Interactive Web Programming

1st semester of 2021

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

Heavily based on **Victoria Kirst** slides

Today's schedule

- Syllabus
- Course Info
- Browsers
- A little bit about **HTML** and **CSS**

- Homework 0 assigned and due **this Tuesday 09/03**

Check out the course website for all this and more:
https://murilocamargos.github.io/iwp/

Syllabus

Course Goals

If you never take another web programming class again, you will leave this course with the following skills:

- Create **attractive, small scale web sites or apps** that at least mostly work on phones
- Have the **vocabulary and background knowledge** to understand technical writing/discussions about the web (e.g. web API documentation; random blog posts)
- Have the **foundation** to pursue the areas of web programming that you're interested in (if you choose)

(Course Non-goals)

It is **not** a class to take to learn how to code.

- Programming Languages is a prereq. It should be sufficient.

It is **not** a class that will turn you into a senior frontend/backend developer.

- Nor is any class; software takes years of experience to develop expertise.

It is **not** a class that will teach you all there is to know about web programming.

- For example, we will **not** teach how to support old browsers, legacy devices, etc.

The course, in detail

- **Frontend fundamentals (Client)**:
  - HTML
  - CSS
  - JavaScript
  - D3

- **Backend fundamentals (Server):**
  - Server on NodeJS + Express
  - Database via MongoDb and Mongoose

CSS

HTML (~1 day)

- Key concepts: inline, block, inline-block

CSS (~1.5 weeks)

- Multiple rendering styles: natural, flex, positioned, float
- Mobile layouts
- Transforms and animations (maybe)
- **FYI: No libraries or compiled CSS**

Modern JS / ES6+

Later in the quarter, we will read and write JavaScript that looks sort of like this:

```
(async () => {
  let choice = 'e';
  do {
    choice = await askQuestion('Enter choice');
    await processChoice(choice);
  } while (choice != 'e');
})();
```

Modern JS / ES6+

JavaScript (~5 weeks)

- JavaScript classes
- Relevant functional programming
  - Lambdas
  - Generator functions and async/await
  - "Fat arrow" vs function
  - Closures
- Creating and using Promises
- Understanding the Event Loop
- Modules and encapsulation

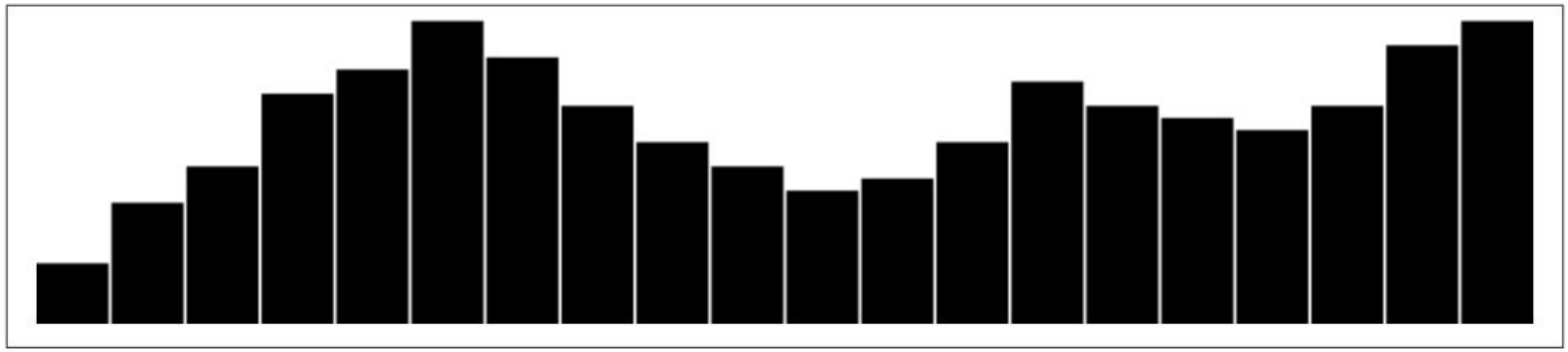
NO frontend framework; minimal libraries

No Angular/React/JQuery/etc

D3

D3 (~2 weeks)

- Fundamentals and Scalable Vector Graphics (SVG)
- Drawing with data
- Scales and Axes
- Updates, Transitions and Motion



Backend

The coverage of server-side programming will be light.

Backend stack:

NodeJS + Express + MongoDB via Mongoose (~4 weeks)

- What is a server
- What is npm
- How to serve static web pages
- How to server JSON via REST APIs
- Writing to and loading from a database
- Authentication via OAuth2 (i.e. login via Gmail account)

Course info

Disclaimer

This is the first ever offering of this course, meaning:

- **Everything is subject to change.** Including everything I've just told you and everything I'm about to tell you.

- **There will be all the mistakes of a new course!**
  - Bugs in homework
  - Awkward lectures
  - Things that are too hard / too easy

Please be patient with us! We are also soliciting your constructive feedback.

Course Structure

"Homework 0" + ~6 homeworks

- We'll create a web app throughout the course
- Each homework will increment this web app
- Each homework with have a multiple choice "mini-homework" attached to it
- **Individual** assignments; no pairs or groups

0 exams

- No final, no midterm, no exams

Lateness policy

- Every homework may be submitted up to 48 hours after the deadline, without penalty.

- Homework submitted on time will receive a small bonus to their homework score.

- Submissions are **not accepted** beyond the 48-hour grace period. The grace period is strictly enforced.

- The final project must be turned in on time.

Browser and Text editor/IDE

- **Text editor:** You can use whatever you want. We recommend VSCode.

- **Browser:** Your code must work on Chrome, as that is what I'll use when grading your homework. It will not be tested in any other browser.

- **Homework turn-in:** We are using GitHub Classroom for assignment turnin.

Complete **Homework 0** to get all set up with your homework workflow in this course!

Lectures

Tue-Thu, 14h00-15h30 online

- Lectures will **not** be recorded
- Nothing will be graded in lecture
- But please come!
  - If you attend and do not feel the lectures are helpful, please send us a feedback!

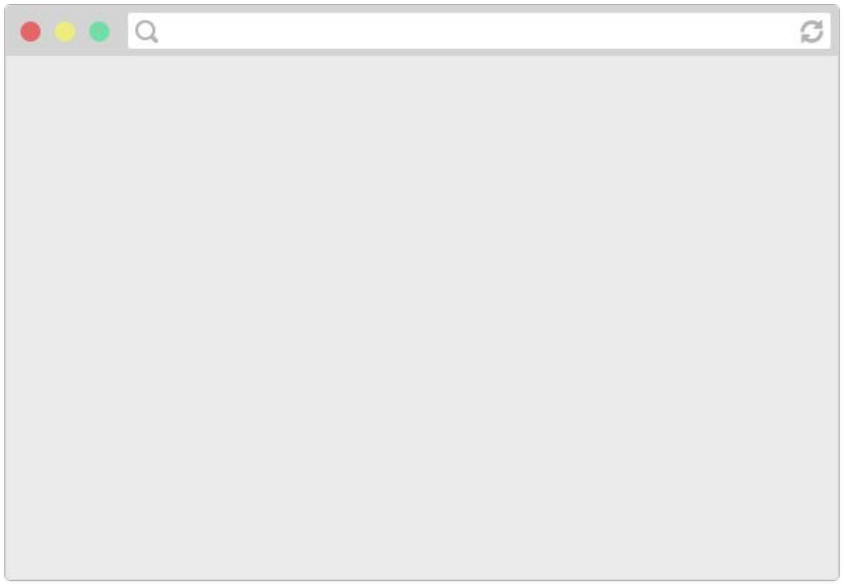
Questions?

Today's schedule

- ~~Syllabus~~
- ~~Course Info~~
- Browsers
- A little bit about **HTML** and **CSS**

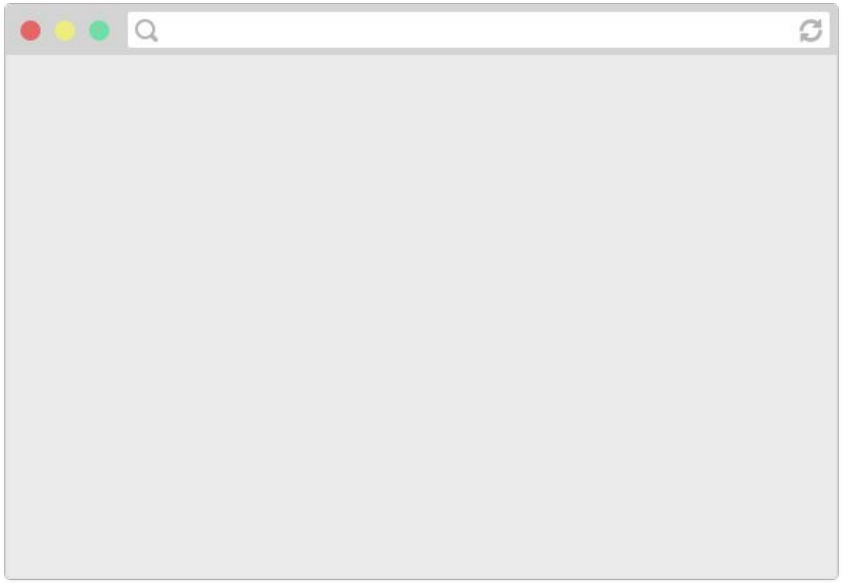
Browsers

How do web pages work?



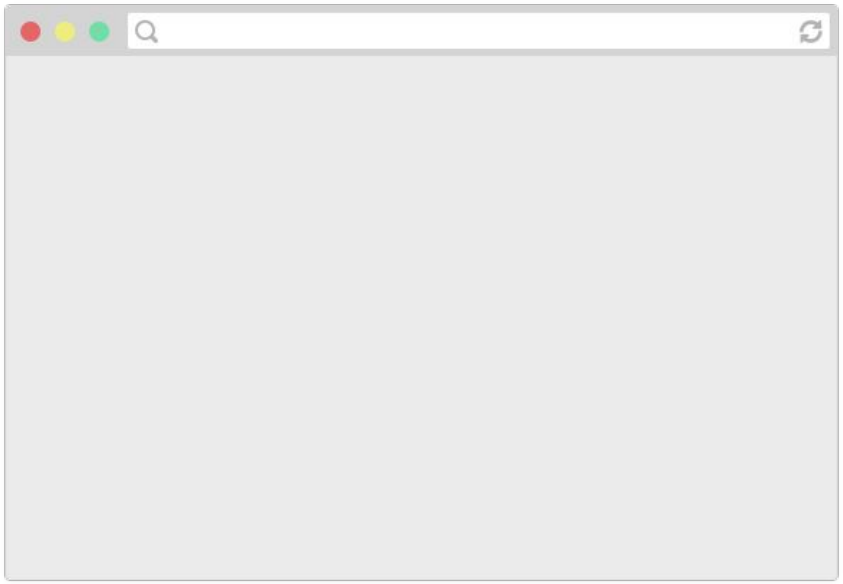
Browsers are applications that can display web pages.
E.g. Chrome, Firefox, Safari, Internet Explorer, Edge, etc.

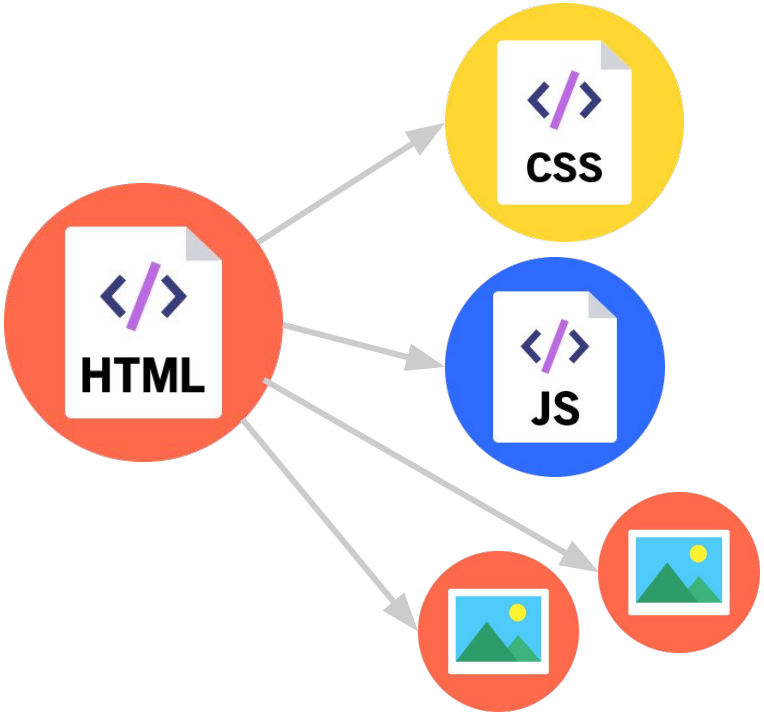
How do web pages work?



Web pages are written in a markup language called **HTML**, so browsers display a web page by reading and interpreting its HTML.

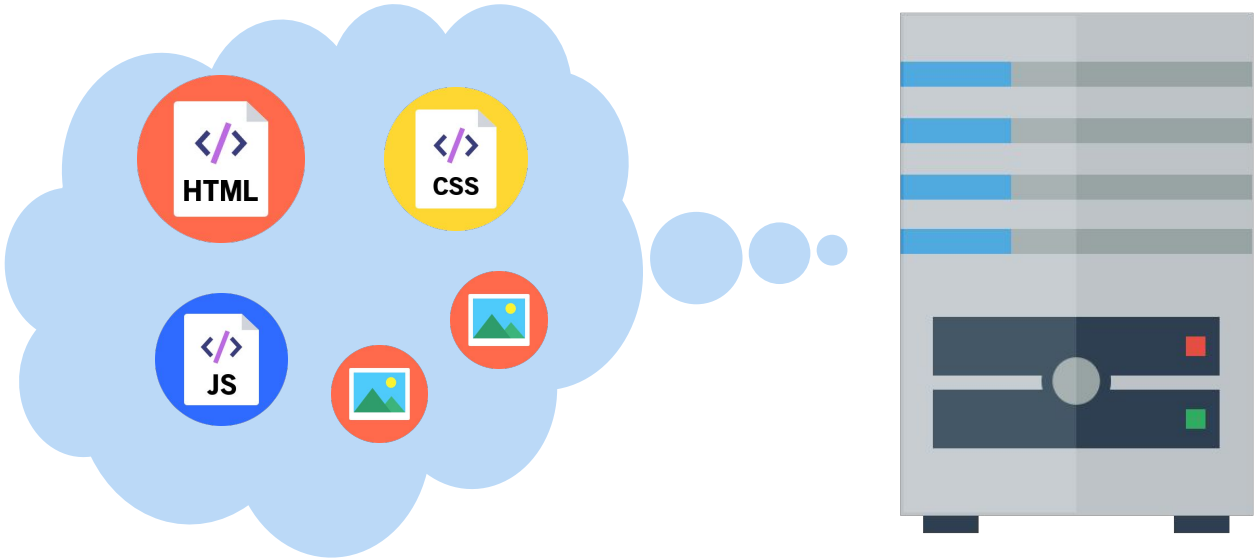
How do web pages work?





The HTML file might link to other resources, like images, videos, as well as **JavaScript** and **CSS** (stylesheet) files, which the browser then also loads.

How do web pages work?

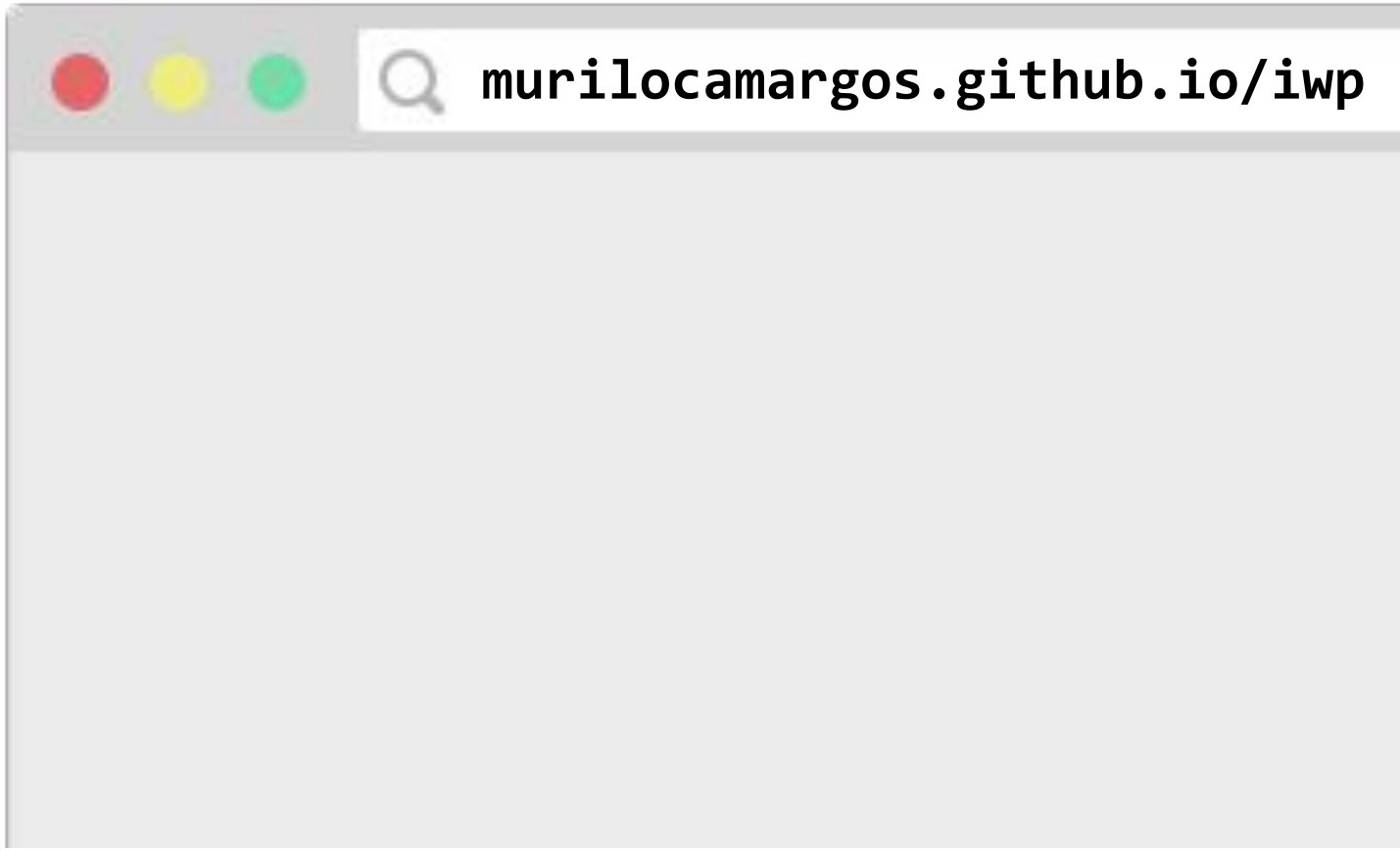


A **web server** is a program running on a computer that delivers web pages in response to requests.

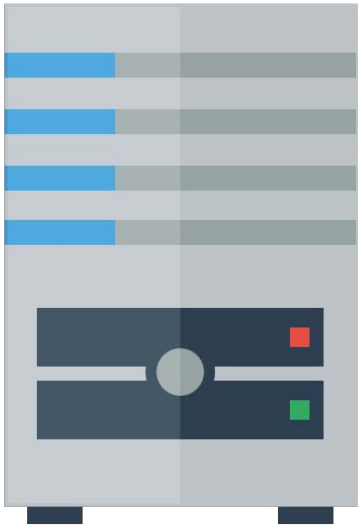
It either stores or generates the web page returned.

How do web pages work?

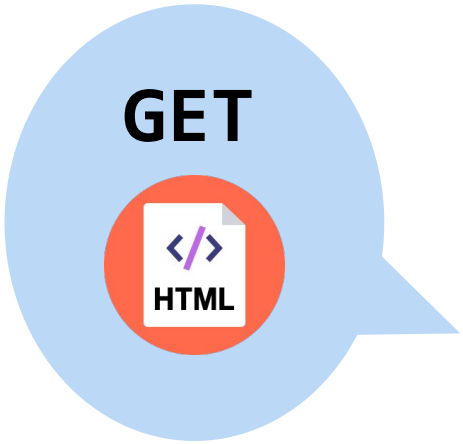
1. You type in a URL, which is the address of the HTML file on the internet.

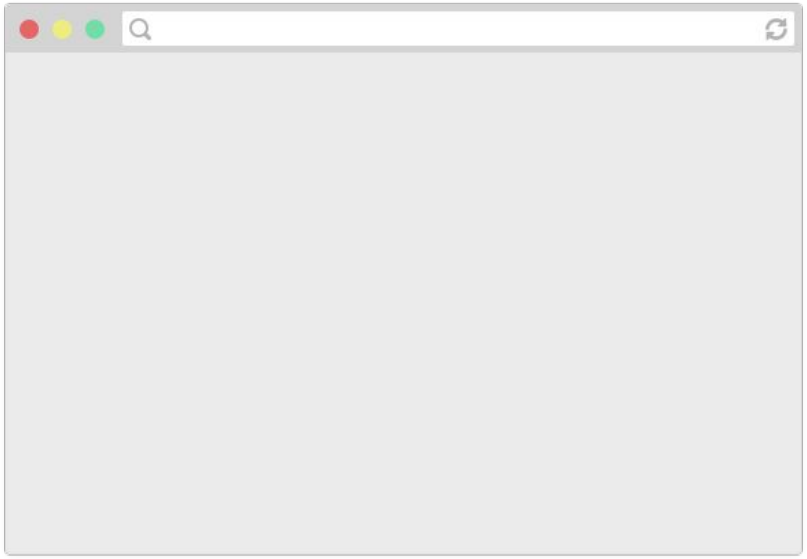


How do web pages work?

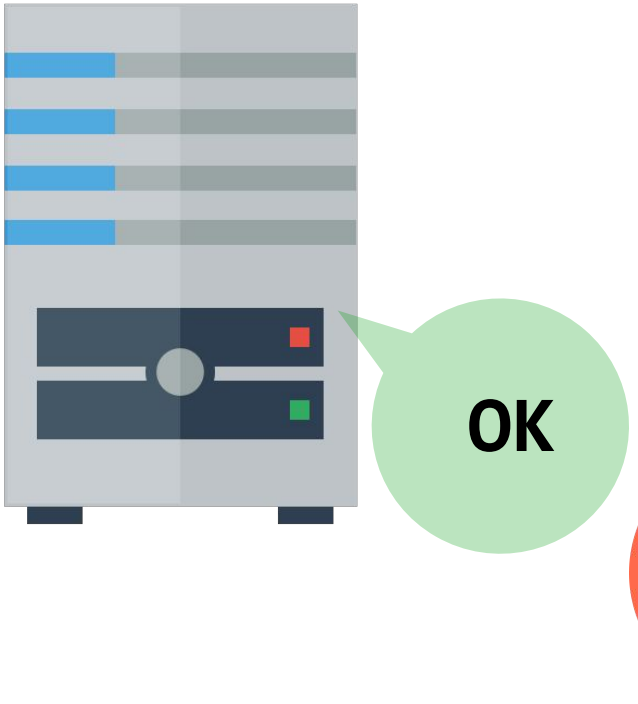


2. The browser asks the web server that hosts the document to send that document.

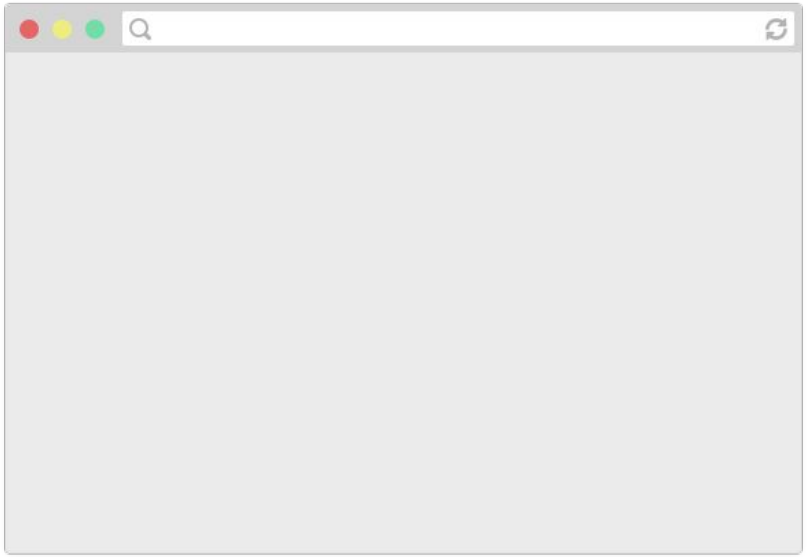




How do web pages work?

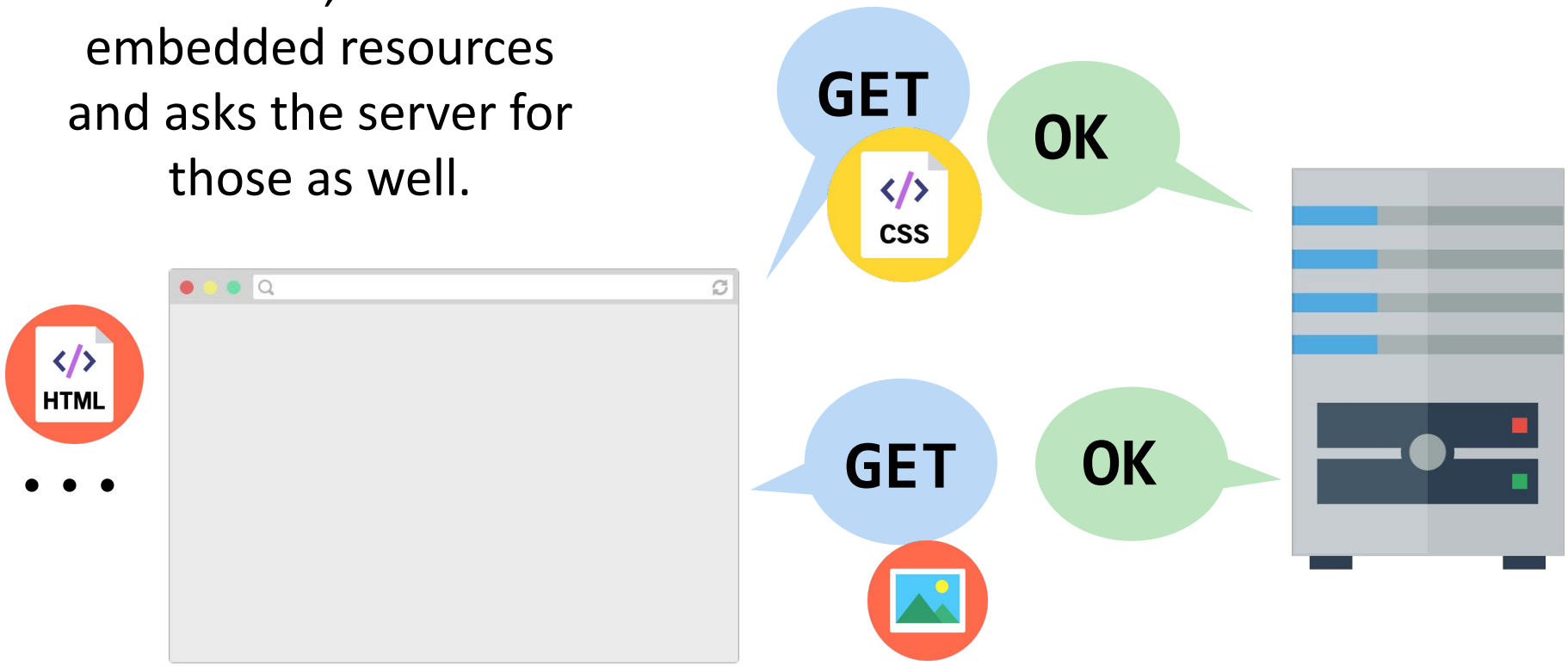


3. The web server responds to the browser with HTML file that was requested.



How do web pages work?

4. The browser reads the HTML, sees the embedded resources and asks the server for those as well.



How do web pages work?

5. The web page is loaded when all the resources are fetched and displayed.

P.S.

(That was obviously very hand-wavy. We'll get more detailed when we talk about servers later in the quarter.)

HTML and CSS

What is HTML?

HTML (**H**yper**t**ext **M**arkup **L**anguage)

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

```
<p>
  HTML is <em>awesome!!!</em>
  <img src="puppy.png" />
</p>
```

Basic HTML page structure

(i.e. copy/paste boilerplate)

```
<!DOCTYPE html>
<html>
  <head>
   <title>CS 193X</title>
  </head>

  <body>
   ... contents of the page...
  </body>
</html>
```

Saved in a *filename*`.html` file.

Basic HTML page structure

(i.e. copy/paste boilerplate)

Metadata that doesn't appear in the viewport of the browser

```
<!DOCTYPE html>
<html>
  <head>
    <title>CS 193X</title>
  </head>

  <body>
    ... contents of the page...
  </body>
</html>
```

E.g. **<title>** shows up as the name of the tab

Contents that render in the viewport of the browser

HTML elements

```
<p>
  HTML is <em>awesome!!!</em>
  <img src="puppy.png" />
</p>
```

- An element usually has start and ending tags (** ** and ** **)
  - **content**: stuff in between start and end tags
- An element can be self-closing (**img**)
- An element can have attributes (**src="puppy.jpg"**)
- Elements can contain other elements (**p** contains **em** and **img**)

Some HTML elements

(to place within **<body>**)

Top-level heading **h1, h2, … h6**	`<h1>Moby Dick</h1>`
Paragraph	`<p>Call me Ishmael.</p>`
Line break	`since feeling is first ` `who pays any attention`
Image	`<img src="cover.png" />`
Link	`<a href="google.com">click` `here!</a>`
Strong (bold)	`<strong>Be BOLD</strong>`
Emphasis (italic)	`He's my <em>brother</em> and all`

Exercise: Course web page

Let's write some HTML to make the following page:

Exercise: Course web page

HTML boilerplate

```
<!DOCTYPE html>
<html>
  <head>
    <title>Programação Web
Interativa</title>
  </head>

  <body>
    ...
  </body>
</html>
```

Plaintext contents of the page

```
Programação Web Interativa

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

Ver Ementa
```

CodePen

Solution

```
<!DOCTYPE html>
<html>
  <head>
    <title>Programação Web Interativa</title>
  </head>
  <body>
    <h1>Programação Web Interativa</h1>
    <strong>Datas importantes:</strong><br/>
    01/03: Começaram nossas aulas!<br/>
    01/03: A tarefa 0 está disponível.<br/>
    <br/>
    <a href="https://murilocamargos.github.io/iwp/syllabus">
      Ver Ementa
    </a>
  </body>
</html>
```

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.

CSS

CSS

CSS: **C**ascading **S**tyle Sheets

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

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

CSS

A CSS file is composed of **style rules**:

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

selector: Specifies the HTML element(s) to style.
property: The name of the CSS style.
value: The value for the CSS style.

Saved in a *filename*`.css` file.

CSS

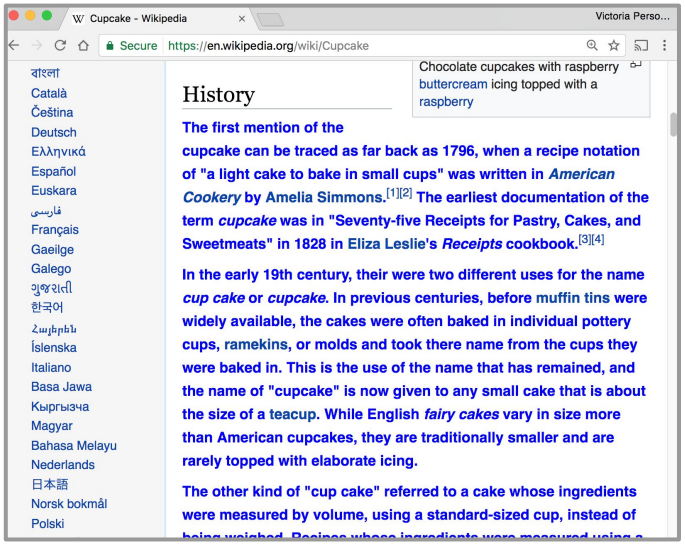
```
// NOT REAL CSS
fork {
   color: gold;
}
```

"All forks on the table should be gold"

CSS

```
p {
    color: blue;
    font-weight: bold;
}
```

"All **<p>** elements on the page should be blue and bold"



Linking CSS in HTML

(i.e. copy/paste boilerplate)

```
<!DOCTYPE html>
<html>
  <head>
    <title>IWP</title>
    <link rel="stylesheet" href="filename.css" />
  </head>

  <body>
   ... contents of the page...
  </body>
</html>
```

Some CSS properties

There are over 500 CSS properties! Here are a few:

Font face (mdn)	`font-family: Helvetica;`
Font color (mdn)	`color: gray;`
Background color (mdn)	`background-color: red;`
Border (mdn)	`border: 3px solid green;`
Text alignment (mdn)	`text-align: center;`

Aside: Mozilla Developer Network (MDN) is the best reference for HTML elements and CSS properties

- The actual W3 spec is very hard to read (meant for browser developers, not web developers)

Main ways to define CSS colors:

140 predefined names (list)

```
color: black;
```

rgb() and rgba()

```
color: rgb(34, 12, 64);
color: rgba(0, 0, 0, 0.5);
```

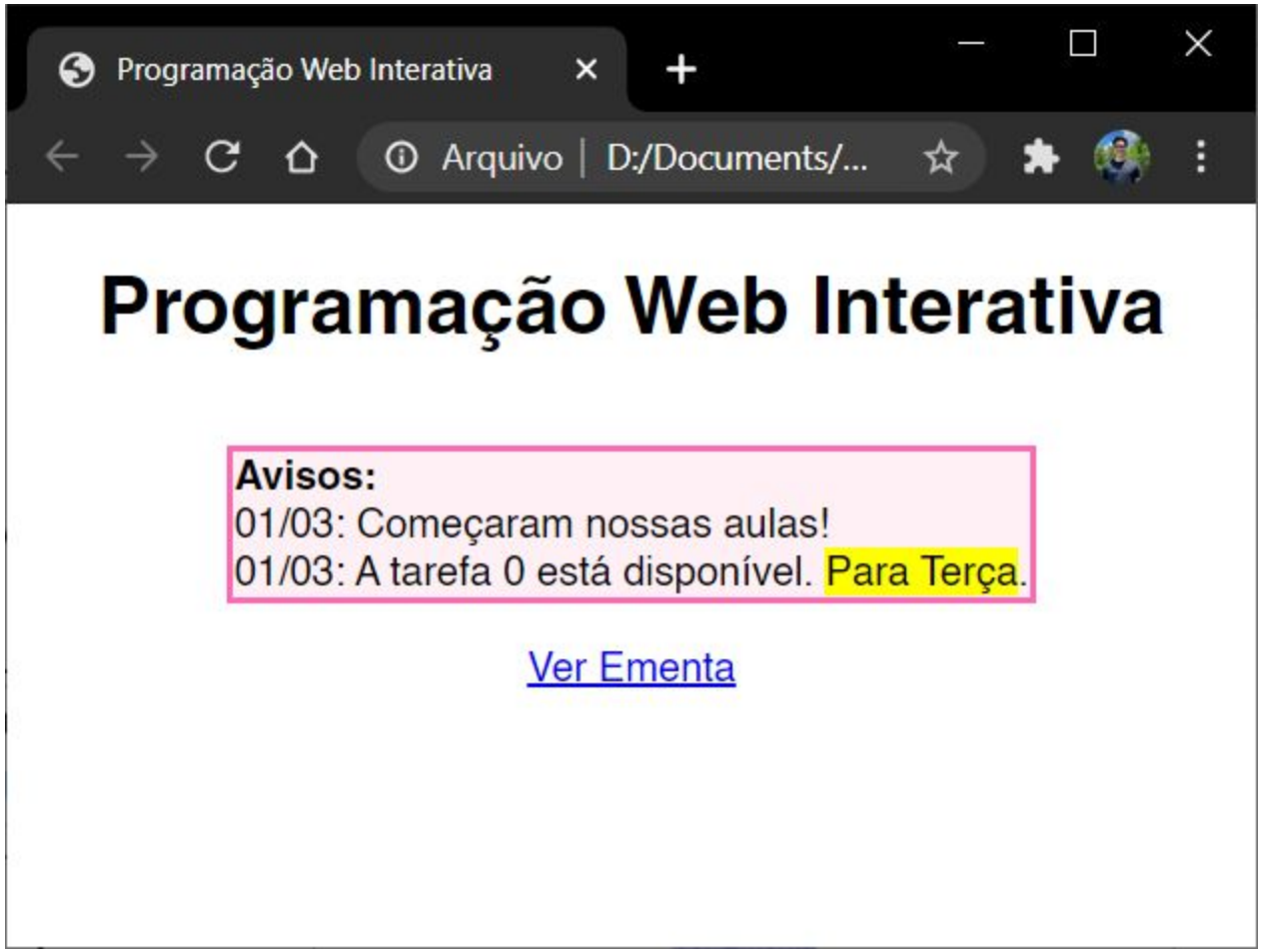
Hex values

```
color: #00ff00;
color: #0f0;
color: #00ff0080;
```

- The "a" stands for **alpha channel** and is a **transparency** value
- Generally prefer more descriptive over less:
  1. Predefined name
  2. rgb / rgba
  3. Hex

Exercise: Course web page

Let's write some CSS to style our page:



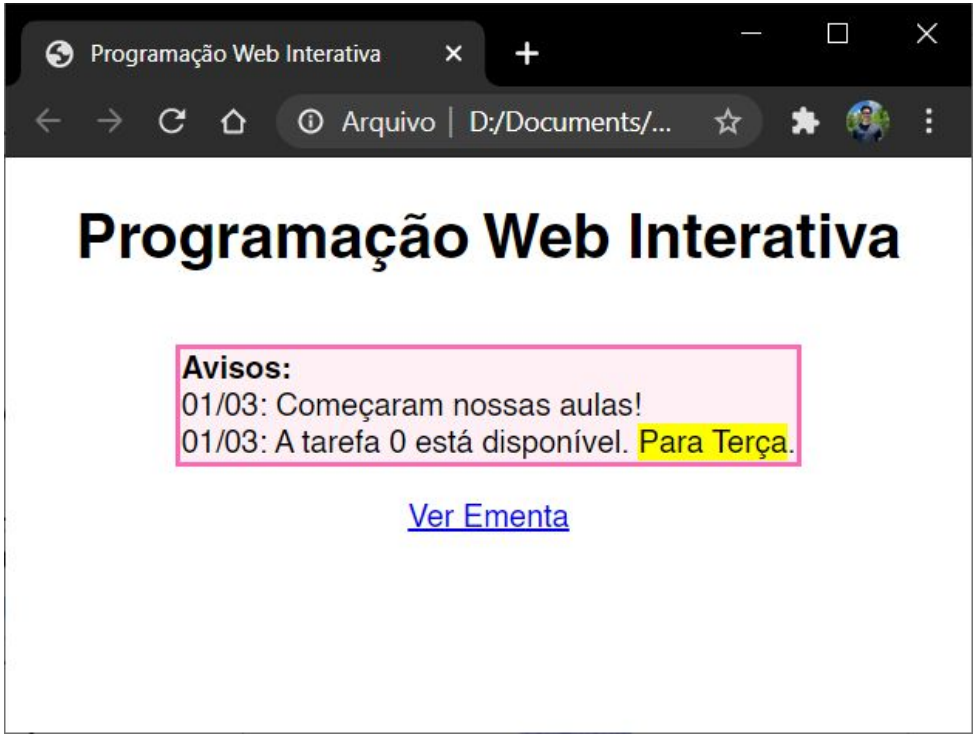
Exercise: Course web page

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

CSS exercise debrief

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.)

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

- Couldn't set `text-align: center;` to the `<a>` or `<strong>` tags directly, but could center `<p>` and `<h1>`
- Had to set a `width` on the box to make it hug the text … any other way to do this?
- How to center the box?! How do you highlight?!

Q: Why is HTML/CSS so bizarre??

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

block vs **inline** display

Next time!

Homework 0 is **out now,** due Tuesday March 9