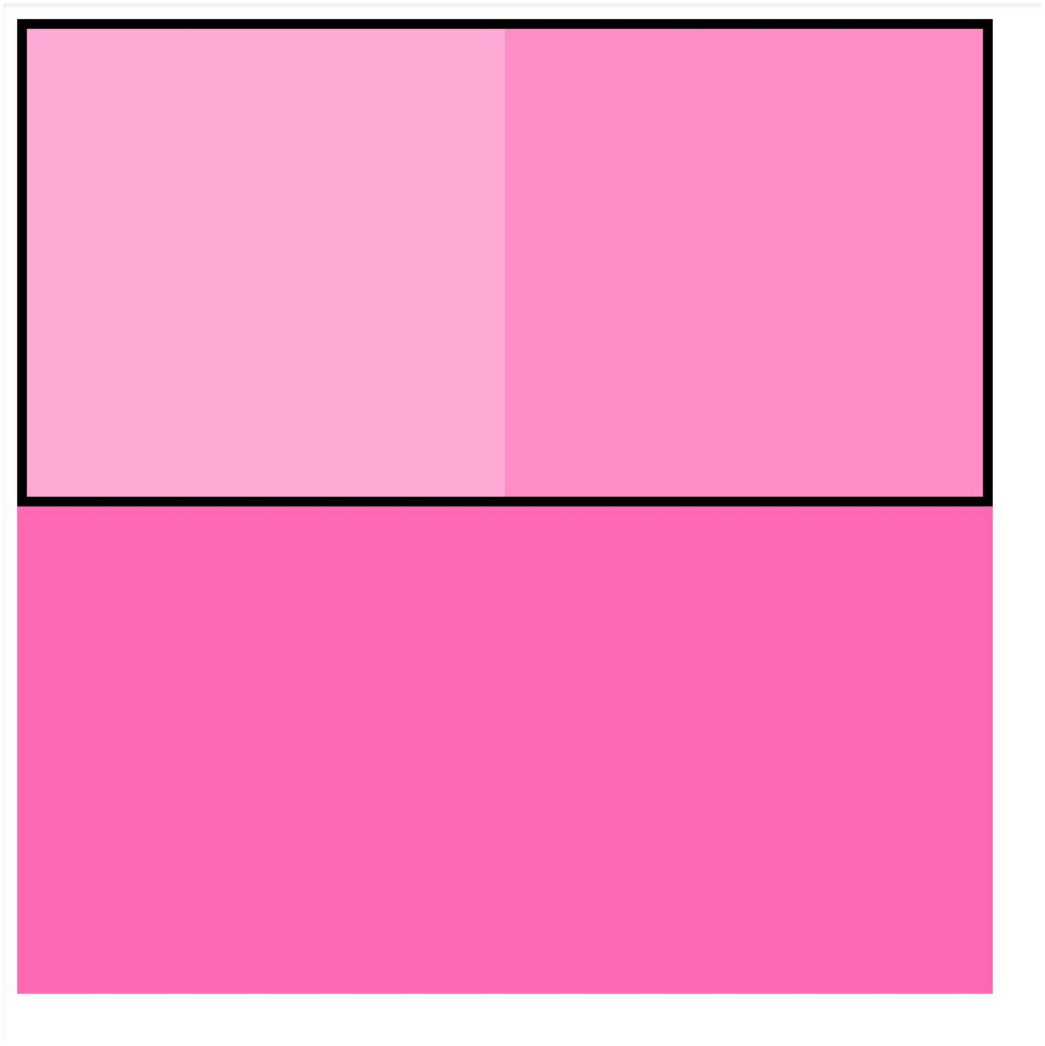


Note: Using `border-box` will include padding in the `width` and `height` as well.
Note: You **cannot** select padding-box or margin-box.

Fixed example



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



Another rendering mode: `position`

Moving things with `position`

Positioned layout lets you define precisely where an element should be in the page (mdn).

You can use positioned layout doing the following:

1. Define a **`position`** method:
   `Static, fixed, absolute, relative`
2. Define **`offsets`**: `top, left, bottom,` and `right`
3. (optional) Define **`z-index`** for overlapping layers (mdn)

Let's check it out!

Moving things with `position`

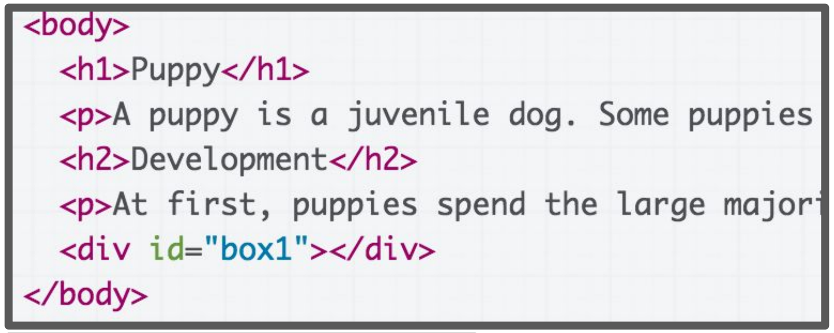
To specify exactly where an element goes, set its **`top, left, bottom,`** and/or **`right`** offset.

The meaning of these offset values depend on the reference point set by **`position`**:

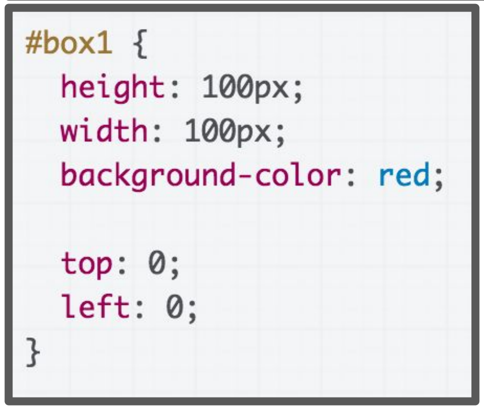
- **`static`**: no reference point; static block can't move *(this is the default style for every element)*
- **`fixed`**: a fixed position within the viewport
- **`absolute`**: a fixed position within its "containing element"
- **`relative`**: offset from its normal static position

`position: static`
(nothing happens!)

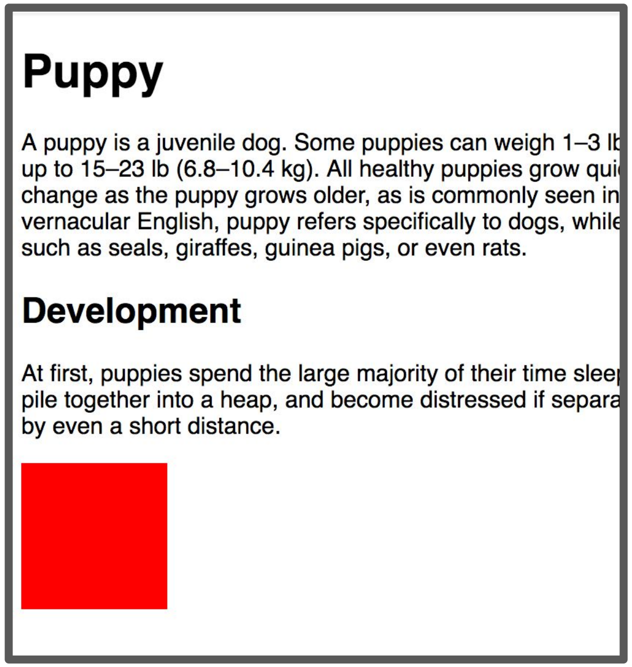
- `static` is the default value for `position`
- If you use `top` / `left` / `bottom` / `right` without setting a non-static `position`, nothing will happen



```
<body>
  <h1>Puppy</h1>
  <p>A puppy is a juvenile dog. Some puppies
  <h2>Development</h2>
  <p>At first, puppies spend the large majori
  <div id="box1"></div>
</body>
```



```
#box1 {
  height: 100px;
  width: 100px;
  background-color: red;

  top: 0;
  left: 0;
}
```



Puppy

A puppy is a juvenile dog. Some puppies can weigh 1–3 lb
up to 15–23 lb (6.8–10.4 kg). All healthy puppies grow qui
change as the puppy grows older, as is commonly seen in
vernacular English, puppy refers specifically to dogs, while
such as seals, giraffes, guinea pigs, or even rats.

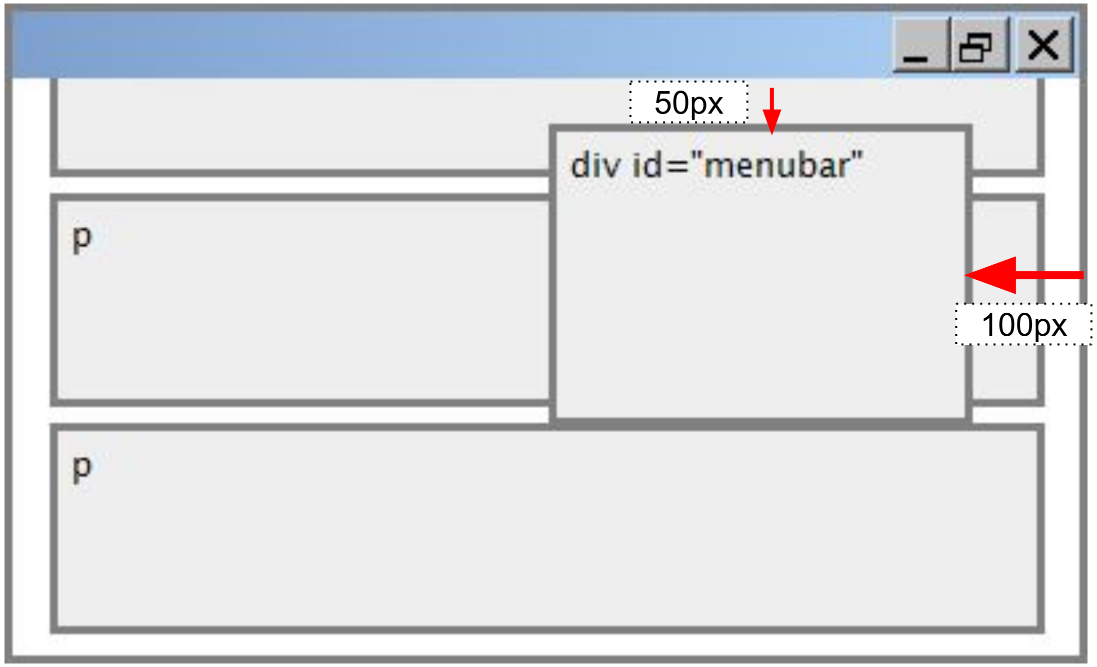
Development

At first, puppies spend the large majority of their time slee
pile together into a heap, and become distressed if separa
by even a short distance.

`position: fixed`

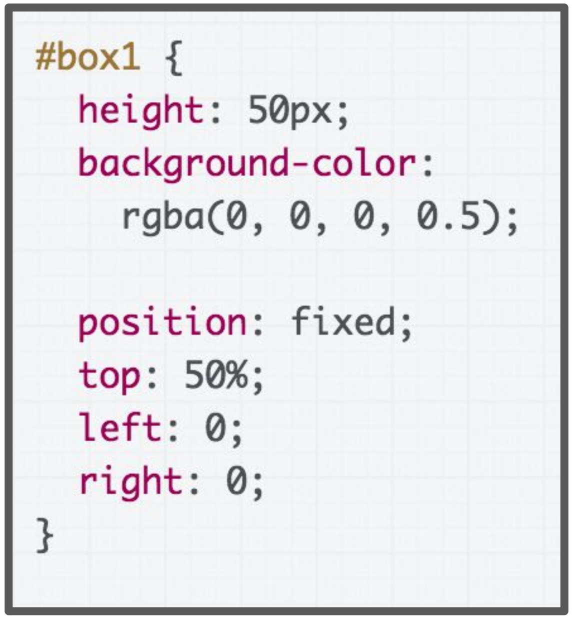
```
#menubar {
  position: fixed;
  top: 50px;
  right: 100px;
}
```

- For **fixed positioning**, the offset is the distance positioned relative to the viewport.
- The element does not move when scrolled.
- Element is **removed from normal document flow**, positioned on its own layer

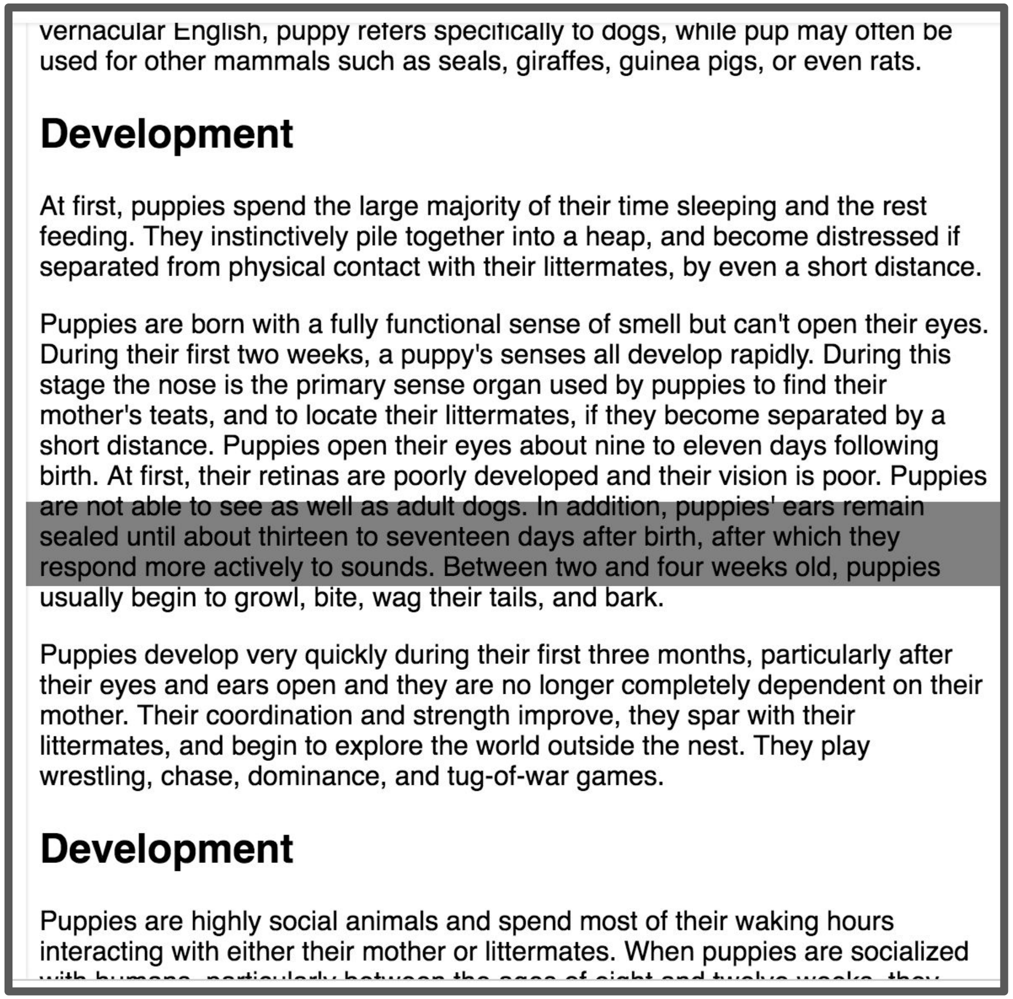


Often used to implement UIs; control bars that shouldn't go away

position: fixed



```
#box1 {
  height: 50px;
  background-color:
    rgba(0, 0, 0, 0.5);

  position: fixed;
  top: 50%;
  left: 0;
  right: 0;
}
```



vernacular English, puppy refers specifically to dogs, while pup may often be used for other mammals such as seals, giraffes, guinea pigs, or even rats.

Development

At first, puppies spend the large majority of their time sleeping and the rest feeding. They instinctively pile together into a heap, and become distressed if separated from physical contact with their littermates, by even a short distance.

Puppies are born with a fully functional sense of smell but can't open their eyes. During their first two weeks, a puppy's senses all develop rapidly. During this stage the nose is the primary sense organ used by puppies to find their mother's teats, and to locate their littermates, if they become separated by a short distance. Puppies open their eyes about nine to eleven days following birth. At first, their retinas are poorly developed and their vision is poor. Puppies are not able to see as well as adult dogs. In addition, puppies' ears remain sealed until about thirteen to seventeen days after birth, after which they respond more actively to sounds. Between two and four weeks old, puppies usually begin to growl, bite, wag their tails, and bark.

Puppies develop very quickly during their first three months, particularly after their eyes and ears open and they are no longer completely dependent on their mother. Their coordination and strength improve, they spar with their littermates, and begin to explore the world outside the nest. They play wrestling, chase, dominance, and tug-of-war games.

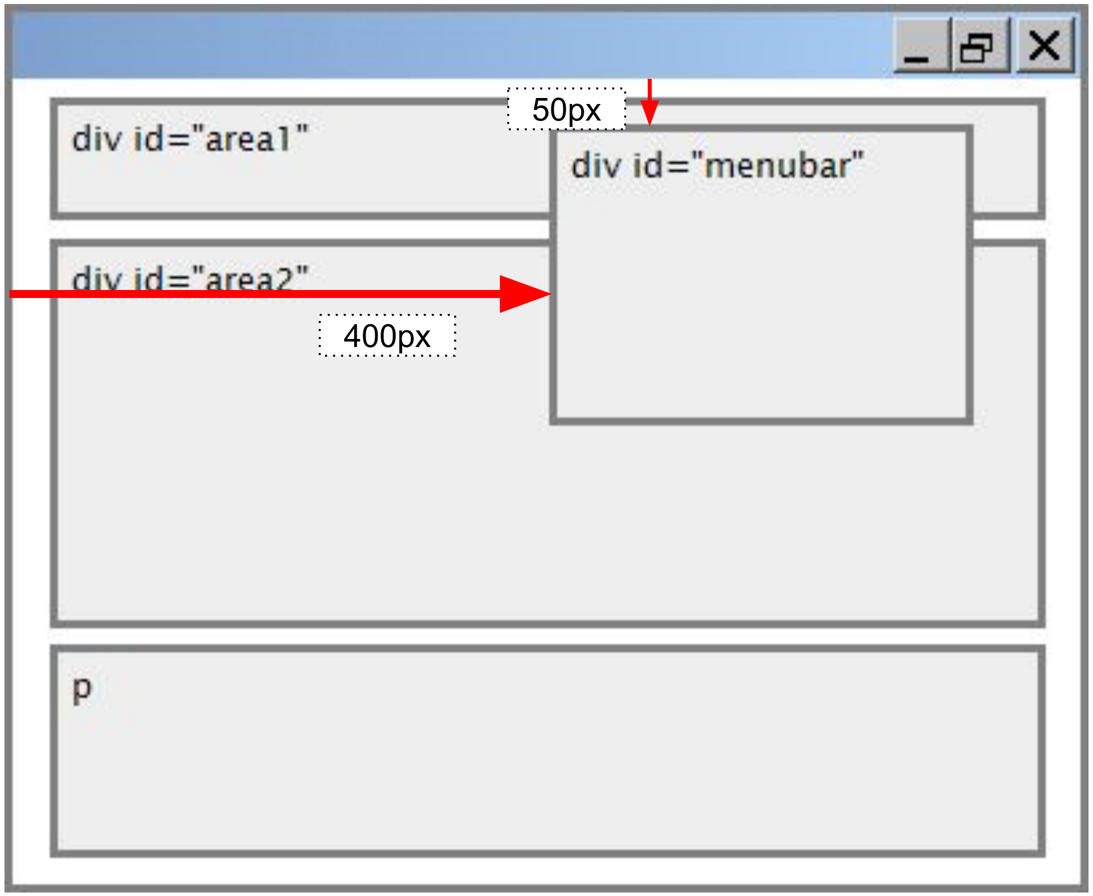
Development

Puppies are highly social animals and spend most of their waking hours interacting with either their mother or littermates. When puppies are socialized

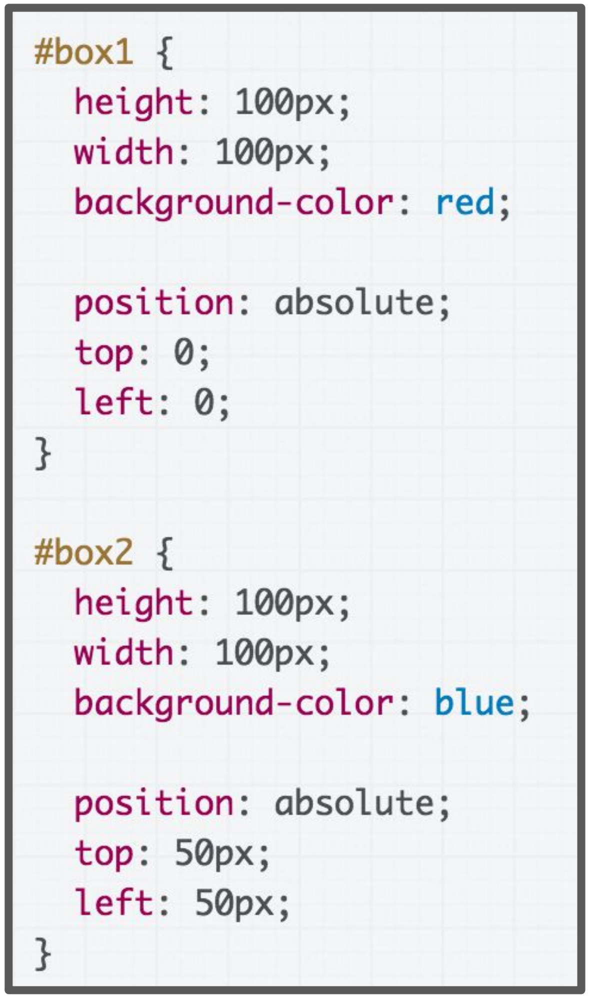
position: absolute

```
#menubar {
  position: absolute;
  left: 400px;
  top: 50px;
}
```

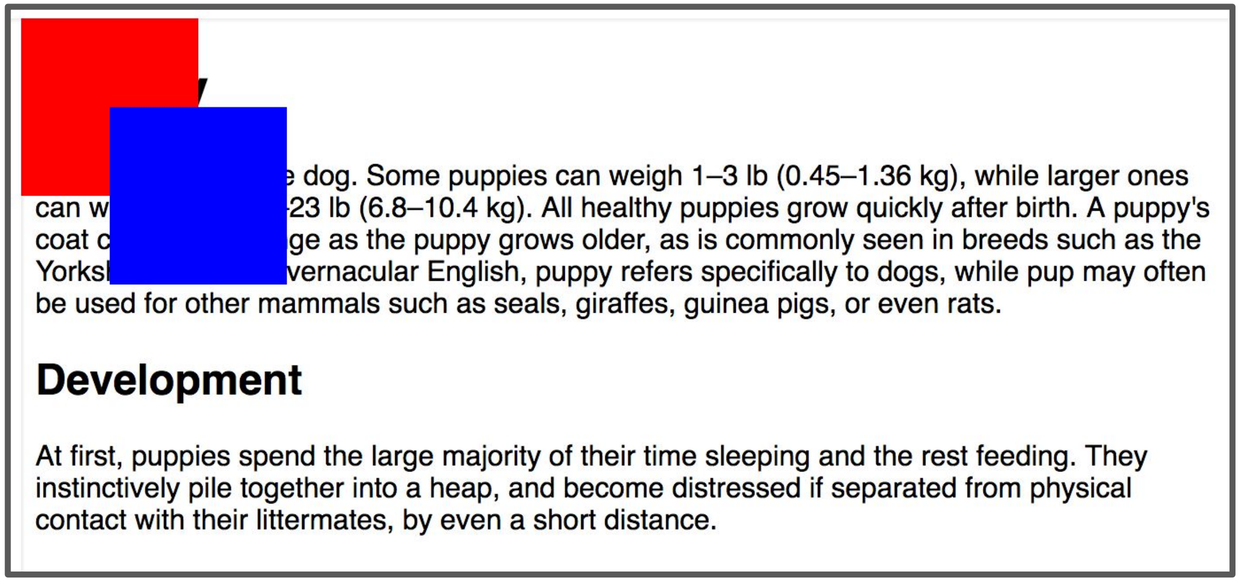
- For **absolute positioning**, the offset is the distance from the "containing element", which is the html element by default
- Element is removed from normal document flow, positioned on its own layer



position: absolute



```
#box1 {
  height: 100px;
  width: 100px;
  background-color: red;

  position: absolute;
  top: 0;
  left: 0;
}

#box2 {
  height: 100px;
  width: 100px;
  background-color: blue;

  position: absolute;
  top: 50px;
  left: 50px;
}
```



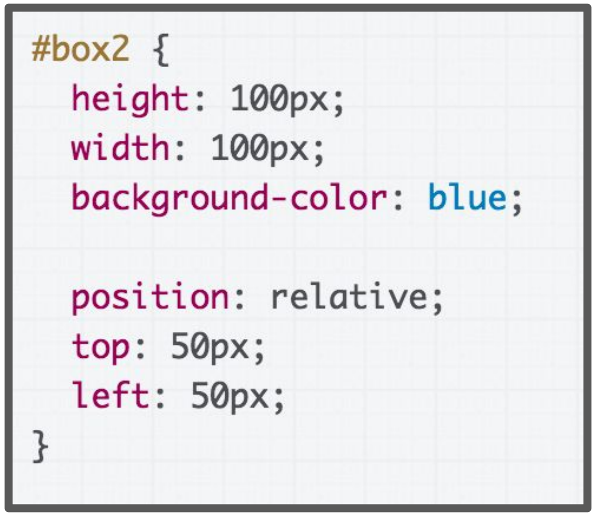
e dog. Some puppies can weigh 1–3 lb (0.45–1.36 kg), while larger ones
can w -23 lb (6.8–10.4 kg). All healthy puppies grow quickly after birth. A puppy's
coat c ge as the puppy grows older, as is commonly seen in breeds such as the
Yorksl vernacular English, puppy refers specifically to dogs, while pup may often
be used for other mammals such as seals, giraffes, guinea pigs, or even rats.

Development

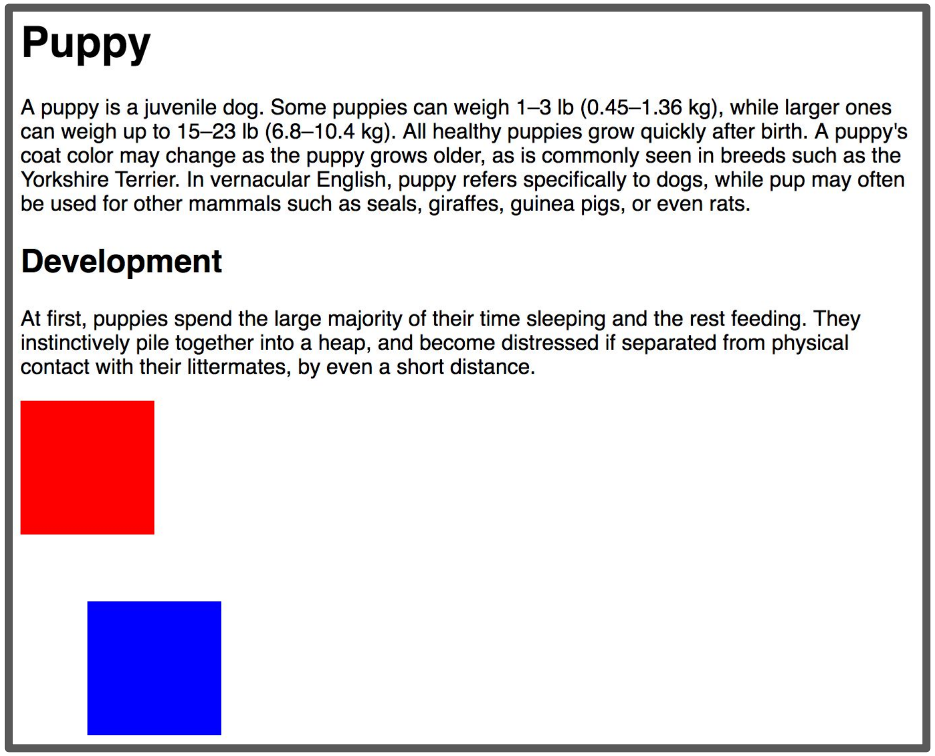
At first, puppies spend the large majority of their time sleeping and the rest feeding. They instinctively pile together into a heap, and become distressed if separated from physical contact with their littermates, by even a short distance.

position: relative

For `position: relative;` the element is placed where it would normally be placed in the layout of the page, but shifted by the `top / left / bottom / right` values.



```
#box2 {
  height: 100px;
  width: 100px;
  background-color: blue;

  position: relative;
  top: 50px;
  left: 50px;
}
```



Puppy

A puppy is a juvenile dog. Some puppies can weigh 1–3 lb (0.45–1.36 kg), while larger ones can weigh up to 15–23 lb (6.8–10.4 kg). All healthy puppies grow quickly after birth. A puppy's coat color may change as the puppy grows older, as is commonly seen in breeds such as the Yorkshire Terrier. In vernacular English, puppy refers specifically to dogs, while pup may often be used for other mammals such as seals, giraffes, guinea pigs, or even rats.

Development

At first, puppies spend the large majority of their time sleeping and the rest feeding. They instinctively pile together into a heap, and become distressed if separated from physical contact with their littermates, by even a short distance.

Relative absolute positioning

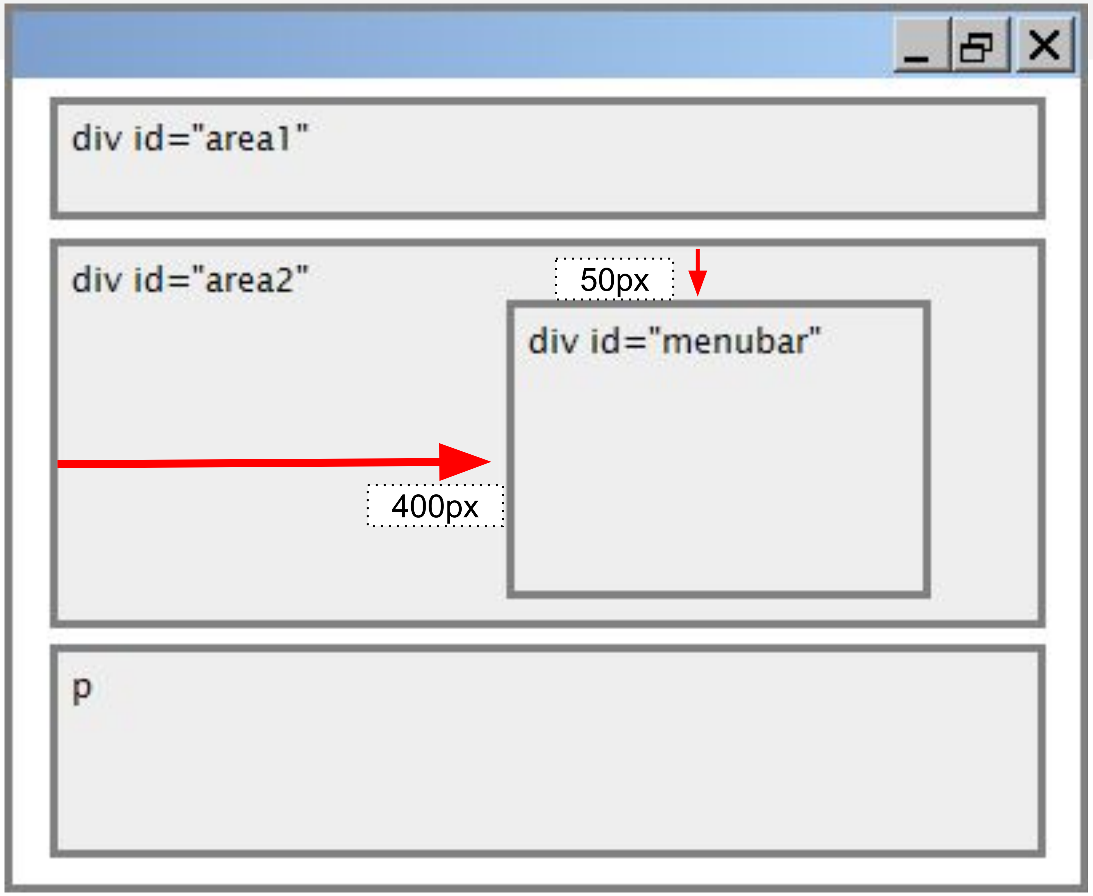
Let's revisit the definition of absolute positioning:

- **`absolute`**: a fixed position within its "containing element"
- The containing element is the viewport by default

You can change the containing element by setting "**`position: relative;`**" on some parent of your absolutely positioned element!

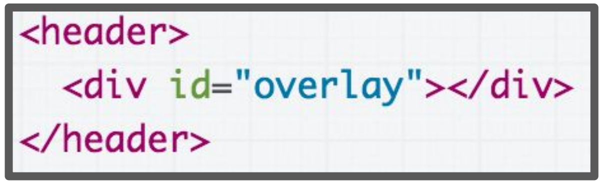
Relative absolute positioning

```
#area2 {
  position: relative;
}

#menubar {
  position: absolute;
  left: 400px;
  top: 50px;
}
```

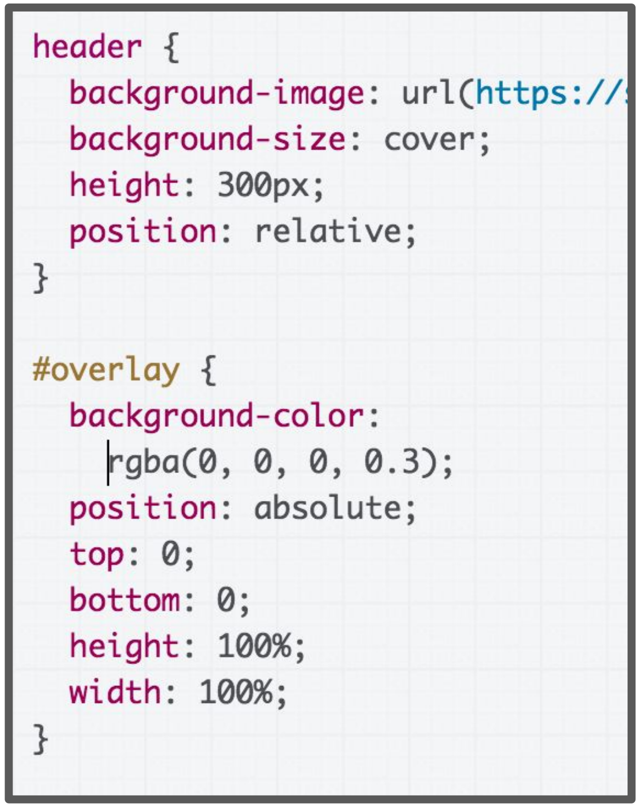


Offsets are relative to the first parent element that has **`position: relative`** which in this case is **#area2**

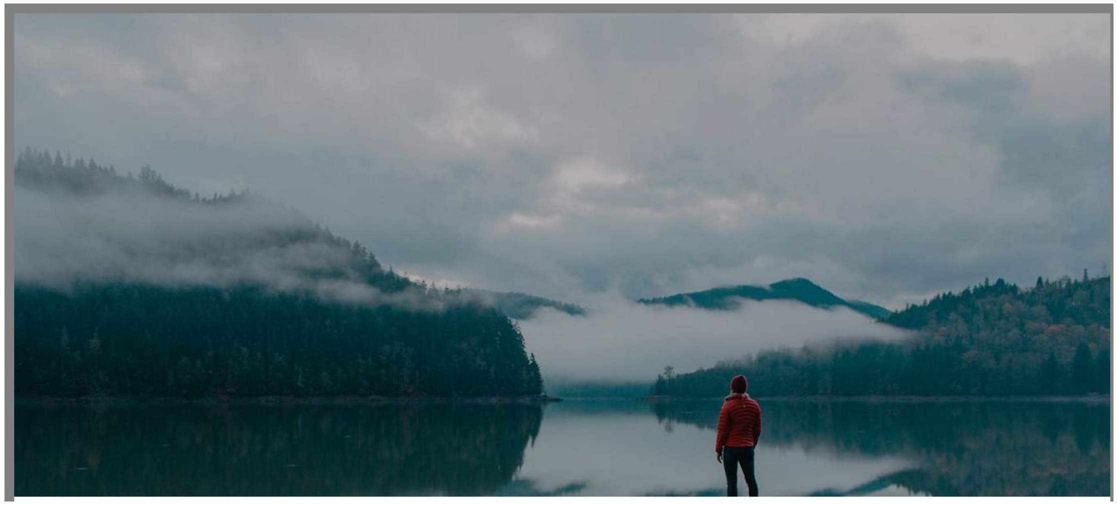
Common use case: Overlay



```html
<header>
  <div id="overlay"></div>
</header>
```



```css
header {
  background-image: url(https://
  background-size: cover;
  height: 300px;
  position: relative;
}

#overlay {
  background-color:
    rgba(0, 0, 0, 0.3);
  position: absolute;
  top: 0;
  bottom: 0;
  height: 100%;
  width: 100%;
}
```

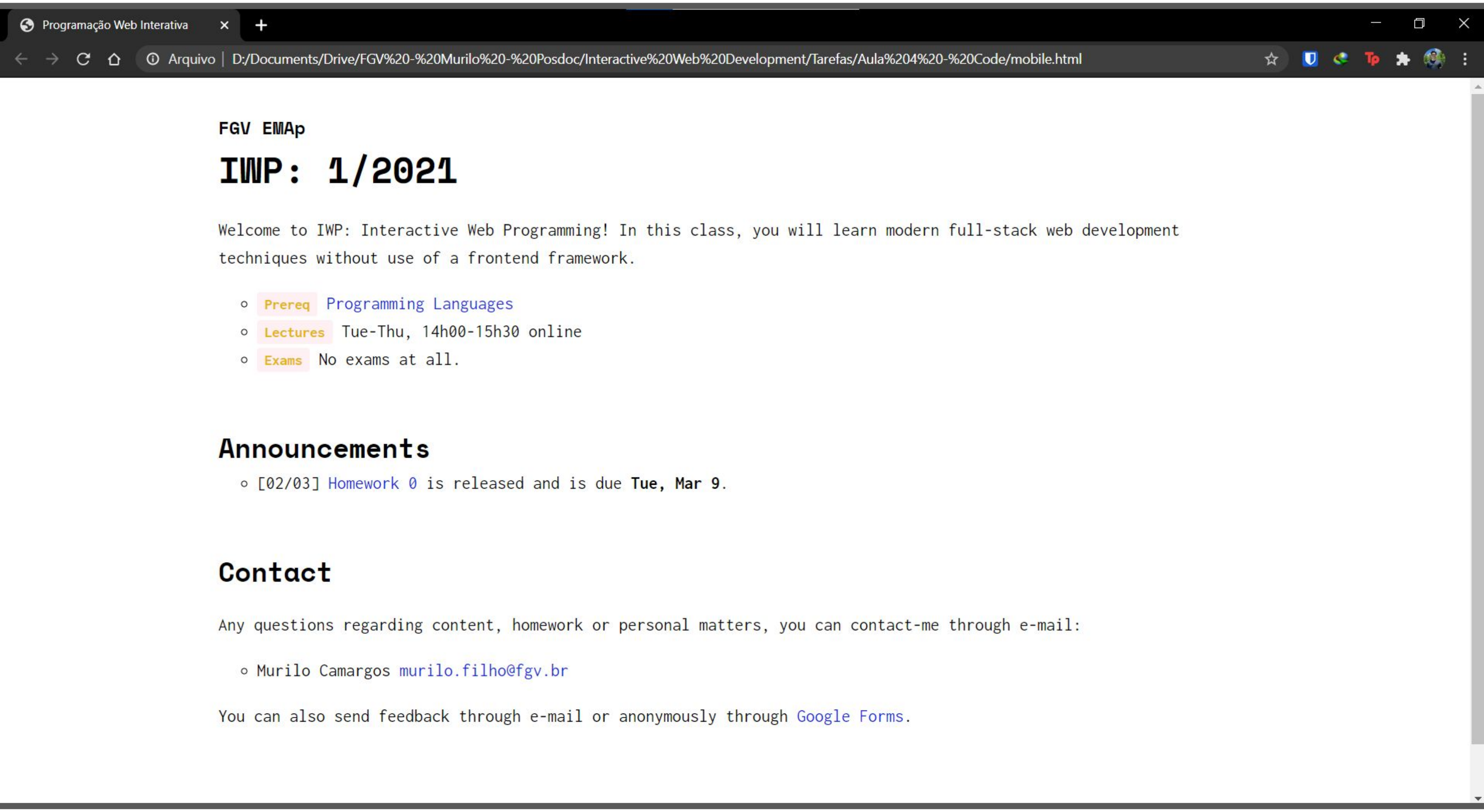


(CodePen)

Let's revisit Squarespace again!
(link to solution)

Mobile web

Say you have the following website:


Programação Web Interativa
Arquivo
D:/Documents/Drive/FGV%20-%20Murilo%20-%20Posdoc/Interactive%20Web%20Development/Tarefas/Aula%204%20-%20Code/mobile.html

FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

Announcements

- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

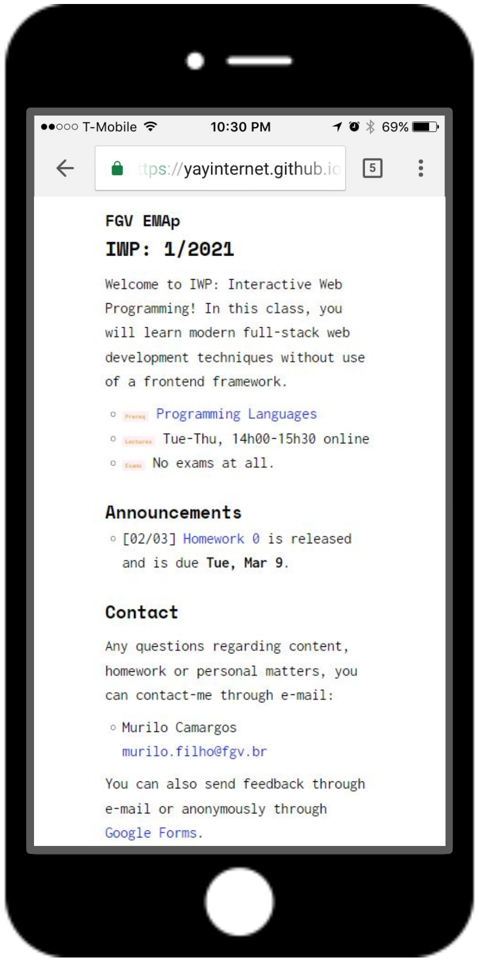
Contact

Any questions regarding content, homework or personal matters, you can contact-me through e-mail:

- Murilo Camargos murilo.filho@fgv.br

You can also send feedback through e-mail or anonymously through Google Forms.

Q: What does it look like on a phone?



FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

Announcements

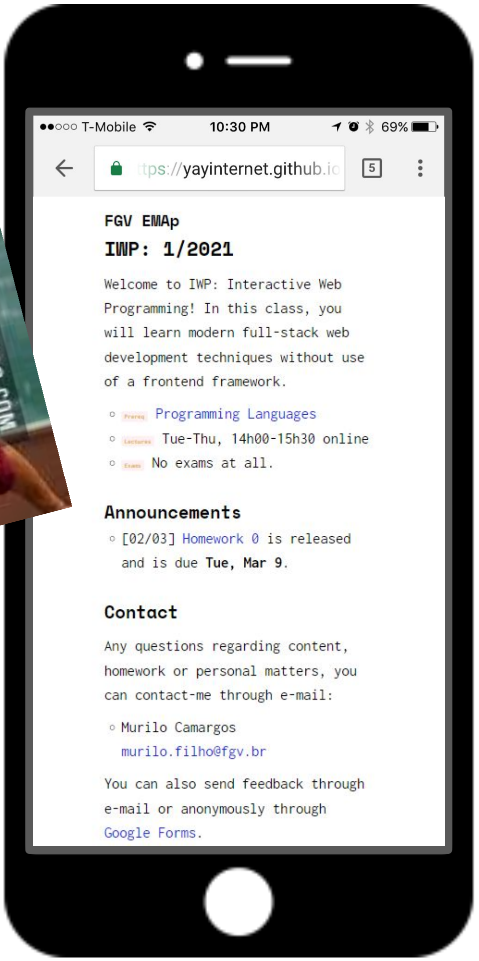
- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

Contact

Any questions regarding content, homework or personal matters, you can contact-me through e-mail:

- Murilo Camargos murilo.filho@fgv.br

You can also send feedback through e-mail or anonymously through Google Forms.



FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

Announcements

- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

Contact

Any questions regarding content, homework or personal matters, you can contact-me through e-mail:

- Murilo Camargos murilo.filho@fgv.br

You can also send feedback through e-mail or anonymously through Google Forms.

Not terrible… but pretty small and hard to read.

Responsive web design

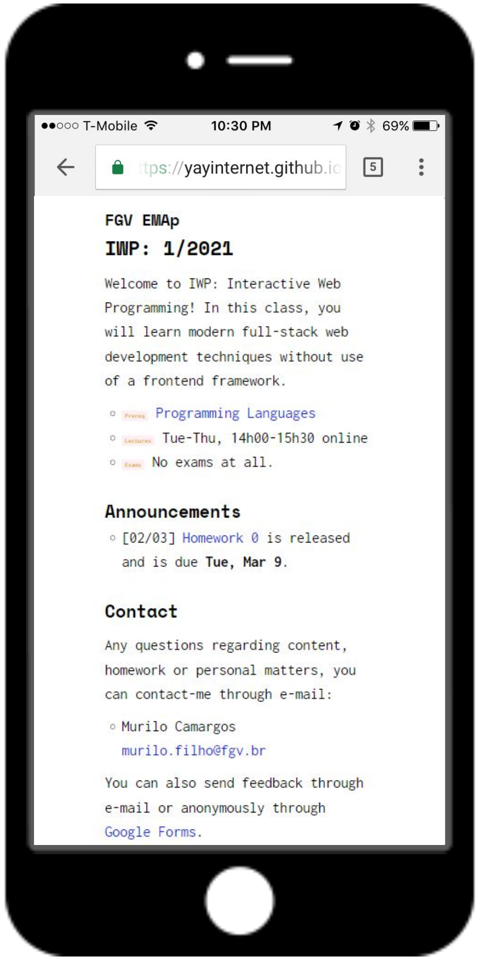
We want to write our CSS in a way that can look nice in a wide range of screen sizes:

- 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



●●○○○ T-Mobile 10:30 PM 69%

ttps://yayinternet.github.io

FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Programming Languages
- Tue-Thu, 14h00-15h30 online
- No exams at all.

Announcements

- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

Contact

Any questions regarding content, homework or personal matters, you can contact-me through e-mail:

- Murilo Camargos murilo.filho@fgv.br

You can also send feedback through e-mail or anonymously through Google Forms.

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

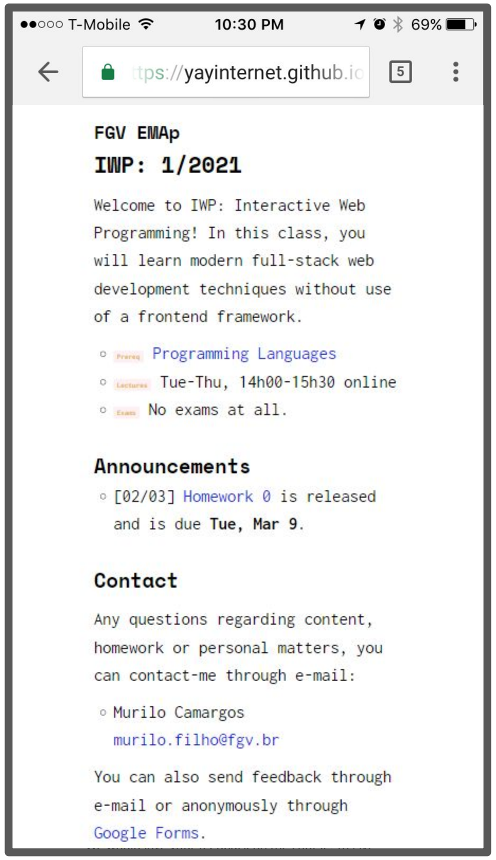
To prevent phone browsers from rendering the page at desktop width and zooming out, use the **meta viewport tag**:

```
<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



●●○○○ T-Mobile 10:30 PM 69%

tps://yayinternet.github.ic

FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

Announcements

- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

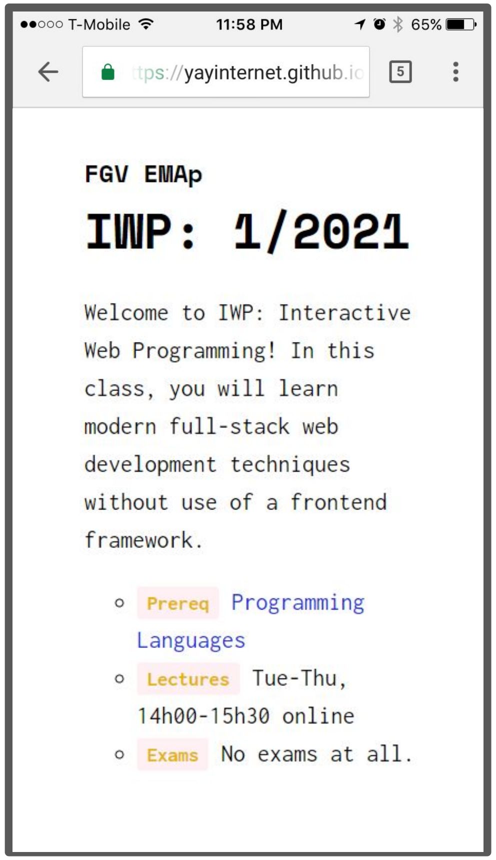
Contact

Any questions regarding content, homework or personal matters, you can contact-me through e-mail:

- Murilo Camargos murilo.filho@fgv.br

You can also send feedback through e-mail or anonymously through Google Forms.

Without the meta viewport tag



●●○○○ T-Mobile 11:58 PM 65%

tps://yayinternet.github.ic

FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

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

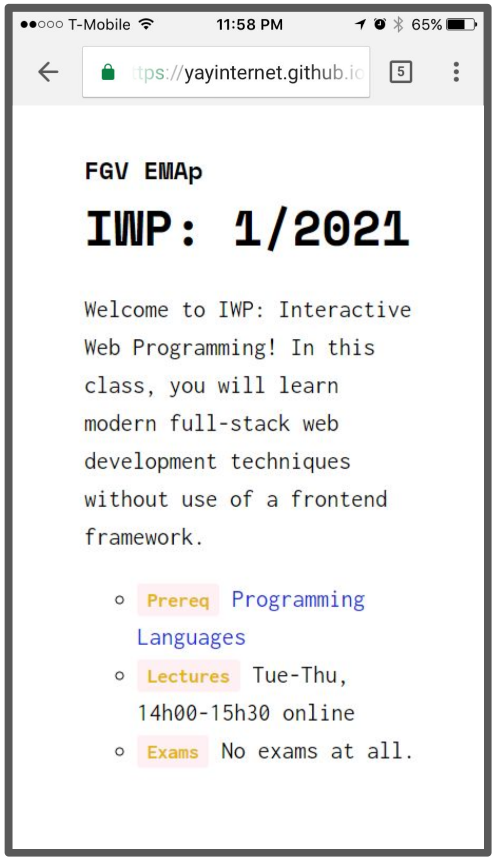
```
<meta name="viewport"
content="width=device-width, initial-scale=1">
```

(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 too big on mobile. Can we set a different margin property for mobile?


FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

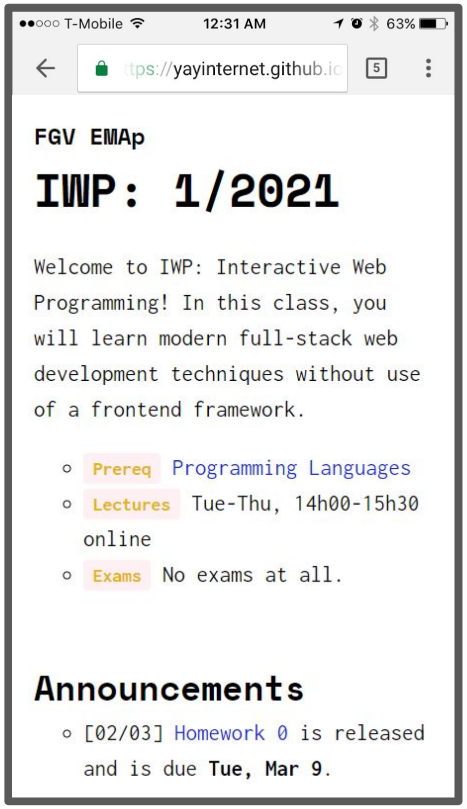
- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

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 {
    padding: 1em 0;
    width: 100%;
  }
}
```

You can create much more complex media queries as well.



FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

Announcements

- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

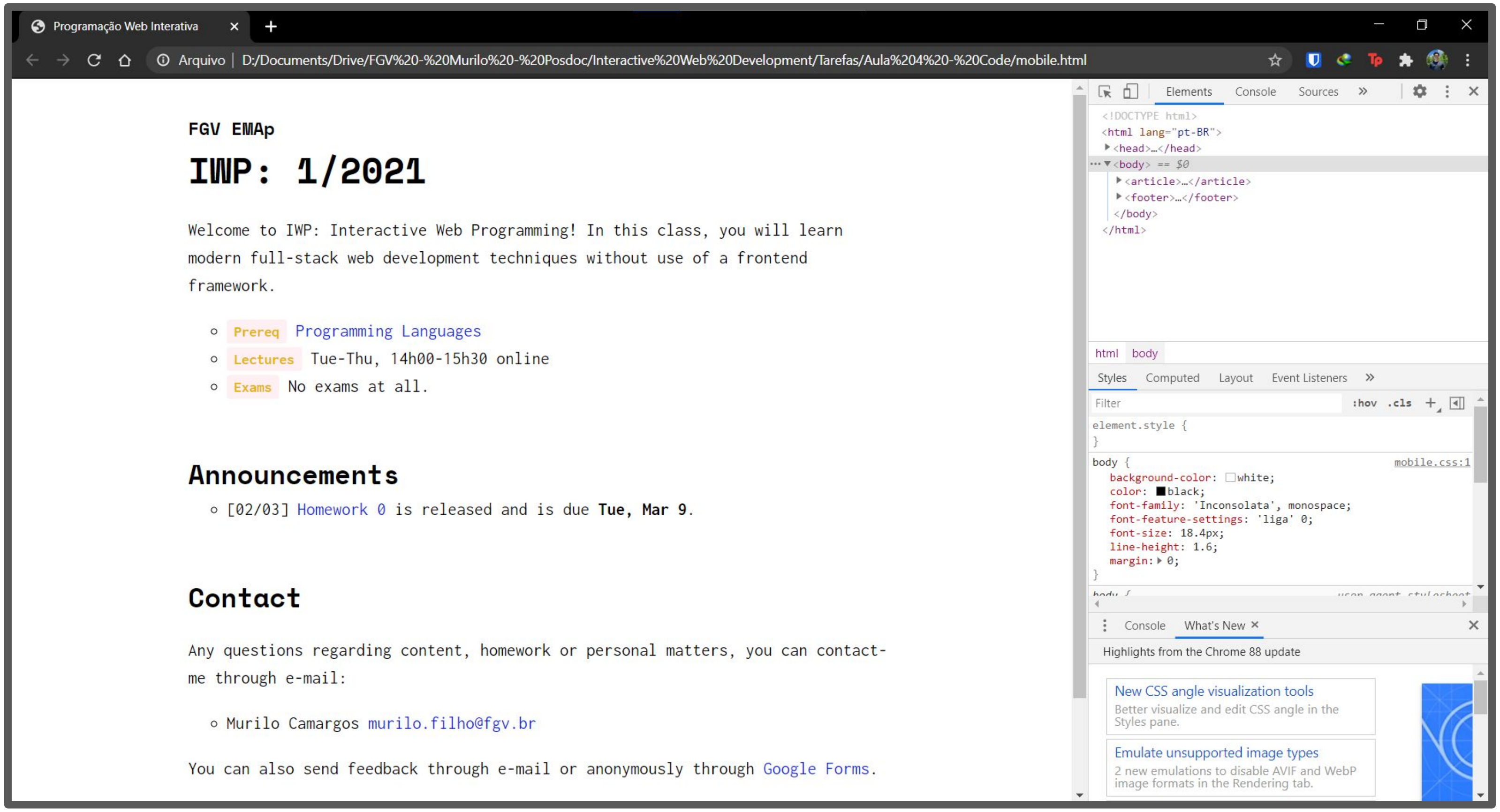
Development strategies

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:



Programação Web Interativa
Arquivo | D:/Documents/Drive/FGV%20-%20Murilo%20-%20Posdoc/Interactive%20Web%20Development/Tarefas/Aula%204%20-%20Code/mobile.html
Elements
Console
Sources
html
body
Styles
Computed
Layout
Event Listeners
Filter
:hov
.cls
Console
What's New

FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

Announcements

- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

Contact

Any questions regarding content, homework or personal matters, you can contact-me through e-mail:

- Murilo Camargos murilo.filho@fgv.br

You can also send feedback through e-mail or anonymously through Google Forms.

```
<!DOCTYPE html>
<html lang="pt-BR">
  <head>…</head>
  <body> == $0
    <article>…</article>
    <footer>…</footer>
  </body>
</html>
```

```
element.style {
}
body {                                                   mobile.css:1
  background-color: white;
  color: black;
  font-family: 'Inconsolata', monospace;
  font-feature-settings: 'liga' 0;
  font-size: 18.4px;
  line-height: 1.6;
  margin: 0;
}
```

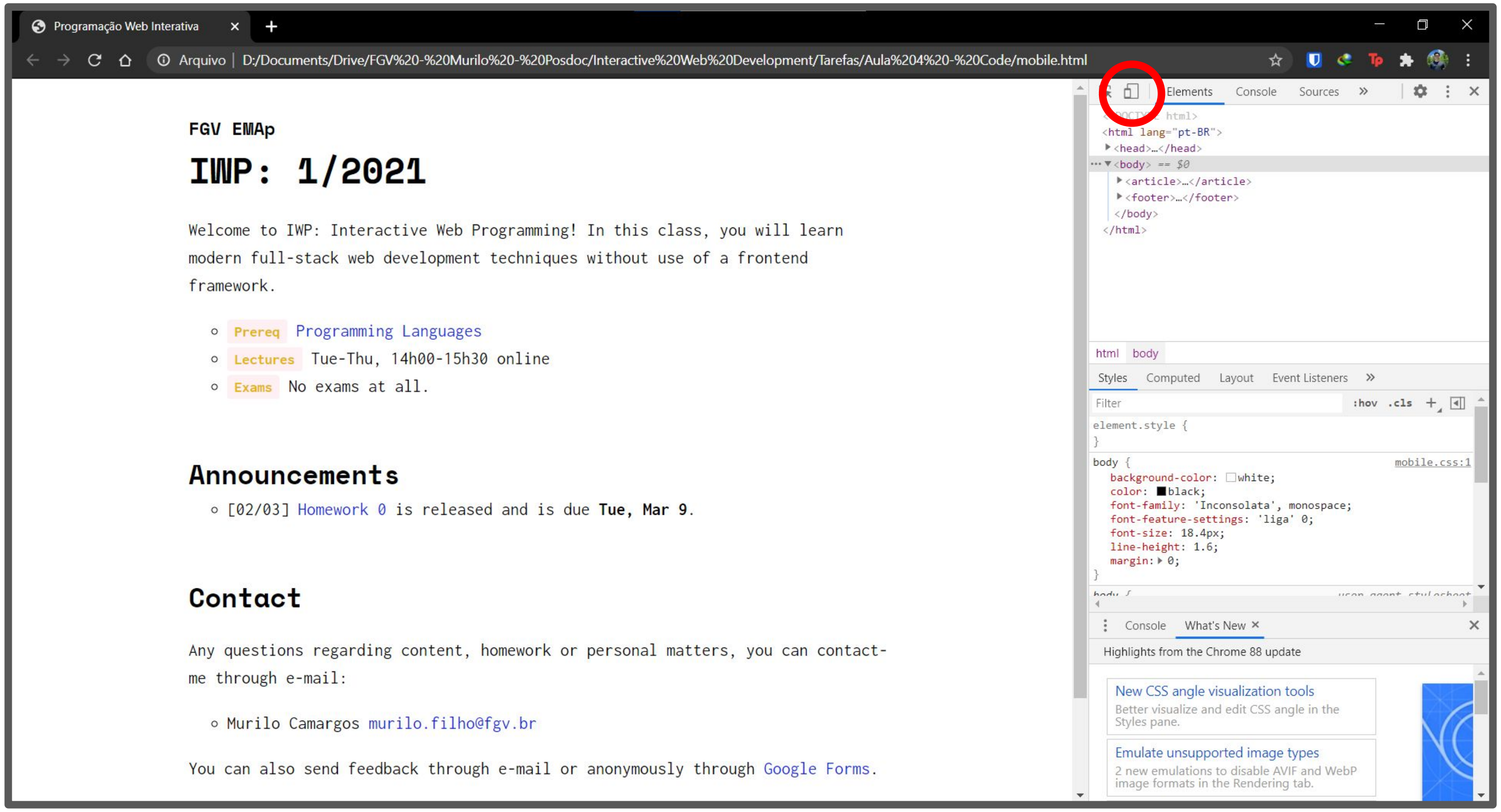
Highlights from the Chrome 88 update

New CSS angle visualization tools
Better visualize and edit CSS angle in the Styles pane.

Emulate unsupported image types
2 new emulations to disable AVIF and WebP image formats in the Rendering tab.

Chrome device mode

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



Programação Web Interativa

Arquivo | D:/Documents/Drive/FGV%20-%20Murilo%20-%20Posdoc/Interactive%20Web%20Development/Tarefas/Aula%204%20-%20Code/mobile.html

FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

Announcements

- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

Contact

Any questions regarding content, homework or personal matters, you can contact-me through e-mail:

- Murilo Camargos murilo.filho@fgv.br

You can also send feedback through e-mail or anonymously through Google Forms.

```
<!DOCTYPE html>
<html lang="pt-BR">
 ▸<head>…</head>
…▾<body> == $0
   ▸<article>…</article>
   ▸<footer>…</footer>
  </body>
</html>
```

```
element.style {
}
body {                                          mobile.css:1
    background-color: white;
    color: black;
    font-family: 'Inconsolata', monospace;
    font-feature-settings: 'liga' 0;
    font-size: 18.4px;
    line-height: 1.6;
    margin: ▸ 0;
}
```

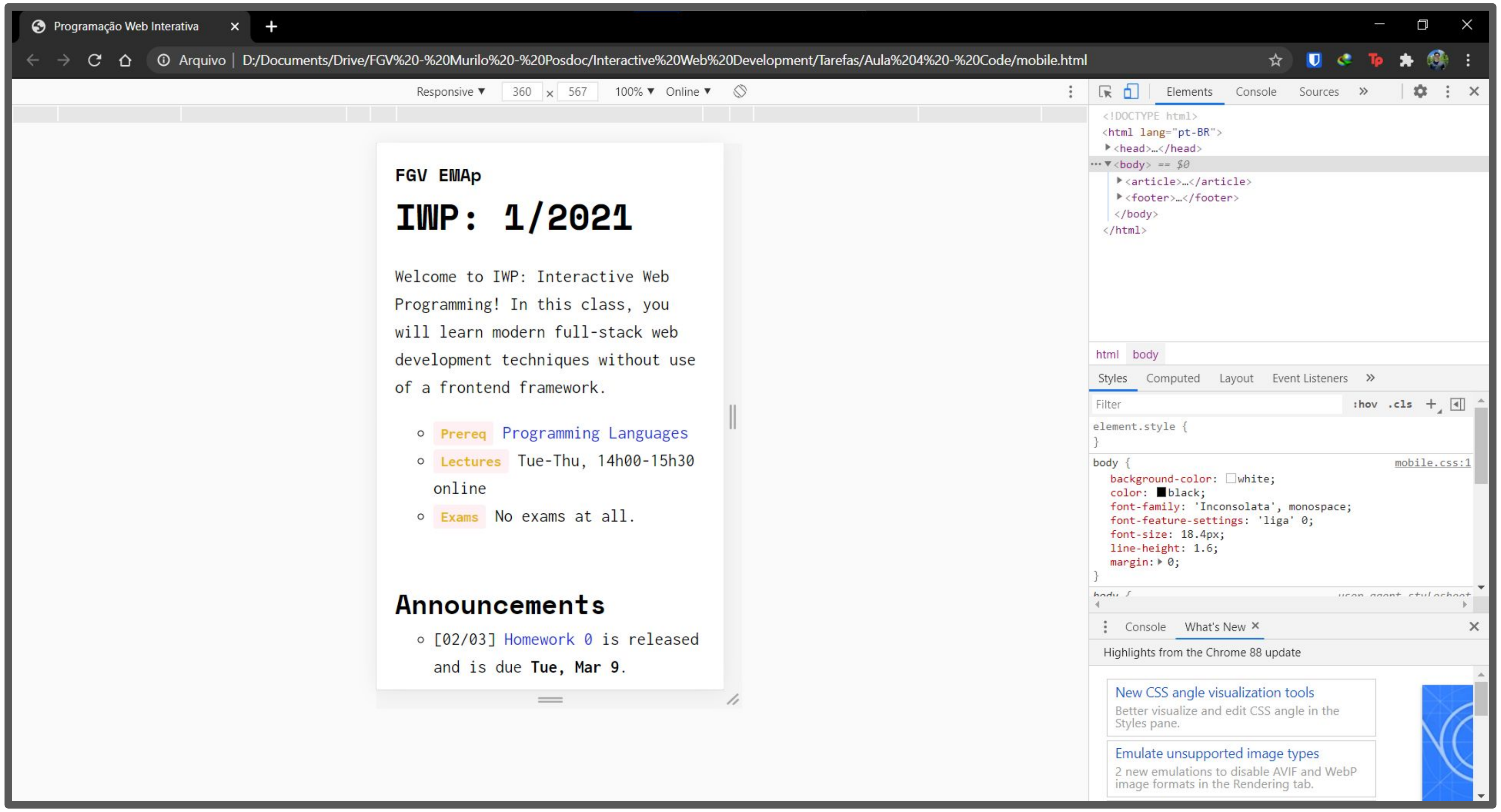
Highlights from the Chrome 88 update

New CSS angle visualization tools
Better visualize and edit CSS angle in the Styles pane.

Emulate unsupported image types
2 new emulations to disable AVIF and WebP image formats in the Rendering tab.

Chrome device mode

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



Programação Web Interativa
Arquivo | D:/Documents/Drive/FGV%20-%20Murilo%20-%20Posdoc/Interactive%20Web%20Development/Tarefas/Aula%204%20-%20Code/mobile.html
Responsive ▼ 360 × 567 100% ▼ Online ▼
Elements Console Sources
html body
Styles Computed Layout Event Listeners
Filter :hov .cls
Console What's New

FGV EMAp

IWP: 1/2021

Welcome to IWP: Interactive Web Programming! In this class, you will learn modern full-stack web development techniques without use of a frontend framework.

- Prereq Programming Languages
- Lectures Tue-Thu, 14h00-15h30 online
- Exams No exams at all.

Announcements

- [02/03] Homework 0 is released and is due **Tue, Mar 9**.

```
<!DOCTYPE html>
<html lang="pt-BR">
 ▶<head>…</head>
…▼<body> == $0
   ▶<article>…</article>
   ▶<footer>…</footer>
  </body>
</html>
```

```
element.style {
}
body {                                          mobile.css:1
   background-color: ☐white;
   color: ■black;
   font-family: 'Inconsolata', monospace;
   font-feature-settings: 'liga' 0;
   font-size: 18.4px;
   line-height: 1.6;
   margin: ▶ 0;
}
```

Highlights from the Chrome 88 update

New CSS angle visualization tools
Better visualize and edit CSS angle in the Styles pane.

Emulate unsupported image types
2 new emulations to disable AVIF and WebP image formats in the Rendering tab.

Chrome device mode

Advantages of Chrome device mode:

- Super convenient
- Mostly accurate

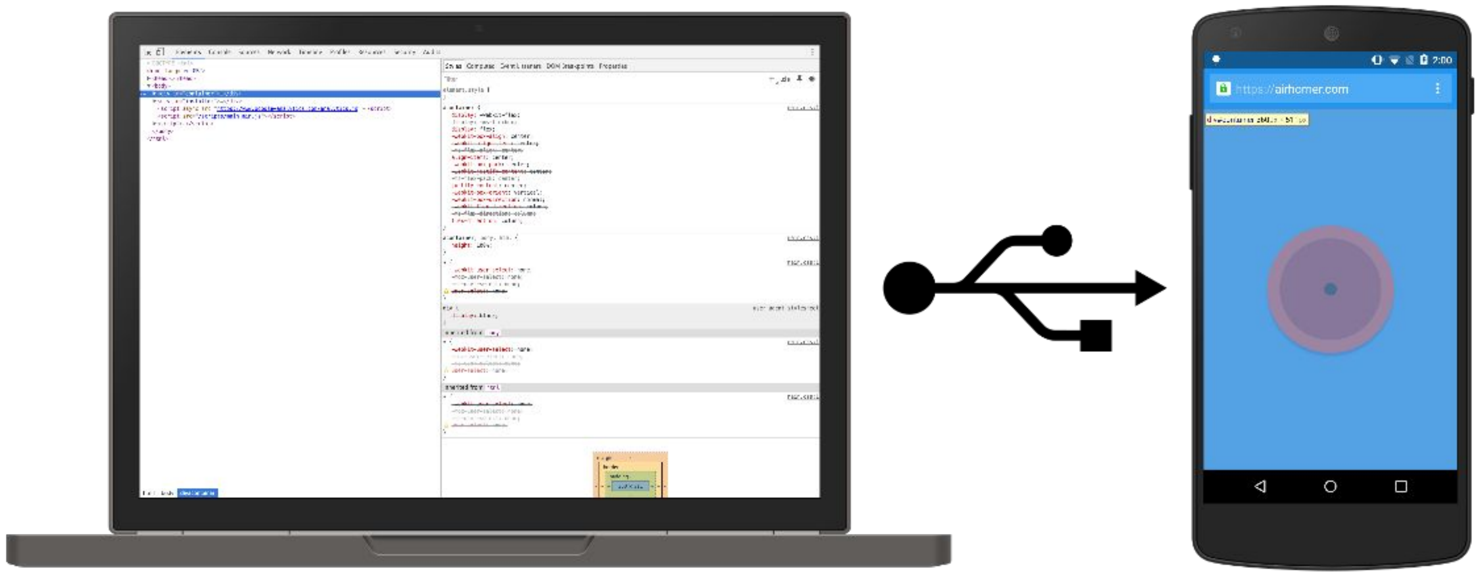
Disadvantages of Chrome device mode:

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

You should always test on real devices, too.

Chrome remote debugging

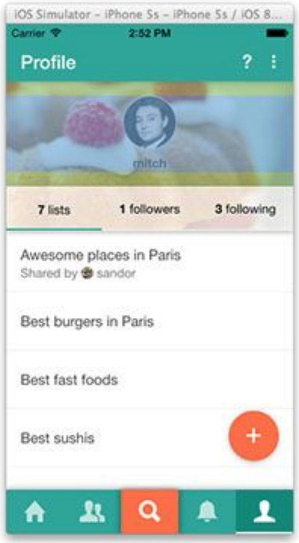
If you have an Android phone, you can debug web pages on your phone via Chrome remote debugging.

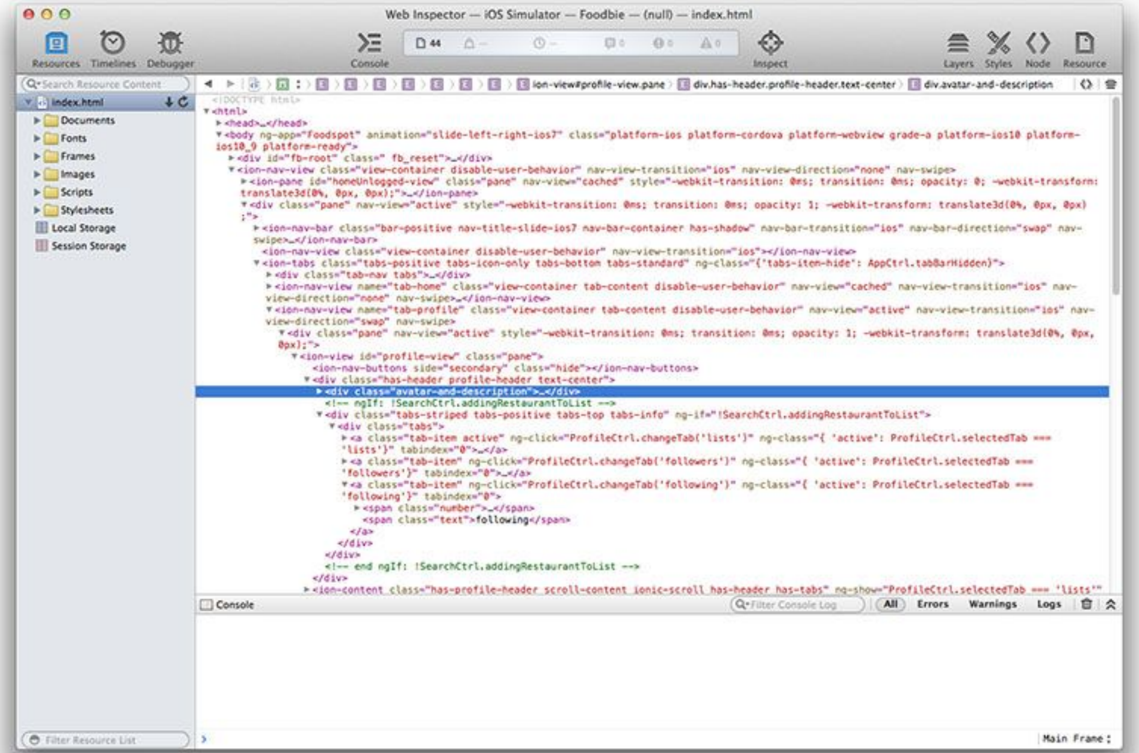


(You can also load a server running locally on your laptop, on your phone via port forwarding. But we haven't talked about servers yet.)

Safari remote debugging

If you have an iPhone, you can debug web pages on your phone via [Safari remote debugging]().





```
<!DOCTYPE html>
▼<html>
  ▶<head>…</head>
  ▼<body ng-app="Foodspot" animation="slide-left-right-ios7" class="platform-ios platform-cordova platform-webview grade-a platform-ios10 platform-
  ios10_9 platform-ready">
    ▶<div id="fb-root" class=" fb_reset">…</div>
    ▼<ion-nav-view class="view-container disable-user-behavior" nav-view-transition="ios" nav-view-direction="none" nav-swipe>
      ▶<ion-pane id="homeUnlogged-view" class="pane" nav-view="cached" style="-webkit-transition: 0ms; transition: 0ms; opacity: 0; -webkit-transform:
      translate3d(0%, 0px, 0px);">…</ion-pane>
      ▼<div class="pane" nav-view="active" style="-webkit-transition: 0ms; transition: 0ms; opacity: 1; -webkit-transform: translate3d(0%, 0px, 0px)
      ;">
        ▶<ion-nav-bar class="bar-positive nav-title-slide-ios7 nav-bar-container has-shadow" nav-bar-transition="ios" nav-bar-direction="swap" nav-
        swipe>…</ion-nav-bar>
         <ion-nav-view class="view-container disable-user-behavior" nav-view-transition="ios"></ion-nav-view>
        ▼<ion-tabs class="tabs-positive tabs-icon-only tabs-bottom tabs-standard" ng-class="{'tabs-item-hide': AppCtrl.tabBarHidden}">
          ▶<div class="tab-nav tabs">…</div>
          ▶<ion-nav-view name="tab-home" class="view-container tab-content disable-user-behavior" nav-view="cached" nav-view-transition="ios" nav-
          view-direction="none" nav-swipe>…</ion-nav-view>
          ▼<ion-nav-view name="tab-profile" class="view-container tab-content disable-user-behavior" nav-view="active" nav-view-transition="ios" nav-
          view-direction="swap" nav-swipe>
            ▼<div class="pane" nav-view="active" style="-webkit-transition: 0ms; transition: 0ms; opacity: 1; -webkit-transform: translate3d(0%, 0px,
            0px);">
              ▼<ion-view id="profile-view" class="pane">
                 <ion-nav-buttons side="secondary" class="hide"></ion-nav-buttons>
                ▼<div class="has-header profile-header text-center">
                  ▶<div class="avatar-and-description">…</div>
                   <!-- ngIf: !SearchCtrl.addingRestaurantToList -->
                  ▼<div class="tabs-striped tabs-positive tabs-top tabs-info" ng-if="!SearchCtrl.addingRestaurantToList">
                    ▼<div class="tabs">
                      ▶<a class="tab-item active" ng-click="ProfileCtrl.changeTab('lists')" ng-class="{ 'active': ProfileCtrl.selectedTab ===
                      'lists'}" tabindex="0">…</a>
                      ▶<a class="tab-item" ng-click="ProfileCtrl.changeTab('followers')" ng-class="{ 'active': ProfileCtrl.selectedTab ===
                      'followers'}" tabindex="0">…</a>
                      ▼<a class="tab-item" ng-click="ProfileCtrl.changeTab('following')" ng-class="{ 'active': ProfileCtrl.selectedTab ===
                      'following'}" tabindex="0">
                        ▶<span class="number">…</span>
                         <span class="text">following</span>
                       </a>
                     </div>
                   </div>
                   <!-- end ngIf: !SearchCtrl.addingRestaurantToList -->
                 </div>
                ▶<ion-content class="has-profile-header scroll-content ionic-scroll has-header has-tabs" ng-show="ProfileCtrl.selectedTab === 'lists'"
```

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).

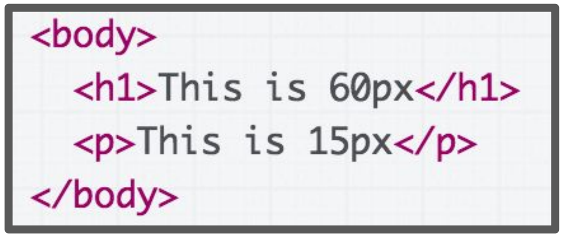
Advantages:

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

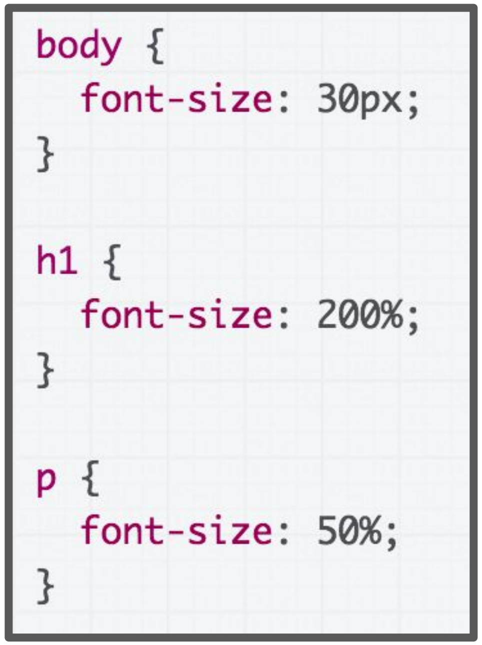
Q: Should we be using relative units on `font-size`?

Relative font sizes: percent

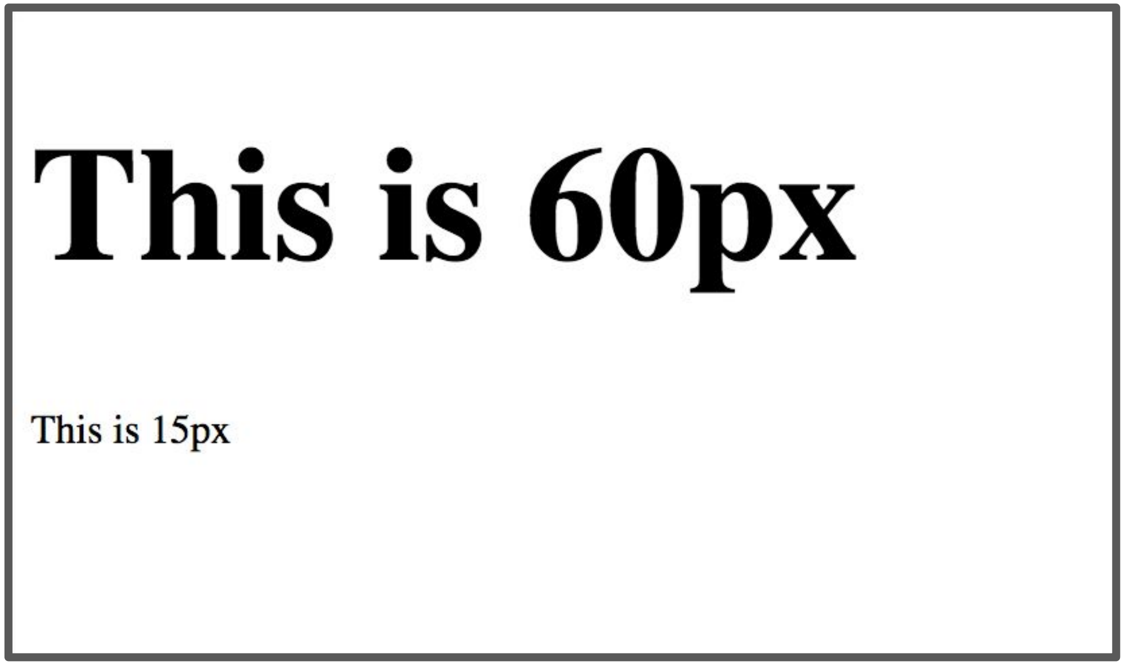
You can define font sizes in terms of percentage:



```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%;
}
```

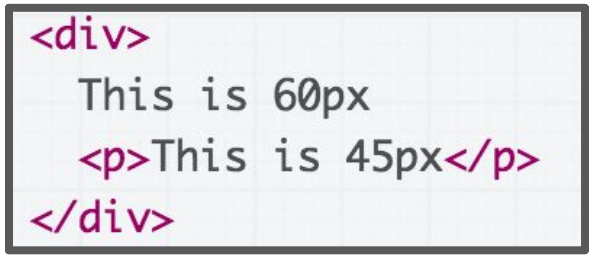


This is 60px

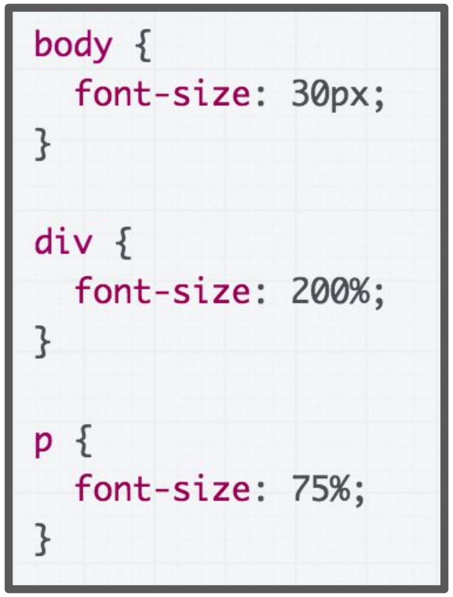
This is 15px

Relative font sizes: percent

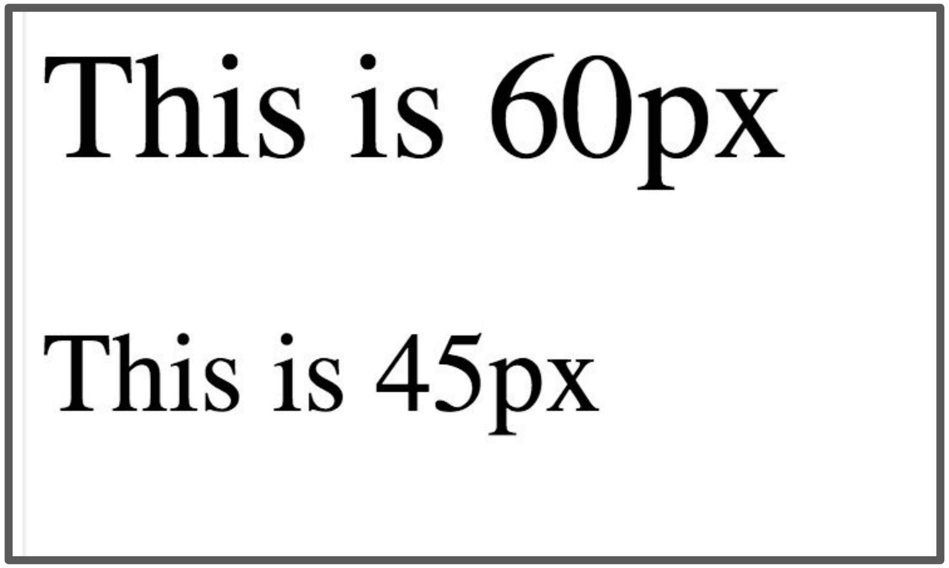
Percent on font-size behaves exactly like percentage on width and height, in that it's relative to the parent:



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



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

div {
  font-size: 200%;
}

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

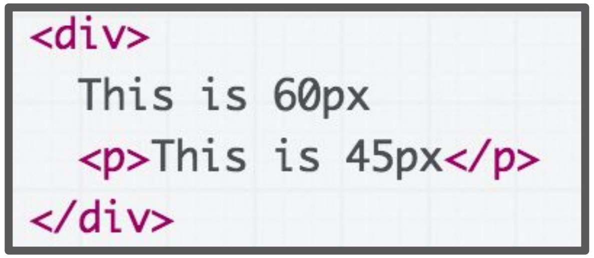


This is 60px

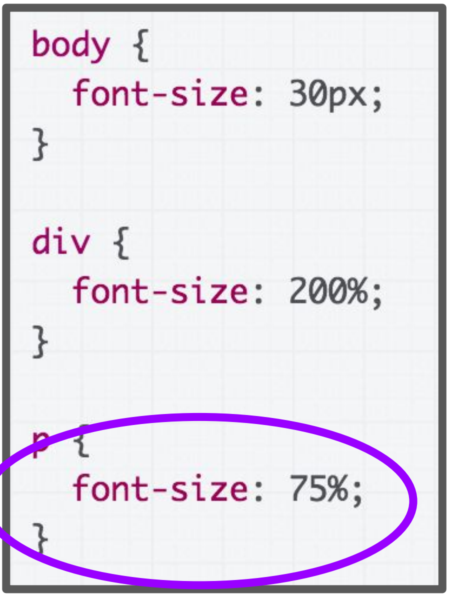
This is 45px

Relative font sizes: percent

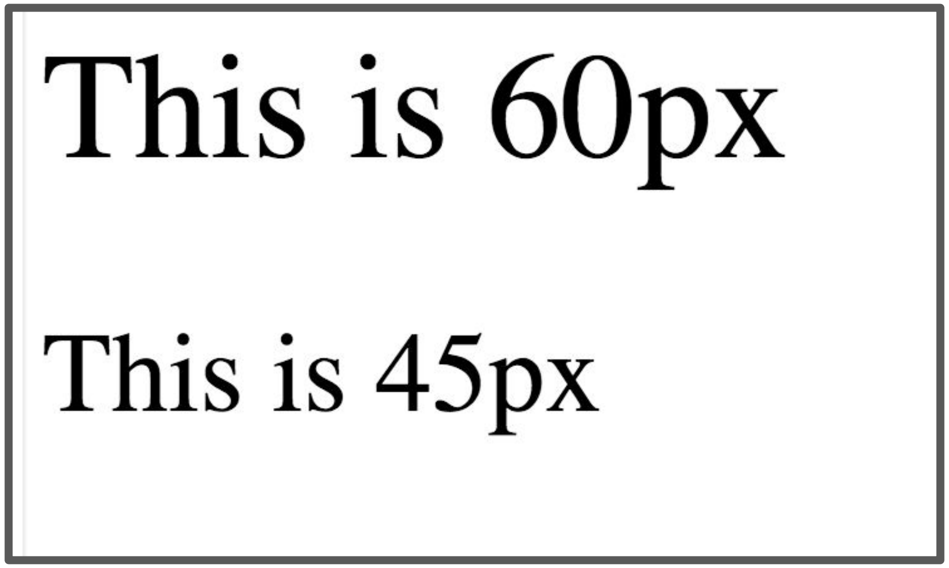
Percent on font-size behaves exactly like percentage on width and height, in that it's relative to the parent:



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



```css
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

Relative font sizes: `em`

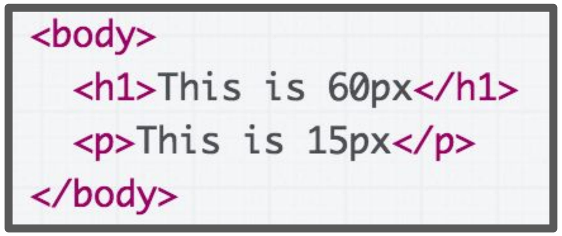
But instead of percentages, relative font sizes are usually defined in terms of `em`:

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

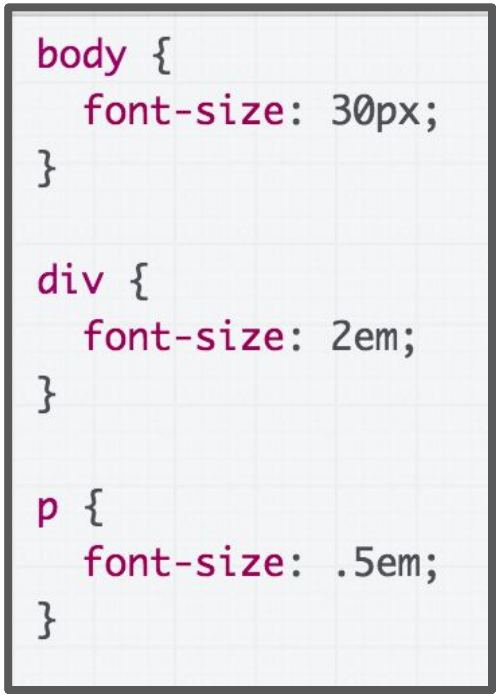
In other words,

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

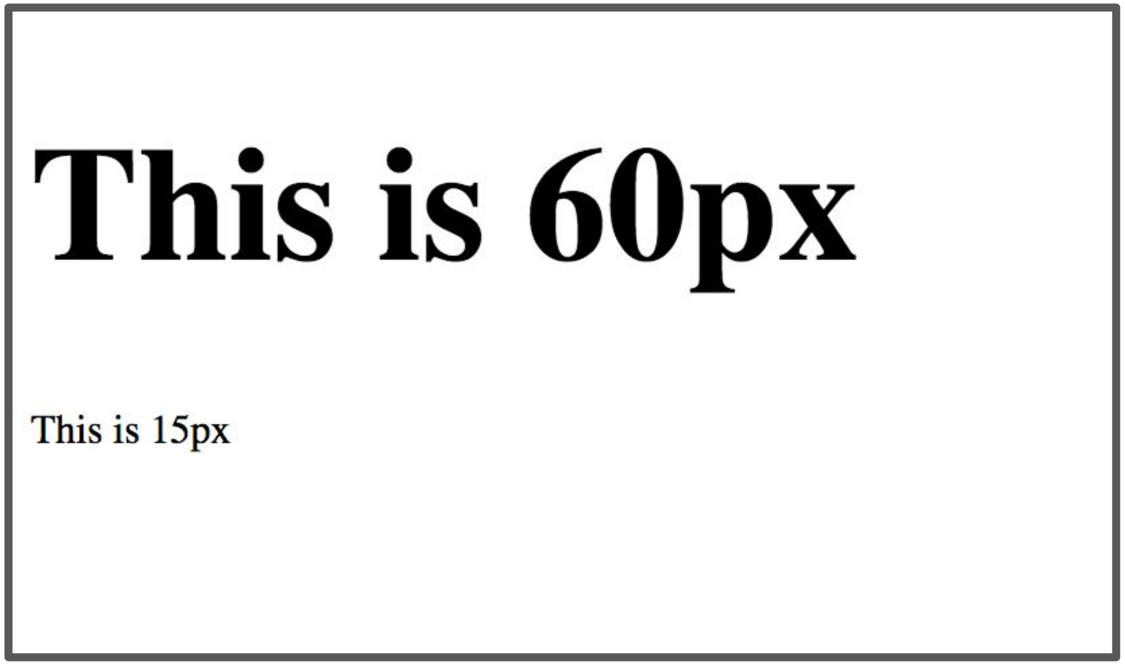
Relative font sizes: `em`



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



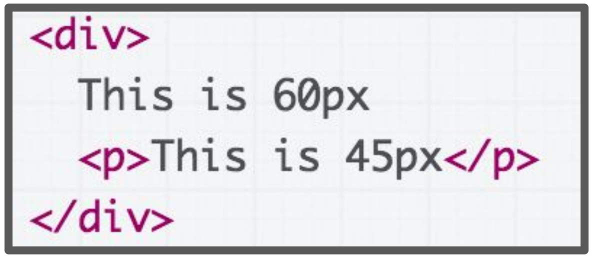
```css
body {
  font-size: 30px;
}

div {
  font-size: 2em;
}

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



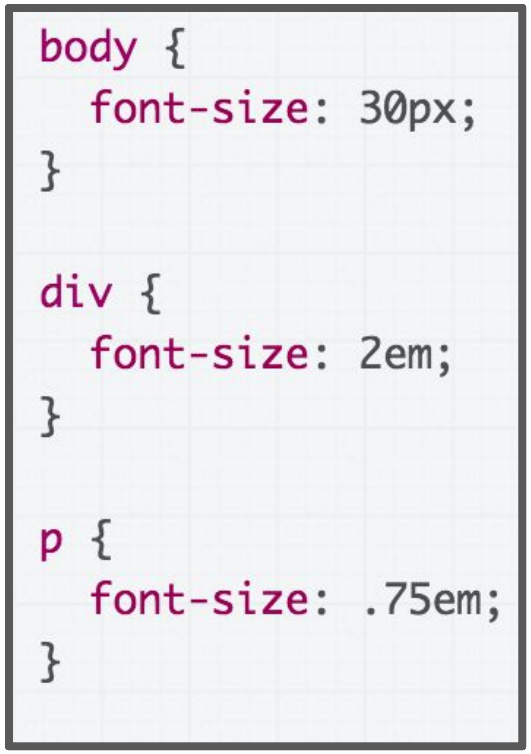
This is 60px

This is 15px

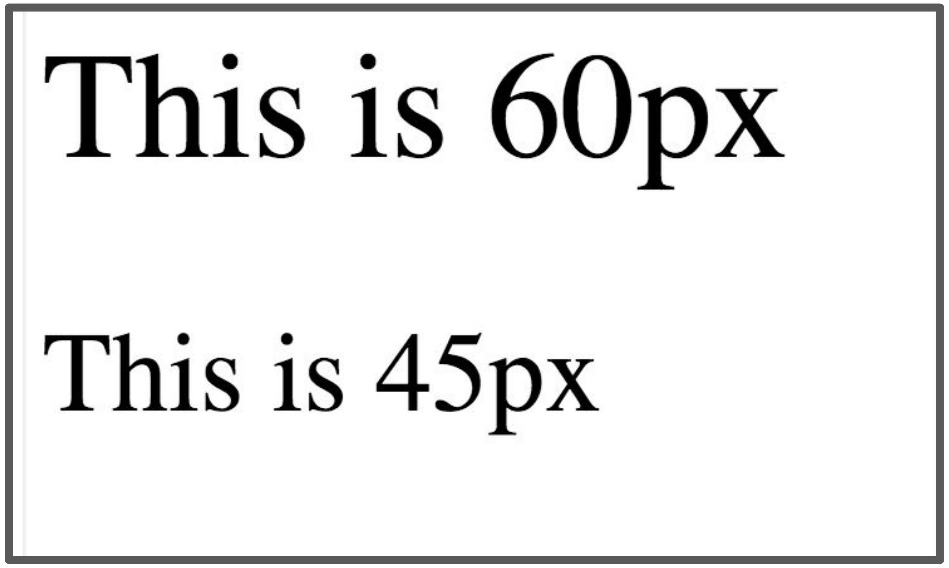
Relative font sizes: `em`



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



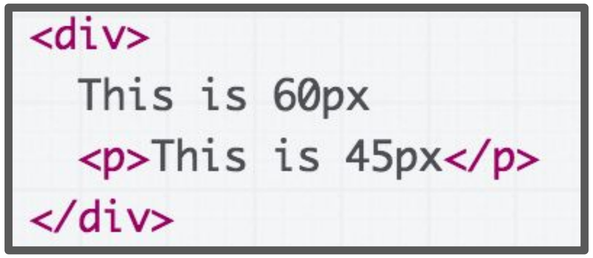
```
body {
  font-size: 30px;
}

div {
  font-size: 2em;
}

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



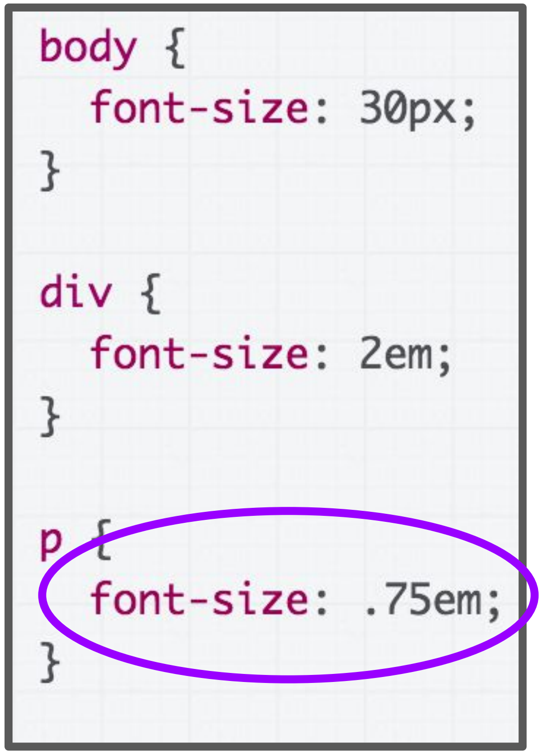
This is 60px

This is 45px

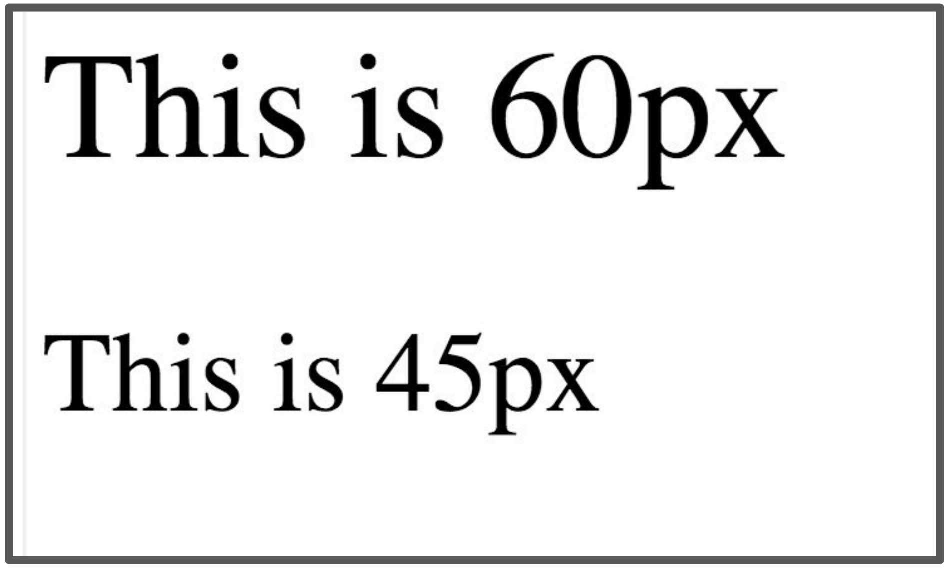
Relative font sizes: `em`



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



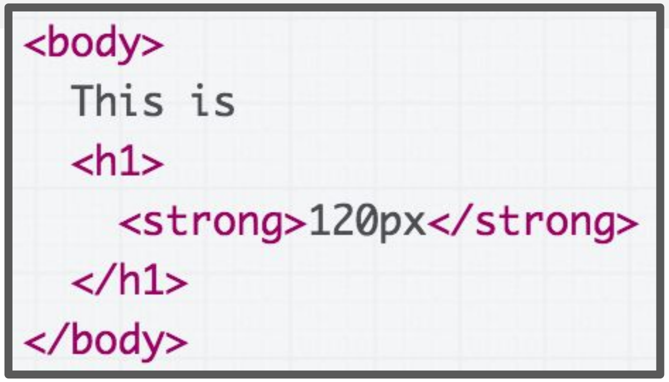
```
body {
  font-size: 30px;
}

div {
  font-size: 2em;
}

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



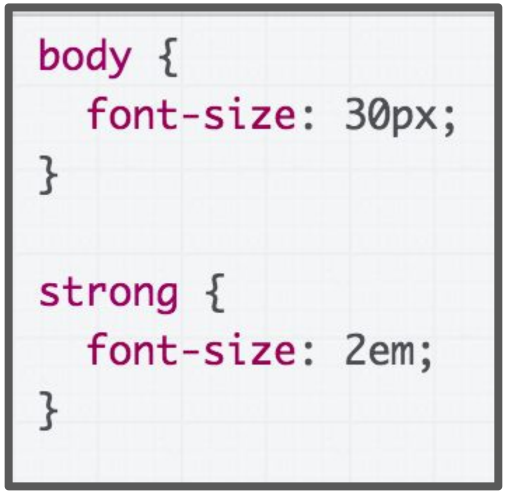
This is 60px

This is 45px

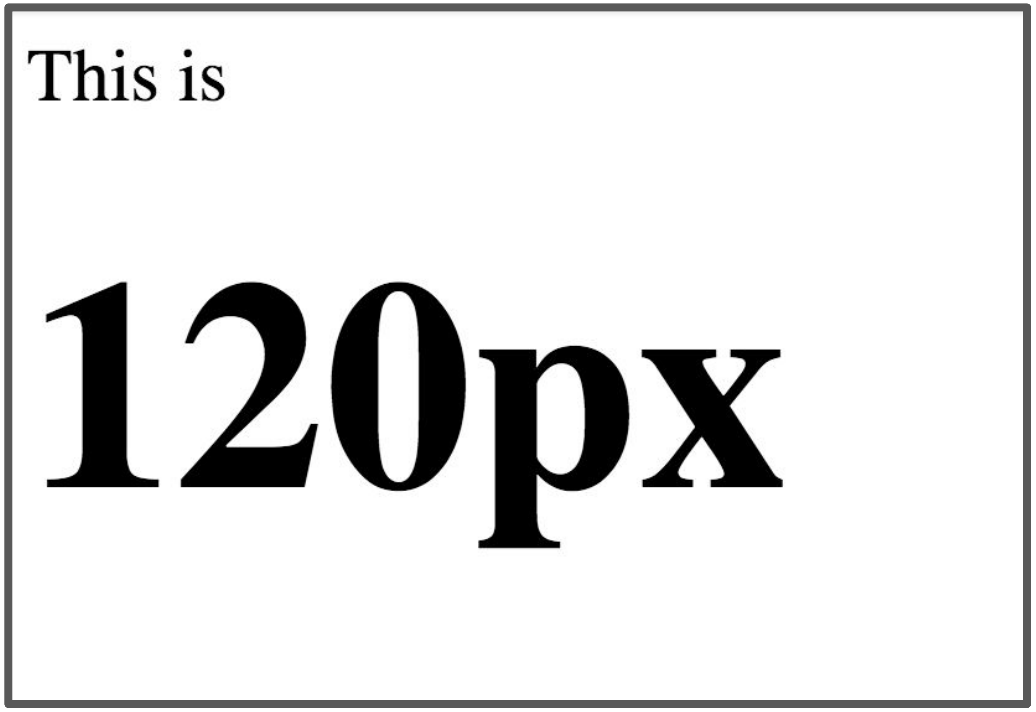
p's inherited font size is 2em, which is 60px. So 0.75em is 0.75*60 = 45px.



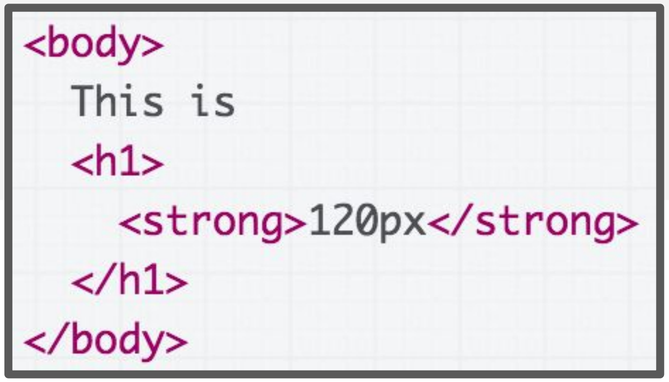
```
<body>
  This is
  <h1>
    <strong>120px</strong>
  </h1>
</body>
```



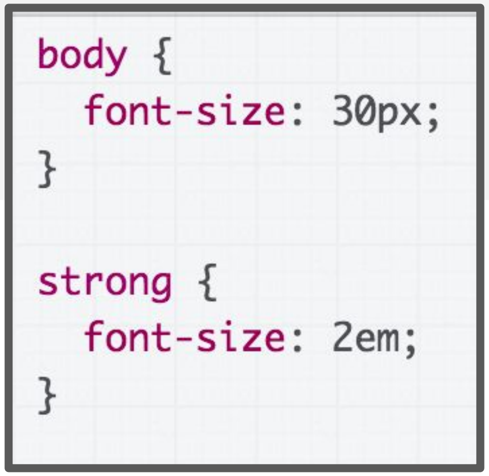
```
body {
  font-size: 30px;
}

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



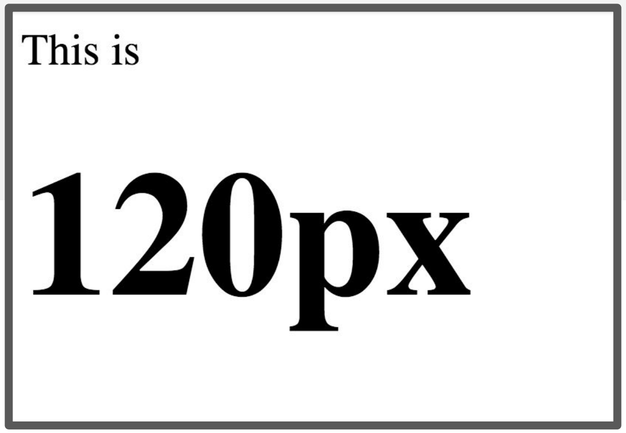
Wait, why is `<strong>` 120px and not 60px?



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

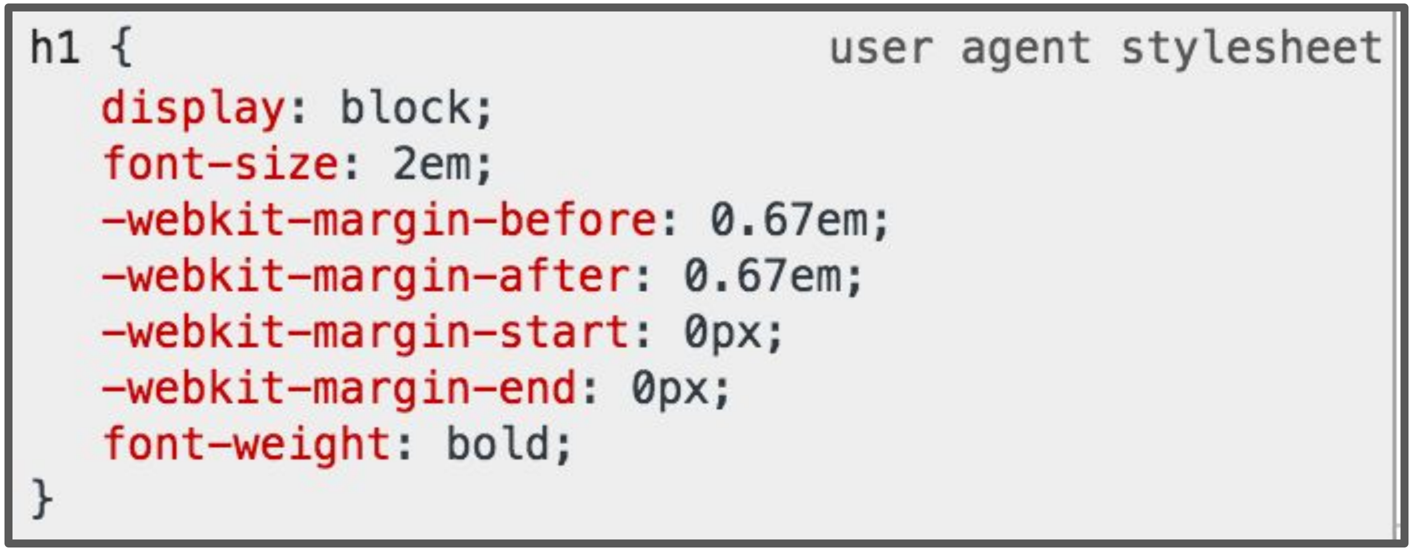


```
body {
  font-size: 30px;
}

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



This is

120px



```
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;
}
```

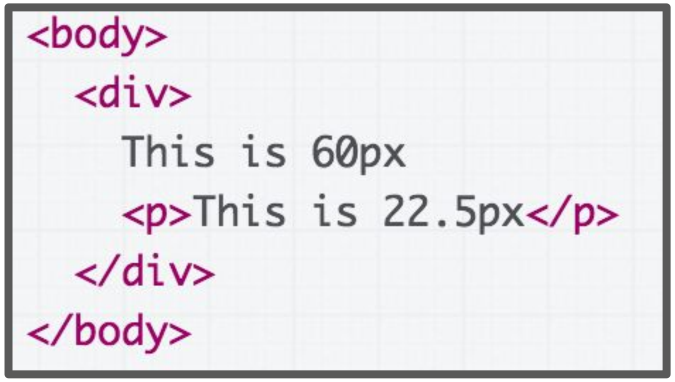
In the Chrome Inspector, we see the default browser `font-size` for h1 is 2em. So it's 30*2*2 = 120px.

Relative font sizes: **`rem`**

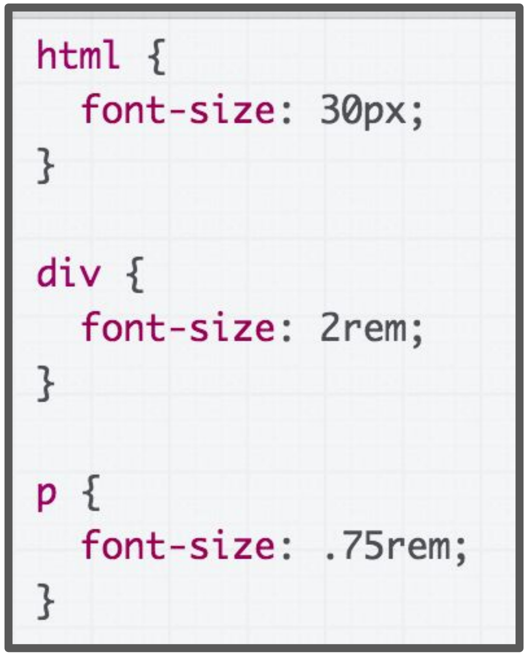
If you **do not** want your relative font sizes to compound through inheritance, use `rem`:

- `rem` represents the `font-size` of the <u>r</u>oot element
    - `1rem` = the root (`html` tag) font size
    - `2rem` = 2 times root font size

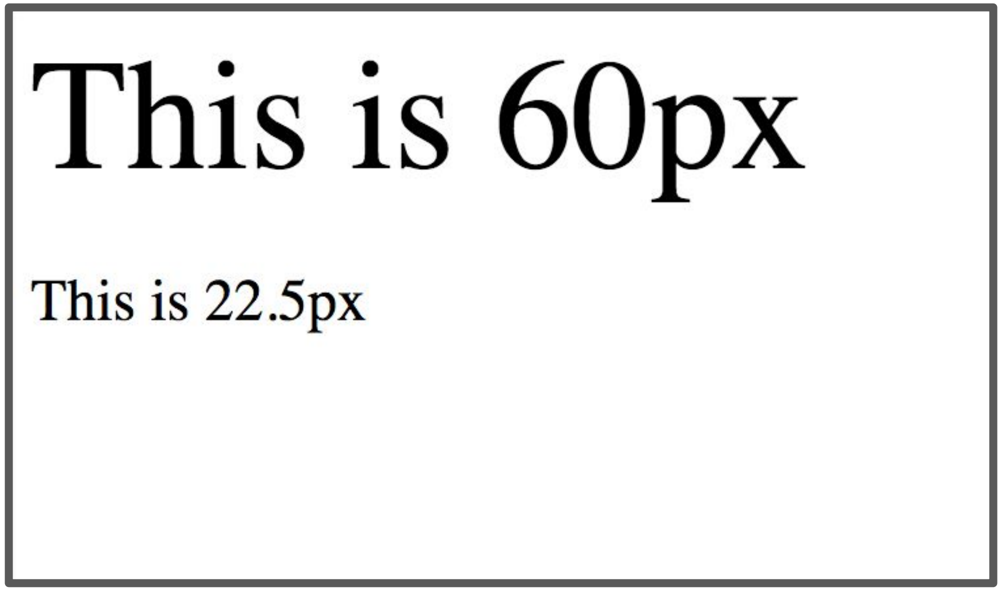
Relative font sizes: `rem`



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



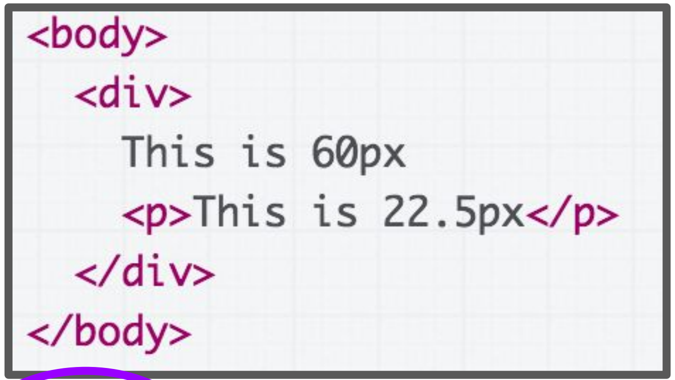
```css
html {
  font-size: 30px;
}

div {
  font-size: 2rem;
}

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



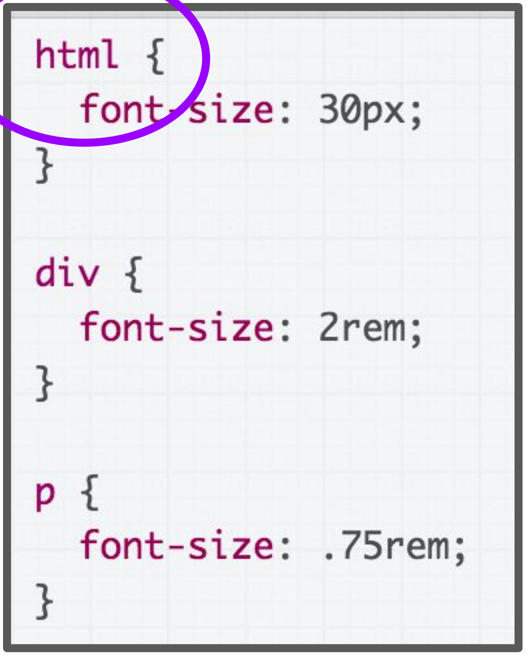
This is 60px

This is 22.5px

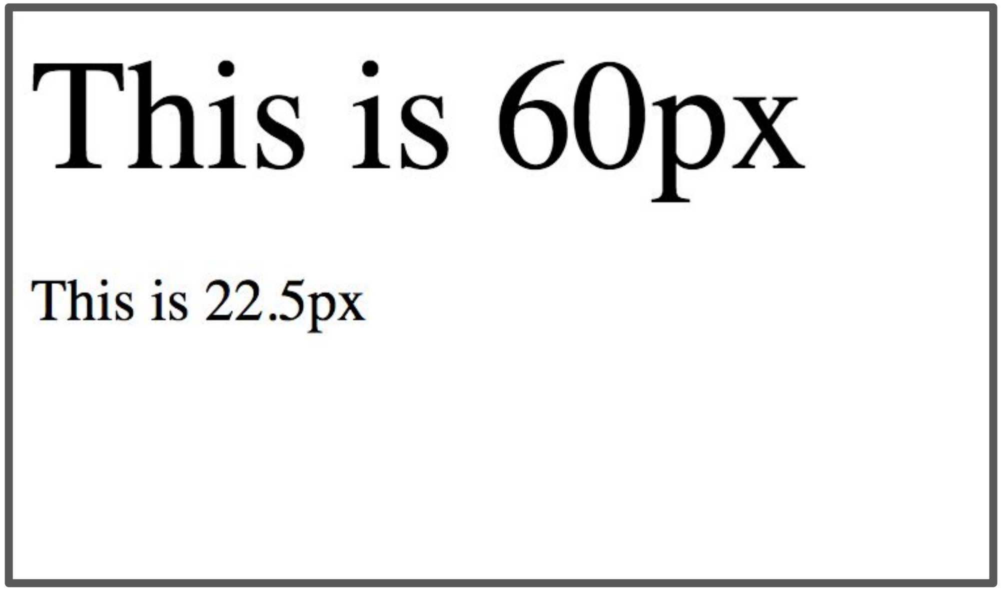
Relative font sizes: `rem`



```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;
}
```

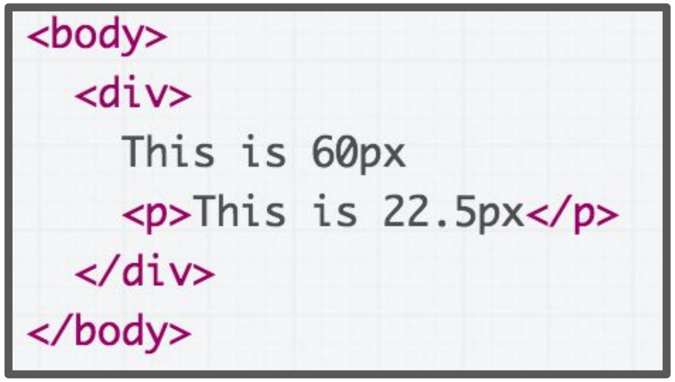


This is 60px

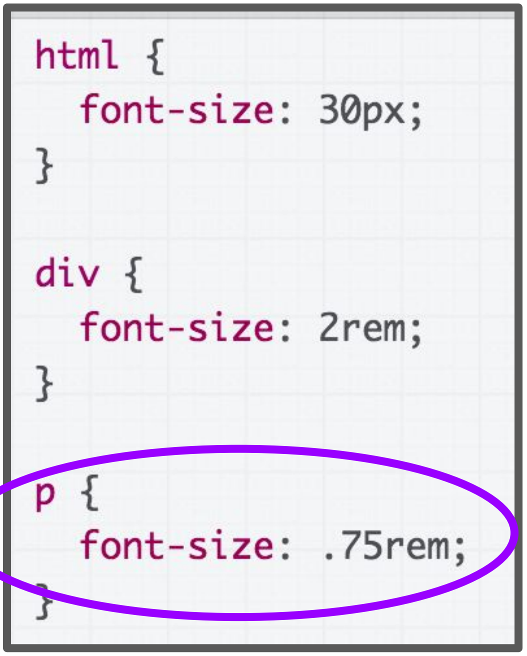
This is 22.5px

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

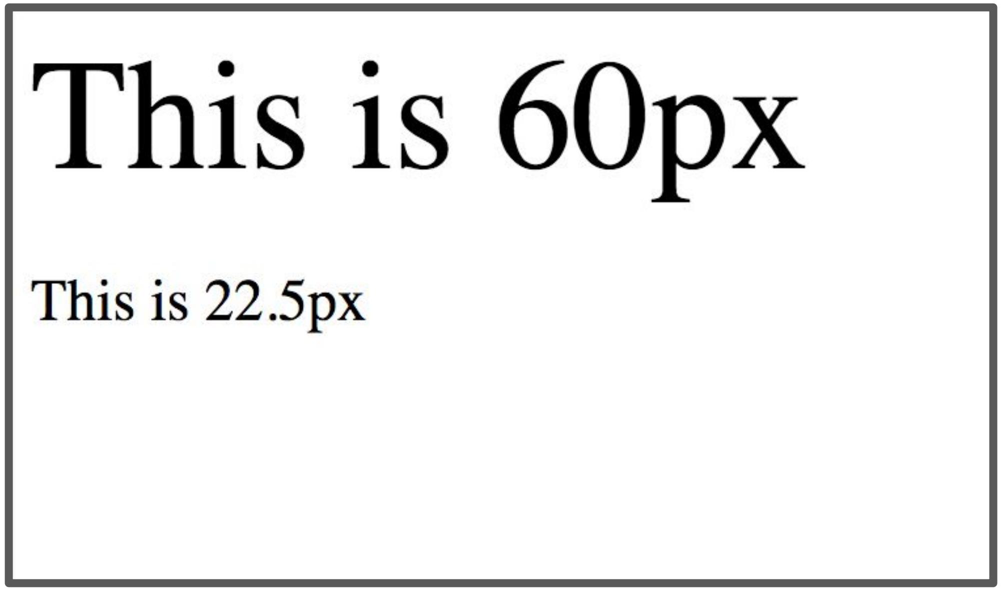
Relative font sizes: `rem`



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



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

div {
  font-size: 2rem;
}

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



This is 60px

This is 22.5px

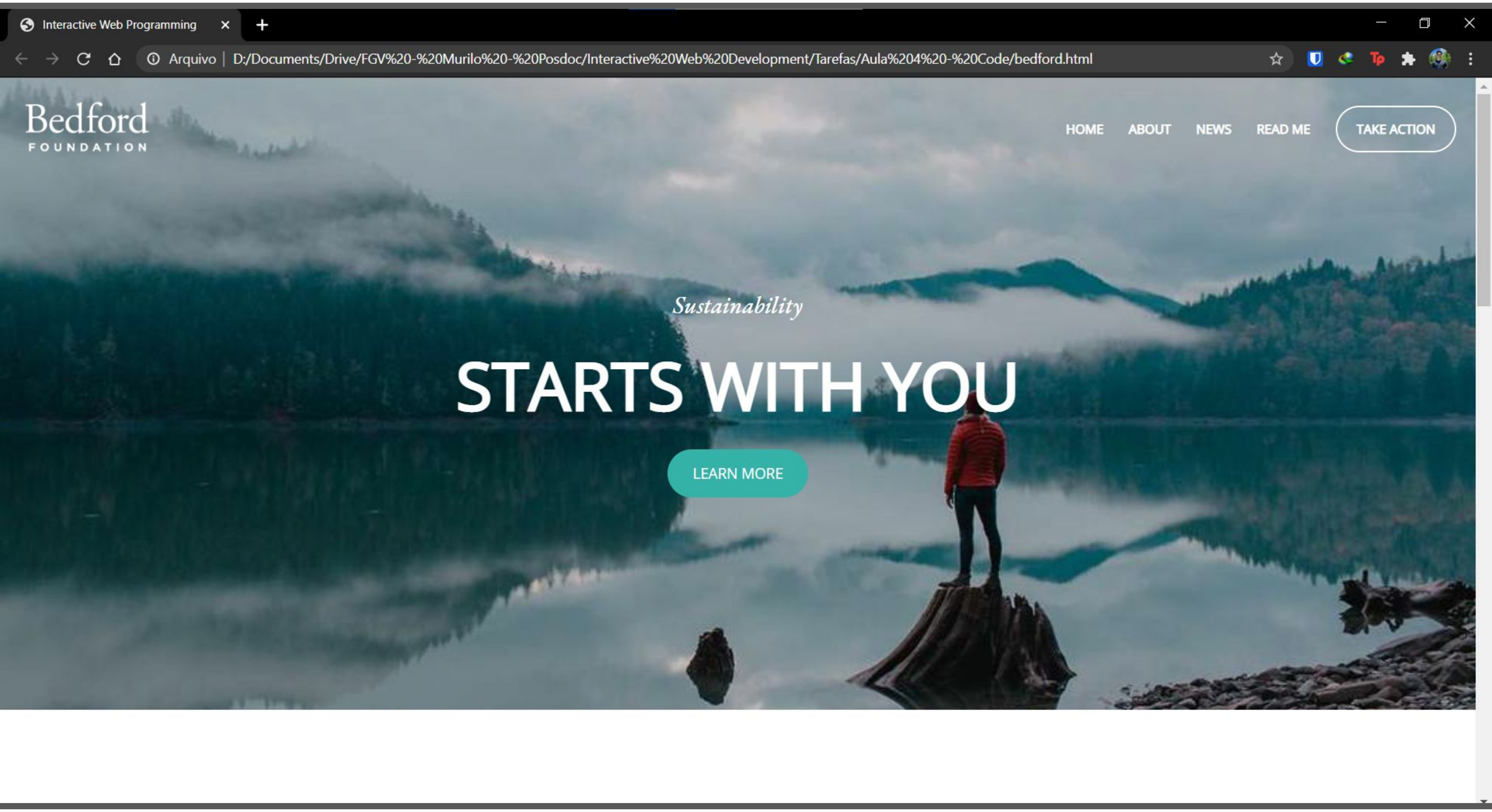
`.75em` is calculated from the root, which is 30px, so 30*.75 = 22.5px.

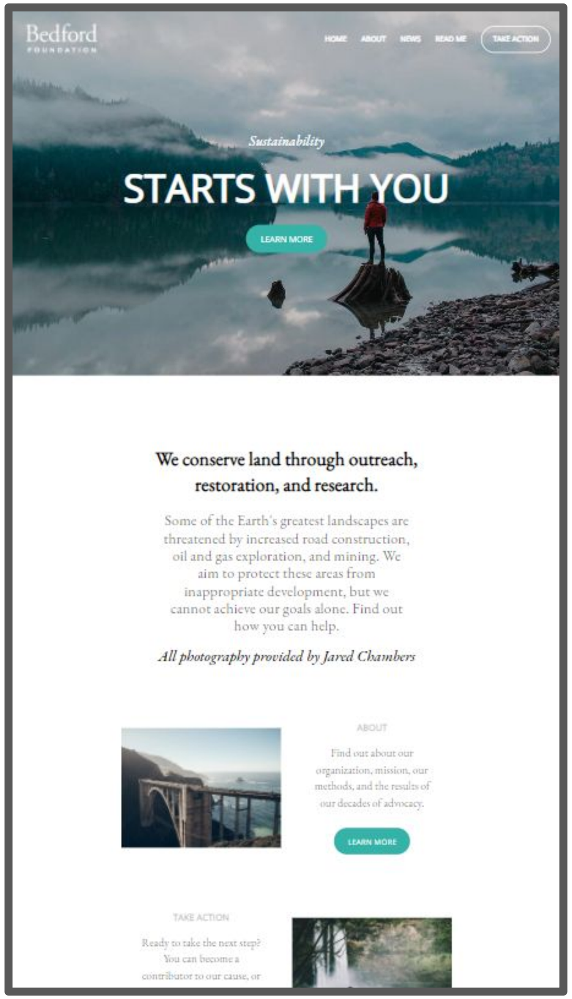
Relative font conclusions

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?


Interactive Web Programming
Arquivo
D:/Documents/Drive/FGV%20-%20Murilo%20-%20Posdoc/Interactive%20Web%20Development/Tarefas/Aula%204%20-%20Code/bedford.html
Bedford
FOUNDATION
HOME
ABOUT
NEWS
READ ME
TAKE ACTION
Sustainability
STARTS WITH YOU
LEARN MORE



We conserve land through outreach, restoration, and research.

Some of the Earth's greatest landscapes are threatened by increased road construction, oil and gas exploration, and mining. We aim to protect these areas from inappropriate development, but we cannot achieve our goals alone. Find out how you can help.

All photography provided by Jared Chambers

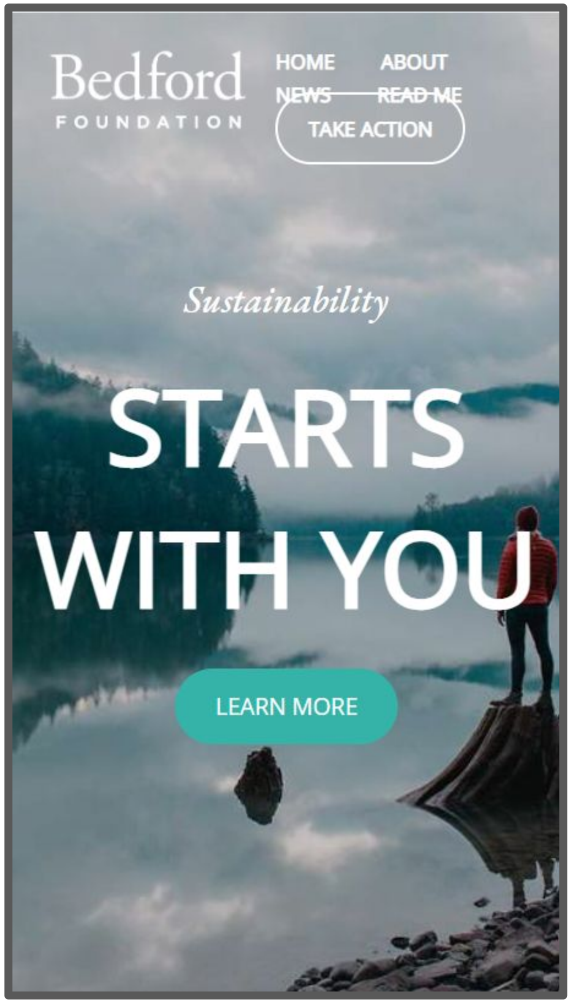
ABOUT

Find out about our organization, mission, our methods, and the results of our decades of advocacy.

LEARN MORE

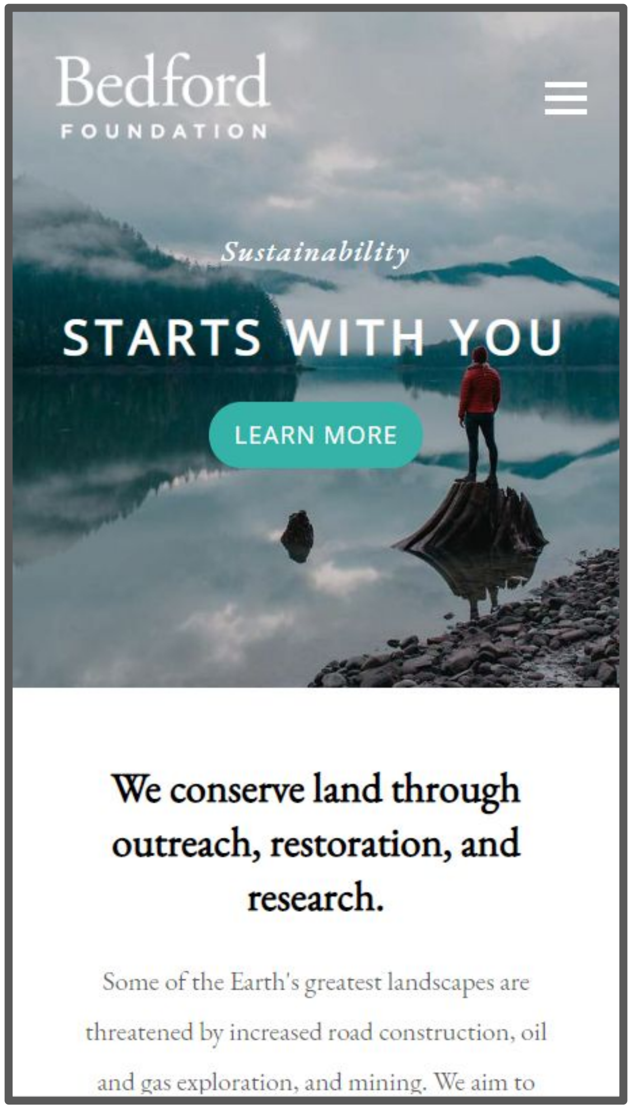
TAKE ACTION

Ready to take the next step? You can become a contributor to our cause, or



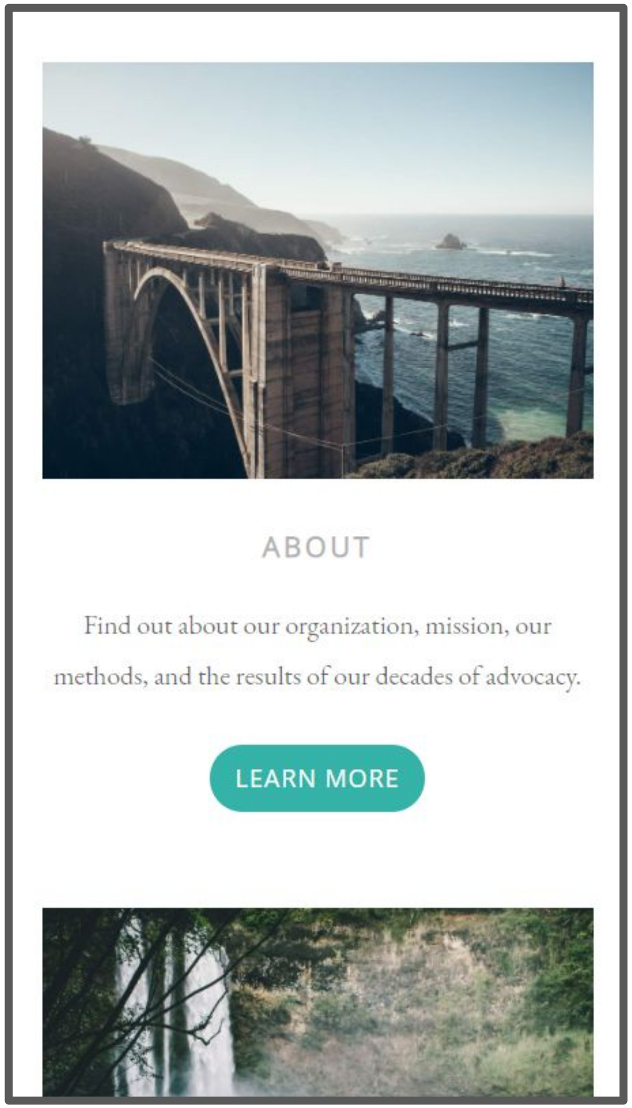
Without the meta viewport tag

With the meta viewport tag



We conserve land through outreach, restoration, and research.

Some of the Earth's greatest landscapes are threatened by increased road construction, oil and gas exploration, and mining. We aim to



Find out about our organization, mission, our methods, and the results of our decades of advocacy.

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`

More on responsive web design

Random useful CSS

`calc`

You can use the `calc` CSS function to define numeric values in terms of expressions:

```
width: calc(50% - 10px);
width: calc(100% / 6);
```

(MDN details of calc)

CSS variables

Variables are a brand-new CSS feature (caniuse).

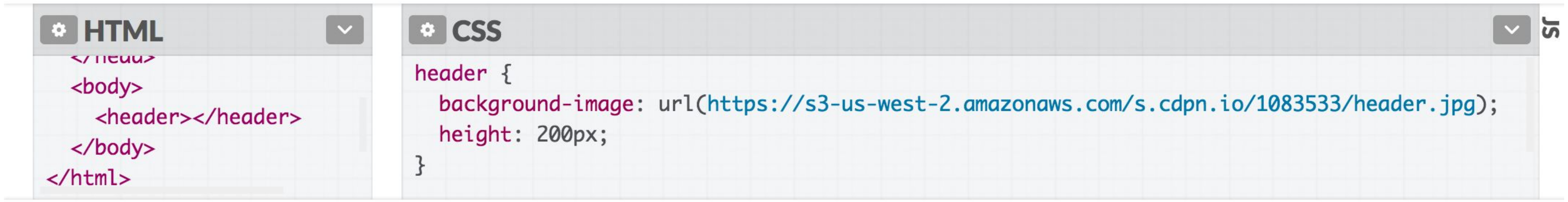
```
:root {
   --primary-color: hotpink;
}

h1 {
  background-color: var(--primary-color);
}
```

(MDN details of CSS variables)

background properties

An easy way to render images stretched and cropped to a given size: set it as a background image for an element.

```
background-image: url(background.png);
```



```
</head>
  <body>
    <header></header>
  </body>
</html>
```

```
header {
  background-image: url(https://s3-us-west-2.amazonaws.com/s.cdpn.io/1083533/header.jpg);
  height: 200px;
}
```

(CodePen)

background properties

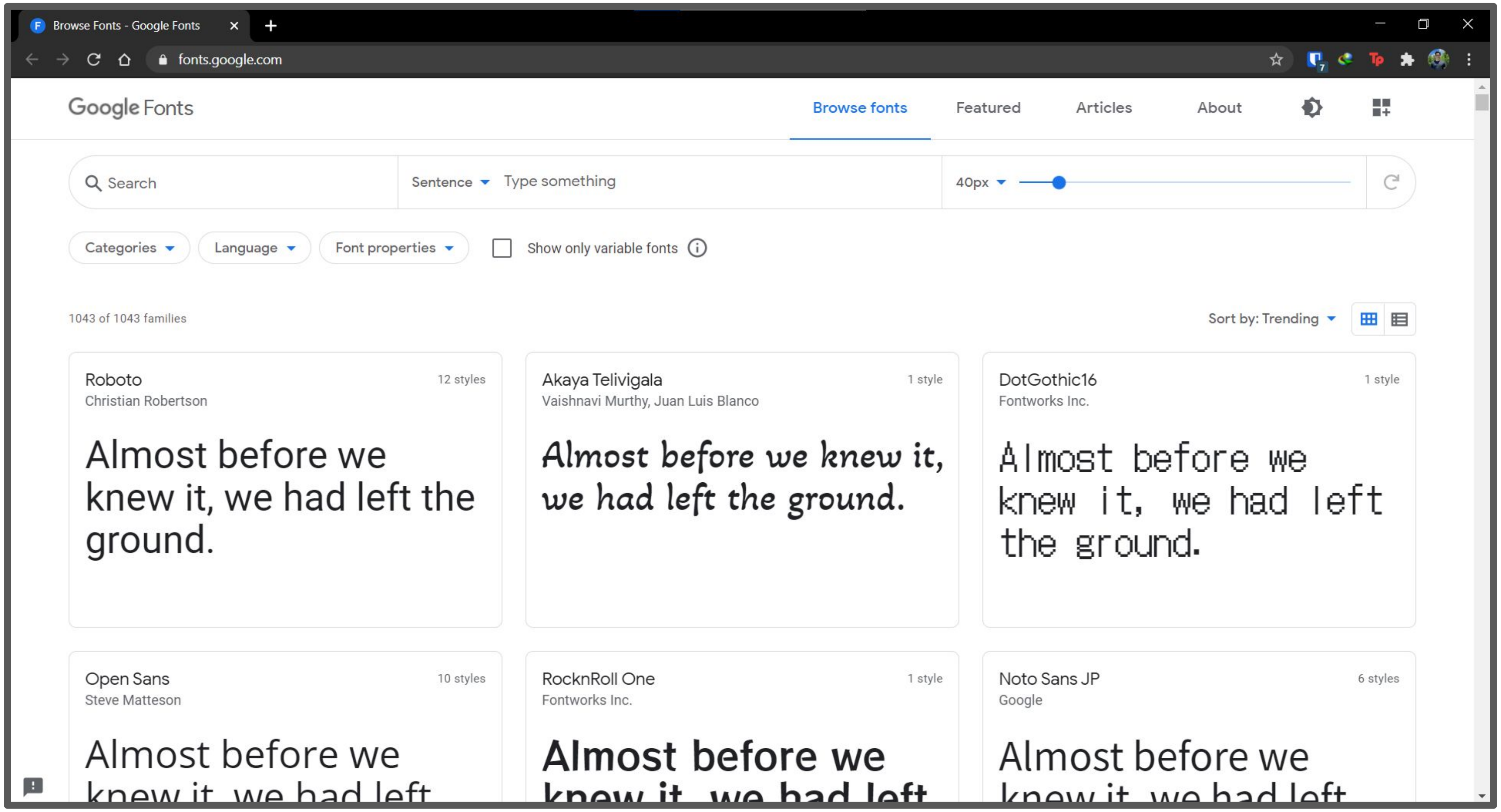
You can then use additional background properties to further style it:

```
background-size: cover;
background-size: contain;
background-repeat: no-repeat;
background-position: top;
background-position: center;
```

(CodePen: Try resizing the window)

Web Fonts

You can use Google Fonts to choose from better fonts:


Browse Fonts - Google Fonts

fonts.google.com

Google Fonts

Browse fonts Featured Articles About

Search Sentence Type something 40px

Categories Language Font properties Show only variable fonts

1043 of 1043 families

Sort by: Trending

Roboto 12 styles
Christian Robertson
Almost before we knew it, we had left the ground.

Akaya Telivigala 1 style
Vaishnavi Murthy, Juan Luis Blanco
Almost before we knew it, we had left the ground.

DotGothic16 1 style
Fontworks Inc.
Almost before we knew it, we had left the ground.

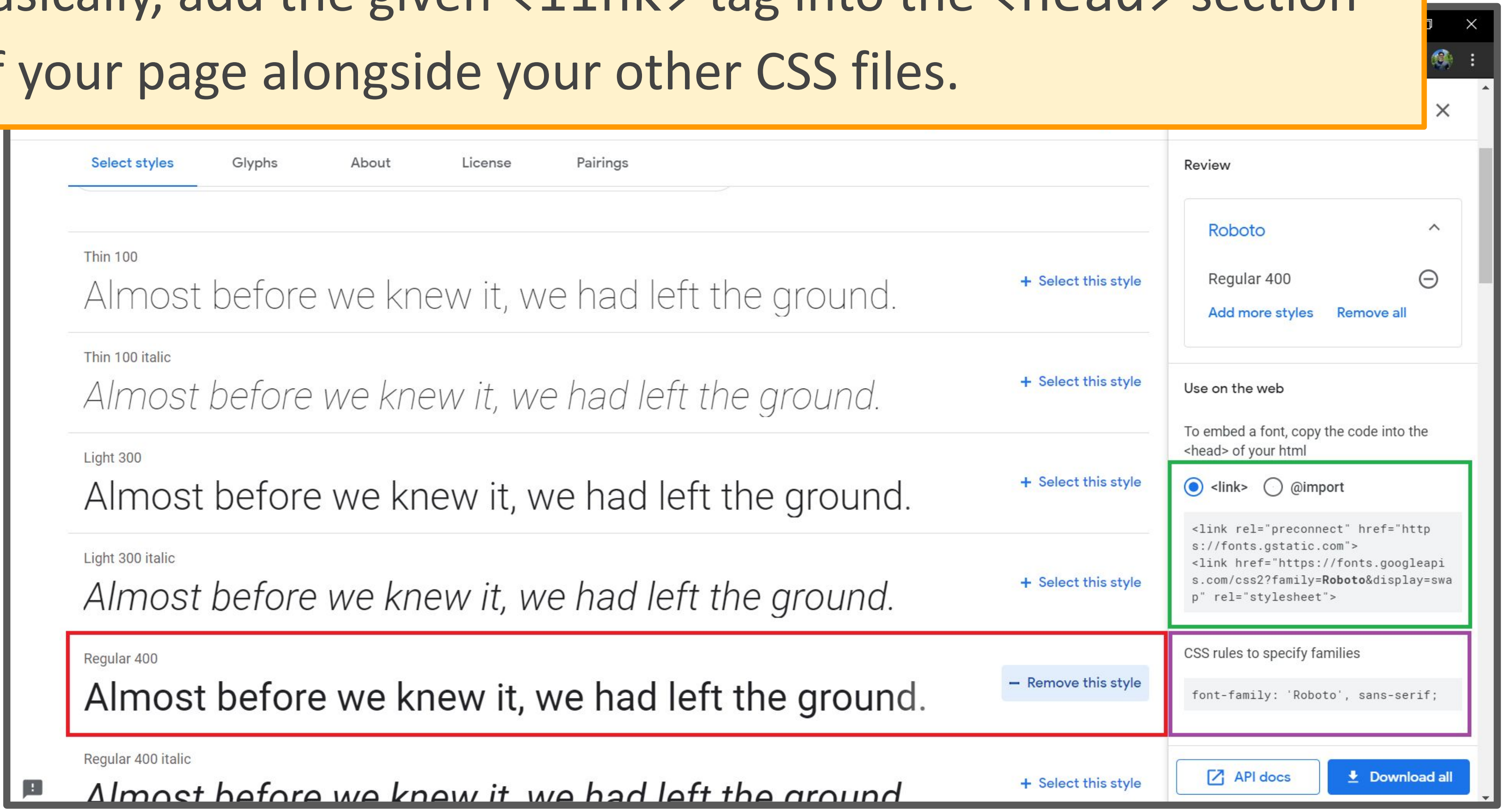
Open Sans 10 styles
Steve Matteson
Almost before we knew it, we had left

RocknRoll One 1 style
Fontworks Inc.
Almost before we knew it, we had left

Noto Sans JP 6 styles
Google
Almost before we knew it, we had left

Web Fonts

The instructions are pretty self-explanatory:
Basically, add the given `<link>` tag into the `<head>` section of your page alongside your other CSS files.



Thin 100
Almost before we knew it, we had left the ground.
+ Select this style

Thin 100 italic
Almost before we knew it, we had left the ground.
+ Select this style

Light 300
Almost before we knew it, we had left the ground.
+ Select this style

Light 300 italic
Almost before we knew it, we had left the ground.
+ Select this style

Regular 400
Almost before we knew it, we had left the ground.
– Remove this style

Regular 400 italic
Almost before we knew it, we had left the ground.
+ Select this style

Review

Roboto
Regular 400
Add more styles Remove all

Use on the web

To embed a font, copy the code into the <head> of your html

[✓ <link>] @import

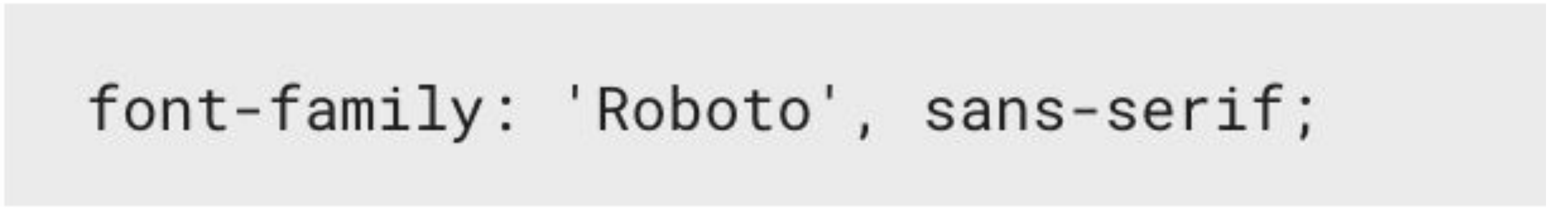
```
<link rel="preconnect" href="https://fonts.gstatic.com">
<link href="https://fonts.googleapis.com/css2?family=Roboto&display=swap" rel="stylesheet">
```

CSS rules to specify families

```
font-family: 'Roboto', sans-serif;
```

Aside: Fallback fonts

Notice that the Google Font example shows a comma-separated list of values for `font-family`:



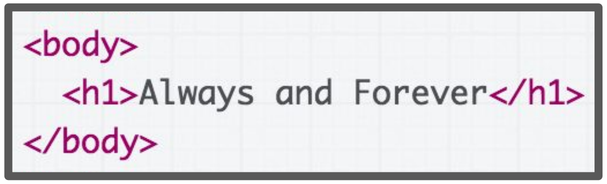
```
font-family: 'Roboto', sans-serif;
```

- Each successive font listed is a fallback, i.e. the font that will be loaded if the previous font could not be loaded
- There are also six generic font names, which allows the browser to choose the font based on intent + fonts available on the OS.
- It's good practice to list a generic font at the end of all your `font-family` declarations.

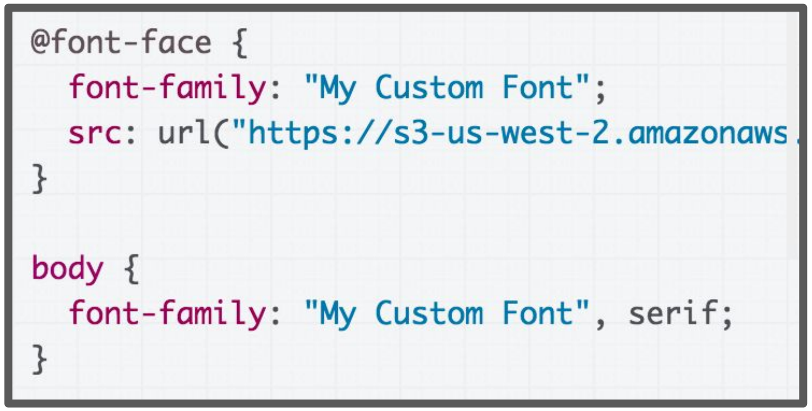
Hosted fonts with `@font-face`

You can also load your own font via [`@font-face`]:

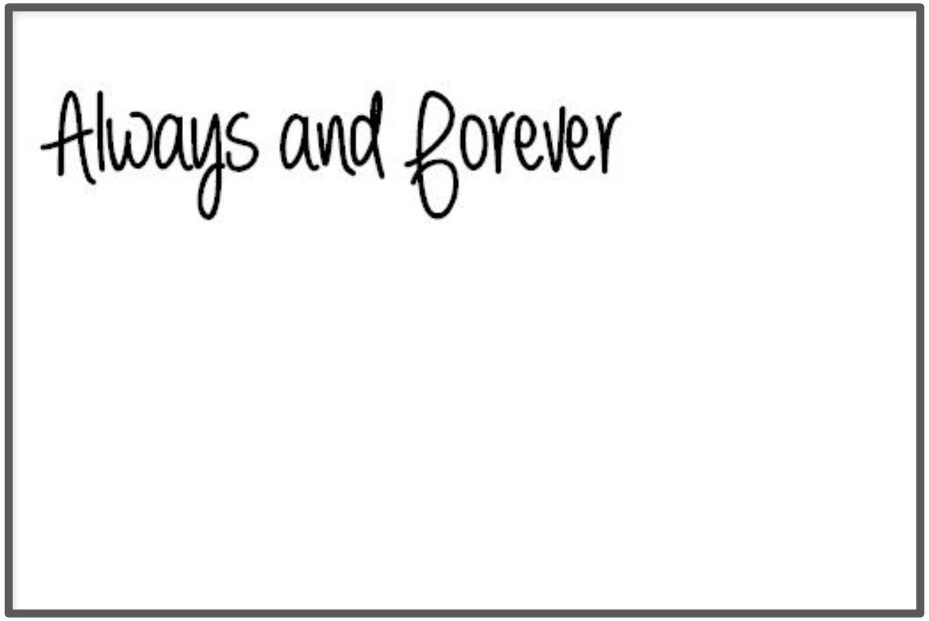
- Give it your own font name
- Link to where the font file is found



```
<body>
  <h1>Always and Forever</h1>
</body>
```



```
@font-face {
  font-family: "My Custom Font";
  src: url("https://s3-us-west-2.amazonaws.

}

body {
  font-family: "My Custom Font", serif;
}
```



Always and Forever

[CodePen]

CSS wrap-up

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

Later this semester:

- More flexbox patterns
- CSS animations
- Possibly `grid`