The Maersk Mc-Kinney Moller Institute

Human Computer Interaction

Lecture 8: CSS

by Henrik Lange

sdu.dk

#sdudk

September 2020

Plan

- → Week 1: Setting The Stage
- → Week 2: Mapping, Personas and Journeys
- → Week 3: Sketching
- → Week 4: Decisions and Storyboards
- → Week 5: Prototyping
- → Week 6: Prototype Testing
- → Week 7: Graphic Design
- → **Week 8: Building Modern UI (CSS)**
- → Week 9: Building Modern UI (Bootstrap, jQuery & Leaflet)
- → Week 10: Accessibility and
- → Week 11: Usability Te
- → Week 12: Building Modern UI

Detailed Plan for rest of semester

Tuesday Nov. 16th	Tuesday Nov. 23th	Tuesday Nov. 29th	Tuesday Dec. 6th	Tuesday Dec. 14th
Week 8 Lecture 8+9	Week 9 Lecture 10+11	Week 10 Instructor Class	Week 11 Instructor Class	Week 11 Recorded Lecture 12
Lecture: Live in classroom *8-12*	Lecture: Live in classroom *8-12*	Lecture: No Lecture	Lecture: No Lecture	Lecture: *Recorded* *Available Tuesday Morning*
Exercises: No exercise class	Exercises: No exercise class	Exercises: Instructor class *8-12*	Exercises: Instructor class *8-12*	Exercises: None

Portfolio Hand-in
Dec. 13th

Agenda

- → Last Week – Catch Up
- → HTML
  - → Quick Recap
  - → Exercise
- → CSS
  - → Import
  - → Selectors
  - → Specificity
  - → The Box Model
  - → Responsiveness
  - → Display
- → Tomorrow – Instructor Session
- → Next Week – Prepa

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Last Week

SDU

September 2020

The Maersk Mc-Kinney Moller Institute

This Week: HTML & CSS

SDU

sdu.dk

#sdudk

September 2020

HTML

September 2020

HTML CSS & JavaScript

- → HyperText Markup Language

- → A browser **only** understands HTML, CSS & JavaScript
  - → All variations are translated to these language
  - → Frameworks are pre-written code you can call
  - → Most media types are supported by browsers
  - → There are minor differences in browser interpretation

- → HTML Defines the structure and ties the 3 together
- → CSS is only concerned with how things look
- → Once HTML and CSS is loaded, the browser creates the DOM
- → JavaScript can manipulate the DOM after it is created
- → JavaScript is Turing complete; HTML and CSS is single-purpose
- → The languages are not limited to browsers

A General Template

```
<!DOCTYPE html>
<html>
    <head>
        <title>Title of the document</title>
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel="stylesheet" type="text/css" href="style.css">
    </head>
    <!--Comment-->
    <body>
        <h1>Header</h1>
        <p>General content</p>
        <a href="url">A link</a>
        <script src="myscripts.js"></script>
    </body>
</html>
```

CAUTION:

PowerPoint automatically formats ASCII signs!
If you copy this to your code, you will most likely get a syntax error!
Especially the citationmarks tend to be problematic.

HTML CSS & JavaScript

Structure:

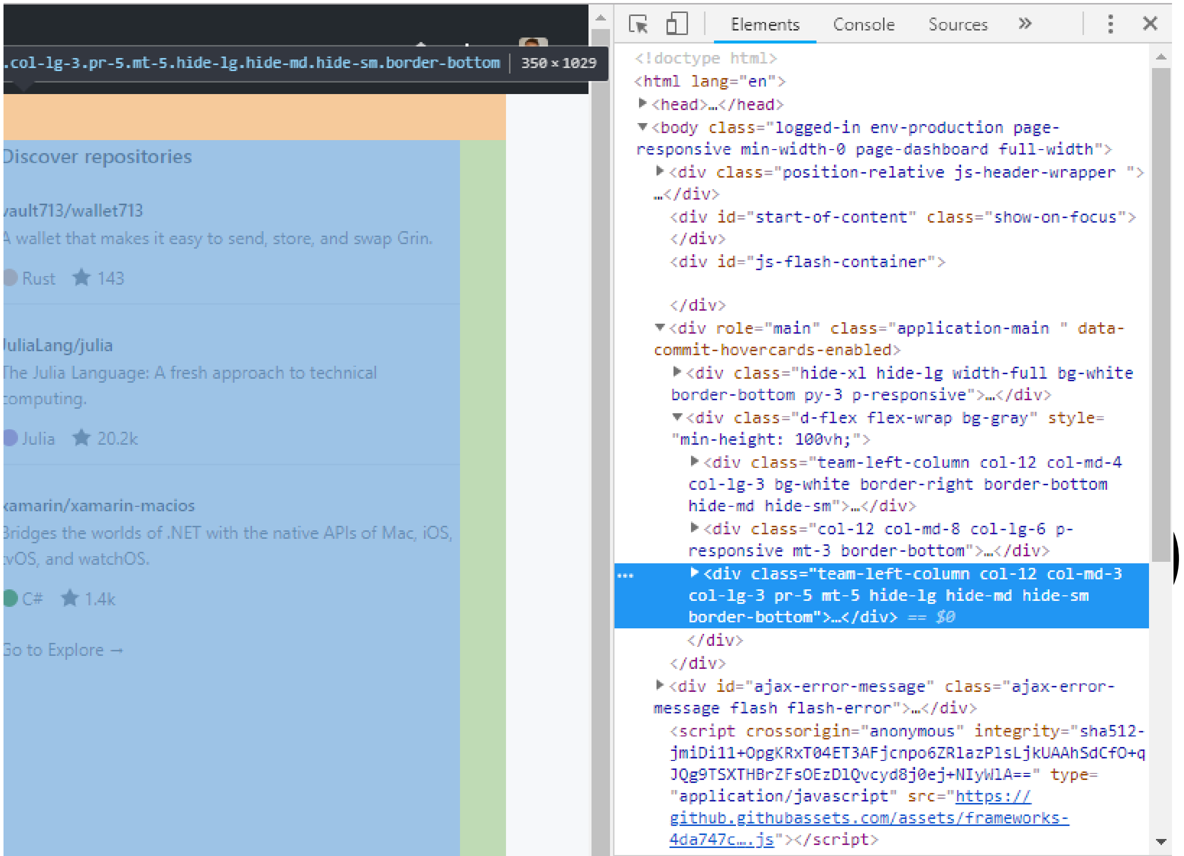
```
<element attribute="value">
   Element Content
</element>
```

→ <html> wraps the <head> and <body>
→ <head> contains functional, invisible content
→ <body> contains the actual document

→ Title, CSS-imports, metadata etc. goes in the head section
→ JavaScript imports and actual javascript goes in the bottom of the body
- → JavaScript Manipulates the DOM after it has loaded, but will not function correctly in all browsers if it is not the last part of the code.
- → The reason is that JS is sequential rather than part of the DOM so the scripts can not manipulate DOM elements before they are loaded

The Document Object Model - DOM

→ Right-click on any page in Chrome and select “inspect”
→ Hover over DOM element to highlight in document



.col-lg-3.pr-5.mt-5.hide-lg.hide-md.hide-sm.border-bottom | 350 × 1029

Discover repositories

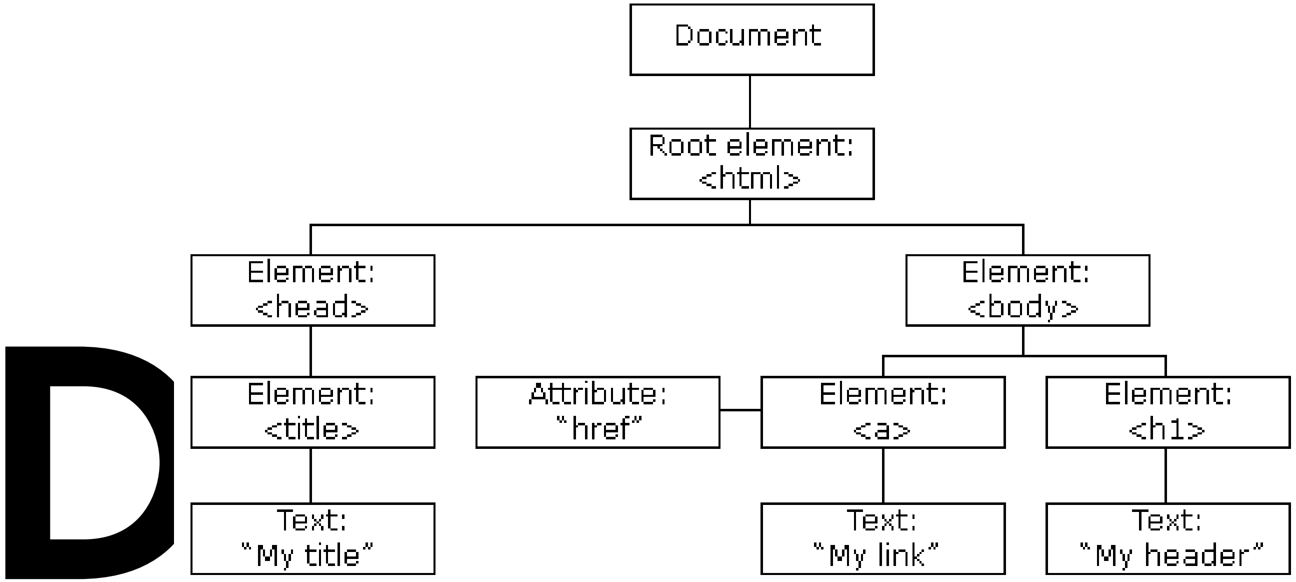
vault713/wallet713
A wallet that makes it easy to send, store, and swap Grin.
Rust ★ 143

JuliaLang/julia
The Julia Language: A fresh approach to technical computing.
Julia ★ 20.2k

xamarin/xamarin-macios
Bridges the worlds of .NET with the native APIs of Mac, iOS, tvOS, and watchOS.
C# ★ 1.4k

Go to Explore →

```
<!doctype html>
<html lang="en">
▶<head>…</head>
▼<body class="logged-in env-production page-
responsive min-width-0 page-dashboard full-width">
  ▶<div class="position-relative js-header-wrapper ">
  …</div>
   <div id="start-of-content" class="show-on-focus">
   </div>
   <div id="js-flash-container">

   </div>
  ▼<div role="main" class="application-main " data-
  commit-hovercards-enabled>
    ▶<div class="hide-xl hide-lg width-full bg-white
    border-bottom py-3 p-responsive">…</div>
    ▼<div class="d-flex flex-wrap bg-gray" style=
    "min-height: 100vh;">
      ▶<div class="team-left-column col-12 col-md-4
      col-lg-3 bg-white border-right border-bottom
      hide-md hide-sm">…</div>
      ▶<div class="col-12 col-md-8 col-lg-6 p-
      responsive mt-3 border-bottom">…</div>
…     ▶<div class="team-left-column col-12 col-md-3
      col-lg-3 pr-5 mt-5 hide-lg hide-md hide-sm
      border-bottom">…</div> == $0
     </div>
   </div>
  ▶<div id="ajax-error-message" class="ajax-error-
  message flash flash-error">…</div>
   <script crossorigin="anonymous" integrity="sha512-
   jmiDi11+OpgKRxT04ET3AFjcnpo6ZRlazPlsLjkUAAhSdCfO+q
   JQg9TSXTHBrZFsOEzD1Qvcyd8j0ej+NIyWlA==" type=
   "application/javascript" src="https://
   github.githubassets.com/assets/frameworks-
   4da747c….js"></script>
```


Document

Root element: <html>

Element: <head>

Element: <title>

Text: “My title”

Element: <body>

Attribute: “href”

Element: <a>

Text: “My link”

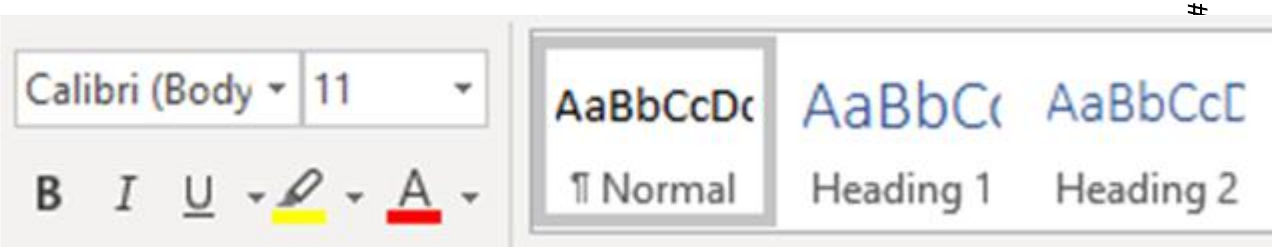
Element: <h1>

Text: “My header”

September 2020

Headers and Paragraphs

→ Document elements are comparable to those of MS Word
→ The **style** attribute contains CSS – we will get back to that



Header1

Header2

Some text with **bold**, *italic* and underlined text
Even a link

```
<h1>Title</h1>
<h2>Header</h2>
<p>
    Some text with <b>bold</b>, <i>italic</i>
and <u>underlined</u> text
    <br/>
    <a href="URL">Even a link</a>
</p>
```

Links

<a href="URI" title=“hover description">LINK TEXT</a>

→ **title**: is text the browser can display when the mouse hovers over the image

→ Can contain any elements that should be part of the clickable link.

→ CSS note:

- → Style: “decoration: none;”

Images

<img src="URI" alt="description" title="hover description"/>

→ <img> element has no end tag and is terminated with /

→ **alt**: text to display if image is not loaded

→ **title**: is text the browser can display when the mouse hovers over the image

→ width and height can be set with HTML attributes

- → width="300px" height="400px"
- → You should use CSS instead

Video

Local

```
<video width="320" height="240" poster="image.gif" autoplay controls>
  <source src="video.mp4" type="video/mp4">
  <source src="video.ogg" type="video/ogg">
  Your browser does not support the video tag.
</video>
```

YouTube

```
<iframe width="420" height="315"
src="https://www.youtube.com/embed/videoId?autoplay=1">
</iframe>
```

Lists

- Bullet
- Point
- List

```
<ul>
   <li>Bullet</li>
   <li>Point</li>
   <li>List</li>
</ul>
```

1. Numbered
2. List
3. With
4. Items

```
<ol>
   <li>Numbered</li>
   <li>List</li>
   <li>With</li>
   <li>Items</li>
</ol>
```

Tables

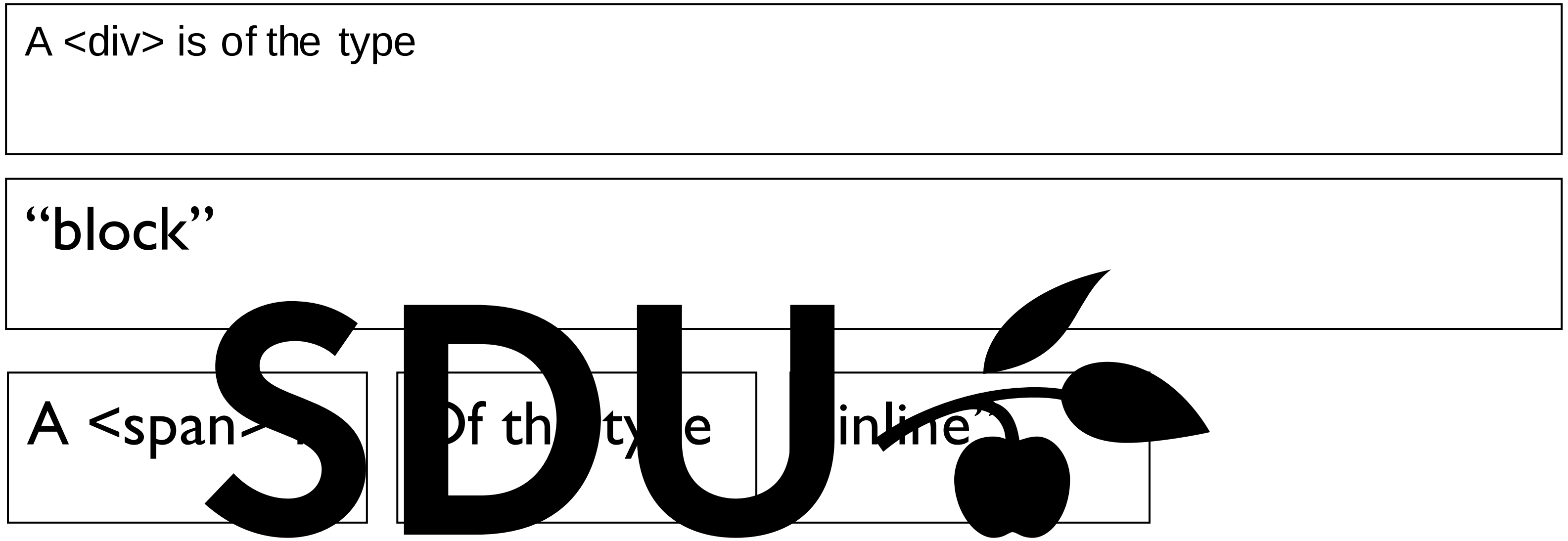
- → No column definition
- → Tables contain rows
- → Rows contain cells
- → Just like DB content!
- → Attributes to merge cells
    - → Rowspan
    - → Colspan

Name	Gender	Age
Mads	Male	21
Annie	Female	24

```
<table>
    <tr>
        <th>Name</th>
        <th>Gender</th>
        <th>Age</th>
    </tr>
    <tr>
        <td>Mads</td>
       <td>Male</td>
       <td>21</td>
    </tr>
    <tr>
        <td>Annie</td>
       <td>Female</td>
       <td>24</td>
    </tr>
</table>
```

Divs & Spans

- → The <div> element is the most popular way to group and sort elements that don't adhere to a specific HTML element type.
- → Block, inline and inline-block are CSS Display properties



A <div> is of the type

“block”

A <span> is

Of the type

“inline”

Forms

```
<form action=“/” method=“GET”>
    <fieldset>
        <legend>Explanation</legend>
        <input type=“text” placeholder=“write here”/>
        <input type=“password” tabindex=“1” autofocus/>
        <input type=“file” name=“file”/>
        <input type=“submit” tabindex=“2” >
        <button></button>
    </fieldset>
</form>
```

```
<form action=“url.html” method=“GET”>
        <label for=“username”>Username</label>
        <input type=“text” placeholder=“Enter username
        here” id=“username” name=“username”/>

        <label for=“password”>Password</label>
        <input type=“password” placeholder=“Enter
        password” id=“password” name=“password”/>
</form>
```

Name and ID should always be identical

Internet Explorer can not always tell the difference

Forms input types

Common

- Type
- Name
- Autofocus
- Tabindex
- Placeholder
- Value

Common

- Text
- Password
- Submit (use button element instead)
- Reset
- Button (use onclick attribute)
- Checkbox
- Radiobuttons
- Dropdown menu (select option)
- Hidden

HTML 5 additions

- Color
- Date
- Email
- Month
- Number
- Range
- Search
- Tel
- Time
- Url
- Week

Common HTML attributes not specific for inputs

- Class
- Id

HTML Entities

- → <br/> is a linebreak
  - → No linebreaks from text are printed by default
- → &nbsp; is a space
  - → Only single spaces from text is printed by default
- → Complete reference:
  - → https://dev.w3.org/html5/html-author/charref

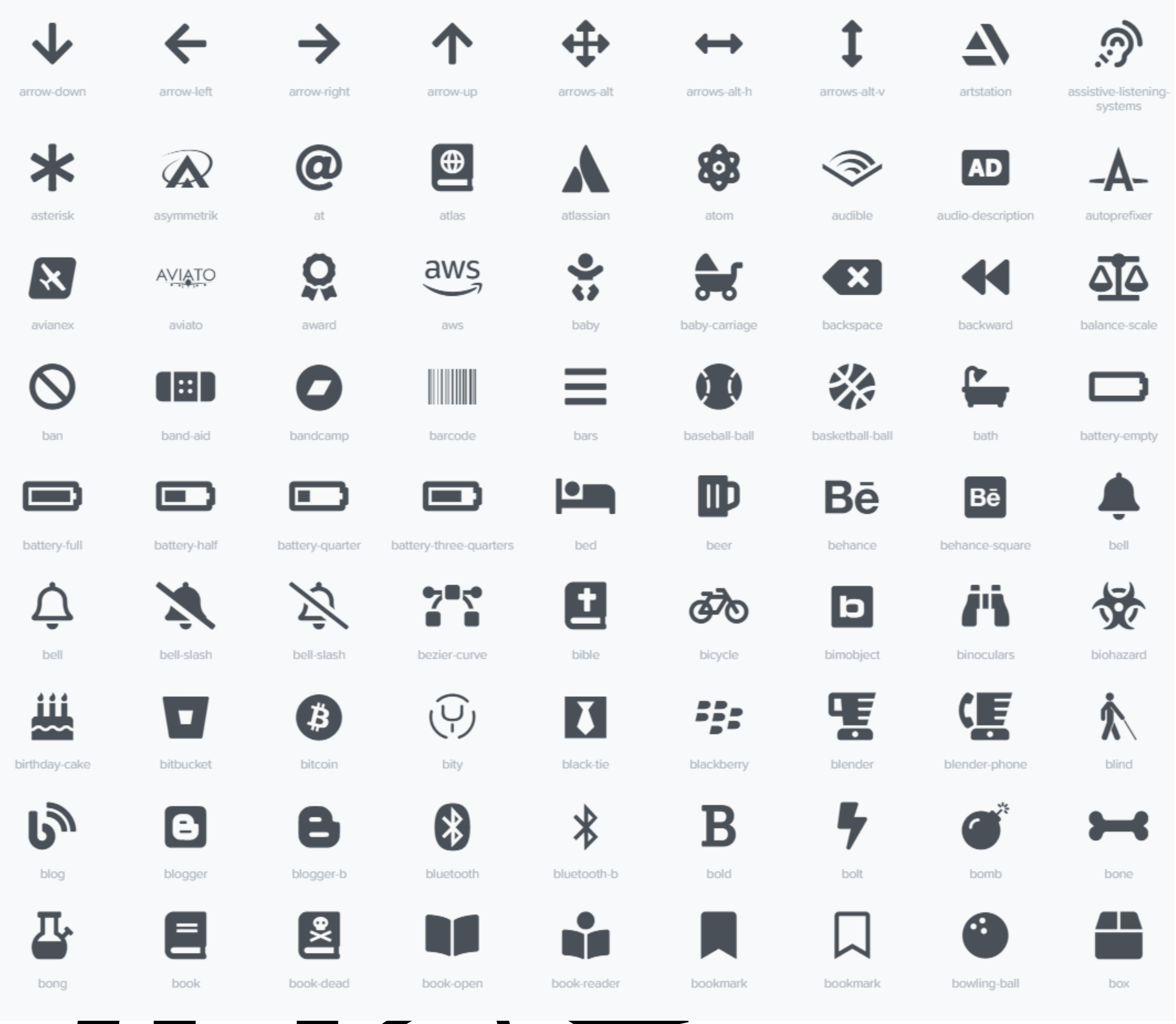
Sign	Description	Entity Name	Entity Number
	Space	&nbsp;	&#160;
<	Less than	&lt;	&#60;
>	Greater than	&gt;	&#62;
&	Ampersand	&amp;	&#38;
"	Double quotation mark	&quot;	&#34;
'	Single quotation mark/apostrophe	&apos;	&#39;
¢	Cent	&cent;	&#162;
£	Pound	&pound;	&#163;
¥	Yen	&yen;	&#165;
€	Euro	&euro;	&#8364;
©	Copyright	&copy;	&#169;
®	Registered trademark	&reg;	&#174;

Colors

- → Blue, Red, White, Black
- → Hex: #AA00FF
- → rgb(255,0,0)
- → rgba(255,0,0,0.5)
  - → a is for alpha

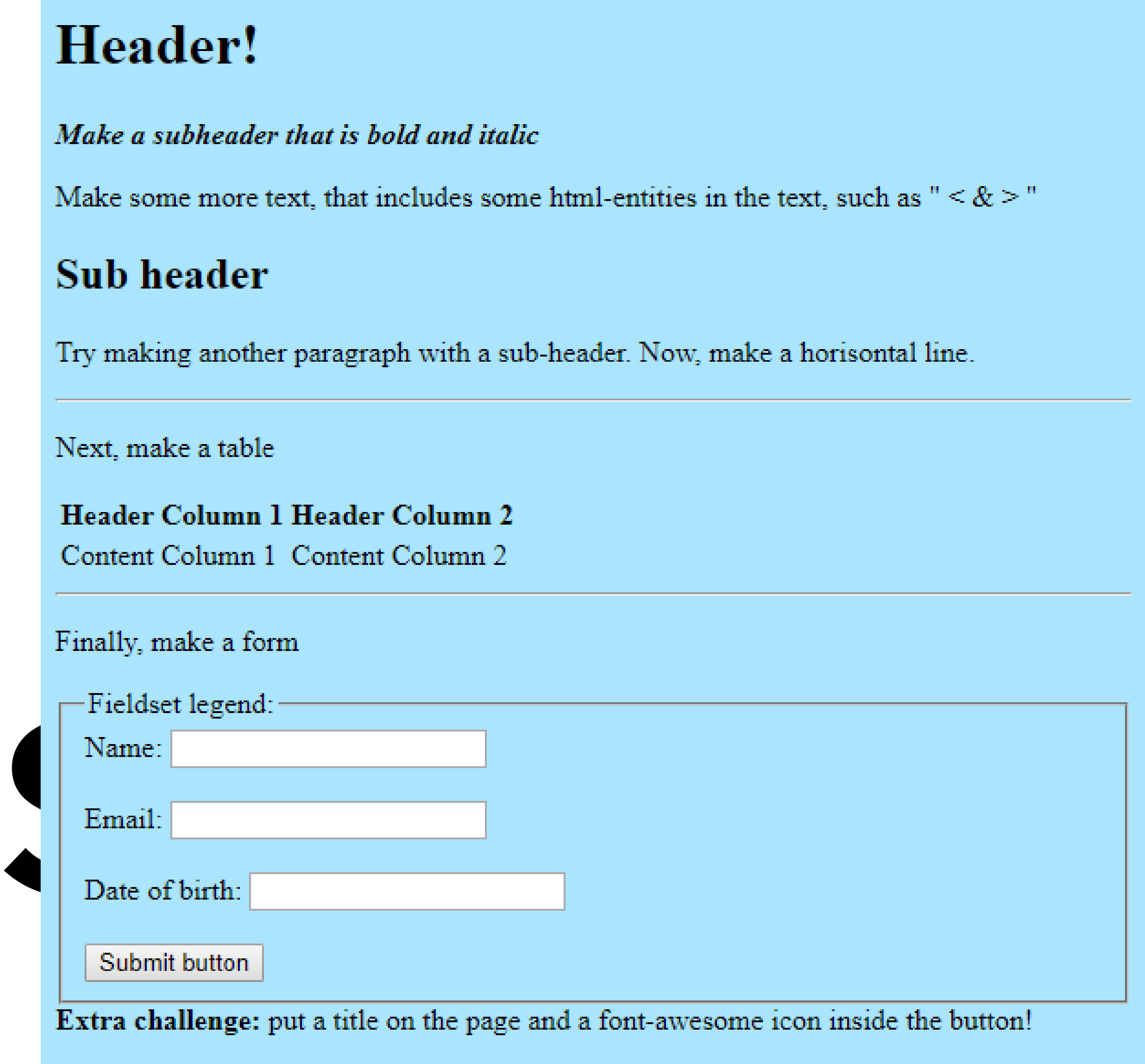
Font Awesome

- Use in your HTML by including this in the head:
  - <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.3/css/all.css" integrity="sha384-UHRtZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTSRyMA2Fd33n5dQ8lWUE00s/" crossorigin="anonymous">
  - Instructions/Link can be found here:
  - https://fontawesome.com/start
- See all free icons
  - https://fontawesome.com/icons?d=gallery&m=free
- Use an icon like this
  - <i class="fas fa-arrow-down"></i> =
- Can be colored and sized like fonts
- All types of icons, including jokes, tools, arrows, logos, etc



Exercise

→ Let's build this!



Header!

Make a subheader that is bold and italic

Make some more text, that includes some html-entities in the text, such as " < & > "

Sub header

Try making another paragraph with a sub-header. Now, make a horisontal line.

Next, make a table

Header Column 1	Header Column 2
Content Column 1	Content Column 2

Finally, make a form

Fieldset legend:

Name:

Email:

Date of birth:

Extra challenge: put a title on the page and a font-awesome icon inside the button!

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

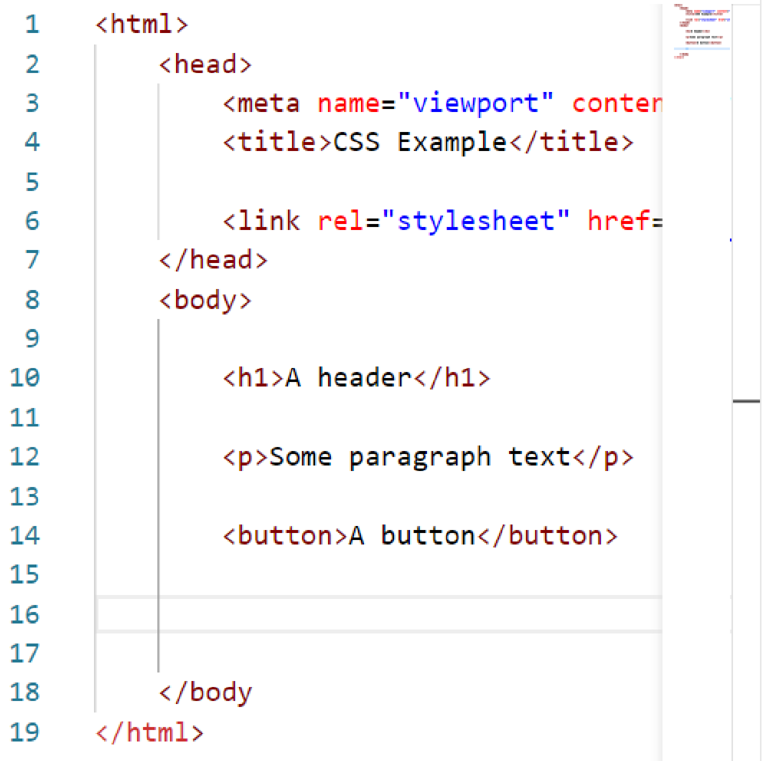
CSS

September 2020

CSS

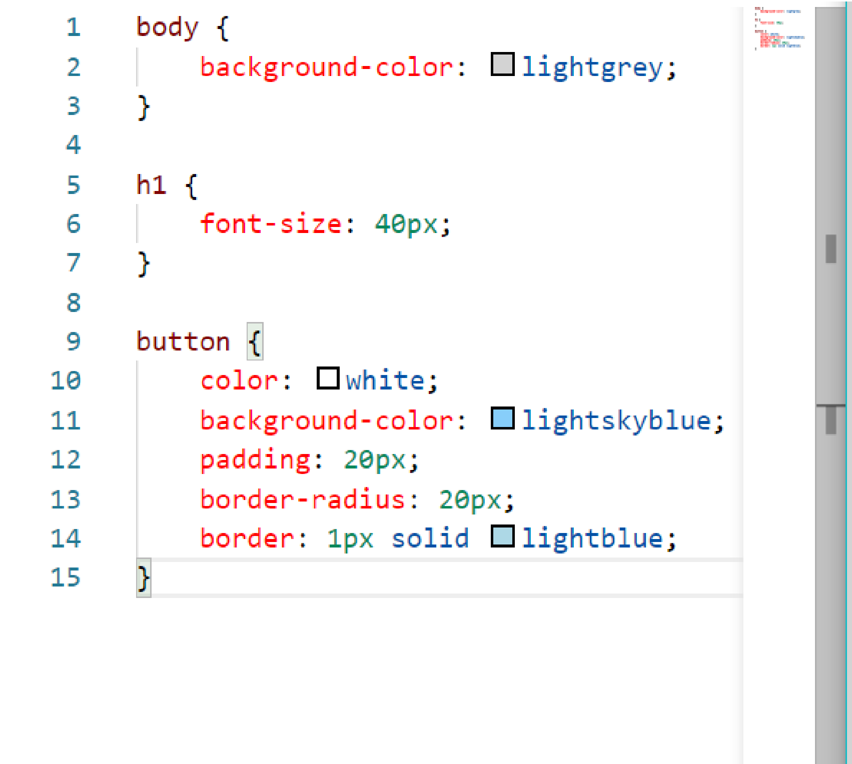
- → Cascading Style Sheets
- → Defines how the HTML looks
- → Most HTML markup attributes have CSS equivalents
- → HTML is the content and structure
- → CSS is the way that content looks

HTML



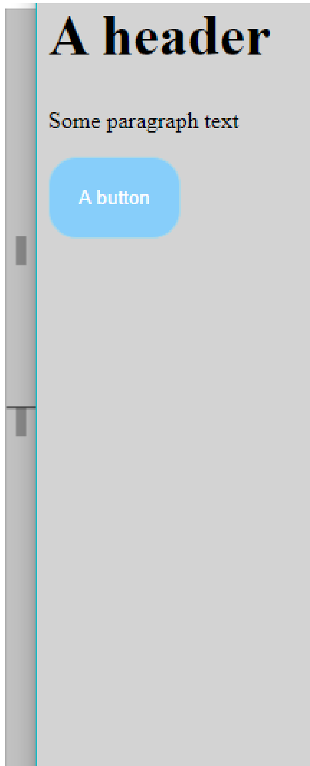
```
<html>
    <head>
        <meta name="viewport" conter
        <title>CSS Example</title>

        <link rel="stylesheet" href=
    </head>
    <body>

        <h1>A header</h1>

        <p>Some paragraph text</p>

        <button>A button</button>



    </body
</html>
```

CSS



```
body {
    background-color: □lightgrey;
}

h1 {
    font-size: 40px;
}

button {
    color: □white;
    background-color: □lightskyblue;
    padding: 20px;
    border-radius: 20px;
    border: 1px solid □lightblue;
}
```

Result



Where do we put it?

- → CSS can generally be in 3 different places:
- → HTML attributes:
  - → <div style=“color: red”>RED</div>
- → HTML Elements:
  - → <style>CSS GOES HERE</style>
- → Separate .css file
  - → Link in HTML head
  - → <link rel="stylesheet" type="text/css" href="style.css">
  - → This is generally the preferred way to do it!

- → CSS can import .css files from .css files or <style> elements using the @import annotation.

Syntax

- → The name of an element type points to elements of that type
- → Class names are written with a leading dot (.class)
- → ID’s are written with a leading hashtag (#id)
- → Multiple pointers are written after each other
  - → Without space!
- → Space denotes nesting. The example would point to the subelement.

```
<element class=“class” id=“id”>
   <subelement>content</subelement>
</element>

CSS:
element.class#id subelement {
   /*comment (not // !) */
   color: blue;
   background-color: #000000;
   border: 2px solid rgba(0, 255, 124, 0.5);
```

Selector Syntax Examples

div p b

→ Would point to all bold text, in paragraphs, in div elements

div.invisible

→ Would point to all div elements with the class “invisible”

span.blue.button

→ Would point to all span elements were the classes “blue” and “button” are both present (<span class=“blue button”>)

#title

→ Would point to the element with the ID “title”

div#here .text b

→ Would point to italic text, in any element in the “text” class that are nested in a div with the ID “here”

Selector Syntax Examples

```
input[type=“text”] {
  border-radius: 5px;
}
```

→ This is how you select a specific type of form input

```
a:hover {
    color: lightblue;
}
```

→ This pseudo-selector takes all links when the mouse hovers over it

→ Other pseudo classes:

- → https://www.w3schools.com/css/css_pseudo_classes.asp

Pseudo class selectors

unvisited link

```
a:link {
  color: #FF0000;
}
```

visited link

```
a:visited {
  color: #00FF00;
}
```

mouse over link

```
a:hover {
  color: #FF00FF;
}
```

selected link

```
a:active {
  color: #0000FF;
}
```

Selector Syntax Examples

HTML:

```
<meta name="viewport"
  content="width=device-width, initial-scale=1.0">
```

CSS:

```
@media only screen and (max-width: 600px) {
  body {
    background-color: lightblue;
  }
}
```

→ The media annotation syntax only works when viewport is defined in HTML

Elements, Classes and ID’s

```
<div class=”classA” id=“idA” style=“color: blue;”>
   What color will this be?
</div>
<style>
   div { color: red; }
   .classA { color: green; }
   #idA { color: purple }
</style>
```

Elements, Classes and ID’s

```
<div class=”classA” id=“idA” style=“color: blue;”>
    What color will this be?
</div>
<style>
    div { color: red; }
    .classA { color: green; }
    #idA { color: purple }
</style>
```

BLUE

But why?

Specificity

→ In most programming languages, a variable will retain the **last** value that was assigned to it.
→ This is **NOT** the case for CSS
→ Instead, CSS relies on a point-system, where the assignment with the most points win
→ This is called “Specificity”

	!important	HTML style attribute	ID	Class Pseudo-class Attribute	Element
div.text {}				1	1
div#title b {}			1	0	2
a:hover {}				1	1
<div style=“color: black”>		1	0	0	0

**a {
color: white !important;
}**
	1	0	0	0	1

The Maersk Mc-Kinney Moller Institute

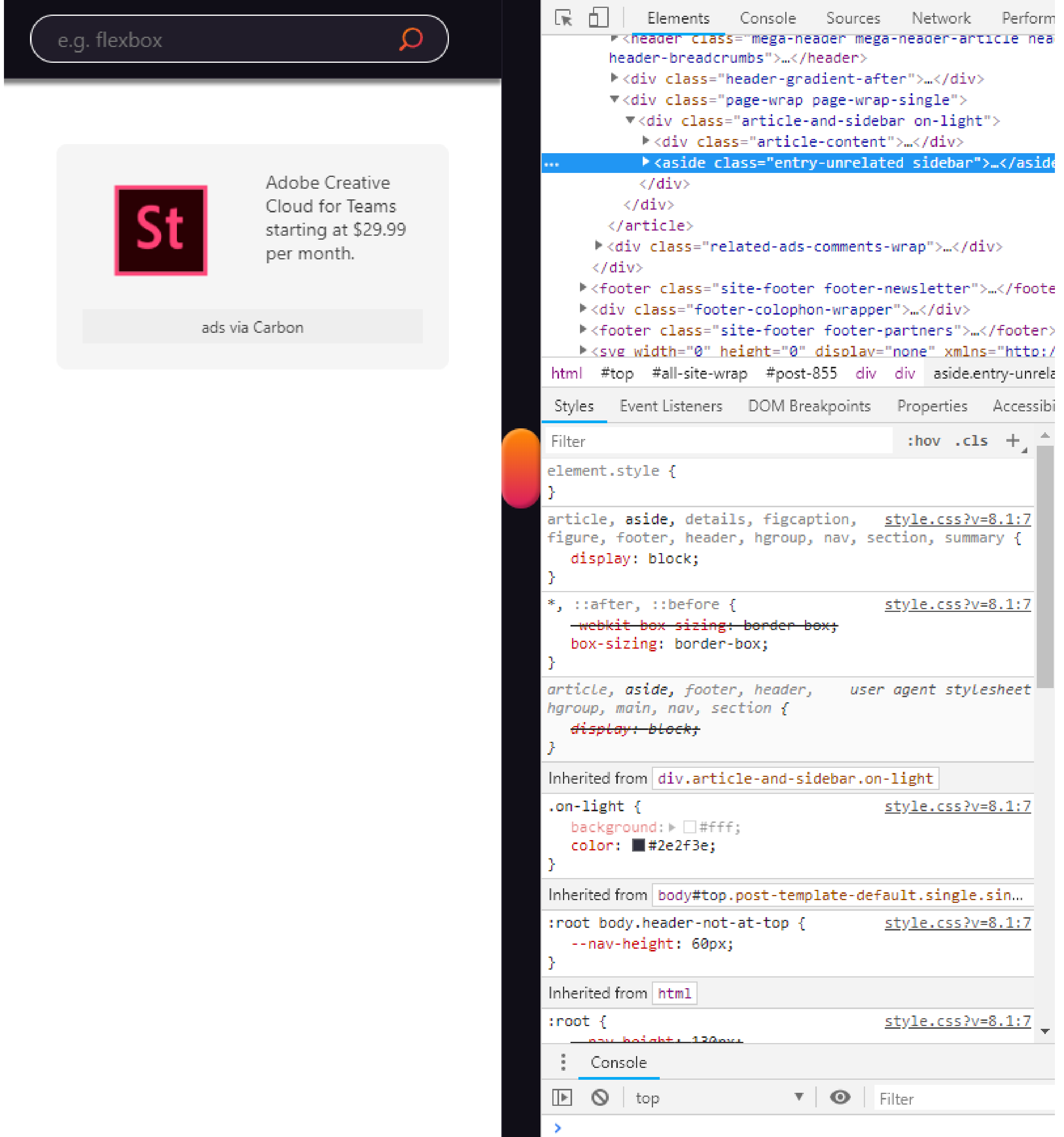
Developer Tools

→ This is important to understand when you import external libraries

→ Chrome developer tools helps us to see what is chosen

  → ~~Lines through ignored code~~

→ Select DOM element, and Chrome will mark it on the site and show CSS elements with influence



```
▸<header class="mega-header mega-header-article hea
  header-breadcrumbs">…</header>
 ▸<div class="header-gradient-after">…</div>
 ▾<div class="page-wrap page-wrap-single">
   ▾<div class="article-and-sidebar on-light">
     ▸<div class="article-content">…</div>
     ▸<aside class="entry-unrelated sidebar">…</aside
     </div>
   </div>
 </article>
▸<div class="related-ads-comments-wrap">…</div>
</div>
▸<footer class="site-footer footer-newsletter">…</foote
▸<div class="footer-colophon-wrapper">…</div>
▸<footer class="site-footer footer-partners">…</footer>
▸<svg width="0" height="0" display="none" xmlns="http:/
```

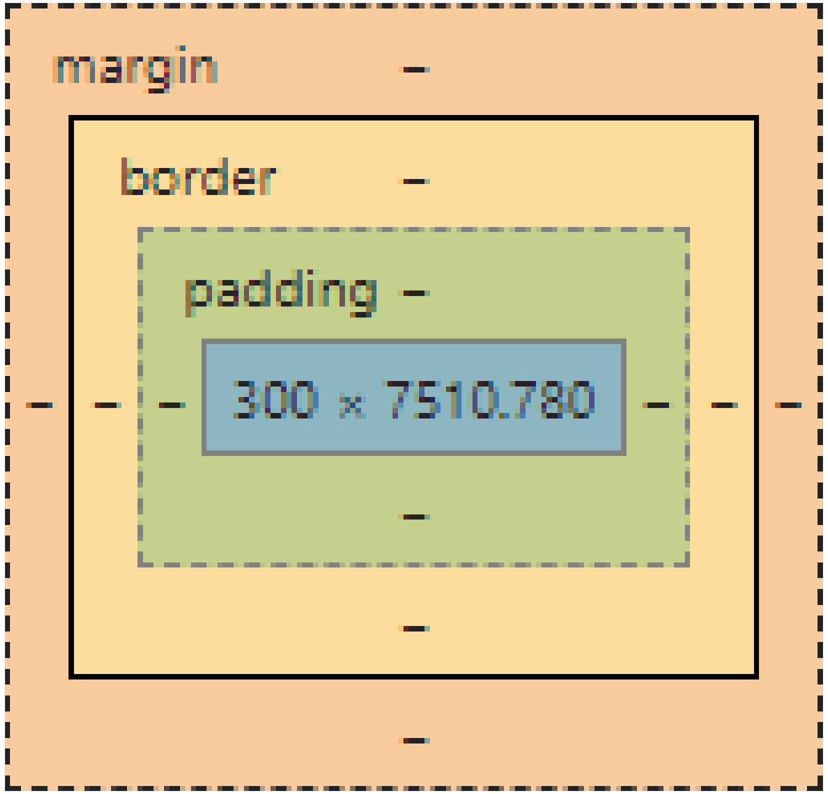
html #top #all-site-wrap #post-855 div div aside.entry-unrela

```
element.style {
}
article, aside, details, figcaption,        style.css?v=8.1:7
figure, footer, header, hgroup, nav, section, summary {
   display: block;
}
*, ::after, ::before {                      style.css?v=8.1:7
   ~~-webkit-box-sizing: border-box;~~
   box-sizing: border-box;
}
article, aside, footer, header,       user agent stylesheet
hgroup, main, nav, section {
   ~~display: block;~~
}
Inherited from div.article-and-sidebar.on-light
.on-light {                                 style.css?v=8.1:7
   background: ▸ □#fff;
   color: ■#2e2f3e;
}
Inherited from body#top.post-template-default.single.sin…
:root body.header-not-at-top {              style.css?v=8.1:7
   --nav-height: 60px;
}
Inherited from html
:root {                                     style.css?v=8.1:7
   ~~--nav-height: 120px;~~
```

```
>
```

The Box Model

- → All elements are in a square
- → Border is the edge of this square
- → Content size defines the size of the element
- → Padding is the space between content and border
- → Margin is distance to next elements margin
- → Chrome can show these numbers and lets you preview alternative sizes
- → If height and width is set
  - → Padding is added on top



Padding

→ Between object and border

→ padding: 0px; //all 4 directions
→ padding: 0px 10px; //top/bottom left/right
→ padding: 0px 0px 0px 0px; //top right bottom left

→ padding-top: 0px;
→ padding-left: 0px;
→ padding-bottom: 0px;
→ padding-right: 0px;
→ **Negative numbers!**

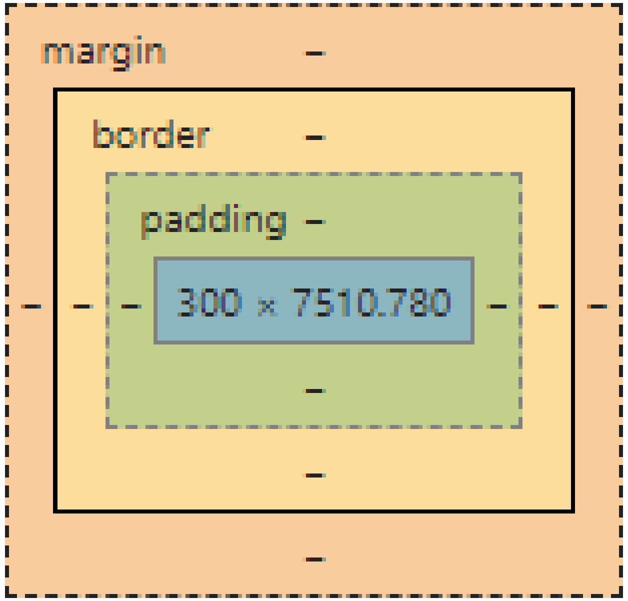
Margin

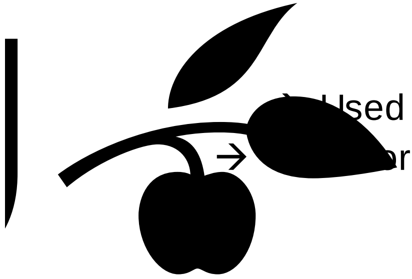
→ Space around the border

→ margin: 0px; //all 4 directions
→ margin: 0px 10px; //top/bottom left/right
→ margin: 0px 0px 0px 0px; //top right bottom left

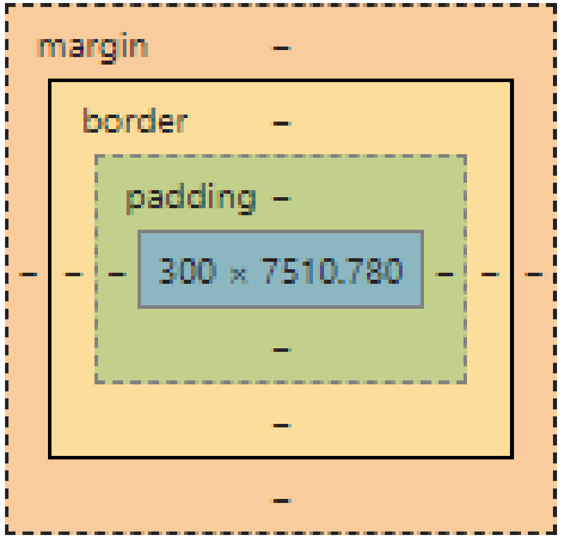
→ margin-top: 0px;
→ left, right, bottom
→ **Negative numbers!**

→ margin: 0 auto;
→ Used to center an element
→ Works when **width** is set





Border



→ border: 0px solid; //all 4 directions
→ border: 0px solid red; //top/bottom left/right

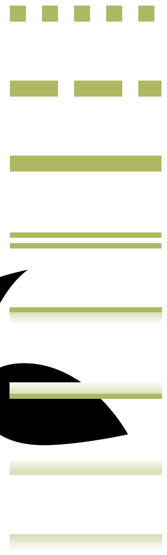
→ border-style: solid;
→ border-style: solid dotted dashed inset;

→ border-top: 1px solid;
→ border-left: 2px dotted;

→ border-radius: 10px;

Border-styles:

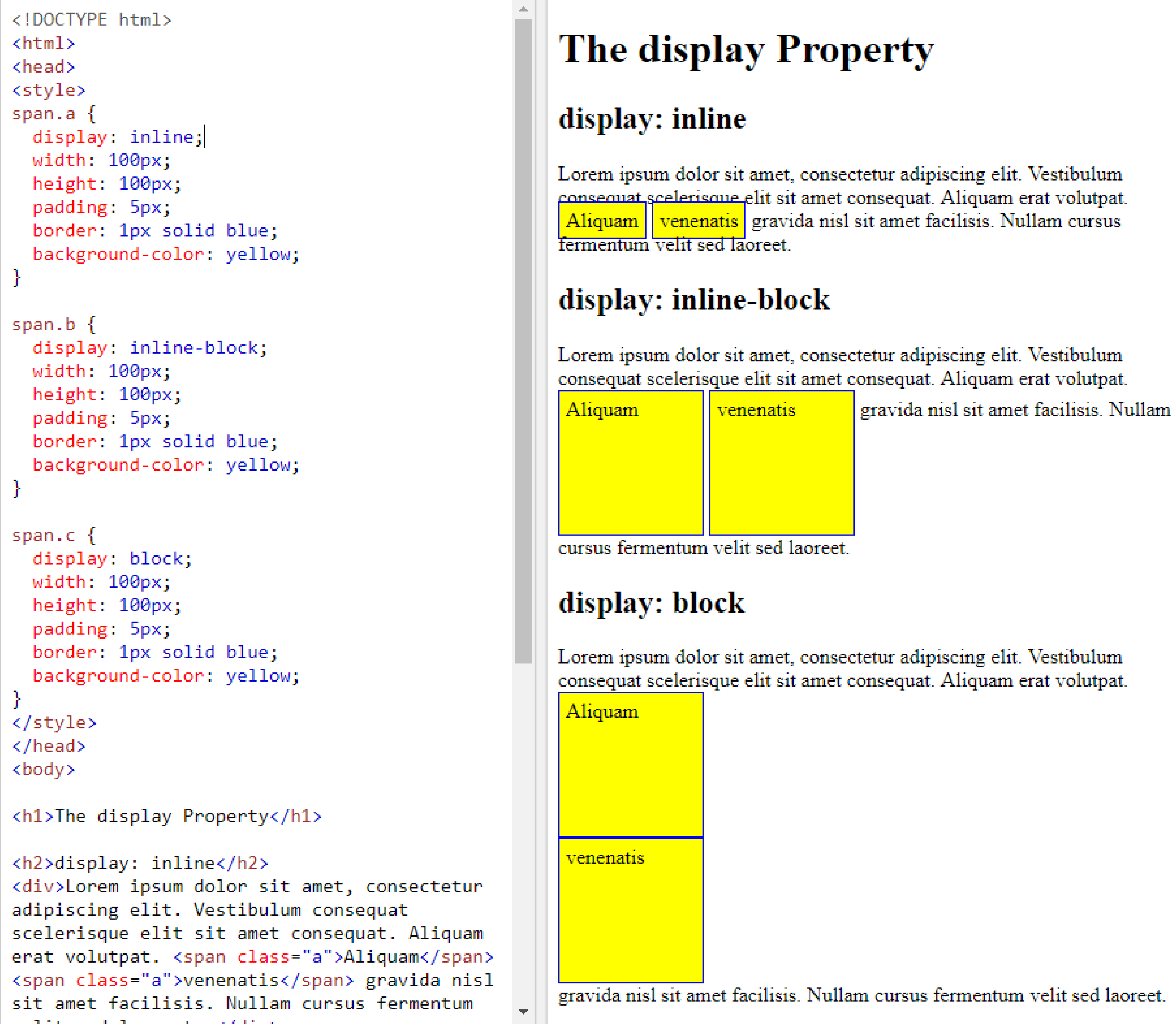
- dotted
- dashed
- solid
- double
- groove
- ridge
- inset
- outset
- none
- hidden



Display: Block, Inline, Inline-Block

→ display: block
  → Respects size properties
  → Adds linebreak after it
→ display: inline
  → Ignores size properties to stay in line
  → No linebreaks inserted
→ display: inline-block
  → Respects size properties
  → Does not add linebreak

→ Use height and width for blocks and inline-blocks



```
<!DOCTYPE html>
<html>
<head>
<style>
span.a {
  display: inline;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}

span.b {
  display: inline-block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}

span.c {
  display: block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}
</style>
</head>
<body>

<h1>The display Property</h1>

<h2>display: inline</h2>
<div>Lorem ipsum dolor sit amet, consectetur
adipiscing elit. Vestibulum consequat
scelerisque elit sit amet consequat. Aliquam
erat volutpat. <span class="a">Aliquam</span>
<span class="a">venenatis</span> gravida nisl
sit amet facilisis. Nullam cursus fermentum
```

The display Property

display: inline

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat. Aliquam venenatis gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet.

display: inline-block

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat. Aliquam venenatis gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet.

display: block

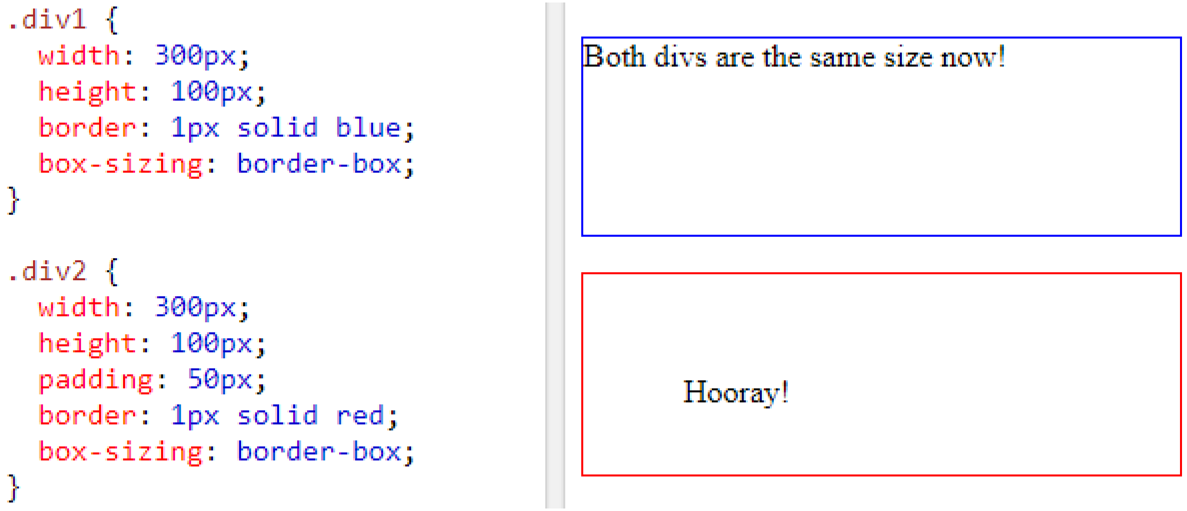
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat.

Aliquam

venenatis

gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet.

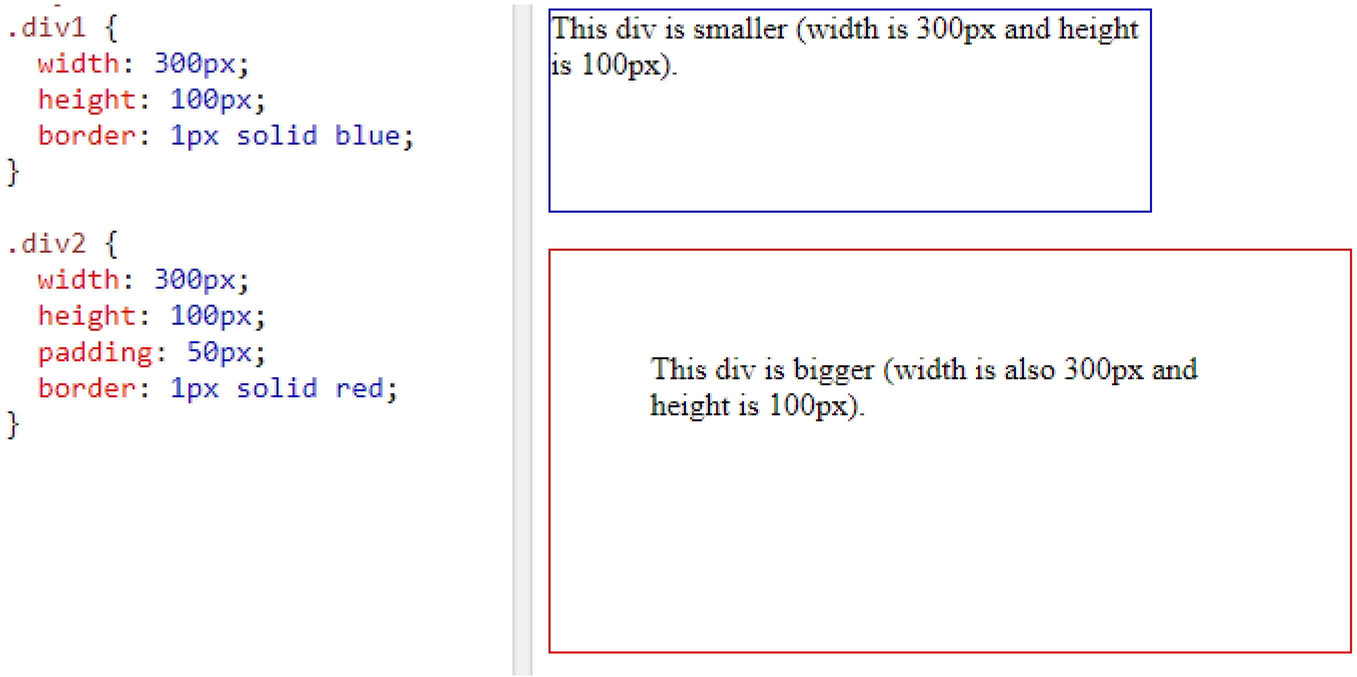
Border-box



```
.div1 {
  width: 300px;
  height: 100px;
  border: 1px solid blue;
  box-sizing: border-box;
}

.div2 {
  width: 300px;
  height: 100px;
  padding: 50px;
  border: 1px solid red;
  box-sizing: border-box;
}
```

Both divs are the same size now!

Hooray!



```
.div1 {
  width: 300px;
  height: 100px;
  border: 1px solid blue;
}

.div2 {
  width: 300px;
  height: 100px;
  padding: 50px;
  border: 1px solid red;
}
```

This div is smaller (width is 300px and height is 100px).

This div is bigger (width is also 300px and height is 100px).

https://www.w3schools.com/css/css3_box-sizing.asp

Position: Absolute, Relative or Fixed

- → Absolute position in document
- → Relative to parents (standard)
- → Fixed in window
- → Sticky is relative or fixed, defined by scroll position
- → Static falls in with page flow, ignoring top, right, etc.

Use top, right, bottom and left move element from standard position.

> In most cases, you only select 2 of these, e.g. top & right

Use z-index to decide which elements are in front of –and behind each other.

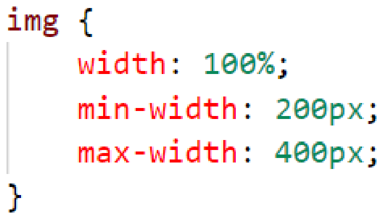
```
div {
    position: absolute;
    top: 10px;
    left: 25%;
    z-index: 2;
}
```

Element Sizes

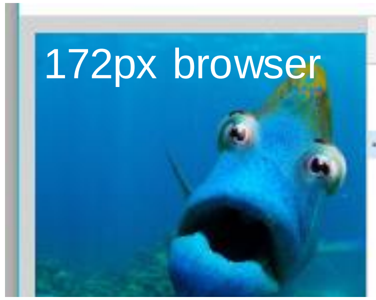
→ width
→ heigth
→ min-height
→ max-height
→ min-width
→ max-width

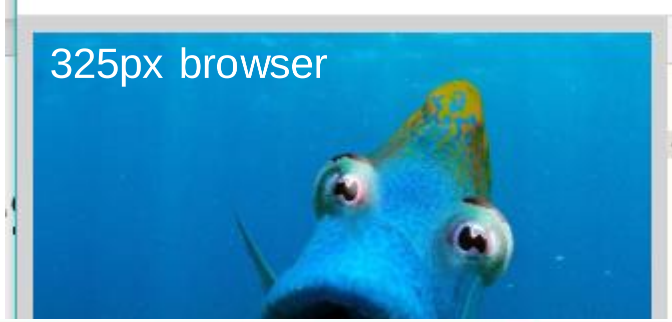
→ 100px
→ 40%

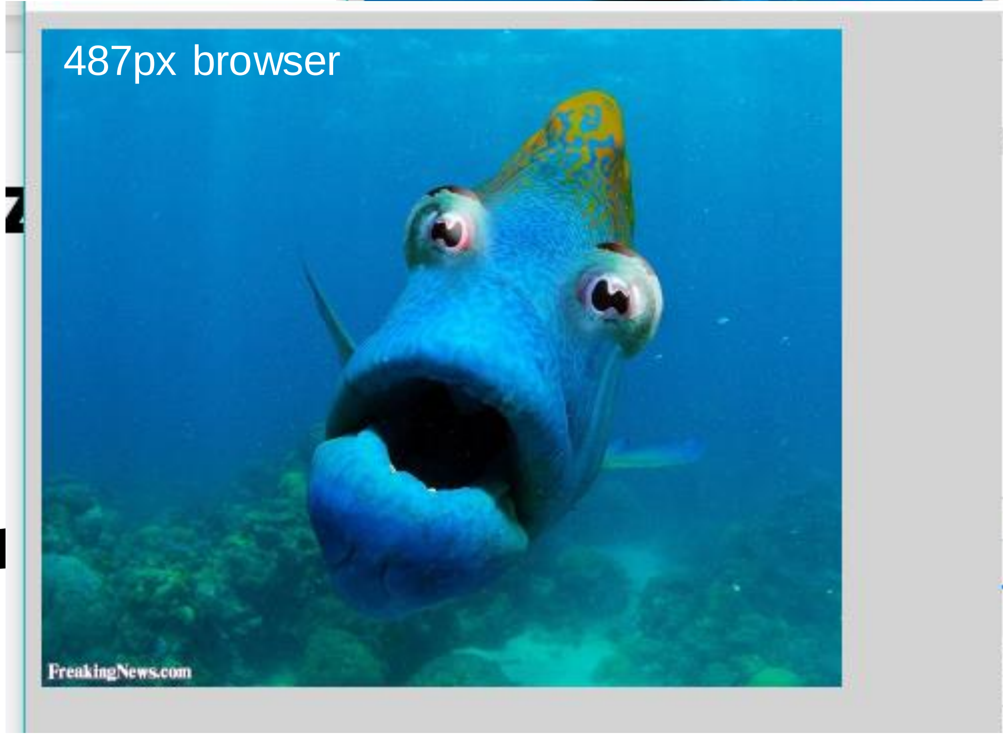
Combine them!



```
img {
    width: 100%;
    min-width: 200px;
    max-width: 400px;
}
```







Text alignment

text-align:

- left
- right
- center
- justify
- initial
- inherit

Left

Lorem ipsum dolor sit amet, in hinc evertitur omittantur pri, aeque bonorum his ut. Eu qui saperet commune, et erant dicam maluisset mea. Solum eirmod ad ius, sale ullum altera mei no. Te aliquam sadipscing sed.

Right

Lorem ipsum dolor sit amet, in hinc evertitur omittantur pri, aeque bonorum his ut. Eu qui saperet commune, et erant dicam maluisset mea. Solum eirmod ad ius, sale ullum altera mei no. Te aliquam sadipscing sed.

Center

Lorem ipsum dolor sit amet, in hinc evertitur omittantur pri, aeque bonorum his ut. Eu qui saperet commune, et erant dicam maluisset mea. Solum eirmod ad ius, sale ullum altera mei no. Te aliquam sadipscing sed.

Justified

Lorem ipsum dolor sit amet, in hinc evertitur omittantur pri, aeque bonorum his ut. Eu qui saperet commune, et erant dicam maluisset mea. Solum eirmod ad ius, sale ullum altera mei no. Te aliquam sadipscing sed.

Images

→ Background images

→ Multiple per element allowed

```
body {
  background-image: url("img1.gif"), url(”img2.gif");
  background-color: #cccccc;
}
```

→ Overflow

→ What happens if the image is bigger than the element that contains it?

```
img {
    width: 200px;
    height: 25%;
    overflow: hidden;
}
```

→ overflow: scroll, hidden, auto or visible

Images – object-fit

→ How should the image be fitted into its element?

```
img {
   width: 200px;
   height: 25%;
   object-fit: cover;
}
```

fill – default – image is stretched to fit
contain – image scales inside element with aspect ratio intact
cover – image scales to fill element with aspect ratio intact
scale-down – like contain, but only scales dow
none – image is not resized

Modals

Learn more about modals

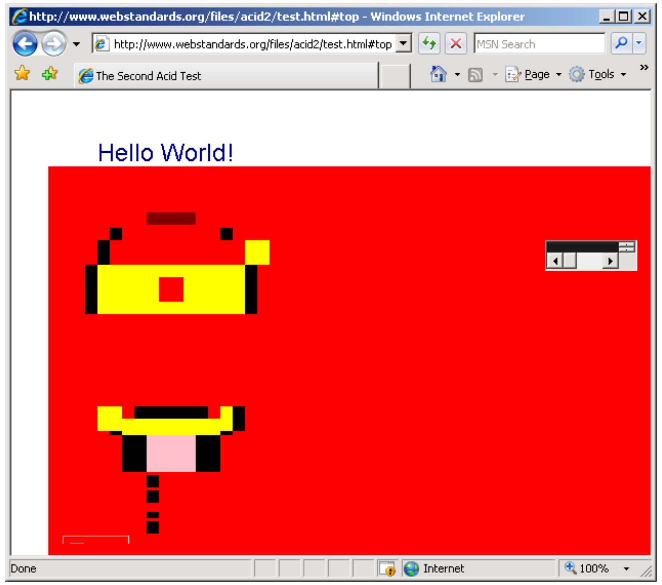
https://www.w3schools.com/howto/howto_css_modals.asp

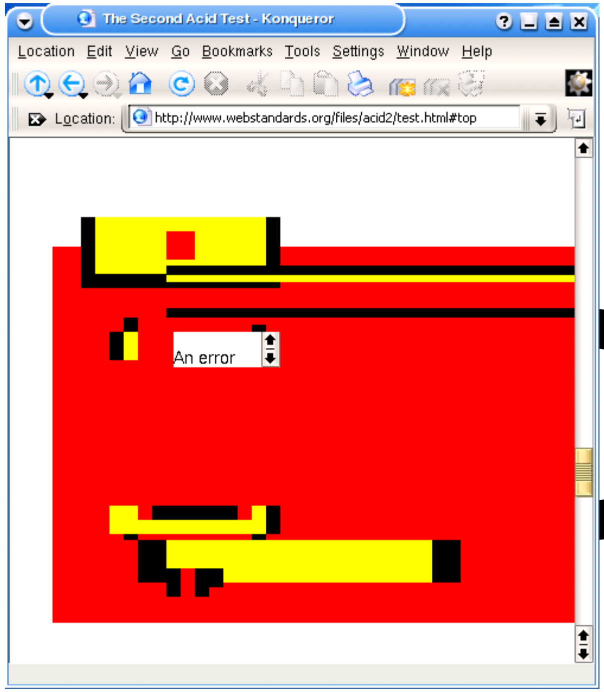
So much more…

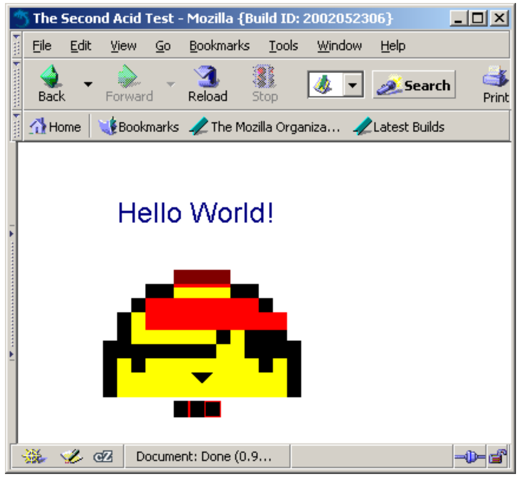
- → CSS is continuously expanded and can already do much more than what we covered
  - → Animate positions on page load
  - → Add content before or after elements
  - → Change the order elements are shown in
  - → Change the users cursor
  - → Set shadows and draw simple objects
- → If you want to do something specific, and you don’t know if CSS supports it (without JavaScript), just search the web for “my-idea pure CSS”.

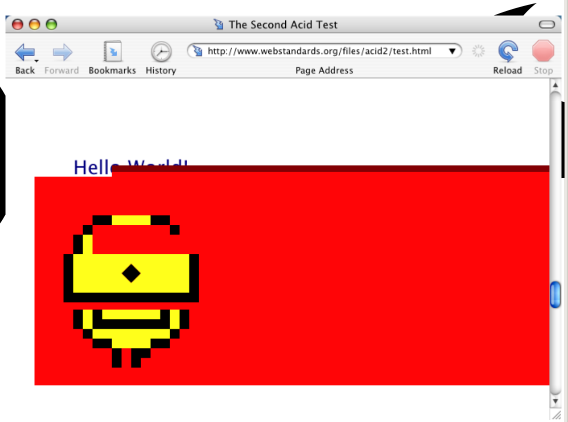
Beware Browsers!

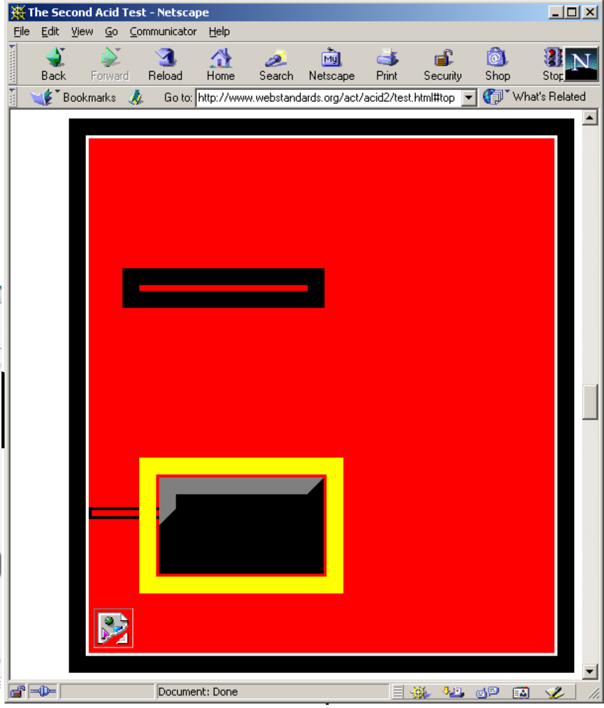
Acid Test 2 – Historically
http://www.howtocreate.co.uk/acid/


http://www.webstandards.org/files/acid2/test.html#top - Windows Internet Explorer
http://www.webstandards.org/files/acid2/test.html#top
MSN Search
The Second Acid Test
Page
Tools
Hello World!
Done
Internet
100%


The Second Acid Test - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location:
http://www.webstandards.org/files/acid2/test.html#top
An error



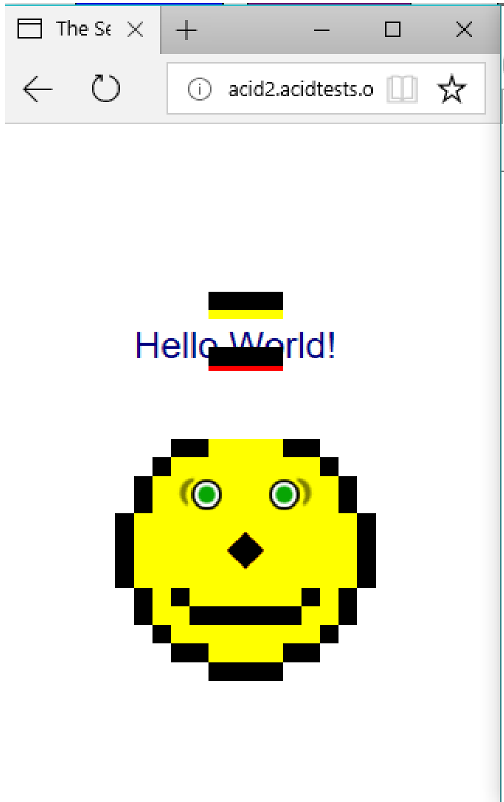



The Second Acid Test - Netscape
File Edit View Go Communicator Help
Back
Forward
Reload
Home
Search
Netscape
Print
Security
Shop
Stop
Bookmarks
Go to:
http://www.webstandards.org/act/acid2/test.html#top
What's Related
Document: Done

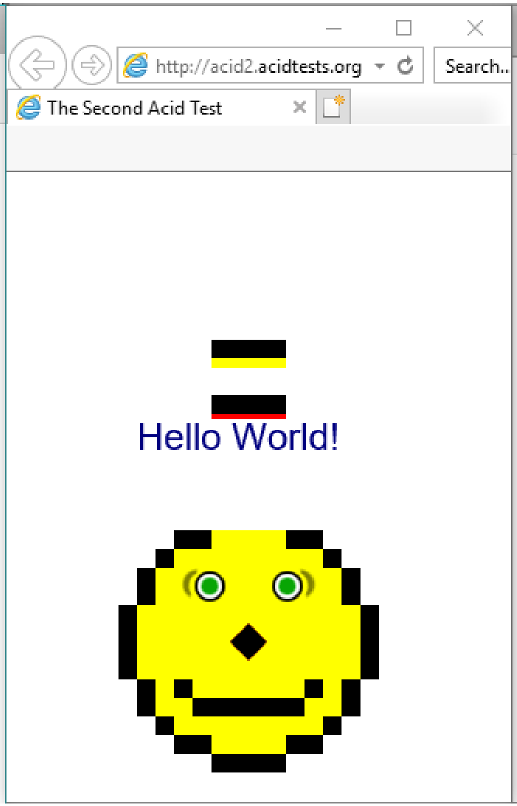
Beware Browsers!

Acid Test 2 – Windows 10 2019
http://www.howtocreate.co.uk/acid/

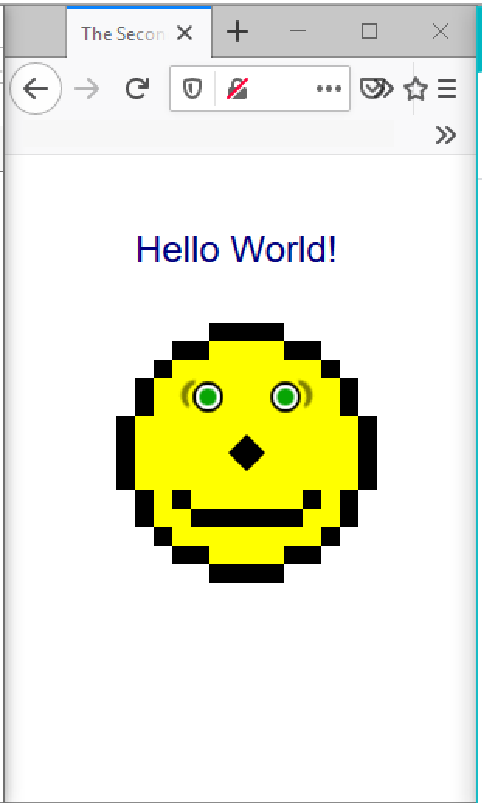
Microsoft Edge 41.16299



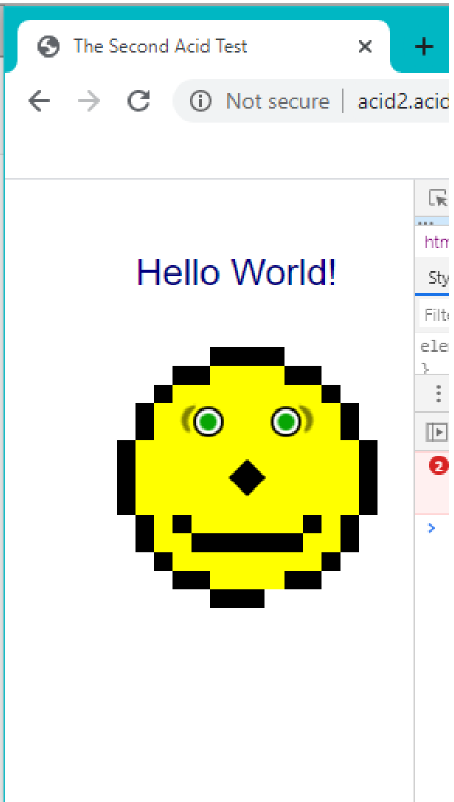
Internet Explorer 11.1622



Firefox 71.0

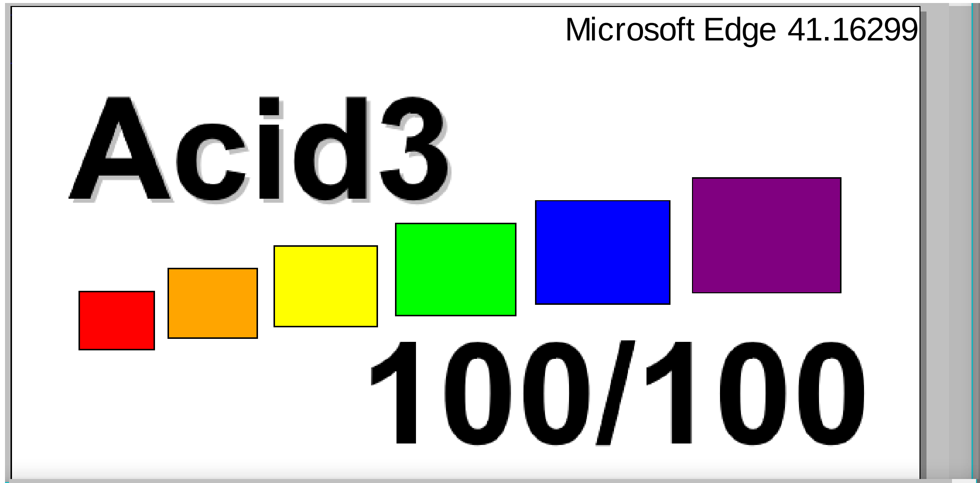


Google Chrome 79.0



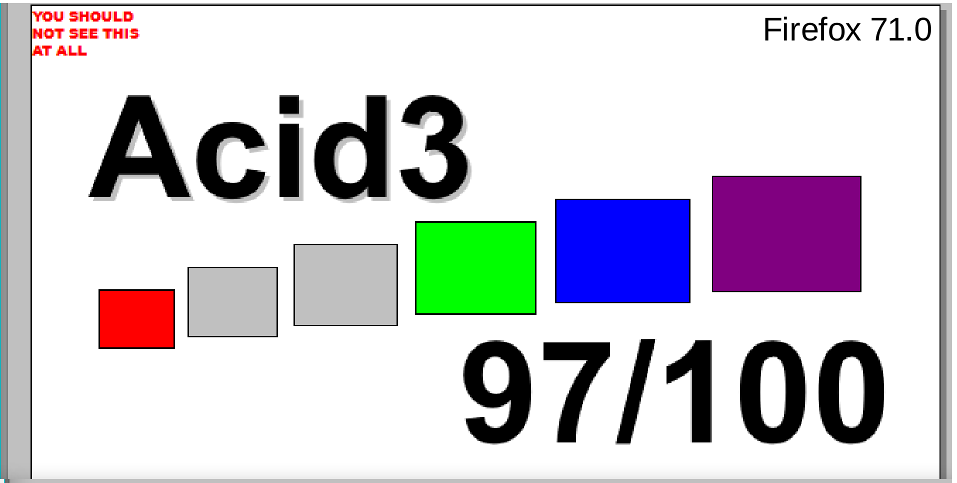
Beware Browsers!

Acid Test 3 – Windows 10 2019
http://www.howtocreate.co.uk/acid/


Microsoft Edge 41.16299

Acid3

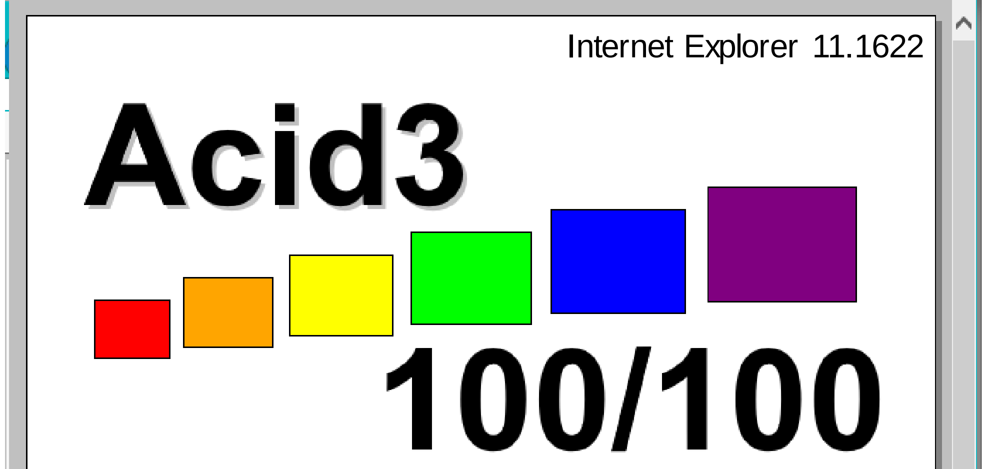
100/100


YOU SHOULD
NOT SEE THIS
AT ALL

Firefox 71.0

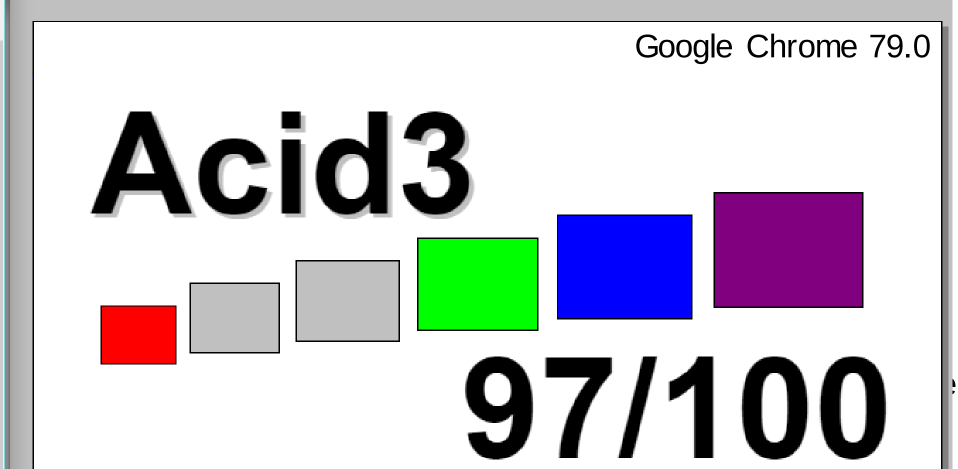
Acid3

97/100


Internet Explorer 11.1622

Acid3

100/100

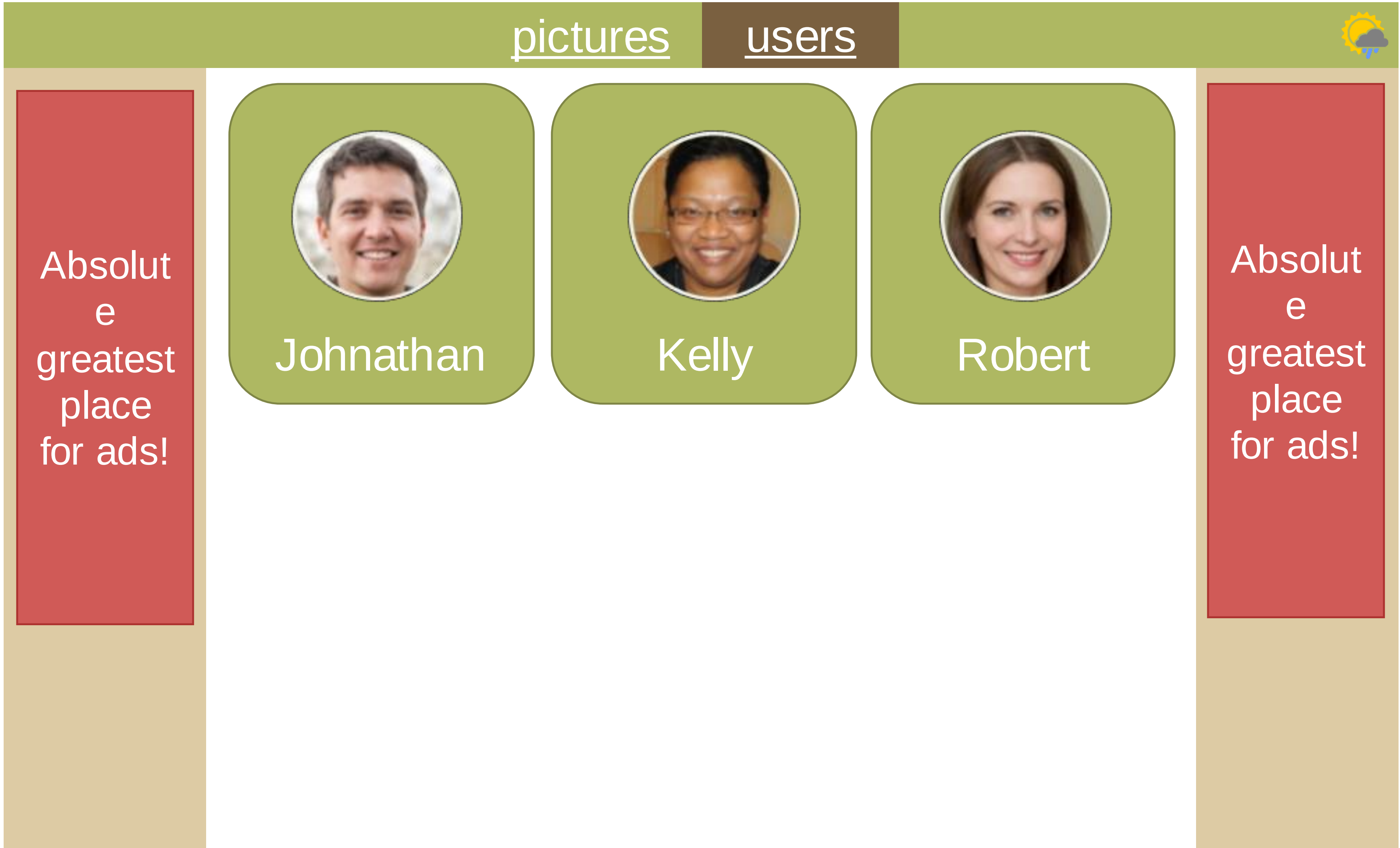

Google Chrome 79.0

Acid3

97/100

Exercise

→ Let's build something like this!



Absolute greatest place for ads!

Absolute greatest place for ads!

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Exercise: “CSS”

SDU

September 2020

Convert your design into code!

- → Imagine that the job is switching hands right now
  - → The designers came up with the graphic designs you made last time and want you to convert it to a homepage
  - → You get the designs, the colorschemes and any image that was used in the design
    - → So if there is a picture with text over it, you get the picture, and then you have to place the text over it, using HTML and CSS.

- → Create your designs in HTML and CSS
  - → No JavaScript, Razor syntax or data implementation necessary
  - → No libraries or frameworks allowed, including jQuery, Bootstrap & Leaflet!
  - → For those of you building a map on your frontend – just put a placeholder for now, as we will talk about frontend maps, using Leaflet, next week.

- → Since this design was created by someone else, you should make the page look as close as possible to the design
  - → If you do something that you feel is prettier than the design, that is not the objective. The objective is to make a front-end that follows the design.

The Maersk Mc-Kinney Moller Institute

Next Lecture: Frameworks jQuery, Bootstrap & Leaflet

sdu.dk

#sdudk

September 2020

Next week

- **jQuery** is a JavaScript library that historically has let you write more functionality with less code
  - Today, it is primarily used for compatibility with older browsers
- **Bootstrap** is a CSS library that makes it much faster to create good-looking front-ends. It has some JavaScript functionality which is built on jQuery
- **Leaflet** is a map system, much like Google Maps. I’m only including this because several of your designs have maps in them.