

Fra i går...

Hvordan gik det med eftermiddags opgaverne?

Javascript og DOM

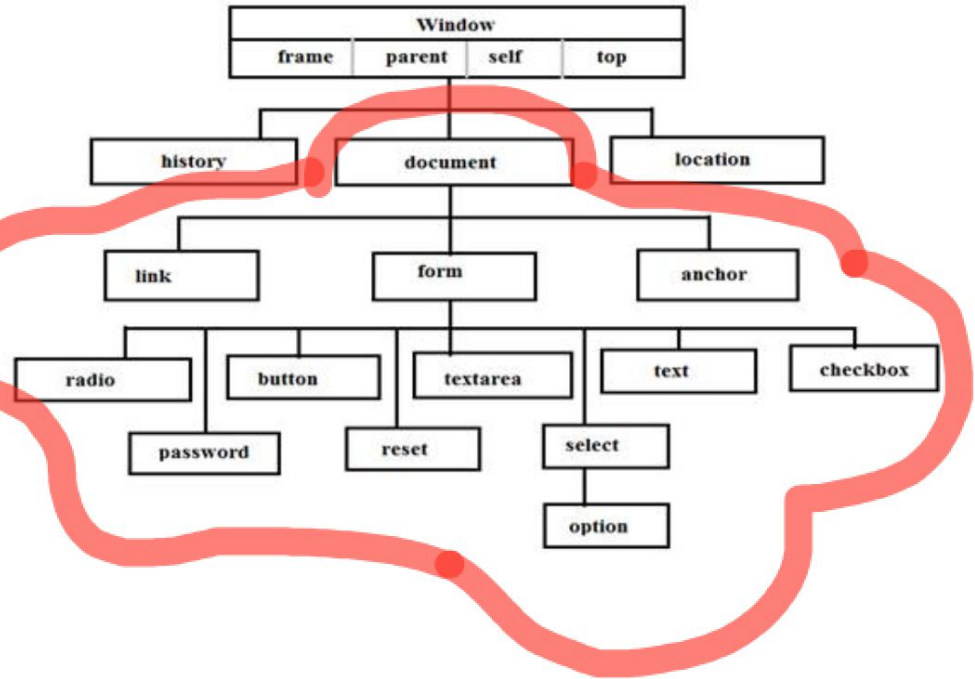
Dagens program

- DOM (Document Object Model)
- HTML struktur recap
- Vælg et DOM-element i javascript
- Læg nyt indhold i DOM-element
- EventListeners og eventhandlers på DOM-element
- Ændringer af css (tilføj/fjern classes og style-property)
- Vælg flere DOM-elementer
- Loops
- Ny datatype: array

DOM

Window- og document-objektet

Browser Object Model



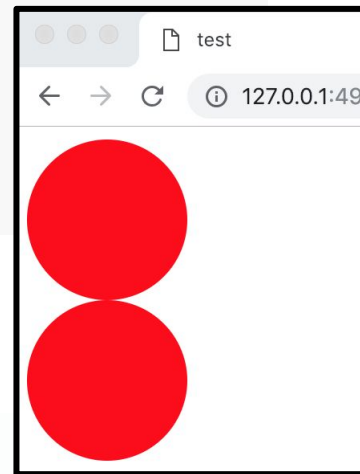
DOM

(Document Object Model)

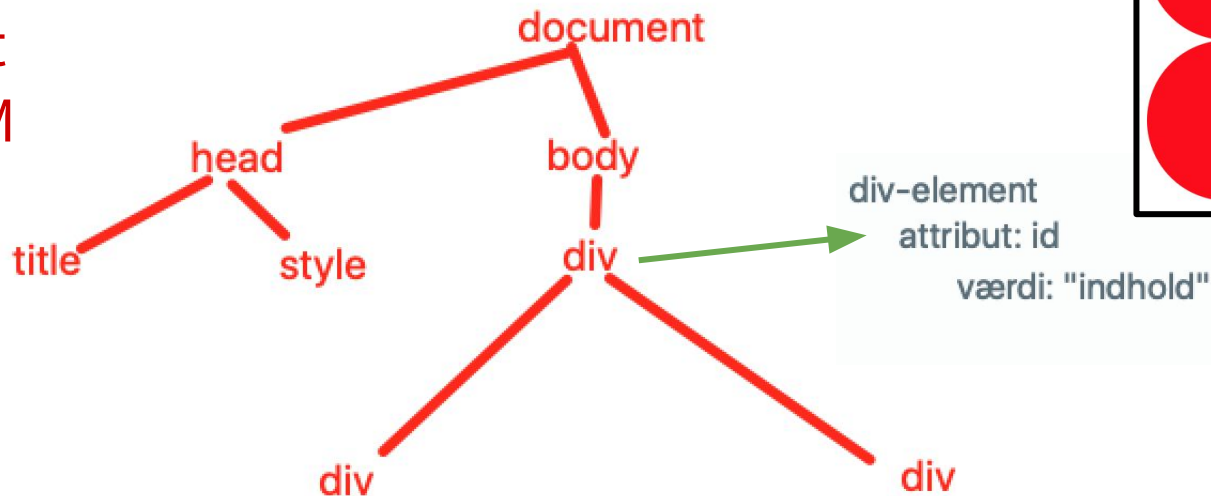
```

1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title>test</title>
5      <style>.kugle{background-color:red; height:100px; width:100px;border-radius:100px}</style>
6    </head>
7    <body>
8      <div id="indhold">
9        <div class="first kugle"></div>
10       <div class="last kugle"></div>
11     </div>
12   </body>
13 </html>|

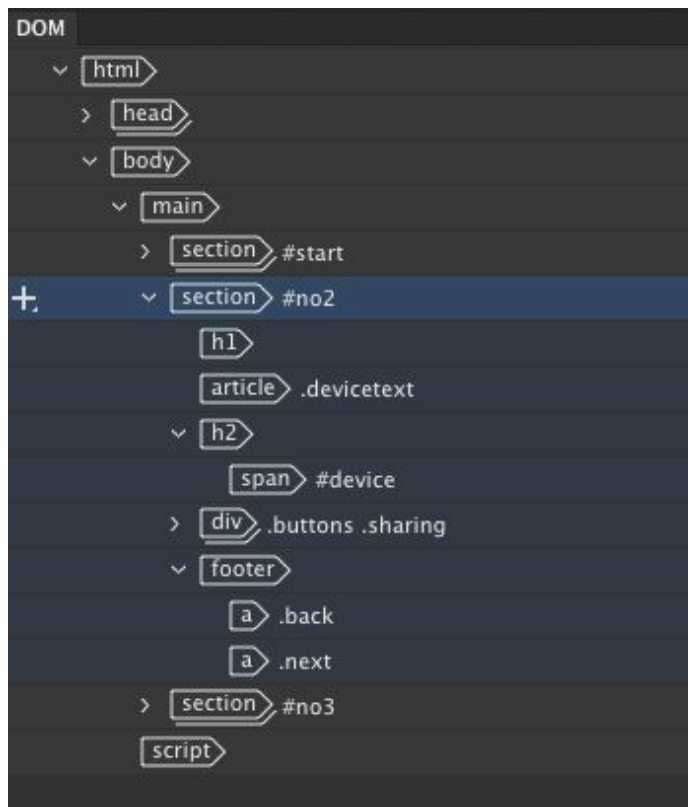
```



Document
ifølge DOM



DOM træ



HTML5 - Semantiske tags

header, nav, main, section, article, footer, details, summary etc.

Hvad er semantiske tags?

Hvad gør de godt for? - bl.a. SEO

Hvordan skal de bruges?

https://www.w3schools.com/html/html5_semantic_elements.asp

https://developer.mozilla.org/en-US/docs/Glossary/Semantics#Semantics_in_HTML

Forberedelse til øvelser

Inden du går i gang med dagens øvelser skal du lave et repository på GitHub til øvelserne, og clone det til en ny mappe på din computer.

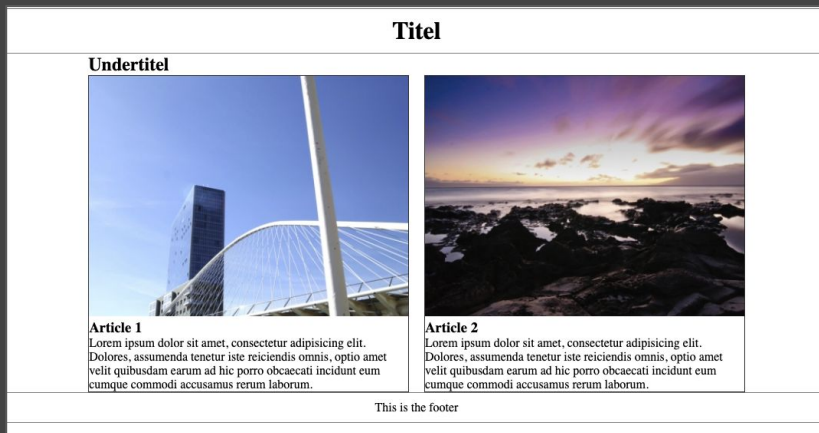
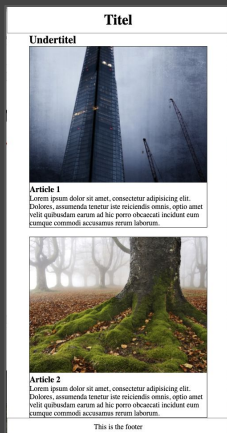
Se vejledningen i slides om Github og Brackets.

Du kan også se denne lille video: <https://youtu.be/UKF0VhwhfYuY>

OBS! Hvis du allerede har lavet et repo til Tema 7, skal du IKKE lave et nyt repo, men bare arbejde videre i en undermappe i dette.

Øvelse 1 - HTML5 struktur

1. lav en ny undermappe, **02-js-DOM** i den mappe hvor du gemmer øvelserne til dette tema. Åbn den i Brackets.
2. Opret en ny html-fil, **html-struktur.html**
(tip! gem et tomt doc som html, og brug EMMET: **! - tab** til at lave et nyt HTML skelet)
3. Opbyg følgende layout med semantiske tags:



4. Commit og push til GitHub

Note!

Eks. på overordnet struktur:

header

main

section

article

article

Footer

Brug flex eller grid (eller begge dele) til at placere elementerne.

Det er ikke nødvendigt at bruge media queries!!!

Se tips til øvelsen på næste slide >>

Tips til øvelse 1

Billeder kan autogenereres på:

<https://placeimg.com>

Vælg en størrelse og evt. Kategori og indsæt den genererede url som src. Attribut.

Ex.: ``

Vælg DOM-element

document.querySelector - (vælg et DOM-element)

Vælg et element ud fra:

ID: `document.querySelector("#idNavn")`


Class: `document.querySelector(".classNavn")`

Element: `document.querySelector("nav .menupunkt a")`

pseudo-selectors:

`document.querySelector("article:first-child")`

`document.querySelector("article:nth-child(2)")`



**Alle CSS-selectors
kan bruges!**

Er der flere elementer med samme class eller tag, vælges kun det første element!

- Hvis man vil have dem alle? - det vender vi tilbage til!

HTML DOM querySelector() Method, w3Schools: https://www.w3schools.com/jsref/met_document_queryselector.asp

Gem querySelector i variabel

Et **DOM-element** valgt med querySelector gemmes tit i en **konstant** variabel, for at man ikke skal gentage selektionen:

```
const info = document.querySelector("#mitElement");  
console.log(info);
```

Hver gang dette element skal bruges i programmet, kan man nu bruge **info**.

Øvelse 2A - Udvælg DOM-elementer

1. Gem en kopi af filen fra øvelse 1 som **02-select.html**
2. Lav querySelectors på alle elementerne fra øvelse 1, og vis dem med console.log()
3. Eksperimentér med **begge** nedenstående metoder. Vælg den du bedst kan lide!
4. Commit og Push til GitHub

Tips!

Du får måske brug for en css-selector til et elements nærmeste søskende:

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

ELLER

Du kan bruge en pseudo selector:

<https://developer.mozilla.org/en-US/docs/Web/CSS/:nth-child>

Opsamling

DOM (Document Object Model) - strukturen i et HTML dokument
querySelector -
css selectors

Læg nyt indhold i
DOM-element

Nyt indhold i element (textContent og innerHTML)

Empty-tags ex.: img, br, hr, input

containertags ex.: body, article, div, h1, p (har slut-tags)

Eksempel på containertag med indhold: `<footer id="info" >her stå noget</footer>`

Man kan ændre på indholdet af containertags med js:

```
const info = document.querySelector("#info");
```

```
info.textContent = "noget helt andet";
```

```
info.innerHTML = "<h1>Noget nyt og anderledes</h1>";
```

Nyt indhold i element (attributter)

Tags der er afhængige af attributter ex.: `img(src)`, `a(href)`

Eksempel på attributter: ``

Man kan ændre på indholdet af attributter med js:

```
const pic = document.querySelector("img");  
pic.src = "etAndetBillede.png";  
Pic.alt = "dette er en alt tekst";
```

Alternativt:

```
pic.setAttribute("src", "etAndetBillede.png");
```

OBS!
Enhver attribut kan ændres
på denne måde!

Øvelse 2B - Udskift tekstindhold i elementer

1. Gem en kopi af filen fra øvelse 2A som **02-indhold.html**
2. Udskift teksten i titlen v.h.af. javascript
3. Udskift tekst og overskrifter i de to article elementer v.h.af. javascript
4. Commit og push til GitHub

Udfordring!

Brug `textContent` til teksten i den ene article, og `innerHTML` med f.eks `` tag i den anden.
Eksperimenter og se forskellen!

Øvelse 3 - Udskift billeder

1. Gem en kopi af filen fra øvelse 2B som **03-indhold.html**

Brug javascript til følgende:

2. Udskift billederne i de to article tags
3. Udskift eller tilføj tekst i alt attributten på billederne.
4. Commit og push til GitHub

Nyt indhold med createElement og appendChild

```
let info = document.querySelector("#info");  
info.innerHTML="<h1>Min overskrift til info</h1><img src='http://mabe-kea.dk/E19/pics/pig.png'>"
```

Alternativ:

```
let h1 = document.createElement("h1");  
let overskrift = document.createTextNode("Min overskrift til info");  
h1.appendChild(overskrift);  
let img = document.createElement("img");  
img.src="http://mabe-kea.dk/E19/pics/pig.png";  
info.appendChild(h1);  
info.appendChild(img);
```

Resultat:

```
<div id="info">  
  <h1>Min overskrift til info</h1>  
    
</div>
```

HTML DOM createElement() Method, : https://www.w3schools.com/jsref/met_document_createelement.asp

HTML DOM appendChild() Method, : https://www.w3schools.com/jsref/met_node_appendchild.asp

textContent vs innerHTML vs createElement/appendChild

- Brug **textContent** til rene tekster
- Brug **innerHTML** til at erstatte/indsætte html
- Brug **insertAdjacentHTML** for at tilføje html
- Brug **createElement** og **appendChild** til html-tilføjelser i en mere struktureret form

Overvejelser ved brug af innerHTML:

<https://developer.mozilla.org/en-US/docs/Web/API/Element/innerHTML>

Øvelse 4 - Tilføj nyt element

Gem **03-indhold.html** i en ny kopi, som du kalder **04-append.html**.

Tilføj nyt javascript i script-tagget, som:

1. opretter et nyt article-element(createElement).
2. tilføjer et billede, en overskrift og noget tekst til den nye article.
3. Indsætter det nye element efter de eksisterende article elementer
4. Commit og push til GitHub.

DOM-manipulationer - endnu flere

```
const info = document.querySelector("#info");
```

```
info.textContent = ... // indsætter tekst i info  
info.textContent += ... // tilføjer tekst til info
```

```
info.setAttribute('attributNavn', 'værdi')  
// sætter en attribut-værdi på info  
info.getAttribute('attributNavn')  
// læser en værdi fra en attribut i info  
info.removeAttribute('attributNavn')  
// fjerner en attribut
```

```
info.src = ... // sætter src-værdi (img-tag)  
info.alt = ... // sætter img's alt-værdi  
info.href = ... // sætter href-værdi (a-tag)
```

```
info.innerHTML = // indsæt html-kode i info  
info.innerHTML += ... // tilføjer html til info  
info.insertAdjacentHTML(position, html-kode)  
// indsætter indhold på en bestemt placering i forhold  
til info
```

Ref: https://www.w3schools.com/jsref/met_node_insertadjacenthtml.asp

```
info.cloneNode(true) // kopierer et element  
let element = document.createElement("img");  
info.appendChild(element) // tilføjer et element  
info.removeChild(element) // fjerner et element
```

Opsamling

textContent

innerHTML

insertAdjacentHTML

createElement > appendChild

EventListener og eventhandler

EventListener & EventHandler

```
let element = document.querySelector("#info");
```

```
element.addEventListener("click", doSomething);
```

```
function doSomething() {  
    // det der skal ske  
}
```

Event - hvad skal der ske?

EventListener

Kald til EventHandler-funktion

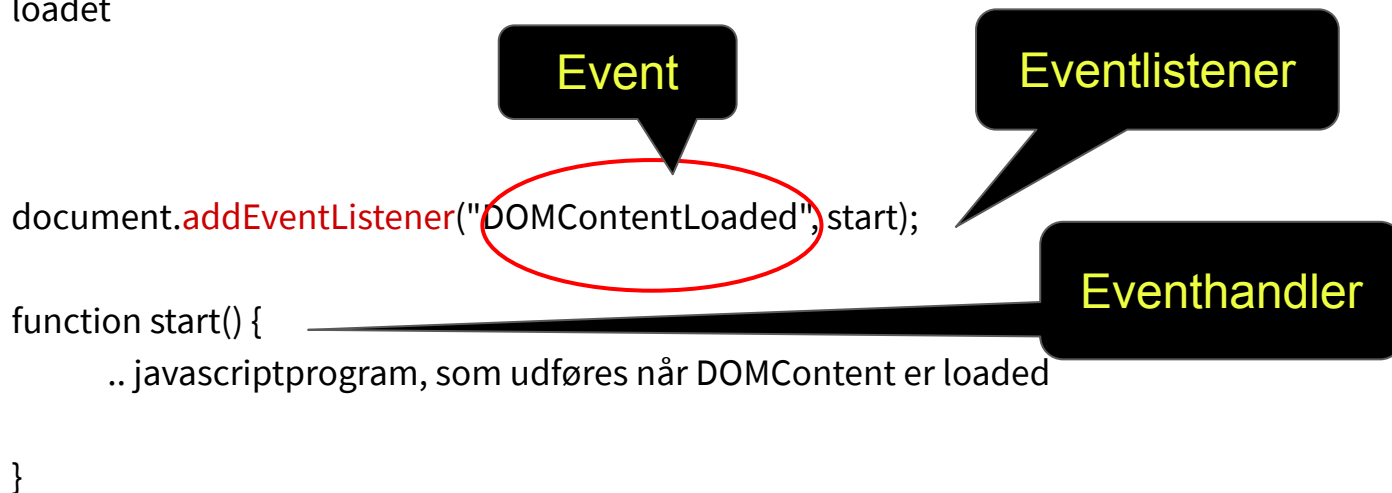
EventHandler
- Den funktion der bliver
udført når Event indtræffer

For at fjerne eventlistener fra element igen:

```
element.removeEventListener("click", doSomething); // holder op med at virke
```

ContentLoaded - eventlistener og -handler

Javascript bør først udføres, når man er helt sikker på, at DOM'en er loadet



JavaScript HTML DOM EventListener, W3Schools: https://www.w3schools.com/js/js_htmldom_eventlistener.asp
DOMContentLoaded, MDN Webdocs: <https://developer.mozilla.org/en-US/docs/Web/Events/DOMContentLoaded>

Øvelse 5 - test om DOM'en er loadet

- Gem **04-append.html** i en ny kopi med navnet **05-testLoad.html**.
- Arbejd videre på indholdet:
Du skal sørge for at scriptet tester, om DOM'en er loadet, inden der bliver ændret på indholdet.

Øvelse 6 - click event

1. Gem **05-testLoad.html** i en ny kopi med navnet **06-skiftBillede.html**.
2. Arbejd videre på indholdet:
Når der klikkes på det første billede, skal billedet skiftes ud.
3. Commit og push til GitHub

TIP!

Ved at tilføje et tilfældigt tal til billedurlen fra placeimg.com får man et nyt tilfældigt billede. Ex.:

```
img1.src = "https://placeimg.com/400/300/arch?t="+tilfældigtTal;
```

Brug jeres viden fra igår til at genere et tilfældigt tal mellem 1 og 10.

Flere events

```
let info = document.querySelector("#info");
```

```
info.addEventListener("click", doSomething);
```

```
info.addEventListener("dblclick", doSomething);
```

```
info.addEventListener("mouseover", doSomething);
```

```
info.addEventListener("mouseout", doSomething);
```

```
function doSomething() {  
    // det der skal ske  
}
```

<https://developer.mozilla.org/en-US/docs/Web/Events>

https://www.w3schools.com/jsref/dom_obj_event.asp

Endnu flere:

mousedown

mousemove

mouseup

touchdown

touchmove

touchend

animationend

transitionend

keydown

keypress

keyup

...

3 måder at kalde funktioner på i eventListeners

```
let element = document.querySelector("#info");
```

1. Almindelig funktionskald:

```
element.addEventListener("click", doSomething);  
function doSomething() {  
    // det der skal ske  
}
```

2. Anonym funktion:

```
element.addEventListener("click", function() {  
    // det der skal ske  
});
```

3. Arrow-funktion:

```
element.addEventListener("click", () =>{  
    // det der skal ske  
});
```

Opsamling

Event

EventListener

addEventListener

EventHandler

Anonym funktion

Arrow-funktion

JS og css

Tilføj eller fjern klasser fra element

```
const info = document.querySelector("#info");
```

```
info.classList.add("classNavn") // indsætter eller tilføjer class
```

```
info.classList.remove("classNavn") // fjerner class
```

```
info.classList.toggle("classNavn") // skiftevis fjerner og tilføjer class
```

Direkte manipulation af elementet

```
const info = document.querySelector("#info");
```

```
info.style.border = "none";  
info.style.backgroundColor = "paleyellow";  
info.style.fontFamily = "Helvetica, Sans-serif";  
info.style.height = "2rem";  
info.style.display = "none";  
info.style.display = "block";
```

JS Syntaks for style-properties:

| | |
|------------------------------------|-------------------------------------------------------------------------|
| Bindestreg erstattes af camelCase: | fontFamily (css: font-family) , backgroundColor (css: background-color) |
| Colon erstattes med lighedstegn: | backgroundColor = "paleyellow" (css: background-color : paleyellow;) |
| Værdier sættes i gåseøjne: | style.height = "2rem"; (css: height : 2rem;) |

Øvelse 7 - styling med javascript

Gem **06-skiftBillede.html** i en ny kopi med navnet **07-boxShadow.html**, og arbejd videre på programmet:

De tre articles skal pålægges en klasse der indeholder en box-shadow (lav en flot en [her](#))
Lav herefter to eventhandlers ([mouseover](#), [mouseout](#)) der toggler shadow classen således at når man ruller over en article, er box-shadow væk, og når man ruller ud er den på igen (oldschool 🤖)

Ekstra:

Tilføj en [transition](#), så skyggen kommer gradvist på.

Prøv med andre properties.

Commit og push til gitHub, når du er tilfreds med opgaven

Opsamling, CSS properties der skal bruges øvelse 7

classList-property

classList.add()

classList.remove()

classList.toggle()

style-property

box-shadow-property

Arrays og loops

Ny datatype: **Array**

Datatyper: tekster, tal, boolean og nu også **array** - kendes på []

Et **array** er en liste/tabel med flere elementer, som regel af samme type eller samme kategori som f.eks. "titler", "navne", "kæledyr", "postnumre" og lign.

Variabler med forskellige datatyper:

```
const minTekst = "en kort eller lang sætning"; // string " "  
const pris = 200; // number  
const lærerTeam = ["Alan", "Martin", "Klaus", "Kamilla"]; // array  
console.log(lærerTeam[0]); // returnerer "Alan"  
console.log(lærerTeam[3]); // returnerer "Kamilla"  
lærerTeam.length // giver antallet af elementer i array'et lærerTeam = 4
```

Tomt array:

```
let lærerTeam=[];
```

OBS!

Første
index er 0.

Læs mere om Array:

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

Loop: forEach

En “loop” kalder man den mekanisme hvor man gentager de samme kodelinjer for flere forskellige variabler - f.eks et arrays elementer.

forEach er én af de loop-syntaxer javascript byder på.

```
const lærerTeam = ["Alan", "Martin", "Klaus", "Kamilla"];  
lærerTeam.forEach(lærer => console.log(lærer));
```

lærer er et variabelnavn, som vi selv finder på,
der repræsenterer hver enkelt værdi i arrayet.

HTML DOM querySelectorAll() Method, W3schools: https://www.w3schools.com/jsref/met_document_queryselectorall.asp

JavaScript Array forEach() Method, W3schools: https://www.w3schools.com/jsref/jsref_foreach.asp

Øvelse 9 - array

Lav en ny html-fil (eller gem en gammel) og kald den **array.html**

Lav et html-skellet med et liste element i <body> med id *display*
`<ul id="display">`

Lav herunder et javascript:

Opret et **array** "playliste" med navnene på dine 10 yndlingstracks.

Loop arrayet igennem og få hvert tracknavn til at stå på hver sin linie på listen.

Commit og push til gitHub, når du er tilfreds med opgaven.

Sound of Music
Dancing in the Rain
What a wonderful world
Lille Peter Edderkop
Yesterday
Den knaldrøde gummibåd
Krøller eller hvad
Ved landsbyens gadekær
Smilende Sussi
Så længe jeg lever

Vælg og loop igennem
flere DOM-elementer

querySelectorAll


Eksempel: Alle elementer på siden med class="red", skal forsvinde:

```
document.querySelector(".red").style.display="none";
```

Problem: kun det første element forsvinder!

For at få fat i dem alle:

```
const alle = document.querySelectorAll(".red");
```



Variablen **alle** bliver en
liste (**array**) som vi kan
loop'e igennem med
forEach

For at "gøre noget" ved alle elementer, i eksemplet: få dem til at forsvinde:

```
alle.forEach (element => element.style.display = "none");
```

querySelectorAll + event + this = interaksjon

Det bliver endnu mere interessant, når vi kombinerer tingene.

F.eks. kan vi tilføje en eventListener til alle elementer i et array, så de bliver interaktive:

```
<body>
  <div class="boks"></div>
  <div class="boks"></div>
  <div class="boks"></div>

  <script>
    const alleBokse = document.querySelectorAll(".boks");
    alleBokse.forEach(boks=>boks.addEventListener("click", aktiver));

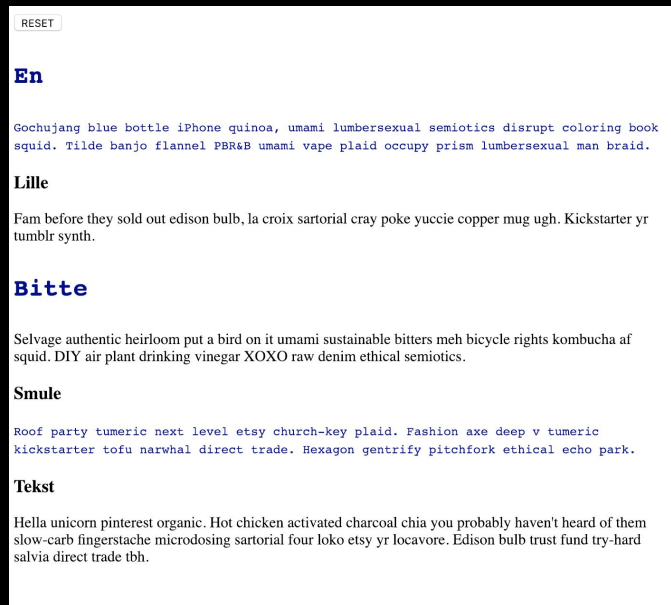
    function aktiver(){
      this.classList.toggle("aktiv");
    }
  </script>
</body>
```

```
<style>
  .boks{
    width: 10vw;
    height: 10vw;
    margin: 1em;
    background-color: orange;
  }

  .aktiv{
    background-color: blue;
  }
</style>
```

Øvelse 8: Eventlisteners på alle p-tags

1. Tag udgangspunkt i denne [klargjorte html-fil](#), og gem den som **08-alle.html**
2. I scriptet - lav en eventlistener der tjekker om DOM'en er indlæst. Sæt herefter en eventlistener på hvert p-tag: når der klikkes på en sætning, skal den styles med en klasse ex: "typo".
3. Lav en reset-knap øverst på siden.
4. Når der klikkes på knappen, skal class "typo" fjernes fra alle p-tags igen
5. Udvid programmet med en klasse, **typo2**, som styles alle overskrifter på samme måde - men med dobbelt størrelse.



Opsamling

querySelectorAll()

forEach

Array

Array-index

Eftermiddagsøvelser

Øvelse: slideshow 1

I denne øvelse skal du både bruge viden fra igår og idag: QuerySelectors, Arrays, EventListeners, tal-operatore.

I denne øvelse skal du lave et lille billedgalleri, hvor billedet skifter til det næste når der klikkes på knappen.

1. Opret et html dokument i stil med billedet til højre og gem det som **slideshow1.html**
2. Opsæt billederne til slideshowet i et array.
(billederne finder du i [HER.](#))
3. Når der klikkes på “Videre” knappen, skal billedet udskiftes med det næste i arrayet.

TIP! For at kunne få det næste element i arrayet, kan man bruge en “tæller” ex.: tal++
Husk at tekststrengene kan sammensættes

Tro på det...



Billeder brugt med tilladelse af Rasmus Bregnhøj

Videre

Øvelse: slideshow 2

Udfordring!!!

Når det sidste billede vises skal teksten i knappen skifte til “Forfra” og slideshowet starte forfra ved næste klik.

1. Gem filen fra slidehow 1, som [slideshow2.html](#)
2. Lav de nødvendige ændringer i scriptet.

TIP!

Brug betingelsessætninger:

```
if(...){  
...  
}else{...}
```

Tro på det...



Billeder brugt med tilladelse af Rasmus Bregnhøj

Forfra

Ekstra Øvelse (optional) : styleBoxes.html

Du skal bruge 4 div'er med class "box" og med hver sin id (box1, box2, box3, box4)

[Klargjort version](#)

Giv box-class lidt css, så boxene ser ens ud.

Når man klikker på dem hver især skal de skifte udseende.

- gerne med en transition --

Så alle 4 ser forskellige ud.

Du kan gøre det ved at tilføje hver sin class, på 2 af boksene.

På de resterende 2, prøv at style direkte med js, Prøv at bruge andre properties end dem i eksemplerne .

Næste trin - til denne sensommereftermiddag

Øvelser 1,2,4 på Github <https://github.com/kvikea/JS-2-exercises>

Efter idag, vil de sikkert forekomme dig temmelig nemme !!

(3'eren springer vi over).