HTML Tables – Styling Tables

Indholdsfortegnelse

Indledning *3*

HTML Tabeller *4*

Styling af tabeller – Ved brug af CSS. *5*

Borders (Grænser) **5**

Collapsed Borders **6**

Cell padding (Luft mellem indhold i celler og borders) **6**

Left-align headings (Styling af overskrifterne i tabellen (th)) **7**

Border Spacing (Afstand imellem cellerne) **7**

Colspan (Celle som spænder over mere end en kolonne) **8**

Cells that span many rows (Celle som spænder over mere end en række) **9**

Adding a Caption (Tilføj billedtekst til din tabel) **10**

A special style for one table (Tilføj en unik style til en specifik tabel) **10**

Add more styles (Giv den gas med forskellige styles) **12**

Kilder *14*

Indledning

Dette emne handler om brugen af tabeller i HTML, og hvordan vi efterfølgende kan style dem i CSS.

Der kan være mange fordele ved at bruge tabeller i HTML. Det kan bl.a. være med til at skabe et hurtigt overblik over noget forskelligt information. Derfor kan brugen af tabeller være med til at øge brugervenligheden på en hjemmeside.

Hovedsagligt anvendes html tabeller til tabulært indhold dvs. indhold/data, som giver mening at placere i en tabel – Det kunne f.eks. være en liste over medarbejdere, hvor du kan finde oplysninger om navn, stilling, e-mail, telefon osv.

HTML tabeller uden ”styling” kan forekomme en smule kedelige, men takket være CSS har vi nu mulighed for nemt at peppe vores HTML tabeller op.

Styling af tabeller er vigtigt – Da vi kan style vores tabeller, så de matcher den sammenhæng de skal bruges i.

I denne opgave vil jeg give nogle kodeeksempler på, hvordan vi rent teknisk opretter HTML tabeller, og hvordan vi efterfølgende kan style dem i CSS. Eksemplerne vil blive udarbejdet i Brackets, men vil derudover også fremgå i dette word dokument.

I de eksempler jeg giver, vil der være relevant og forklarende tekst, som understøtter eksemplerne.

HTML Tabeller

De første grundlæggende tags man skal bruge for at oprette en tabel er: <table> <tr> <th> <td> <caption>.

- Det første tag man skal bruge, når man opretter en HTML tabel er <table> - Dette tag forklarer html-filen, at man ønsker at skabe en tabel.

- En række i en tabel defineres med tagget <tr> (Det står for Table row, altså tabelrække)

- En tabel overskrift defineres med tagget <th> (table header) – f.eks. Fornavn, efternavn, alder.

- Alt data/indhold i tabellen defineres med tagget <td> (tabel data) - f.eks. Per, Nielsen, 30 år. Td elementet indeholder alt data som tabellen indeholder – Bemærk: Det kan indeholde forskellige HTML elementer, text, images, lists, andre tabeller osv.

- Hvis du ønsker en billedtekst til din tabel, skal du bruge tagget <caption>

I brackets har jeg udarbejdet et eksempel på en tabel omkring guitarister, der skal kortlægge Guitarist, Band, Guitarmodel, Forstærkertype.

Eksempel 1. Kun HTML – Ingen CSS.

```html
<!DOCTYPE html>
<html lang="da">

    <head>
        <meta charset=utf-8>
        <title> HTML Tabeller</title>
        <link rel="stylesheet" href="Tabeller.css">
    </head>

    <body>
        <header>
            <h1> HTML Tabeller og Styling Tables</h1>
            <p> Sådan pepper du dine kedelige HTML Tabeller</p>
        </header>

        <table>
            <tr>
                <th> Guitarist</th>
                <th> Band </th>
                <th> Guitar</th>
                <th> Forstæker</th>
            </tr>

            <tr>
                <td> Jimi Page</td>
                <td> Led Zeppelin</td>
                <td> Gibson Les Paul</td>
                <td> Marshall </td>
            </tr>

            <tr>
                <td> David Gilmour</td>
                <td> Pink Floyd</td>
                <td> Fender Stratocaster</td>
                <td> Fender twin Rewerb</td>
            </tr>

            <tr>
                <td> Angus Young</td>
                <td> AC/DC</td>
                <td> Gibson SG</td>
                <td> Marshall</td>
```

HTML Tabeller og Styling Tables

Sådan pepper du dine kedelige HTML Tabeller

Guitarist	Band	Guitar	Forstæker
Jimi Page	Led Zeppelin	Gibson Les Paul	Marshall
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb
Angus Young	AC/DC	Gibson SG	Marshall

Til venstre: Billede af selve koden i html, og til højre ser vi et billede af koden åbnet i en browser.

Som vi kan se i ovenstående eksempel, så ser tabellen ikke speciel spændende eller overskuelig ud. Det er her, at styling med CSS kommer ind i billedet.
Jeg har bevidst valgt at adskille HTML koden og CSS fra hinanden, med henblik på at adskille struktur og layout. Der er forskellige fordele ved at adskille HTML og CSS.

A. At det sikrer en ensartethed imellem siderne
B. At man kan ændre layoutet i en tabel fuldstændigt, uden overhovedet at ændre i indholdet (HTML´en). Dvs. Hvis man ønsker at ændre designet på en tabel, kan dette gøres direkte i CSS´en, da CSS er et sprog som fortæller noget om udseende.

Styling af tabeller – Ved brug af CSS.

Borders (Grænser)

Det første jeg vil starte med at style, er nogle grænser (borders) omkring tabellen. Det vil være med til at gøre tabellen nemmere at læse og indholdet mere overskueligt.

Man skal bruge CSS til at specificere borders (grænser), ellers vil tabellen blive vist uden grænser, ligesom eksemplet på ovenstående side.

Først skal man vælge sin selector (Der kan sagtens være flere selectors). I dette tilfælde vil mine selectors være table, td, tr og th – Det er alle sammen elementer som vedrører tabellen.
Herefter skal man sætte et start-tag { - som fortæller at her starter stylingen og afslutte med et slut-tag }.

Imellem start og slut-tagget indsætter man sin property og value. I dette tilfælde vil min property være ”border” – Det fortæller at jeg ønsker at sætte grænser.
Min value kan f.eks. være 1px solid black - Det angiver at jeg ønsker en solid sort streg med en linjetykkelse på 1 pixel.

Eksempel 2: Border. – CSS til venstre og Browservindue til højre.

```
table {
    width: 600px;
}

table, th, td {
    border: 1px solid black;
}
```

Guitarist	Band	Guitar	Forstæker
Jimi Page	Led Zeppelin	Gibson Les Paul	Marshall
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb
Angus Young	AC/DC	Gibson SG	Marshall

Det første jeg har stylet er selve størrelsen på tabellen:
Her har jeg brugt table, som min selector. Width som min property, og 600px som min value. Selve bredden på tabellen, altså de 600px, har jeg fundet ved at prøve mig lidt frem. Det kan være svært at vide, hvor bred en tabel skal være, da det afhænger af, hvor meget indholdet i de enkelte celler

fylder – **Bemærk: Jeg ændrede senere antal pixel til 800px – Da det passede bedre, da jeg kom længere med opgaven.**

Derefter har jeg sat borders (grænser) på tabellen. Det har gjort ved at bruge table, th, td, som selector. ”Border” som min property, og 1px solid black som min value.

Collapsed Borders

I ovenstående tabel er der luft imellem cellerne. Det kan man nemt fjerne, ved at opsætte en border-collapse. Den slår ligesom cellerne sammen (Se eksempel nedenfor).

Eksempel 3: Tabel med collapsed borders.

```
table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
}
```

Guitarist	Band	Guitar	Forstæker
Jimi Page	Led Zeppelin	Gibson Les Paul	Marshall
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb
Angus Young	AC/DC	Gibson SG	Marshall

Cell padding (Luft mellem indhold i celler og borders)

Cell padding angiver mellemrummet imellem celleindholdet og dets grænser. Hvis man ikke angiver en cell padding, så vises tabelcellerne uden padding som standard. I eksempel 3 nedenfor kan man se, hvordan indholdet i de enkelte celler har fået mere luft ud til kanten/grænserne.

I dette eksempel har td og th været mine selectors, dernæst har min property været padding, da jeg har ønsket mere luft i cellerne, og min value er de 15px.

Eksempel 4: Cell padding – Luft imellem indhold i cellerne og borders.

```
td, th {
    padding: 15px;
}
```

Guitarist	Band	Guitar	Forstæker
Jimi Page	Led Zeppelin	Gibson Les Paul	Marshall
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb
Angus Young	AC/DC	Gibson SG	Marshall

Left-align headings (Styling af overskrifterne i tabellen (th))

Indtil videre har vi ikke rigtig gjort noget for at style vores table heading (th). Det øverste felt, hvor der står guitarist, band, guitar, forstærker.

Som standardindstilling er table headeren altid skrevet med kraftig skrifttype og centeret i midten af cellen (Se eksempel 4 lige oven over).

Men vi har mulighed for at ændre designet på vores table header - Vi kan f.eks. left-aligne vores table header, således at den står ude til venstre i cellen, ligesom de andre celler.

Det kan være hensigtsmæssigt at left-aligne table headeren, da det vil få tabellen til at se mere overskuelig og enkelt ud, da overskriften i tabellen vil flugte med indholdet i de andre celler.

```
table {
    width: 600px;
}

table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
}

td, th {
    padding: 15px;
}

th {
    text-align: left;
}
```

Guitarist	Band	Guitar	Forstæker
Jimi Page	Led Zeppelin	Gibson Les Paul	Marshall
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb
Angus Young	AC/DC	Gibson SG	Marshall

Eksempel 5: Left align heading.

Her kan vi tydeligt se, at table headeren er blevet left-aligned. Nu flugter både th og td pænt hinanden, hvilket for tabellen til at se mere overskuelig ud.

Border Spacing (Afstand imellem cellerne)

Border spacing er med til at definere afstanden imellem cellerne. For at skabe afstand imellem cellerne skal vi bruge den property der hedder ”border spacing”

VIGTIG: Hvis tabellen har collapsed borders, så har border spacing ingen effekt. Man er nødt til at fjerne collapsed borders før man kan lave border spacing – Som vi lærte tidligere, så fjerner collapsed borders luften imellem cellerne, hvilket er netop det vi ønsker at skabe med, border spacing.

I koden til venstre har jeg kommenteret /*border colllapse*/ - For at kunne skabe effekten med border spacing.

Eksempel 6: border spacing.

```
3  table {
4      width: 600px;
5      border-spacing: 5px;
6  }
7
8  table, th, td {
9      border: 1px solid black;
10     /*border-collapse: collapse;*/
11 }
12
13 td, th {
14     padding: 15px;
15 }
16
17 th {
18     text-align: left;
19 }
20
```

Guitarist	Band	Guitar	Forstæker
Jimi Page	Led Zeppelin	Gibson Les Paul	Marshall
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb
Angus Young	AC/DC	Gibson SG	Marshall

Colspan (Celle som spænder over mere end en kolonne)

Man kan få en celle til at spænde mere end en kolonne. Det gør vi ved at bruge attributten colspan. Det er en smart funktion, da vi kan skabe flere celler under den samme table header (th). I eksemplet med guitarister er det en smart funktion, da de fleste guitarister spiller på flere forskellige typer af forstærkere.

Måden dette gøres er ved at gå ind i html´en – Finde det tag, hvor man ønsker at skabe colspan, i dette tilfælde vil det være <th colspan="2"> Forstæker</th>

Bemærk i attribut valuen (Det der står i gåseøjne), står der 2, det angiver hvor man celler man ønsker at skabe under en kolonne. Her kunne jeg også have skrevet 3, hvis guitaristerne havde 3 typer, forstærker de spillede på.

Eksempel 7: Celle som spænder mere end en kolonne.

Guitarist	Band	Guitar	Forstæker	
Jimmy Page	Led Zeppelin	Gibson Les Paul	Marshall	Orange Rockerverb 50
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb	Hiwatt DR-103
Angus Young	AC/DC	Gibson SG	Marshall	Marshall 1959

```
<tr>
    <th> Guitarist</th>
    <th> Band </th>
    <th> Guitar</th>
    <th colspan="2"> Forstæker</th>
</tr>
```

Til venstre kan du se koden i html, og hvordan attributten, colspan, er integreret i det tag, hvor jeg ønsker flere celler i en kolonne.

Cells that span many rows (Celle som spænder over mere end en række)

Man kan også skabe en celle, der strækker sig over to rækker. Det minder meget om det vi gjorde lige før med kolonnerne. Dog har denne funktion en anden attribut som hedder ”rowspan”.

For at danne en celle, som strækker sig over mere end en række skal vi bruge attributten rowspan.

Måden dette gøres er ved at gå ind html´en – Finde det tag, hvor man ønsker at skabe rowspan, i dette tilfælde vil det være <th colspan="2"> band</th>
Bemærk i attribut valuen (Det der står i gåseøjn), står der 2, det angiver hvor man celler man ønsker at skabe under en række. Her kunne jeg også have skrevet 3, hvis Jimmi Page har spillet i tre forskellige bands.

Eksempel 8: Celle som spænder mere end en række.

Guitarist	Jimmy Page
Band	Led Zeppelin
	The Yardbirds

```
<table style="width:800px">

  <tr>
    <th>Guitarist</th>
    <td>Jimmy Page</td>
  </tr>
  <tr>
    <th rowspan="2">Band</th>
    <td>Led Zeppelin</td>
  </tr>
  <tr>
    <td>The Yardbirds</td>
  </tr>
</table>
```

Til venstre kan du se koden i html, og hvordan attributten, rowspan, er integreret i det tag, hvor vi ønsker flere celler i en kolonne.

Adding a Caption (Tilføj billedtekst til din tabel)

Vi har mulighed for at skabe en overskrift/tekst til din tabel. For at gøre dette skal vi have fat i tagget <caption>
Her skal vi tilbage og skrive i vores html dokument, da caption ikke er en del af CSS-sproget.
Vigtigt: Caption tagget skal indsættes direkte efter vores table tag. På den måde forklarer vi HTML, at det er en overskrift/tabeltekst for hele tabellen.

Guitarister og Udstyr

Guitarist	**Band**	**Guitar**	**Forstæker**	
Jimmy Page	Led Zeppelin	Gibson Les Paul	Marshall	Orange Rockerverb 50
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb	Hiwatt DR-103
Angus Young	AC/DC	Gibson SG	Marshall	Marshall 1959

```
<table>
    <caption> Guitarister og Udstyr</caption>
    <tr>
        <th> Guitarist</th>
        <th> Band </th>
        <th> Guitar</th>
        <th colspan="2"> Forstæker</th>
    </tr>

    <tr>
        <td> Jimmy Page</td>
        <td> Led Zeppelin</td>
        <td> Gibson Les Paul</td>
        <td> Marshall </td>
        <td> Orange Rockerverb 50 </td>
    </tr>

    <tr>
        <td> David Gilmour</td>
        <td> Pink Floyd</td>
        <td> Fender Stratocaster</td>
        <td> Fender twin Rewerb</td>
        <td> Hiwatt DR-103 </td>
    </tr>

    <tr>
        <td> Angus Young</td>
        <td> AC/DC</td>
        <td> Gibson SG</td>
        <td> Marshall</td>
        <td> Marshall 1959</td>
    </tr>


</table>
```

Eksempel 9: Tilføj en overskrift/tabeltekst til din tabel.

På siden ovenover kan du se, hvordan eksempel 9 ser ud i et browservindue.

Bemærk: Tabelteksten ”Guitarister og Udstyr”.

Til venstre kan du se selve koden.

Bemærk: Caption-tagget er indsat lige efter table-tagget.

A special style for one table (Tilføj en unik style til en specifik tabel)

Tabeller kan styles på mange forskellige måder, og med meget forskellige udtryk. Vi kan tilføje en unik style til en konkret tabel – Dette gør vi ved at bruge attributten id.
På den måde kan vi fortælle html, at det er et specifikt element vi ønsker at style.

For at vise forskellen på tabellerne, har jeg kopieret min første tabel. Den anden tabel som jeg nu har fået, har jeg givet attributten id =”t01”.

Derefter går jeg ind i den <table> jeg lige har kopieret og tilfører attributten id. Sådan her
<table id="t01">
Derefter går vi ind i vores CSS og skriver table#t01 – Dermed har vi angivet at det kun er den kopieret tabel vi ønsker at style.

Den kopierede tabel (t01) har jeg givet en anden baggrundsfarve. Det har jeg gjort ved at bruge propertyen background-color, og derefter givet den valuen #f1f1c1; Hvilket giver en slags gul baggrundsfarve på tabellen.

Eksempel 10: Brug af id-attributten til at style en bestemt tabel.

Guitarister og Udstyr

Guitarist	**Band**	**Guitar**	**Forstæker**	
Jimmy Page	Led Zeppelin	Gibson Les Paul	Marshall	Orange Rockerverb 50
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb	Hiwatt DR-103
Angus Young	AC/DC	Gibson SG	Marshall	Marshall 1959

Først angiver jeg id inde i HTML

```
<table id="t01">
```

Herefter går jeg ind i CSS, og bruger tagget #t01.

```
#t01 {
    background-color: #f1f1c1;
}
```

Add more styles (Giv den gas med forskellige styles)

Eksempel 11 – Brug af forskellige styles.

Guitarister og Udstyr				
Guitarist	**Band**	**Guitar**	**Forstæker**	
Jimmy Page	Led Zeppelin	Gibson Les Paul	Marshall	Orange Rockerverb 50
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb	Hiwatt DR-103
Angus Young	AC/DC	Gibson SG	Marshall	Marshall 1959

```
#t02 th {

    background-color: black;
    color: white;
    }
#t02 tr:nth-child(even) {
    background-color: #eee;

}

#t02 tr:nth-child(odd) {
    background-color: #fff;
}

#tabeltekst {
    color:#bc1c1c;
    background-color: #eee;
    font-family: sans-serif;
    font-size: 2em;
    border: 1px solid black;
    padding: 0.5em;
}
```

Her til sidst har jeg kopieret min tabel igen.
Bemærk: Det er den samme html kode, som jeg også brugte til den gule tabel, det eneste jeg har ændret på, er CSS´en.

Den nye tabel har jeg kaldt #t02.

Først var jeg inde og style på table headeren, den har jeg givet en sort baggrundsfarve og hvid skrift.

Herefter var jeg inde og give de forskellige tabelrækker farver. Jeg opdelte dem i lige og ulige rækker. Dette har jeg gjort ved at vælge selectoren tr:nth-child (even) og tr:nth-child (odd) for henholdsvis at farve de lige rækker i en farve, og de ulige rækker i anden farve.

Til sidst var jeg inde og style min tabeltekst. Først startede jeg med at finde en mørkerød farve til teksten (#bc1c1;) – Herefter brugte jeg den samme grå baggrundsfarve, der også er anvendt nede i tabellen (#eee;)

Derefter valgte jeg en anden skrifttype font-family: sans serif; - Dernæst skulle skriften være lidt større, det gjorde jeg, ved at sætte font-size til 2em;

Til sidst vil jeg have en grænse/border rundt om min tabeltekst for at ramme den bedre ind. Det gjorde jeg ved at bruge ”border” 1px solid black.

Afslutningsvis skabte jeg noget afstand imellem tekstindholdet i tabelteksten og grænse/borderen, ved at sige padding: 0.5em;

Så efter en omgang styling i CSS kom min tabel til at se væsentligt anderledes ud end da jeg startede. Dette beviser, at igennem styling i CSS, kan vi ændre designet fuldkommen på vores tabel.

Tabellen vist udelukkende med HTML – Ingen CSS overhovedet.

Guitarister og Udstyr

Guitarist	Band	Guitar	Forstæker	
Jimmy Page	Led Zeppelin	Gibson Les Paul	Marshall	Orange Rockerverb 50
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb	Hiwatt DR-103
Angus Young	AC/DC	Gibson SG	Marshall	Marshall 1959

Samme Tabel med nøjagtig samme HTML – Bare stylet med CSS.

Guitarister og Udstyr

Guitarist	Band	Guitar	Forstæker	
Jimmy Page	Led Zeppelin	Gibson Les Paul	Marshall	Orange Rockerverb 50
David Gilmour	Pink Floyd	Fender Stratocaster	Fender twin Rewerb	Hiwatt DR-103
Angus Young	AC/DC	Gibson SG	Marshall	Marshall 1959

Kilder

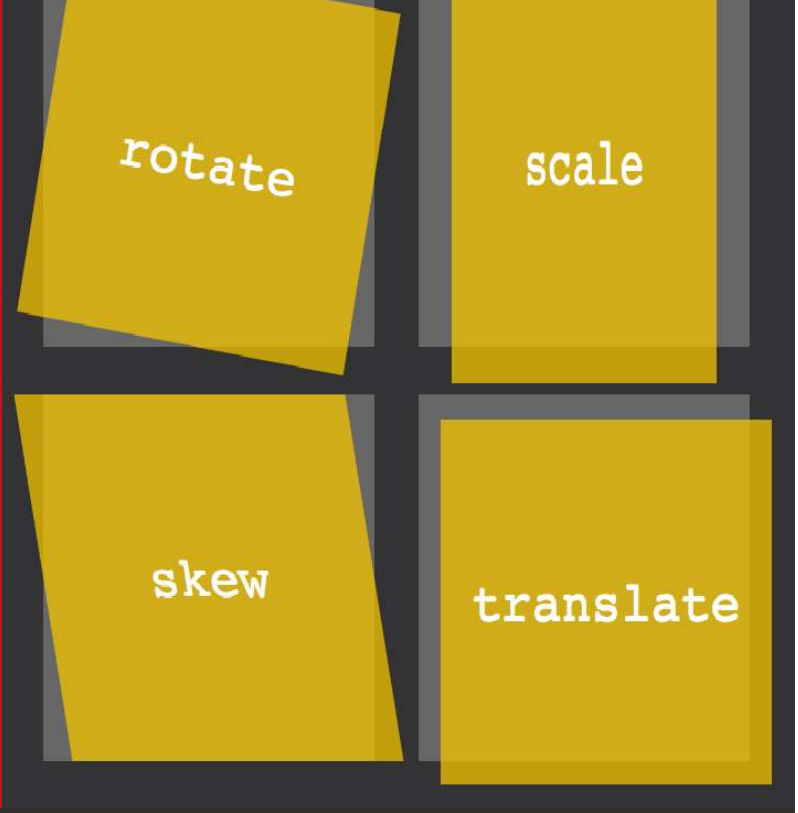
Herunder har jeg angivet en kildeliste, og nogle af de kilder som jeg har brugt i forbindelse med udarbejdelsen af dette dokument. Der er nogle gode steder at finde mere information omkring HTML tabeller og hvordan man styler dem.

Artikler og eksempler:
https://www.w3schools.com/html/html_tables.asp
https://developer.mozilla.org/en-US/docs/Learn/CSS/Styling_boxes/Styling_tables
https://beamtic.dk/tabeller-i-html

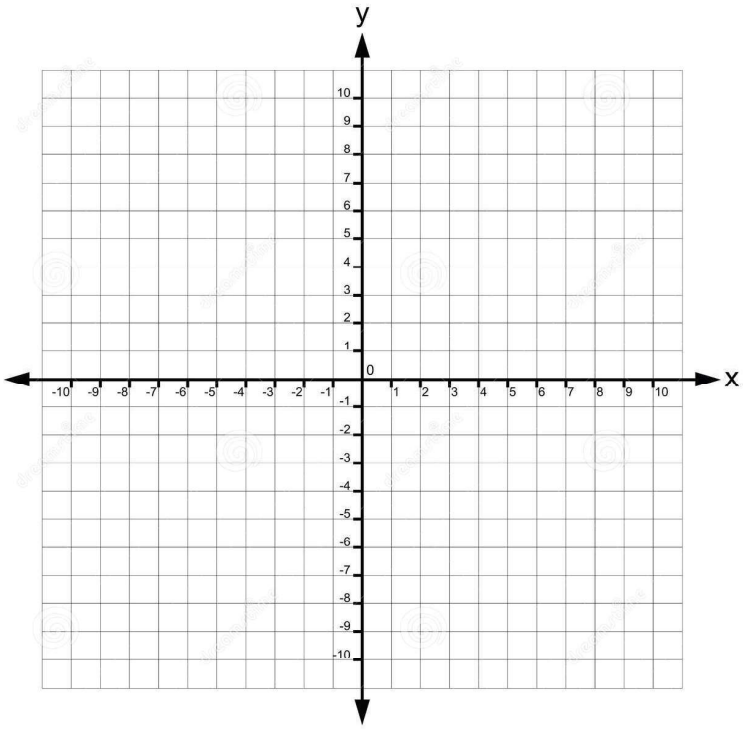
Video:
https://www.lynda.com/HTML-tutorials/Displaying-data/365960/378906-4.html
https://www.youtube.com/watch?v=dK27jWKtJxs

TRANSFORMS

Ændring af et element ved dets koordinatsystem.



2D



TRANSLATION

Flytter et element til en anden position, på de X og Y axis.

- translateX( ) - flytter elementet vandret
- translateY( ) - flytter elementet lodret
- translate( ) - indtast to values, til X og til Y, og elementet vil flytter både vandret og lodret

ROTATION

Roterer et element.v

- rotate(__deg) - roterer et element med uret eller mod uret til en given grad. Minus grader roterer mod uret.

SKEW

Til forvride eller contort et element.

- skewX( ) - contorts på X aske.
- skewY( ) - contorts på Y aske.
- skew( ) - contorts begge X og Y aske.
- transform-origin - anchor point hvor contortion komme fra.

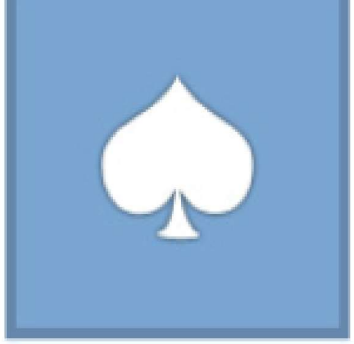
SCALE

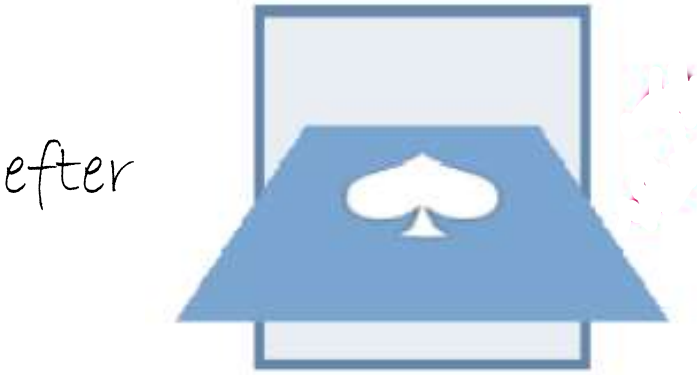
Gør et element større eller mindre inden for angivne parametre for højde og bredde

- scale(_X_ , _Y_) - vil gør et element større eller mindre på sammen til.
- Først nummer lave elementet bredere.
- Andet nummer lave elementet højere.

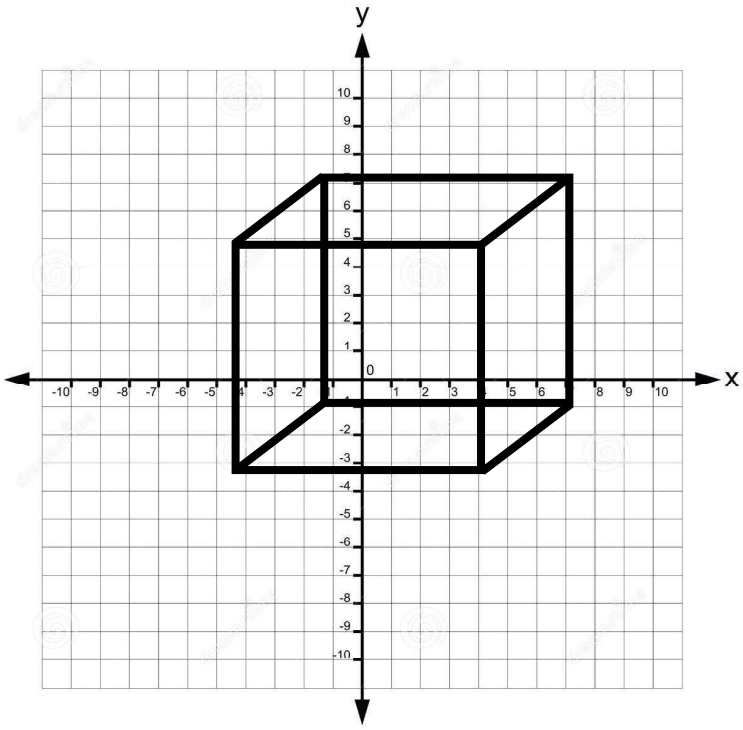
TRANSFORMS

Ændring af et element ved dets koordinatsystem.





3D



TRANSFORM PROPERTIES

- transform : transformation til et element i 2D eller 3D
- transform-origin : giver muligheder til ændre positionen af elementer
- transform-style : skift til "preserve-3d" til laver element i 3D

PERSPECTIVE PROPERTIES

- perspective : hvordan en element i 3D ser ud
- perspective-origin : hvilken side det forsvindingspunkt er og bruger X og Y værdier, som udtrykkes procent fra 0% til 100%

Perspective skal defineres på et containerelement, og ikke det element, som skal være transformed.

3D TRANSFORM METHODS

- translate3d (x,y,z)
- translateX (x)
- translateY (y)
- translateZ (z)

- scaled3d (x,y,z)
- scaleX (x)
- scaleY (y)

- rotate3d (x,y,z,angle)
- rotateX (angle)
- rotateY (angle)
- rotateZ (angle)

matrix3d : skaber 3D transformation med bruge af en matrix af 16 values

EKSEMPEL

```
.rotate3D {
    margin: 100px;
    transform-style: preserve-3d;
    perspective: 360px;
    perspective-origin: left;
}
```

CSS Flexbox

Hvad er CSS Flexbox?

CSS Flexbox er et ”fleksibelt layout modul, der gør det nemmere at designe responsivt layout struktur uden at bruge float og positionering”.

Flexbox er supportet af Google Chrome, Internet Explorer, Mozilla Firefox, Safari og Opera browserne.

Det mest vigtige element ved brug af CSS Flexbox er ”flex containeren”, som ved brug af CSS opsætter bruget af CSS flexbox og gives egenskabet ”display: flex;” for at vises på sin HTML side. Ydermere kan den indeholde følgende egenskaber:

- flex-direction: Definerer hvilken retning ”containeren” opdeler ”flex items”
- flex-wrap: Definerer hvis ”flex items” skal være wrappet (skalerer indholdet når man skalerer browserens vindue størrelse eller ej. Ydermere hvilken rækkefølge flex items er sat i.
- flex-flow: Er en shorthand egenskab der sætter indstillingerne for både flex-direction og flex-wrap.
- justify-content: Er brugt til at sorterer hvordan indholdet placeres, fx hvis det skal være centreret eller ej.
- align-items: Viser hvor på siden at det valgte element skal vises på siden, kan kun bruges vertikalt.
- align-content: Er brugt til at flex items er placeret korrekt udenfor hinanden

Flex Item elementer

” The direct child elements of a flex container automatically becomes flexible (flex) items.”

- Order: Sorterer flex items og gøre det muligt at lave forskellige rækkefølger

- Flex-grow: Specificerer hvor meget et flex item vil gro i forhold til de andre flex items
- Flex-shrink: Specificerer hvor meget et flex item vil skrumpe i forhold til de andre flex items
- Flex-basis: Specificerer længden af en flex item
- Flex: Shorthand af flex-grow, flex-shrink og flex-basis elementerne
- Align-self: Specificerer placering af det valgte item inden i ”the flexible container”

Guide til CSS Flexbox

Af Nikolas Ibfelt og Daniel Mann Andersen

Indholdsfortegnelse

1. Indledning til Flexbox
2. Eksempler af HTML + CSS Flexbox design med forklaringer.
3. Konklusion

Indledning til Flexbox

Flexbox er et værktøj der er lavet til at gøre placeringen af elementer med relation til hinanden nemmere. Tidligere har vi lært om værktøjet “Float” hvor vi kan flytte individuelle elementer rundt på en side, men dette kræver ikke kun en masse kodning, men også tålmodighed. Derfor er Flexbox kommet til verdenen som et værktøj til netop at gøre denne placering nemmere, og kræve mindre arbejde og kode for webdesigneren.

Med Flexbox arbejder du primært med 2 elementer. Det første er **Flex-container** som vil være hvad der indeholder vores **Flex-items**, som er vores andet element. Vi kan altså sige at vores Flex-container er vores arbejdsplads, imens vores Flex-items er vores indhold der skal udfylde vores arbejdsplads. I en Flex-container kan vi bestemme forholdene for vores Flex-items. Hvor i vores Flex-container skal de begynde og slutte? I hvilken rækkefølge? Hvor meget plads skal de være på?

Vores Flex-items kan være alt hvad hjertet begærer om det er tekst, billeder, WordArt eller interaktive knapper, så kan vi tilegne vores individuelle Flex-items denne værdi.

Ved mere garvet brug af CSS Flexbox vil man have mulighed for at bruge dette værktøj til eventuel søgefunktioner på hjemmesider, tilmeldingsformularer og meget mere.

Eksempler af HTML + CSS Flexbox design med forklaringer

Vi har samlet nogle eksempler sammen til et stort eksempel, som både skal vise hvad CSS Flexbox kan gøre i simple termer og også hvordan man kan bruge den til at opsætte websider. Vores første eksempel viser fx. i basis hvad CSS Flexbox er om og hvordan det fungerer. Mens de sidste to eksempler viser lidt mere avancerede tilgangsmåder, men stadig holdt på et simpelt læringsniveau.

```
<div class="eks1-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
</div>
```

Det mest grundlæggende af CSS Flexbox er opstillingen og forholdene mellem vores flex-container (det arbejdsrum vi har reserveret) og vores flex-items (vores indhold, som skal udfylde pladsen i arbejdsrummet). Flex-containeren specificeres som en ‘class’, som vi via CSS kan redigere. Flex-items specificeres modsat efter brug af semantiske tags og de indstillinger som gives i CSS. Det skal noteres at CSS Flexbox tilføjer forskellige modificérbare indstillinger reserveret både til flex-containeren og til flex-items. Som ses i eksemplet er containeren nu blevet til en class vi kan ændre, mens vi har 8 div’er som flex items.

```
.eks1-container {
  display: flex;
  flex-direction: row;
    flex-wrap: wrap;
    background-color: DodgerBlue;
    height: 700px;
    width: 100%;
    justify-content: space-around;
}

.eks1-container div {
  background-color: #f1f1f1;
  width: 300px;
  margin: 10px;
  line-height: 300px;
  font-size: 30px;
    text-align: center;
}
```

For at bygge videre på vores 8 div'er og deres indhold i ft. vores container skal vi nu til at bruge CSS delen af CSS Flexbox. Først udarbejder vi vores indstillinger for vores arbejdsrum (containeren). Her skriver vi display: flex; for at aktiverer CSS Flexbox, flex-direction til row; og flex-wrap: wrap;. flex-direction sætter vores startpunkt og hvilken retning CSS flexbox skal følge, mens wrap sørger for at vi kan udvide vores arbejdsrum til at have plads til vores flex items. justify-content er den sidste indstilling, som har direkte kontrol over vores layout, den er her sat til space-around; som automatisk aligner vores items med mellemrum mellem containers kanter og vores flex items. Derefter kan vi modificerer vores flex items og vi gør sådan at vores text er i midten med text-align.

Konklusion

Vi har her i denne lille guide gået igennem nogle af de mest basale måder at bruge CSS Flexbox på. Vi har vist hvordan man opstiller CSS Flexbox i sin software og vi har vist en lille del af de indstillinger vi kan give både flex-containeren og flex-items.

Flex Container

Container

Flex-Item

Main axis

Cross Axis

Flex Container Properties

flex-direction

flex-wrap

flex-flow

justify-content

align-items

align-content

Flex Items Properties

order

flex-grow

flex-shrink

flex-basis

flex

align-self

Værdier

flex-direction:

column, column-reverse, row, row-reverse

flex-wrap:

nowrap, wrap, wrap-reverse

flex-flow:

column, column-reverse, nowrap, row, row-reverse, wrap, wrap-reverse

justify-content:

center, flex-end, flex-start, space-around, space-between

align-items:

baseline, center, flex-end, flex-start, stretch

align-content

center, flex-end, flex-start, space-around, space-between

order:

style= ”order: 1” eksempel

flex-grow:

style= “flex-grow: 2” eksempel

flex-shrink:

style= “flex-shrink: 2” eksempel

flex-basis:

style= “flex-basis: 100px” eksempel

flex:

style= “flex: 0 0 200px” eksempel

align-self

auto, baseline, center, flex-end, flex-start, stretch

Outline

Outline er en betegnelse der beskriver et html dokuments omrids. Her er der tale om den struktur som html skaber i et dokument. Dette sker gennem nesting af elementer inde i andre elementer.

En dokument outline starter ved bodyelementet, som kan betegnes som roden til alt indholdet på websiden. Fra dette bodyelement kan vi bygge videre på outlinen, gennem nesting af de forskellige sektionselementer, dertil kan man blive ved med at neste sektioner. Alt dette udgør dokumentets outline.

Bemærk hvordan skriftstørrelsen ændrer sig, selvom alle overskrifter er h1. Browseren styre selv skriftstørrelsen baseret på, hvor overskriften befinder sig i dokumentets outline.

```
<body>
    <h1>Eksemple side</h1>
    <nav>
        <h1>Navigations linje</h1>
    </nav>
    <section>
        <h1>Øverste lag</h1>
        <section>
            <h1>Hovedindhold</h1>
            <section>
                <h1>Fremhævet indhold</h1>
            </section>
            <section>
                <h1>Artikler</h1>
                <article>
                    <h1>Artikel 1</h1>
                </article>
            </section>
        </section>
        <aside>
            <h1>Relateret indhold</h1>
            <section>
                <h1>HTML referance</h1>
            </section>
            <section>
                <h1>Bogliste</h1>
                <article>
                    <h1>Bog 1</h1>
                </article>
            </section>
        </aside>
    </section>
</body>
```

Eksemple side

Navigations linje

Øverste lag

Hovedindhold

Fremhævet indhold

Artikler

Artikel 1

Relateret indhold

HTML referance

Bogliste

Bog 1

Figur 1,1: Her ses hvordan en outline i htmlstrukturen afspejler sig i en browser.

Dokument headings

Headings kan også bruges som outline til et dokument. Udover *h1*, som er anvendt i figur 1,1, findes der også *h2, h3… og h6.* Disse forskellige headings indikere hvor i dokuments outline de hører til. Man kan derfor på samme måde lave en outline ved kun at bruge headings. Hvordan man gør, ses på figur 1,2. Resultatet i en browser vil være identisk til figur 1,1. Dette er dog ikke anbefalet at bruge headings, da de ikke indikere nogen semantik mm.

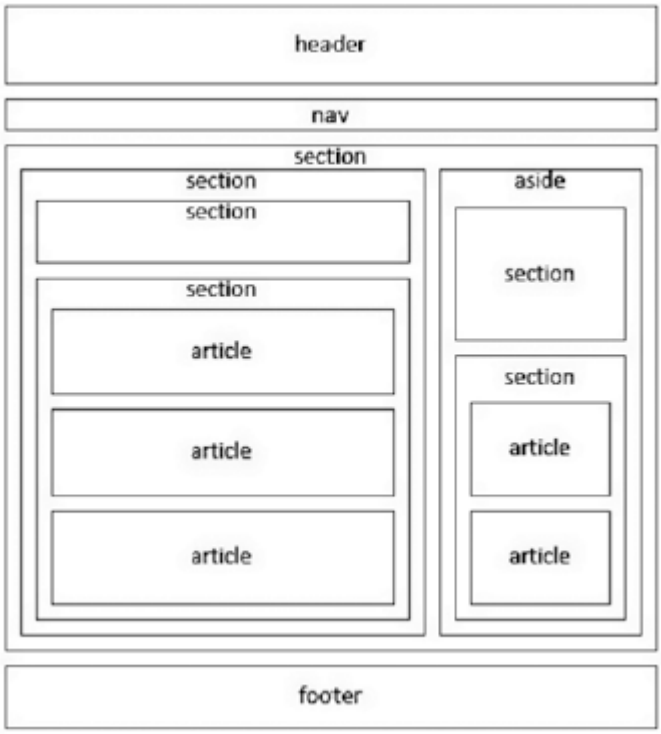
```
<body>
    <h1>My Sample Page</h1>
        <h2>Navigation</h2>
        <h2>Top-level</h2>
            <h3>Main content</h3>
                <h4>Featured content</h4>
                <h4>Articles</h4>
                    <h5>Article 1</h5>
            <h3>Related content</h3>
                <h4>HTML Reference</h4>
                <h4>Book list</h4>
                    <h5>Book 1</h5>
</body>
```

Figur 1,2: Her ses et html eksempel på hvordan man kan skabe en outline ved kun at anvende headere.

Header og footer

Headere og footere bliver oftest brugt som top og bund i bodyen. De indikerer dertil henholdsvis, introduktion og konklusion af dokumentet. Disse to elementer bidrager ikke til dokumentets outline, da de ikke laver nogen ny sektion. De inddeler kun indhold i den sektion de er placeret i. Dertil kan de også placeres i, eks., et sectionelement, hvor de vil have funktionen som henholdsvis introduktion og konklusion.

Planlægning af websidens layout



Figur 1,3: Visuelt eksempel på websides outline, dertil struktur og nesting.

Her ser vi et visuelt eksempel på hvordan elementer kan nestes inde i hinanden, og den skruktur det medfører.

Sectioning roots

Sectioning roots er html elementer der har deres egen outline, men de udgør ikke noget af dokumentets outline. Der findes følgende*: blockqoute, details, figure, fieldset og td*; de to sidste vil ikke blive forklaret da de først bliver beskrevet i henholdsvis kapitel 8 og 6[1].

Blockqoute

En blockqoute angiver at man har citeret en anden kilde. Dertil skal man angive den kilde i *cite* attributtet, dette attribut kan ikke ses på siden, med mindre man bruger *cite* tagget men det kan man læse mere om i kap 5.

```
<h1>Blockqoute</h1>
<blockquote cite="https://www.dr.dk/">
    Dette er et citat fra DR
</blockquote>
```

Blockqoute

Dette er et citat fra DR

Figur 2,1: Her ses en blockqoute i html og i en browser.

```
<blockquote cite="http://www.worldwildlife.org/who/index.html">
    <h1>For 50 years</h1>
    <p>WWF has been protecting the future of nature. The world's leading
    conservation organization, WWF works in 100 countries and is supported
    by 1.2 million members in the United States and close to 5 million
    globally.
    </p>
</blockquote>
```

Figur 2,2: Et eksempel på hvordan man kan placere outline elementer i en blockqoute.

Her ses et mere repræsentativt eksempel på en blockqoute. Det der er vigtigt at notere her er at h1 ikke ingår i dokumentets outline. Så alt hvad der er inde i blockquoten har ingen effekt op dokumentets outline.

Details

En 'detail' består af flere elementer, først skal vi bruge *detail* tagget til at definere at vi vil lave en detail. Dertil kan man vælge om ens detail skal starte åben eller lukket, dette kan man styre med attributterne *open* og *closed*. Dernæst skal man bruge et *summary* tag inde i sin detail, dette vil være den tekst der opsummere hvad detailen indeholder.

```
<details open>
    <summary>Dette er det kolapsbare tekst</summary>
    <h1>Detaljer</h1>
    <p>Dette er de kollapsede detaljer</p>
</details>
```

Figur 2,3: html koden bag en detail. I dette tilfælde har vi en åben detail

▶ Dette er det kolapsbare tekst

▼ Dette er det kolapsbare tekst

Detaljer

Dette er de kollapsede detaljer

Figur 2,4: Her ses detailen i begge stadiger, åbent og lukket.

Bemærk hvordan pilen peger på teksten når der lukket, og hvordan den peger ned, når den er åben.

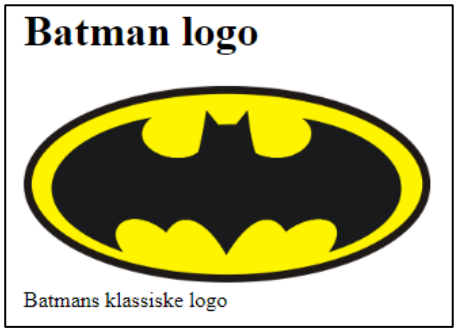
[1] Bog: Pro HTML5 with CSS, JavaScript and multimedia.

Figure

Figure bruges til at gruppere indhold som er selvstændigt, så hvis det fjernes eller flyttes, vil det ikke have indflydelse på resten af dokumentets flow.

```
<figure>
    <h1>Batman logo</h1>
    <img src="batman-logos-png-4.png" alt="bamtans
    logo" width="300px>">
    <figcaption>Batmans klassiske logo</figcaption>
</figure>
```

Figur 2,5: figure html kode.



Figur 2,6: Et figure element med Batmans logo og beskrivende tekst.

For at lave en figure skal man bruge *figure* tagget, dertil skal man angive en kilde til billedet, i figuren, med *src* attributtet. Man kan også vælge at tilføje et *figcaption* attribut. Denne fungerer som en undertekst til figuren, som set på eksempel 2,6.

Igen skal man notere at alt indhold inde i elementet bidrager ikke noget til dokumentets outline, kun den der finder sted inde i elementet.

In-line frames

In-line frames bruges til at indlejre indhold, eller en hjemmeside, i et andet dokument; forklaring følger.

I dette første eksempel har vi indlejret hele DRs hjemmeside på vores egen. Dette har vi gjort ved bare at indtaste DRs URL som *src* til vores *iframe.* Her kan man altså frit bruge hele siden og dets funktioner igennem dette vindue. Nogle hjemmesider har en *no embedding* politik, der forhindre eksterne sider, som denne, at indlejre deres indhold. I et tilfælde som dette, vil det iframe vinduet bare vise en blank kasse.

Der findes nogle få browsere der ikke understøtter *iframes*, i dette tilfæde vil der blive vist den tekst vi har skrevet i *p* tagget, og det er det eneste formål teksten har.

Man kan også indstille højde og bredde på det viste vindue, ved at bruge *height* og *width* attributterne.

```
<iframe src="https://www.dr.dk/">
    <p>Your browser does not support iframes</p>
</iframe>
```

Figur 3,1: Simpel in-line frame med DR.

```
<iframe width="560" height="315"
src="https://www.youtube.com/embed/1sJLey8hUyY"
frameborder="0" allow="autoplay; encrypted-media"
allowfullscreen></iframe>
```

Figur 3,2: En indlejret Youtube video.

Der findes mange forskellige brug til *iframe.* Det største brug af *iframe* er til marketing. Det elemenet gør er at det laver et 'hul' siden, hvor et affiliate firma sender reklamer igennem det hul på din hjemmeside. Et andet brug er hvis man eksempelvis vil ligge en youtube video ind på ens side, som et i eksemplet. For at gøre dette skal man trykke *del* og derefter kopiere koden fra *embed*, hvor man til sidst skal indsætte hele koden i sit dokument. Man kan også indlejre et kort på sin side, som eks. Google maps.

Listing elementer

I html kan man lave en række forskellige lister, både *usorterede, sorterede* og *depiction lists.*

En usorteret liste opstiller en punktliste, denne kan bruges hvis indholdet ikke har en bestemt rækkefølge, som i emneoversigten på figur 4,1.

```
<h1>Indholds emner</h1>
<ul>
    <li>U-sorterede lister</li>
    <li>Sortede lister</li>
    <li>Deskriptions lister</li>
</ul>
```

Indholds emner

- U-sorterede lister
- Sortede lister
- Deskriptions lister

Figur 4,1: En usorteret liste i html og den browser.

```
<ol>
    <li>U-sorterede lister</li>
    <li>Sortede lister</li>
    <li>Deskriptions lister</li>
</ol>
```

Indholds rækkefølge

1. U-sorterede lister
2. Sortede lister
3. Deskriptions lister

Figur 4,2: En sorteret liste, hvor hver punk har et nummer.

En sorteret liste bruges opstille noget i en nummeret rækkefølge. Man kan ændre symbolerne i listen, ved at bruge *type* attributtet, hvor man bla. Kan bruge romertal mm. Det er også muligt at state listen med et andet tal end 1, hvis man bruger attributtet *start*. Dertil skal du tilføje en værdi, som eks. 50, hvor listen så vil starte fra 50.

Denne liste kan bruges som en glose liste over begreber, men kan også bruges til at give korte bestrivelser på eks. en liste af bøger. Den kan også bruges til at indedele eks. fodboldhold op i deres roller, som set i bogen[2].

Et begreb
    En forklaring på begrebet
Et andet begreb
    En forklaring til det andet begreb

```
<dl>
    <dt>Et begreb</dt>
    <dd>En forklaring på begrebet</dd>
    <dt>Et andet begreb</dt>
    <dd>En forklaring til det andet begreb</dd>
</dl>
```

Figur 4,3: En description list, vist som html og i en browser.

[2] Bog: Pro HTML5 with CSS, JavaScript and multimedia – side; 75.

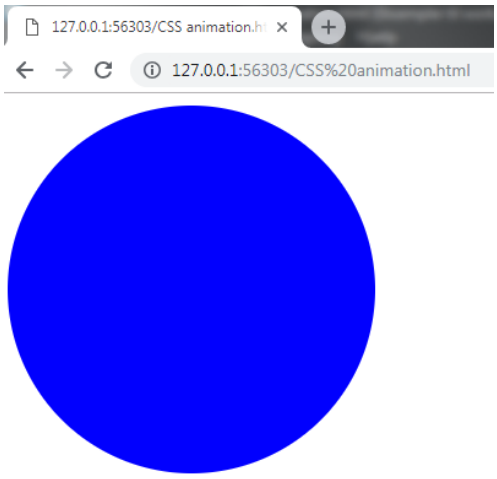
Animation i CSS

Animation i CSS foregår gennem det man kalder *keyframes*, og kan anvendes til at ændre form, farve og størrelse. Der skal minimum 2 *keyframes* til, for at skabe en animation i CSS, en start- og slutframe, disse angives med hhv. 0% og 100%. Man kan også bruge disses alias, *from* (0%) og *to* (100%). Disse frames er en måde at angive en tilstand til dit element du vil animere.

Hvis vi ser på eksemplet, så har vi en keyframe, som vi har tildelt navnet 'colour'. I denne animation vil vi gerne have at vores røde cirkel skal gå fra at være rød, til grøn, til blå og så rød igen til sidst. Vi har anvendt start- og slutframe ved 0% og 100% som rød, dette kan vi gøre ved at adskille de 2 med et komma. Derefter skal vi have noget at bruge denne animation på, her har vi lavet en cirkel som er rød i et *div* element. Den cirkel har vi så givet animationen 'colour', dertil har vi givet animationen en varighed, altså tiden animationen forløber, i dette tilfælde 10 sekunder. Nu har vi så givet vores html element, *circle*, en animation. Denne er opbygget af 4 keyframes; start, 33%, 66%, og slut. Vi har også givet den et navn, colour, dertil har den en varighed på 10 sekunder.

```
@keyframes colour {
        0%, 100% {
            background-color: red;
        }
        33% {
            background-color: green;
        }
        66% {
        background-color: blue;
    }
}
.circle {
    width: 300px;
    height: 300;
    margin: 10px 0 0 0;
    border-radius: 50%;
    background-color: red;
    animation-name: colour;
    animation-duration: 10s;
}
<div class="circle"></div>
```

Figur 5,1: CSS animation og html med div element



Figur 5,2: Den animerede cirkel som skifter farve, her er den midt i animationen.

Man kan også ligge flere animation på det samme element. I figur 5,3 har vi et element med 2 animationen, der er et par ting som man skal være opmærksom på. Først, så skal man skilde de to animationer ved at bruge et komma. Ved *-iteration-count* har vi kun angivet 1 værdi, det betyder at det kun er den første animation der før værdien. Ved *-delay* og *-direction* blev vi nød til at angive standardværdierne for at kunne angive værdierne til animation_2. På figur 5,3 er der også vist et shorthand for at opnå samme effekt. Disse vil dog ikke blive uddybet i dette dokument.

```
.circle {
    animation-name: animation_1, animation_2;
    animation-duration: 5s, 2s;
    animation-iteration-count: 3;
    animation-delay: 0s, 4s;
    animation-direction: normal, alternate;
}
.circle {
    animation: animation_1 5s 3, animation_2 2s 4s alternate;
```

Figur 5,3: 2 animationer op samme element, og et shorthand for samme effekt.

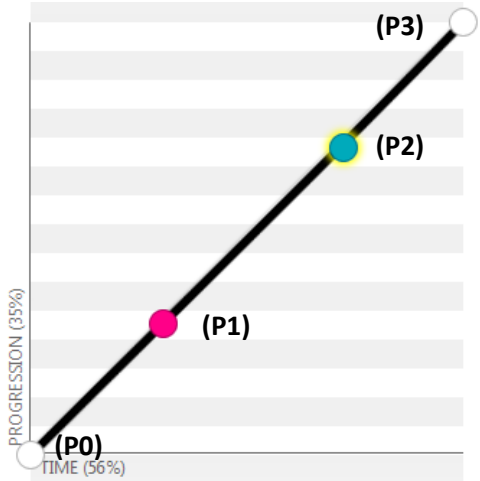
Animations attributter

Der findes 8 attributter som kan anvendt på animationer i CSS. 2 af dem har vi allerede brugt, nemlig 'animation-name' og 'animation-duration'. Nedenfor er en liste over alle attributter og deres funktioner (jeg vil forkorte attributterne ved kun at skrive eks. '–name', i stedet for 'animation-name':

- **-name** – Navnet gives til det sæt af keyframes hvor man har defineret en start- og slutframe og evt. flere værdier som eks. 33% og 66%.
- **-duration** – Som standard er værdien 0s. Denne attribut bruges til at angive tidsrummet animationen foregår over.
- **-timing-funktion** – Som standard er værdien 'ease'. Her kan man angive hvordan overgangen mellem keyframes foregår. Nedenfor er de forskellige værdier man kan bruge:
    - *linear* – Overgangen foregår ligeligt mellem frames.
    - *ease-in* – Overgangen er langsommere i starten.
    - *ease-out* – Overgangen er langsommere i slutningen.
    - *ease-in-out* – Overgangen er langsommere i både starten og slutningen.
    - *ease* – Minder om ease-in-out, men overgangen er ikke ligeså langsom i starten som slutningen.
    - *cubic-bezier* – Her kan man selv definere hvordan overgangen foregår mellem keyframes, mere om dette under 'Cubic bezier'.
- **-delay** – Som standard er værdien 0s. Afgør hvor stor en forsinkelse der er på animationen, dvs. at når animationen igangsættes, så kan selve animationen forsinkes ved brug af denne attribut.
- **-iteration-count** – Som standard kører animationen kun 1 gang. Her kan man vælge hvor mange gange animationen skal gentages. Her kan man sætte værdien til 'infinite', så animationen kører hele tiden.
- **-direction** – Som standard er værdien sat som 'normal'. Dette indikerer hvilken retning animationen kører, her er der et par muligheder:
    - *Normal* – Animationen kører fra 0% til 100%, hvis animationen gentages, vil den starte fra 0% igen.
    - *Alternate* – Animationen kører fra 0% til 100%, men hvis animationen gentages, vil den kører tilbage igen fra 100% til 0%. Hvis den gentages flere gange så gentager denne proces.
    - *Reverse* – Animationen starter fra 100% til 0%, gentagelser vil også starte fra 100% til 0%.
    - *Alternate-reverse* – Først kører animationen baglæns, fra 100% til 0%, men den næste animation vil kører fra 0% til 100%. Ligesom -alternate, bare omvendt.
- **-fill-mode** – Som standard er værdien sat til 'none'. Her kan man specificere hvordan, keyframe tilstandene skal være på html elementet før og efter animationen; Forklaring følger: Her kan man, hvis vi tager udgangspunkt i figur 5,2, afgøre hvilken tilstand cirklen er i før og efter animationen, altså farven. Der findes flere værdier man kan tildele denne attribut:
    - *None* – Før og efter værdierne på cirklen er ikke ændret på cirklen, den forbliver altså rød.
    - *Forwards* – Elementet vil beholde farven fra den sidste keyframe i animationen, efter animationen har kørt.
    - *Backwards* – Hvis man har sat animationen til at køre på en forsinkelse, vil elementet have farven fra den første værdi (0%) i animationen.
    - *Both* – Her får elementet effekten fra både forwards og backwards værdierne.
- **-plays-state** – Standard værdien er sat som 'running'. Man kan her vælge om animationen skal køre (running) eller stå stille (paused).

Cubic Bézier

En cubic bézier en måde at lave sin egen -timing-function. Alle de præ-definerede værdier som liniar og ease følger alle en cubic bézier. For at forstå hvordan man bruger en béizer, skal man forstå en smule af matematikken bag. En cubic bézier er en matematisk formel hvor y-aksen bestemmer funktionen, eller effekten af animationen. x-aksen er perioden hvor animationen finder sted. For at kunne dette i en websammenhæng, skal vi kigge lidt på et par grafer.



Figur 5,4: En cubic bézier over en lineær graf

P0 er det der svarer til 0% af i vores keyframes, og har derfor koordinatet 0,0. Der er ikke sket nogen effekt, og der er heller ikke forløbet nogen tid. P3, som har koordinatet 1,1, svarer til slutningen af animationen, altså 100%. P1 og P2 er dertil med til at bestemme kurven, eller rettere, forløbet animationen kører over.

I eksemplet på figur 5,5 har vi en lille blå kasse, som vi har sat til at flytte sig fra venstre side af skærmen til 300px længer til højre. Denne animation har dog et lidt anderledes forløb, da vi har lavet vores egen bézier. Her kører kassen langsomt i starten, hvor den bygger farten op, og skyder faktisk lidt forbi målet. Til sidst falder den på plads hvor den skal. Følg linket i billedteksten til figur 5,5, for at se bézier der blev brugt.

```
@keyframes kasse {
    0% {
        margin-left: 0;
    }
    100% {
        margin-left: 300px;
    }
}
.kasse {
    width: 100px;
    height: 35px;
    margin-left: 300px;
    border-radius: 10px;
    background-color: lightblue;
    animation-duration: 3s;
    animation-timing-function: cubic-bezier(.82,.11,.81,1.3);
    animation-name: kasse;
    /*
    **Shorthand**
    animation: 3s cubic-bezier(.82,.11,.81,1.3) kasse;
    */
}
```

Figur 5,5: En kasse med en cubic bézier tildelt. (Se dette link til at forstå cubic bézier bedre: http://cubic-bezier.com/#.82,.11,.81,1.3)

Transitions

Transitions har mange ligheder til animationer, hoved forskellen er bare at i en *transition* skal man ikke definere et start- og slutpunkt som i en animation, altså keyframes (0% og 100%). Det man så bruger *transitions* til, er at glatte overgangen mellem to tilstande af et element.

```
.tran {
    width: 200px;
    height: 100px;
    background-color: lightyellow;
    border: 1px solid black;
    transition-property: width, height;
    transition-duration: 1s;
    transition-delay: 0s;
    transition-timing-function: cubic-bezier(.82,.11,.81,1.3)
}
.tran:hover {
    width: 420px;
    height: 150px;
    transition-duration: 3s, 1s;
}
/*Shorthand
    .tran {
    transistion: 3s 0s ease-in, height 1s
}
*/
```

Figur 5,6: Et eksempel på en transition i CSS. Note: .tran klassen er tildelt et div element i html koden, ligesom i de forrige eksempler med animation.

I dette eksempel har vi en kasse med navnet *tran*. Denne kasse har en højde og bredde på hhv. 100px og 200px. Men når vi holder musen over den, har vi sat højden og bredden til at være hhv. 150px og 420px. Hvis der så ikke var nogen *transition* tildelt vil tilstanden ændre sig øjeblikligt. Men ved at tilføje en *transition* glatter vi denne tilstandsændring ud.

I dette eksempel har vi givet kassen 2 *transitioner.* Den første, nede ved *tran:hover,* styrer at når vi muser over kassen vokser den over 3 sekunder på det ene led og 1 sekund på det andet. Men oppe i *.tran* har vi tildelt en *transition* der forløber over 1 sekunder. Det betyder at når vi fjerne musen fra kassen igen, bliver den mindre igen hurtigere end den voksede.

HTML STRUCTURAL ELEMENTS

SECTIONS

SECTION
<section>

- skal være aktuelt
- skal flyde lineært
-organiser indholdet logisk

ARTIKEL
<article>

- inhold, som står alene fra siden
- brugt til syndikeret inhold (som skal bruger igen og igen)
- indholdet er independt fra resten af siden
- eks. blog posts eller komentar

ASIDE
<aside>

- indholde som skal grupe sammen der ikke passe i normal flow af dokument
- eks. sidebar
- thing, der skal placeres til siden, ikke i foran af bruger

NAVIGATION
<nav>

- at gruppere et sæt af links
- eks. menu på en siden
- indhold som er mest links, skal insættes i nav element.

GROUPING

PARAGRAPGH
<p>

- definer et afsnit af tekst, som har en enskelt ide elle tanke

MAIN
<main>

- den primære emne eller formål til et dokument
- identificerer sidens centrale tema
- kan kun være indsæt i body element

DIVISION
* *

- andre grupperingstypes
- mest til organiser elements, billeder, osv

PREFORMATTED
<pre>

- vil blive vist nøjagtigt som det er intastet i HTML

HORIZONTAL RULE
<hr>

- skaber plads imellem to paragraph tags

STRUCTURAL ELEMENTS

En typisk HTML dokument er :

Header
Main
Siderbar (aside)
Navigation menu
Footer

META STRUCTURES

Doctype
Attributes
Head
Body
Meta element
Link
Style
Base

Positioning og media embedding

Af.: Simone Madsen og Steffen Evald

Indholdsfortejnelse:

Indhold

Indholdsfortejnelse: 2
Indledning: 3
Floats 4
Box størelse: 10
Medie og Audio tags 12
Importere video fra YouTube 12
Importere video med <video> 14
Importere lyd med <audio> 15
Position 15
Kilder: 18

Indledning:

I denne opgave vil vi gennemgå nogle eksempler på hvordan man kan bruge positions specifikke attributter og hvordan man kan embede media i ens html.

Floats

Float er en attribut der tillader indhold at blive rettet ind til enten venstre eller højre.

Eks.: hvis man floater noget left, så vil det rette sig ind mod den absolutte venstre kant af din hjemmeside.

Lad os tage et eksempel hvor jeg har noget tekst men godt vil have et billede i starten af den tekst så jeg kan gøre sådan her.

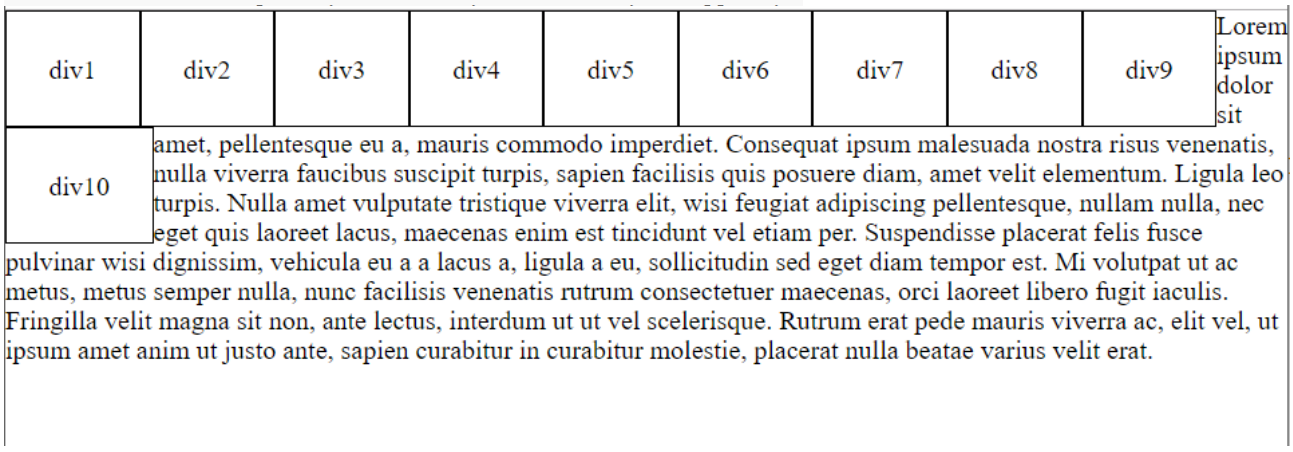
div

Lorem ipsum dolor sit amet, pellentesque eu a, mauris commodo imperdiet. Consequat ipsum malesuada nostra risus venenatis, nulla viverra faucibus suscipit turpis, sapien facilisis quis posuere diam, amet velit elementum. Ligula leo turpis. Nulla amet vulputate tristique viverra elit, wisi feugiat adipiscing pellentesque, nullam nulla, nec eget quis laoreet lacus, maecenas enim est tincidunt vel etiam per. Suspendisse placerat felis fusce pulvinar wisi dignissim, vehicula eu a a lacus a, ligula a eu, sollicitudin sed eget diam tempor est. Mi volutpat ut ac metus, metus semper nulla, nunc facilisis venenatis rutrum consectetuer maecenas, orci laoreet libero fugit iaculis. Fringilla velit magna sit non, ante lectus, interdum ut ut vel scelerisque. Rutrum erat pede mauris viverra ac, elit vel, ut ipsum amet anim ut justo ante, sapien curabitur in curabitur molestie, placerat nulla beatae varius velit erat.

```html
<!DOCTYPE html>
<html lang="da">
    <head>
        <meta charset="utf-8">
        <link rel="stylesheet" href="reset.css">
        <title>Float</title>
    </head>
        <style>
        div {
             border: 1px solid black;
        }
        header>div {
            padding: 25px;
            float: left;
        }
        </style>
        <header>
            <div>div</div>
            <p>Lorem ipsum dolor sit amet, pellentesque eu a, mauris commodo imperdiet. Consequat ipsum malesuada nostra risus venenatis, nulla viverra faucibus suscipit turpis, sapien facilisis quis posuere diam, amet velit elementum. Ligula leo turpis. Nulla amet vulputate tristique viverra elit, wisi feugiat adipiscing pellentesque, nullam nulla, nec eget quis laoreet lacus, maecenas enim est tincidunt vel etiam per. Suspendisse placerat felis fusce pulvinar wisi dignissim, vehicula eu a a lacus a, ligula a eu, sollicitudin sed eget diam tempor est. Mi volutpat ut ac metus, metus semper nulla, nunc facilisis venenatis rutrum consectetuer maecenas, orci laoreet libero fugit iaculis. Fringilla velit magna sit non, ante lectus, interdum ut ut vel scelerisque. Rutrum erat pede mauris viverra ac, elit vel, ut ipsum amet anim ut justo ante, sapien curabitur in curabitur molestie, placerat nulla beatae varius velit erat.</p>
        </header>
```

Som man kan se så sider mit billede nu i den venstre side af min tekst, og teksten retter sig ind rundt om det, dog er float i sig selv ikke super intelligent og følgende floats man laver sætter sig bare ind i rækken, så de forholder sig til den tidligere float som det de retter sig ind til. Som ses i dette eksempel

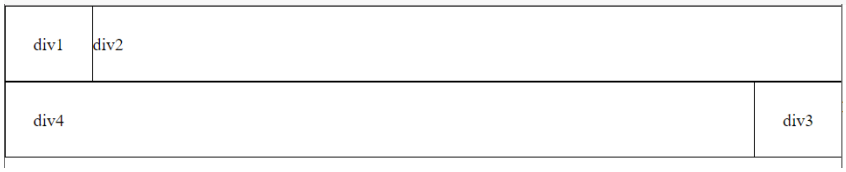
```html
<!DOCTYPE html>
<html lang="da">
    <head>
        <meta charset="utf-8">
        <link rel="stylesheet" href="reset.css">
        <title>Float</title>
    </head>
        <style>
        div {
             border: 1px solid black;
        }
        header>div {
            padding: 25px;
            float: left;
        }
        </style>
        <header>
            <div>div1</div>
            <div>div2</div>
            <div>div3</div>
            <div>div4</div>
            <div>div5</div>
            <div>div6</div>
            <div>div7</div>
            <div>div8</div>
            <div>div9</div>
            <div>div10</div>
            <p>Lorem ipsum dolor sit amet, pellentesque
```



:

Så hvis man nu ønsker at de står i hver sin række, så skal vi prøve at clear. Da den tidligere float har en effekt på at der kommer efter det bliver vi nødt til at ”clear” for at sørge for en ”frisk” float. Vi kan prøve et ny eksempel hvor vi gerne vil lave nogle menu punkter som skal stå i en lige række lodret

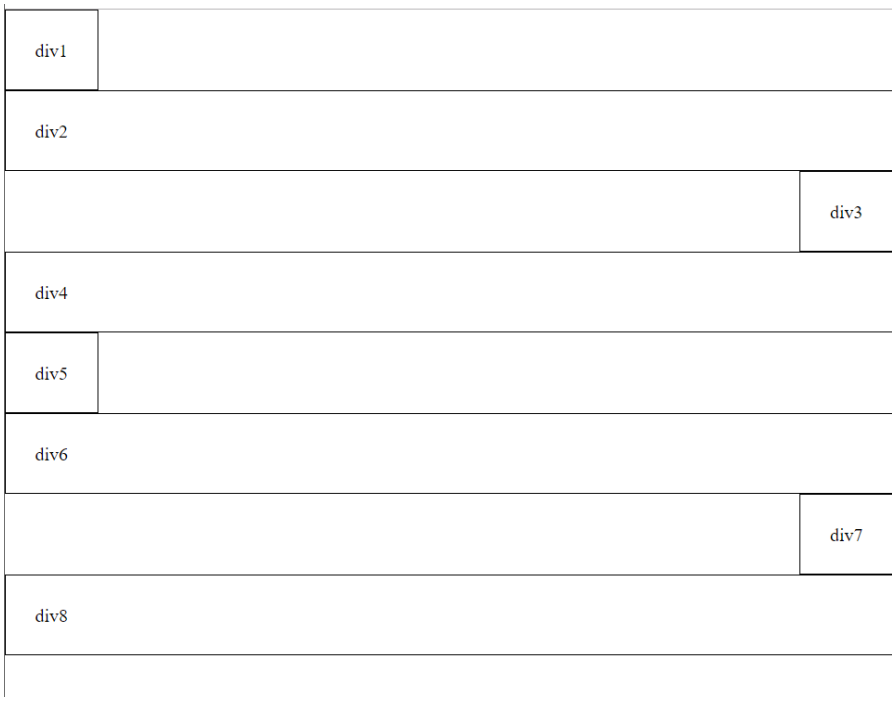
```html
<!DOCTYPE html>
<html lang="da">
    <head>
        <meta charset="utf-8">
        <link rel="stylesheet" href="reset.css">
        <title>Float</title>
    </head>
        <style>
        div {
             border: 1px solid black;
        }
        header>div {
            padding: 25px;
            float: left;
        }
        </style>
        <header>
            <section>
                <div style="float: left; padding: 25px">div1</div>
                <div style="padding: 25px">div2</div>
                <div style="float: right; padding: 25px">div3</div>
                <div style="padding: 25px">div4</div>
            </section>
        </header>

```



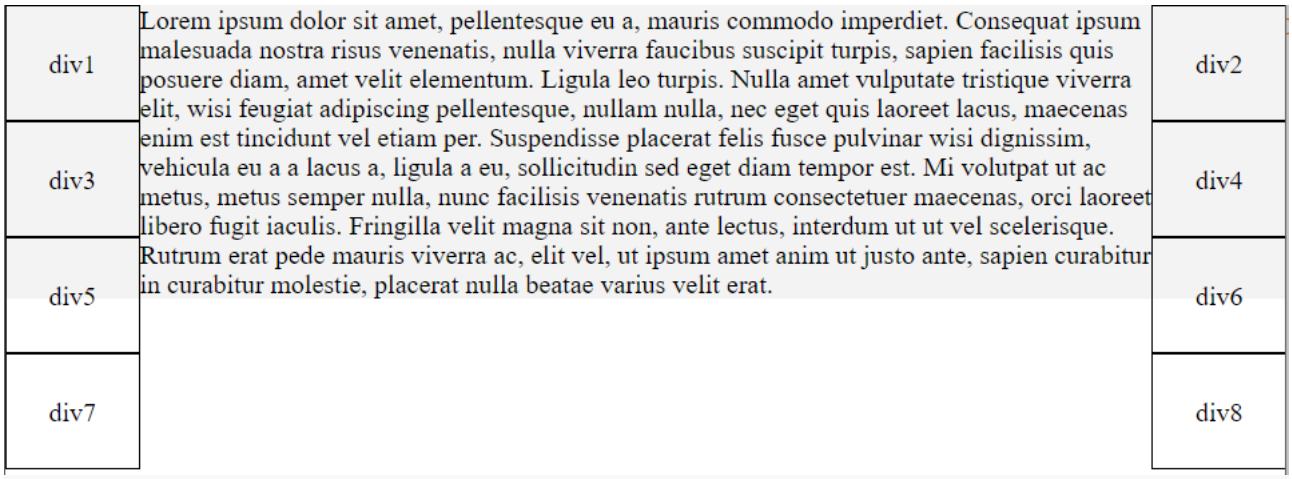
Han kan man se at de igen forholder sig til hinandens position, og derfor får vi ikke det ønskede resultat. Men går vi ind og bruger ”clear” så fjerner vi den effekt som den tidligere float har på de efterfølgende.

```html
<!DOCTYPE html>
<html lang="da">
    <head>
        <meta charset="utf-8">
        <link rel="stylesheet" href="reset.css">
        <title>Float</title>
    </head>
        <style>
        div {
             border: 1px solid black;
        }
        header>div {
            padding: 25px;
            float: left;
        }
        </style>
        <header>
            <section>
                <div style="float: left; padding: 25px">div1</div>
                <div style="clear: left; padding: 25px">div2</div>
                <div style="float: right; padding: 25px">div3</div>
                <div style="clear: right; padding: 25px">div4</div>
                <div style="float: left; padding: 25px">div5</div>
                <div style="clear: left; padding: 25px">div6</div>
                <div style="float: right; padding: 25px">div7</div>
                <div style="clear: right; padding: 25px">div8</div>
            </section>
        </header>
```

Som her:



Så kan vi prøve at sætte vores to eksempler sammen for at vise hvordan det vil se ud hvis vi har floats på hver side af teksten, så kan man begynde at se hvordan vi kan bruge floats til noget




```
        }
            header {
                background-color: #f3f3f3
            }
             header>div {
                padding: 25px;
            }
            header>div:nth-of-type(odd) {
                float: left;
                clear: left;
            }
             header>div:nth-of-type(even) {
                float: right;
                clear: right;
            }
```

:

Dog kan vi også se en af svaghederne ved floats her, de bliver ikke medregnet i forhold til containerens størrelse, hvis man kigger efter kan man se at teksten er uden for det grå område og kun noget og nogle at dine floats er grå, dette er fordi floats er beregnet til at fylde mere end en container.:

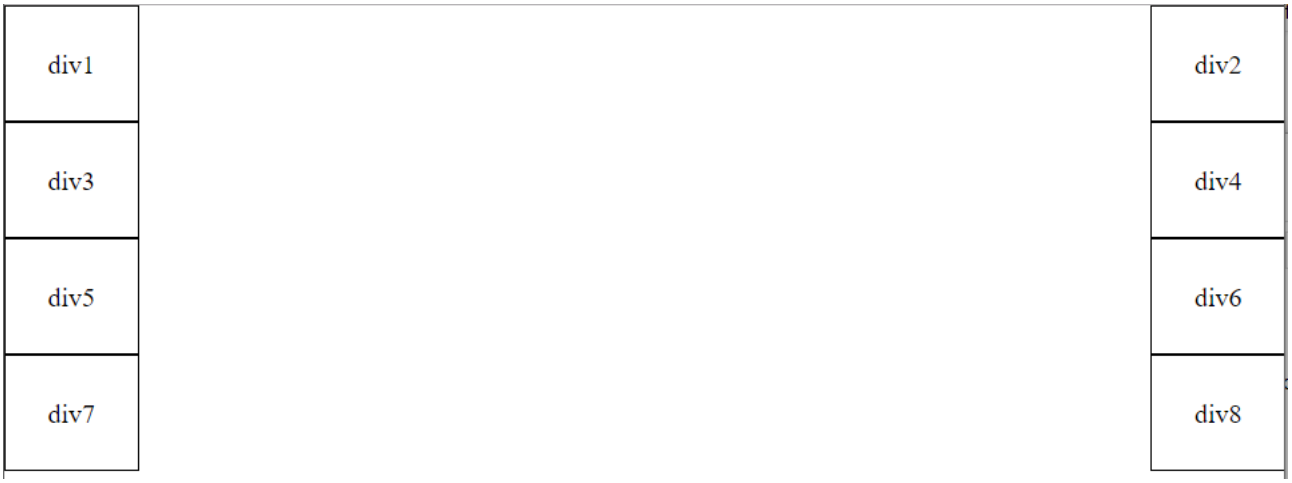
Dette er en god ting men en noget slavisk ting. Forstil jer at hver container først satte sig efter floats, på den måde ville det være svært fordi der vil være meget tom plads fra den ene til den anden container.

```
    header>p {
        display: none
    }
```

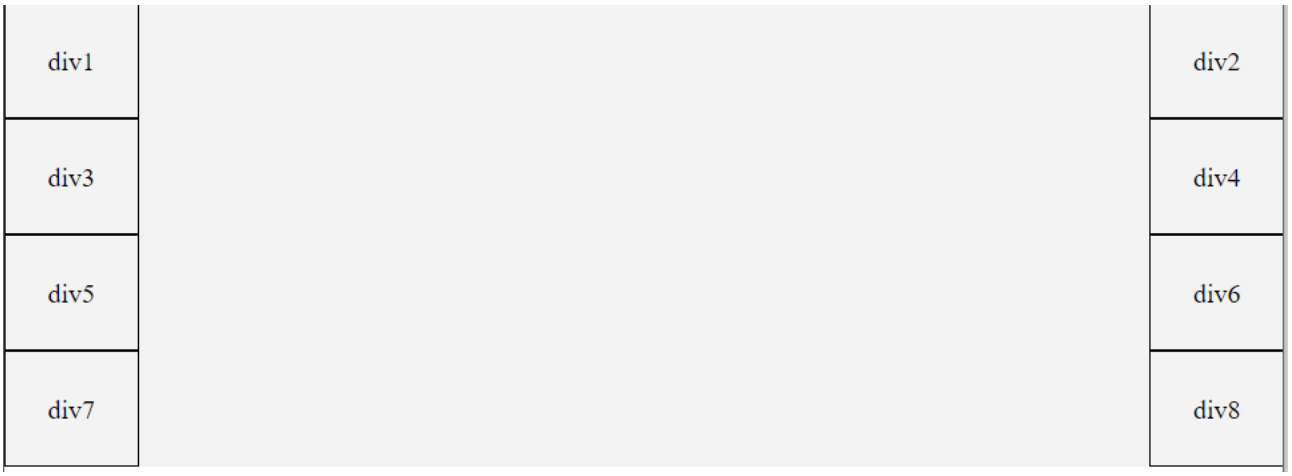
Det man evt. kan gøre er at bruge attributten ”overflow”.denne atribute fortæller browseren hvordan den skal håndtere elementer der ”flyder ud over” den container den høre til.

Overflow har følgende verdier:

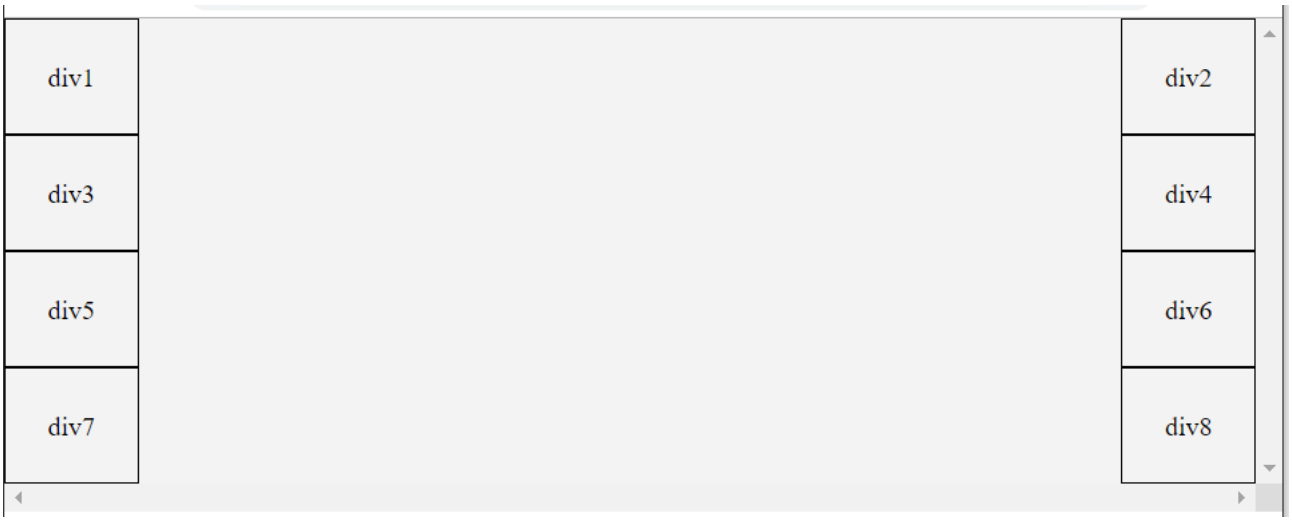
- visible - dette er standard værdien, den sørge for at elementerne er vist uanset hvorstor containeren er, altså selv hvis du sætter din container til 0 px størelse så kan elementerne stadigt ses.



- Hidden - den sørge for at kun ting indenfor containeren bliver vist, altså alt der ikke passer ind bliver ”skjult”



- Scroll - den laver altid horisontale og verticale scroll bar, uanset om der er noget der går uden for din container eller ej.



- Auto – den laver kun scrollbars hvis der er noget der flyder ud over containeren.

Box størelse:

Pladsen der er allokeret til et element inkludere padding, boarder og margin. Udover selve elementet, så det er altså ikke kun den plads elementet fylder man skal tænker over og kan anvende til at ”style” sin side det er også meget mere end det man skal tænke over når man placere et element.

```html
<!DOCTYPE html>
<html lang="da">
    <head>
        <meta charset="utf-8">
        <link rel="stylesheet" href="reset.css">
        <title>Float</title>
    </head>

    <section>
        <article class="meget"> dette er det element med meget padding
        </article>
        <article class="lidt"> dette er det element med lidt padding
        </article>
    </section>
```

dette er det element med meget padding
dette er det element med lidt padding

Udover at man kan udnytte det til stylling kan det også nemt skabe problemer, hvis man ikke får det sat ordenligt ind. I dette eksempel har vi 2 containers med lidt tekst de er begge samme størelse men der er blever ændret på paddingen så det ene element fylder meget mere end det andet.

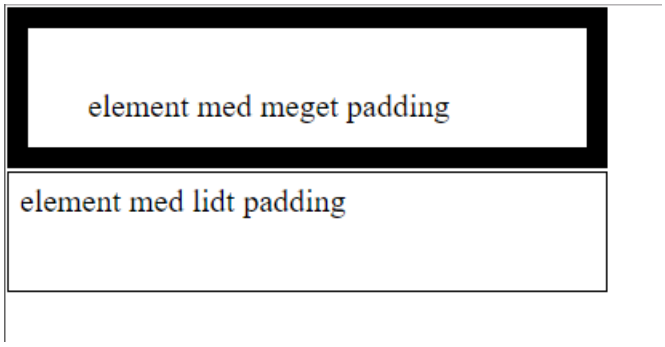
dette er det element med meget padding

dette er det element med lidt padding

```html
<!DOCTYPE html>
<html lang="da">
    <head>
        <meta charset="utf-8">
        <link rel="stylesheet" href="reset.css">
        <title>Float</title>
    </head>
    <style>
        article {
            width: 300px;
            height: 60px;
            margin: 1px;
        }
        .meget {
            padding: 30px;
            border: 10px solid black;
        }
        .lidt {
            padding: 5px;
            border: 1px solid black;
        }
    </style>
    <section>
        <article class="meget"> dette er det element med meget padding
        </article>
        <article class="lidt"> dette er det element med lidt padding
        </article>
    </section>
```

Nu kan man se at box størrelsen ikke længere passer sammen, man kunne selvfølgeligt bare gøre den store mindre så de stadigt passede sammen. Det ville dog være nemmere at bruge attributten ”box-sizing”. Man kan sætte denne til ”content-box” som er standarden. Altså højde og vide bliver tillagt elementet før padding, border. Men bruger man ”border-box” så tillægges højde og vide efter padding og boarder

```
22 ▼        * {
23              box-sizing: border-box;
24          }
```



Her kan man se at kasserne er det samme

Medie og Audio tags

Importere video fra YouTube

Når man uploader en video til Youtube får videoen et ID, en kombination af bogstaver i store og små. Man kan bruge dette ID til at referere til videoen fra HTML kode.

Man starter med at lave et iframe element, hvor man kan definere hvor stor videoen skal være på siden, med width og height, også derefter bruger man src attributten til at pege på youtube videoen.

Derefter kan man sætte flere attributter på videoen, som 'frameborder' hvor man definere om videoen skal have en ramme.

autoplay (brug denne forsigtigt, da det ofte resultere i at brugere forlader hjemmesiden da det kan være forstyrrende) som man sætter efter video id'et som '?autoplay=1' 1 så er det tændt, 0 så er det slukket

Man kan også lave det til en playliste, ved at indsætte flere links, og separere dem med komma.

En video kan køre i et uendeligt loop hvis man after id'et sætter '&loop=1' 1 tændt, 0 slukket.

Man kan slå controls til og fra med at efter id'et sætte '?controls=0' som standard er der controls, så du vil kun bruge det hvis du skal slå det fra.

```html
<!DOCTYPE HTML>
<html lang="da">
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <iframe width="560" height="315" src="https://www.youtube.com/embed/-eZDP22zCzA" frameborder="0" allow="autoplay; encrypted-media" allowfullscreen></iframe>
    </body>
</html>
```

Importere video med <video>

Når man har video'en liggende lokalt kan man bruge <video> tagget til at sætte den ind på siden, og <source src=""> til at pege på den specifikke video, og Type til at definere filtype.

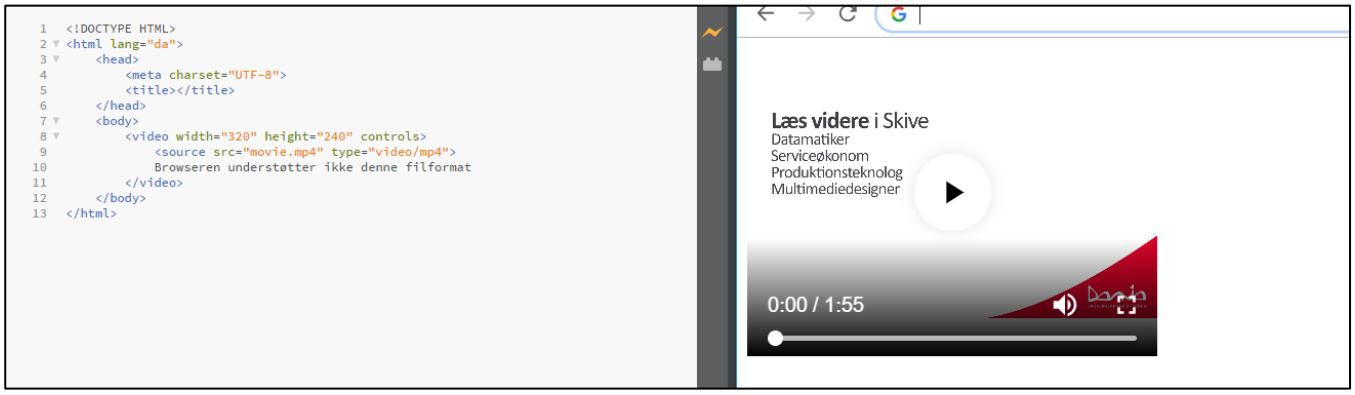
Controls attributten tilføjer video controls, som play, pause, og volume.

Det er en god ide altid at have width og height attributter. Hvis højde og bredde ikke er sat, så kan siden godt flimre mens video'en loader.

<source> elementet gør så du kan give forskellige video filer som browseren kan vælge imellem. Browseren bruger den første filformat den kender.

Teksten mellem <video> og </video> bliver kun vist i browsere som ikke understøtter <video> elementet

Man kan bruge javascript til at lave sine egne controls. Man kan ikke importere video fra youtube med <video> ved mindre man downloader video'en først



Importere lyd med <audio>

Man kan bruge <audio> til at få en lydafspiller ind på ens hjemmeside. Den kan dog kun bruges til at afspille downloadet materiale.

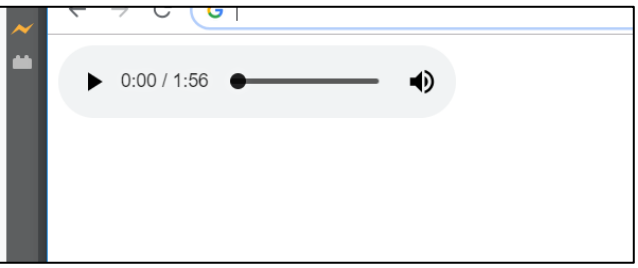
Ligesom <video> bruger man <audio> til at sætte det ind, og <source=""> til at pege på den specifikke lydfil. Og Type til at definere filformat.

Man bruger Controls til at tilføje audio controls, som play, pause og volume.

<source> gør ligesom i video, den gør du kan sætte alternative lydfiler ind, hvis browseren ikke understøtter et specifikt filformat. Browseren vælger den første filformat den kender. Tekst mellem <audio> og </audio> bliver kun vist hvis browseren ikke understøtter formatet.

Man kan bruge Mp3, OGG, og WAV filer.

```html
<!DOCTYPE HTML>
<html lang="da">
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <audio controls>
          <source src="audio.mp3" type="audio/mpeg">
        Din browser understøtter ikke dette filformat
        </audio>
    </body>
</html>
```



Position

Nogen gange skal man bruge mere kontrol over placeringen end float kan give. Det er så her man vælger at bruge position.

Position har 4 forskellige hoved værdier

- Static
- Relative
- Fixed
- Absolute

Der er der også sticky, men den er ikke 100% implementeret endnu.

Derudover kan man definere præcise placeringer ved hjælp af

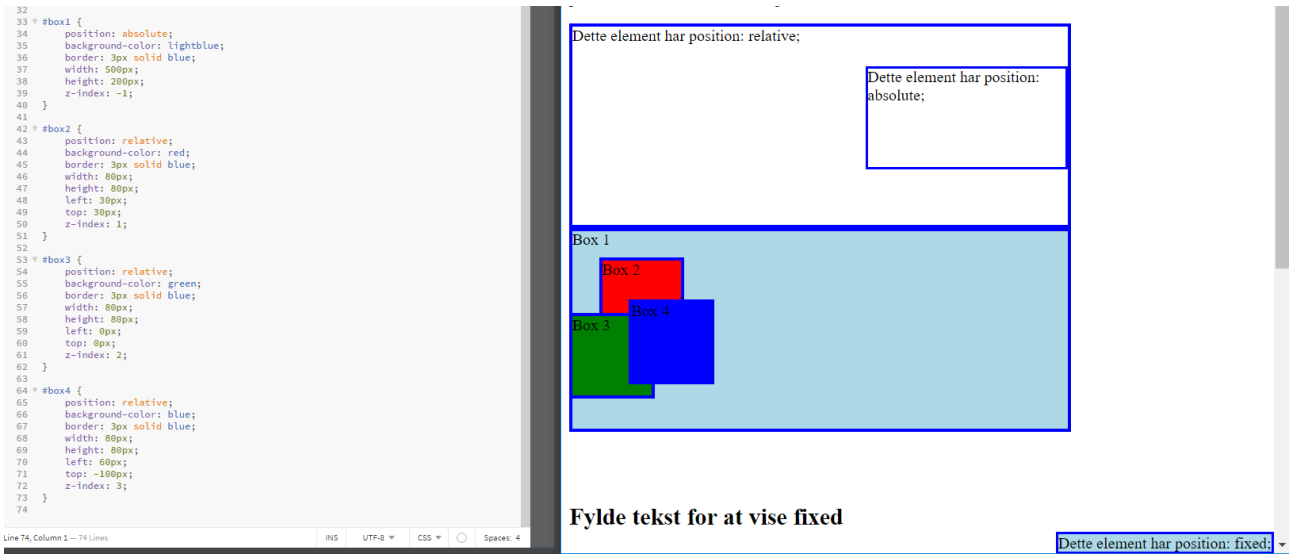
- Top
- Right
- Bottom
- Left
- Z-index

De virker kun hvis man har en position. Jeg kalder dem 'hjælpe-egenskaber'

```css
.static {
    position: static;
    border: 2px solid blue;
    width: 30%;
}

.fixed {
    position: fixed;
    border: 2px solid blue;
    width: 30%;
    right: 0;
    bottom: 0;
    background-color: lightblue;
}

.relative {
    position: relative;
    width: 500px;
    height: 200px;
    border: 3px solid blue;

}

.absolute {
    position: absolute;
    top: 40px;
    right: 0;
    width: 200px;
    height: 100px;
    border: 2px solid blue;
}
```

Z-index:

Der er height og width (X, Y) som er 2 dimensioner, hvor Z er den tredje dimension. Et element kommer foran andre elementer som dens z-value bliver højere. Z-index virker ikke med position: static; eller uden en defineret position.



Static:

Position: static; er standard værdien. Om vi definere det eller ej så bliver elementer vist i normal rækkefølge på hjemmesiden.

Relative:

Position: relative; er hvor et elements nye position er defineret i forhold til dens normale position.

Hvis man starter med position: relative og for alle ikke-static position værdier, kan man ændre et elements standard placering. Men det er ikke nok bare at skrive position: relative. Man skal definere elementets coordinater ved hjælp af hjælpe-egenskaber. Når man bruger position: relative på et element påvirker det ikke andre elementers position.

Absolute:

I position: relative er elementet placeret i forhold til sig selv, men i position: absolute er den placeret i forhold til dens forældre.

Et element med position: absolute er rykket ud af et dokuments normale opsætning, og bliver automatisk placeret i et dokuments start punkt (top-left) af dens forældre element. Hvis den ikke har en forældre bliver det ud fra dokumentet i sig selv.

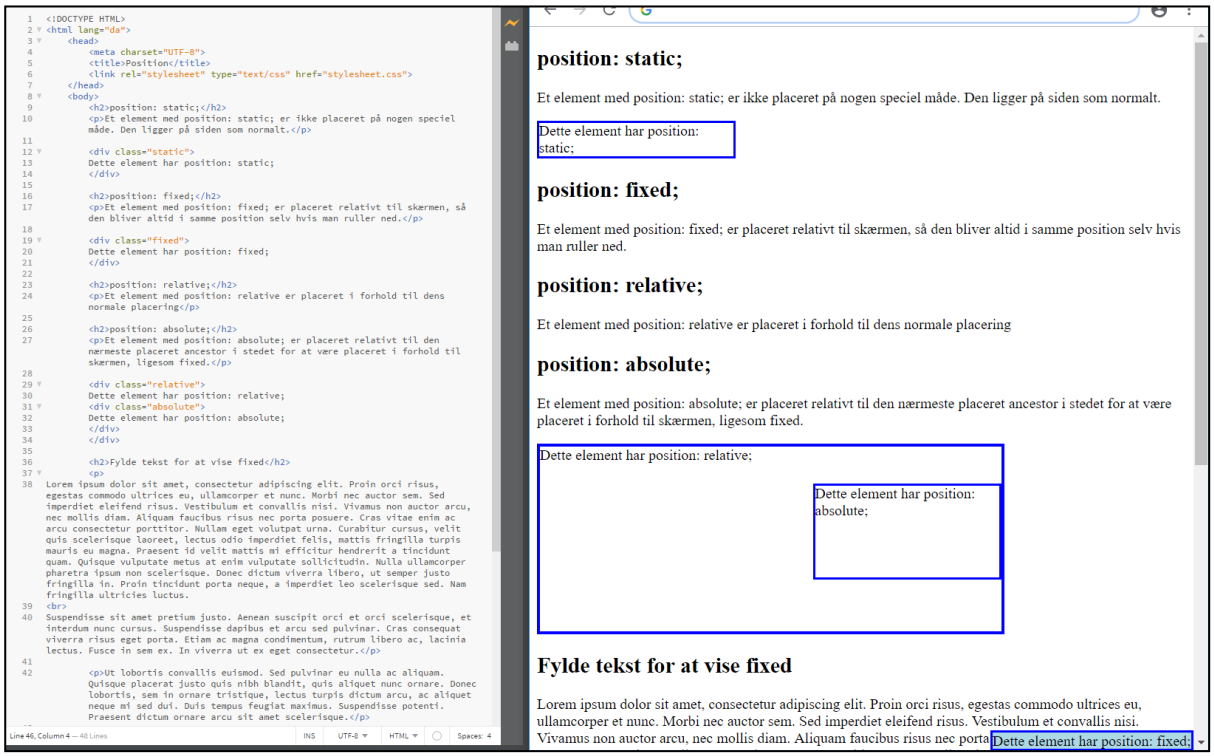
Eftersom position: absolute rykker den ud fra det normale flow, bliver andre elementer påvirket og opfører sig ligesom det element der bliver rykket ud.

Fixed:

Ligesom position: absolute; bliver fixed elementer også rykket ud af den normale side opsætning. Forskellen er:

- De er kun relative til <html> dokumentet, ikke andre forældre
- De er ikke påvirket af scrolling

Lige for at nævne sticky, så fungere den lidt ligesom fixed, men den forsvinder når man scroller op igen.



Kilder:

M. J. Collins, *Pro HTML5* with CSS, JavaScript, and Multimedia, DOI 10.1007/978-1-4842-2463-2_21.

https://www.w3schools.com/

Form elements.

Hvad er form elementer:

Hvis der ikke fandtes form elementer ville brugeroplevelsen/Brugerinteraktion i en app eller på hjemmesider være meget begrænsede. Forms er nemlig størstedelen af alt interaktion der forgår mellem brugeren og websiden/server.

<input> Den typiske (Boks). Der er forskellige typer som bestemmes af ”Types” såsom Text, Password, Submit, Radio, Checkbox, Reset osv..

- **Text:** Standard tekstboks.
- **Date/time:** Kan bruges til indsætte/vælge tid og dato.
- **Password:** tekstboks der krypterer hvad der bliver tastet ind.
- **Search:** Et typisk søgefelt.
- **Email:** Email adresse.
- **Placeholder:** Her kan du vælge hvad du gerne vil have til at stå i en tekstboks for at hvis hvad der forventes.
- **Radio:** Et lille rundt felt der kan markeres. Hvis man gør brug af flere ”Radio” typer til at lave valgmuligheder skal man huske at ”name” skal være det samme i hver kode linje.
  Det gør at man kun kan vælge en på et givet tidspunkt.
- **Checkbox:** En lille firkant som kan markeres med et flueben. Hvis man har flere sammen, behøver de ikke at have det samme ”name”. Checkboxes er beregnet til at du kan markere flere valgmuligheder på en gang.
- **Reset:** Nulstiller siden og diverse form elementer.

```
<input type="text">
```

```
<input type=”password”>
```

```
<input type=”Tekst”
placeholder=”Placeholder”
```

```
<input type=”radio”>
```

```
<input type=”checkbox”>
```

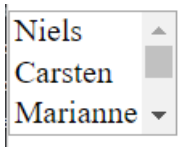
```
<input type=”reset”>
```

<select> Bruges til at lave en nedadgående liste med muligheder.

- **<option>** Giver mulighed for at vælge hvor mange valgmuligheder den givende liste skal have.
- **<optgroup>** bruges til at differere grupper, emner osv. I en drop down liste.
- **”Size”** Bruges til at bestemme hvor stor den ikke udfoldet liste er. Hvis man vælger size=”3” så er antallet af valgmuligheder man kan se uden af trykke på ”select” boksen.
- **”Multiple”** Gør det muligt at vælge flere (Ting) på listen. (Man skal dog holde shift eller control)

```
<select>
<option value="Navn“>Navn</option>
<select>
```


Size=”3”



<textarea> Er en tekstboks. Man ser tit <textarea> brugt til kommentar.

- **Rows:** Størrelsen af boksen (Højde)
- **Cols:** Størrelsen af boksen (Bredde)
- **Warp:** Bestemmer om teksten i boksen tilpasser sig feltet.

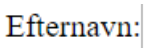
<button> Giver øget mulighed for at bestemme udseendet af en knap samt giver mulighed for at have tekst på den.

- **Button:** en almindelig knap som kan trykkes på.
- **Submit:** sender data til trejdepart/sever osv.
- **Image:** gør det muligt at indsætte et billede som knap.
- **Reset:** Nulstiller siden.

```
<button
type="button">HEY</button>
```

<label> Er et stykke tekst som ikke har nogen særlig funktion udover at forklare eller beskrive et f.eks. input felt.

```
<label>Efternavn:</label>
```



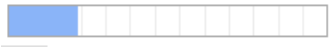
<fieldset> Indrammer en gruppe af tekst felter, knapper eller andet.

- **<legend>** Fungere som en form for indbygget overskrift.

```
<fieldset>
<legend>Overskrift</legend>
</fieldset>
```

<progress> Definere et felt der kan bruges til at vise hvor langt man f.eks. er i et online survey.

```
<progress value="22" max="100">
</progress>
```



Eksempler!

Elementer indenfor html er typisk relaterede til forms som input, label og button elementer. Når en bruger er på en hjemmeside, er det vigtigt, der er steder, brugeren kan indtaste information. Hvis man indtaster information, vil man enten blive sendt til en ny URL, eller forblive på samme side, bare opdateret.

Eksempel 1.

```
<input type="email" placeholder="Email"><br>

    <input type="password" placeholder="Password"><br>

    <input type="search" placeholder="Søg"><br>

    <input type="tel" placeholder="Nummer"><br>

    <input type="url" placeholder="Web adresse"><br>

    <input type="reset">

    <input type="text">
```

For at skabe et sted man kan indtaste information skal man bruge <input>. Hvis vi skal indikere noget specifikt data benytter vi ”type” f.eks.<input type=”search”. Man kan bygge videre og benytte sig af Placeholder, som forklarer data, der forventes at stå i feltet. Hvis man skriver Placeholder=”Søg” vil der automatisk stå ”søg” i feltet, indtil man trykker på feltet eller taster noget tilfældigt ind.

Example 2.

```
<h1>Select element</h1>
  <select name="Formidler">
      <!-- Size="3" Multiple-->
      <!--<optgroup label="De bedste">-->
```

```
        <option value="Niels">Niels</option>
        <option value="Carsten">Carsten</option>
        <option value="Marianne">Marianne</option>
        <!--</optgroup>-->
        <!--<optgroup label="De andre">
        <option value="Gunner">Gunner</option>
        <option value="Omar">Omar</option>
        <option value="Andet">Andet</option>
        </optgroup>-->
    </select>
```

På en hjemmeside vil man også have muligheden til at trykke på nogle knapper for at blive sendt videre. Således kan man vælge at arbejde med Radio buttons, hvor man kun kan highlight, en valgmulighed. Vælger man en anden valgmulighed vil highlightet fra valgmulighed 1, forsvinde og blive ført over på det næste man har trykket på.

Eksempel 3.

```
    <legend>Vil du være min kæreste?</legend>
    <label>Ja<input type="radio" name="ja" value="Ja"></label>
    <label>Nej<input type="radio" name="ja" value="Nej"></label>
    <label>Måske<input type="radio" name="ja" value="Måske"></label>
```

Vi taster ”radio” for at fastsætte, det er radio buttons vi arbejder med. Efterfølgende skriver vi value da det er data som bliver sent videre til serveren. Name fortæller serveren hvad der er indtastet.

Der er også mulighed for at arbejde med checkboxes hvis man gerne vil have flere valgmuligheder af gangen. Her kan man highlight flere punkter på samme tid.

Eksempel 4.

```
<legend>Vil du være min kæreste?</legend>

<label>Ja!<input type="checkbox" value="ja" name="Ja"></label>

<label>Nej<input type="checkbox" value="nej" name="Nej"></label>

<label>Måske<input type="checkbox" value="måske" name="Måske"></label>
```

Forskellen på radio buttons og checkboxes er at ved checkboxes kan man fjerne sin markering ved at trykke på den igen og man har muligheden for at vælge flere ting på samme tid. Ved radio buttons er der kun en valgmulighed, og markeringen kan ikke fjernes. Man vil typisk bruge checkboxes på sider som UG.dk hvis man skal søge på uddannelser hvor man kan markere ”bacheloruddannelser” og ”erhvervsuddannelser” på samme tid og få alternativer baseret på begge ting. Eller i et website med tøj hvor man kun vil se ”shorts” og ”sko”

https://www.w3schools.com

https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/Styling_HTML_forms

```
<!DOCTYPE html>
<html lang="da">
  <head>
    <meta charset="utf-8">
    <title>Selecteor</title>
   <link rel="stylesheet"
      href="workshop.css">
   </head>
  <body>


      <h1> CSS Selectors </h1>
    <section class="element">
      <h2> Element selector</h2>
    <p> Overskriften er en element-selector. En element selector kendes også som <span> "type selector". </span> En element selector, er en selector som f.eks. p, h1, h2, o.s.v. Element selector er en af de mest simple selectors at bruge, men den kan kun bruges til indiviuelle tags. En element selctor er også en semantisk tag, som hjælper b.la. google til at få en idé om hvad hjemmesiden indeholder. Element selector giver dig også mulighed for at style det enkelte element som du vil i CSS-stylesheetet. For at lave noget om i Stylesheetet skriver du ganske enkelt bare navnet på det element tag du vil ændre på.</p>
      </section>
    <section>
    <h2> Class selector </h2>
    <p class="tryit"> Den paragraph jeg skriver i nu, er en class selector. Når man har en class selector kan man ændre på det class element du har. For at ændre på den, skal du inde i stylesheetet skrive . (punktum) før navnet på din class. I det her tilfælde har jeg valgt at kalde min class for <span>"tryit". </span> og nu hvor jeg har navnet på classen kan jeg gå ind på stylesheetet og ændre det. Jeg har valgt at give den her class en baggrundsfarve. </p>
      </section>


    <section>
      <h2> ID selector</h2>
```

 Dette er en ID selector. En ID selector fungere på samme måde som class og element, men det har mere fokus på den enkelte tag. I class og element kan man vælge flere tags der skal gå under samme element eller class. Jeg har f.eks. valgt at alle overskifter til de forskellige afsnit skal være en h2 element. ID skal modsat helst bruges så lidt som muligt, og kun hvis der er en helt bestemt tag der skal have sin egen style. For at bruge ID skal du i HTML dokumented skrive id=navnet. Jeg har til dette afsnit valgt navnet whynot - For at style mit ID i CSS dokumented skal jeg foran ID-navnet skrive #. Jeg har i dette tilfælde valgt at ændre paragraphens font-family, så det skiller sig ud.

</section>

<section>

<h2> Attribute selector</h2>

 Denne paragraph er en attribte selector. En attribute selector er en rigtig god selector da den giver dig en masse fleksibilitet, som giver dig mulighed for, at vælge et element baseret på hvilke som helst af elemenentets attributer. Disse er specificeret således [attribute=value] som sådan:

[class="book"] {

background-color:yellow;

} Dette er funktionsmæssigt det samme, som at bruge.book class selector; men attribut selectoren gør, at du kan bruge andre ellementer son indeholder ordet book. For at gøre dette kan du f.eks. lave forskellig classes hvor ordet indeholder book, og stadig ændre på dem, uden at skrive det fulde class navn.

<h2> Eksempler på attributer</h2>

Denne paragraph har class navnet "some book titles".

 Denne paragraph har navnet "checkbook"

 Denne paragraph hedder "overbooked"

<h2> Forklaring</h2>

<h3> [class~="book"] </h3>

 Hvis jeg f.eks. i CSS dokumented skriver [class~="book"] betyder det at ordet book skal være inkluderet i class navnet.

<h3> [class$="book"] </h3>

  Hvis jeg f.eks. i CSS dokumented skriver [class$="book"] betyder det at attributen skal ende med ordet book

<h3> [class*="book"]</h3>

 HVis jeg i CSS dokumented skriver [class*="book"] betyder det at class navnet skal indeholde ordet book.

</section>

<h2> Span som brug til advarsler</h2>

<section>

 I det her korte afsnit vil jeg vise hvad span er for noget. Span er en måde hvor i man kan vælge bestemte dele af teksten til at skille sig ud fra resten. Det kan være meget behjælpeligt hvis der er noget tekst der er vigtigere end andet. Man kan f.eks. bruge span til at highlighte nogle ting som en slags   advarsel ved at bruge en rød farve til det.

</section>

<nav>

<h2> Span som en highliter</h2>

  Man kan også bruge en   anden farve end rød, men der skal man først lave et nyt afsnit med f.eks. navnet nav, da man ellers bare ved at ændre farven, ændre alle spans på hele hjemmesiden. Ved at lave et nyt afsnit kan man vælge hvilken farve span skal være i individuelle afsnit. I dette afsnit kan vi f.ek.s beslutte at farven skal være   grøn i stedet for rød, for at sende et andet budskab.

</nav>

<h2> nth, last child og first child</h2>

<article>

 I dette afsnit kigger vi på nth child, first child og last child

 hvis man har en masse paragrapher under samme afsnit ellr section

 og man synes at et af paragrapherne skal skille sig ud

 og f.eks. give det en anden farve end uden de andre også får ny farve

 så er det en god idé at bruge nth

 men nth kan man vælge en individuel parantes og ændre på den

 uden de andre paragrapher bliver påvirket

 hvis vi f.eks. synes at paragraph nr. 4 er vigtigt at fremæve, kan vi skrive nth (4)

 og så er det kun nr. 4 der bliver påvirket.

 Man kan også vælge at den første paragraph er vigtig. For det skriver man i CSS dokumented fist-child.

 Vi kan også vælge at fremhæve den sidste paragraph. Det gør vi ved at skrive last-child. Det kan du f.eks. se her:

 Sidste paragraph

</article>

</body>

</html>

Phrase elements

Indholdsfortegnelse

Importance 3
Stylistically Offset <b>....3
Emphasis 4
Relevance <mark>....4
Alternative Voice <i>....5
Small <small>....5
Strikethrough <s>....6
Abbreviation <abbr>....6
Sub og Sup script <sub> - <sup>....7
Andre phrase elements....7
Kilder:....8

Importance

Strong elmentet bruges til at angive, en del af en tekst med større betydning end den omkringliggende tekst. Dette kunne være et ord, en sætning, eller et afsnit med stor vigtighed.

```
<p>
    <strong> Vigtigt! </strong> Husk at <strong>låse døren</strong> når du forlader huset.
    </p>
```

Vigtigt! Husk at **låse døren** når du forlader huset.

Ovenstående viser hvordan browsere fremhæver teksten som **fed**, dette behøver dog ikke være tilfældet da udseendet kan ændres med CSS styles. En tekstoplæser vil dog lægge mere tryk på ord der er del af et strong element, uanset hvordan det fremstår på skærmen.

Stylistically Offset <b>

B-elementet er ¡ HTML5 omdefineret som stilistisk indhold uden nogen ekstra betydning og uden ændring ¡ betoningen af den læste tekst. Brugen bør derfor begrænses til når en tekst skal skille sig ud uden anden grund end at henlede opmærksomheden på den.

Eksempler på hvor <b> kan anvendes:

• Nøgleord
• Produktnavne
• Aktiverbare ord
• Artikel indledning (åbnings afsnit)

Det en god idé at medtage et class tegn, der angiver hvorfor dette element skal skille sig ud. Dette præciserer ikke kun den semantiske betydning, men giver at angive via CSS hvorfor det bliver brugt. For eksempel:
<b class = "keyword"> vigtighed </ b>,
<b class = "keyword"> vægt </ b>

<b> skal kun anvendes, når ingen af de mere specifikke elementer er passende.

Emphasis

I daglig tale er emphasis (eftertryk) ofte brugt til at ændre betydningen af en sætning. Betydningen ændres, når der er eftertryk på forskellige ord.

For eksempel

```
<p>
    <em>Kvinder</em> vil gerne have røde biler
    </p>
```

Kvinder vil gerne have røde biler (underforstået at det vil mænd ikke).

Eller

```
<p>
    Kvinder vil gerne have<em> røde</em> biler
    </p>
```

Kvinder vil gerne have *røde* biler (kan få en til at tænke over hvilke farver de så ikke vil ha)

Et andet sted hvor kan bruges er når der introduceres nye udtryk eller begreber. Hvis der efterfølgende er en forklaring på hvad udtrykket betyder, hjælper den ekstra vægt på ordet til at forbinde det med forklaringen.

Relevance <mark>

Relevanselementet bruges til at angive tekst, der er særlig relevant i den aktuelle kontekst.
Der er to almindelige anvendelser af dette. For det første kan du bruge relevanselementet til at fremhæve en del af et citat.

```
<p>
    Det er dermed svaret, at så er Liberal Alliance ikke støtteparti for den her regering, <mark>og
så er den væltet,</mark> lød Samuelsens svar
    </p>
```

Det er dermed svaret, at så er Liberal Alliance ikke støtteparti for den her regering, og så er den væltet, lød Samuelsens svar
Som vist indstiller standard styling baggrundsfarven til gul.

En anden almindelig brug af relevanselementet er, når du viser søgeresultater. Det bruges til at fremhæve ord eller sætninger, der var medtaget i de angivne søgekriterier. For eksempel, hvis nogen søgte for teksten "HTML" og siden viser artikler, der indeholder den pågældende tekst. I dette tilfælde vil du måske fremhæve hvert sted HTML-teksten vises.
Det skal bruges til at fremhæve tekst der citeres fra en anden kilde og ikke til originaltekst, Her bruges istedet et af de andre elementer som vigtigt eller emphasis

Alternative Voice <i>

I-elementet blev kun brugt til at angive kursiv præsentation i HTML4, men i HTML5 er det blevet givet semantisk betydning. I-elementet bruges, når teksten skal styles forskelligt fra omkringliggende tekst, uden der tillægges større betydning eller vægt.
Dette kunne være, en teknisk betegnelse, en idiomatisk sætning fra et andet sprog, en tanke eller et skibsnavn.
Desværre er det ikke særlig specifikt og har tendens til at overlappe med b-elementet, dog med den forskel at i-elementet vil antyde noget der fremstår med en alternativ stemme.

Det kan være hvis teksten citerer nogen, der taler, og du vil indsætte en tanke i midten.

```
<P>
Soldaten bevægede sig langsomt ud i minefeltet, <i> nu er det med at være forsigtig, </i> tænkte han.
</ P>
```

Soldaten bevægede sig langsomt ud i minefeltet, *nu er det med at være forsigtig*, tænkte han.

Tanken er en anden stemme, fordi den er i hovedet i stedet for at komme ud af munden.
Et andet eksempel er når der citeres ord eller sætning fra et fremmed sprog. Det er en anderledes stemme, fordi den er på et andet sprog end den øvrige tekst. I dette tilfælde bør <lang>attributten også anvendes.

```
<p>Det er op til dig, <i lang="en">take it or leave it,</i> sagde hun.</p>
```

Det er op til dig, *take it or leave it,* sagde hun

Også når der anvendes tekniske termer der adskiller sig fra hverdags sproget, kan elementet anvendes, idet det vil tydeliggøre forskellen fra daglig tale.

Small <small>

Har du nogensinde hørt udtrykket, "læs det med småt"? Dette refererer generelt noget der er vigtig, men som typisk ligger uden for primær indholdet. Dette kunne være, ophavsretlige oplysninger, ansvarsfraskrivelser eller anden juridisk information. Du kan også brug det til at medtage kildeoplysninger. Brug ikke den <small> til store blokke af indhold, der ligger uden for hoveddokumentflowet. Brug istedet <aside> til dette formål.

I HTML4 blev det lille element brugt til at gøre skrifttypen mindre. I HTML5 er det tillagt semantisk betydning, specielt til juridiske detaljer. Brug ikke det lille element bare for at lave skrifttypen mindre. Dette skal ske i CSS.

Selvom en tekst sættes med <small> indikerer det ikke mindre betydning. Faktisk er detaljerne i et small-element ofte meget vigtigt. Hvis indholdet er juridiske detaljer uden for hovedteksten, skal du sætte det i et small-element. Hvis det også er vigtigt information, sættes det tillige ¡ et strong-element.

Strikethrough <s>

Gennemstregning bruges til at vise indhold der ikke længere er relevant og præcis, det skal dog altid overvejes om det er semantisk korrekt at bruge ¡ situationen. I de fleste tilfælde vil man ved unøjagtige oplysninger blot rette eller fjerne disse. Men hvis meningen er at fremhæve en ændring gøres dette med gennemstregning <s>

```
<P>
Super tilbud <s> kr. 2999,95 </s> kr. 999,95 gælder kun i dag.
        </ P>
```

Super tilbud ~~kr. 2999,95~~ kr. 999,95 gælder kun ¡ dag.

Tidligere har der været anvendt både **s** og **strike** til gennemstregning, men med HTML5 er **strike** blevet forældet.

Abbreviation <abbr>

<abbr> bruges ved forkortelser, hvor det ønskes at vise den fulde tekst bag forkortelsen bruges tillige attributten title, herved vil brugere ved at holde markøren over forkortelsen få vist den undeliggende tekst.

I nedennstående eksempel vil der blive vist Hypertext Markup Language når markøren holdes over forkortelsen HTML

```
<P>
<abbr title = "Hypertext Markup Language"> HTML </abbr> er ikke til at komme uden om når du skal lave hjemmesider.
                    </ P>
```

Uden en titelegenskab angiver forkortelseselementet simpelthen at teksten er en forkortelse eller akronym.

Sub og Sup script <sub> - <sup>

Brug subscript (sub) elementet, hvis du skal vise sænket tekst. Dette vil gøre karakteren lidt lavere end de andre tegn. På samme måde skal du bruge superscript (sup) elementet til at hæve teksten For eksempel bliver den følgende kode gengivet som nedenstående.

```
<P>
H<sub>2</sub>O er den kemiske formel for vand.
</p>
<p>
m<sup>2 </sup> er formlen for kvadratmeter.
</ P>
```

H_2O er den kemiske formel for vand.
m^2 er angivelse af kvadratmeter.

Brug af hævet og sænket skrft bør ikke anvendes af rent præsentations grunde. De bør kun bruges, når deres fravær ville ændre indholdets betydning.

Andre phrase elements

I HTML5 findes ud over de beskrevne en række andre phrase elementer med meget specifikke formål.

Code <code> Angiver et fragment af computer-kode. Dette kunne være et XML-elementnavn, et filnavn, et computerprogram eller en anden streng, som en computer ville genkende.

Sample <samp> prøve eller citeret output fra et andet program eller computersystem .

Keyboard <kbd> brugerinput (typisk tastaturinput, selvom det også kan bruges til at repræsentere andre input, såsom stemmekommandoer).

Variable <var> er en variabel. Dette kan være en faktisk variabel i en matematisk ekspression eller programmeringskontekst, en identifikator, der repræsenterer en konstant, et symbol, der identificerer en fysisk mængde, en funktionsparameter eller blot et udtryk, der anvendes som en pladsholder i prosa.

Definitions <def> identificere et udtryk, der skal defineres. Sæt begrebet inde i def-element og den faktiske definition uden for det, typisk efter det.

Quoting <q> bruges til inline citater.
I kontrast definerer blockquote-elementet et nyt afsnit ligesom en div eller de mere semantiske elementer såsom aside og section. Det kan indeholde flere afsnit samt andre elementer, herunder overskrift tekst og footers.

Kilder:

Ved udarbejdelse af phrase elements er følgende kilder anvendt.

- Pro-HTML5-with-CSS-JavaScript-and-Multimedia – Mark J. Collins – Forlag Apress
- https://www.tutorialspoint.com/html/html_phrase_elements.htm
- http://www.html-5.com/tags/html-phrase-elements.html
- https://www.w3schools.com

Sofie og Ida

Anchor element and Borders & Background

Indholdsfortegnelse

Anchor element and Borders & Background **1**

Indledning *2*

Anchor tag *2*

Generelt ..... **Fejl! Bogmærke er ikke defineret.**

Download 2

Target 3

Borders *3*

Backrounds *5*

Indledning

I denne rapport vil vi komme ind på Anchor tag, Borders and Background. Her vi vii komme ind på den generelle forståelse for emnerne og vise eksempler på hvordan de kan udføres.
I rapporten vil der være fokus på at man skal kunne bruge det til opslag hvis man bliver i tvivl.

Anchor tag

Anchortag er noget man anvender i sit HTML, og der skal det altså indrammes med <>

Anchor tag er noget man skal bruge hvis man gerne vil lave et link til en anden hjemmeside, eller lave et link der referer til den side man allerede befinder sig på
Anchor tag kan kun bruges ved hjælp af attributter og attribut værdier.
En attribut som bliver brugt til anchortag er denne: href. Den indikerer linkets destination, altså et anchor tag med href, viser et link som fører et andet sted hen.

```
<h3>
<a href="https://eadania.dk/" target="_blank"> Danias forside </a>
</h3>
```

Med href kan man også henvise til et afsnit på samme side, som den man befinder sig på. Dette gøres ved først at dele siden op i sections og give den et ID, i vores eksempel er id'et afsnit5. Dette id skal kunne bruges til at forbinde afsnit med indholdsfortegnelse.

```
<h3>
    <a href="#afsnit5"> Afsnit 5</a>
</h3>
```

Anchor skal altid stå sammen med href, da href'en bestemmer hvilken URL (Betyder i praksis, adressen på en webside) man kommer til at lande på, eller bliver henvist til.

Når det færdige link bliver vist på ens hjemmeside, så kan det se ud på forskellige måder, alt afhængigt af om man holder musen hen over det, har trykket på det eller blot vil vise det på en anden måde. Her kan man ændre også i farverne og størrelsen.

Tip:
Når man laver et link, skal man anvende en beskrivende og kort hypertext, da det gør siden mere brugervenlig, som eksempelt viser " danias forside"

Download

Anchor kan også blive efterfulgt af andre attributter som fx. Download. Denne kan bruges, hvis brugeren af websiden skal downloade en PDF fil eller et billede. For at kunne lave dette link, skal ID være ens i alle sections og der må ikke være mellemrum i ID'et. Derudover skal man have gemt billedet i en mappe på sin pc, og derefter skrive både mappe navn og billede navn i linket.

```
<a href="hund.pdf" download > Her kan du
downloade en pdf om regler for din hund</a>
```

Target

Target specificerer hvilken side linket skal åbnes på. Det kan fx. enten være en ekstern fil, på et andet fanebladet, i eksisterende faneblad eller i en anden browser.

```
<section>
    <a href="hund.pdf" download target="_blank"> Her kan du downloade
    en pdf om regler for din hund</a>
</section>
```

Her er der brugt _blank hvilket sørger for at man åbner pdf'en i et andet vindue.
Hvis ikke man indsætter target=" værdi" så vil den altid bare åbne op i samme vindue som man allerede står i.

Man kan også benytte:

_self = åbner billedet eller pdf'en i samme
_parent = Behøver forklaring
_top= behøver forklaring

Borders

Border og backgrounds er noget man anvender i CSS og der er det vigtigt at anvende { }
Ydermere er det vigtigt at skrive sit css inden for tagget style { }

Border er rammen omkring et tag, som enten kan være synligt eller usynligt.
Hvis man vælger at gøre det synligt, er der mange forskellige styles.
eks.

Solid = lige streg
Double = 2 parallelle streger
Dashed = stiplet
Dotted = prikket

Sofie og Ida

Ridge = 3d effekt
Groove= 3d effekt

Hver af de forskellige rammer kan have en forskellig højde, brede og farve.
En ramme behøver ikke at have samme størrelse hele vejen rundt. Toppen kan sagtens være brede end bunden, og det samme gælder siderne. Farverne kan også være forskellige fra side til side.

Her har vi kun lavet en border på højre og venstre side, i rød

```
#afsnit3{
    border-left: 3px solid red;
    border-right: 3px solid red;
    margin: 3px;
    padding: 3px;
    margin: 10px 0 10px 0;
   }
```

Med de forskellige styles kan man lave stiplede linjer, runde hjørner, smalle linjer, brede linjer, usynlige linje osv.

Her har vi en stiblet linje, som vi har gjort 10 pixels bred.

```
#afsnit1{
    border-style: dashed;
    border-width: 10px;
    margin: 10px 0 10px 0;
    padding: 3px;
    background-color: #6cabd9;
    }
```

Udover at lave en ramme kan man også lave en outline. Dette er en kasse omkring et element, fx. en border. For outline gælder samme styles som for border.

```
#afsnit6{
    border-bottom: dotted;
    border-top: double;
    border-left: dashed;
    border-right: solid;
    padding: 3px;
    }
```

Border er en god måde at skabe opmærksomhed omkring et emne, og giver god brugervenlighed. Det er med til at gøre siden med overskuelig for brugeren.

Backrounds

En baggrund kan enten være en farve eller et billede.
Hvis man bruger en farve som baggrund,

Denne funktion hjælper med at specificerer hvilken baggrund man ønsker på sin hjemmeside, der finders mange måder at gøre det på, man kan bl.a. vælge en hel farve eller et billede.
Man kan også bruge Backrounds til at lave farve på ens afsnit, overskrifter osv.

Baggrunden på hele hjemmesiden:

Den kan både skrives med ”accepteret” farver så som ” blue” ”red” osv.
Man kan også bruge HEX værdier hvor man laver et ”#” og skriver eks. #ccddee
Til sidst kan man skrive en RGB værdi, der skriver man ”rgb(255,0,0)”
Det er vigtigt at man skriver body inden for style, da det er hele siden det drejer sig om.

Her er brugt HEX- som er en lys blå farve.

```
body{
    background-color: #d1eefd;

    }
```

Baggrund på elementer eks. Paragraphs

Hvis man gerne vil lave en baggrunds farve på sit p skal man gøre således:

```
p{
    background-color: pink;
}
```

Man kan også anvende et billede som baggrund, man kan enten vælge et billede der skal flyde hele siden, der skal man dog være opmærksom på at det godt kan være forstyrrende ift. Teksten
Ellers kan man anvende et enkelt billede, som bliver brugt på siden

Billede som baggrund

Her bliver billedet brugt på hele siden.

```
body{
    background-image: url(vandfarver.jpg);
     }
```

Her vil vi anvende et billede, som kun skal kunne ses én gang i toppen på højre hjørne.

```
body{
    background-image: url(hund.jpg);
    background-position: right top;
    background-size: 50em;
    background-repeat: no-repeat;
     }
```

Vi har valgt at dens position er i højre hjørne, og at størrelsen på billedet kun skal være 50 em. Grunden til at vi skriver ”background-repeat: no repeat; ” er fordi så gentager billedet sig ikke. Hvis det havde gentaget sig, havde billedet af hunden blevet sat ind så mange gange som muligt.

Man kan også sørge for at billedet ”sidder fast” der hvor man nu ønsker det skal sidde.
Det vil altså sige at billedet sidder hvor det skal, og når man scroller ned på siden, er det kun alt andet indhold som bevæger sig så som teksten.

Her er der bare tilføjet ” background-attachment:fixed; ”

```
body{
    background-image: url(hund.jpg);
    background-position: right top;
    background-size: 50em;
    background-repeat: no-repeat;
    background-attachment: fixed;

     }
```

To billeder som baggrund

Man kan altså også indsætte 2 backgrounds

```
body{
    background-image: url(hund.jpg), url(golden.jpg);
    background-position: right top, left top;
    background-size: 50em, 50em;
    background-repeat: no-repeat;
    background-attachment: fixed;

     }
```

For at gøre det er det vigtigt at huske komma imellem dem.
Og husk at give hver af billederne en værdi, så eks at den ene er i venstre top og andet billede er i højre top.

Placering af billede inde i en border.

Når man vil placere billedet, kan man anvende origin, da det hjælper os med at bestemme hvor i vores border det skal befinde sig

Man kan vælge at gøre så ledes:

```
#afsnit5{
    border-style: double;
    margin: 10px 0 10px 0;
    border-color: #7ad2f8;
    padding: 3px;
    background: url(blomst.jpg);
    background-repeat: no-repeat;
    background-origin: padding-box;
    background-size: 5em;
    background-position: right top;
    }
```

I eksemplet har vi altså valgt at billedet skal ligge helt op af den inderste border kant.

Man kan dog også bruge

Border-box = hvilket vil gøre sådan at billedet ligger sig til den yderste kant af ens border.

Content-box= det vil gøre således at det ligger sig mere ind imod teksten.

På den måde kan man altså justere billedet placering.

Sofie og Ida

Kilder:

https://www.w3schools.com/css/css_background.asp
https://www.w3schools.com/tags/tag_a.asp
https://www.w3schools.com/css/css_border.asp