

Litt om CSS



Jan R Sandbakken

Litt om CSS

Jan R Sandbakken

Version 1.0 2026-02-12

Innholdsfortegnelse

Litt om CSS	1
Overordnet HTML	1
Hva er CSS?	3
Globale CCS-settninger	4
Boksmodellen	5
px	6
rem og em	7
%	7
vw og vh	7
Content box vs Border box	8
Display	8
1. ID og Class	9

Litt om CSS

Overordnet HTML

Et HTML-dokument kan strukturelt sett se slik ut:

```
<html>
├── <head> ... metadata, CSS, title ...
└── <body> — hele synlige innholdet på siden
    ├── <header> ... toppbanner, logo, navigasjon ...
    ├── <nav> ... hovedmeny, lenker ...
    ├── <main> ... hovedinnholdet, unikt for denne siden
    │   ├── <section> ... logisk gruppering av innhold
    │   ├── <article> ... en artikkel, blogginnlegg, etc.
    │   └── <div> ... ekstra beholder for styling/layout
    ├── <aside> ... sidepanel, ekstra info, reklame ...
    └── <footer> ... bunntekst, copyright, kontaktinfo ...
```

Her er et HTML-eksempel, med syntaks og viktige elementer:

```

<!DOCTYPE html>
<html lang="no">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Litt om CSS</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <header>
    <h1>Litt om CSS</h1>
    <p>Av Jan R Sandbakken</p>
  </header>

  <main>
    <section>
      <h2>Introduksjon</h2>
      <p>Dette er et eksempel på et avsnitt i HTML.</p>
      <pre>
Dette er preformatert tekst.
Beholder mellomrom og linjeskift.
      </pre>
    </section>

    <section>
      <h2>Eksempel på div</h2>
      <div class="boks">
        Dette er en div som ofte brukes for å style grupper av innhold.
      </div>
    </section>
  </main>

  <footer>
    <p>© 2026 Jan R Sandbakken</p>
  </footer>
</body>
</html>

```

Forklaring:

- **<header>** – overskriftsseksjon på toppen av dokumentet
- **<body>** – alt det synlige innholdet (inkl. header, footer, sidepanel ...)
- **<main>** – hovedinnhold
- **<section>** – logiske grupper av innhold (f.eks. kapitler, emner)
- **<div>** – generell beholder. Brukes for layout og styling
- **<p>** – avsnitt
- **<pre>** – preformatert tekst, beholder linjeskift og mellomrom

- `<footer>` – bunntekst

CSS kan ikke kompensere for dårlig strukturert HTML. God HTML først, så CSS.

Hva er CSS?

CSS står for *Cascading Style Sheets*, som representerer:

- Cascading → Regler “kaskaderer”, dvs. nye regler kan overstyre gamle basert på spesifisitet og rekkefølge.
- Style → Angir hvordan elementer ser ut: farge, størrelse, plassering, marg, osv.
- Sheets → Samles i en fil (ofte style.css) og kobles til HTML.

Hvordan CSS henger sammen med HTML

1. Selektor – hva du vil style (f.eks. **p**, **.boks**, **#header**)
2. Egenskap – hva du vil endre (f.eks. **color**, **font-size**, **margin**)
3. Verdi – hvordan du vil endre det (f.eks. **blue**, **1.2rem**, **10px**)

Eksempel:

```
/* Stiler alle avsnitt */
p {
  color: darkblue;
  line-height: 1.5;
}

/* Stiler alt med klasse "boks" */
.boks {
  background-color: #f0f0f0;
  padding: 1rem;
  border: 1px solid #ccc;
}
```

Hovedprinsippene er at:

- CSS separerer struktur (HTML) fra utseende (CSS) → lettere vedlikehold.
- En regel består alltid av selektor + deklarasjonsblokk
- CSS arves nedover i dokumentet (f.eks. **color** fra `<body>` arves til `<p>`)

Konstruksjonen som bestemmer verdier, ser generelt slik ut:

```
selektor {
  egenskap: verdi;
  egenskap: verdi;
}
```


ved at

- **selektor** (selector) bestemmer hvilke HTML-elementer regelen gjelder for
- `{ ... }` Regelblokk (rule block) inneholder én eller flere deklarasjoner
- **egenskap** (property) angir hva vil endre (f.eks. margin eller display)
- **verdi** (value) er verdien man setter på egenskapen (f.eks. red eller 10px)

Når det gjelder arv, er det slik at noen egenskaper arves (som `color`) og andre ikke (som `padding`), og i tillegg er det et arvehierarki som gjelder I HTML/CSS (som f.eks `<body>` → `<div>` → `<p>`). Vi skal komme tilbake til dette, men de formelle spesifikasjonene er å finne på

- [MDN CSS Reference](#)
- [CSS Specification \(W3C\)](#)

Globale CCS-settinger

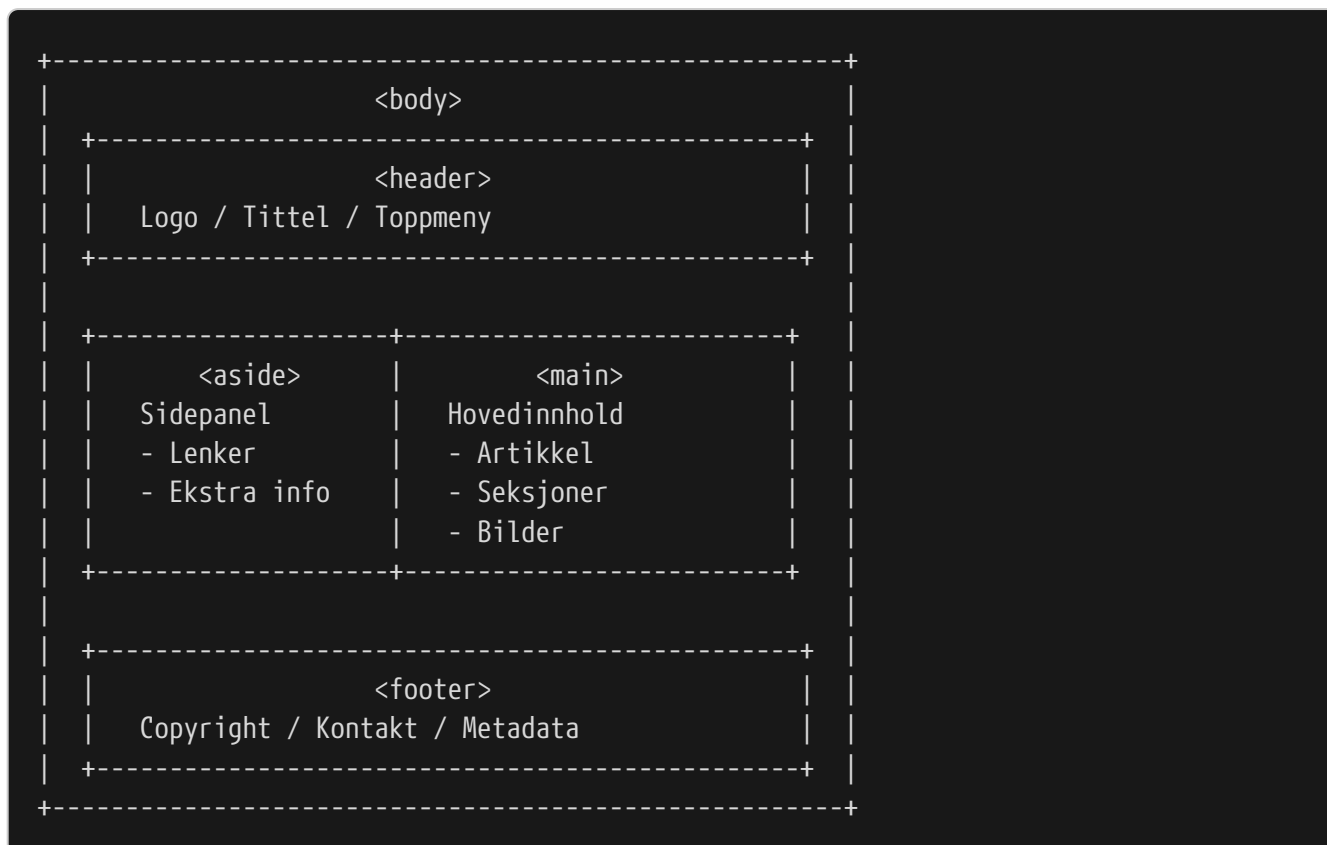
Her ser vi eksempel på noen globale settinger

```
html {
  font-size: 16px; /* gir grunnstørrelse for rem */
}

body {
  font-family: system-ui, sans-serif; /* lettest standardfont */
  line-height: 1.6;                  /* behagelig linjeavstand */
  color: #222;                       /* tekstfarge */
  background-color: #f5f5f5;         /* bakgrunnsfarge */
}
```

I HTML settes ting som bør være globalt, men sjelden knyttes til visuelt design. HTML-settinger er typisk språk og metadata, globale layout-verdier, scroll-behavior etc.

BODY representerer det synlig innholdet, laget som faktisk tegnes i vinduet. Her er det typisk å sette bakgrunnsfarge, font-family, fontfarge, marger og padding, layout-begrensning som max-width mm. BODY representerer hele sider, inklusive sidepanel, *footer* og *header* mm, og dette styrer hva som er naturlig å sette her og f.eks. ikke i MAIN (som representerer hovedinnholdet, og som *kan* ha egen bakgrunnsfarge, marger etc. uten å påvirke topp og bunn og sidepanel fra BODY, hvis ønskelig). Følgende figur illustrerer situasjonen:



Boksmodellen

De fleste elementer har boksegenskaper, f.eks. overskrifter, **p**, **div**, **span**, **section**, **article**, **image** m.fl. Dvs de kan ha marger, rammer og ulike former for luft rundt seg. Dette er konkret vist i figuren under.



Måter å spesifisere størrelse på er: **px**, **em**, **rem**, **%**, **vw**, **vh**

De brukes på ikke bare på **margin**, **padding** osv, men også på **width**, **font-size** o.l.

px og **em** er de klart vanligste.

px

Verdier kan settes direkte, som her:

```
div {  
  margin: 10px;  
  padding: 20px;  
}
```

Men siden disse elementene angir luft på flere sider, kan angi flere verdier samtidig, og der rekkefølge av tallene (top → right → bottom → left, med klokka) angir :

```
div {  
  margin: 10px 20px 30px 40px;  
}
```

	10px	
40px		20px
	30px	

Man har også definert betydningen

```
div {  
  margin: 10px;           /* alle */  
  margin: 10px 20px;      /* T/B  R/L */  
  margin: 10px 20px 30px; /* T  R/L  B */  
  margin: 10px 20px 30px 40px; /* T  R  B  left */  
}
```

Videre har man mulighet for å bruke **margin-top**, **margin-right**, **margin-bottom** og **margin-left** også, og det tilsvarende også for **padding** og **border**. Men sistnevnt består egentlig av tre deler,

- **width**
- **style**
- **color**

for tykkelse, rammestil og farge, så disse ser dermed slik ut:

```
border-top-width:
border-top-style:
border-top-color:
```

osv.

rem og em

rem og **em** angir størrelse relativt ift. font-size hos hhv. rot (HTML) eller hos seg selv (eller forelderelementet).

1em = 1 × gjeldende font-size, 2em = 2 × gjeldende font-size osv.

Mer konkret:

- font-size → baseres på forelder
- padding og margin etc → baseres på elementets egen beregnede font-size

Hvis ingen font-size er eksplisitt satt, brukes nettleserens standardverdi, som ofte er 16px.

%

Prosent er relativ til foreldreelementet.

```
.container {
  width: 800px;
}

.box {
  width: 50%;
}
```

vw og vh

vw og **vh** står for hhv. vertikal og horisontal viewport.

- 1vw = 1% av skjermes bredde i px
- 1vh = 1% av skjermen bredde i px høyde

```
div {
  width: 50vw;
  height: 100vh;
}
```

Alle typer angivelse kan blandes:

```
div {
  width: 20rem;
  padding: 2em;
  margin: 5%;
  height: 50vh;
}
```

Content box vs Border box

Vi har to typer bokser, *content* og *border box*. For en contentbox (som er default) av en spesifikk størrelse kommer padding og border i tillegg for den endelig størrelsen av boksen. For border-box er spesifikk størrelse den faktiske størrelse. (Margin gjelder alltid utenfor boksen, uansett hvilken box-sizing man bruker.)

```
/* content-box: standard */
div.content-box {
  box-sizing: content-box;
  width: 300px;
  padding: 20px;
  border: 5px solid black;
}

/* border-box: praktisk for layout */
div.border-box {
  box-sizing: border-box;
  width: 300px;
  padding: 20px;
  border: 5px solid black;
}
```

Disse påvirker tolkningen av WIDTH og HEIGHT for innholdet.

Display

Blokker kan også vises på ulike måter. De kan vises under hverandre eller ved siden av hverandre, hvilket styres med ulike valg for display:

- `display: block`
- `display: inline`
- `display: inline-block`

Det fins i tillegg to valg `flex` og `grid` som vi evt. behandler senere.

Betydningen er som følger:

- Blokk-elementer legger seg under hverandre

- Inline-elementer legger seg ved siden av hverandre
- Inline-block gjør det samme, men respekterer width/height

En **display: block** starter på ny linje og tar hele tilgjengelige bredden (som standard). Den respekterer både **width**, **height**, **margin**, **padding** og **border**.

BLOCK-ELEMENTER

```
+-----+
|      div      |
+-----+

+-----+
|      div      |
+-----+

+-----+
|      div      |
+-----+
```

display: inline gir inline-elementer. Den starter ikke på ny linje, de legger seg ved siden av hverandre, flyter inni tekstlinjen og ignorerer **width** og **height** (dvs. at høyde og bredde tilpasses innholdet). Den respekterer heller ikke vertikal margin/padding slik blokker gjør.

```
+-----+ +-----+ +-----+
| box 1 | | box 2 | | box 3 |
+-----+ +-----+ +-----+
```

display: inline-block er lik, bare at den respekterer **width**, **height**, **margin** og **padding** fullt ut

Margin skaper avstand mellom alle disse.

Normal flyt – blokk

Blokkelementer plasseres vertikalt under hverandre, fyller tilgjengelig bredde og skyver påfølgende elementer nedover.

Normal flyt – inline

Inline-elementer plasseres horisontalt i samme linje som tekst, bruker bare nødvendig bredde og brytes automatisk til ny linje når det ikke er mer plass.

Normal flyt kan endres med **position**, **float**, **flex** og **grid**, som gir alternative måter å plassere elementer på.

1. ID og Class

```
id > class > element
```