

# Javascript og DOM

Martin Bregnhøj  
mabe@kea.dk

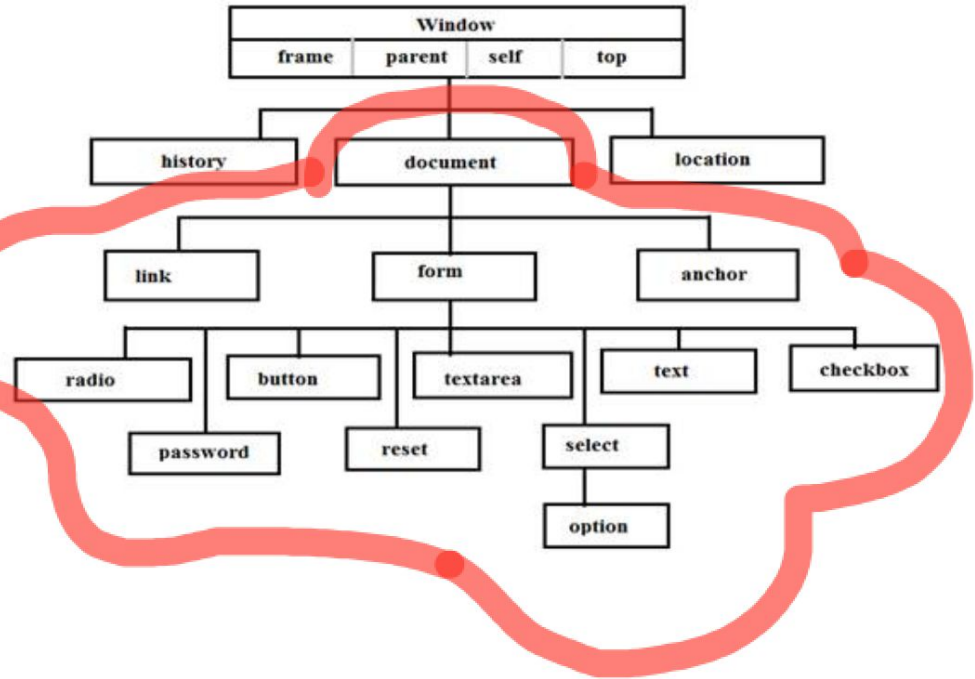
# Agenda

- 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

# DOM

# Window- og document-objektet

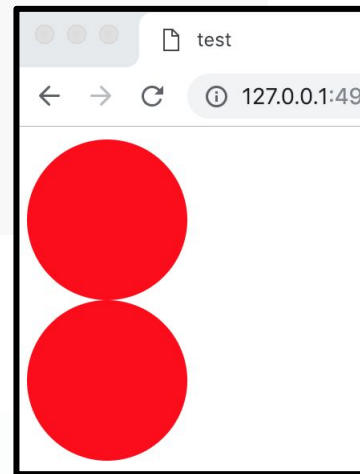
## Browser Object Model



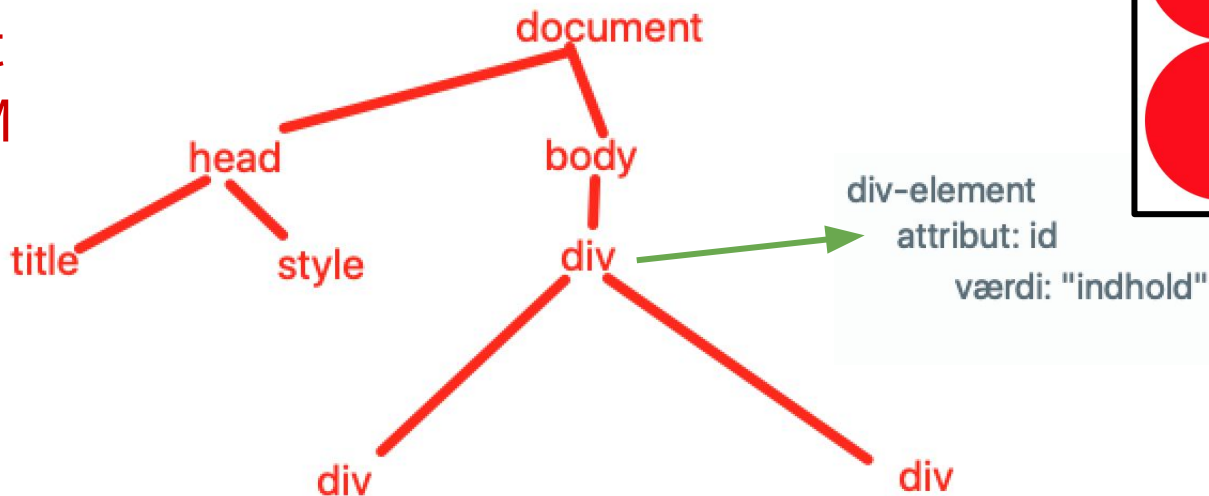
**DOM**

(Document Object Model)

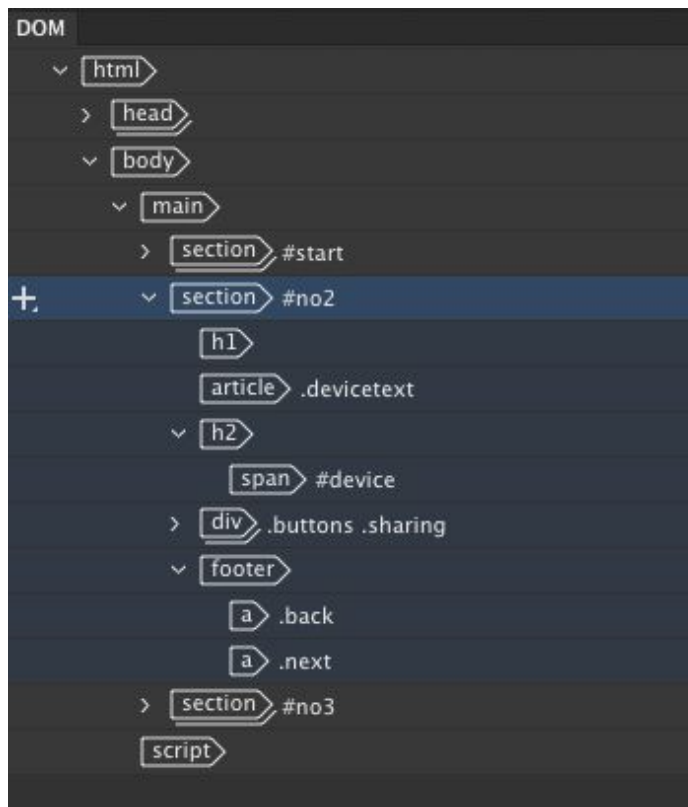
```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>test</title>
5     <style>.kugle{background-color:red; height:100px; width:100px;border-radius:100px}</style>
6   </head>
7   <body>
8     <div id="indhold">
9       <div class="first kugle"></div>
10      <div class="last kugle"></div>
11    </div>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



Document  
ifølge DOM



# DOM træ



# HTML struktur recap

# 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://www.w3schools.com/html/html5_semantic_elements.asp)

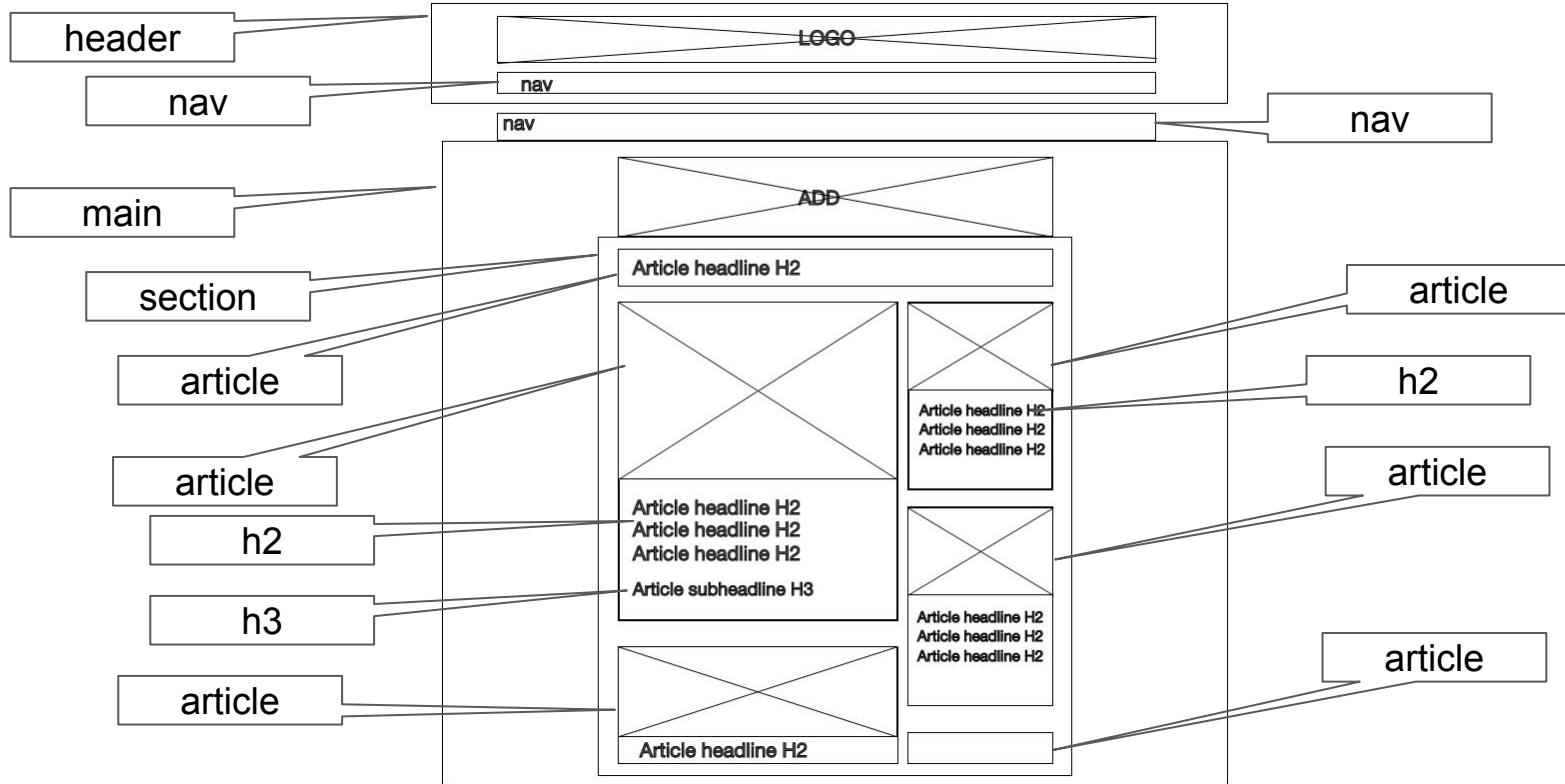
[https://developer.mozilla.org/en-US/docs/Glossary/Semantics#Semantics\\_in\\_HTML](https://developer.mozilla.org/en-US/docs/Glossary/Semantics#Semantics_in_HTML)



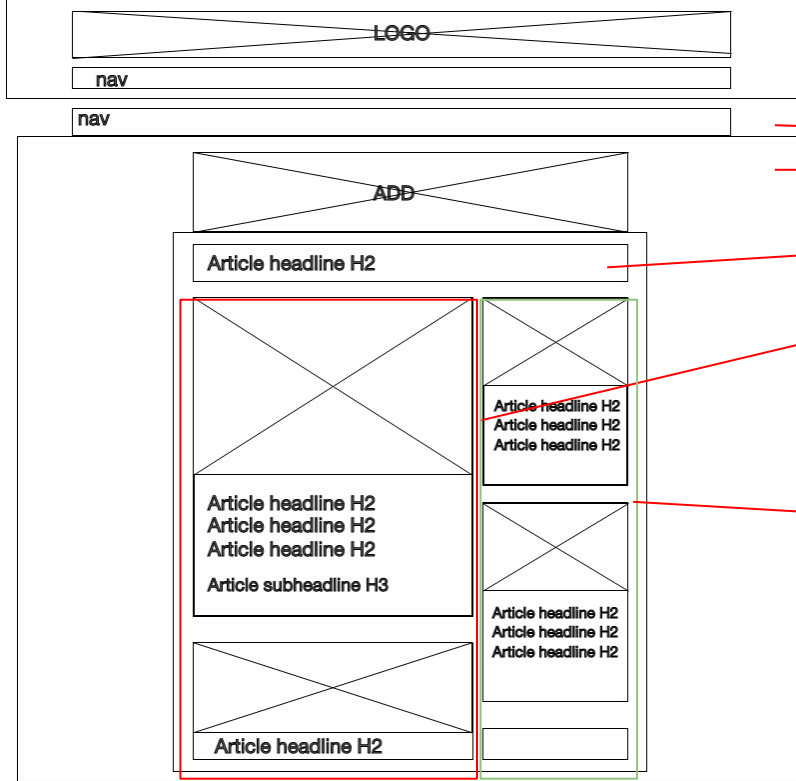
# Semantisk html struktur



# Wireframe (lettere forenklet)



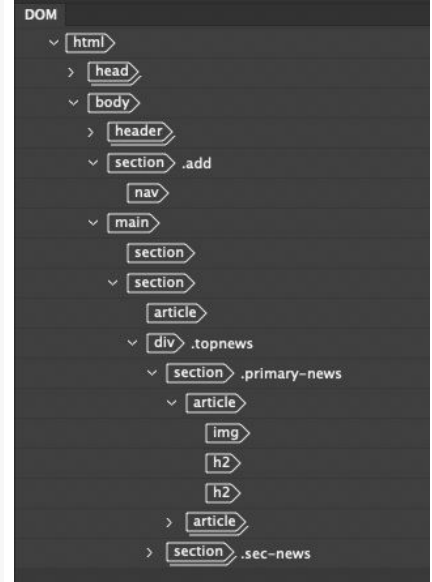
## WIREFRAME



## HTML

```
9 <body>
10 <header>
11   <img src="" alt="">
12   <nav></nav>
13 </header>
14 <section>
15   <nav></nav>
16 </section>
17 <main>
18   <section></section>
19   <section>
20     <article class="toparticle"></article>
21     <div class="topnews">
22       <section class="primary-news">
23         <article>
24           <img src="" alt="">
25           <h2></h2>
26           <h3></h3>
27         </article>
28         <article>
29           <img src="" alt="">
30           <h2></h2>
31           <h3></h3>
32         </article>
33       </section>
34       <section class="sec-news">
35         <article>
36           <img src="" alt="">
37           <h2></h2>
38         </article>
39         <article>
40           <img src="" alt="">
41           <h2></h2>
42         </article>
43       </section>
44     </div>
45   </section>
46 </main>
47 </body>
```

## DOM TREE



# Begreber

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

Semantisk HTML - html elementer der indikerer hvad de indeholder (godt for SEO og struktur)

# 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 VS Code og Github workflow.

Du kan også se denne lille video:

<https://kea.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=c88a7992-292e-439f-a610-ad8900b6b0e4&start=0>

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.

# Tips til øvelse 1

Billeder kan autogenereres på:

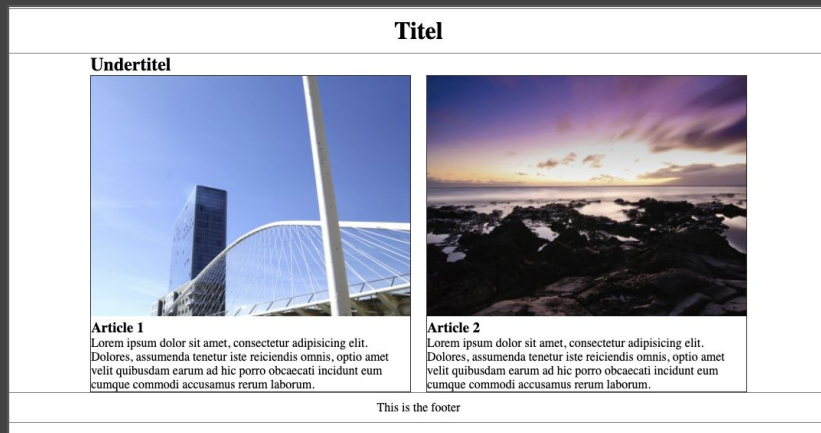
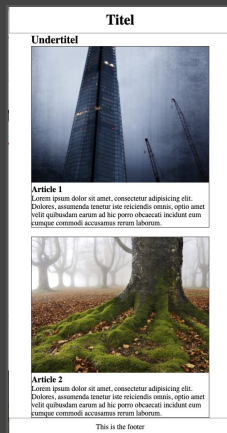
<https://placeimg.com>

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

Ex.: ``

# 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. Billederne vælger du selv.:



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 og indholdet.

Det er ikke nødvendigt at bruge media queries!!!

# 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](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**.

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!

# 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](https://www.w3schools.com/jsref/met_document_createelement.asp)

HTML DOM appendChild() Method, : [https://www.w3schools.com/jsref/met\\_node\\_appendchild.asp](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 **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>

# 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](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
```



# Begreber

querySelector - syntaksen for at vælge et DOM element

CSS selectors - identifikationen af det enkelte DOM element i en querySelector  
Sætning

textContent

innerHTML

createElement > appendChild

# 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](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>

## Øvelse 2B - Udskift tekstindhold i elementer

1. Gem en kopi af filen fra øvelse 2A
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 `<b>` 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

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

# 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

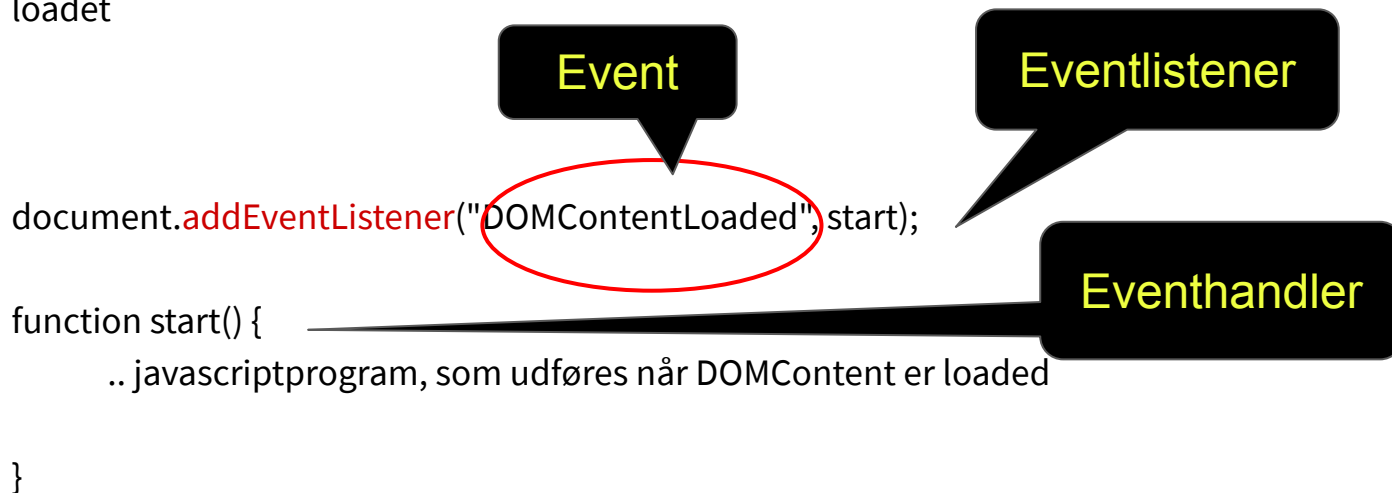
EventHandler  
- Den funktion der bliver  
udført når Event indtræffer

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](https://www.w3schools.com/js/js_htmlDOM_eventlistener.asp)  
DOMContentLoaded, MDN Webdocs: <https://developer.mozilla.org/en-US/docs/Web/Events/DOMContentLoaded>



# 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](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  
});
```

# Begreber

Event

EventListener

EventHandler

Anonym funktion

Arrow-funktion

## 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](https://placeimg.com) får man et nyt tilfældigt billede. Ex.:

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

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

# 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](https://github.com/martinbregnhøj/Tema7E20/tree/master/billeder).  
(<https://github.com/martinbregnhøj/Tema7E20/tree/master/billeder>)
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