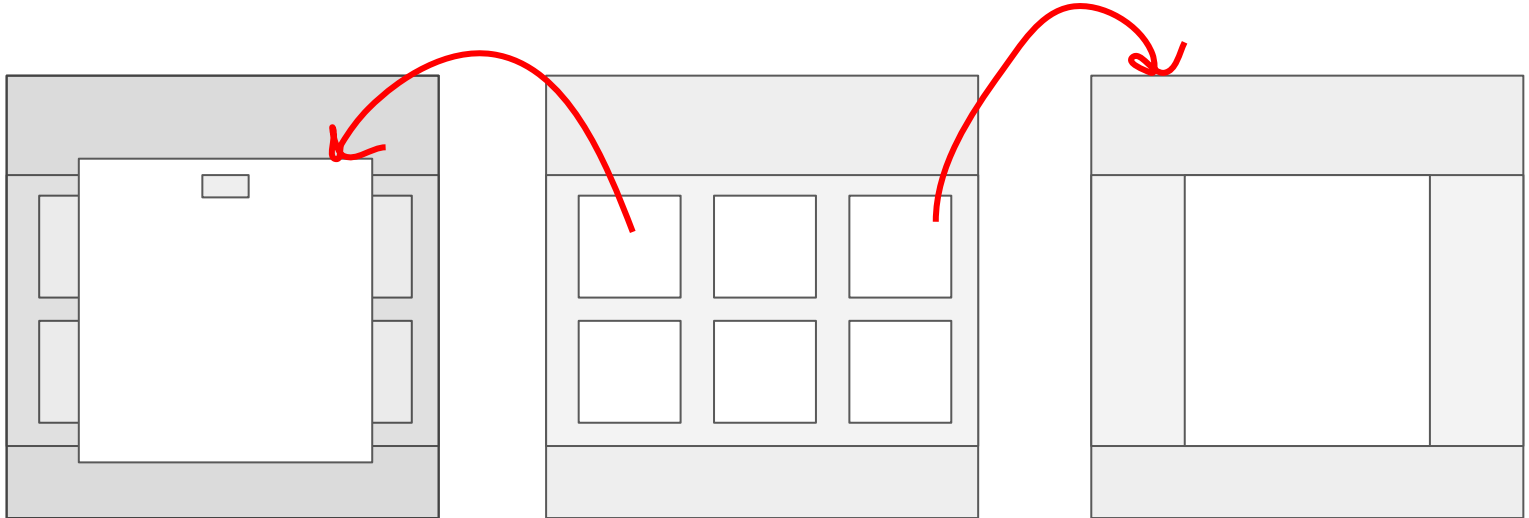
Single view



Fra mandag

Babushka - hvad **skal** der være i menuen:

I opgaven:

> *Der skal være en menu på siden, som giver mulighed for, at man får særskilte oversigter over forretter, hovedretter og desserter*

Så andre rettyper **må** I gerne springe over!
- men det er kun fint at tage dem med.

Martins side er kun et forslag

I behøver ikke at lave en topgrafik! - men det er fint at gøre det.

Mere fra mandag

Startfilen fra igår

Vi ser på den sammen - idag er det vores grundfil.

Agenda

1. Loop view og single view?
2. Single view i popup eller på egen side?
3. Single view i popup - hvordan?
4. Single view på egen side - hvordan?

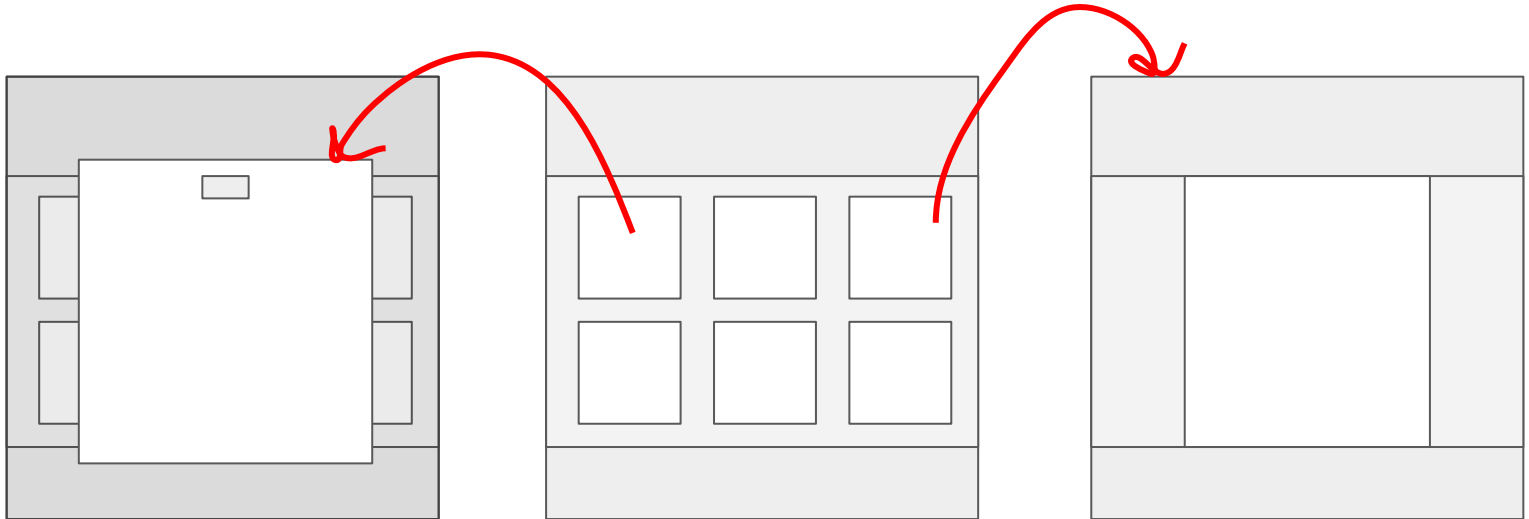
List-view kontra Detail-view

- ★ Loop-view eller list-view er en oversigts-visning af mange objekter (en liste)
- ★ Single-view eller detail-view er en detaljeret visning af ét objekt

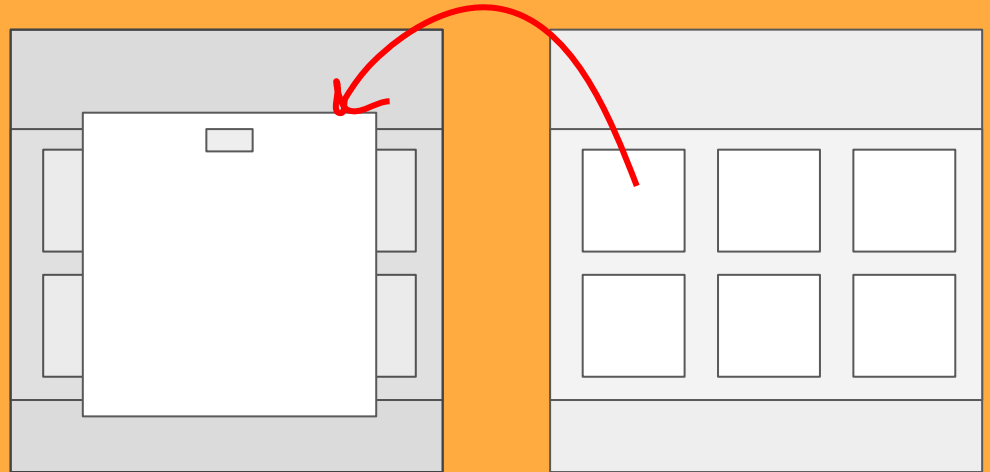
list
loop

detail
single

2 måder: detail-view i popup eller på ny side



detail-view i pop-up



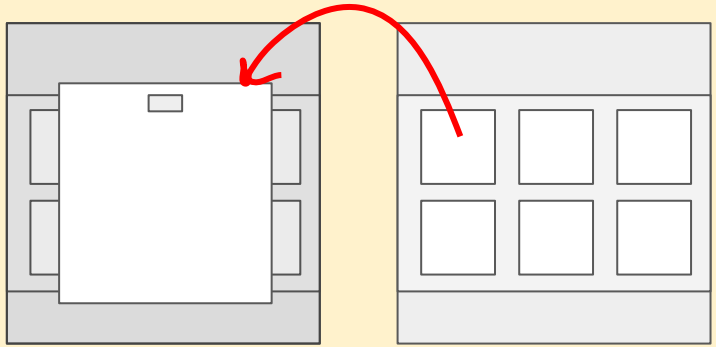
vis side med popup!

Fordele / ulemper

Fordele:

Giver en god brugeroplevelse, fordi der ikke skiftes side

Kan styles meget elegant med lightbox-effekter



Ulemper:

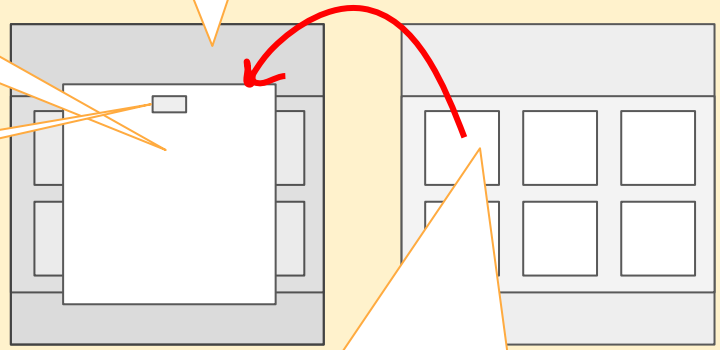
Single-viewet har ikke sin egen url-adresse

- kan ikke deles
- kan ikke findes af søgemaskiner

Popup-strategi

1. Lav et popupvindue, som lægger sig ovenpå det hele.

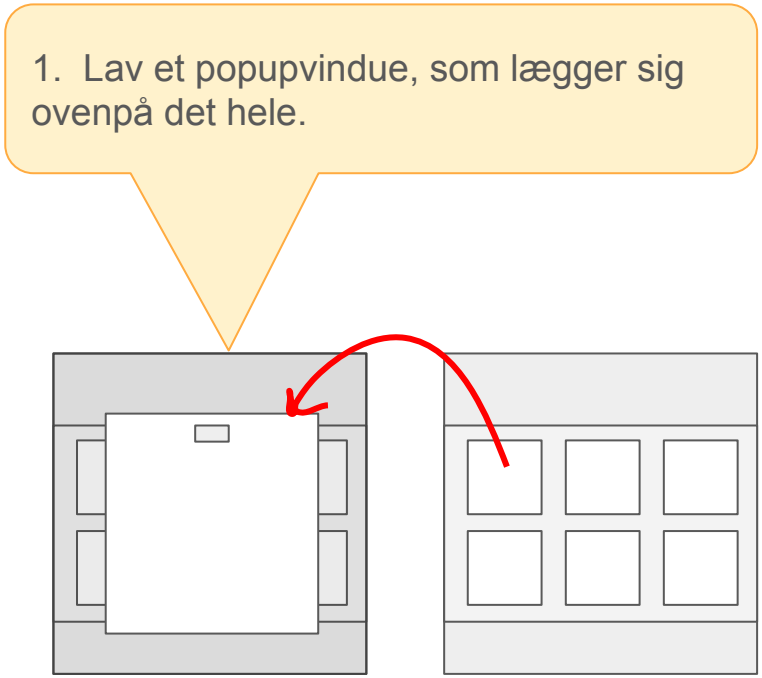
2. læg indholdselementer i popup-en



4. Luk-knap:
Når den klikkes, skal popup'en lukke

3. eventhandlers på alle elementer i loop-view: ved klik skal popup åbne og udfyldes

1. Lav popup



Man kan fx. lægge popup'en øverst eller nederst i dokumentet

Det skal styles, så det fylder det hele (vw og vh: 100)

Det skal styles, så det lægger sig ovenpå det eksisterende indhold (position: fixed)

Giv den en baggrundsfarve, så den kan ses, mens vi udvikler.

```
    <style>
        img { ··· }

        #liste { ··· }

        #detalje {
            position: fixed;
            top: 0;
            left: 0;
            width: 100vw;
            height: 100vh;
            background: hsla(282, 22%, 31%, 0.53);
        }

        .person { ··· }

        .valgt { ··· }

    </style>
</head>

<body>
    <h1>MMD 2019</h1>
    <nav> ··· </nav>

    <section id="liste"> </section>

    <section id="detalje"> </section>

    <template> ··· </template>

    <script> ··· </script>
</body>

</html>
```

Øvelse 1 - lav en popup

Opret en ny mappe, **single**, i **tema7**-mappen.

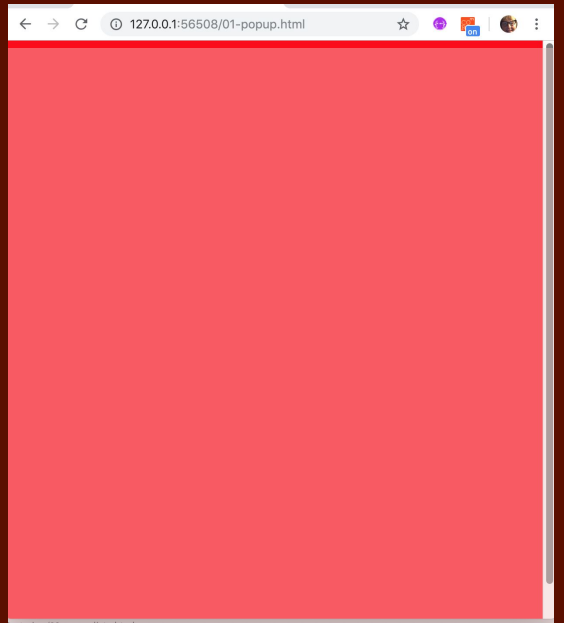
Gem en kopi af din **startfil fra igår** i mappen under navnet **01-popup.html** heri.

Læg et popup-vindue (et section-tag) øverst (eller nederst) i dokumentet.

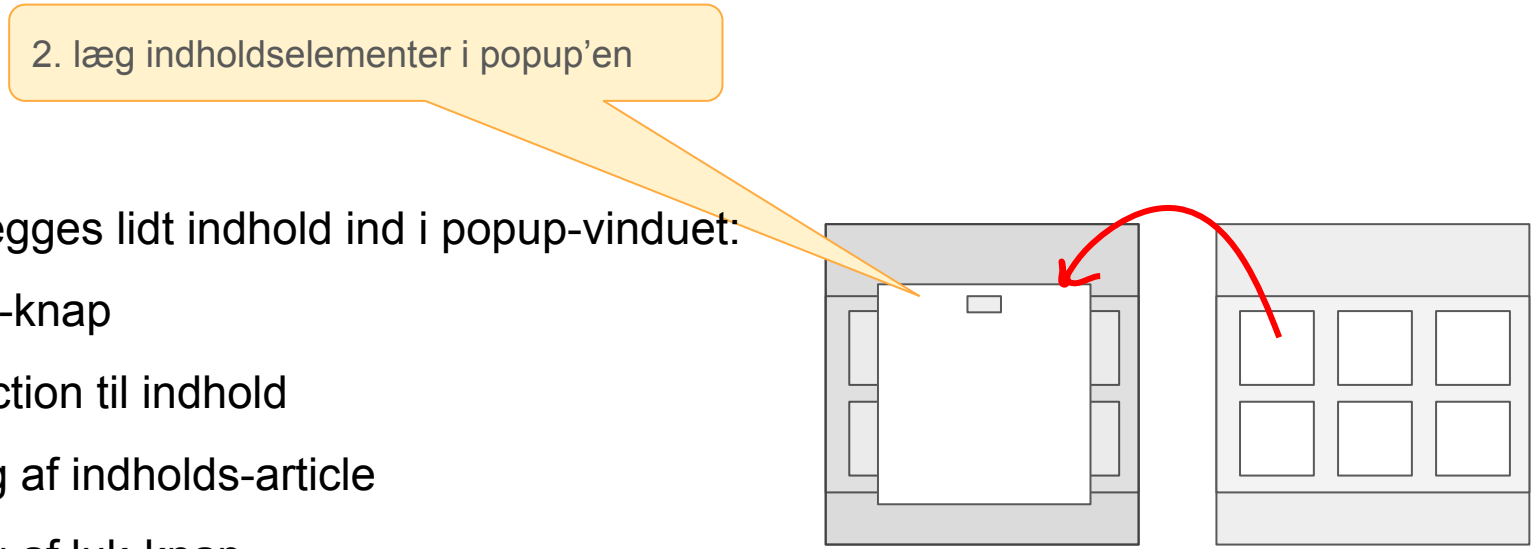
Style det, så det fylder hele vinduet og lægger sig ovenpå det eksisterende indhold.

Giv også popup-vinduet en baggrundsfarve, så det er til at få øje på.

Test om det fungerer.



2. Indholdselementer i popup



Der skal lægges lidt indhold ind i popup-vinduet:

- En luk-knap
- En section til indhold
- Styling af indholds-article
- Styling af luk-knap

```html
<section id="liste"> </section>

<section id="detalje">
    <article class="person">
        <button class="luk">X</button>
        <h2></h2>
        <img src="" alt="">
        <p>Github:
            <a class="githubLink" href=""></a>
        </p>
    </article>
</section>

<template>
    <article class="person">
        <h2 class="navn"></h2>
        <img src="" alt="" class="profil-billede">
    </article>
</template>
```

```css
#detalje {
    position: fixed;
    top: 0;
    left: 0;
    width: 100vw;
    height: 100vh;
    background: hsla(282, 14%, 12%, 0.74);
}

#detalje .luk {
    position: fixed;
    top: 0;
    right: 0;
    font-size: 4rem;
}

#detalje .person {
    width: 60vw;
    min-height: 50vh;
    margin: 4rem auto;
    padding: 4rem;
    background: white;
}
```

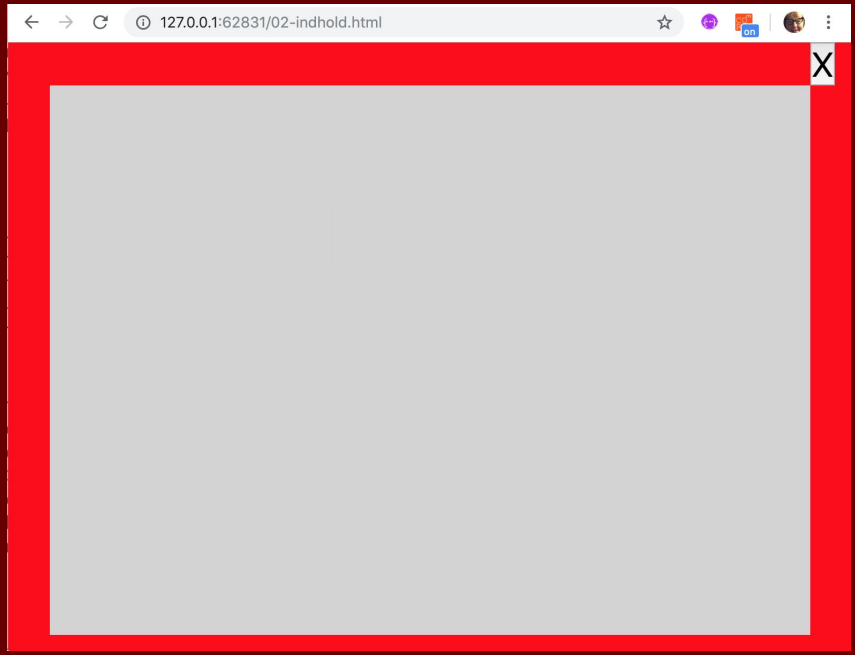
Øvelse 2

Arbejd videre med **01-popup.html** i mappen under navnet **02-indhold.html**.

Læg div'er i popup'en til indhold og lukkeknap

Style, så vinduet bliver funktionelt

Style til sidst popup'en, så den er skjult (kan også gøres med js..)

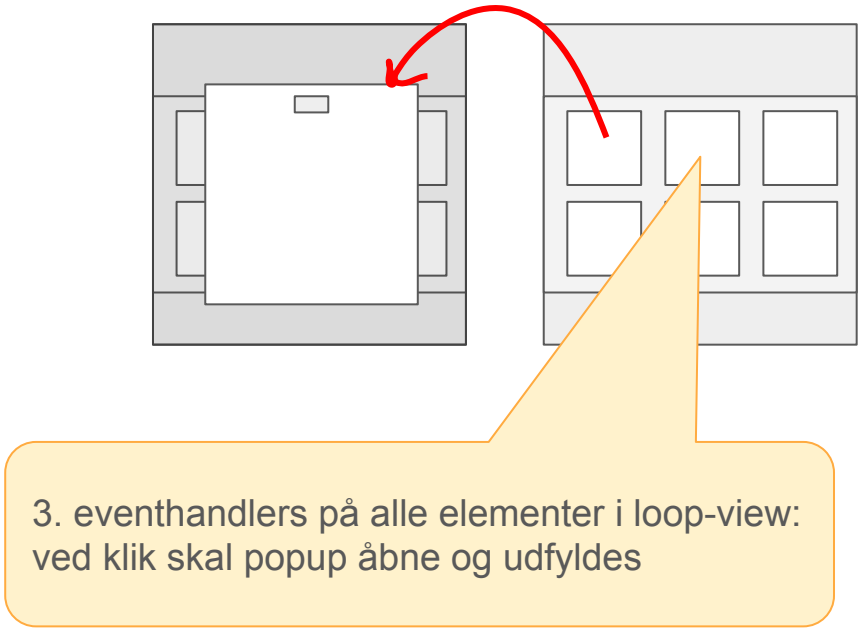


3. Eventhandlers på alle person-elementer i list-view

Ved klik på en person, åbnes og udfyldes popup

Derfor:

- **eventlisteners** på alle elementer
- **eventhandler-funktionen** skal udfylde popup-vinduet med detaljer om den valgte person.
- **Styling** af popup-vinduet indhold



```
//funktion der viser personer i liste view
function vis() {
    antal = personer.feed.entry.length;
    const dest = document.querySelector("#liste"); // container til articles med en person
    const skabelon = document.querySelector("template").content; // select indhold af html skabelon
    dest.textContent = ""; // ryd container inden ny loop
    personer.feed.entry.forEach(person => { // loop igennem json (personer)
        // tjek hvilket køn personen har og sammenlign med aktuelt filter eller vis alle, hvis
        filter har værdien "alle"
        if (person.gsx$sex.$t == filter || filter == "alle") {
            const klon = skabelon.cloneNode(true);
            klon.querySelector(".navn").textContent = person.gsx$id.$t + " " + person.gsx$navn.$t;
            klon.querySelector(".profil-billede").src = person.gsx$billede.$t;
            klon.querySelector(".person").addEventListener("click", () => visDetaljer(person));
            dest.appendChild(klon);
        }
    })
}
```

I hver iteration tilføjes det nye element (article) en click-eventlistener. Når foreach er færdig, er alle artikler blevet klikbare.

I en anonym function kan vi kalde visDetaljer OG medsende en parameter - der indeholder al den relevante data til at vise i html

```
function visDetalje(person) {···}
```

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

```
//funktion der viser personer i liste view
function vis() {
    antal = personer.feed.entry.length;
    const dest = document.querySelector("#liste"); // container til articles med en person
    const skabelon = document.querySelector("template").content; // select indhold af html skabelon
    dest.textContent = ""; // ryd container inden ny loop
    personer.feed.entry.forEach(person => { // loop igennem json (personer)
        // tjek hvilket køn personen har og sammenlign med aktuelt filter eller vis alle, hvis
        filter har værdien "alle"
        if (person.gsx$sex.$t == filter || filter == "alle") {
            const klon = skabelon.cloneNode(true);
            klon.querySelector(".navn").textContent = person.gsx$id.$t + " " + person.gsx$navn.$t;
            klon.querySelector(".profil-billede").src = person.gsx$billede.$t;
            klon.querySelector(".person").addEventListener("click", () => visDetaljer(person));
            dest.appendChild(klon);
        }
    })
}
```

```
// func. der viser person i detalje view (medsend person!)
function visDetalje(person) {
    document.querySelector("#detalje").style.display = "block";

    document.querySelector("#detalje h2").textContent = person.gsx$navn.$t;
    document.querySelector("#detalje img").src = person.gsx$billede.$t;
    document.querySelector("#detalje img").alt = `Portræt af ${person.gsx$billede.$t}`;
    document.querySelector("#detalje .githubLink").href = `https://github.com/${person.gsx$github.$t}`;
    document.querySelector("#detalje .githubLink").textContent = `github.com/${person.gsx$github.$t}`;
}
```

En detaljevisning vil oftest have et andet og mere (html)indhold end på en liste

Øvelse 3

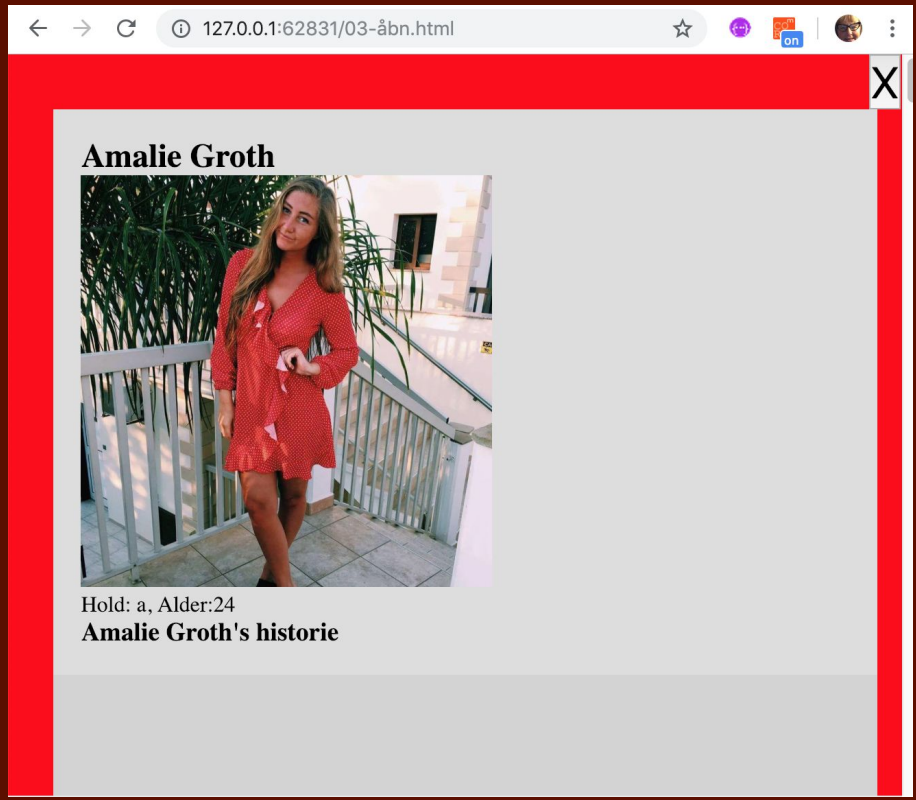
Gem en kopi af din **02-indhold.htm** i mappen under navnet **03-åbn.html**.

Når man klikker på en person i loop-view, skal popup med single-view åbnes.

Vis artiklerne er klikbare med css eller js: cursor: pointer

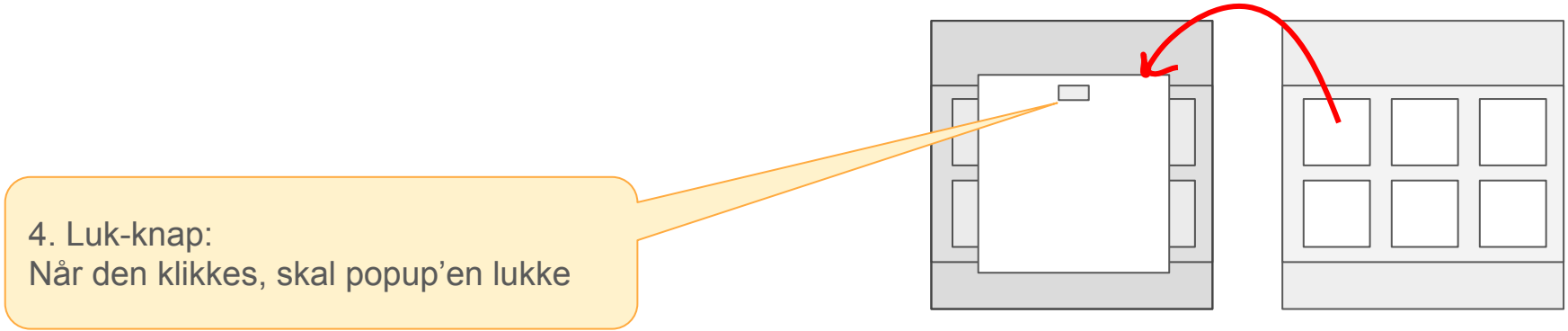
Hvis din #popup ikke kan scrolle:

overflow: scroll;



4. Luk-knappen

Læg en lukke-knap ind i popup-vinduet.
Lav en eventListener til knappen: når den klikkes, skal popup-vinduet lukke.



```
// func. der viser person i detalje view (medsend person!)
function visDetalje(person) {
    document.querySelector("#detalje").style.display = "block";

    document.querySelector("#detalje .luk").addEventListener("click", skjulDetalje);

    document.querySelector("#detalje h2").textContent = person.gsx$navn.$t;
    document.querySelector("#detalje img").src = person.gsx$billede.$t;
    document.querySelector("#detalje img").alt = `Portræt af ${person.gsx$billede.$t}`;
    document.querySelector("#detalje .githubLink").href = `https://github.com/${person.gsx$github.$t}`;
    document.querySelector("#detalje .githubLink").textContent = `github.com/${person.gsx$github.$t}`;
}

function skjulDetalje() {
    document.querySelector("#detalje").style.display = "none";
}
```

Øvelse 4

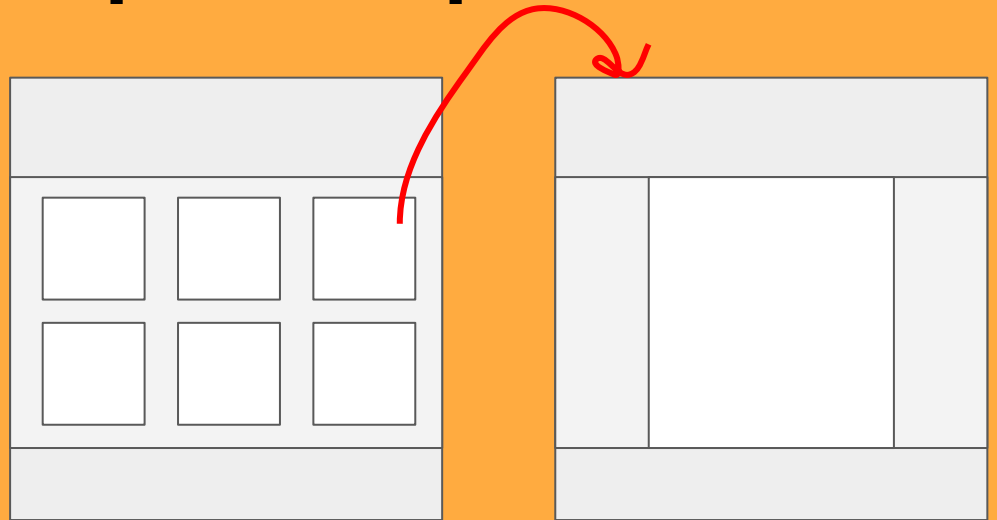
Gem en kopi af **03-åbn.html** i mappen under navnet **04-luk.html**.

Sæt en luk-knap ind på popup-vinduet og style, så den ser ok ud. (husk også cursor: pointer)

Lav en eventListener på knappen, som sørger for at lukke popupvinduet igen.

Så er popup-vinduet færdigt - bortset måske fra lidt ekstra styling -

detail-view på separat side

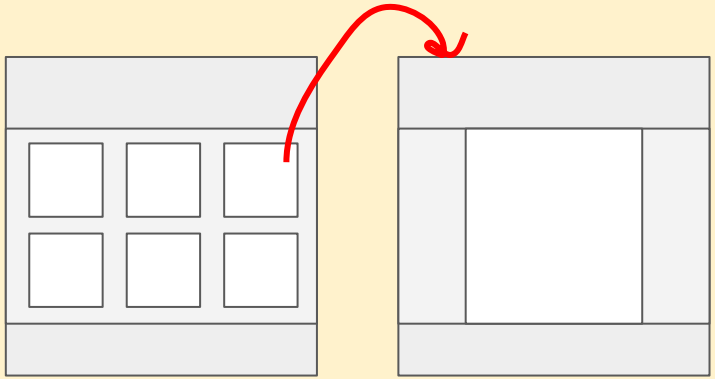


Fordele/ ulemper

Fordele:

Single-viewet får sin egen url-adresse

- siden kan nemmere deles
- siden kan findes af Google



Ulemper:

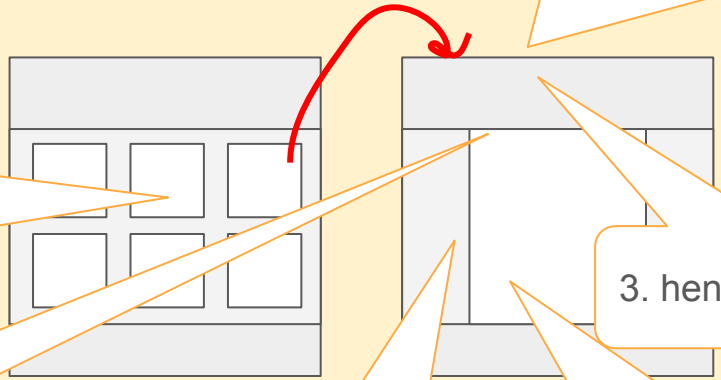
- der skiftes side, når der klikkes på en person - en underside skal altså laves

Separat-side strategi

2. Lav detalje.html:
(html og styling udfra #detalje)

1. Når der klikkes på et element i liste-viewet, skal browseren hoppe over til en ny side, detalje.html

En variabel skal medbringes



3. hent url-variablen

4. hent alle personer og find den rigtige

5. udfyld #detalje (genbrug function visPersoner() med indholdet fra visDetalje())

6. luk-knappen skal laves om til en tilbageknap
Få tilbageknappen til at virke

1. kald med url-variabel

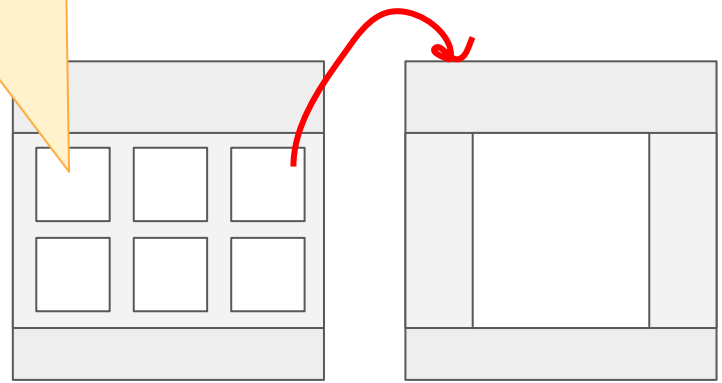
I loop-view:
Eventlisteners på alle elementer.
eventhandler:
spring over til den nye side.
I javascript kan man hoppe til en ny adresse med **location.href**

Medbring en variabel, som kan fortælle hvilket element, der blev klikket på

Variablen kan sendes som **url-variable**

1. Når der klikkes på et element i loop-viewet, skal browseren hoppe over til en ny side, single.html

En variabel skal medbringes



window.location

I javascript har vi et indbygget objekt, window.location
window.location har en lang række egenskaber
En af dem er href:

```
alert(window.location.href)
```

vil vise den aktuelle sides url i en alert-popup
Mens:

```
window.location.href = "http://kea.dk"
```

får browseren til at springe over til en helt anden side: keas webside.

"window" kan udelades, så vi skriver: location.href

url-variabler

Når en side kaldes med en url-adresse, kan man sende variabler med til siden. Sætter jeg denne url-adresse ind i browserens adressefelt:

https://kvikea.dk/hej.html?navn=Flemming

vil browseren hente siden https://kvikea.dk/hej.html, og den vil desuden give siden en variabel, navn, som er lig med "Flemming".

Vi kommer senere til, hvad modtagersiden gør, for at få fat i variablen.

hvilke url-variabler skal gemmes?

Den variabel, der skal overføres til den nye side, skal sætte den nye side i stand til at slå personen op, hvis alle personer hentes herfra.

Derfor skal den variabel vi overfører være **unik**.

Vi kan **ikke** bruge en variabel, hvis to personer kan have **samme værdi**

I personlisten kunne det være **email** eller **github**-kontonavn

Vi ved, at to personer **ikke** kan have **samme github-konto** eller **samme email**. Vi kan også bruge feltet : **id**

Derfor er de hver for sig en **unik nøgle**.

```
//funktion der viser personer i liste view
function vis() {
    const dest = document.querySelector("#liste"); // container til articles med en person
    const skabelon = document.querySelector("template").content; // select indhold af html skabelon
    (article)
    dest.textContent = "";
    personer.feed.entry.forEach(person => { // loop igennem json (personer)
        if (person.gsx$køn.$t == filter || filter == "alle") { // tjek hvilket køn personen har og
        sammenlign med filter eller vis alle
            const klon = skabelon.cloneNode(true);
            klon.querySelector(".navn").textContent = person.gsx$navn.$t;
            klon.querySelector(".navn").textContent += " " + person.gsx$hold.$t;
            klon.querySelector(".profil-billede").src = person.gsx$billede.$t;

            dest.appendChild(klon);
            dest.lastElementChild.addEventListener("click", () => {
                location.href = `detalje.html?id=${person.gsx$id.$t}`;
            });
        }
    });
}
```

Bliver fx til:

location.href = "06-single.html?id=21";

Øvelse 5

Gem en kopi af **04-luk.html** under navnet **05-kald.html**.

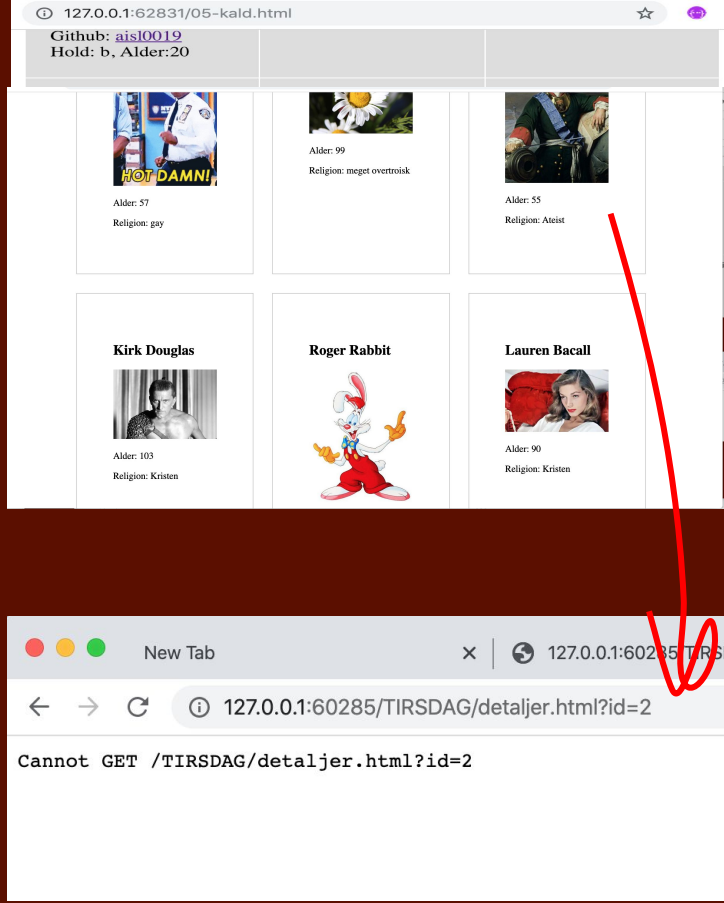
Fjern al html og css, som har med popup-vinduet at gøre
Fjern også i scriptet alt, hvad der har med popup'en at gøre: variablen popup, eventlisteners på personer og på lukke-knapper.

Eller -du kan også gemme en ny kopi af **Startfilen fra igår** under navnet **05-kald.html**.

Læg en eventlistener på alle personer, som får browseren til at hoppe til **06-detalje.html** medbringende personens id .

Test, at det virker: url'en i single.html skal have en variabel med.

vi laver **06-detalje.html** om lidt, men test at browseren prøver OG medbringer den rigtige url-variabel.



2. ny separat side: detalje.html

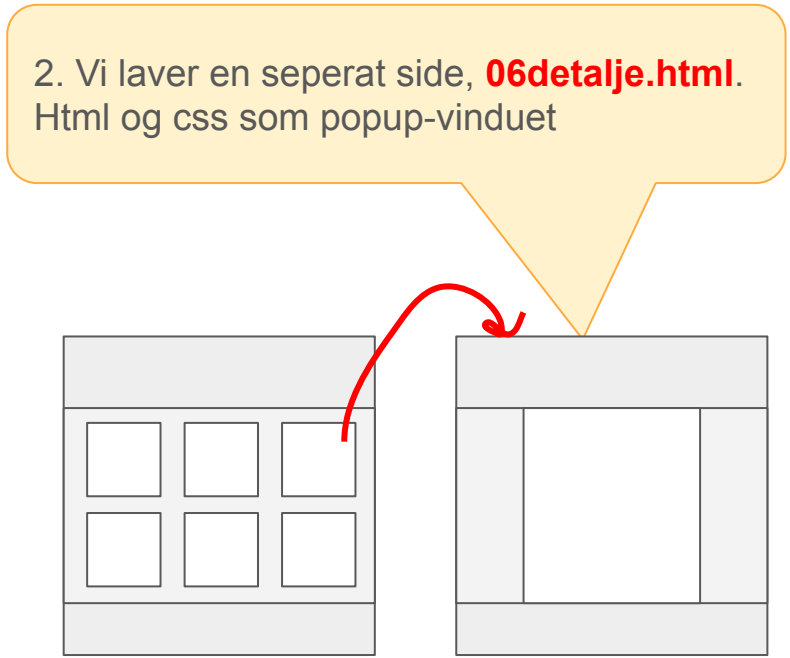
Lav en single-side.Tag udgangspunkt i en side med popup-vindue (fx. 03-åbn.html):

I html'en:

- væk med nav og liste
- bevar kun popup-vinduet, kaldet #detalje

I css'en:

- bevar kun styling omkring #detalje - dog skal intet længere være skjult eller have position: fixed (kun luk-knappen)



```
    <style>
        img {
            width: 100%;
            height 60%;
            object-fit: contain;
        }

        #detalje {
            position: fixed;
            width: 100vw;
            height: 100vh;
            top: 0;
            left: 0;
            background-color: rgba(0, 0, 0, .8);
            overflow: scroll;
        }

        #detalje #luk {
            position: fixed;
            top: 0;
            left: 0;
            font-size: 4rem;
            margin: 1em;
            cursor: pointer;
        }

        #detalje .person {
            width: 60vw;
            min-height: 50vh;
            margin: auto;
            padding: 4rem;
            background: white;
        }

    </style>
</head>

<body>

    <section id="detalje">
        <button id="luk">◁</button>
        <article class="person">
            <h2 class="navn">Navn</h2>
            <img src="" alt="" class="profil-billede">
        </article>
    </section>
```

Midlertidig hardcodet #detalje for at kunne tilpasse stylingen
-Fjernes inden de dynamiske værdier testes..

```
    <!-- En midlertidig hardcodet #detalje til brug for styling
    - slettes når tipasset -->
    <section id="detalje">
        <article class="person">
            <button class="luk">◁</button>
            <h2> Mit navn </h2>
            <img src="https://kvikea.dk/pix1.jpg" alt="">
            <p>Github:
                <a class="githubLink" href="#">github.com/kvikea</a>
            </p>
        </article>
    </section>

    <!--
    <section id="detalje">
        <article class="person">
            <button class="luk">◁</button>
            <h2></h2>
            <img src="" alt="">
            <p>Github:
                <a class="githubLink" href=""></a>
            </p>
        </article>
    </section>
-->
```

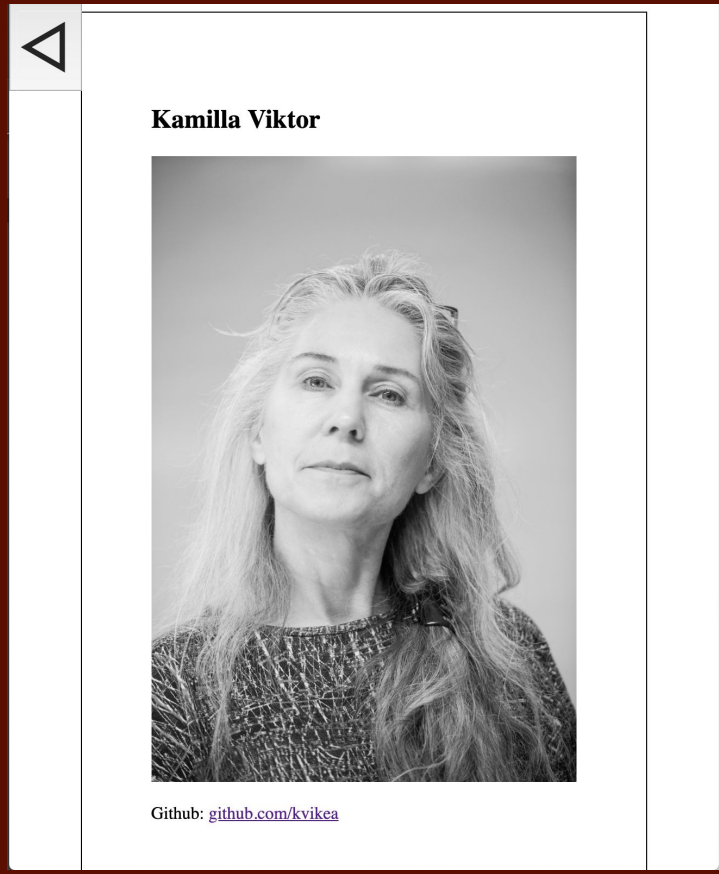
Øvelse 6

Gem en kopi af **03-åbn.html** under navnet **06-detalje.html**.

Fjern html og styling, som ikke har med #detalje at gøre. Tilpas stylingen for #detalje så den ikke længere er fixed eller skjult. (luk-knappen kan stadig være fixed og bruges som tilbageknap)

Eller - lav **06-detalje.html**. fra bunden.
i html : genbrug hele #detalje-sektionen
og al relevant styling.

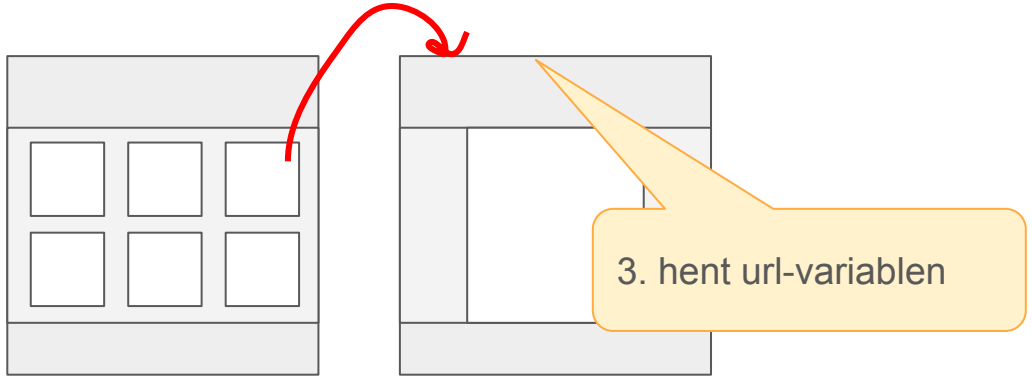
Lav evt. en midlertidig kopi af #detalje med rigtige værdier i, så du kan se hvad du styler..



3. Få fat i url-variablen med URLSearchParams

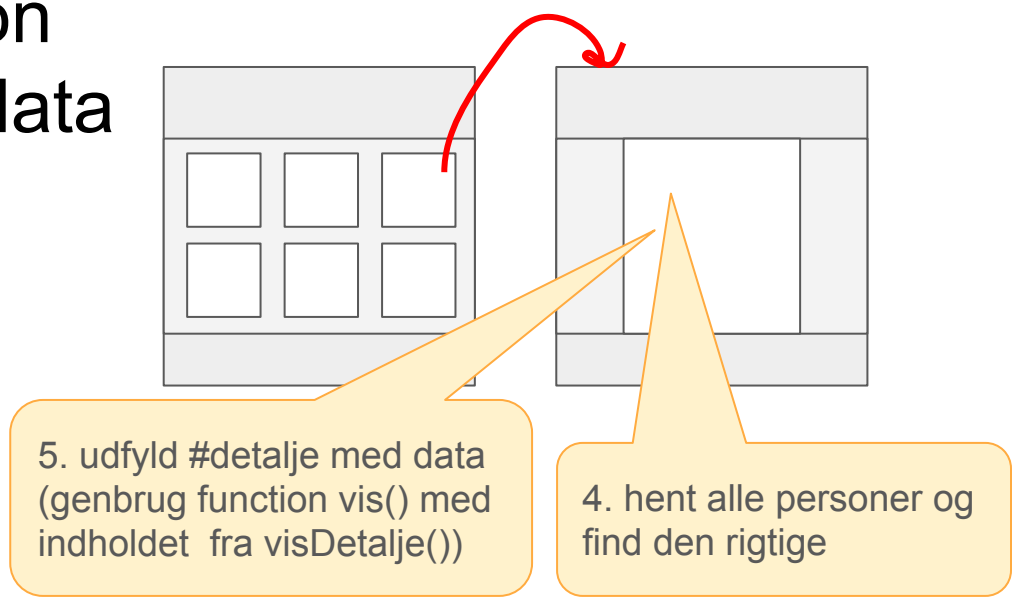
const urlParams = new URLSearchParams(window.location.search)
const id = urlParams.get("id");

Du skal skrive “ ?id=..... “ i adresselinien
eller klikke fra **05-kald.html** for at teste url-variablerne




```
const urlParams = new URLSearchParams(window.location.search)
const id = urlParams.get("id");
console.log(id);
let personer = [];
const idSheet = "1XWWbfWszD7f4jHqp51V_oT3pkHuR-ceEUw4YtrvK7F0"; // fiktive personer T7 F20202
const endpoint = `https://spreadsheets.google.com/feeds/list/${idSheet}/od6/public/values?alt=json`;
```

4. Find den rigtige person
5. Udfyld #detalje med data

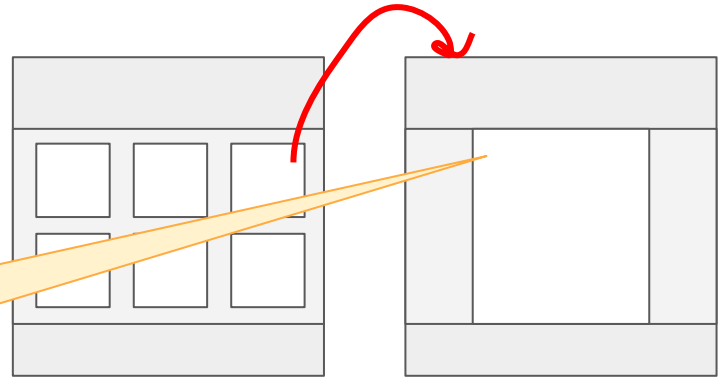


I scriptet:

- slet alt om filtrering, hvis der er noget endnu
- i loopet, og i if-sætningen: - erstat ned: if (person.gsx$id.$t == id) {
- //… vis indholdet fra visDetalje-funktionen - omringet af if sætningen
- };

```
<section id="detalje" class="skjul">
    <button id="luk">X</button>
    <article class="person">
        <h2 class="navn">Navn</h2>
        <img src="http://ale-kea.dk/images/alan_thumb.JPG" alt="" class="profil-billede">
    </article>
</section>


<script>
    const urlParams = new URLSearchParams(window.location.search)
    const id = urlParams.get("id");
    console.log(id);
    let personer = [];
    const idSheet = "1XWWbfWszD7f4jHqp51V_oT3pkHuR-ceEUw4YtrvK7F0"; // fiktive personer T7 F20202
    const endpoint = `https://spreadsheets.google.com/feeds/list/${idSheet}/od6/public/values?alt=json`;

    const popop = document.querySelector("#detalje");

    document.addEventListener("DOMContentLoaded", start);

    function start() {
        document.querySelector("#detalje #luk").addEventListener("click", (skjulDetalje));
        loadData();
    }

    async function loadData() {
        const response = await fetch(endpoint);
        personer = await response.json();
        console.log(personer);
        visPerson();
    }

    function visPerson() {

        personer.feed.entry.forEach(person => {
           if (person.gsx$id.$t == id) {
                popup.querySelector("h2").textContent = person.gsx$navn.$t;
                popup.querySelector("h2").textContent += " " + person.gsx$efternavn.$t;
                popup.querySelector("img").src = person.gsx$billede.$t;
                popup.querySelector("img").alt = person.gsx$navn.$t;
            }
        });
    }
```

6. Eventlistener og eventhandler til Tilbage-knap



6. luk-knappen skal laves om til en tilbageknap
Få tilbageknappen til at virke

```
document.querySelector("#detalje #luk").addEventListener("click", (skjulDetalje));

function skjulDetalje() {
    history.back();
}
```

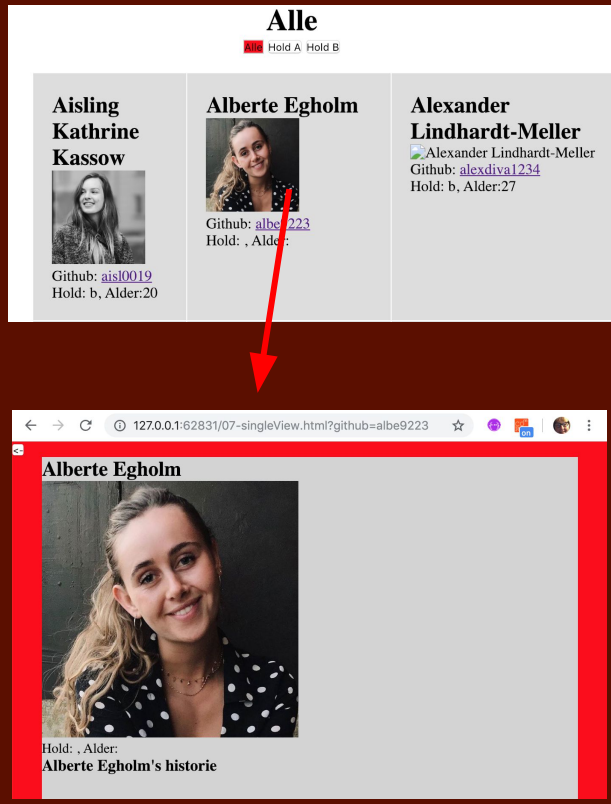
https://developer.mozilla.org/en-US/docs/Web/API/Window/history

Øvelse 7

Arbejd videre med **06-detalje.html**

I scriptet skal du :

- hente url-variablen ved hjælp af URLSearchParams.
- fjern alt overflødigt som filtrering osv.
- tilpas visPersoner(), med ny if-sætning
- Erstat luk-knappen med en tilbage-knap, som kan springe tilbage til **05-kald.html**
- husk, når du tester, at **06-detalje.html skal testes ved at klikke på en person i 05-kald.html**,



Babushka og singleview - hvad skal I lave

I skal lave **to løsninger**:

1. En Babushka-side med popup-detail-view

2. En Babushka-side, som åbner en ny side til detalje-visning

Lav de to løsninger i en mappe, **babushka**, og aflever et link til **mappen på Github**.

Du skal også uploade mappen til dit domæne.

Links til de to løsninger på domænet kan du lægge i **Readme-filen på dit github-repositiory** for mappen.