

Fra i går...

Hvordan gik det med eftermiddags opgaverne?

Javascript og DOM

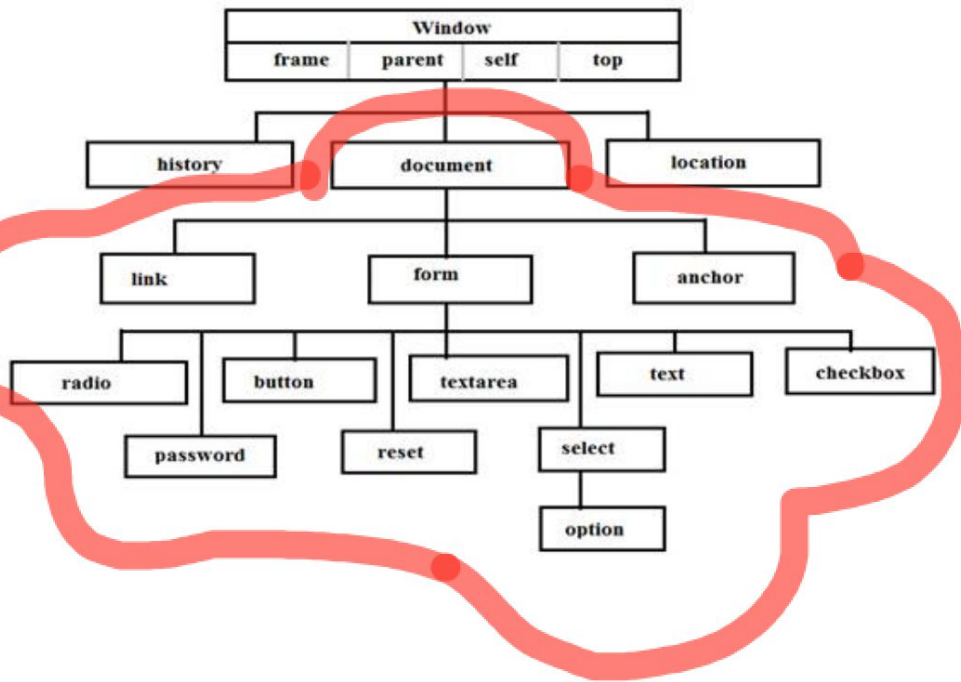
Dagens program

- DOM (Document Object Model)
- 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

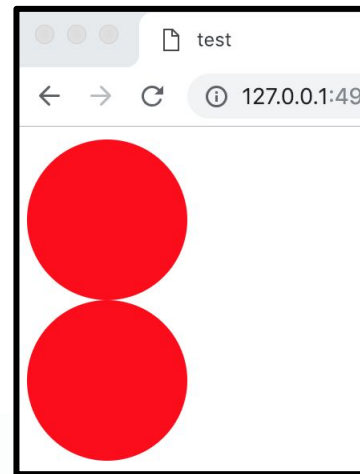
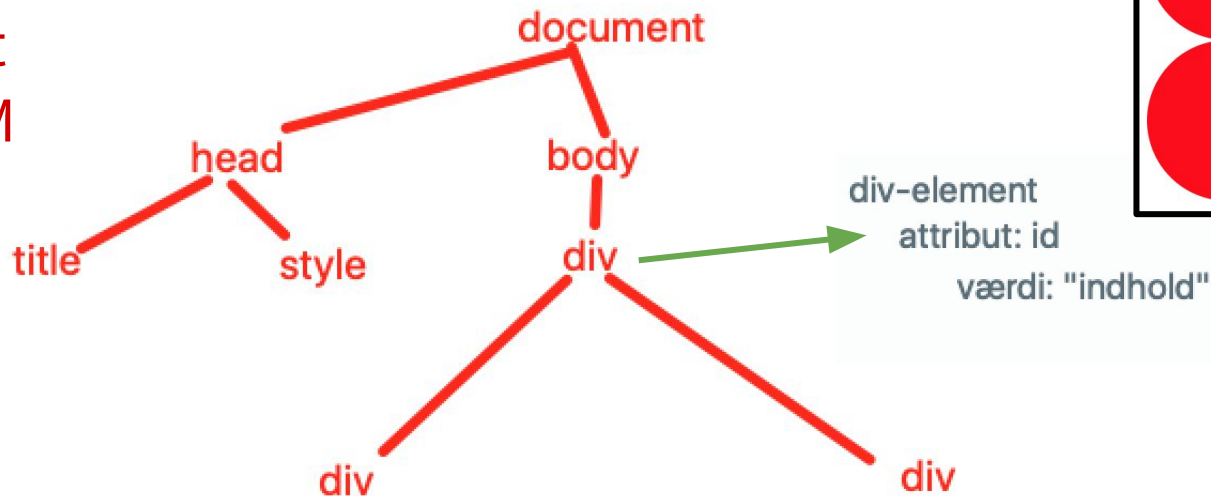


DOM

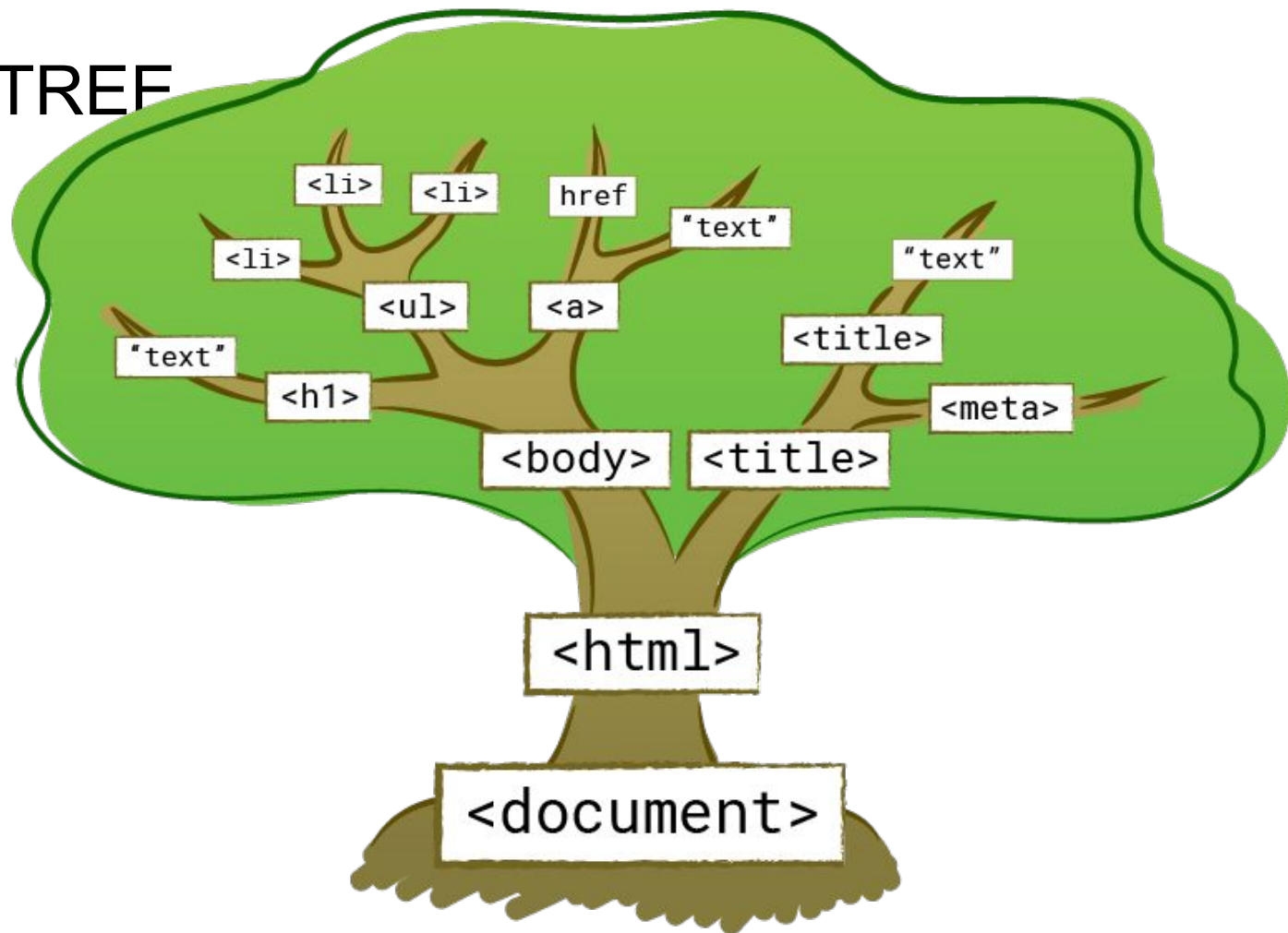
(Document Object Model)

```
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 TREE



HTML5 - Semantiske tags

header, nav, main, section, article, footer, details, summery 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

<http://html5doctor.com/lets-talk-about-semantics/>


Vælg DOM-element

document.querySelector - (vælg et DOM-element)

`document.querySelector("#idNavn")`

`document.querySelector(".classNavn")`

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



Alle CSS-selectorer
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!

Gem querySelector i variabel

En **selector** gemmes tit i en **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**.

DISCLAIMER FOR VIDEOS !

Enhver reference til:
henviser fra d.d. til:

tema 5

tema 7

```
let elm = document.querySelector(".element");  
const elm = document.querySelector(".element");
```



Øvelse 01 - Udvælg DOM-elementer

1. I mappen **undervisningsopgaver**, laver du en ny undermappe, **02-js-DOM**. Åbn den i Brackets
2. Opret en ny html-fil, **01-select.html**
(tip gem et tomt doc som html, og brug EMMET: **!** - **tab** til at lave et nyt HTML skelet)
3. I body skal du lægge et div-tag med id="content"
4. Inde i #content lægger du tre div'er uden class eller id
5. I den første lægger du et link til Keas hjemmeside, i den anden et link til din github-side og i den tredje et billede af en frø (url: <http://mabe-kea.dk/E19/pics/frog.png>)
6. Lav querySelectors på hver af elementerne: #content, hver af de tre div's i #content og linket i den første, linket i den anden og billedet i den tredje, og vis dem med console.log()

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

Opsamling

DOM

querySelector

attribut

Læg nyt indhold i
DOM-element

Nyt indhold i element (textContent og innerHTML)

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

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

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

Man kan ændre på indholdet af containertags med js:

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

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

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

W3schools: HTML DOM textContent Property, : https://www.w3schools.com/jsref/prop_node_textcontent.asp

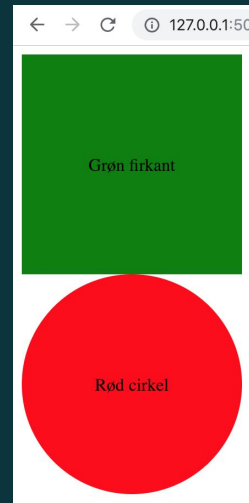
W3schools: HTML DOM innerHTML Property, : https://www.w3schools.com/jsref/prop_html_innerhtml.asp

Øvelse 2 - Læg tekstindhold i elementer

1. Opret en ny html-fil, **02-indhold.html**, i mappen **02-js-DOM**
2. I body skal der være en grøn firkant og en rød cirkel - brug et style i head i stedet for et eksternt stylesheet - du må gerne. De to figurer skal være tomme (ingen tekst endnu)
3. Læg teksten “Grøn firkant” i firkanten og “Rød cirkel” i cirklen, med et javascript

(se næste slide for css-forslag til opgaven).

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



Css-tips til øvelse 02

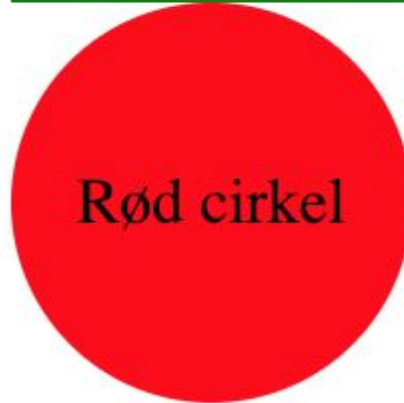
1. Eksempel css-egenskaber til opgave 02:

```
height: 100px; width: 100px
```

```
line-height: 100px (vertical midt på)
```

```
border-radius: 50%; (rund kant)
```

```
text-align: center; (horisontalt midt på)
```



Øvelse 3 - Læg html-indhold i elementer

Gem **02-indhold.html** i en ny kopi, **03-billedIndhold.html**.

Læg nyt javascript i script-tagget, som:

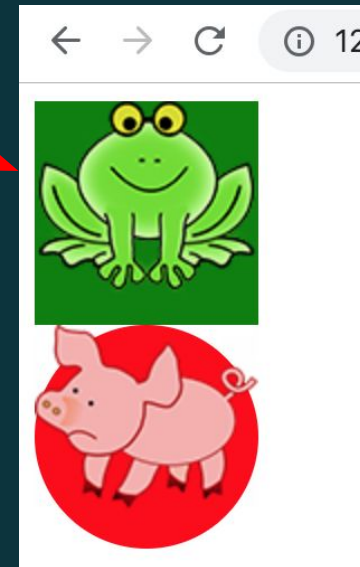
1. Sætter en frø ind i firkanten
2. En gris ind i cirklen.

Her er url'er til billederne:

<http://mabe-kea.dk/E19/pics/frog.png> og

<http://mabe-kea.dk/E19/pics/pig.png>

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



Nyt indhold med createElement og appendChild

```
const info = document.querySelector("#info");  
info.innerHTML = "<h1>Min overskrift til info</h1><img src='http://helf-kea.dk/test/gris.png'>";
```

Alternativ:

```
const h1 = document.createElement("h1");  
const overskrift = document.createTextNode("Min overskrift til info");  
h1.appendChild(overskrift);  
const img = document.createElement("img");  
img.src="http://helf-kea.dk/test/gris.png";  
info.appendChild(h1);  
info.appendChild(img);
```



```
<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 små html-bidder (Bør så vidt undgås da det kan hackes!)
- Brug **createElement** og **appendChild** til html-tilføjelser

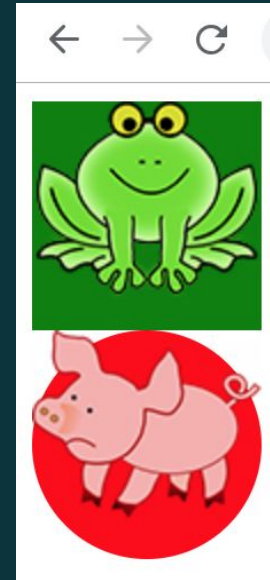
Øvelse 4 - Tilføj nyt element i element

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

Tilføj nyt javascript i script-tagget, som:

1. Opretter et nyt img-element(createElement).
2. Sætter det nye elements billedfil til ["http://mabe-kea.dk/E19/pics/fox.png"](http://mabe-kea.dk/E19/pics/fox.png)
3. Sætter det nye element ind i cirklen. (sæt højde og bredde op til det 200px)

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



DOM-manipulationer - endnu flere

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

Opsamling

textContent

innerHTML

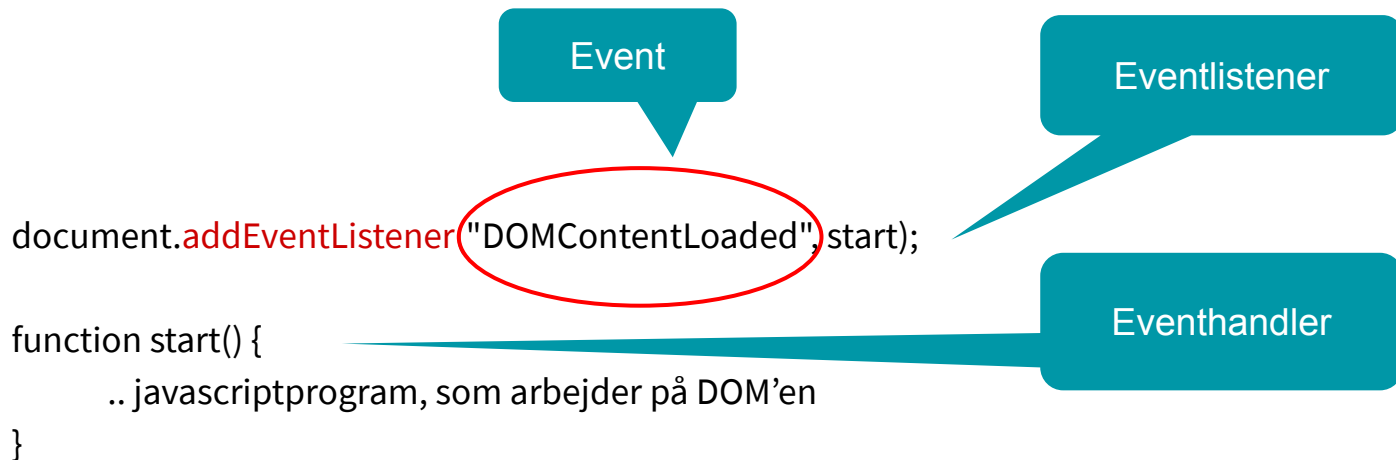
createElement

appendChild

EventListener og eventhandler

ContentLoaded - EventListener & EventHandler

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>

EventListener & EventHandler

```
let element = document.querySelector("#info");  
  
element.addEventListener("click", doSomething);  
  
function doSomething() {  
    // det der skal ske  
}
```

For at fjerne eventlistener fra element igen:

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

Øvelse 5 - test om DOM'en er loadet

- Gem **03-billedIndhold.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 det lægger billederne ind.

Øvelse 6 - frøhop

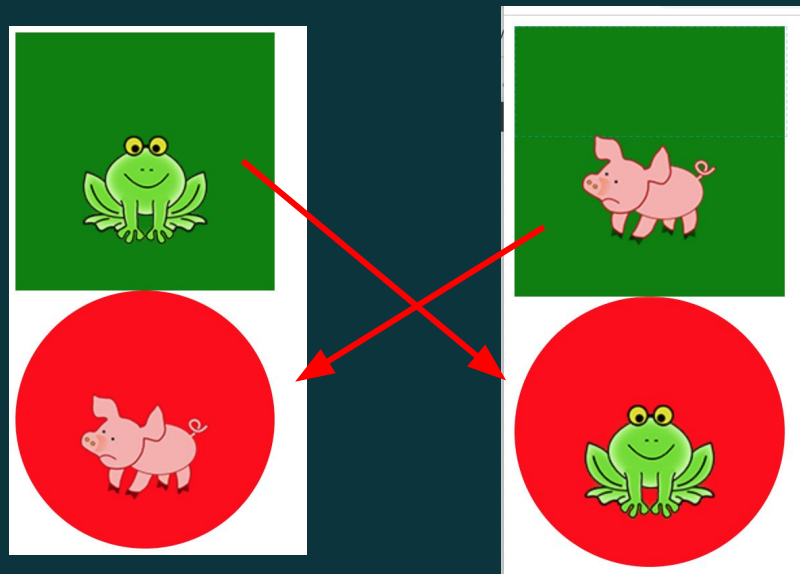
Gem **05-testLoad.html** i en ny kopi med navnet **06-klikHop.html**.

Arbejd videre på indholdet:

Når der **klikkes** på den **grønne firkant**, skal de to dyr bytte plads.

Når der **klikkes** på den **røde cirkel**, skal den **grønne firkant** ikke længere kunne bruges til ombytninger.

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



Flere events

```
const 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 funktion:

```
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

```
let 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

```
let 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 navne på style-properties:

- bindestreg erstattes af camelCase
fontFamily (css: font-family)
backgroundColor (css: background-color)

Øvelse 7 - styling med javascript

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

Når man klikker på felterne skal de skifte baggrundsfarve.

Ekstra:

Tilføj en **transition**, så farveskiftet sker gradvist.

Prøv med andre properties.

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

Opsamling

style-property

backgroundColor-property

classList-property

classList.add()

classList.remove()

Vælg flere DOM-elementer

querySelectorAll

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

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

Problem: kun det første element forsvinder! - For at få dem alle:

```
let all = document.querySelectorAll(".red");
```

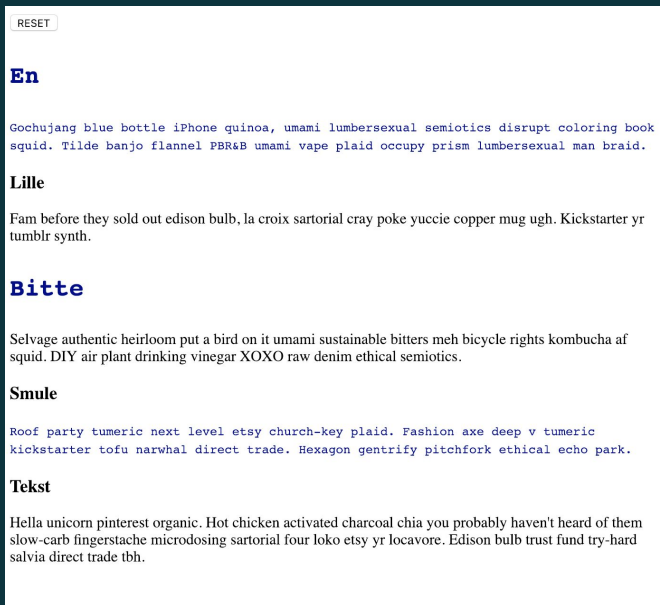
For at “gøre noget ved alle elementer”, i eksemplet: få dem til at skjule sig:

```
all.forEach (element => {  
    element.style.display="none";  
});
```

Variablen **all** bliver en liste(**Array**) som vi kan køre igennem med et **forEach loop**!

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.



Arrays og loops

Ny datatype: Array []

Datatyper: tekster, tal, boolean og nu også **array**

Et **array** er en liste/tabel med flere elementer

Variabler med forskellige datatyper:

```
let minTekst = "en kort eller lang sætning"; // string " "  
let pris = 200; // number  
let 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.

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

lærer: et variabelnavn, du selv finder på, der repræsenterer den enkelte af arrayets værdier

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

Tag udgangspunkt i denne [klargjorte version](#), og kald
filen **array.html**

Lav et script:

Opret et **array** “playliste” med navne
på dine 10 yndlingstracks.

Loop arrayet igennem og få hvert tracknavn
til at stå i div-boxen “display” , på hver sin linie

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

Opsamling

querySelectorAll()

forEach

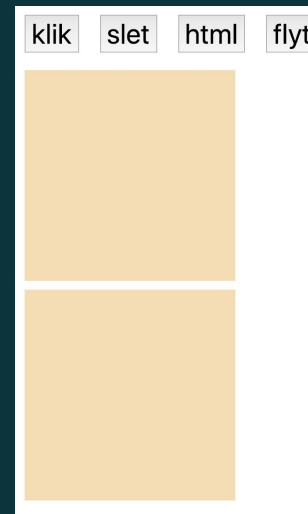
Array

Array-index

Eftermiddagsøvelser

Øvelse 10: klikpaaknap.html

1. Hent denne [klargjorte html-side](#).
2. Der er en knap “klik” og en div med id, “box”
Når man klikker på knappen “klik”, skal “box” vise teksten: “KLIK”.
3. Der er også en knap, slet
Når man klikker på “slet”, skal “box” resettes
(viskes ren) = vise teksten: “ ”.
4. Knappen “html” skal vise en <h2> og en <p> i boksen.
5. Lav så en knap “flyt”, der kan flytte “box”’s html over i den anden boks, “bax” !



! Hvis du hidtil har brugt navngivne funktioner, så prøv med anonyme eller arrow

Øvelse : 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).