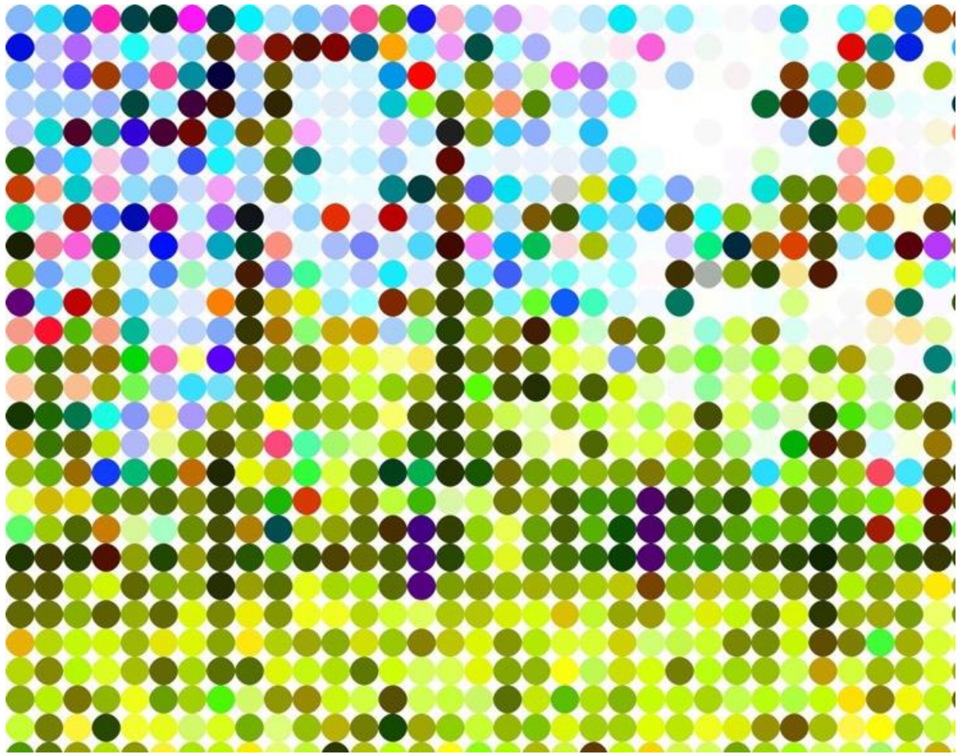
SHARE NETWORK



Inspiration Day 3

Share Academy

Dorien & Bilal

WIFI

SSID:

Lokaal Lokaal Gast

Password:

allpalacesaretemporarypalaces

Download slides:
https://code.share-network.org/inspiration-days

What’s up?

Chapter 1	Introduction to interactivity
Chapter 2	CSS interactivity & animation
Chapter 3	CSS exercises
	Lunch break
Chapter 4	Javascript introduction
Chapter 5	Javascript exercises
Chapter 6	Work on your website
	End of Inspiration Days :-(

Website progress

How are you doing with the website?

Show a litte preview?

Anyone stuck / need help?

Recap html

HTML = Hyper Text Markup Language

Used in a semantic way, elements have meaning

Used for structure, not for styling

Most element have a opening and a closing tag

Browsers already have some default styling for elements

Html files have the .html extension

Recap CSS

CSS = Cascading Style Sheet

A CSS file contains style rules

A CSS rule defines which element(s) to select and how to style them

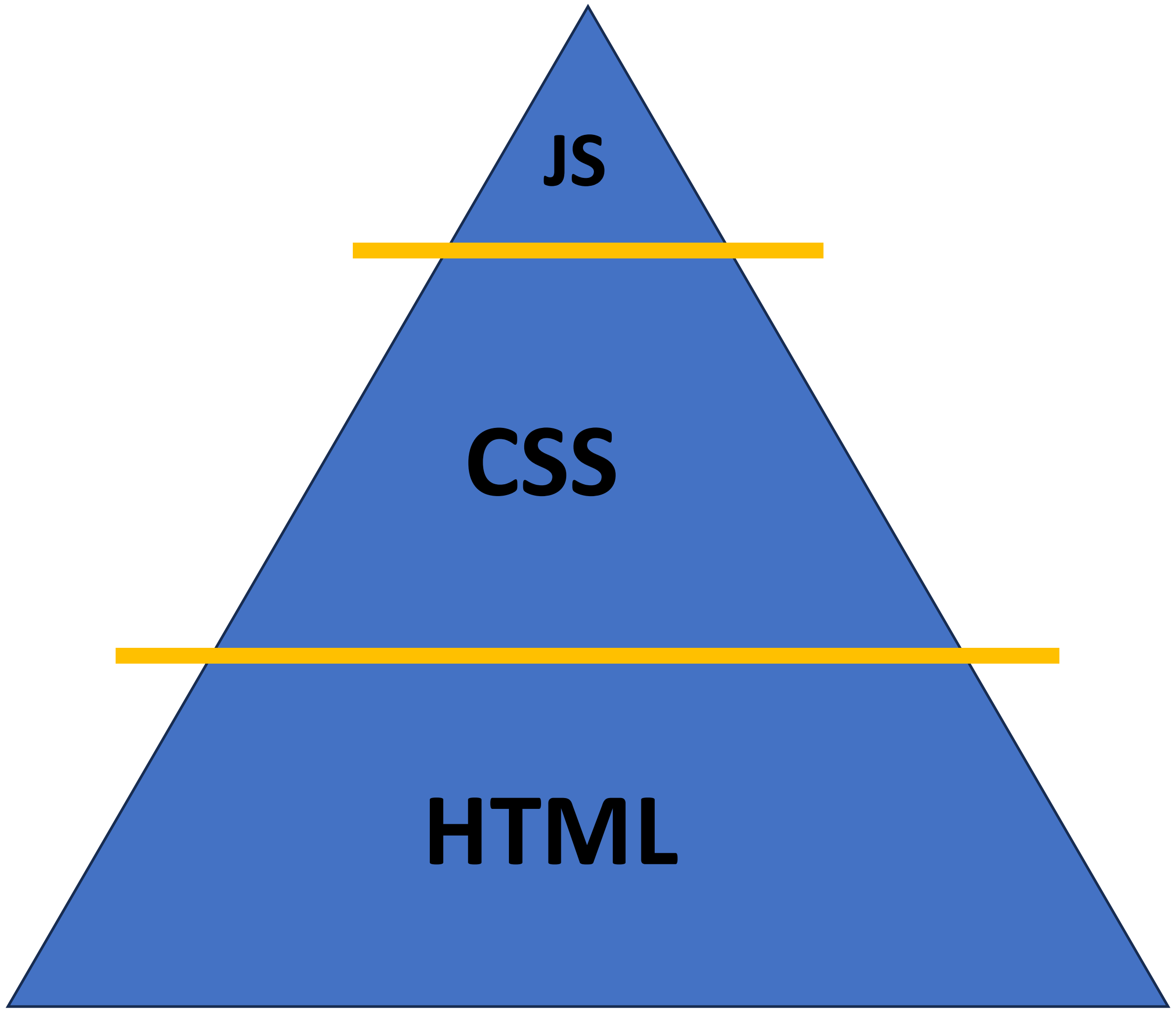
“Link” a CSS file to your HTML page with the <link> element

The webbrowser applies the CSS rules to the selected HTML elements

HTML = Structure and semantic content

HTML + CSS = structure + style = Beautiful webpages

CSS files have the .css extenstion


JS
CSS
HTML

Interactivity

We guide users through our website

We give them feedforward and feedback

It increases engagement

It’s fun!

CSS & Interactivity

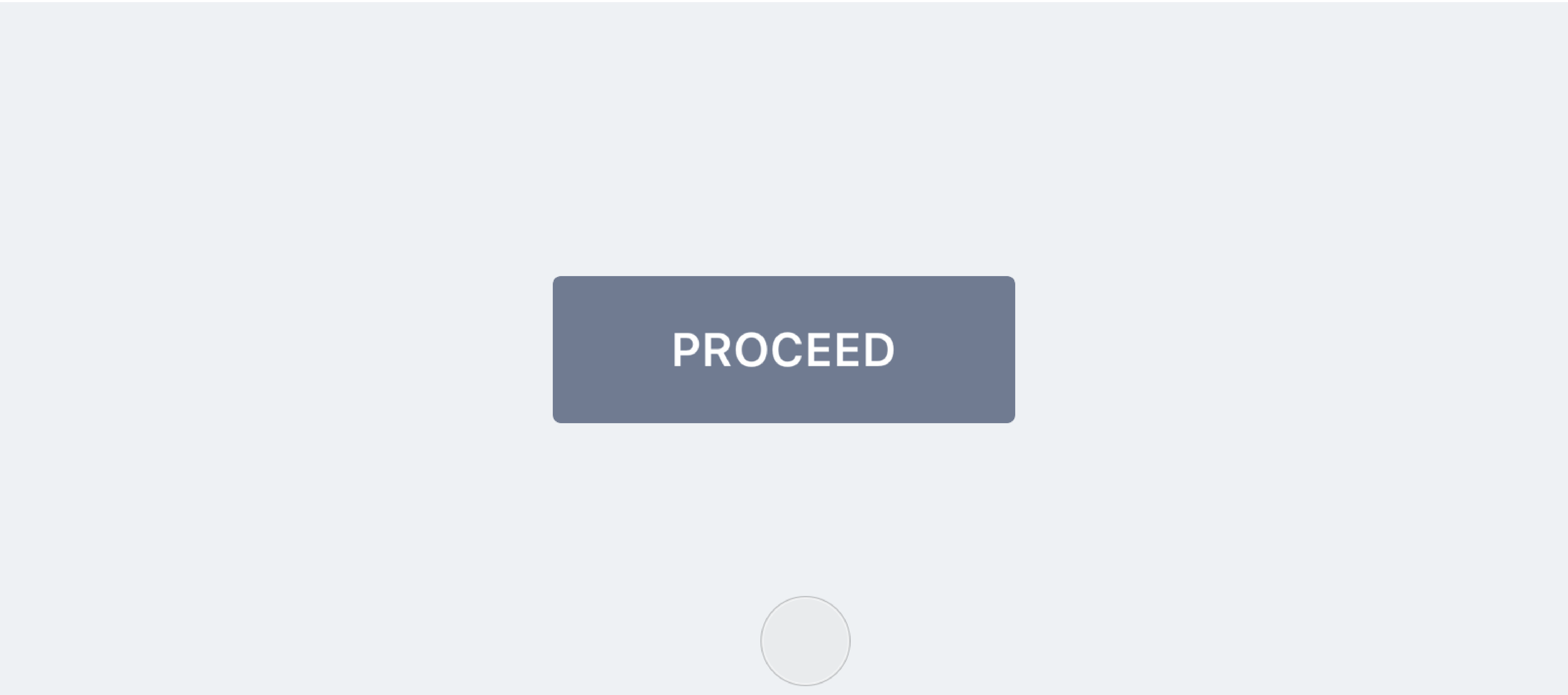
States of elements (pseudo elements)

Transitions

Animation

Button animation

with feedforward & feedback



Pseudo classes

Define a state of an element

:valid

:focus

:hover

:invalid

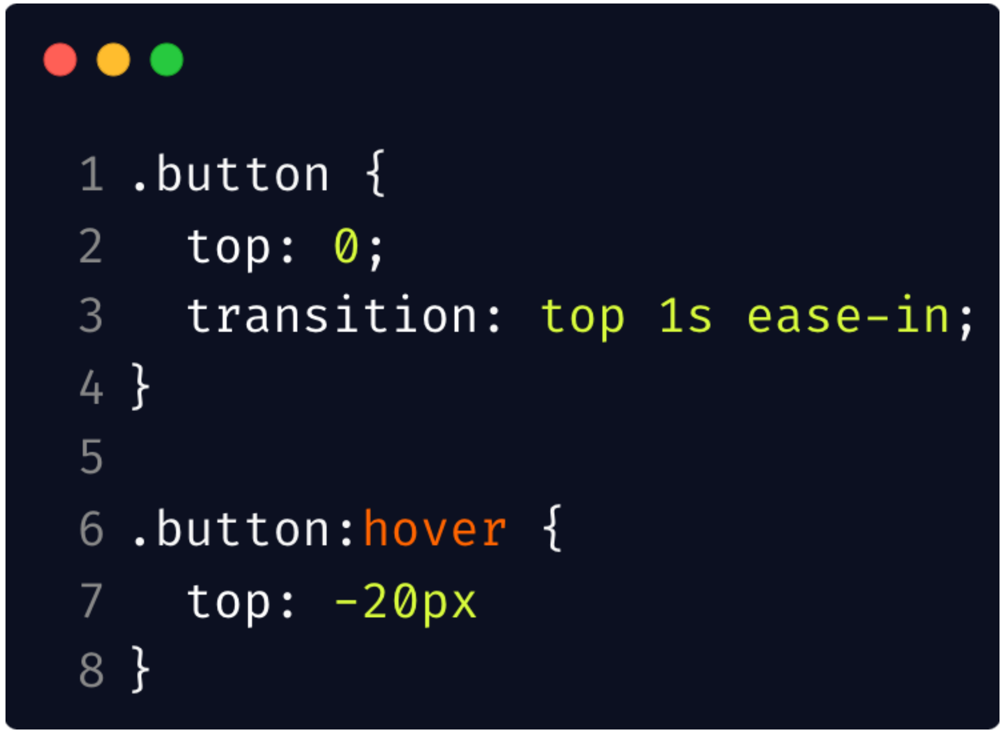
:active

:focus-within

:checked

Transitions

Define a beginning and an end



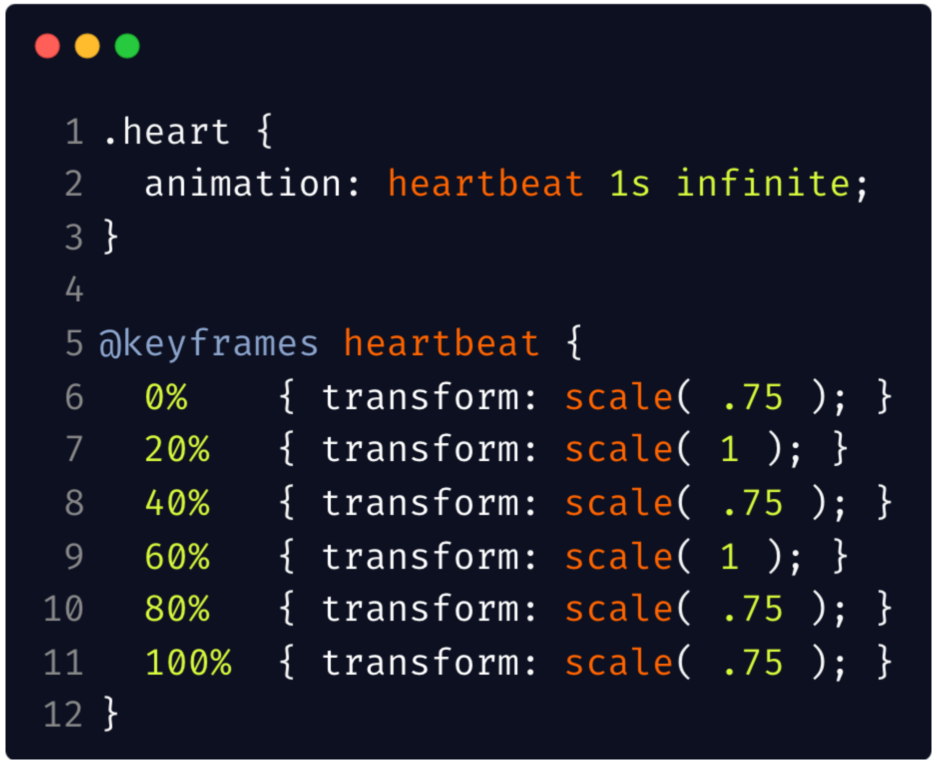
```css
.button {
  top: 0;
  transition: top 1s ease-in;
}

.button:hover {
  top: -20px
}
```

Tutorial on CSS transitions

Animations

Beginning, an end and steps in between



```css
.heart {
  animation: heartbeat 1s infinite;
}

@keyframes heartbeat {
  0%    { transform: scale( .75 ); }
  20%   { transform: scale( 1 ); }
  40%   { transform: scale( .75 ); }
  60%   { transform: scale( 1 ); }
  80%   { transform: scale( .75 ); }
  100%  { transform: scale( .75 ); }
}
```

[Tutorial on keyframe animations]

Exercise time

Let’s play

Go to https://code.share-network.org/inspiration-days/

1. Download the css-interactivity.html
2. Put in in a folder on your computer _*like inspiration-day-3*_
3. Open the folder in VS Code
4. Create a style.css and link it to the html file _*remember?*_

Lunch break

Javascript

Overview

- History of Javascript
- Applications and usage
- Client & server side
- Libraries & frameworks
- Documentation & reference

Coding exercises

- Step by step exercises

Building a simple Hamburger menu

History of Javascript

- Javascript was “born” in 1997
- Brandon Eich / Netscape Navigator (browser)
- Used for simple in-browser scripts
  - Form validation / HTML element manipulation
- Google’s V8 engine (2008) – built into Google Chrome
- ECMAScript
- Node.js - Javascript on the server and standalone
- Remember: **Javascript is NOT Java**!

version 0.9 beta
File Edit View Go Bookmarks Options Directory Help
Back Forward Home Reload Images Open Find Stop
Location: about:
Guided Tour
What's New
Questions
Net Search
Net Directory
Newsgroups

Mosaic Netscape version 0.9 beta
Copyright © 1994 Mosaic Communications Corporation,
All rights reserved.

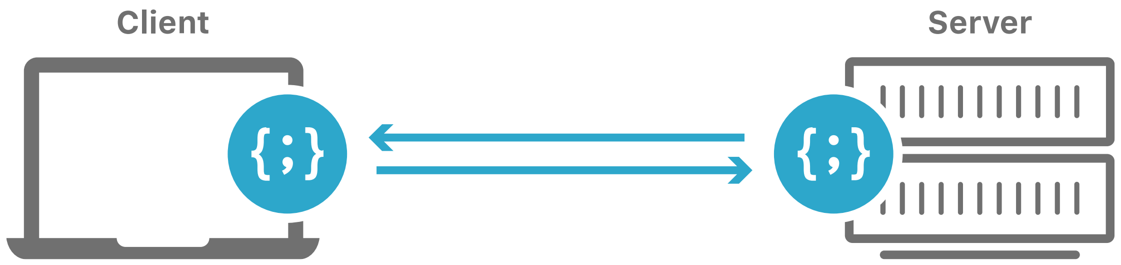
This is *BETA* software subject to the license agreement set forth in the README file.
Please read and agree to all terms before using this software.

Report any problems to win_cbug@mcom.com

Mosaic Communications, Mosaic Netscape, and the Mosaic Communications logo are trademarks of Mosaic Communications Corporation.

Any provision of Mosaic Software to the U.S.Government is with "Restricted rights" as follows: Use, duplication or disclosure by the Government is subject to restrictions set forth in subparagraphs (a) through (d) of the Commercial Computer Restricted Rights clause at FAR 52.227-19 when applicable, or in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, and in similar clauses in the NASA FAR Supplement. Contractor/manufacturer is Mosaic Communications Corporation, 650 Castro Street, Suite 500, Mountain View, California, 94041.

Client vs Server-side



Client side

- Executed on computer of website visitor (browser = client)
- Script attached to a HTML page
- Strict security. No access to computer files, system, camera etc. unless explicitly allowed to by the user

Server-side

- Executed on the server (for example via Node.js)
- Full access to (server) files

Application & usage

On the client (browser) side examples

- Add interactivity to webpages
- Manipulate the DOM / page elements
- Animation
- Form / input validation
- Image sliders
- Simple games
- Charts
- Fetch data from other websites or API’s
- Control interactive maps
- Mobile app development
- …

Server-side examples:

- Web server
- Data retrieval from a database
- Sending emails
- Image processing
- Processing user login (authentication)
- Handling file uploads
- Writing files to disk

Javascript libraries

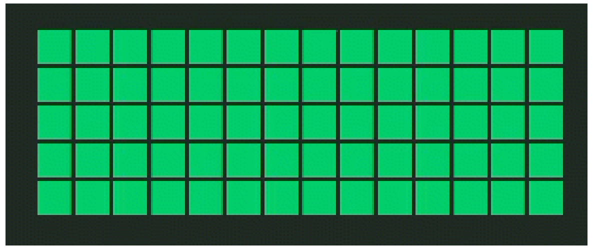
Libraries

Handy (mostly free) Javascript files you can download and include in your webpage and that you can use for a specific purposes like: *animation, form validation, data visualization*

For example:

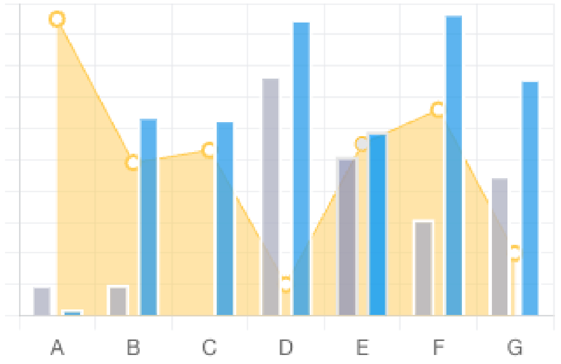
https://gsap.com/

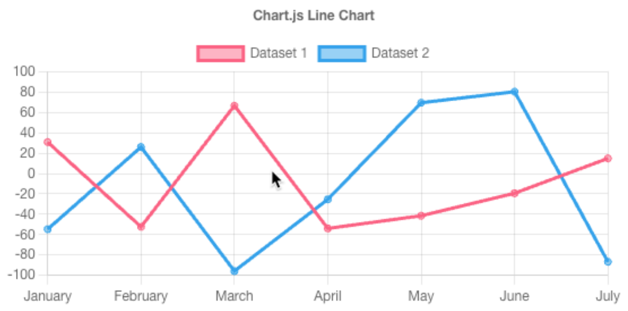
For complex animations on a webpage



https://chartjs.org

For creating charts from data





Javascript frameworks

Frameworks provide the building blocks, structure and methodology to build bigger systems.

- Develop faster
- Prebuilt components and utilities
- Best practices
- Community & support

Popular Javascript frameworks:

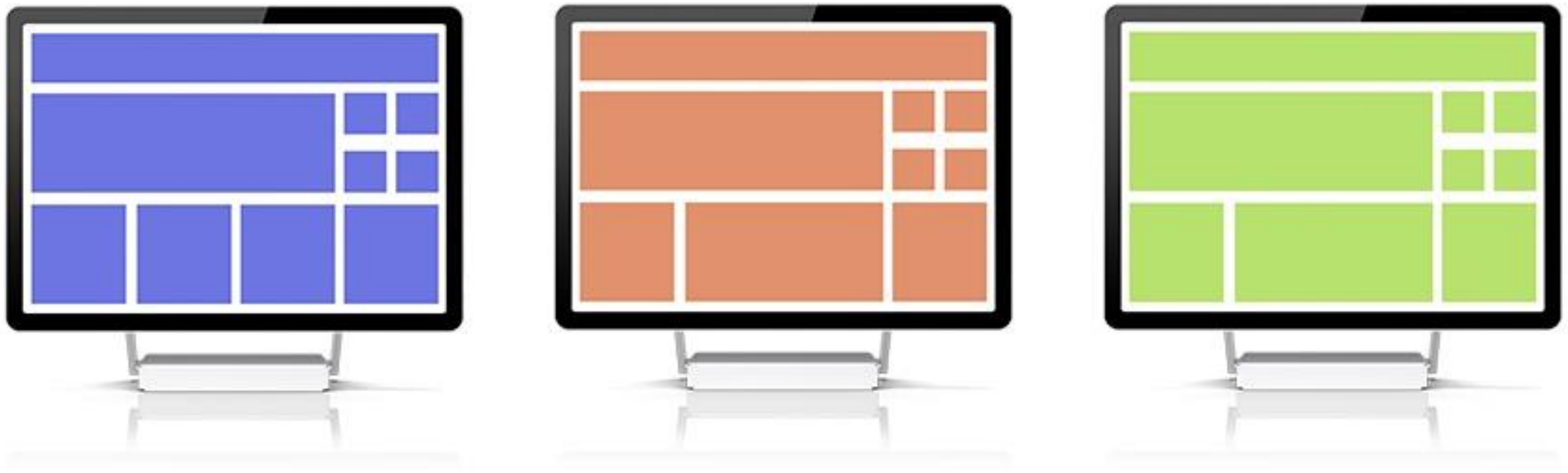
- React – https://react.dev/
- Vue.JS - https://vuejs.org/
- Svelte - https://svelte.dev/

Mostly used for building websites that feel and behave like applications/

Single page applications or SPA’s

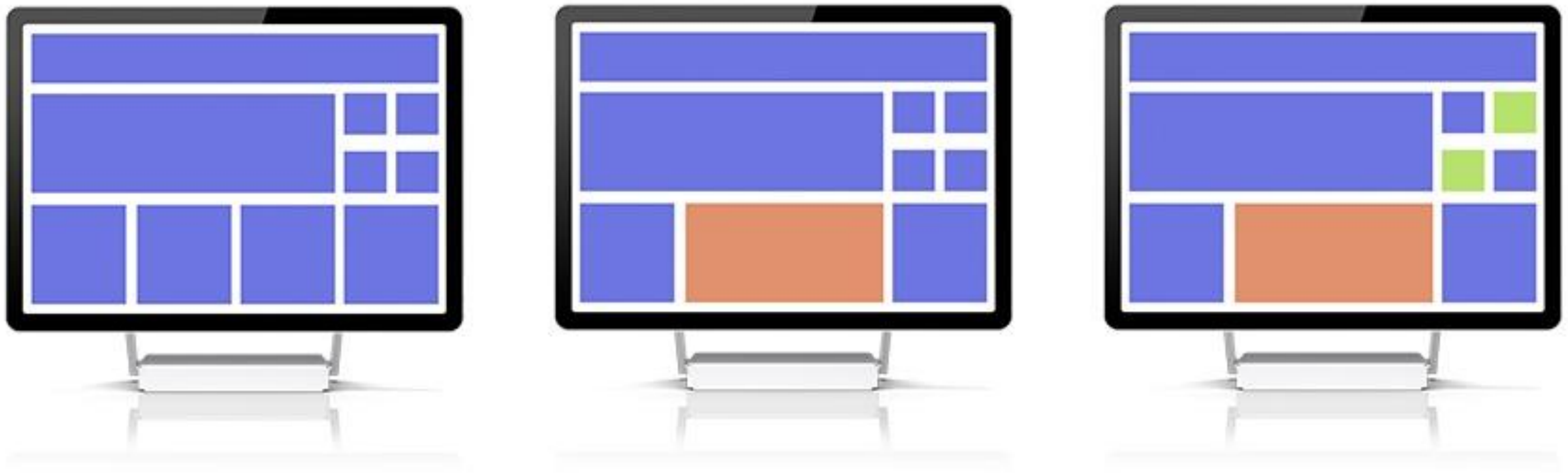
Traditional

Every request for new information gives you a new version of the whole page.



Single Page Application

You request just the pieces you need.

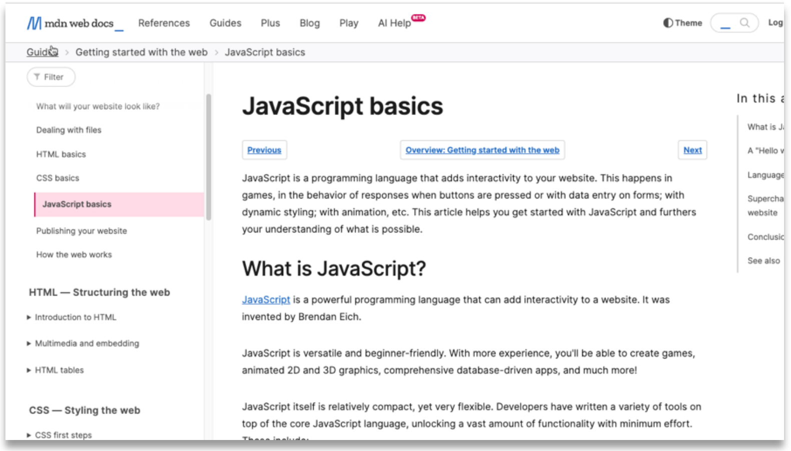


Documentation & reference

When learning a new programming language….

Read the documentation!

- Read the available language guides
- Learn about syntax: how to write and read this language
- Learn the usage (why/when to use this language, when not)
- Learn about the data types: strings, numbers, lists, objects….
- Follow beginner tutorials
- Try out code and create small projects
- Practice, practice practice!
- [Codecademy]()



JavaScript basics

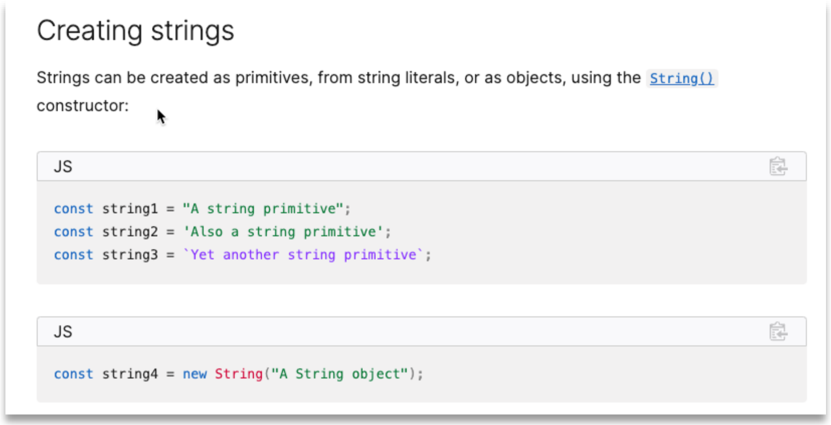
JavaScript is a programming language that adds interactivity to your website. This happens in games, in the behavior of responses when buttons are pressed or with data entry on forms; with dynamic styling; with animation, etc. This article helps you get started with JavaScript and furthers your understanding of what is possible.

What is JavaScript?

JavaScript is a powerful programming language that can add interactivity to a website. It was invented by Brendan Eich.

JavaScript is versatile and beginner-friendly. With more experience, you'll be able to create games, animated 2D and 3D graphics, comprehensive database-driven apps, and much more!

JavaScript itself is relatively compact, yet very flexible. Developers have written a variety of tools on top of the core JavaScript language, unlocking a vast amount of functionality with minimum effort.



Creating strings

Strings can be created as primitives, from string literals, or as objects, using the `String()` constructor:

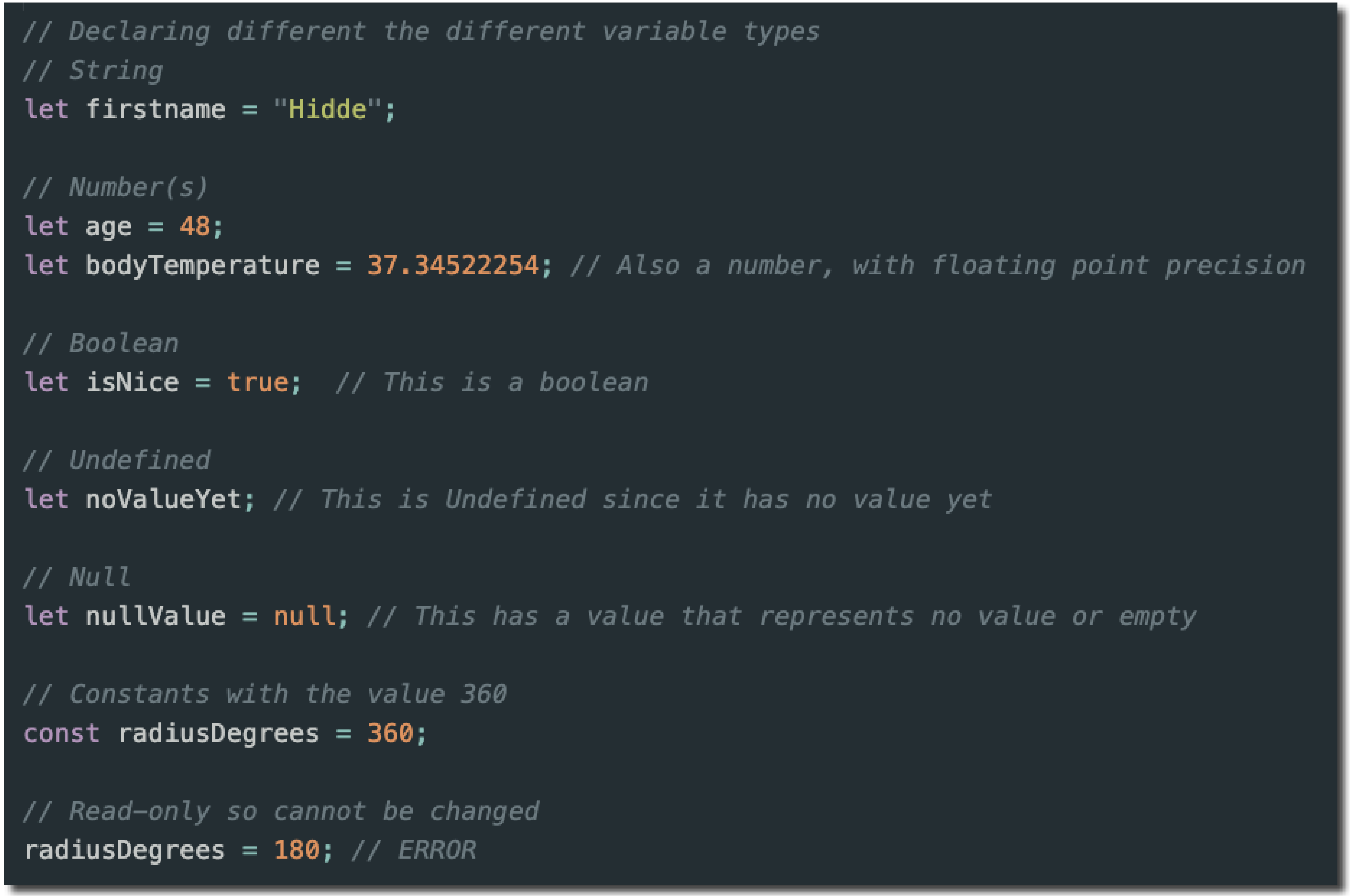
JS

```
const string1 = "A string primitive";
const string2 = 'Also a string primitive';
const string3 = `Yet another string primitive`;
```

JS

```
const string4 = new String("A String object");
```

https://developer.mozilla.org/en-US/docs/Web/JavaScript
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction



```
// Declaring different the different variable types
// String
let firstname = "Hidde";

// Number(s)
let age = 48;
let bodyTemperature = 37.34522254; // Also a number, with floating point precision

// Boolean
let isNice = true;  // This is a boolean

// Undefined
let noValueYet; // This is Undefined since it has no value yet

// Null
let nullValue = null; // This has a value that represents no value or empty

// Constants with the value 360
const radiusDegrees = 360;

// Read-only so cannot be changed
radiusDegrees = 180; // ERROR
```

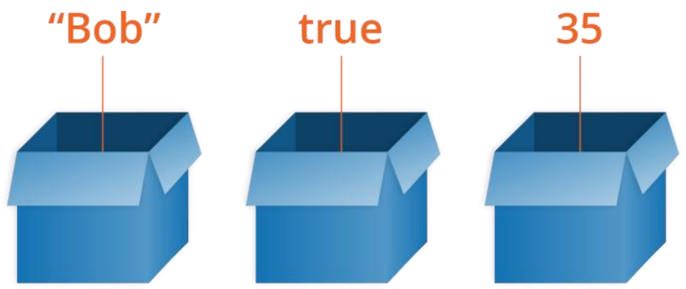
Javascript interactivity

Variables

Dom manipulation

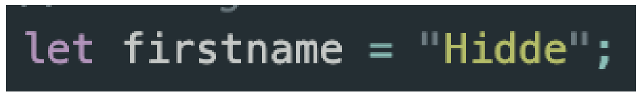
Listening to events

Variables

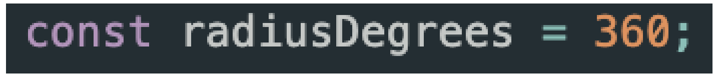


- Variables are the basic building blocks in programming
- Containers with a name, that store your data
- Variables can store different **types** of data
- A variable always has a clear name, so you can refer to it in your code

You define a variable in your Javascript code with **let** or **const**



```
let firstname = "Hidde";
```



```
const radiusDegrees = 360;
```

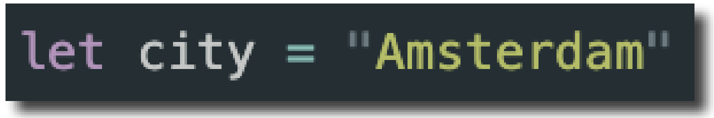
A variable declared with **let** can be changed after initialization

A variable declared with **const** can not be changed **AFTER** initialization, it is **read only**

Data types

String

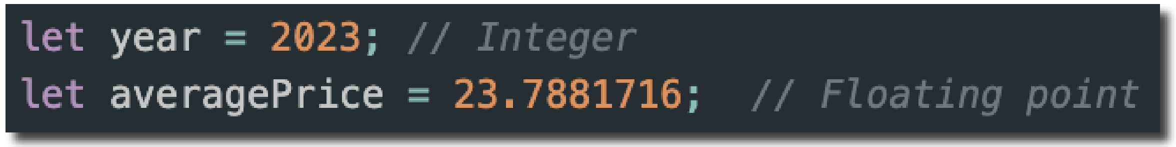
For sequences of characters (text)



```
let city = "Amsterdam"
```

Number

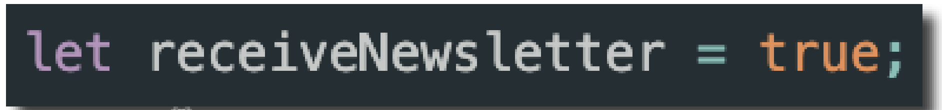
All kinds of numbers, integers, floating point numbers



```
let year = 2023; // Integer
let averagePrice = 23.7881716;  // Floating point
```

Boolean

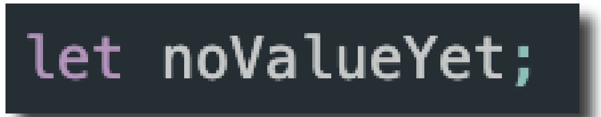
Can only be **true** or **false**



```
let receiveNewsletter = true;
```

Undefined

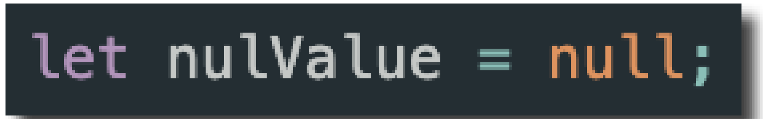
No value **yet** (special type of variable)



```
let noValueYet;
```

Null

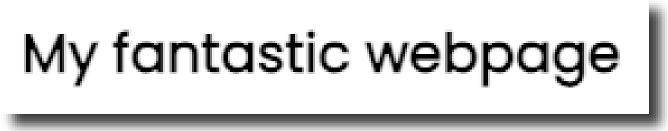
Value that represents **no value** or **empty**

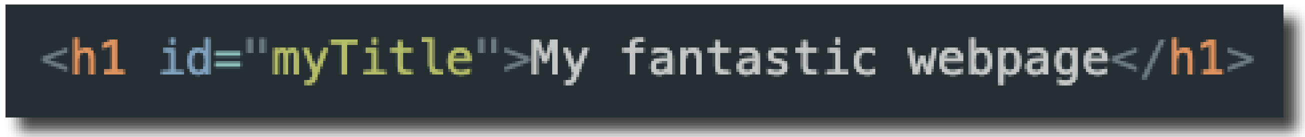


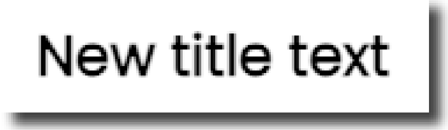
```
let nulValue = null;
```

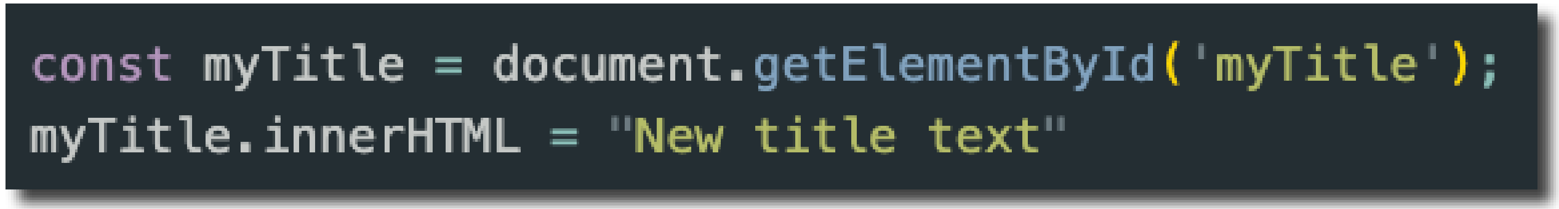
Dom manipulation

- You can select html elements
- And then change their contents, behaviour, style etc.


My fantastic webpage


```html
<h1 id="myTitle">My fantastic webpage</h1>
```


New title text


```javascript
const myTitle = document.getElementById('myTitle');
myTitle.innerHTML = "New title text"
```

Listening to events

click

mousemove

hover

scroll

submit

load

resize

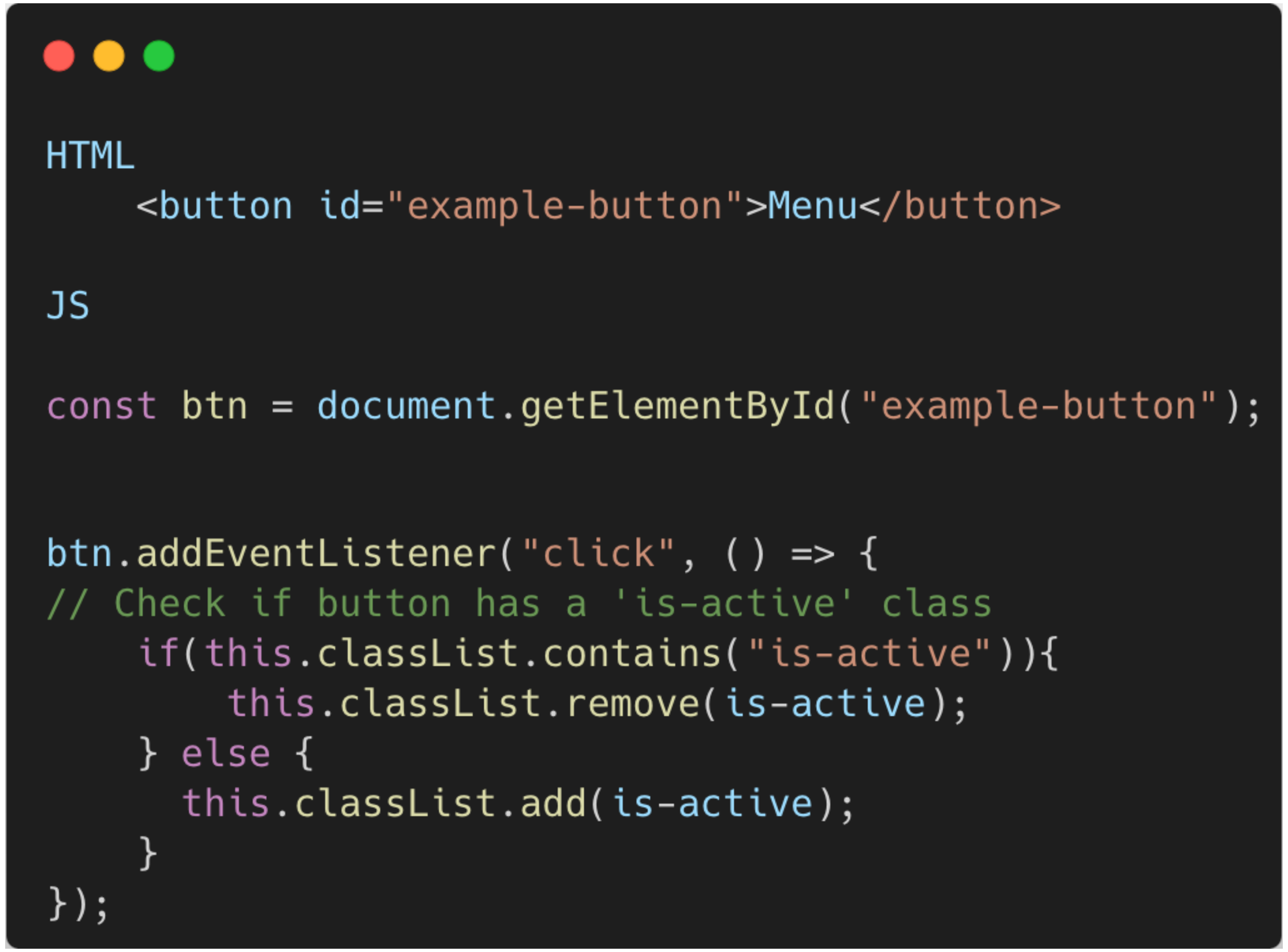
keyup

change

Listening to events

Do stuff in between the execution of an event.

This example checks if there is a class on the button that is clicked and then removes it or adds it.



```
HTML
    <button id="example-button">Menu</button>

JS

const btn = document.getElementById("example-button");


btn.addEventListener("click", () => {
// Check if button has a 'is-active' class
    if(this.classList.contains("is-active")){
        this.classList.remove(is-active);
    } else {
      this.classList.add(is-active);
    }
});
```

Exercise time

Let’s play

Go to https://code.share-network.org/inspiration-days/

1. Download the javascript.html
2. Put in in the same folder as css.html
3. Create a app.js file

Graduation ceremony

December 12th, 15:00 – 17:00

Short presentation (max. 5 minutes)

- Your website / initial ideas / structure
- What you have learned
- What your next steps are

Try and "finish" your website as far possible.

Questions/ Feedback

Dorien/Bilal - teacher@share-network.org

Where to go from here

Javascript courses, exercises, challenges & documentation

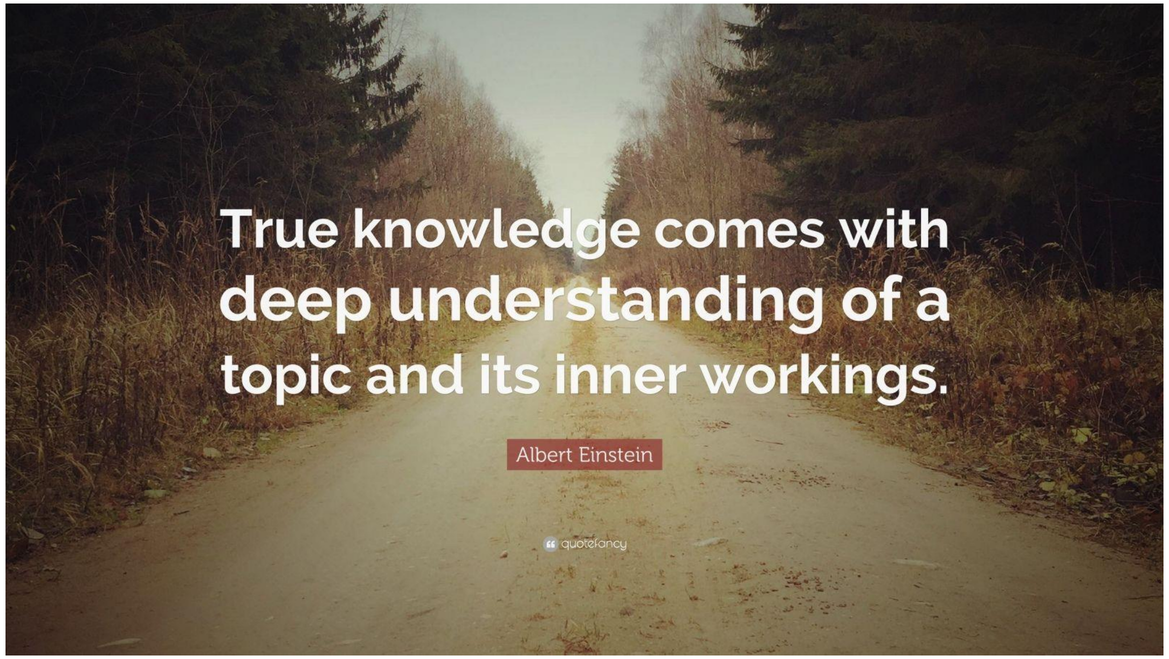
https://developer.mozilla.org/en-US/docs/Web/JavaScript

https://javascript.info/

https://edabit.com/tutorial/javascript

https://edabit.com/challenges

https://phuoc.ng/collection/html-dom


True knowledge comes with deep understanding of a topic and its inner workings.
Albert Einstein
quotefancy