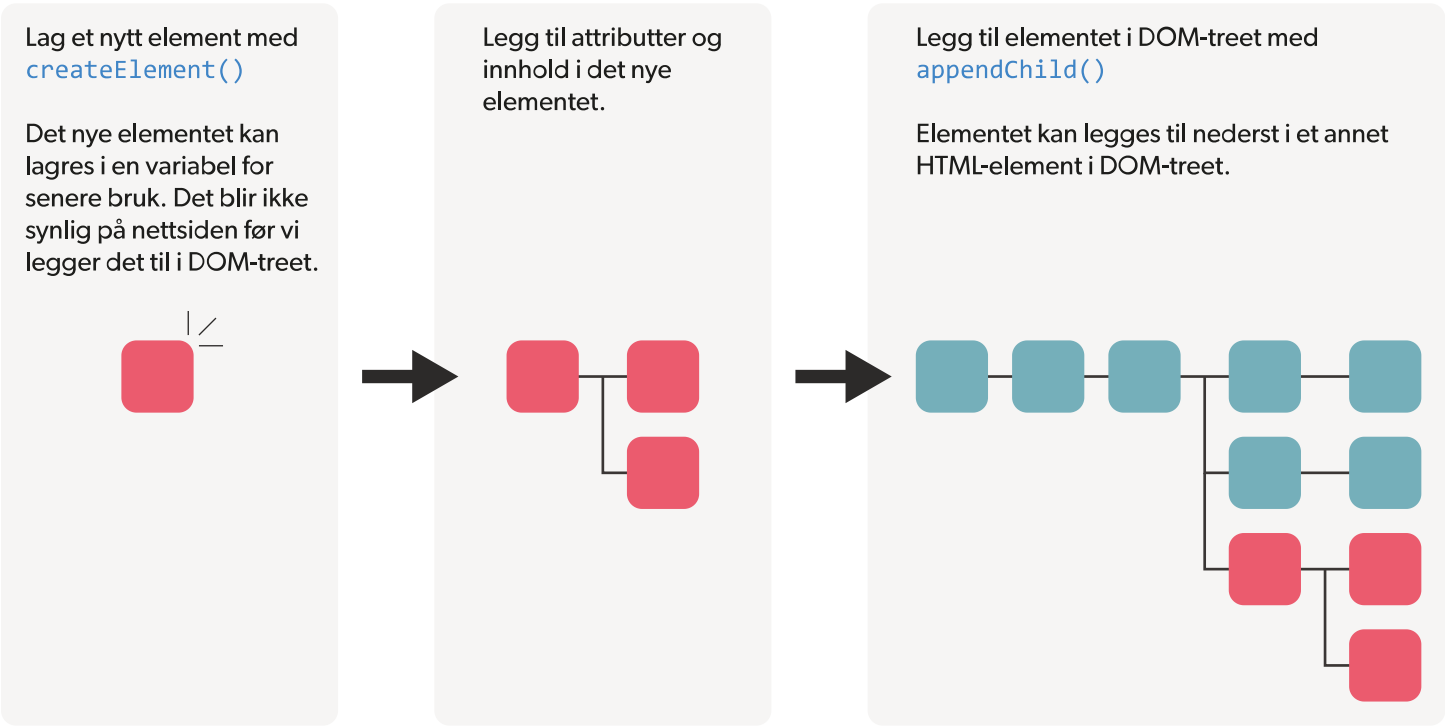
Lage HTML-elementer med JavaScript

Vi har allerede sett at vi kan legge til HTML-elementer ved å bruke egenskapen `innerHTML` . Det finnes også en annen framgangsmåte vi kan bruke for å legge til nye elementer.

I figuren nedenfor vises gangen i denne varianten. Vi må først lage et nytt element, før vi kan gi elementet attributter og innhold, og til slutt legge det til i DOM-treet (og da også på nettsiden).



Lage elementene

For å lage et nytt HTML-element bruker vi metoden

```
document.createElement();
```

```
let nyttAvsnittEl = document.createElement("p");
```

Legg merke til at vi skriver `"p"` og ikke `"<p>"`. Det nye elementet blir lagret i en variabel slik at vi kan jobbe videre med det. Vi kan også bruke denne variabelen til å gjøre endringer på elementet på et senere tidspunkt, slik som vi har sett på tidligere i dette læringsløpet.

Vi kan lage alle mulige HTML-elementer på denne måten, men hvis vi for eksempel ønsker å lage et `<li>` -element, er det viktig at det finnes et `<ul>` - eller `<ol>` -element som vi kan plassere det i.

Legge til innhold

Når vi har laget et nytt element, kan vi redigere attributter og innhold, som vi har sett på tidligere. Vi kan for eksempel legge til litt tekst og CSS-klassen `.blaa`:

```
nyttAvsnittEl.innerHTML = "Dette avsnittet er splitter nytt!";
nyttAvsnittEl.className = "blaa";
```

Her bruker vi egenskapen `className`, men metoden `setAttribute()` kan også brukes:

```
nyttAvsnittEl.innerHTML = "Dette avsnittet er splitter nytt!";
nyttAvsnittEl.setAttribute("class", "blaa");
```

Nå har vi et ferdig `<p>` -element med innhold og ett attributt. Det eneste som mangler, er å legge det til i DOM-treet, slik at det blir synlig på nettsiden.

Legge til elementer i DOM-treet

For å legge til vårt nye `<p>` -element må vi ha et sted der det er mulig å legge det til. Det vil si at vi må finne et element som vi kan plassere det nye `<p>` -elementet i.

For å legge til et nytt element bruker vi metoden `appendChild()` ; denne metoden legger til elementet nederst i elementet vi velger. Vi kan for eksempel legge det nye elementet nederst i `<body>` -elementet:

```
let bodyEl = document.querySelector("body");
bodyEl.appendChild(nyttAvsnittEl);
```

I koden nedenfor bygger vi opp alt HTML-innhold med JavaScript.

CSS JS Re

LIVE

```
// Henter body-elementet
let bodyEl = document.querySelector("body");

// Lager et div-element som vi kaller hoved
let hovedEl = document.createElement("div");
// Gir div-elementet id-en hoved
hovedEl.id = "hoved";
// Legger til div-elementet nederst i body-elementet
bodyEl.appendChild(hovedEl);

// Lager en overskrift
let h1El = document.createElement("h1");
// Legger til tekst i overskriften
h1El.innerHTML = "Lorem ipsum!";
// Legger til overskriften nederst i div-elementet
hovedEl.appendChild(h1El);

// Lager et avsnitt
let avsnittEl = document.createElement("p");
// Legger til tekst i avsnittet
avsnittEl.innerHTML = "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent congue, felis id egestas aliquam, felis orci facilisis est, suscipit fringilla magna sapien vel tellus. Nam sed elementum risus, hendrerit hendrerit tortor. Vivamus nec hendrerit quam.";
// Legger til avsnittet nederst i div-elementet
hovedEl.appendChild(avsnittEl);
```

Resources 1×

I koden ovenfor kan du se en fullstendig kode som utvider en tom HTML-side med et `<div>` -element, en overskrift og et avsnitt. Legg merke til at `<body>` -elementet i utgangspunktet er tomt.

Attributtet data-*

Vi har foreløpig brukt `id` -attributtet for å identifisere et element. Det fungerer fint når vi bare skal ha ett element med samme id, men hva om vi ønsker å lage flere elementer som er knyttet til samme unike verdi?

Det kan løses ved å bruke `class` -attributtet, men det er egentlig ment å brukes i forbindelse med CSS, og det blir fort rotete om et element skal få flere klasser knyttet til seg. Som en løsning på dette ble `data-*` -attributtet innført. Stjernen (`*`) kan byttes ut med en valgfri tekst. Dette attributtet lar oss derfor lage våre egne attributter, og vi kan legge til så mange vi ønsker.

La oss se på et eksempel der vi har noen firkanter med ulike egenskaper. Ved å bruke `data-*` -attributtet får vi enkelt tilgang til firkantenes viktigste egenskaper:

```
<div data-form="kvadrat" data-areal="100"></div>
<div data-form="kvadrat" data-areal="64"></div>
<div data-form="rektangel" data-areal="100"></div>
<div data-form="rektangel" data-areal="80"></div>
```

Legg merke til at flere av elementene har de samme verdiene i sine attributter. Elementene har også mer enn ett attributt. Vi kunne ikke ha brukt `id` -attributtet til noen av delene.

Vi kunne ha gitt disse elementene hver sin id eller en felles klasse, hvis vi for eksempel ønsker enkel tilgang til dem med `querySelector()` , men vi kan også bruke `querySelectorAll()` for å hente alle:

```
let firkanterEl = document.querySelectorAll("div");
```

Husk at hvis disse elementene ligger inni et annet `<div>` -element, for eksempel `<div id="hoved">` , så må vi i stedet skrive:

```
let firkanterEl = document.querySelectorAll("#hoved > div");
```

Nå kan vi gå gjennom elementene våre og hente ut den informasjonen vi ønsker:

```
for (let i = 0; i < firkanterEl.length; i++) {
  let firkant = firkanterEl[i];
  console.log(firkant.dataset.form);
  console.log(firkant.getAttribute("data-areal"));
}
```

Legg merke til at vi henter ut attributtene på to ulike måter her. Vi kan bruke `getAttribute()` , som vi har sett på tidligere, eller vi kan bruke `dataset` -objektet, som er enda enklere. Attributtene vi har lagt til, blir egenskaper i `dataset` -objektet.

Hvis vi ønsker å legge til et `data-*` -attributt på et element med JavaScript, kan vi bruke `setAttribute()` , men også her er det enklere å bruke `dataset` -objektet. Vi kan for eksempel nummerere firkantene:

```
for (let i = 0; i < firkanterEl.length; i++) {
  let firkant = firkanterEl[i];
  firkant.dataset.nummer = i;
}
```

I eksemplet nedenfor har vi plassert flere firkanter på en nettside ved å bruke `createElement()` . Der bruker vi `data-*` -attributtet for å holde orden på arealene, og for å finne det største arealet blant firkantene.

Legg merke til at verdiene `i data-*` -attributter lagres som tekst. Det betyr at vi må gjøre dem om til tall hvis vi skal bruke dem som tall. I dette eksemplet kunne vi selvfølgelig ha bestemt det største arealet *mens* vi laget elementene, men vi ville vise hvordan vi kan gjøre det etter å ha hentet inn elementene.

Mange problemer kan løses uten `data-*` -attributter, men disse attributtene kan ofte gjøre problemene mye lettere å løse. De kan for eksempel brukes til å sende informasjon fra et element vi klikker på, for eksempel en farge vi ønsker å bruke på et annet element:

```
<div data-farge="#00F4b5"></div>
```

Her kan vi bruke `data-farge` for å finne fargen som skal brukes.

De kan også brukes hvis vi skal lage et rutenett, som i et spillbrett. Da er det nødvendig å ha oversikt over hvilken rad og hvilken kolonne en rute tilhører. Det kan vi for eksempel løse slik:

```
<div data-rad="3" data-kolonne="2"> </div>
```

Da vet vi umiddelbart hvilken rute i spillbrettet vi har med å gjøre.

Ofte er det slik at hvis du jobber med et problem og synes det er vanskelig å håndtere all informasjonen på en god måte, så kan det hende at `data-*` -attributter kan hjelpe deg.

Oppgaver

1. Lag et tomt HTML-dokument med to CSS-klasser som gir ulike bakgrunnsfarger.
2. Bruk metodene `createElement()` og `appendChild()` for å lage en liste med fire listepunkter. Listepunktene skal bruke de to CSS-klassene du laget i forrige oppgave, slik at oddetallspunktene har den ene fargen, mens partallspunktene har den andre.
3. Gjenta det du gjorde i forrige oppgave, men bruk i stedet en løkke som lager listepunktene. Listen skal ha 12 listepunkter. Teksten kan være den samme i alle listepunktene.
4. Lag en funksjon som lager en liste som i forrige oppgave, men antallet listepunkter skal angis som et argument til funksjonen.
5. Lag en tabell med 3 kolonner og 4 rader med metodene `createElement()` og `appendChild()` .
6. Lag et rutenett av `<div>` -elementer på tre ganger tre ruter. Hver rute skal få et `data-*` -attributt som lar deg nummerere rutene.
7. Hent `<div>` -elementene fra forrige oppgave med JavaScript. Bruk `data-*` -attributtene for å skrive ut rutenes nummer inni rutene.