
CSS

***HTML:***
*Strukturen (Skjellettet)*

***CSS:***
*Stilen og utseendet (maling og design)*

## hva_er_CSS {

**Hoved oppgaven i CSS er å gi nettsider et visuelt utseende.**

CSS står for **Cascading Style Sheets**.

**Cascading** (fossende/rangerende) betyr at reglene har en bestemt rekkefølge og prioritet; hvis to regler prøver å style det samme elementet, avgjør "cascading"-prinsippet hvilken stil som vinner.

En **Style Sheet** er en liste over regler som beskriver hvordan HTML-elementer skal vises, inkludert farger, fonter og layout.

Til slutt er **Style** selve utseendet og designet som skiller en nettside fra et enkelt, uformatert HTML-dokument. Sammen sørger disse tre elementene for at vi kan lage moderne, brukervennlige og visuelt tiltalende nettsider.

{

eksempel {

```
p {
color: red;
}

p {
color: blue;
}

{
```

**eksempel {**

```
p {
Color: red;
}

p {
Color: blue;
}
```

**{**

***Den siste vil gjelde***

*Fordi nettleseren leser CSS-regler fra topp til bunn og den siste regelen med lik spesifisitet vinner, blir den røde stilen umiddelbart* ***overstyrt*** *av den blå stilen før elementet tegnes på skjermen, slik at du kun ser* ***blå*** *tekst.*

## Begrunnelse {

Nettsiden blir **bare blå**. 🟦
Dette skjer så raskt under innlastingen av siden at du **ikke** vil se et blunk av rødt:

- Når nettleseren tolker HTML-filen, ser den at det er en lenke til et CSS-stilark
- Den leser CSS-filen **sekvensielt (fra topp til bunn)**.
- Når den kommer til den første regelen (p { color: red; }), noterer den seg at paragrafene skal være røde.
- Umiddelbart etterpå leser den den andre regelen (p { color: blue; }). Siden denne regelen har lik spesifisitet og kommer **etter** den røde regelen, **overstyrer** den den forrige i nettleserens minne.
- Før elementene tegnes på skjermen, er den **endelige, gjeldende** regelen for fargen til <p> allerede fastsatt til å være **blå**.

Du ser altså aldri det røde; nettleseren bruker kun den **siste gyldige** stilen den fant.

{

```html
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="style.css">
    <title>Document</title>
</head>
```

## kobling_HTML_med_CSS {

For at stilene du skriver i CSS-filen skal påvirke HTML-elementene, må du **lenke** de to filene sammen. Den mest ryddige og profesjonelle måten å gjøre dette på er å bruke et **Eksternt Stilark**

**Opprett en CSS-fil:** Du lager en egen fil (som regel kalt **style.css**) der all CSS-koden din bor.

**Koble i <head>:** Inne i <head>-seksjonen på HTML-siden din, legger du til en <link>-tagg som peker til style.css-filen.

**Resultatet:** Nettleseren vet nå at den må hente stilene fra style.css og bruke dem på HTML-strukturen før den tegner nettsiden for brukeren.

{

***Huske-regel***

***1. Priotert***
***Id*** = *Metaforisk: Fødsels og personnummer*

***2. Priotert***
***Clas*** = *Metaforisk: Etternavn*

***3. Priotert***
***HTML-tag*** = *Metaforisk:  Navn*

I HTML-filen

```
<img id="hoved-logo" src="" alt="">
<a class="meny-knapp" href=""></a>
<h1> </h1>
<p> </p>
<div> </div>
```

## Selektorens_rolle {

**Definisjon:** Selektoren er den første delen av en CSS-regel og forteller nettleseren *hvilke* elementer på nettsiden stilen inne i krøllparentesene skal gjelde for.

**Eksempel:** I regelen p { color: red; } er **p** selektoren. Den velger *alle* paragraf-elementer (<p>) på nettsiden.

**Element-selektorer:** Velger elementer basert på HTML-taggen (f.eks. <h1>, <p>, ).

**Klasse-selektorer:** Velger elementer basert på class-attributtet (starter med punktum, f.eks. .meny-knapp). Dette er den mest fleksible typen.

**ID-selektorer:** Velger ett unikt element basert på id-attributtet (starter med #, f.eks. #hoved-logo).

{

I CSS-filen

```
#hoved-logo {
    color: red;
}

.meny-knapp {
    color: red;
}

h1 {
    color: red;
}

p {
    color: red;
}

div {
    color: red;
}
```

HTML_tagg_er_firkantet_boks {

Alle HTML-elementer, som bilder, paragrafer og knapper, er firkantede **bokser**. For å kontrollere avstand og layout, må vi forstå boksens fire lag: .

| Lag | Hva det gjør | Formål | CSS-eksempel |
|---|---|---|---|
| **Margin** | **Utvendig** avstand. | Lager luft *mellom* bokser. | margin: 20px; |
| **Border** | Linjen/rammen rundt. | Visuell avgrensning. | border: 1px solid black; |
| **Padding** | **Innvendig** avstand. | Lager luft *inne i* boksen (mellom innhold og ramme). | padding: 15px; |
| **Content** | Selve innholdet. | Tekst, bilde, video. | width: 300px; |

{

# Forside

Lorem ipsum dolor sit amet consectetur adipisicing elit. Porro cumque voluptatibus, quas magni hic quis. Ex magni cupiditate quos. Maxime eos blanditiis quasi, tenetur ullam repudiandae error reiciendis? Quo, corrupti?

**Huske regel**

**Aktivere FLEXBOKS**

```
main {
   Display: flex
}
```

**Aktivere GRID**

```
main {
    Display: grid
}
```

## Flexboks og Grid {

Fremtidens layout handler om **Flexbox** og **Grid**, og de funker litt annerledes: Du flytter ikke på selve elementene, men du gir beskjed til **boksen elementene ligger inni** (forelderen).

Hvis du for eksempel har en navigasjonsbar (<nav>) med lenker (<a>), skal du sette stilen display: flex; på <nav>-taggen, ikke på lenkene. <nav> blir da "sjefen" (containeren), og du bruker dens egenskaper til å si:

"Hei, Nav-boks! Plasser alle lenkene dine pent på midten og gi dem lik avstand." Dette gir deg superkrefter for å lage kule og ryddige nettside-layouter.

{

**Øvelse {**

Prøv deg på Grid med spille som du finner i denne linken:

https://cssgridgarden.com/

Prøv deg på Flexboks med spille som du finner i denne linken:

https://flexboxfroggy.com/

**{**