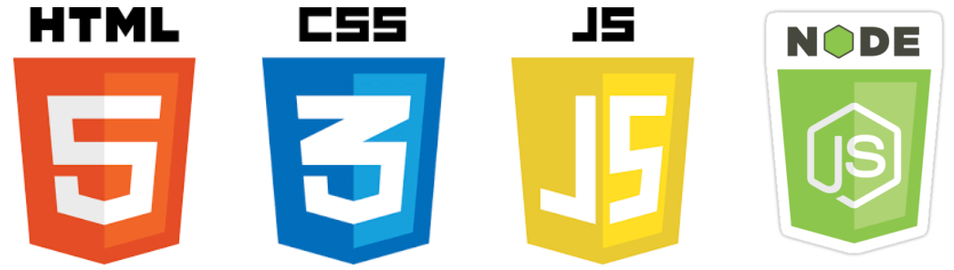
App Dev

Stefano Balietti

Center for European Social Science Research at Mannheim University (MZES)
Alfred-Weber Institute of Economics at Heidelberg University

@balietti | stefanobalietti.com | @nodegameorg | nodegame.org

Building Digital Skills: 5-14 May 2021, University of Luzern



Goals of the Seminar:

1. **Writing and understanding asynchronous code:** event-listeners, remote functions invocation.

2. **Basic front-end development:** HTML, JavaScript, CSS, debugging front-end code.

3. **Introduction to front-end frameworks**: jQuery and Bootstrap

4. **Introduction to back-end development:** NodeJS Express server, RESTful API, Heroku cloud.

Outputs of the Seminar:

1. **Web app:** in NodeJS/Express.
2. **Chrome extensions:** architecture and examples.
3. **Behavioral experiment/survey:** nodeGame framework.
4. **Mobile development:** hybrid apps with Apache Cordova, intro to Ionic Framework, progressive apps (PWA).

Your Instructor: Stefano Balietti

http://stefanobalietti.com

Currently

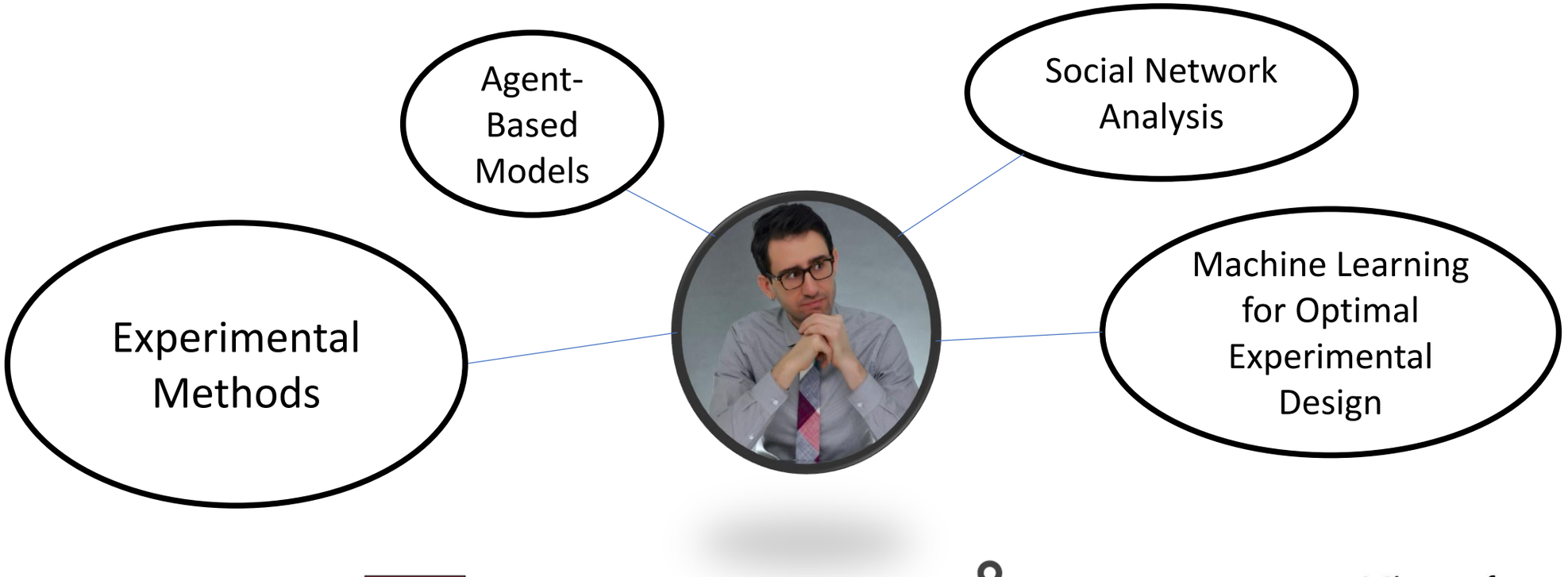
- Fellow in Sociology Mannheim Center for European Social Research (MZES)
- Postdoc at the Alfred Weber Institute of Economics at Heidelberg University

Previously

- Microsoft Research - Computational Social Science New York City
- Postdoc Network Science Institute, Northeastern University
- Fellow IQSS, Harvard University
- PhD, Postdoc, Computational Social Science, ETH Zurich

My Methodology

Interface of computer science, sociology, and economics



ETH zürich

HARVARD UNIVERSITY

network science institute

Microsoft Research

Building Platforms

Garch-in-Gretl (GiG) for econometrics Gretl software

~**5000** weekly downloads

Patterns Configuration Module for Drupal Web Content Management System

2,622 active users, **30,448** downloads

```
 _______  _______  _   __  _  _______
|       \|       \| \  |  \|       \
 \$$$$$$$| $$$$$$$| $$  | $$ \$$$$$$$
   | $$   | $$__     \$$\/  $$   | $$
   | $$   | $$  \     >$$  $$    | $$
   | $$   | $$$$$    /  $$$$\    | $$
   | $$   | $$_____ |  $$ \$$\   | $$
   | $$   | $$      \| $$  | $$   | $$
    \$$    \$$$$$$$$ \$$   \$$    \$$
 __    __  ______  _______  __      _______
|  \  _  |  \ /      \ |       \ |  \     |       \
| $$ / \ | $$|  $$$$$$\| $$$$$$$\| $$     | $$$$$$$\
| $$/  $\| $$| $$  | $$| $$__| $$| $$     | $$  | $$
| $$  $$$\ $$| $$  | $$| $$    $$| $$     | $$  | $$
| $$ $$\$$\$$| $$  | $$| $$$$$$$\| $$     | $$  | $$
| $$$$  \$$$$| $$__/ $$| $$  | $$| $$_____| $$__/ $$
| $$$    \$$$ \$$    $$| $$  | $$| $$     \| $$    $$
 \$$      \$$  \$$$$$$  \$$   \$$ \$$$$$$$$ \$$$$$$$
```

Fast, scalable JavaScript for large-scale real-time online experiments

nodeGAME®

v6

www.nodegame.org

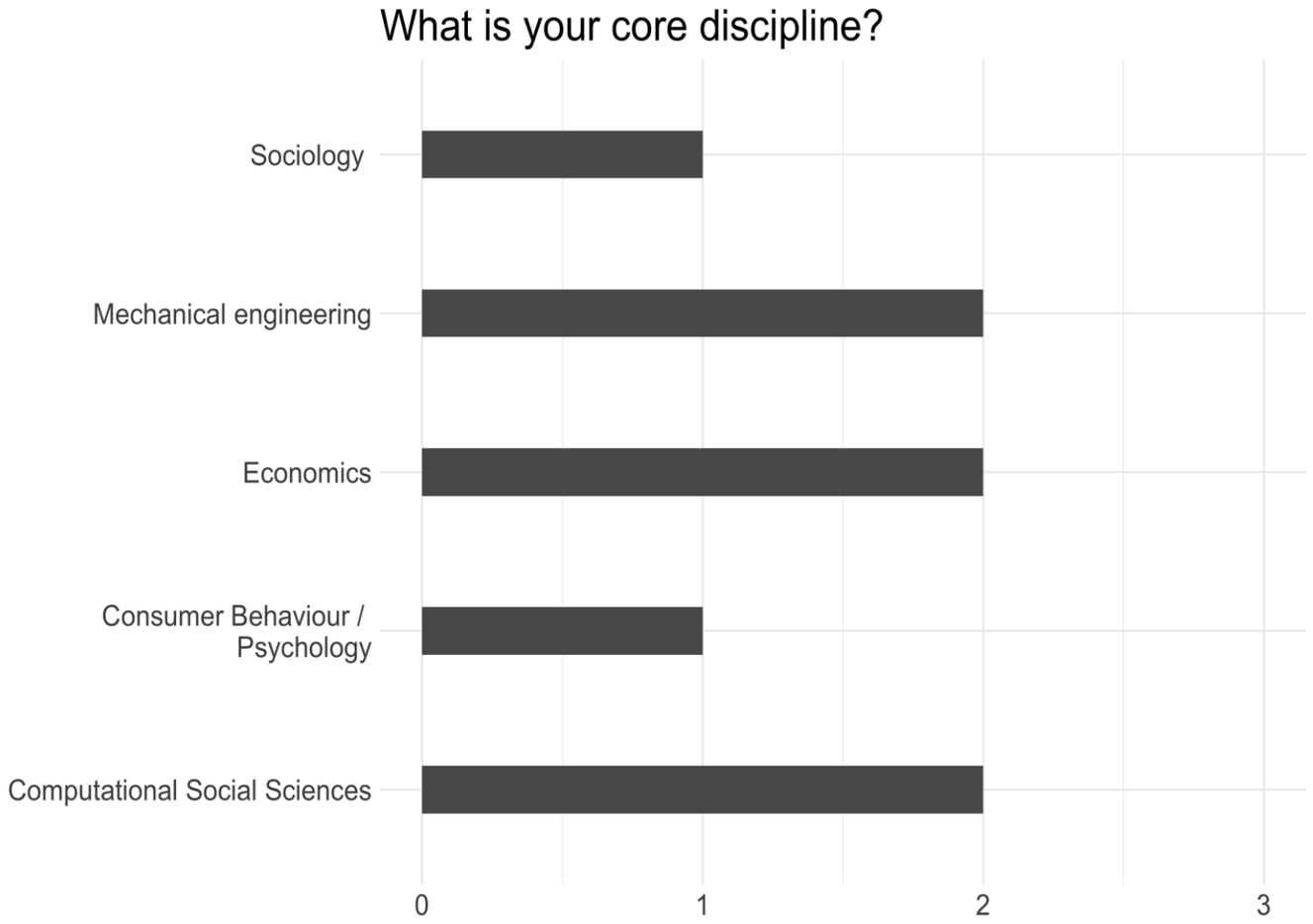
Microsoft

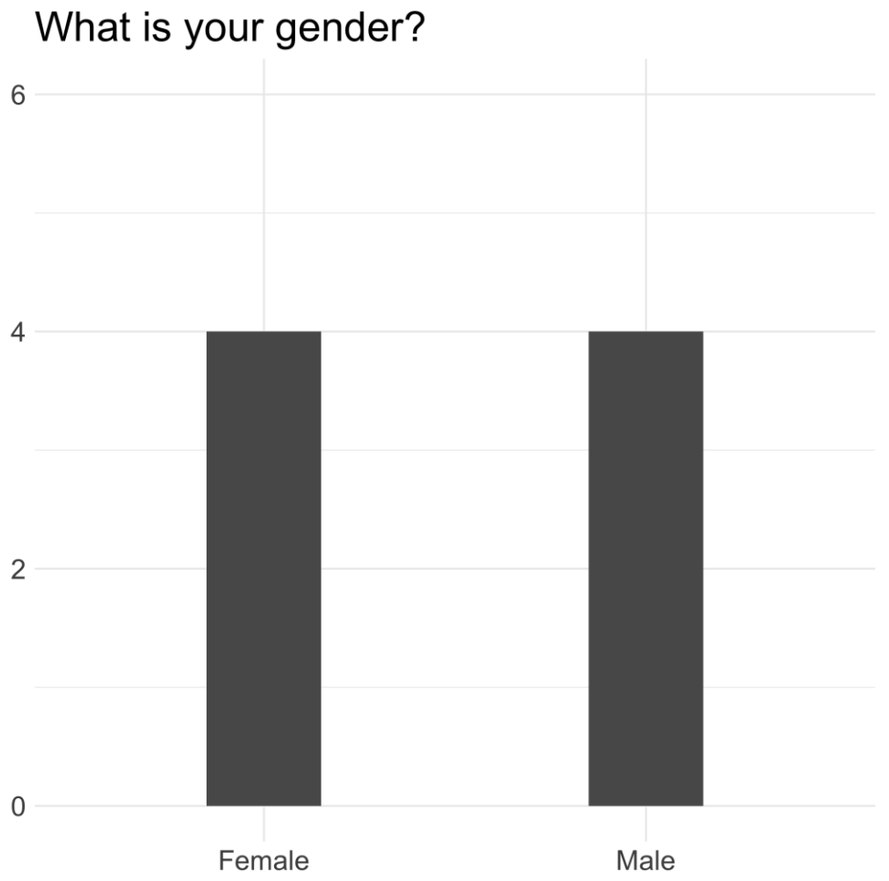
My Vision

Simulating Societal Processes in Virtual Labs

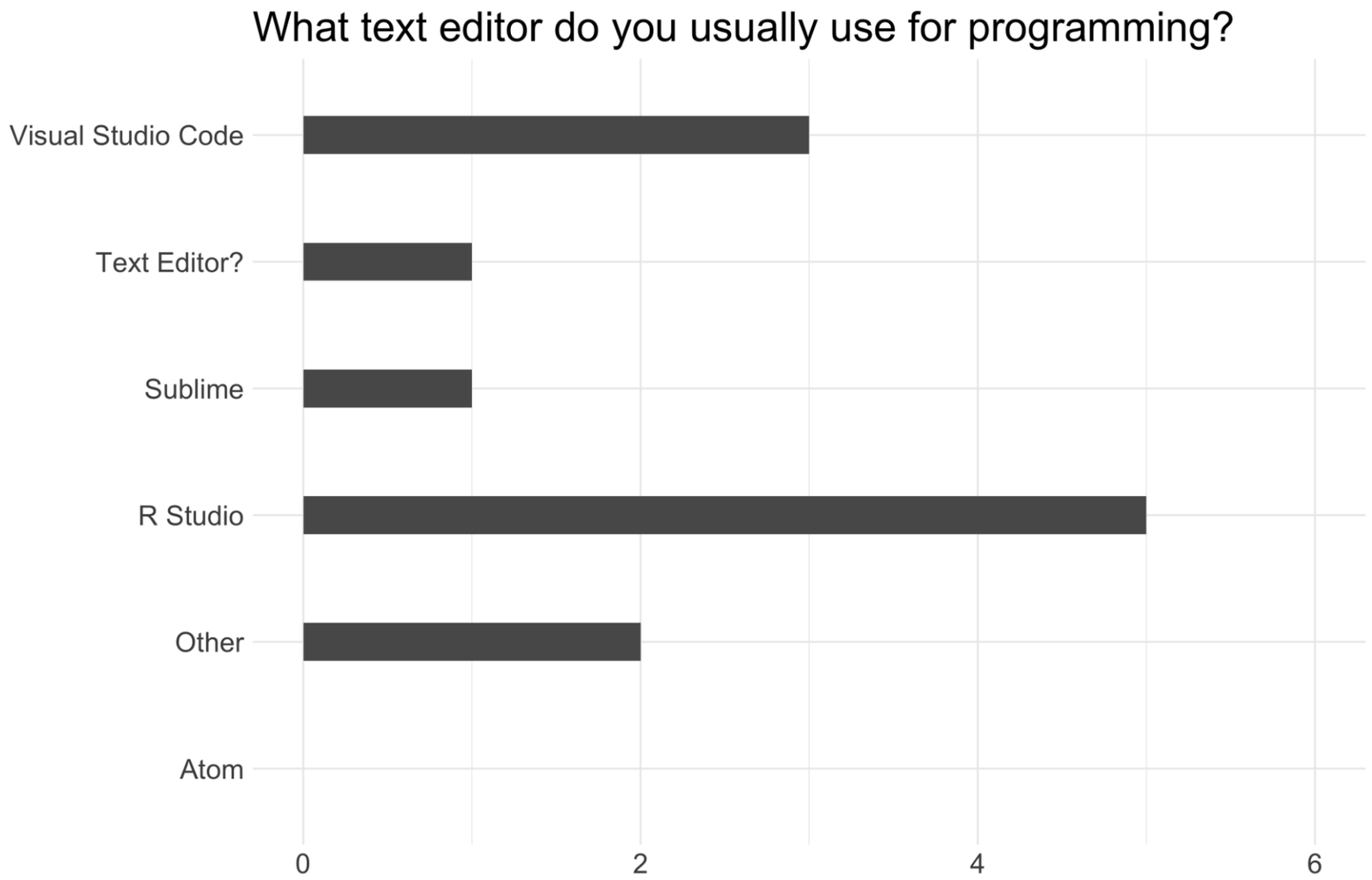
- Consensus, social influence, and polarization
- Group fairness, inequality, redistribution
- Incentives schemes for collective intelligence
- Optimal experimental design

About You (from web survey)

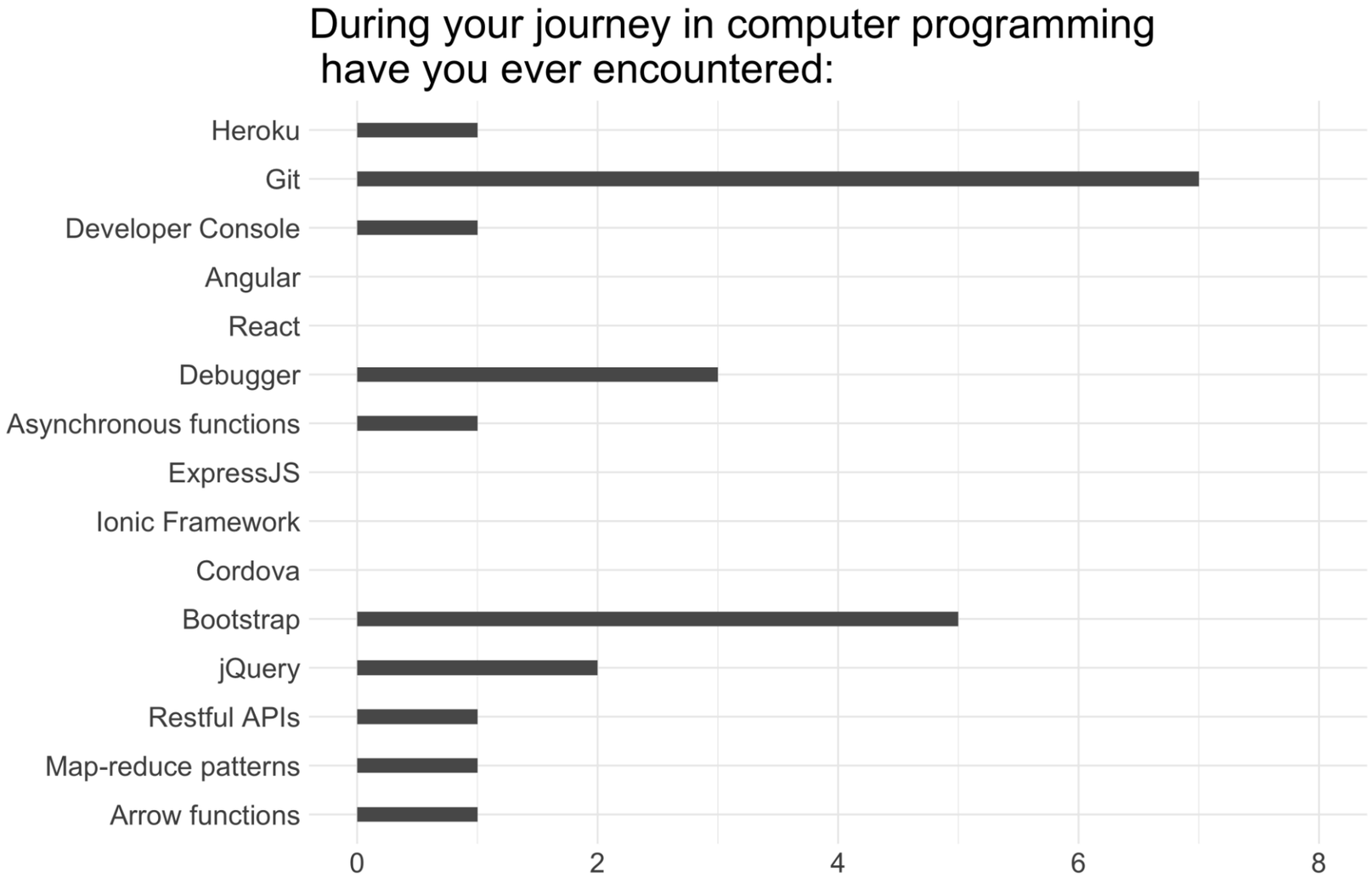




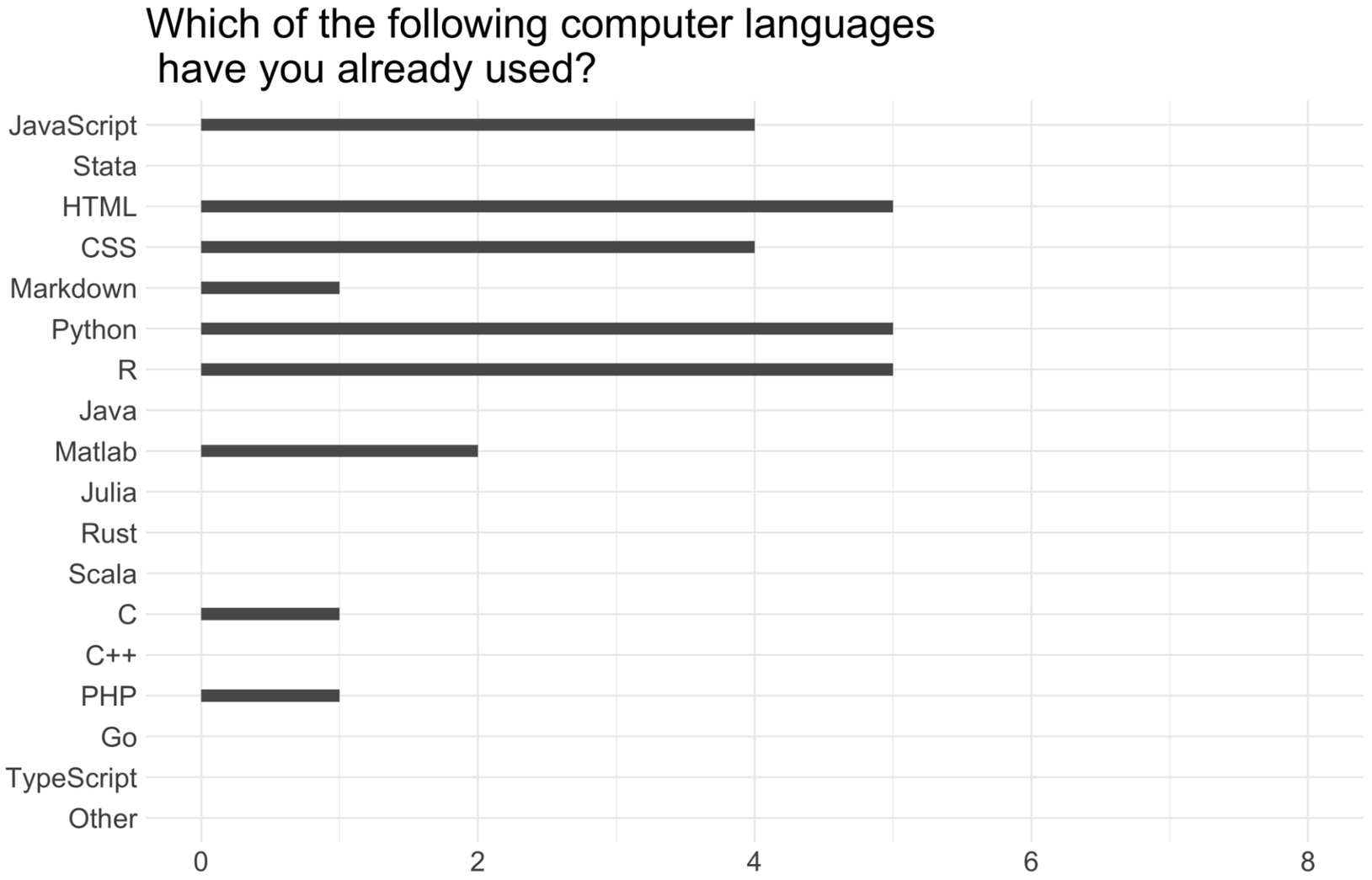
About You (from web survey)



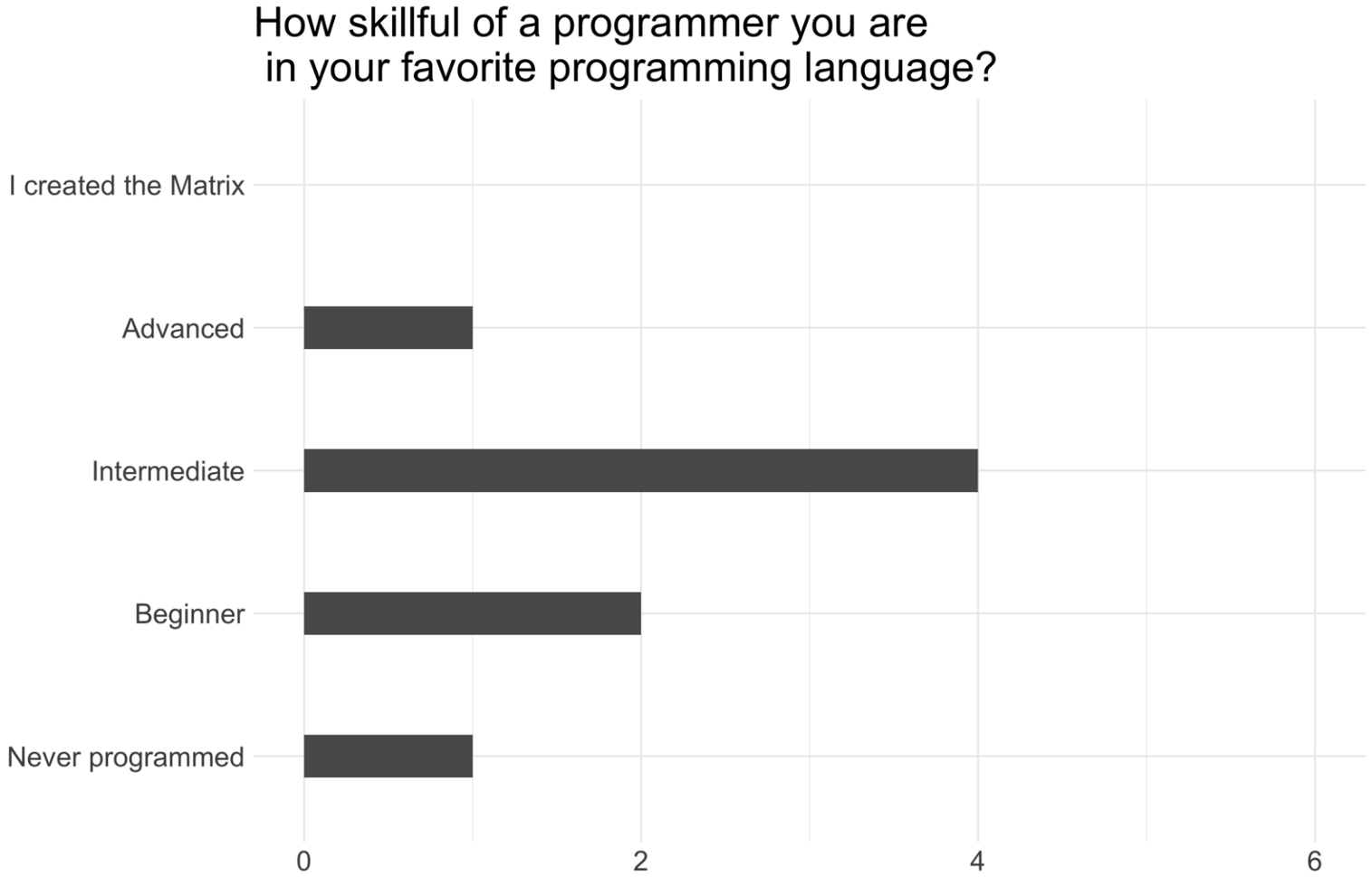
About You (from web survey)



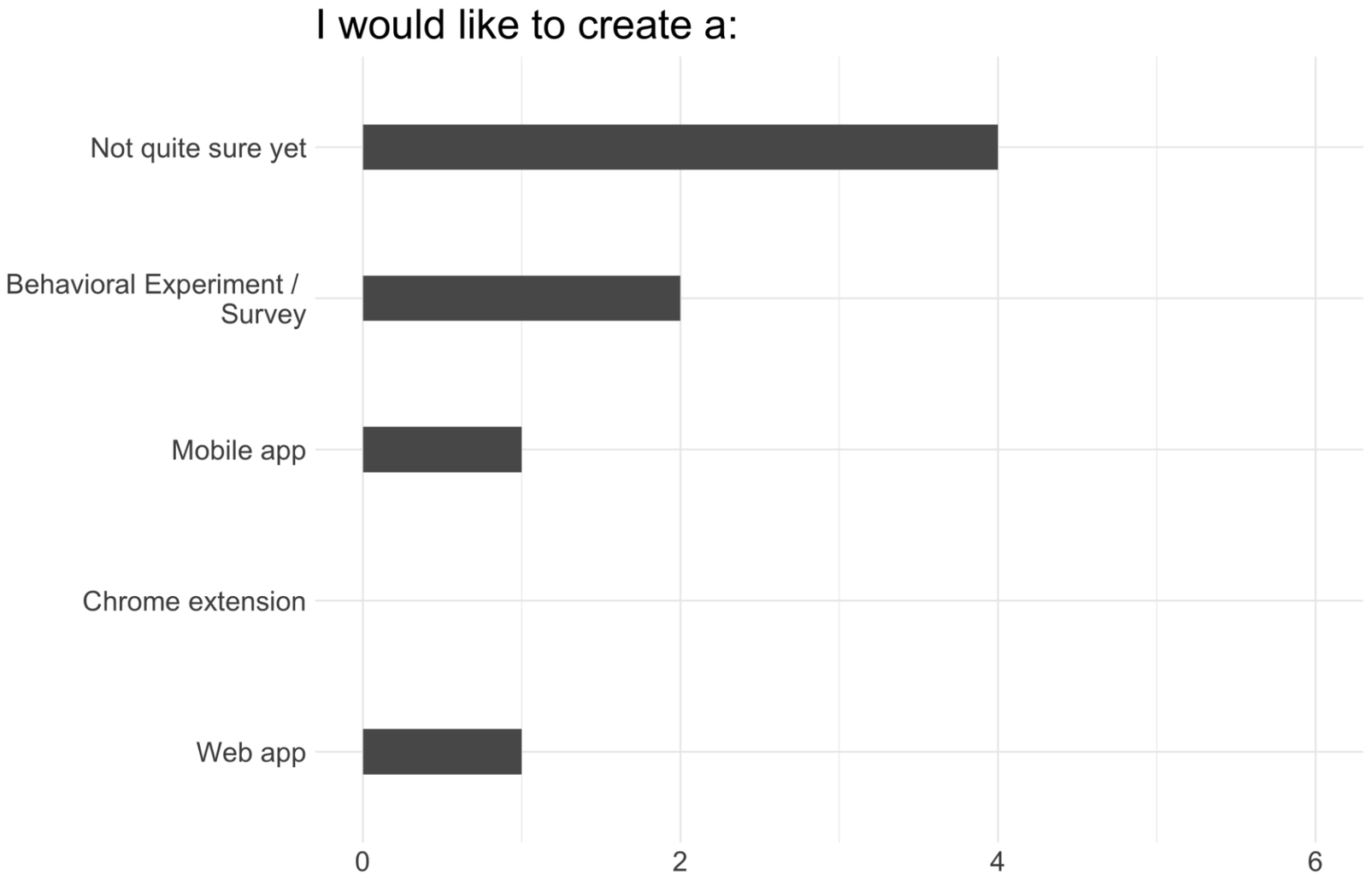
About You (from web survey)



About You (from web survey)



About You (from web survey)



Approach in This Course

- **Based on previous feedback,** lean slides and focus on exercises to do at your own pace.
- So, I just watch you coding online? That's weird.
- **I will code along** with you at a slow pace, take breaks, stimulate questions.

How to Get the Most out of This Course

- We will get a solid ground on goals 1-4
- Then pick a project that interests you most.
- No idea? Just do exercises ☺

How to Get the Most out of This Course

- We will get a solid ground on goals 1-4
- Then pick a project that interests you most.
- No idea? Just do exercises ☺

- Exercises marked with **Optional** are *optional*
- Exercises marked with **Bonus** are *optional* and *more complex*
- All exercises have a solution in the *solutions/* folder, but don't look in there too early!

Prerequisites

- Follow up from the fundamentals of *computer programming course*, so some knowledge of **programming** is expected

- Some knowledge of **JavaScript** is expected

JavaScript is NOT Java

"Java is to JavaScript as ham is to hamster." (Jeremy Keith)

Image source

JavaScript is NOT Java

"Java is to JavaScript as car is to carpet." (Chris Heilmann)

53.12 incl. VAT

Image source

€61.39 incl. VAT

Image source

JavaScript

- JavaScript was developed in May 1995 by *Brendan Eich* for Netscape Communications Corp
- Was created in **10 days** in order to accommodate the Navigator 2.0 Beta release
- Initially called **Mocha**, later renamed **LiveScript** in September, and later **JavaScript** in the same month

JavaScript

- Microsoft introduced **JScript** as reverse-engineered implementation of Netscape's JavaScript in 1996 in Internet Explorer 3

- In 1996 Netscape submitted JavaScript to European Computer Manufacturers Association (ECMA) to create and industry standard

- In 1997 **ECMAScript** was released

- Between 1997 and 2009 5 standard have been released.

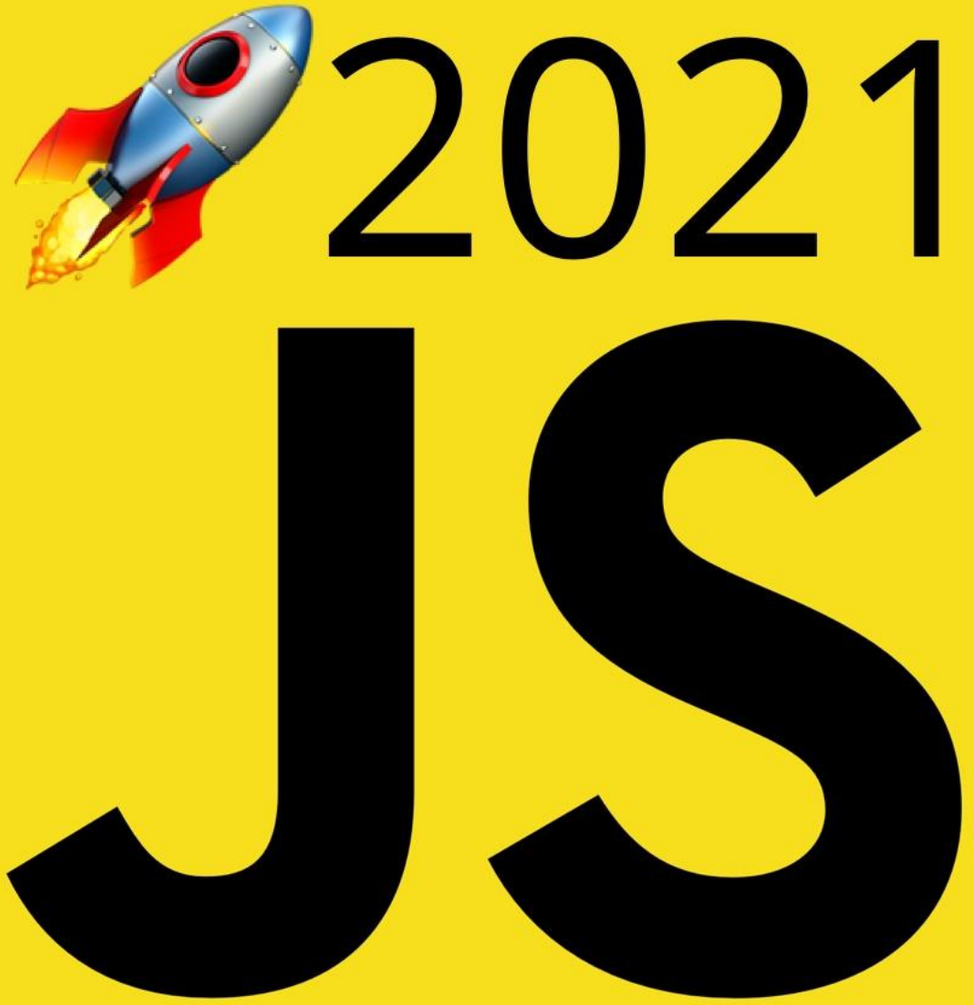
- *July 2015 ECMASCRIPT V6 released.*

JavaScript Releases

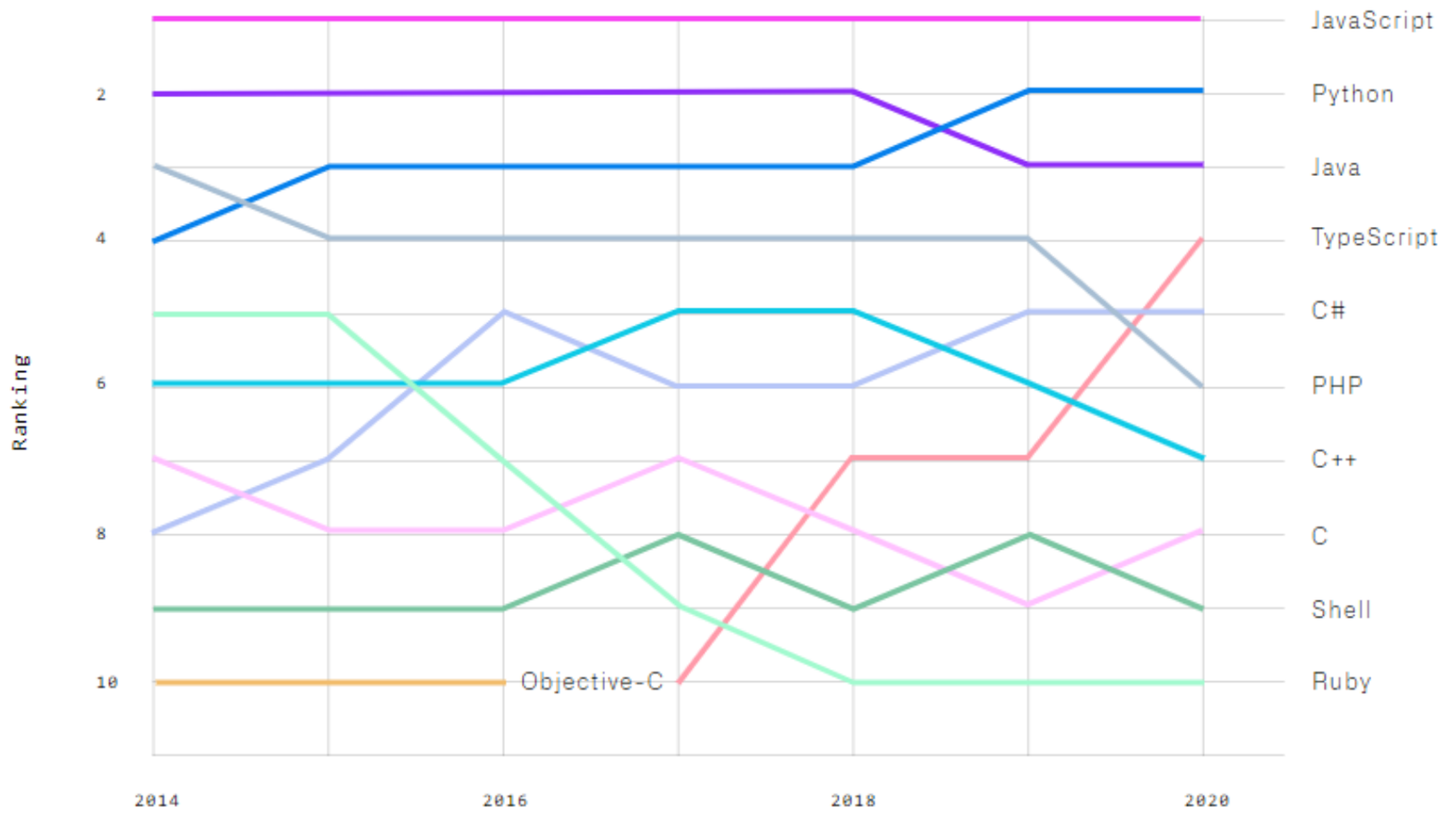
- ES2016 a.k.a. ES7
- ES2017 a.k.a. ES8
- ES2018 a.k.a. ES9
- ES2019 a.k.a. ES10
- ES2020 a.k.a. ES11
- ES2021 a.ka. ES12

  (scheduled for June)

Image source



JavaScript is #1 Language on Github



Github.com

Let’s look back at the code and communities built on GitHub this year...

Based on the data collection range of October 2019 - September 2020.

56 M+
total developers on GitHub

72 %
of Fortune 50 companies use GitHub Enterprise

60 M+
new repositories created in the last year

1.9 B+
contributions added in the last year

https://octoverse.github.com/

Tentative Schedule

Goals: May 5th – May 7th

- Asynchronous code in JavaScript,
- NodeJS and NPM,
- The golden triad of web development: HTML, CSS, and JavaScript (Part 1).

- The golden triad of web development: HTML, CSS, and JavaScript (Part 2),
- Introduction to Web frameworks: JQuery and Bootstrap.

- A simple web app with Express.
- Introduction to the Cordova, PWA, Ionic Framework, Chrome-based browser extensions, and nodeGame framework for behavioral research.

Outputs: May 12nd and May 14th

- Targeted material and exercises and custom support for students projects.

You Get the Certificate If

Attend **all days**.

No problems if you *miss a few hours*.

Quick Setup Checkpoint

You have installed

- NodeJS
- Git
- Modern Text Editor (Visual Studio Code or Atom recommended)

Quick Setup Checkpoint

You have installed

- NodeJS
- Git
- Modern Text Editor (Visual Studio Code or Atom recommended)

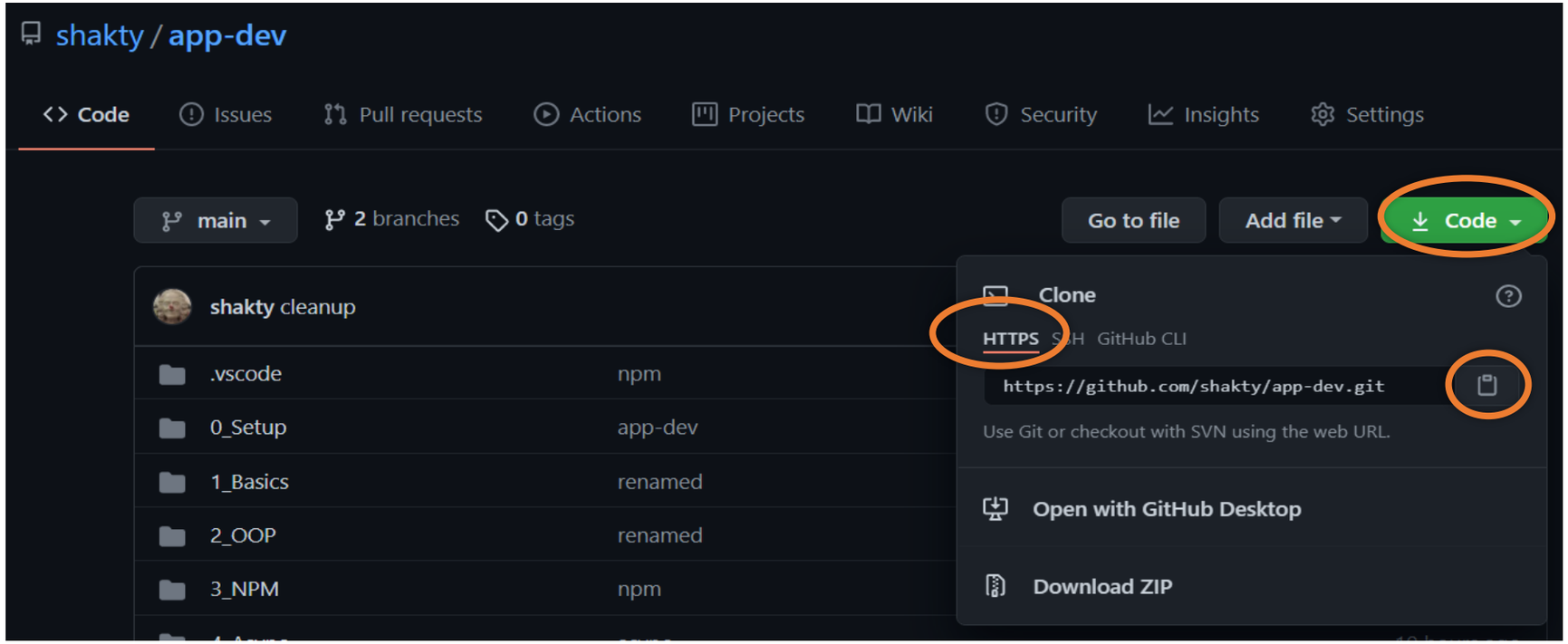
Fork the repository of exercises onto your GitHub account

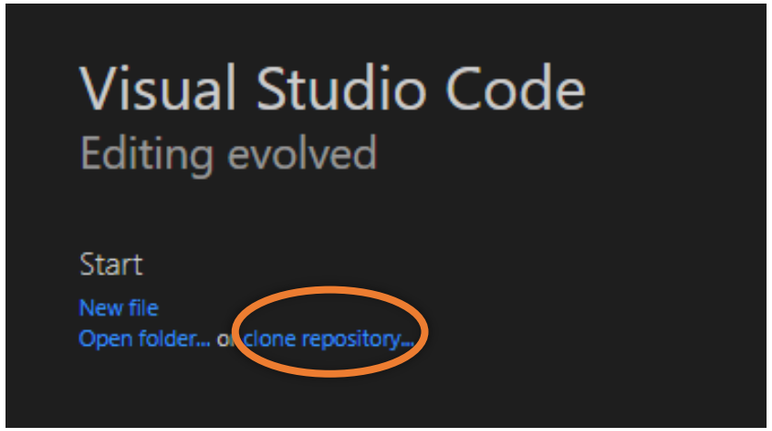
https://github.com/shakty/app-dev

Clone the forked repository onto your machine

Forking Instructions

https://github.com/shakty/app-dev



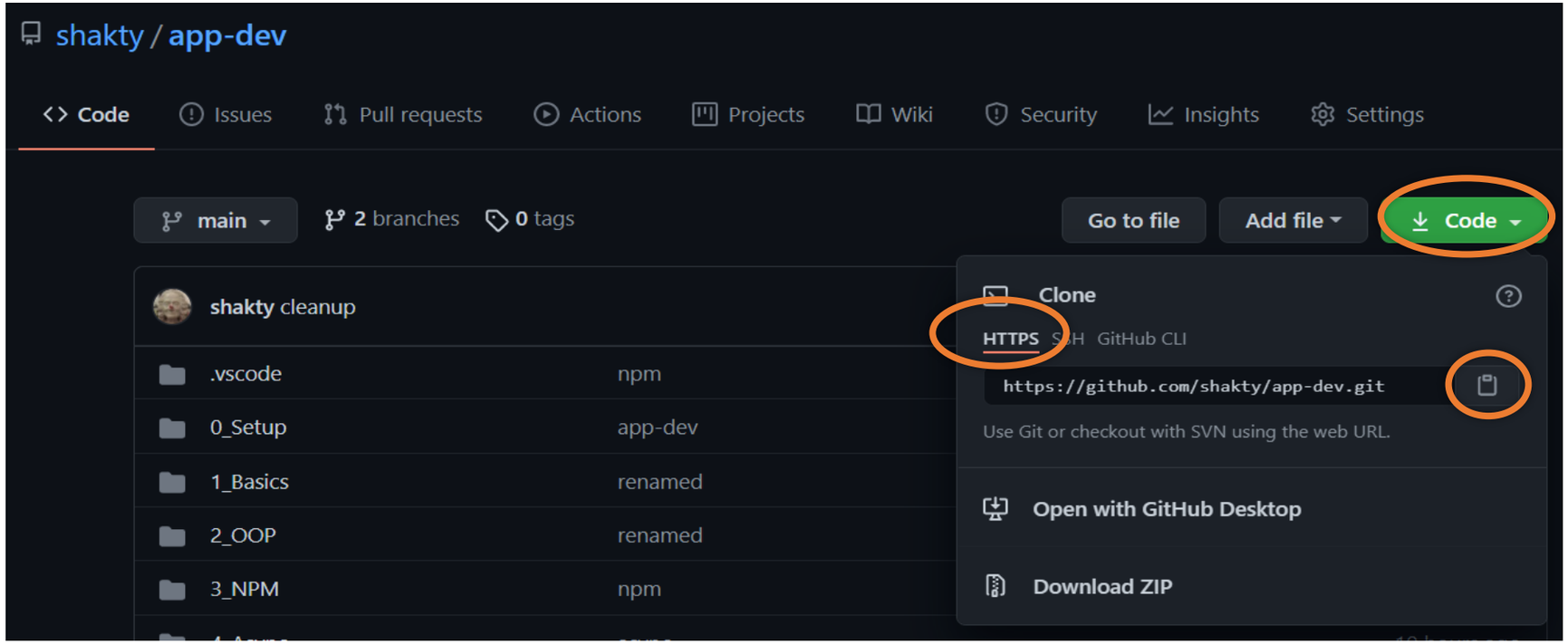


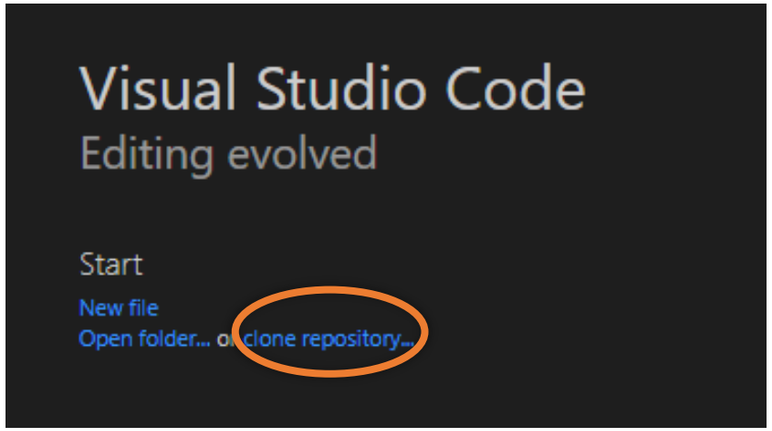
For Atom, I made this video:

https://www.youtube.com/watch?v=MDU2p9YtvIA

Forking Instructions

https://github.com/shakty/app-dev





For Atom, I made this video:

https://www.youtube.com/watch?v=MDU2p9YtvIA

Do not mix up with spooning :)

Module 1: NodeJS and NPM

Module 1: NodeJS and NPM

Learning Goals

- You should already know some JavaScript, soft reboot
- Search and install NodeJS packages from NPM
- What is the package.json file
- The node_modules directory
- Load packages into NodeJS programs
- Requiring and exporting local files

Node.JS

- **Node.JS** was invented in 2009 by *Ryan Dahl* and other developers working at Joyent

- Combination of Google's V8 JavaScript engine, an event loop, and a low-level I/O API

- **npm**, the node package manager, in 2011

- Versions: 0.10, 0.12, 4.0 … 16.0!

Node.JS

- **Node.JS** was invented in 2009 by *Ryan Dahl* and other developers working at Joyent

- Combination of Google's V8 JavaScript engine, an event loop, and a low-level I/O API

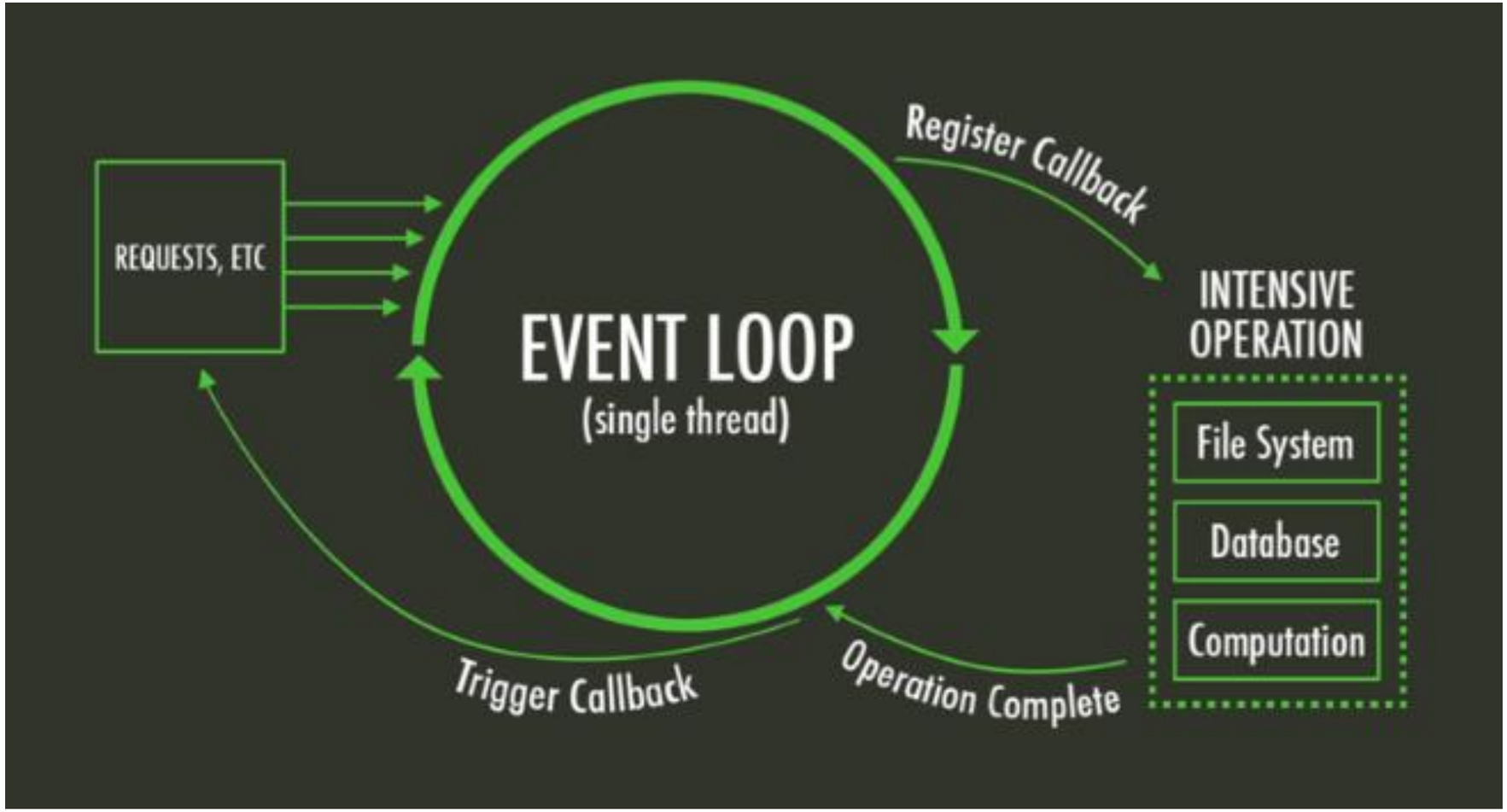
- **npm**, the node package manager, in 2011

- Versions: 0.10, 0.12, 4.0 … 16.0!

Module 1: References

- https://nodejs.org/en/
- https://www.npmjs.com/
- https://docs.npmjs.com/cli/v6/configuring-npm/package-json
- https://www.geeksforgeeks.org/node-js-modules/

Module 2: Asynchronous Code



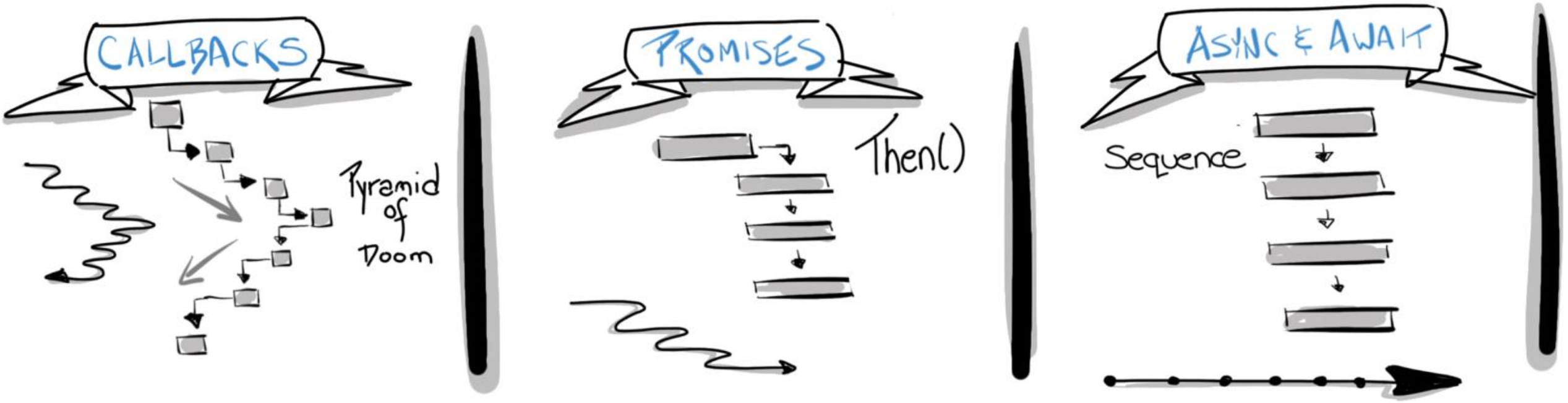
Picture source

Module 2: Asynchronous Code

Learning Goals

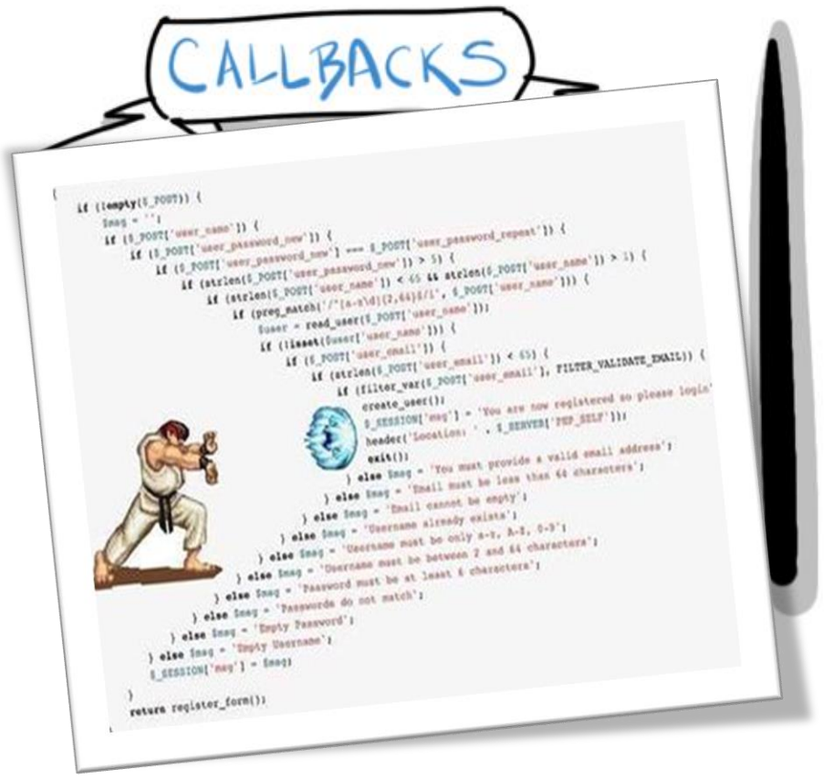
- Writing async code with callbacks
- Writing async code with promises
- Writing async code with async/await pattern
- The Bread and Butter of async programming

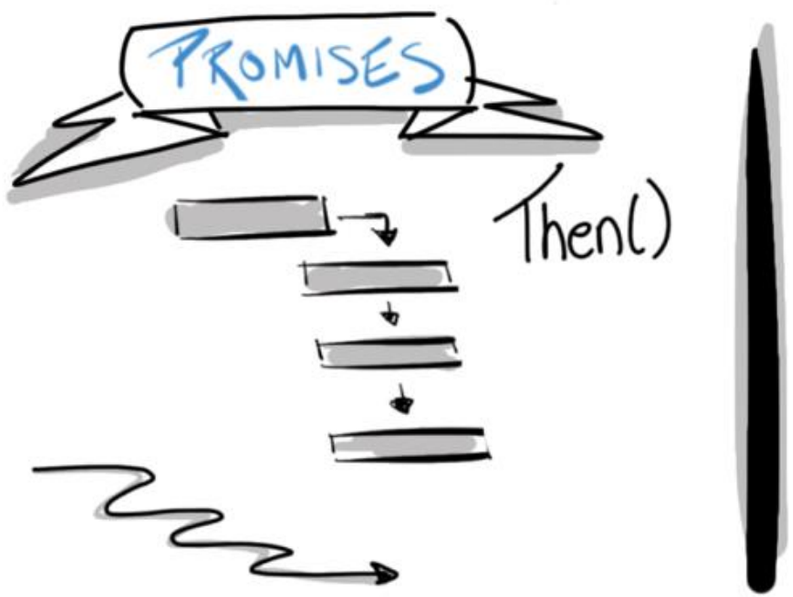
Asynchronous Code

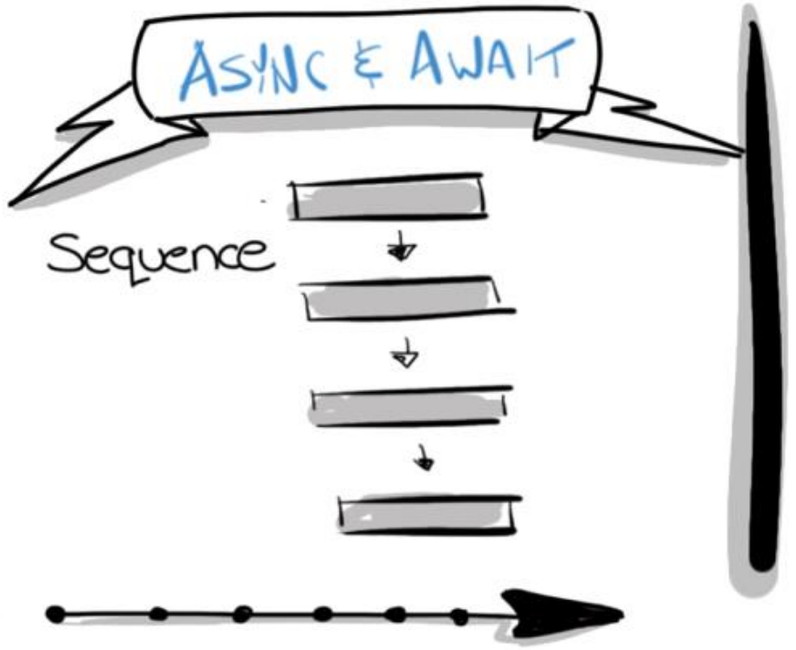


Picture source

Asynchronous Code





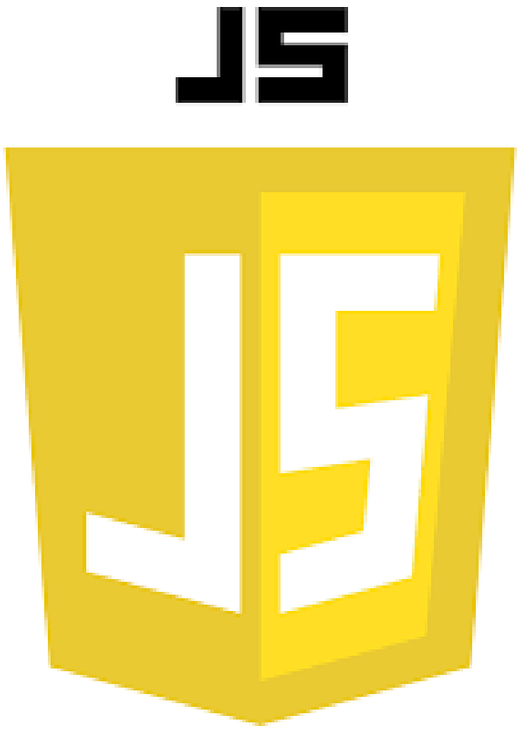


[Picture source]

Module 2: References

- https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Asynchronous
- https://javascript.info/async

Module 3: HTML, JS, CSS

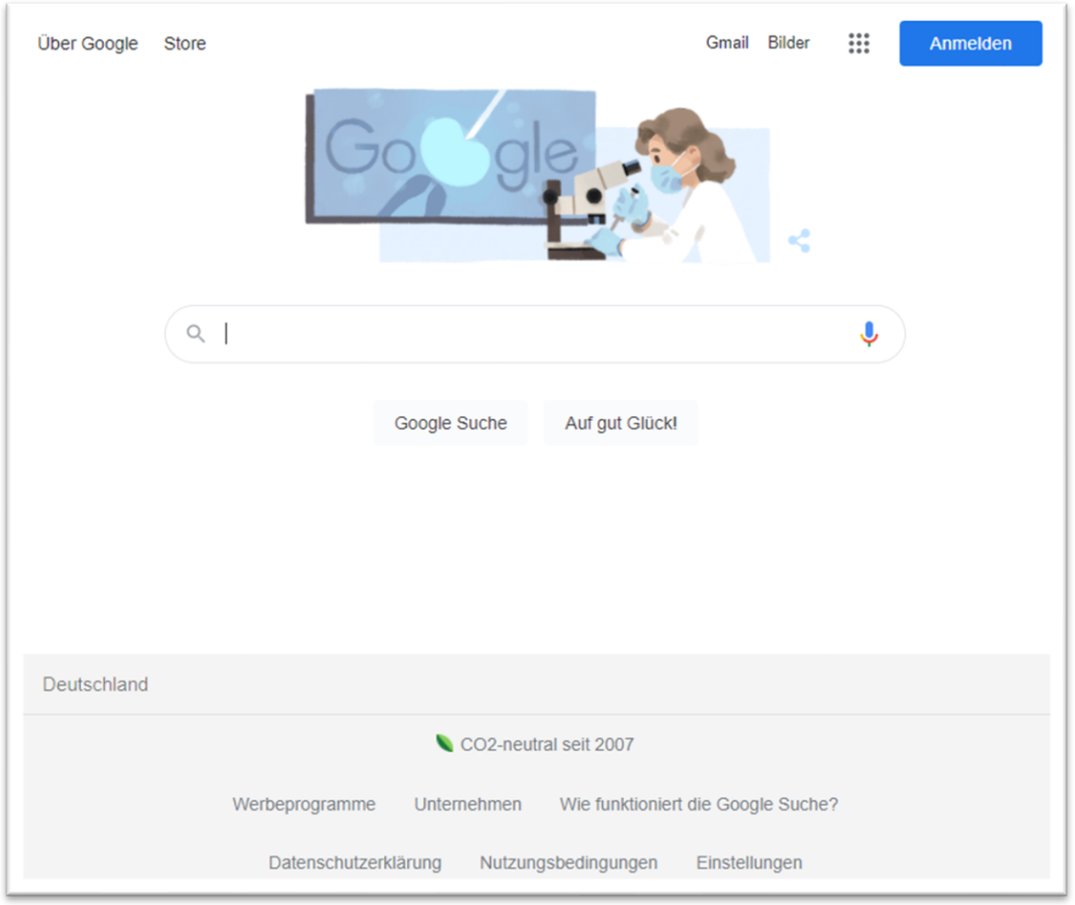


Module 3: HTML, JS, CSS

Learning Goals

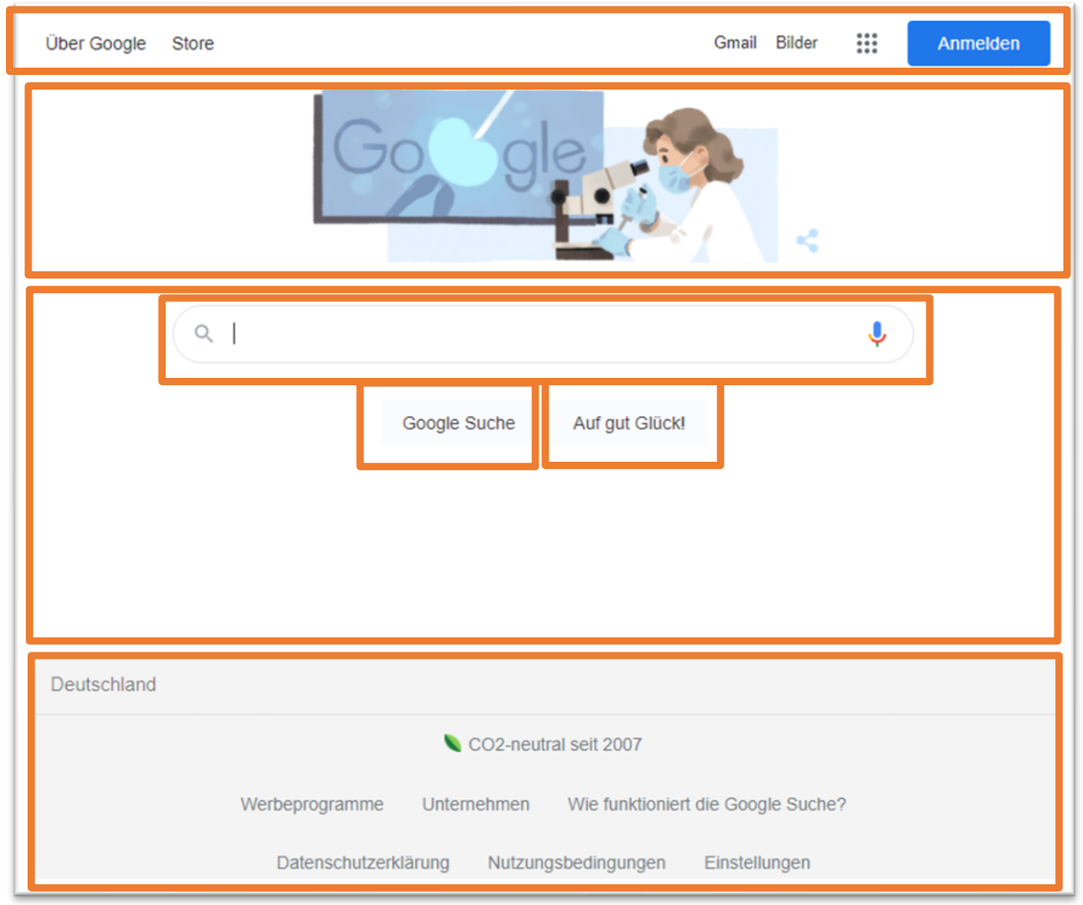
- Create an **old school** web app to cheer us up in difficult times.
- HTML structures
- *Basic* CSS styling rule
- Add interaction to an HTML page, creating new elements, modifying existing ones
- Debugging front-end code

The Browser: Behind the Scenes



Every web page that we visit is rendered by the browser using a combination of the following three technologies:

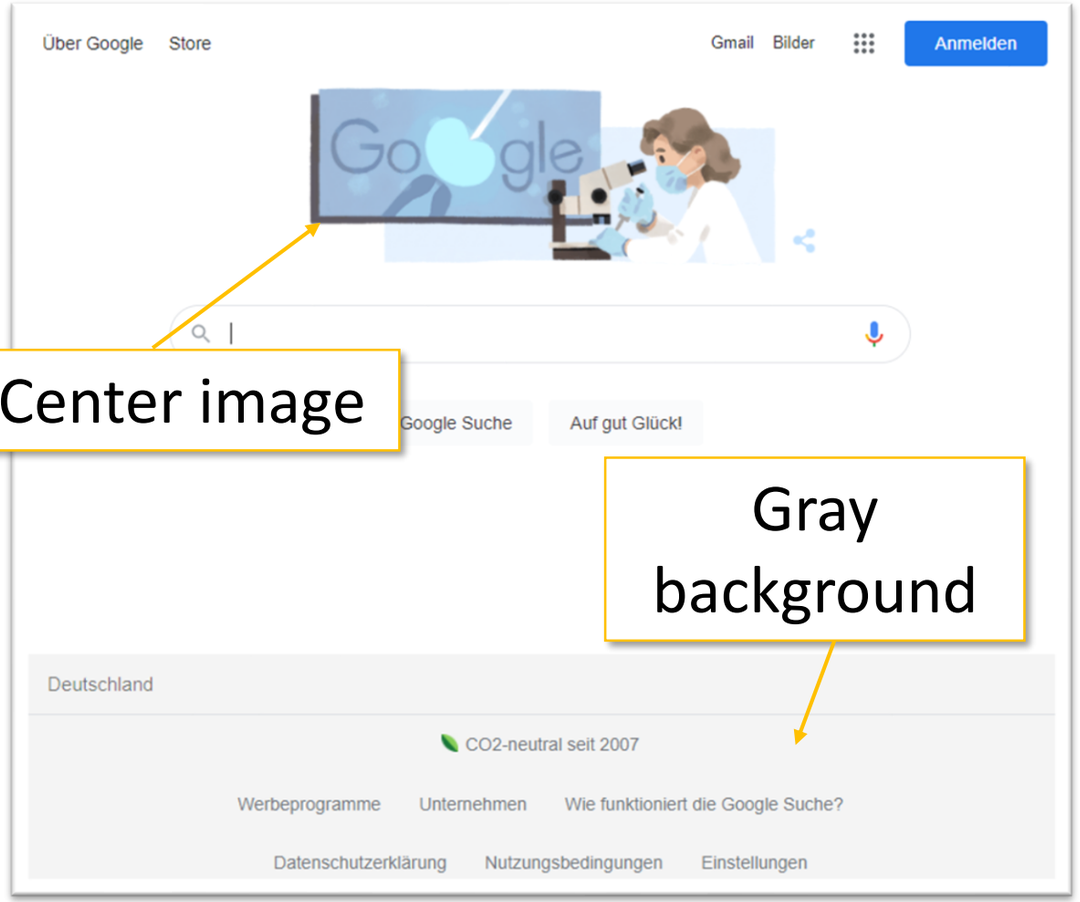
The Browser: Behind the Scenes



Every web page that we visit is rendered by the browser using a combination of the following three technologies:

Structure and Content

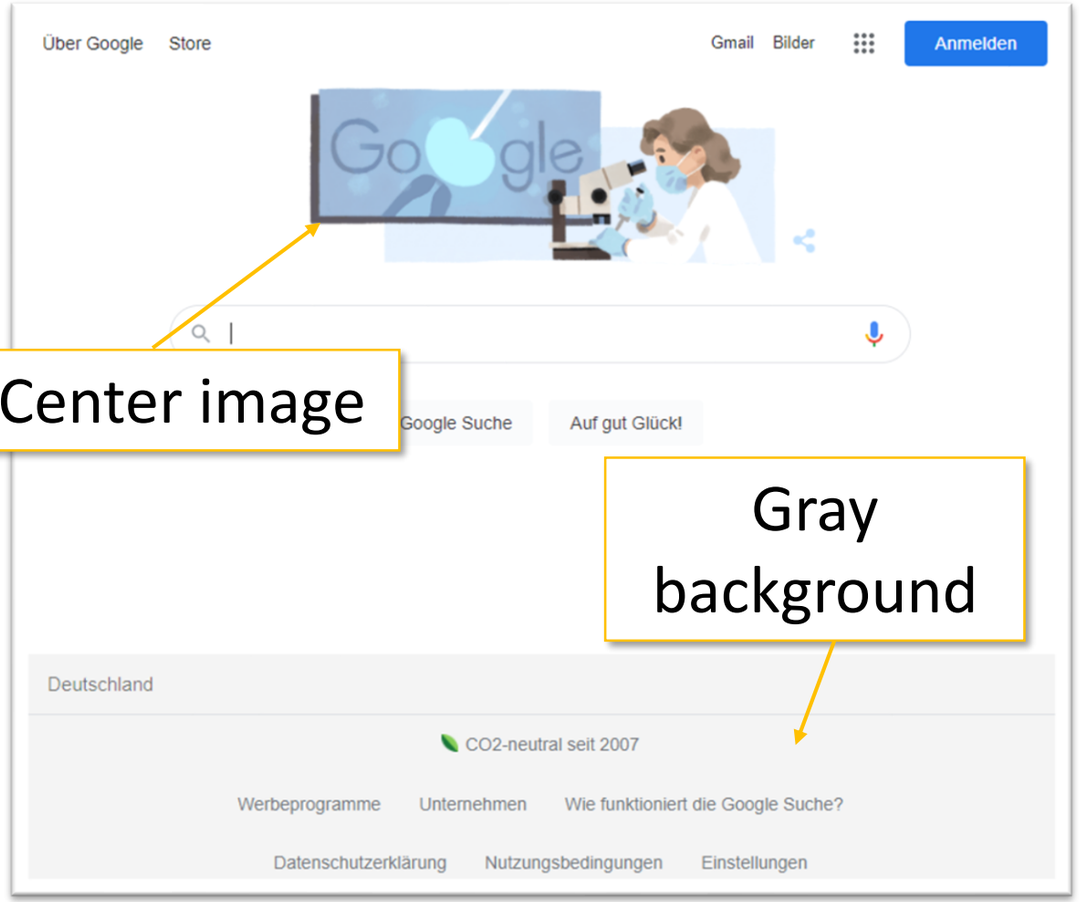
The Browser: Behind the Scenes



Every web page that we visit is rendered by the browser using a combination of the following three technologies:

Styling and simple interactions

The Browser: Behind the Scenes



Every web page that we visit is rendered by the browser using a combination of the following three technologies:

Complex interactions, communication with remote servers, logging, tracking, etc.

Some of the HTML Page's Inhabitants

DOM Tree

```
<HTML>
   <HEAD>
      <LINK>
      <SCRIPT>
   </HEAD>
   <BODY>
   …
   </BODY>
</HTML>
```

Presentation Tags

<P>

<DIV>

<SPAN>

Images and Links

<IMG>

<A>

Forms

<INPUT>

<TEXTAREA>

Attributes

<DIV id=“header”>

<SPAN class=“bold”>

<IMG SRC=“image.jpg” />

<A HREF=“newpage.htm”>

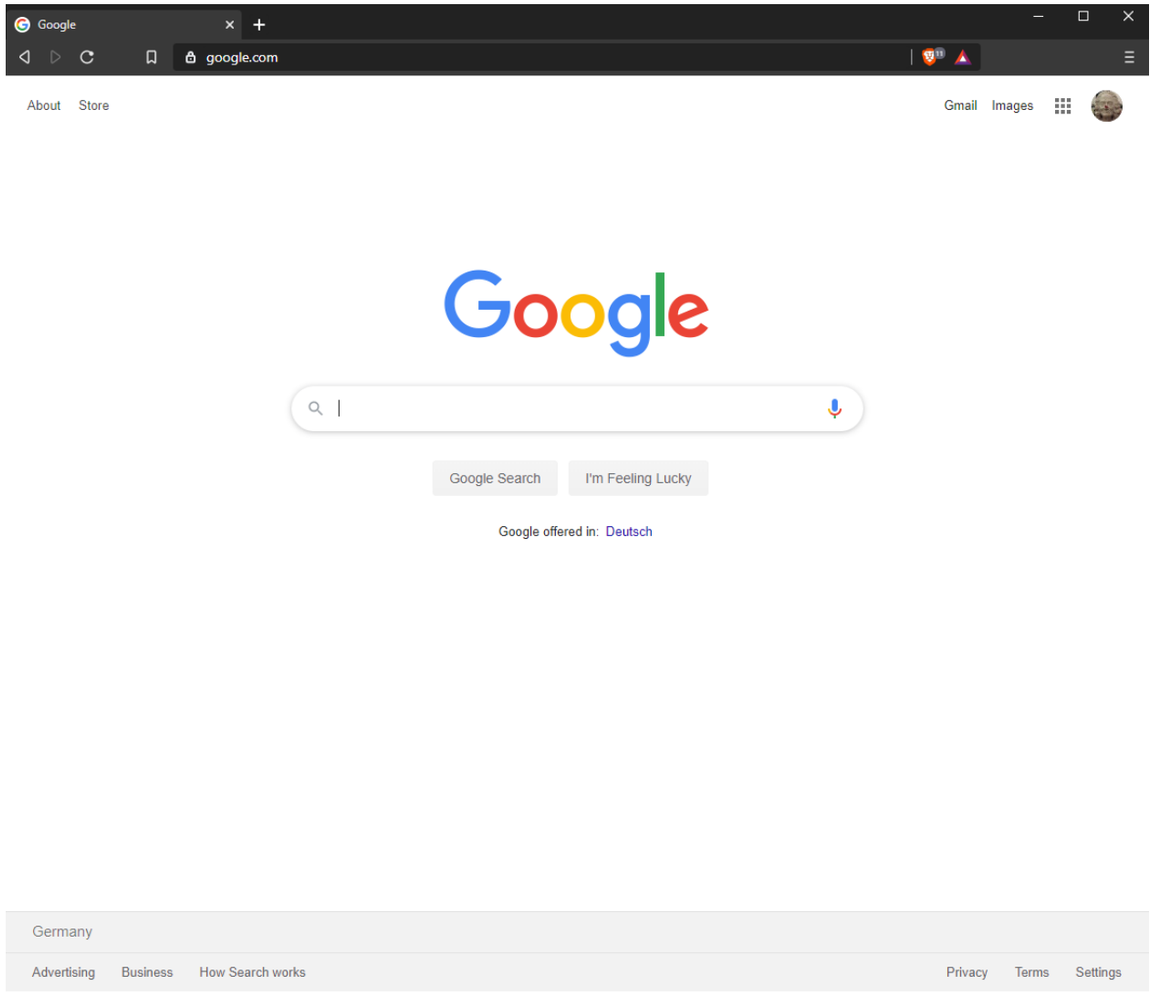
<INPUT disabled>

CSS Declarations

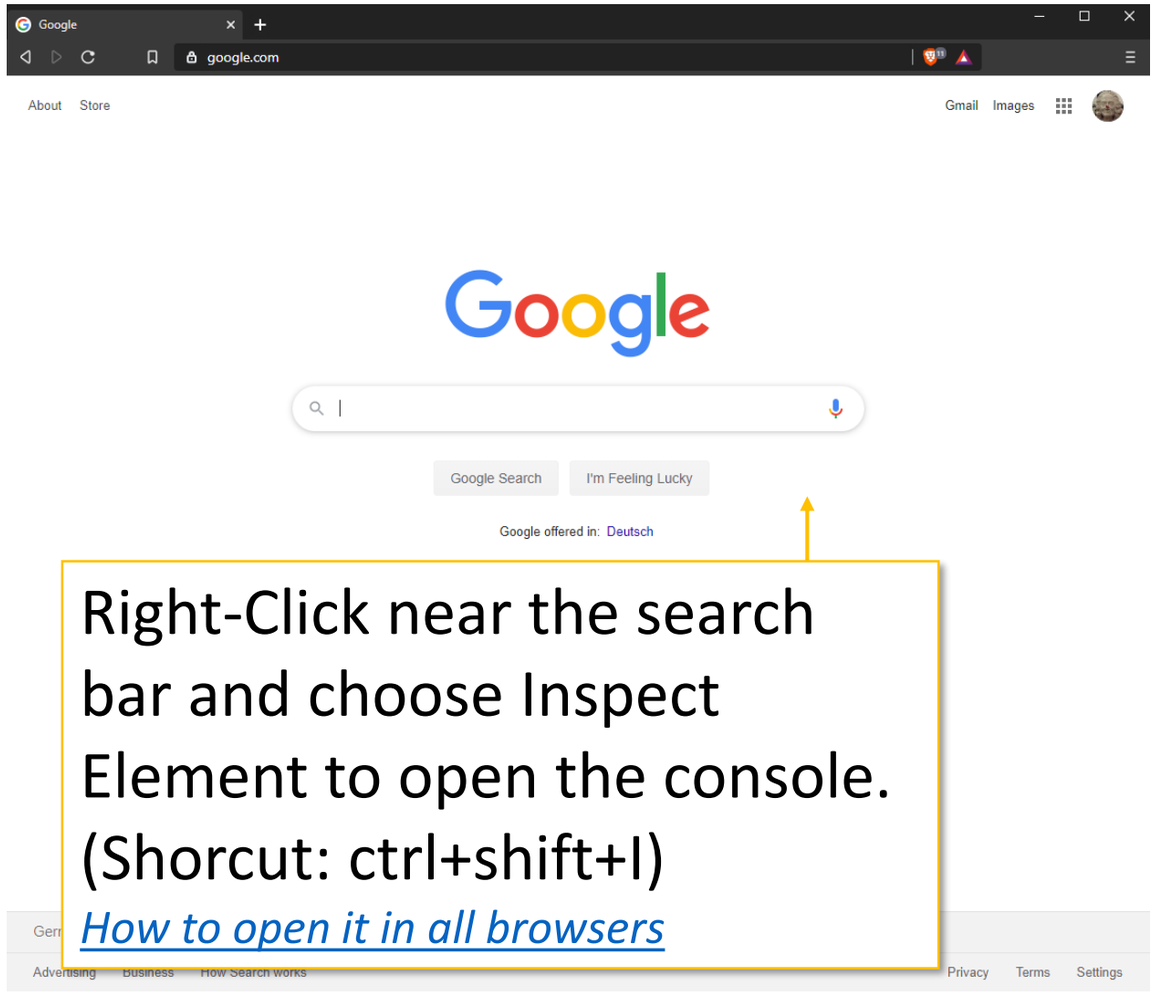
.bold { font-weight: bold };

#header { width: 600px };

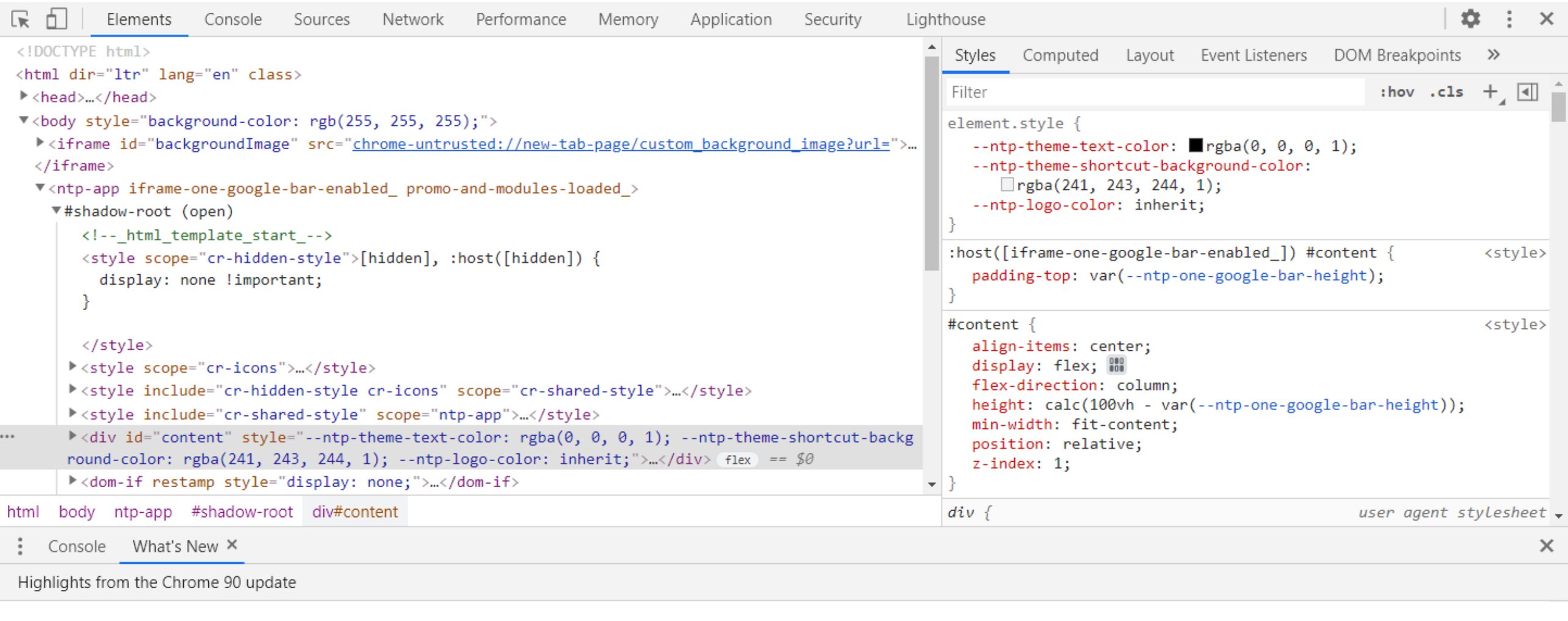
HTML, CSS, JavaScript



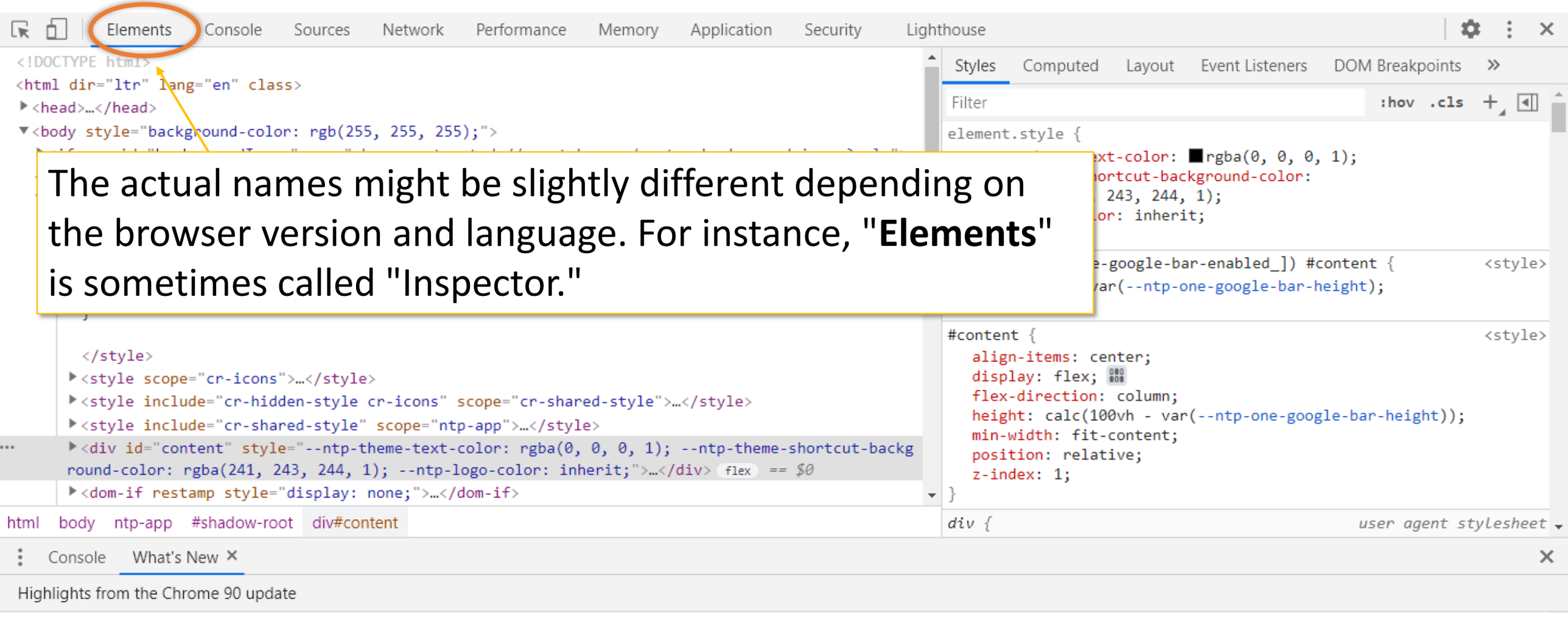
HTML, CSS, JavaScript



Developer Tools: Elements

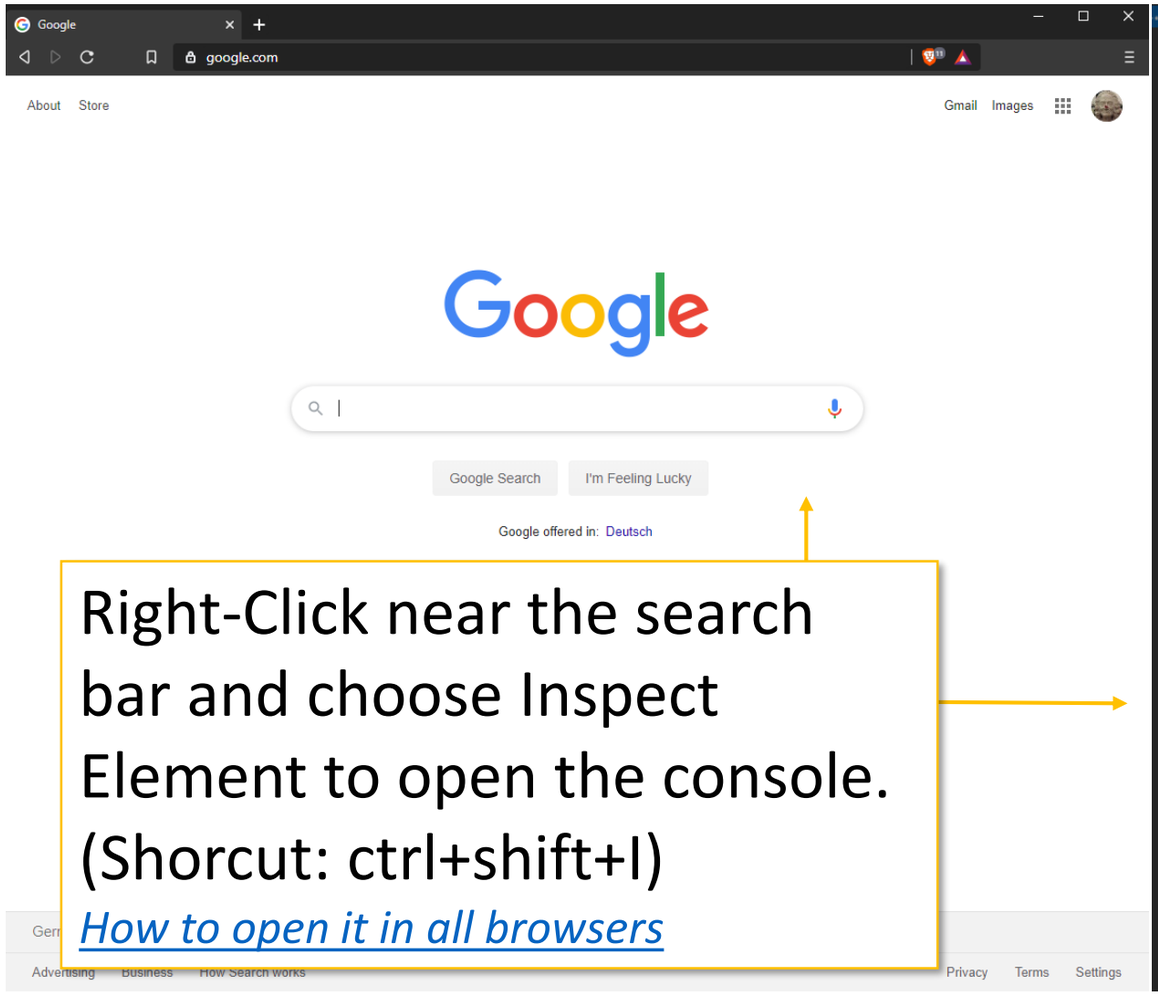


Developer Tools: Elements



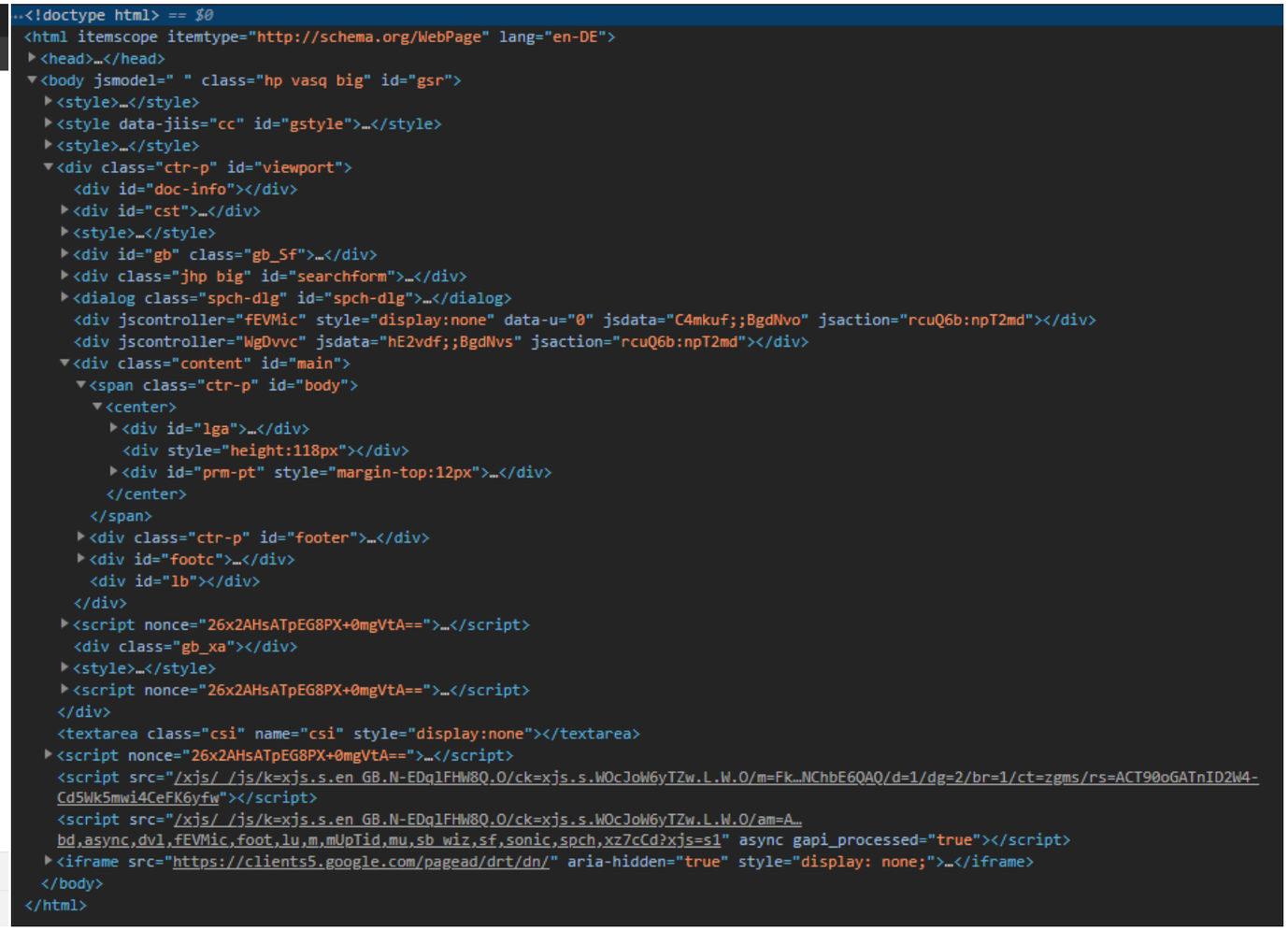
The actual names might be slightly different depending on the browser version and language. For instance, "**Elements**" is sometimes called "Inspector."

HTML, CSS, JavaScript

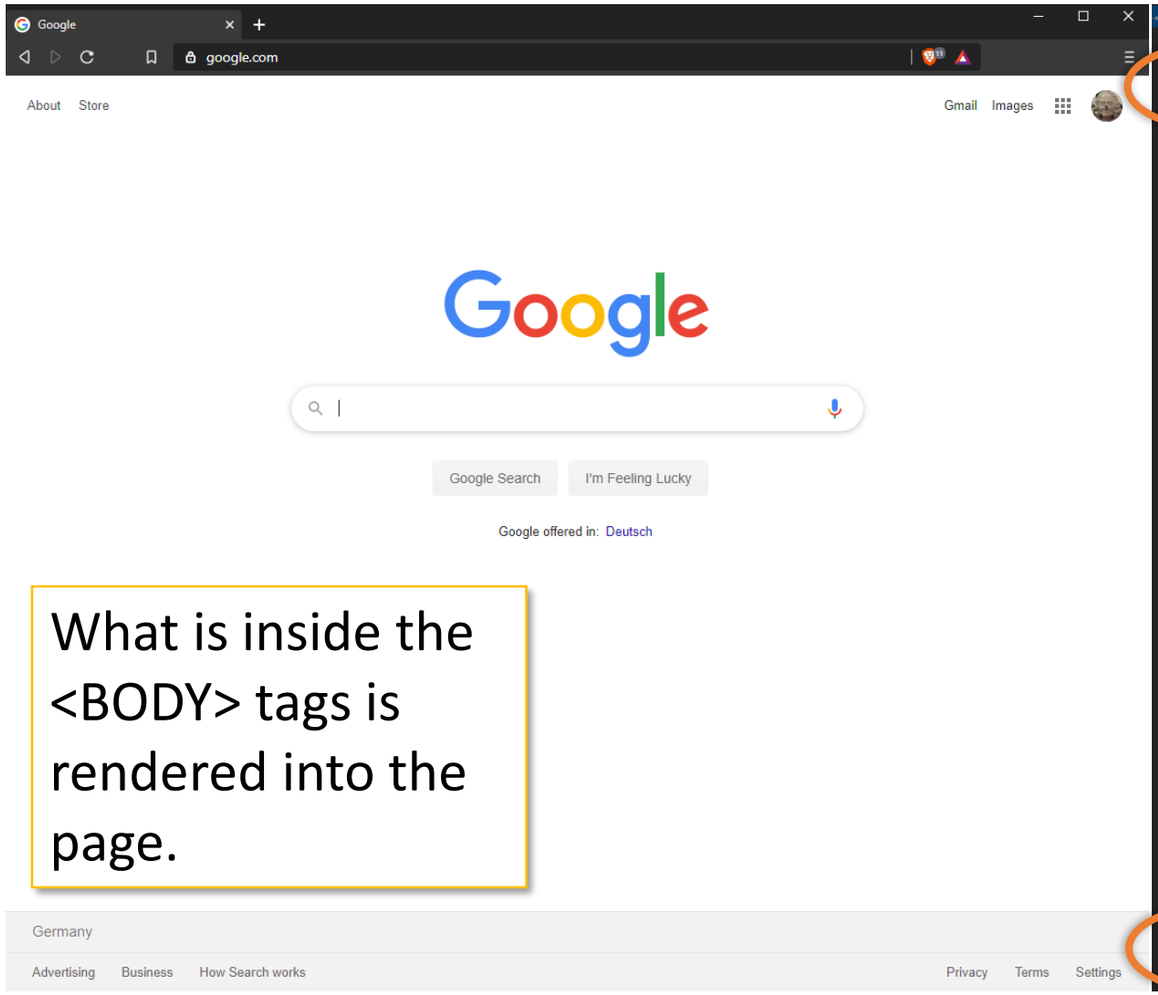


Right-Click near the search bar and choose Inspect Element to open the console. (Shorcut: ctrl+shift+I)

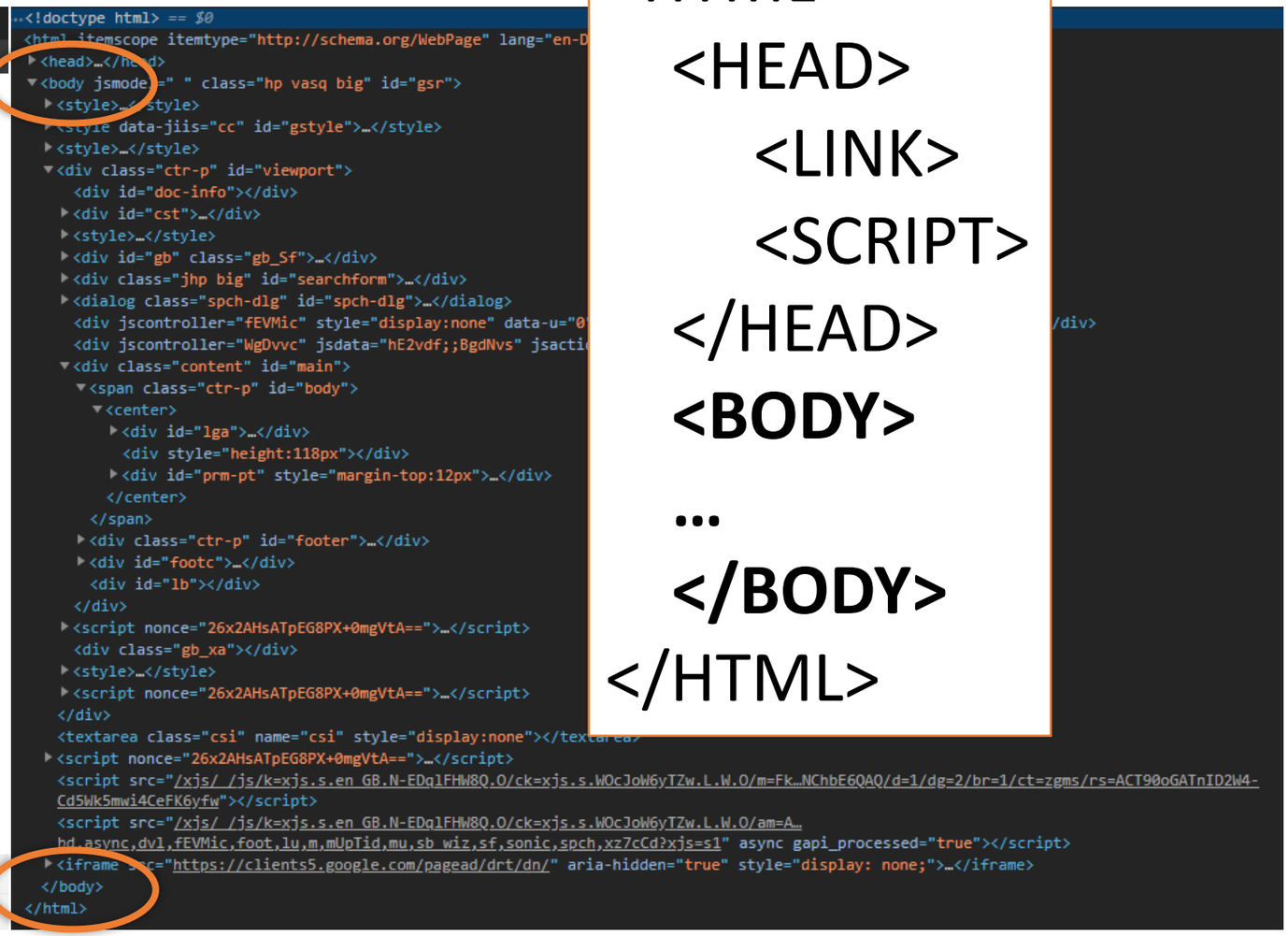
How to open it in all browsers



HTML, CSS, JavaScript



What is inside the <BODY> tags is rendered into the page.



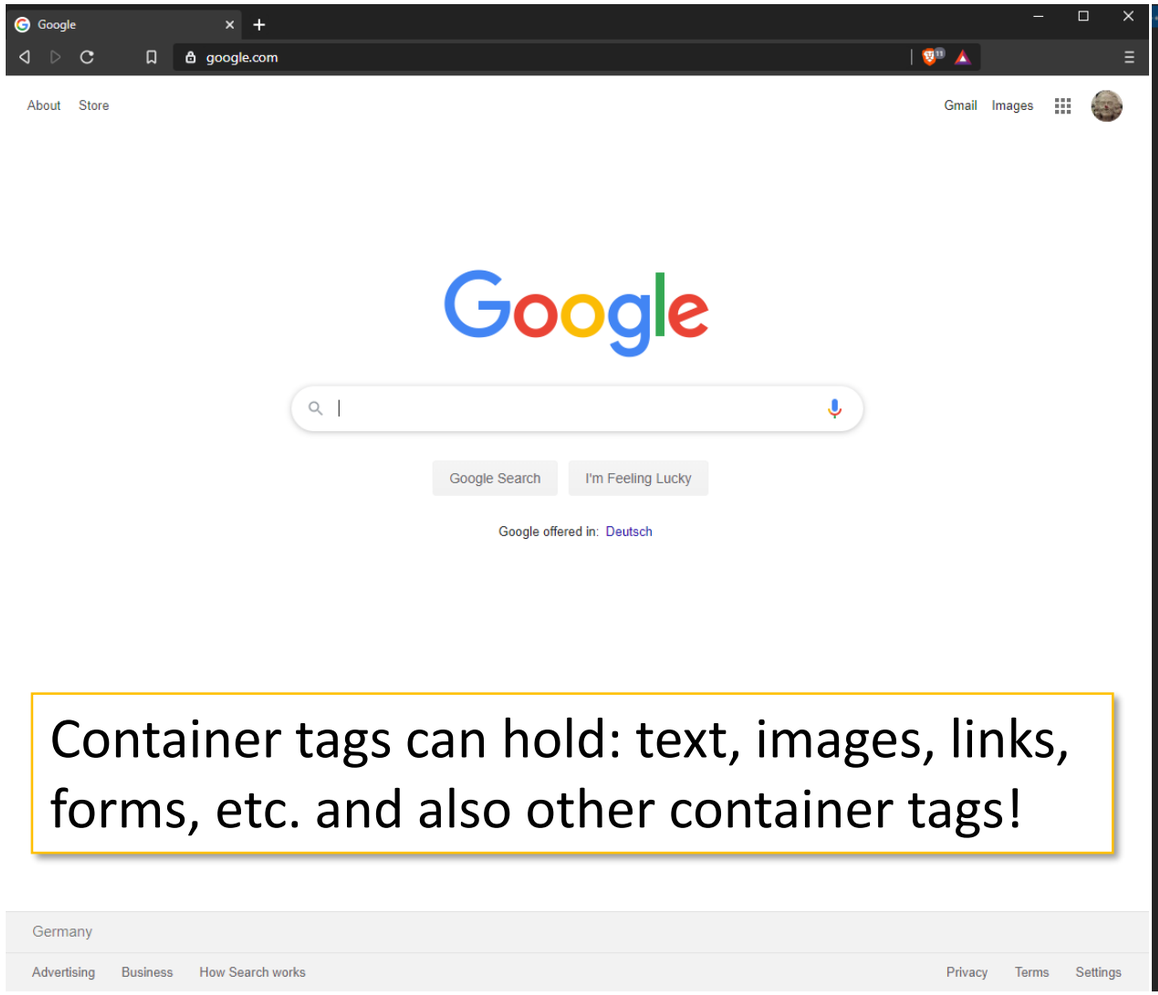
DOM Tree

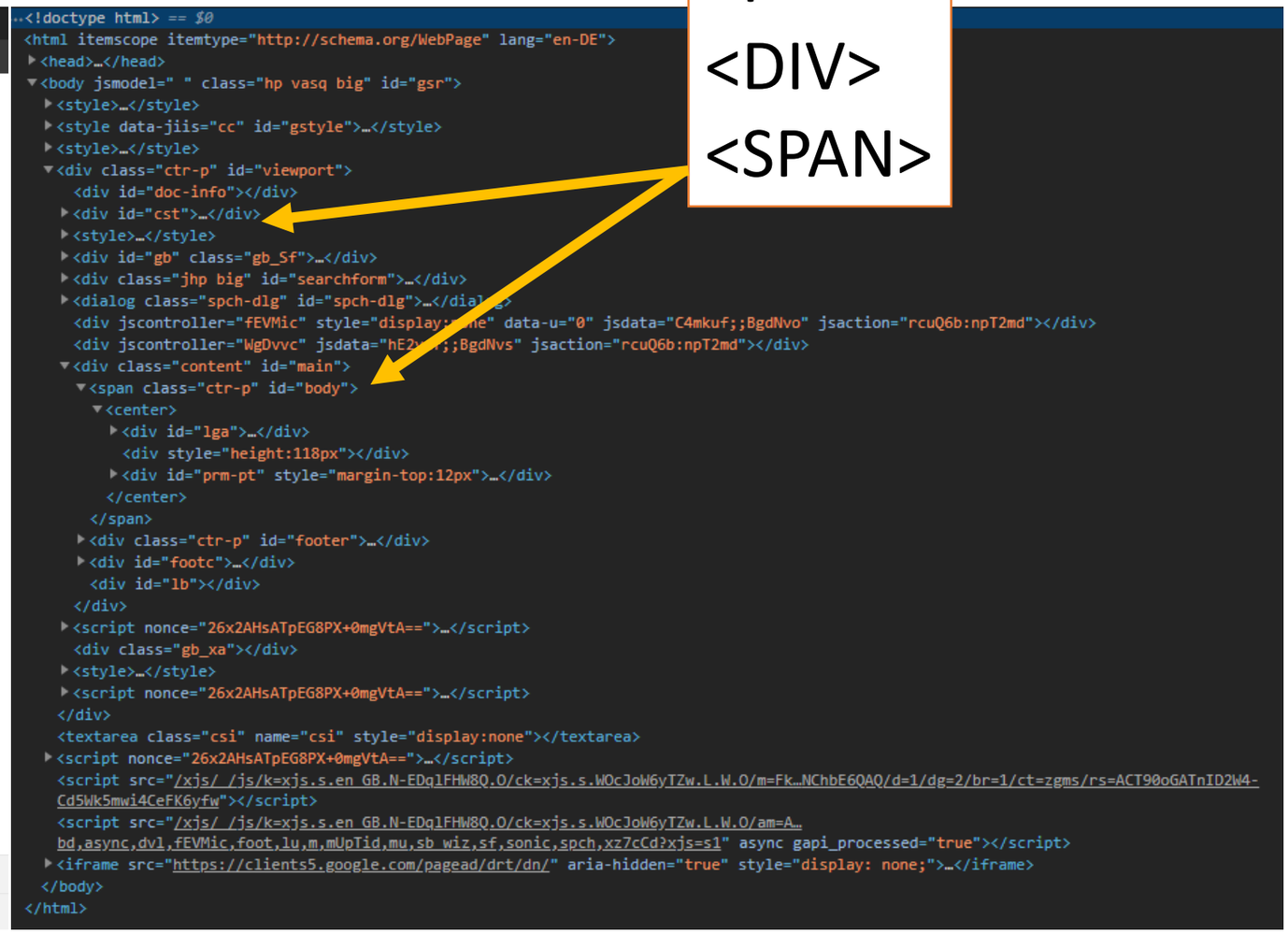
<HTML>
  <HEAD>
    <LINK>
    <SCRIPT>
  </HEAD>
  <BODY>
  …
  </BODY>
</HTML>

HTML, CSS, JavaScript

Container Tags

<P>
<DIV>
<SPAN>





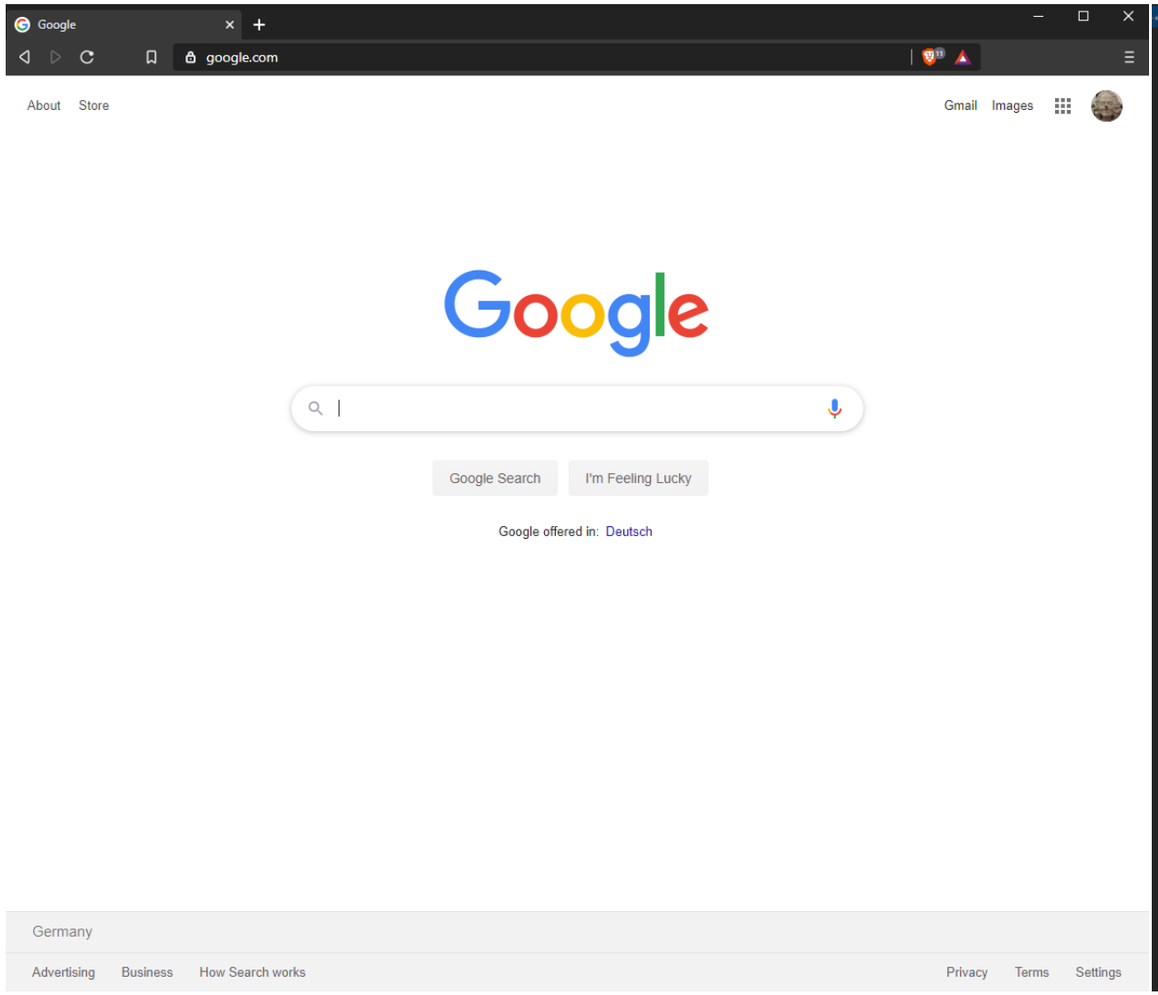
Container tags can hold: text, images, links, forms, etc. and also other container tags!

https://stackoverflow.com/questions/30879707/why-is-a-div-called-a-div-why-is-a-span-called-a-span

HTML, CSS, JavaScript

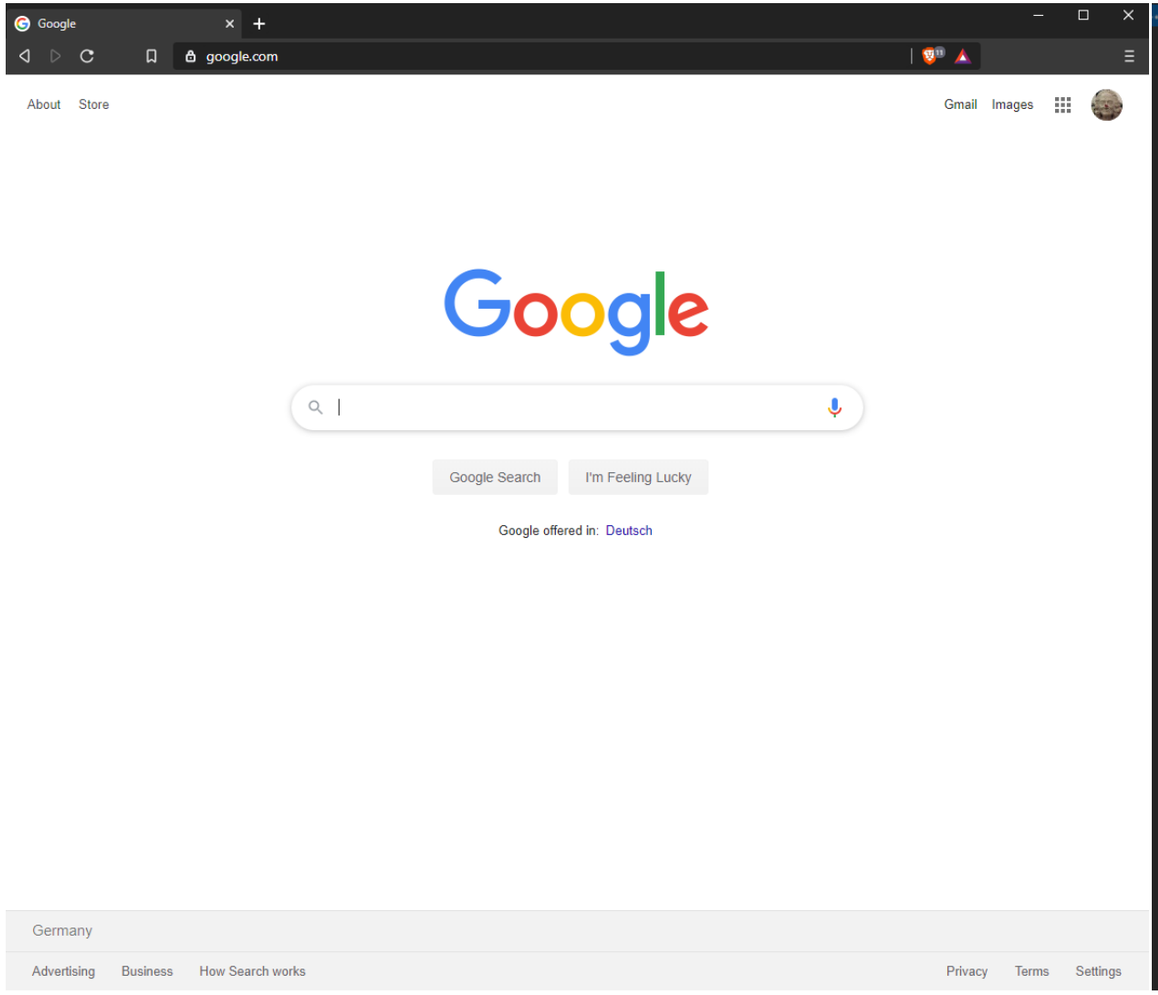
Style Tags

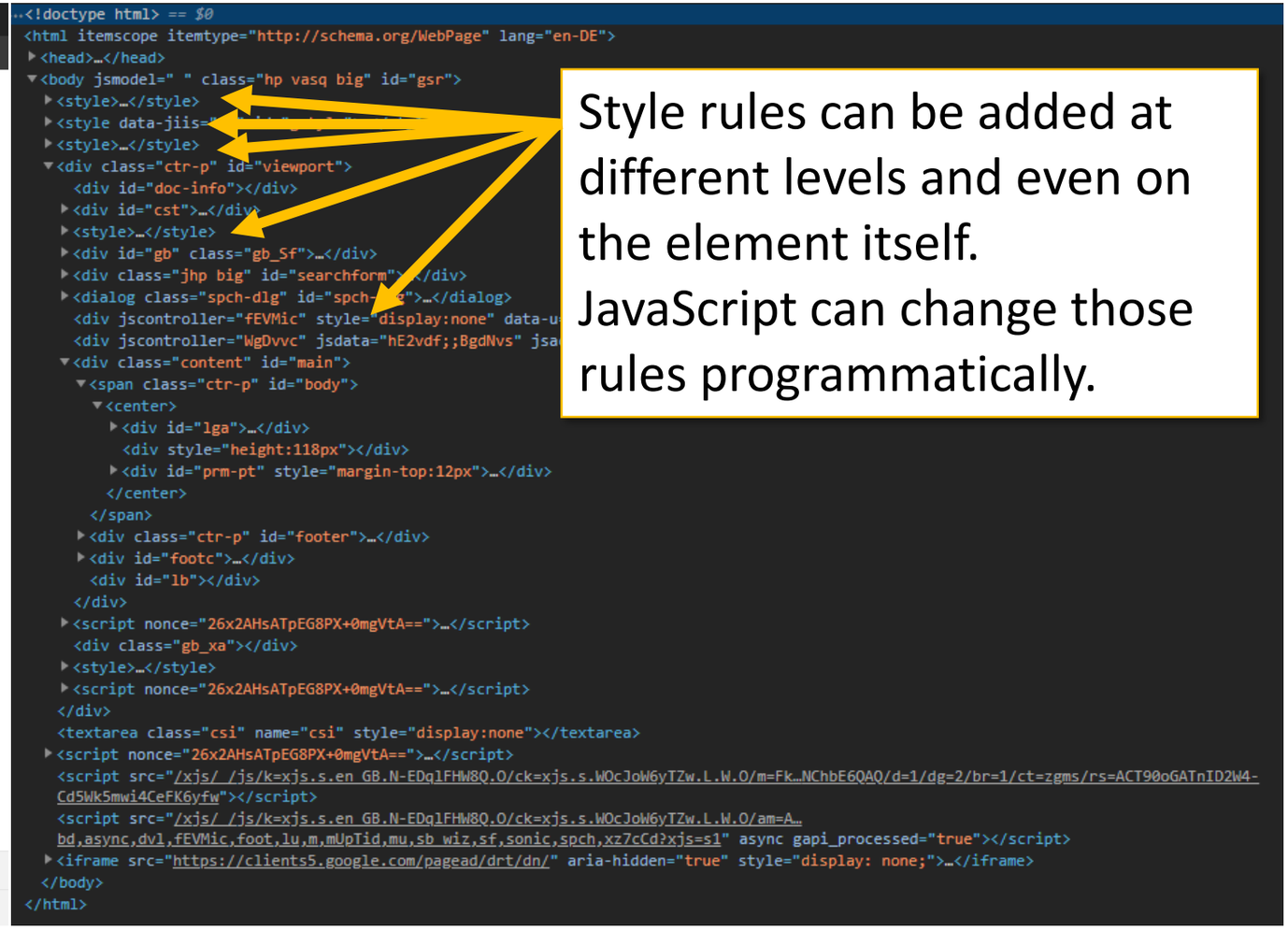
<STYLE>
<LINK>



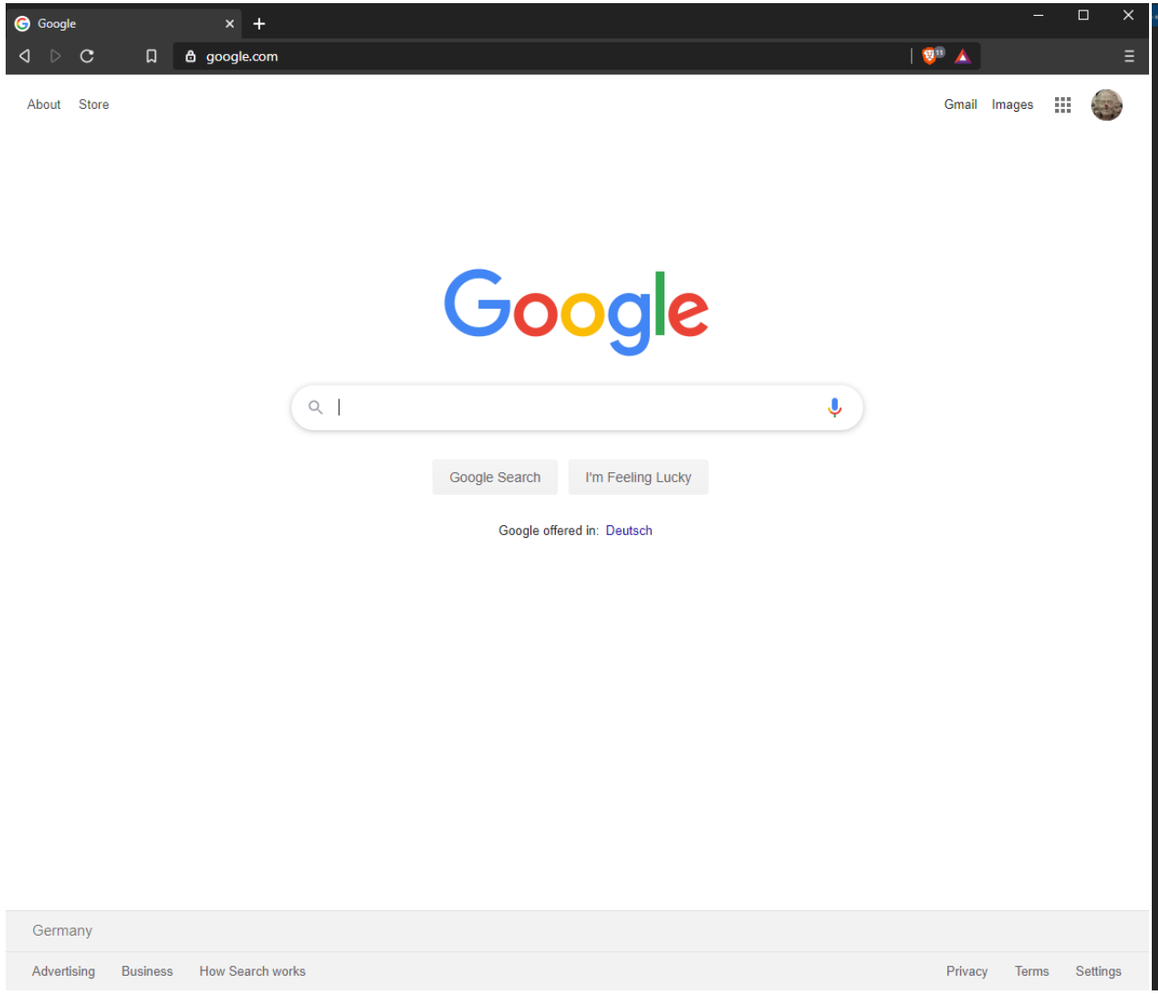

```
<!doctype html> == $0
<html itemscope itemtype="http://schema.org/WebPage" lang="en-DE">
 <head>…</head>
 <body jsmodel=" " class="hp vasq …">
  <style>…</style>
  <style data-jiis="cc" id="gstyle">…</style>
  <style>…</style>
  <div class="ctr-p" id="viewport">
   <div id="doc-info"></div>
   <div id="cst">…</div>
   <style>…</style>
   <div id="gb" class="gb_Sf">…</div>
   <div class="jhp big" id="searchform">…</div>
   <dialog class="spch-dlg" id="spch-dlg">…</dialog>
   <div jscontroller="fEVMic" style="display:none" data-u="0" jsdata="C4mkuf;;BgdNvo" jsaction="rcuQ6b:npT2md"></div>
   <div jscontroller="WgDvvc" jsdata="hE2vdf;;BgdNvs" jsaction="rcuQ6b:npT2md"></div>
   <div class="content" id="main">
    <span class="ctr-p" id="body">
     <center>
      <div id="lga">…</div>
      <div style="height:118px"></div>
      <div id="prm-pt" style="margin-top:12px">…</div>
     </center>
    </span>
    <div class="ctr-p" id="footer">…</div>
    <div id="footc">…</div>
    <div id="lb"></div>
   </div>
   <script nonce="26x2AHsATpEG8PX+0mgVtA==">…</script>
   <div class="gb_xa"></div>
   <style>…</style>
   <script nonce="26x2AHsATpEG8PX+0mgVtA==">…</script>
  </div>
  <textarea class="csi" name="csi" style="display:none"></textarea>
  <script nonce="26x2AHsATpEG8PX+0mgVtA==">…</script>
  <script src="/xjs/_/js/k=xjs.s.en_GB.N-EDqlFHW8Q.O/ck=xjs.s.WOcJoW6yTZw.L.W.O/m=Fk…NChbE6QAQ/d=1/dg=2/br=1/ct=zgms/rs=ACT90oGATnID2W4-Cd5Wk5mwi4CeFK6yfw"></script>
  <script src="/xjs/_/js/k=xjs.s.en_GB.N-EDqlFHW8Q.O/ck=xjs.s.WOcJoW6yTZw.L.W.O/am=A…bd,async,dvl,fEVMic,foot,lu,m,mUpTid,mu,sb_wiz,sf,sonic,spch,xz7cCd?xjs=s1" async gapi_processed="true"></script>
  <iframe src="https://clients5.google.com/pagead/drt/dn/" aria-hidden="true" style="display: none;">…</iframe>
 </body>
</html>
```

HTML, CSS, JavaScript

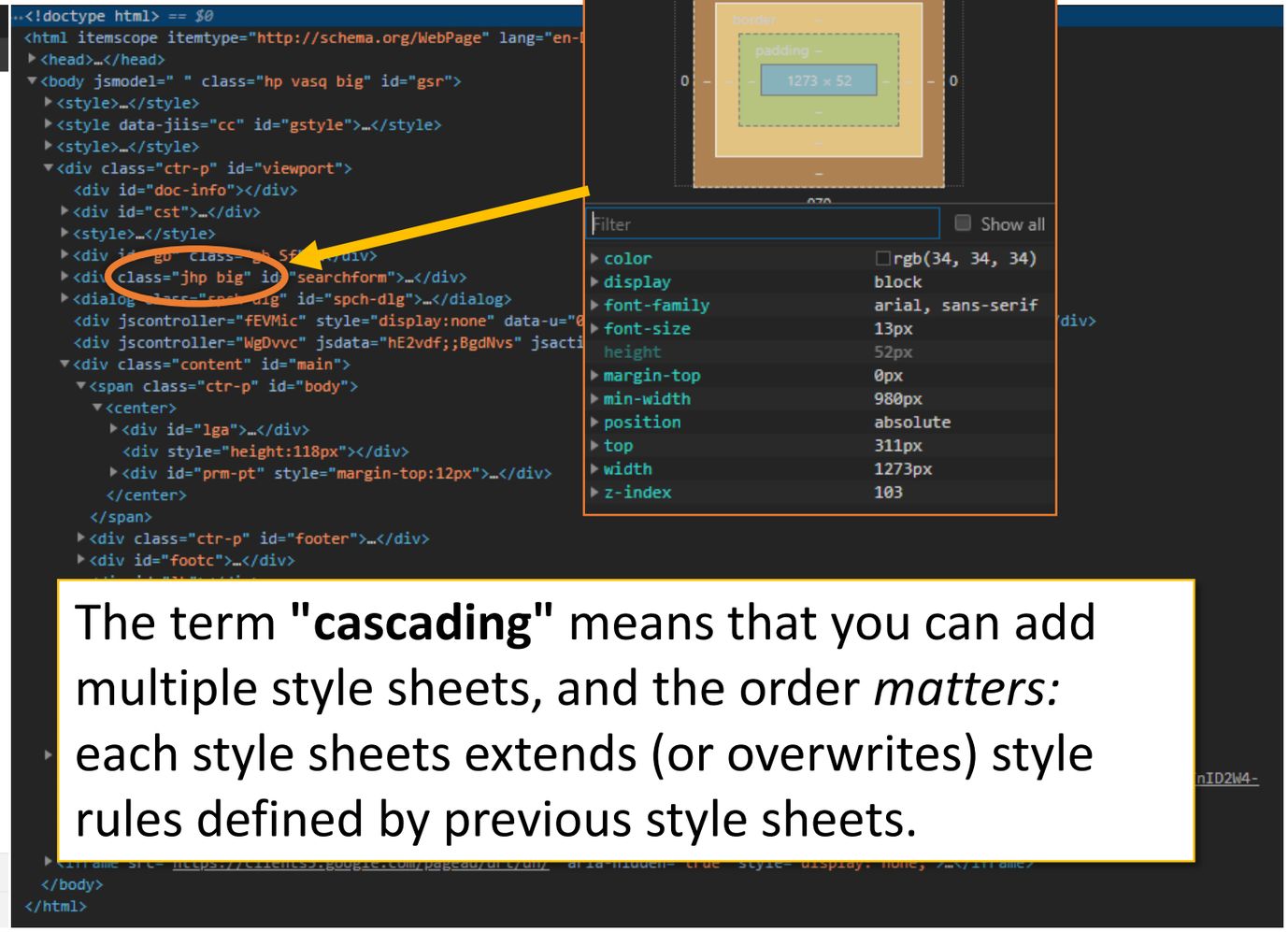




HTML, CSS, JavaScript



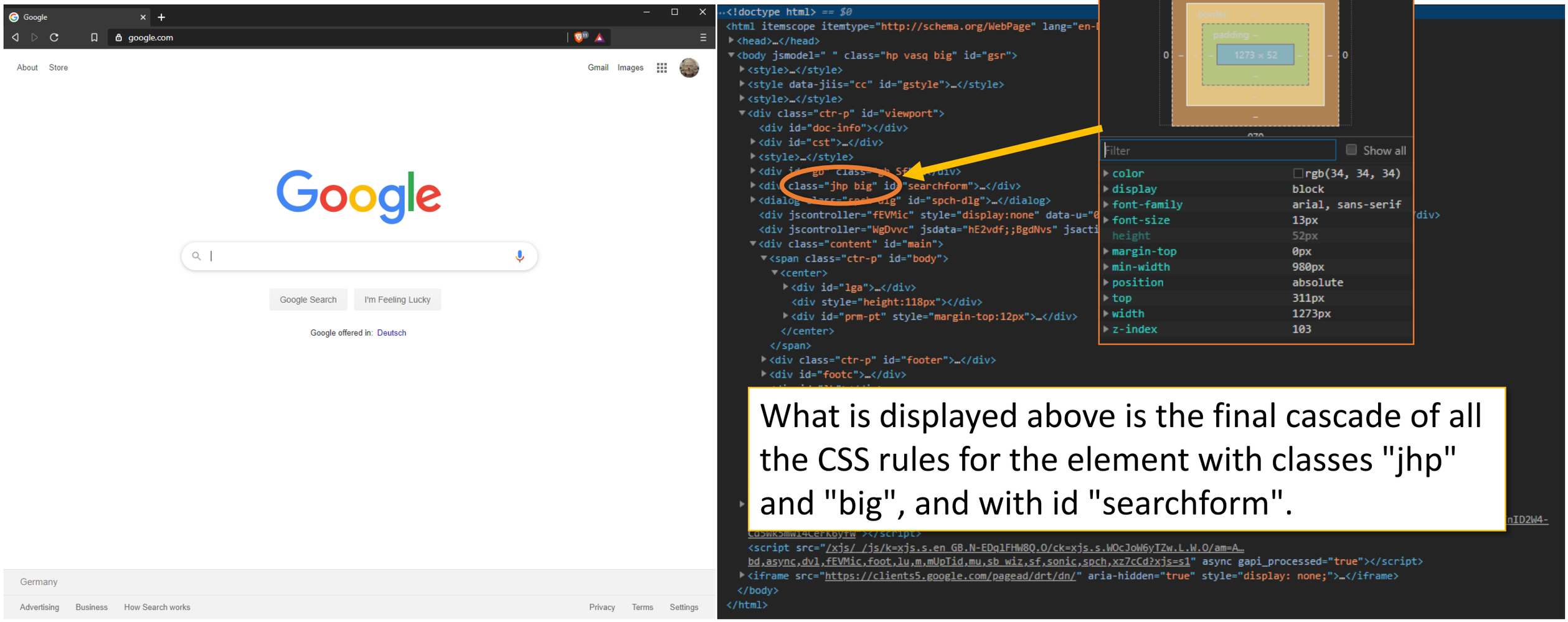
CSS (Cascading Style Sheets)



The term **"cascading"** means that you can add multiple style sheets, and the order *matters:* each style sheets extends (or overwrites) style rules defined by previous style sheets.

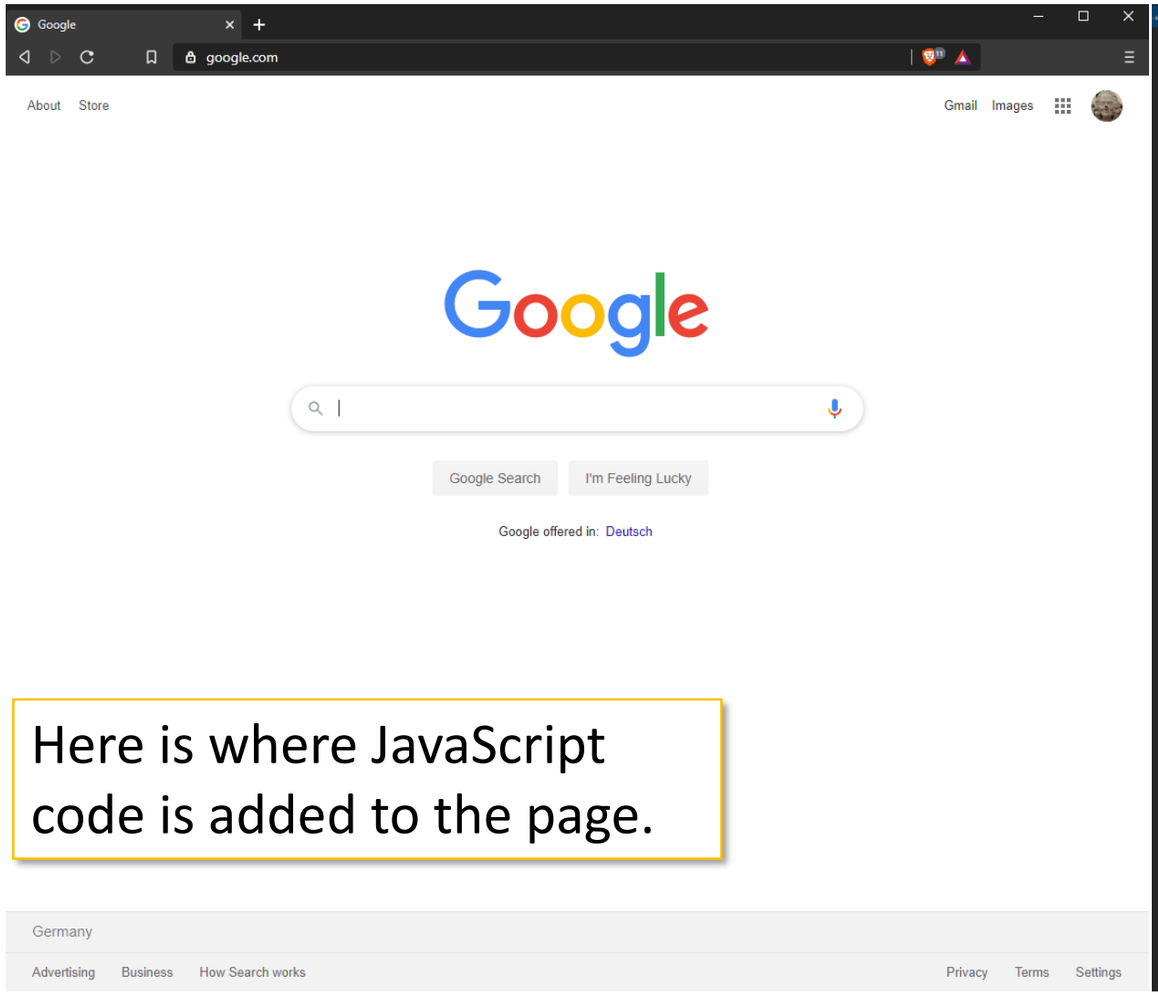
HTML, CSS, JavaScript

CSS (Cascading Style Sheets)

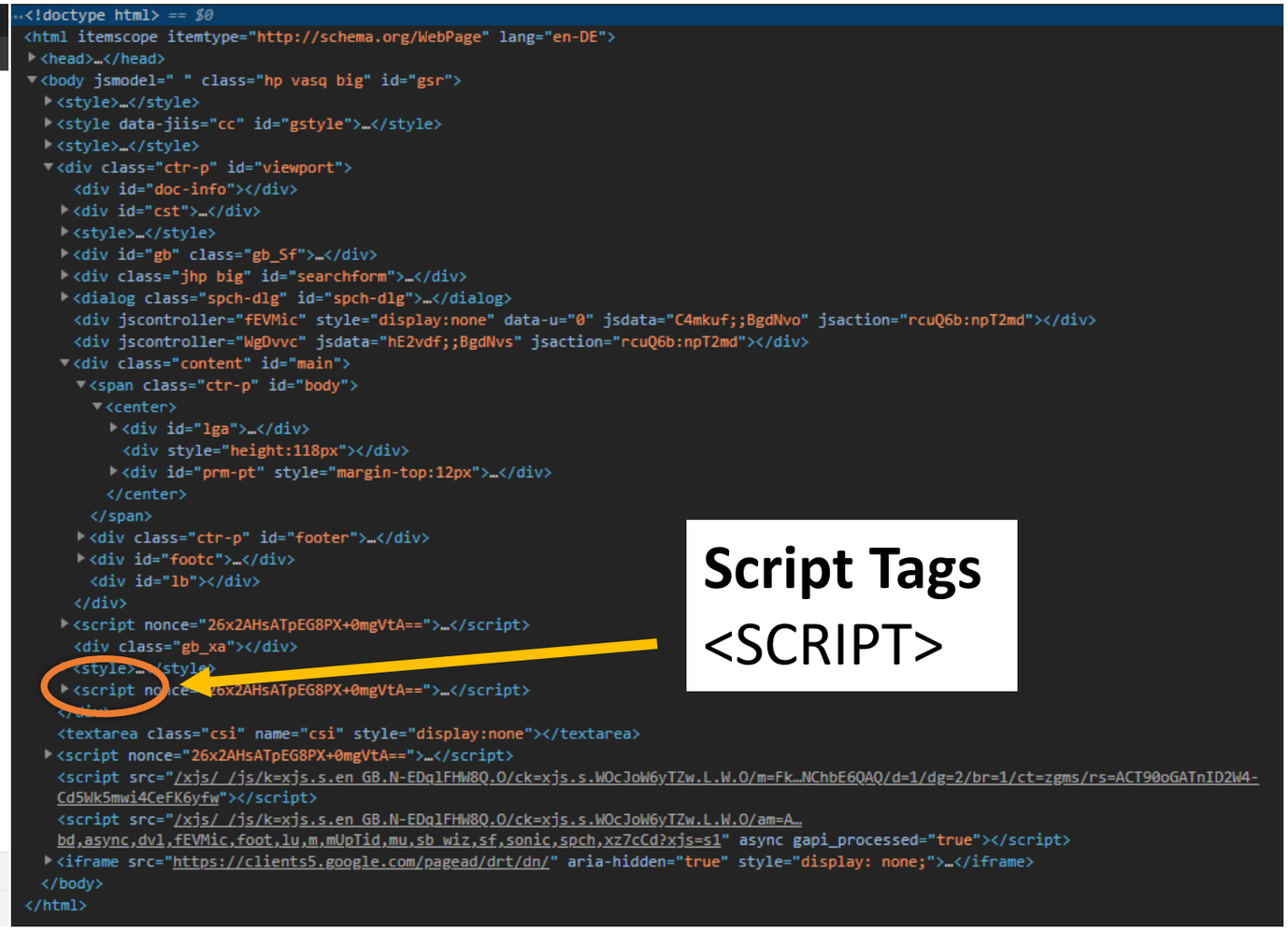


What is displayed above is the final cascade of all the CSS rules for the element with classes "jhp" and "big", and with id "searchform".

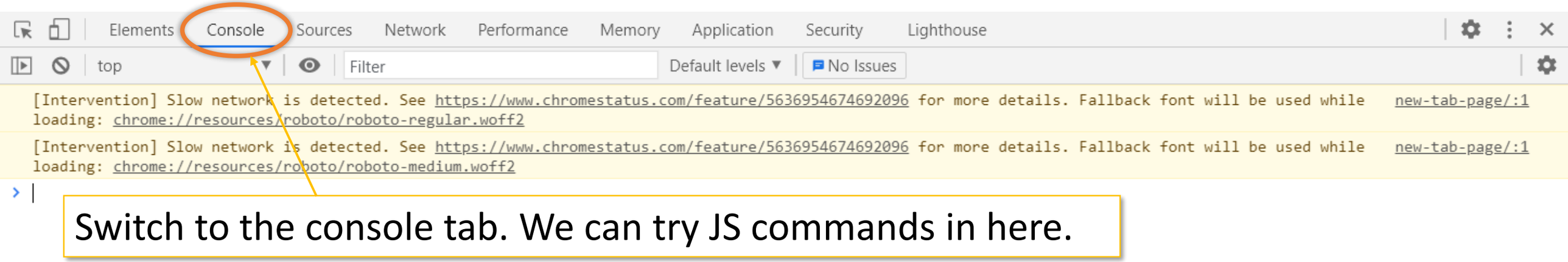
HTML, CSS, JavaScript



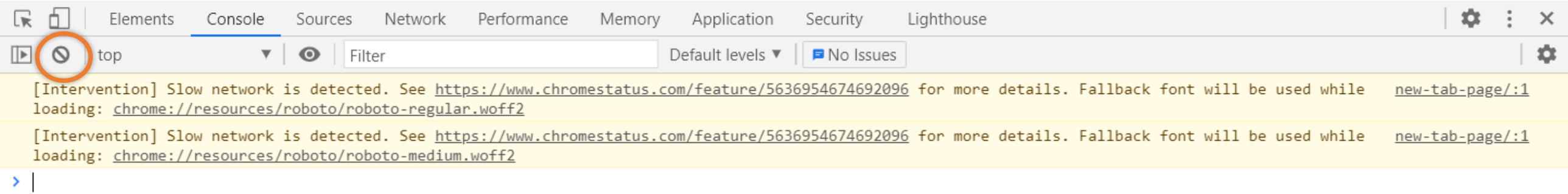
Here is where JavaScript code is added to the page.



Developer Tools: Console



Developer Tools: Console



Clear any pre-existing output: click on the button or type: `clear()`

Then, write something of your own with `console.log()`

```
console.log('Hello World');
```

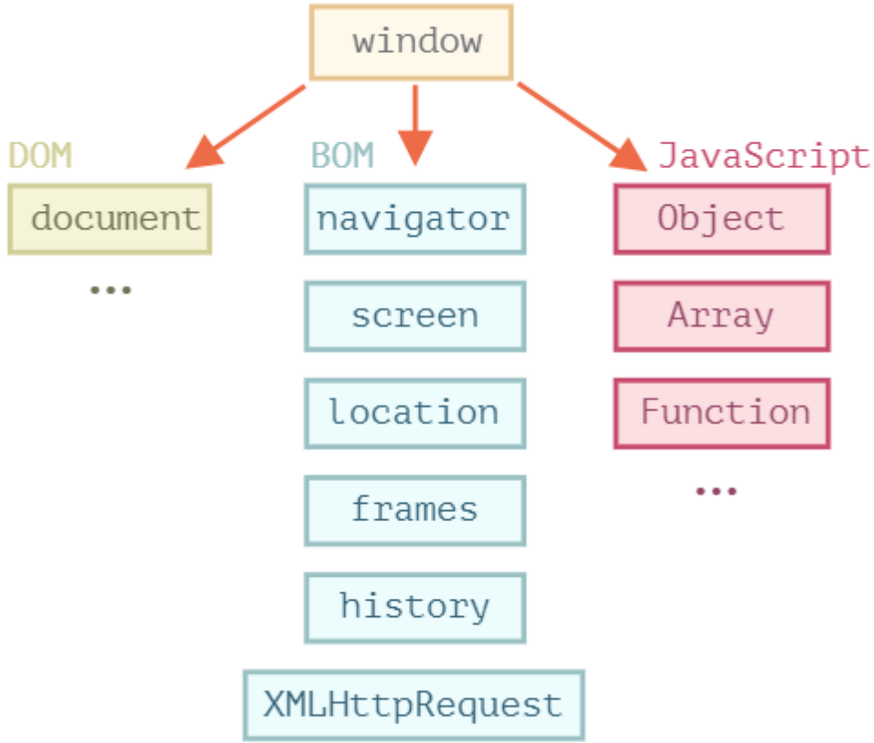
What Can JS in the Browser do?

Every JS object in the browser is child of the `window` object, including:

1. DOM (Document Object Model) objects (i.e., all things displayed on the page) exposed through the `document` object

2. Extra info and methods about the browser itself

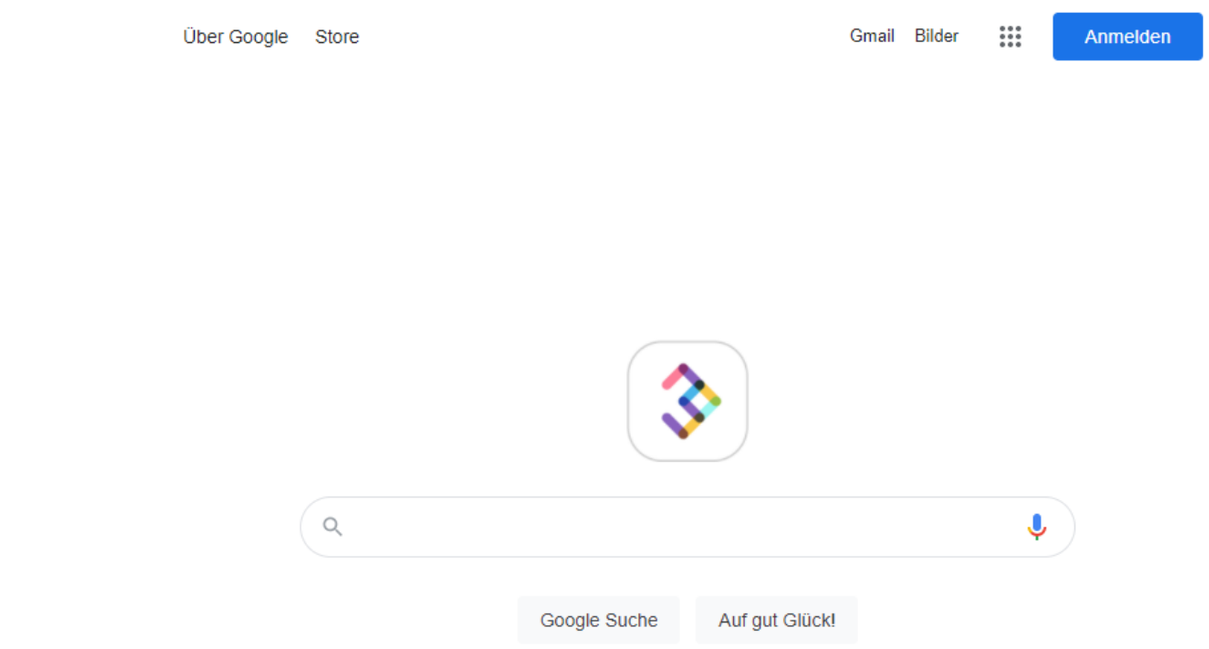
3. JavaScript language itself

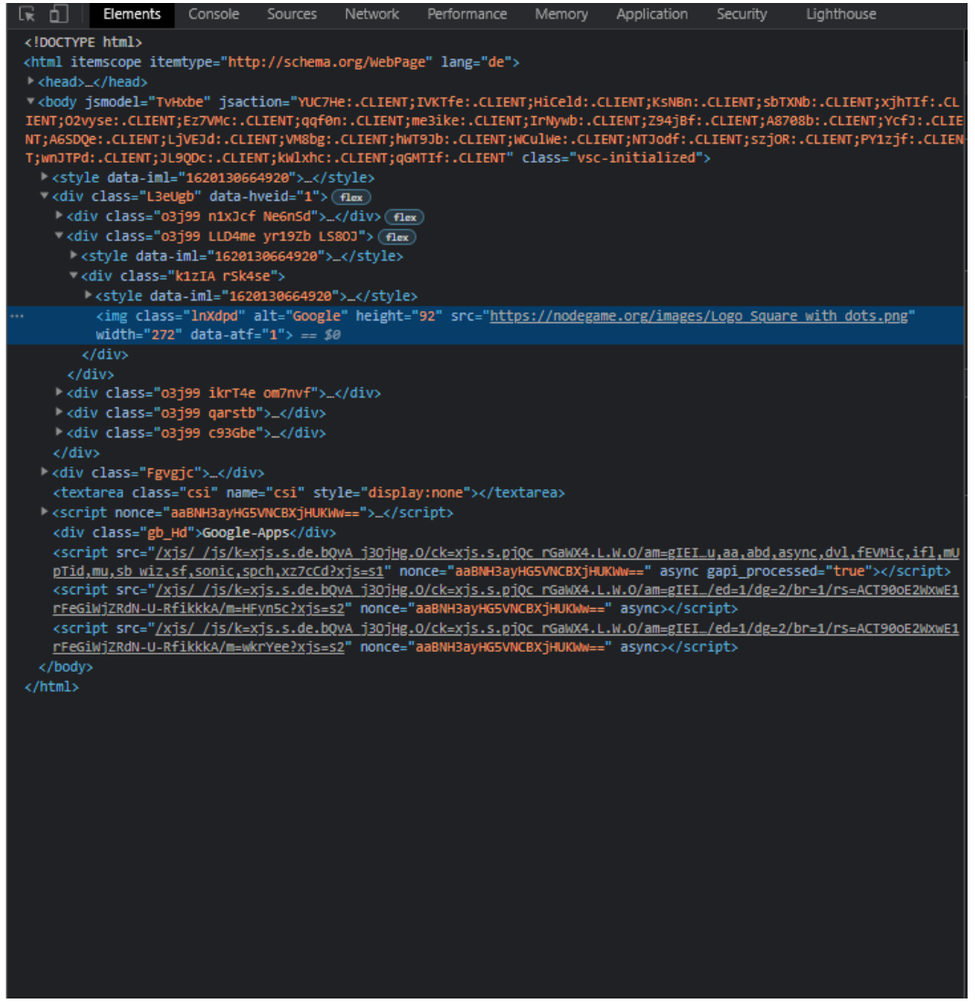


Img source: JavaScript.info

Hands On: Messing Around with Google.Com

Go to Google.com, open DevTools and change the logo to something else.





Hands On: Messing Around with Google.Com

Let's manipulate the page elements programmatically:

```
// Locate the HTML element with given id.
let logo = document.getElementById("logo");
```

If Google shows a special logo, you should check the DOM for the right ID/class. For instance, this command might be an alternative:

let logo = document.querySelector('.lnXdpd');

Hands On: Messing Around with Google.Com

Changes in the Inspector are immediately reflected on the page.

For example, if add a rule:

"display: none"

the selected element will be hidden in the page.

Hands On: Messing Around with Google.Com

Go to Google.com and manipulate the page elements programmatically:

```
// Locate the HTML element holding with given id.
var logo = document.getElementById("logo");
```

How to change the image displayed?

Hands On: Messing Around with Google.Com

Go to Google.com and manipulate the page elements programmatically:

```
// Locate the HTML element holding with given id.
var logo = document.getElementById("logo");
```

How to change the image displayed?

DOM objects are glorified JavaScript objects with properties and methods.
The browser reads those properties and displays them accordingly.

Hands On: Messing Around with Google.Com

Go to Google.com and manipulate the page elements programmatically:

```
// Locate the HTML element holding with given id.
var logo = document.getElementById("logo");
// Change one of its attributes (pick any image you like).
logo.srcset = "https://nodegame.org/images/Logo_Square_with_dots.png";
```

Google does thousands of A/B testing, so the exact name of the property might be slightly different from mine. If not working, try setting `srcset` to null, and set the property `src`.

Hands On: Messing Around with Google.Com

Go to Google.com and manipulate the page elements programmatically:

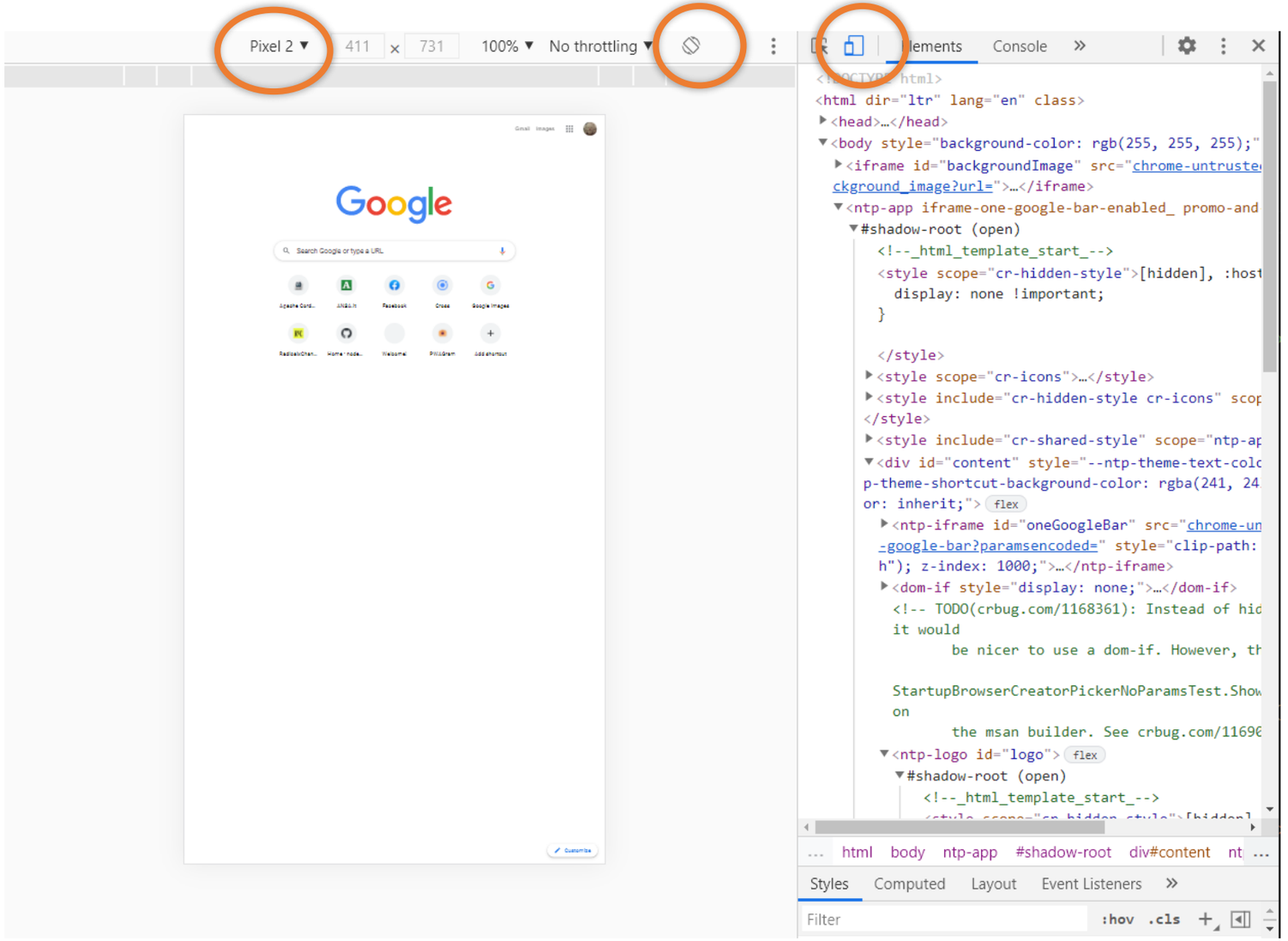
```
// Locate the HTML element holding with given id.
var logo = document.getElementById("hplogo");
// Change one of its attributes (pick any image you like).
logo.srcset = "https://nodegame.org/images/Logo_Square_with_dots.png";
// Defines an onclick event-handler (anonymous function).
logo.onclick = function() {

                    Let's do Something Here!

};
```

Hands On: Messing Around with Google.Com

Go to Google.com and manipulate the page elements programmatically:

```
// Locate the HTML element holding with given id.
var logo = document.getElementById("hplogo");
// Change one of its attributes (pick any image you like).
logo.srcset = "https://nodegame.org/images/Logo_Square_with_dots.png";
// Defines an onclick event-handler (anonymous function).
logo.onclick = function() {
    // Redirect to a new page using the location object.
    window.location.href = "https://nodegame.org";
};
```

Test in Mobile View



JavaScript: Creating new Elements

```
// Create a new DIV element.
let myDiv = document.createElement("div");
```

JavaScript: Creating new Elements

```
// Let's clear the page.
document.body.innerHTML = '';
```

JavaScript: Creating new Elements

```
// Create a new DIV element.
let myDiv = document.createElement("div");

// Add something inside.
myDiv.innerHTML = 'I am cool div.';
```

JavaScript: Creating new Elements

```
// Create a new DIV element.
let myDiv = document.createElement("div");

// Add something inside.
myDiv.innerHTML = 'I am cool div.';

// Add the element to the page.
document.body.appendChild(myDiv);
```

SPAN vs DIV

```
// Add a SPAN element inside our DIV.
let mySpan = document.createElement("span");
```

SPAN vs DIV

```
// Add a SPAN element inside our DIV.
let mySpan = document.createElement("span");
```

Note! <p> and are "block" elements, while is an inline element. *What does it mean?* If you append several elements, they will be displayed one *next to* the other; if you append several elements they will be displayed one *below* the other. So, generally <SPAN> elements are nested inside <DIV>, and not vice versa.

SPAN vs DIV

```
// Add a SPAN element inside our DIV.
let mySpan = document.createElement("span");
```

Note! <p> and are "block" elements, while is an inline element. *What does it mean?* If you append several elements, they will be displayed one *next to* the other; if you append several elements they will be displayed one *below* the other. So, generally <SPAN> elements are nested inside <DIV>, and not vice versa.

However, did you know that you can change this behavior with a CSS "display" rule?

SPAN vs DIV

```
// Add a SPAN element inside our DIV.
let mySpan = document.createElement("span");

// Add something inside.
mySpan.innerHTML = 'I am a child of myDiv.';
```

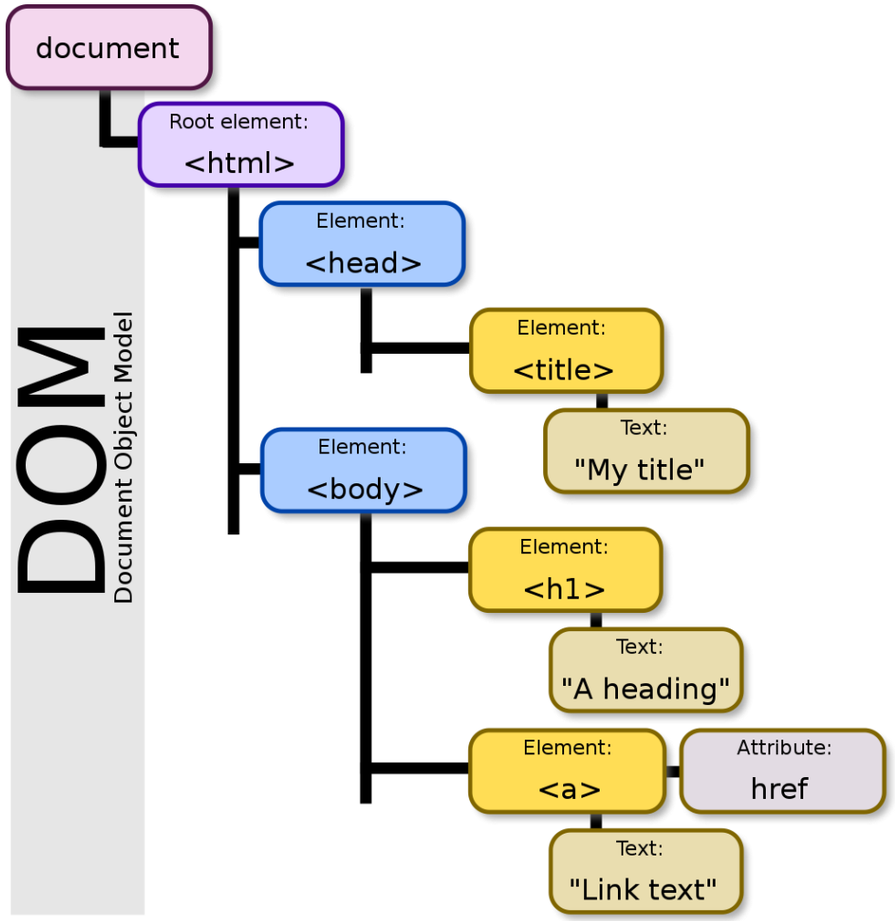
SPAN vs DIV

```
// Add a SPAN element inside our DIV.
let mySpan = document.createElement("span");

// Add something inside.
mySpan.innerHTML = 'I am a child of myDiv.';

// Add the element to the page.
myDiv.appendChild(mySpan);
```

More on appendChild

Every element in the DOM tree has a "parent" element and might have one or more "children" elements.

`appendChild` is a method available in every HTML element to add a *new* element **at the bottom** of the list of its children.

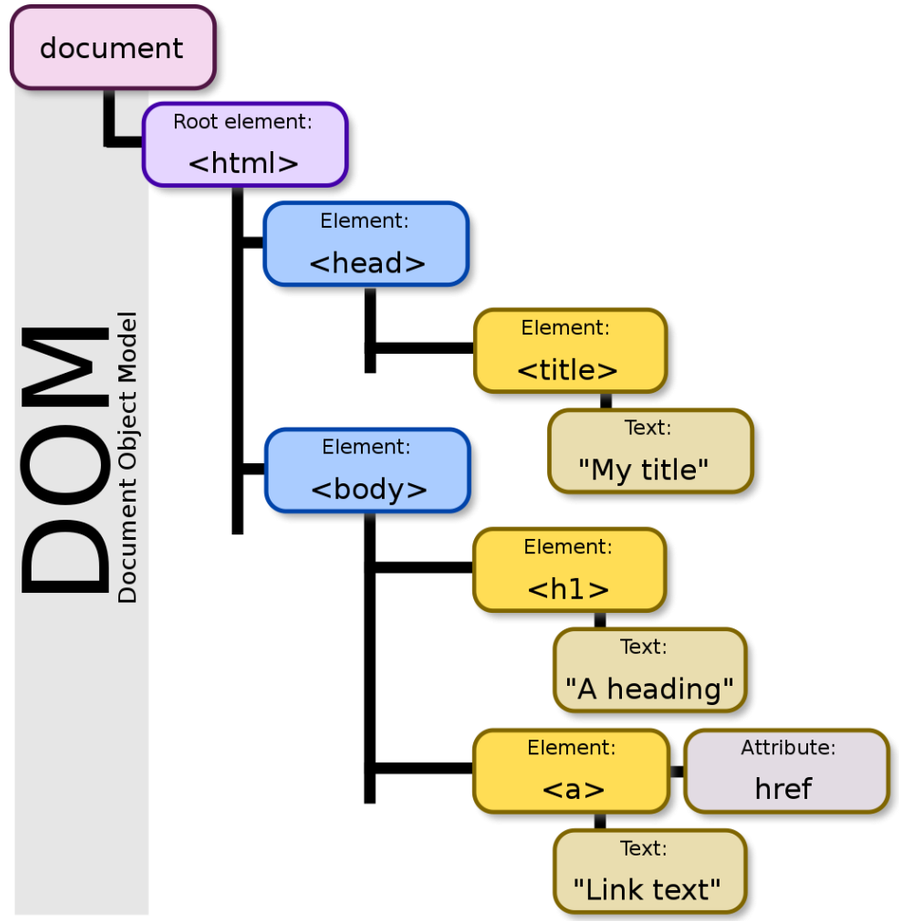


https://en.wikipedia.org/wiki/Document_Object_Mode

More on appendChild

Every element in the DOM tree has a "parent" element and might have one or more "children" elements.

`appendChild` is a method available in every HTML element to add a *new* element **at the bottom** of the list of its children.



Bonus question. What happens if you append an element that is already appended somewhere in the DOM under a new parent?

https://en.wikipedia.org/wiki/Document_Object_Mode

Module 4: References

- https://www.stefanobalietti.com/teaching/programming-fundamentals/
- https://javascript.info/
- https://developer.mozilla.org/en-US/docs/Web
- https://css-tricks.com/
- https://www.w3schools.com/html/
- https://www.w3schools.com/css/

Module 4: jQuery

Image source

Image source

Module 4: jQuery

Learning Goals

- Learn how to import and use jQuery
- Select elements
- Perform simple animations

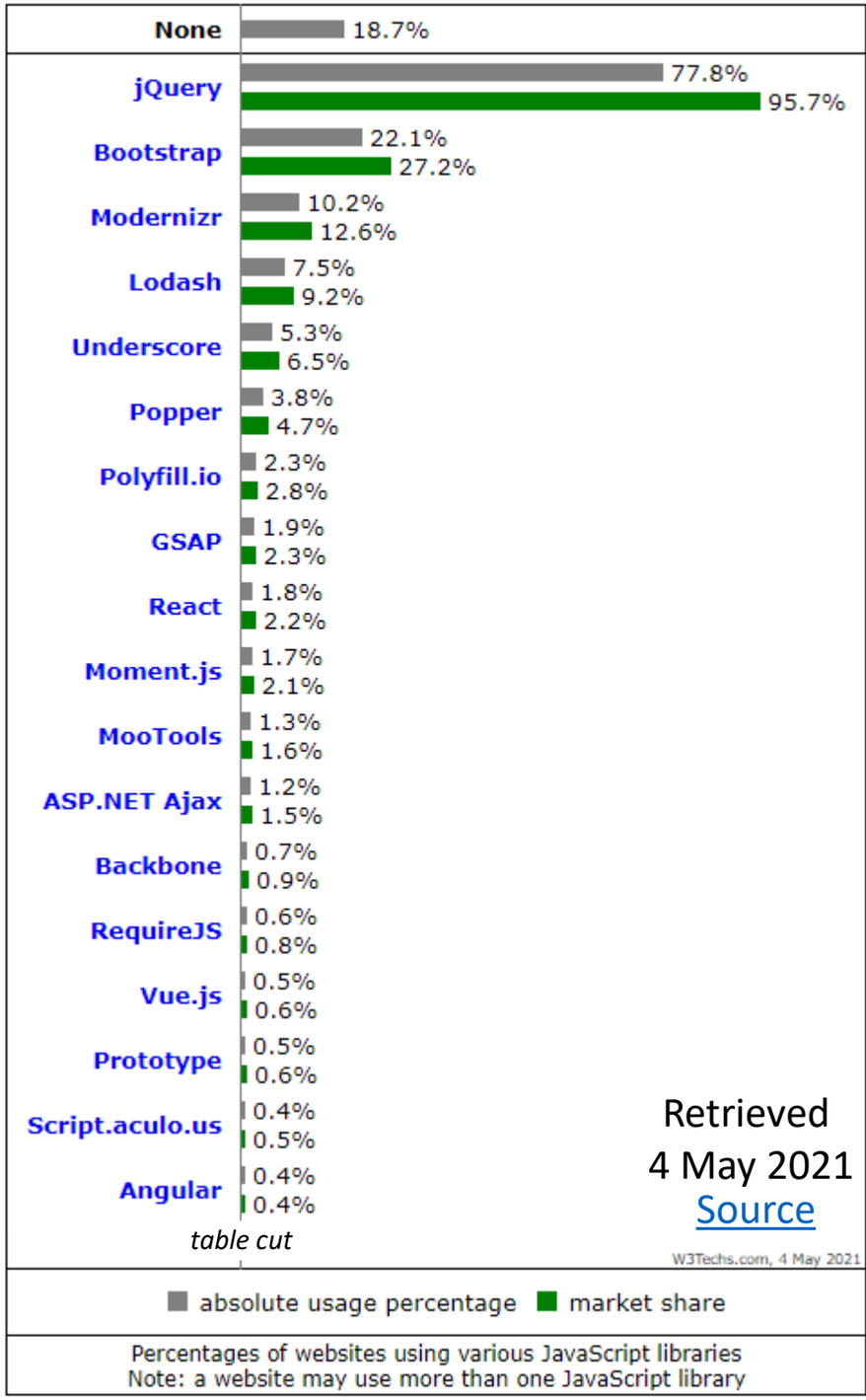
Module 4: jQuery

- **Free** and **open source** JS library to simplify:
  - DOM traversal and manipulation,
  - event handling,
  - CSS animation,
  - Ajax requests.

Module 4: jQuery

- **Free** and **open source** JS library to simplify:
  - DOM traversal and manipulation,
  - event handling,
  - CSS animation,
  - Ajax requests.

- Most widely deployed JS library, 3 to 4 times more usage than any other JS library



Module 4: jQuery

- Easy to use
- Easy to embed

Module 4: jQuery

- Easy to use
- Easy to embed

But,

- relatively slow
- Not a framework to build large, complex apps, such as Vue, Angular, or React

Module 4: References

- https://jquery.com/
- https://github.com/jquery/jquery
- https://www.w3schools.com/jquery/
- https://polyfill.io/

Module 5: Bootstrap

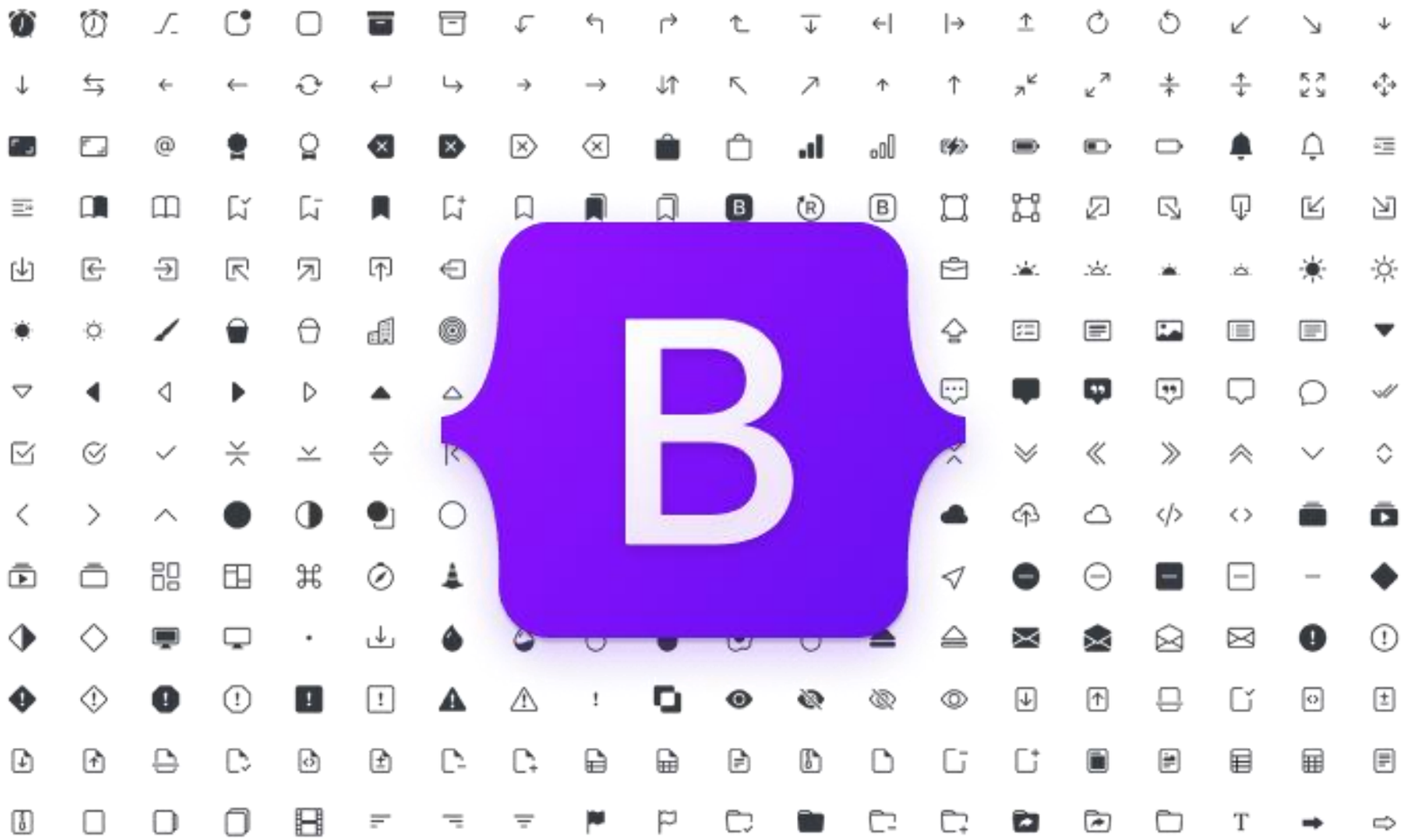


Image source

Module 5: Bootstrap

Learning Goals

- Improve the **old school** web app to cheer us up in difficult times.
- Include bootstrap CSS and JS in your app
- Understand grid layouts
- Understand and use Bootstrap components

Module 5: Bootstrap

- Free and open-source mobile-first front-end web development framework.

- It contains CSS- and (optionally) JavaScript-based design templates for:
  - typography,
  - forms,
  - buttons,
  - navigation,
  - …

- With over 150k stars is in the top-ten of most starred GitHub projects

Module 5: Bootstrap

- Current stable version is **v4**: uses jQuery for JS animations
- Soon-to-be-released **v5**: replaces jQuery with vanilla JS
- Great documentation

A responsive site should adapt to the container

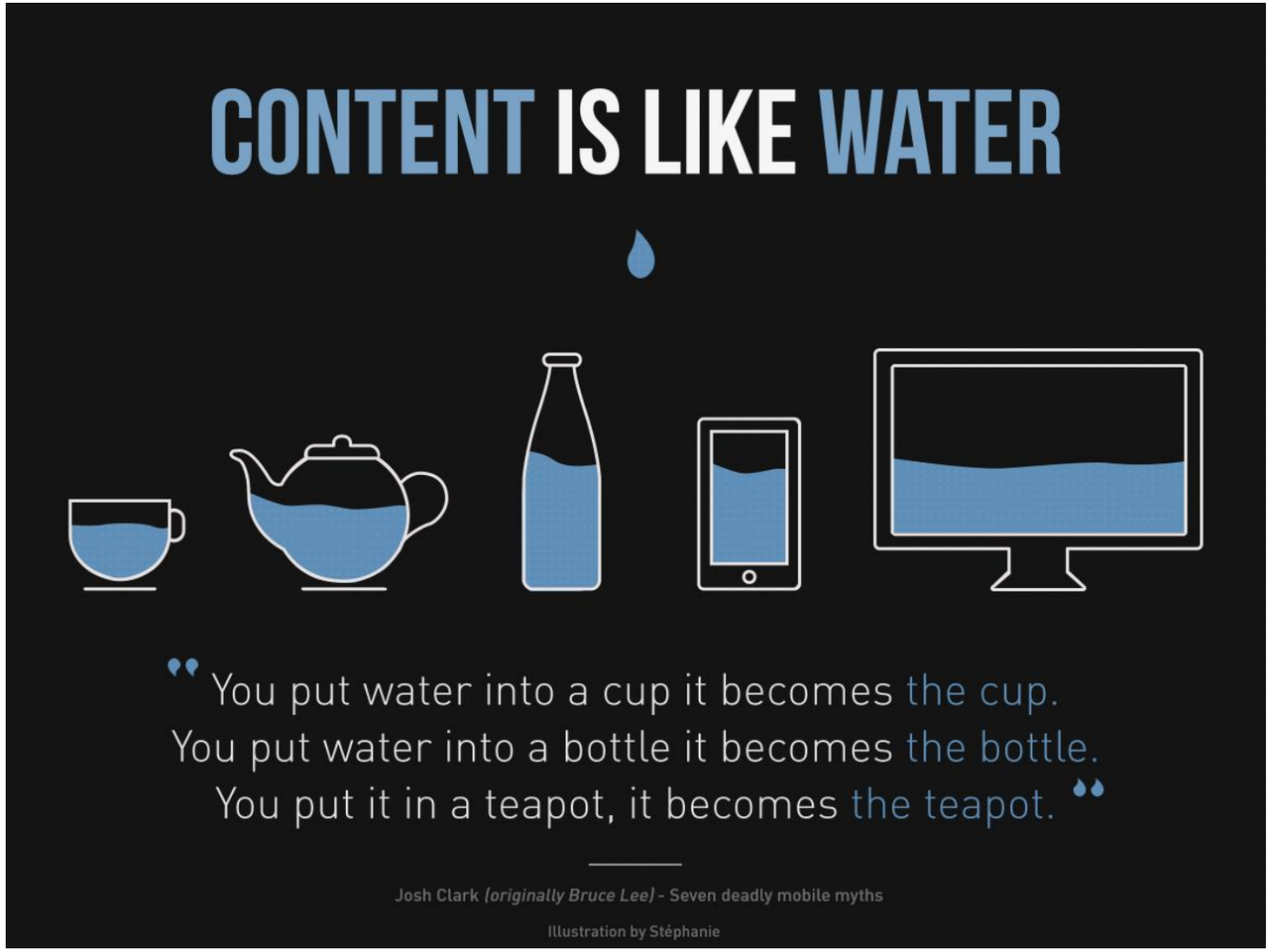
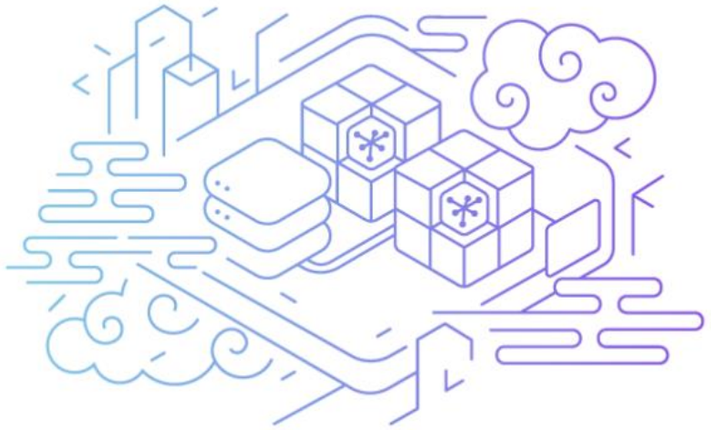


Image source

Module 5: Resources

- https://getbootstrap.com/
- https://getbootstrap.com/docs/4.6/getting-started/introduction/
- https://github.com/twbs/bootstrap
- https://www.ipraxa.com/blog/bootstrap-5-whats-new-whats-changed/

Module 6: Web Server



HEROKU

Express 4.17.1

Fast, unopinionated, minimalist web framework for Node.js

Module 6: Web Server

Learning Goals

- Setup the **ExpressJS** server in NodeJS
- Understanding middleware
- Creating custom routes and password protected routes
- Creating RESTful APIs
- Upload your server on the **Heroku** cloud

Module 6: Web Server

- **Express JS** provides a minimalistic and fast web server API to server static assets and templates
- Express JS is the most installed server in NodeJS ecosystem (can be used in production in tandem with the NginX server---not covered here)
- **Heroku** is a cloud platform *as a service* supporting several programming languages.
- Heroku focuses on **apps** and provides the backend infrastructure to run them.

Module 5: Resources

Learning Goals

- https://expressjs.com/
- https://expressjs.com/en/4x/api.html
- https://scotch.io/tutorials/whats-new-in-expressjs-5-0
- https://www.geeksforgeeks.org/different-servers-in-node-js/
- https://javascript.plainenglish.io/fastify-express-benchmark-4c4aebb726d6
- https://www.heroku.com