2014

HTML OG CSS BEGYNDER KOMPEDIE

Kom godt i gang med HTML5 og CSS3

I dette kompendie bliver du præsenteret for grundlæggende HTML og CSS. Der er indlagt opgaver, som du selv skal løse ud fra oplysningerne i teksten.

Indhold

HTML og CSS

Basis opbygning af hjemmeside-dokument

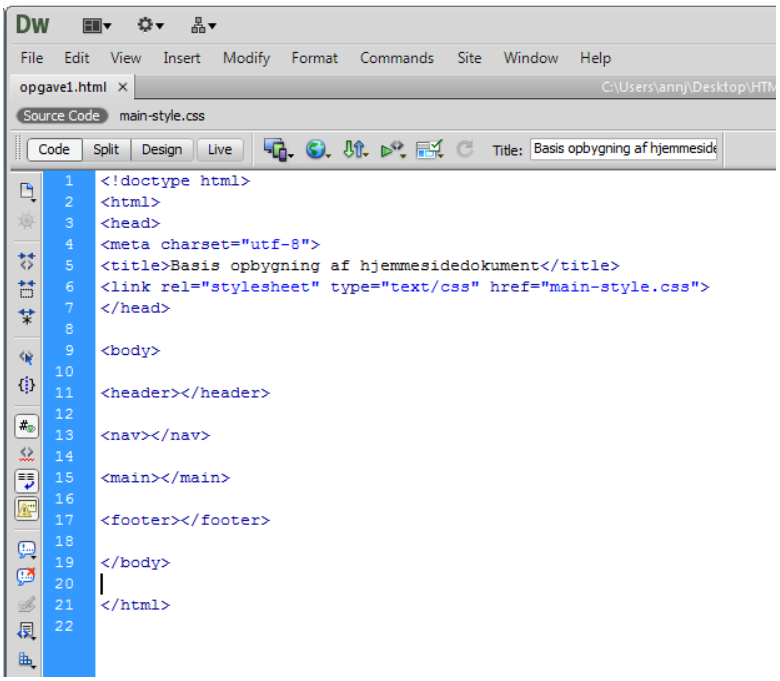
En hjemmeside kan bestå af et enkelt dokument… Men det mest almindelige er at, et **site** som en hjemmeside også kaldes, består af flere **html-dokumenter** som er forbundet med **links**.

I denne tutorial har jeg valgt at skrive site når der er tale om en samling af forbundne dokumenter – og hjemmeside når der er tale om det enkelte dokument.

Opgave 1.a

- Lav en mappe til dit site så der er styr på hvor dokumenterne bliver gemt. Navngiv den f. eks. ditNavn_html-css_start - hvor ditNavn erstattes med dit fornavn eller forbogstaver
- Åbn DreamWeaver og opsæt sitet. Brug dokumentet "Opsætning af site i DreamWeaver" til at hjælpe dig.

Dette er en grund-opbygning af en hjemmeside som det kan se ud på skærmen og i **HTML**-koden. HTML står for **HyperText Markup Language**, hvilket kan oversættes til noget i stil med: sprog til opmærkning af tekster til elektronisk overførsel. HTML er det sprog der fortæller browserne, hvordan strukturen – eller skellettet – på hjemmesiden er.



Hypertext er mere præcist tekst man kan aktivere og derved få yderligere oplysninger.

En **browser** er et program på din computer, som giver adgang til internettet. Browseren fungerer som et vindue, der viser oplysningerne fra de forskellige websites

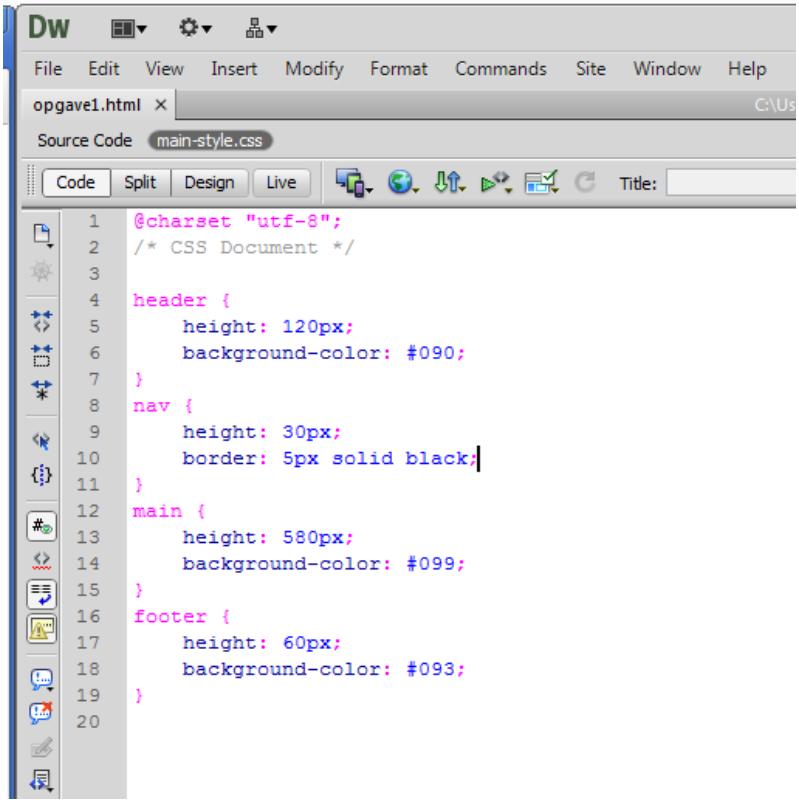
En hjemmeside bygges op af en masse **objekter** - også kaldet **elementer**. Et element skal åbnes <…..> og lukkes igen </…..>. Dette kaldes også for åbne- og lukke**tags**.

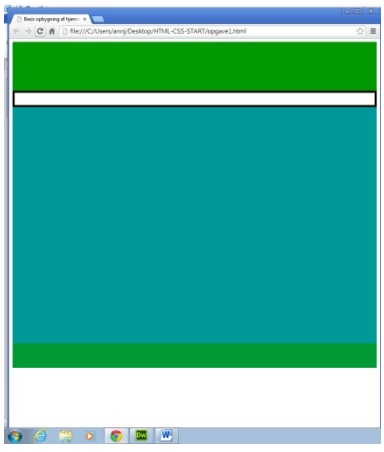
- <!doctype html> fortæller browseren den skal tolke teksten som skellet til en hjemmeside – det fortæller også hvilken generation af sproget der er anvendt.
- <html> fortæller: ” nu begynder dokumentet du skal oversætte”
- <head> fortæller: ” nu kommer der en masse som er vigtigt for browseren at vide – men det skal ikke vises for alle andre”.
- <meta charset=”utf-8”> fortæller hvilket tegnsæt browseren må oversætte til – med utf-8 må man gerne bruge æ, ø og å hvilket er ret smart hvis siden skal vises i Danmark.
- <title> her skal man fortælle hvad det er for en hjemmeside man er inde på. Titlen vises i fanebladet i browseren.
- <link rel=”stylesheet” >fortæller browseren at der er et andet dokument det lige skal snakke med, når det skal vise hjemmesiden på skærmen. Det andet dokument styrer nemlig hvordan skelettet skal se ud.
- <body> fortæller browseren; nu kommer det brugerne af hjemmesiden må se. Så det du gerne vil vise skal placeres imellem< body> -åbnetag og </body> -lukketag.
- <header>, <nav>, <main> og <footer> er alle struktur-tags som fortæller nu kommer der noget indhold som har et bestemt formål.

Den struktur du så på forrige side er endnu helt uden indhold. For at vi kan se den, har jeg linket til et dokument hvor der er beskrevet højder og baggrundsfarver for elementerne.

Herunder ses det dokument der styrer udseendet på sitet. Det kalder vi **CSS** – **Cascading StyleSheet**.

Selvom vi også her har sat utf-8 som tegnsæt, kan det ikke anbefales at bruge æ, ø eller å i dokumentet.





I CSS har vi:

- **Selectorer** (header, nav, main, footer) som fortæller browseren hvilket HTML-element der skal påvirkes
- { krølleparentes start (tuborgklammer, curly brackets) fortæller nu kommer der en blok af beskeder - **declarations -** om det element.
- } krølleparentes slut fortæller nu slutter declarations om det element.
- **Properties** (height, background-color, border) – egenskaber vi vil ændre udseendet på
- **Values** (120px, #093, solid, black) – den værdi hver enkelt egenskab skal have.

En selector fortæller hvilket element der må styles på. Det der skal ændres, skrives mellem { og } og kaldes declarations.
Declarations fortæller hvad der helt præcist skal ske med selectorens element. En declaration består af en property med mindst en tilhørende value. En property er en egenskab og value er den værdi egenskaben skal have.

Opgave 1.b

- Opret og gem et HTML med den samme struktur som ovenfor – kald det opgave1
- Opret og gem et CSS med de samme selectorer og properties. Values må du gerne selv bestemme.
- Forbind dokumenterne med hinanden og se det i browseren. Tag et kik på billedet side 1. I linje 6 kan du se hvordan det gøres.

Vi vil gerne have lidt luft imellem elementerne på hjemmesiden. Til det har vi en property der hedder margin.

Margin laver luft omkring elementet ved at skubbe til det element som står skrevet efter det i HTML-strukturen. Margin lægges til størrelsen på dit element, da der nu reserveres mere plads til det. Hvis der kun skrives én værdi sættes der samme margin på alle fire side af elementet.

Opgave 1.c

- Tilføj declarationen **margin: 10px;** til nav selectoren
- Tilføj declarationen **margin: 10px 0px;** til footer selectoren.
- Se i browseren hvad det gør. Er der noget der overrasker?
- ……..

<header>, <nav>,<main> og <footer> betegnes alle som block-elementer. Det betyder de fylder hele bredden og de laver et linjeskift når de afsluttes.

For at være sikker på alle browsere understøtter f.eks. <main> kan det være en fordel at tilføje det til CSS. Tilføj declarationen **display:block;** til main selectoren

At starte et spindelvæv

Så skal der bygges videre på hjemmesiden. Vi fortsætter med den samme mappe og det samme StyleSheet – men i et nyt html-dokument

Opgave 2.a

- Lav en kopi af dit HTML med grundstrukturen – gem det under navnet opgave2. Sørg for det er dette dokument du arbejder videre i.

En hjemmeside uden forbindelser til andre sider er ikke meget være i WWW (det verdens omspændende spindelvæv)

Derfor skal vi lave 4 links (forbindelser) til andre sider. <nav> elementet er beregnet til samlinger af links – derfor skal de placeres mellem navåbne- og navlukketagget.

Et link består af; **<a href**=”adresse.html”> noget tekst at klikke på

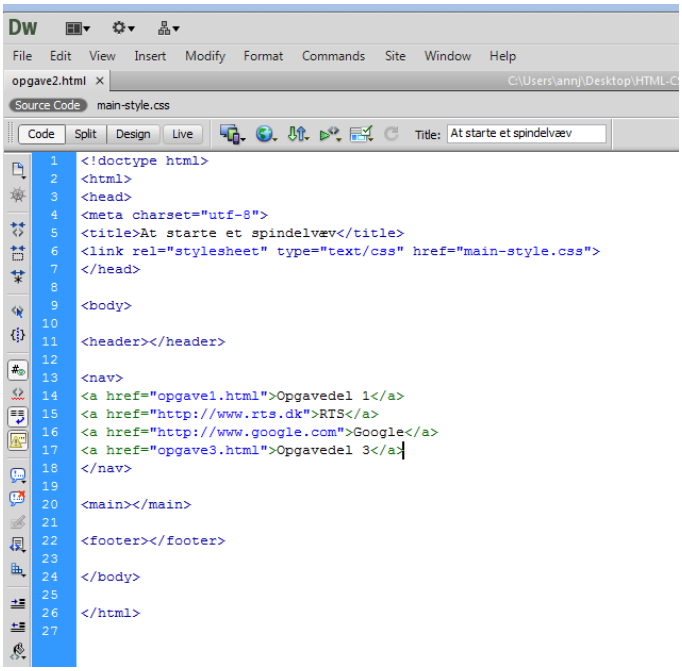
Når vi begiver os ud på nettet, er det trygt at have et udgangspunkt at vende tilbage til. Derfor kaster vi et anker <a> – så vi kan komme tilbage igen. Men vi skal også bruge en adresse at rejse til. Den skriver vi i attributten href. Teksten mellem åbne- og lukketagget er det vi skal trykke på for at magien sker.

Nav er forkortelse af navigation. **A** står for anchor. **Href** betyder ”hyper-reference”. En Attribut er en tilføjelse til et element. Der er forskellige attributter til forskellige elementer. Nogle af de mest almindelige er: href, src, alt, title og type

Opgave 2.b

- Opret 4 hyper-referencer i dit nav-element i HTML-dokumentet – se det i browseren
- Ser det ikke lidt gnidret ud?
- Opret en ny selector a i dit CSS som peger på alle dine <a>. Prøv at sætte baggrundsfarve på og ændre på farven på din tekst(color) – og sæt margin: 10px; på også.

```css
@charset "utf-8";
/* CSS Document */
* {
    margin: 0;
    padding: 0;
}
html {
    background-color: #FFC;
    font-size: 1em;
}
header {
    height: 120px;
    background-color: #090;
}
nav {
    height: 30px;
    border: 5px solid black;
    margin: 10px;
}
a {
    margin: 15px;
    /*padding: 2px 10px;*/
    background-color: #0C9;
    color: #FFF
}
main {
    display: block;
```

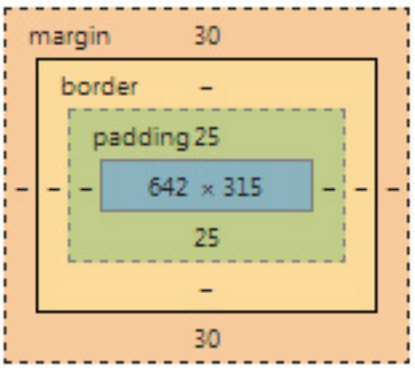


På illustrationen ovenover er der i CSS markeret en linje i deklarationen for a.
/* og */ markerer en udkommentering. Når noget er udkommenteret springer browseren det over. Man kan bruge det til små forklaringer om koden eller som her til at fjerne en property midlertidigt.

I CSS arbejder man ud fra en **box-model**. Den går i korte træk ud på at

width + padding + border = faktisk synlige bredde på boxen (elementet)

height + padding + border = faktisk synlige højde på boxen



Padding skaber luft – men det skubber til det der ligger inde i elementet

Border er en ramme rundt om dit element.

På illustrationen her kan du se hvordan en border placeres mellem margin og padding.

Som standard lægges værdien af padding og border også til størrelsen af din box – der reserveres mere plads ligesom med margin.

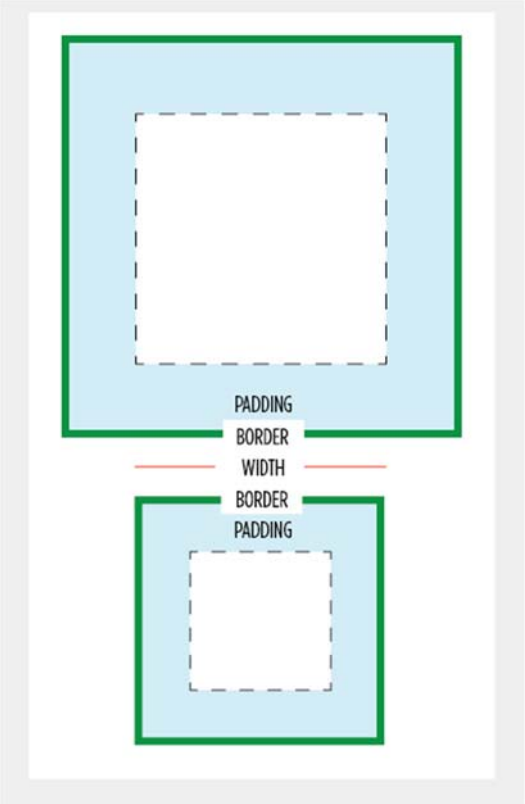
Denne box ender med at reservere 752px i bredden og 425px i højden. Havde der været angivet en værdi for border, skulle den også have været lagt til.

Opgave 2.c

- Sæt en declaration for padding på din a selector
- Sæt en declaration for padding på din nav selector og se forskellen i browseren
- Sæt to værdier på for padding og se forskellen i browseren.
- Gem dit arbejde

Som standard lægges værdien for padding og border altså til bredde og højde af din box. Det er en smule besværligt hele tiden at skulle trække fra og lægge til når man ændrer i en af værdierne.

Der er heldigvis hjælp at hente i CSS3. Det er en egenskab som kaldes **box-sizing.** Med den kan man selv bestemme om værdien skal lægges sig uden på dit element eller inden i det



Hvis ikke man ændrer på noget er standarden box-sizing: **content-box;**

Alle værdien gør din box større.

Hvis den værdi du angiver skal være den faktiske bredde, så skal du skrive;

box-sizing**: border-box;**

Her lægger padding og border sig inde i din box. Boxen bevarer den størrelse du har sat. Men vær opmærksom på du nu har mindre plads til dit indhold.

Margin bliver stadig lagt uden om din box og reserverer mere plads.

Der er flere opgaver med box-modellen i det kompendie som hedder Farvelader.

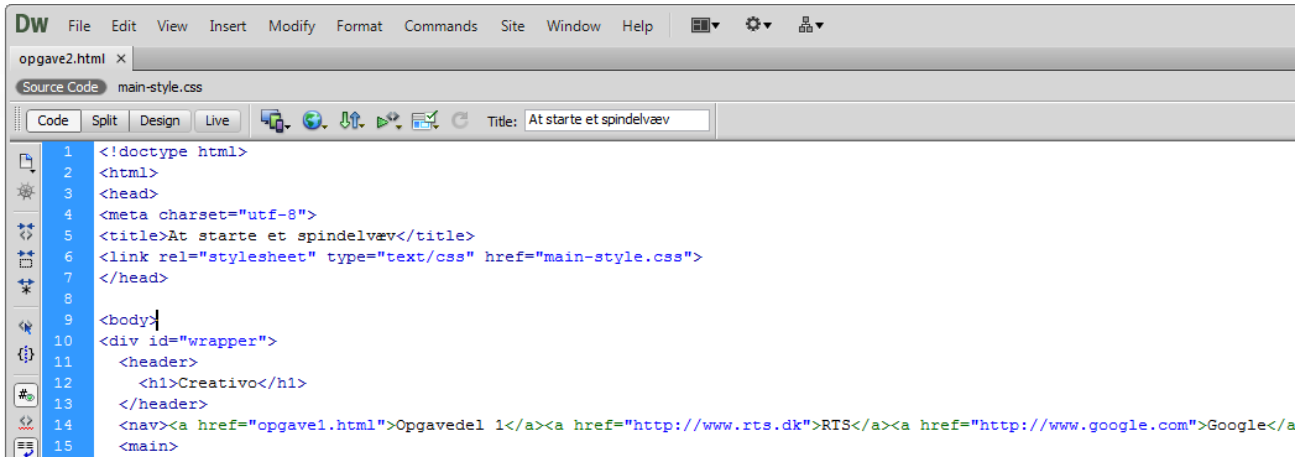
Placering af hjemmesiden på skærmen

Lige nu fylder vores side den fulde bredde af browseren. Det vil jeg gerne ændre på. Den skal kun være 960px bred – og så vil jeg have den skal stå pænt på midten af skærmens bredde.

Man kan gøre flere forskellige ting for at ændre på bredden.

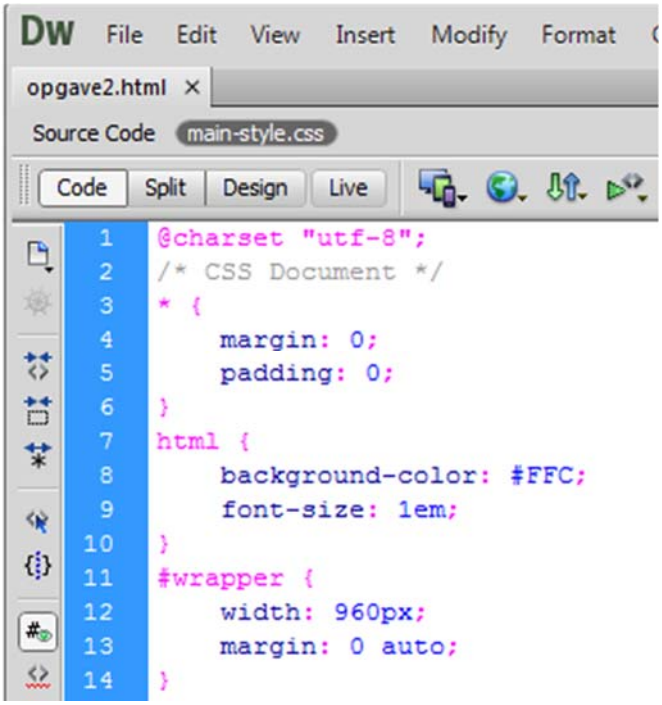
Den besværlige måde er at tilføje declarationen **width: 960px**; til alle selectorer i CSS.

Den smarte måde er at lave en boks rundt om de elementer man allerede har. Den giver man en **attribut** f.eks. **id**=”wrapper”**, så** man kan bruge den som selector i CSS. I CSS giver man sin id **width: 960px**;



Når man skal fortælle i CSS at declarationen gælder et id bruger man tegnet **#** f. eks. #wrapper{}
Vi kalder både tegnet for havelåge og hashtag

Det er absolut en fordel at man ikke skal nærlæse 600 linjer CSS for at være sikker på at ændre til samme værdi alle steder – derfor bruger vi den smarte måde. Det samme gælder når vi skal til at have lige meget luft på hver side af hjemmesiden.



Først er vi nødt til at nul-stille den margin og padding som alle browsere har som standard indstilling – det er desværre forskellige standarder.

Den hurtige løsning her er at oprette en selector som nulstiller på samtlige elementer vi kan finde på at bruge.

Vi kalder det **stjerne-reglen** fordi tegnet er ***** (asterisk)

Det upraktiske ved denne regel er, at vi nu er nødt til at deklarere hver gang vi vil have luft ved et element. Det problem klarer vi på et andet tidspunkt

I dette tilfælde har jeg også sat en lys baggrundsfarve på html. Den kommer til at være synlig på hver side af wrapper.

På wrapper er defineret bredden på hjemmesiden og så er der deklareret at der skal være 0 margin i top og bund – og auto margin i siderne. Når browseren kender bredden på elementet kan den selv fordele lige meget luft på hver side.

Opgave 2.d

- Opret en stjerne-regel. Du må gerne sætte en anden værdi på end 0 – men hold det under 25px.
- Sæt en baggrund på din html. Man kan bruge baggrundsbilleder eller mønstre som bliver gentaget. Se også f. eks. hvordan du kan kode et farveforløb i CSS
- Opret en div med tilhørende styling til at styre bredden på din hjemmeside.

Siden skal have noget indhold

Nu er tiden kommet til overskrifter, brødtekst og billeder.

Når man læser en avis er man ikke i tvivl om hvad der er vigtigst. Man kan se det på skriftens størrelse. Sådan er det også på hjemmesider. Overskrift hedder **Heading** på HTML og forkortes h. De får et nummer 1-6 hvor 1 er vigtigst – således hedder den vigtigste overskrift <h1>, den næst vigtigste <h2> og så der ned af.

Man må gerne have mere end en h1 på sin hjemmeside. Men kun en for hvert element, således at man må have en <h1> i sin <header>, en <h1> i sin <main> og en <h1> i hver <article>. Men man må ikke have 2 stk. <h1> i samme article. Hvis man har behov for flere hovedoverskrifter i en <article> skal man dele den op i sektioner. Hver <section> må nemlig også gerne have en <h1>.

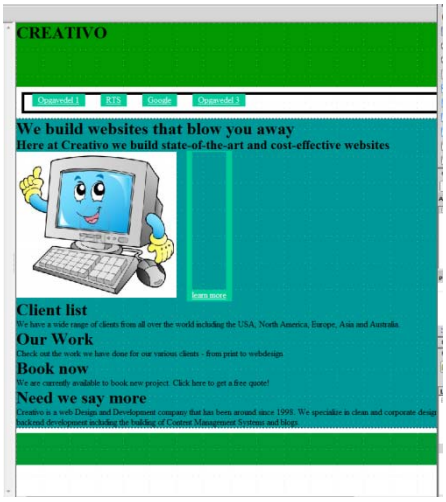
Det kan godt blive lidt uoverskueligt – men vi prøver lige engang.

Opgave 2.e

- Giv din hjemmeside en hovedoverskrift
- Giv din <main> en hovedoverskrift
- Opret mindst to styk <article> med hver en overskrift og noget brødtekst. Brug bare mumletekst. HUSK de skal placeres mellem <main> og </main>
- Sæt et billede ind i den ene <article >
- Opret et par <section> med hovedoverskrift i den anden <article>

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>At starte et spindelvæv</title>
<link rel="stylesheet" type="text/css" href="main-style.css">
</head>

<body>
<header>
  <h1>CREATIVO</h1>
</header>
<nav><a href="opgave1.html">Opgavedel 1</a><a href="http://www.rts.dk">RTS</a><a href=
"http://www.google.com">Google</a><a href="opgave3.html">Opgavedel 3</a> </nav>
<main>
  <h1>We build websites that blow you away</h1>
  <h2>Here at Creativo we build state-of-the-art and cost-effective websites</h2>
  <img src="computer-05.jpg" width="300px"/> <a href="#">learn more</a>
  <article>
    <section>
      <h1>Client list</h1>
      <p>We have a wide range of clients from all over the world including the USA, North America,
 Europe, Asia and Australia.</p>
    </section>
    <section>
      <h1>Our Work</h1>
      <p>Check out the work we have done for our various clients - from print to webdesign</p>
    </section>
    <section>
      <h1>Book now</h1>
      <p>We are currently available to book new project. Click here to get a free quote!</p>
    </section>
  </article>
  <article>
    <h1>Need we say more</h1>
    <p>Creativo is a web Design and Development company that has been around since 1998. We
specialize in clean and corporate design, usability, front and backend development including the
building of Content Management Systems and blogs.
  </article>
</main>
<footer></footer>
</body>
</html>
```



Sådan her kan det se ud før der er ordnet noget i CSS.

Nu har vi <h1> fire steder i hjemmesiden – de skal alle se forskellige ud. Derfor er vi nødt til at **specifisere** det. Det gør man i CSS ved at skrive flere selectorer på stien ind til elementet. Man skriver dem med et mellemrum mellem hver selector.

```css
@charset "utf-8";
/* CSS Document */
* {
    margin: 0;
    padding: 0;
}
html {
    background-color: #FFC;
    font-size: 1em;
}
header {
    height: 120px;
    background-color: #090;
}
h1 {}
header h1 {
    text-transform: uppercase;/* sørger for alt er store bogstaver*/
    font-size: 500%;
    color: #FFF;
    text-align: center;
}
main h1 {
    font-family: Georgia, "Times New Roman", Times, serif;
}
article h1 {
    color: #900;
    font-size: 100%;
}
section h1 {
    color: #006;
    font-size: 200%;
}
h2{}
p{}
nav {
    height: 30px;
    border: 5px solid black;
    margin: 10px;
}
```

Når du løser næste opgave så prøv at se i browseren hver gang du har lavet en ændring – eller brug split-view og refresh hver gang, så kan du følge med.

Du vil så se hvad C´et i CSS reelt betyder;
Cascading = når noget flyder over og påvirker det det passerer på vej ned.

Hvis du taster noget ved h1{} så arver alle de sammen den egenskab.

Når du taster noget i main h1{} så arver både article h1 og section h1 den egenskab.

Prøv også at lægge mærke til hvilke selectorer der påvirker hele hjemmesidens bredde – og hvilke der kun påvirker lige omkring sig.

Opgave 2.f

- Gør dine <h1> forskellige alt efter om de er i <header><main><article> eller <section>
- Få linket i <article> til at se anderledes ud end links i nav
- Hvis du har en <h2> så style den. F. eks. color eller font-style
- Sørg for at alle dine <p> har samme style

Bokse som ligger ved siden af hinanden

Opgave 3.a

- Lav en kopi af opgave2.html – kald kopien opgave3.html
- Lav også en kopi af din CSS – kald kopien standard-style.css
- Ret linket i <head> så det er det rigtige stylesheet der snakkes med
- I standard-style.css ændrer du width på wrapper til 1000px

Nu skal du prøve at lave en navigation som er i venstre side, med indholdet ved siden af.
Du skal også til at arbejde med procenter.

Når vi arbejder med % kan vi bruge decimaltal (komma-tal).
% udregnes i forhold til **forældre-elementets** bredde.

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>DOM opbygning af hjemmesidedokument</title>
</head>

<body>
<div id="wrapper">
  <header>
    <h1></h1>
  </header>
  <nav><a href="#">Et</a><a href="#">To</a><a href="#">Tre</a><a href="#">Fire</a></nav>
  <main>
    <h1></h1>
    <h2></h2>
    <img src="#" alt="her kommer der et billede">
    <article>
      <section>
        <h1></h1>
        <p></p>
        <p></p>
      </section>
      <section>
        <h1></h1>
        <p></p>
        <p></p>
        <p></p>
      </section>
      <section>
        <h1></h1>
        <p></p>
        <p></p>
      </section>
    </article>
    <article>
      <h1></h1>
      <p></p>
      <p></p>
      <p></p>
      <h2></h2>
      <p></p>
      <p></p>
    </article>
  </main>
  <footer></footer>
</div>
</body>
</html>
```

Her er grund – opbygningen fra opgave1.html udbygget med elementer fra opgave2.html.

Og sådan ser den stadig ud – når al teksten er fjernet.

Her kan du tydeligt se forældre-barn elementerne. Hver gang der er tabbet ind er der et barn.

Når wrapper er 1000px bred er det let at lave omregningen;

2px omregnes til 0.2%	10px
til 1%	15px
til 1.5%	

Samtidig ændrer vi margin og padding til % også.

Opgave 3.b

- Du starter med at udkommentere height på nav og main.
- Det erstatter du med width i stedet. Det kan f.eks. være 30% på nav og 70% på main
- Sæt box-sizing : border-box; på nav og main
- Skift margin i nav ud med padding
- Ret også lidt i margin og padding på main så der kommer lidt luft mellem baggrundsfarven og teksten inde i boksen.

Boxene står stadig under hinanden ikke!

Opgave 3.c

- Giv main og nav declarationen **float:left;**

VUPTI så lagde de sig pænt ved siden af hinanden. Men HOV, vores <footer> hoppede helt op under vores <header>

Når man bruger **float** så tager man elementet ud af dets flow, hvilket påvirker elementerne der kommer efter i HTML. Man kan populært sige at de floatede elementer bliver skubbet ud af køen.

Som små børn søger de at komme så tæt på deres forælder – her er det nærmeste søskende element uden float <header> og de placerer sig lige efter det.

Måden vi fikser det på er en property som hedder **clear.** Hvis elementerne er float:left alle sammen kan man skrive **clear:left;** Hvis der både er float:left; og float:right; skal man skrive **clear:both;** for at få det på plads igen.

Opgave 3.d

- Giv footer deklarationen **clear:left;**
- Se din side i browseren

Nu skulle din side gerne minde om denne. Hvis du har meget tekst, vil footer måske ikke være synlig og der vil være kommet en scroll-bar i højre side af skærmen.

Der mangler at blive rettet op på navigationen.

Lister

Lige nu har dine links i navigationen placeret sig ved siden af hinanden. Det er fordi <a> er et inline-element som ikke laver et linieskift.

Det er god fornuft at lave en liste til at samle sine interne links i. Søgemaskinernes robotter bruger bl.a. strukturen til at vurdere sidens vigtighed.

Der findes tre tags til at oprette lister med.

- = ordered list: en liste hvor første punkt er vigtigere end andet punkt – punkt-tegnene kan enten være tal eller bogstaver
- = unordered list: en liste hvor alt er lige vigtigt – punkt-tegnene kan være forskellige figurer eller et billede
- <dl> = description list: en liste hvor noget skal beskrives – minder en del om en tabel i strukturen

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>

<ol start="50">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

```
<dl>
  <dt>Coffee</dt>
    <dd>Black hot drink</dd>
  <dt>Milk</dt>
    <dd>White cold drink</dd>
</dl>
```

Opgave 4a

- Lav en kopi af opgave3.html og gem det som opgave4.html – sørg for at arbejde videre i det nye dokument. Der fortsættes med det samme stylesheet.
- Opret endnu et link – det skal henvise til opgave3.html
- Lav en liste uden om dine links – inde i <nav> - gem og se det i browseren
- Skift din liste ud med en anden liste <ol> skiftes til

Nu bør den del af din html ligne dette.

```
</header>
<nav>
<ul>
<li><a href="opgave1.html" title="link til opgave 1">Opgavedel 1</a></li>
<li><a href="http://www.rts.dk" title="link til RTS">RTS</a></li>
<li><a href="http://www.google.com" link til Google>Google</a></li>
<li><a href="opgave2.html" title="link til opgave 2">Opgavedel 2</a></li>
<li><a href="opgave3.html" title="link til opgave 3">Opgavedel 3</a></li>

</nav>
<main>
```

Du kan få DW til at hjælpe dig med at holde orden i din kode. Klik og vælg *apply source formatting*

Men der skal ændres på udseendet. Det ser jo ikke super godt ud.
Man kan lige skimte prikkerne i borderen og "knapperne" har forskellige længder.

At style på navigationen

Når du skal style på navigationen er der flere elementer i spil. <nav>, <ul>, <li> og <a>. De har hver sin opgave i forhold til strukturen – og det rammer også i CSS.

For at fjerne prikkerne foran hvert liste-punkt skal man bruge declarationen list-style med værdien none. Du kan også bruge list-style-type – de gør det samme og har samme værdier. Du kan placere den på både ul og li.

Opgave 4b:

- Fjern prikker på listen
- Sæt font-family , font-size og color på nav ul til noget andet end det du ellers har
- Se i browseren.

Når du bruger nav ul som selector er du sikker på udelukkende at ramme de <ul> som er barn til <nav>. Det vil sige at har du en <ul> inde i <main> vil den ikke blive ramt.

Prikkerne forsvandt – men der skete ikke noget med teksten – ØV

Opgave 4c:

- Klip (ctrl+x) styling for font-family , font-size og color
- Paste (ctrl +v) det ind i en selector for li
- Se i browseren

Stadig ingen ændring

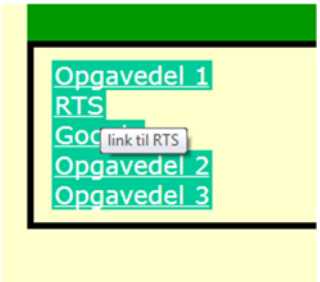
Når man vil påvirke den tekst man klikker på i et link er det <a> man skal ramme.

Opgave 4d:

- Opret en selector for teksten på dine links i listen i nav
- Flyt stylingen derhen
- Se i browser

Virkede det nu?

Min side ser således ud i browseren.



```css
nav {
    /*height: 30px;*/
    box-sizing: border-box;
    width: 30%;
    float: left;
    border: 5px solid black;
    padding: 1%;
}
nav ul {
    margin: 0;
    padding: 0;
    list-style: none;
}
nav ul li {
    width: 90%;
}
nav ul li a {
    font-family: Verdana, Geneva, sans-serif;
    font-size: 1.2em;
}
a {
    margin: 1.5%;
    padding: 0.2% 1%;
    background-color: #0C9;
    color: #FFF
}
```

I CSS ser det således ud.

Som I kan se bliver mine links i nav stadig påvirket af den generelle regel for a. Hvis der skal være en egenskab med forskellige værdier er du nødt til at deklarere det begge steder.

Men… Der er stadig forskellig længde på linksene. Du kan selvfølgelig sætte baggrundsfarve på li – så får de samme længde, men du kan stadig kun klikke på teksten i linket.

Du fik tidligere at vide at <nav>, <ul> og <li> var block elementer som laver et linieskift – og <a> var et inline element som ikke laver linieskift. Det kan man lave om på med CSS med egenskaben display.

Det vil også se bedre ud hvis der er mellemrum mellem rækkerne af links. Skal det mon gøres med margin eller padding?

Opgave 4e:

- Sæt dine links i nav til display:block;
- Sørg for at der kommer luft mellem rækkerne af links (margin eller padding?)
- Leg med forskellige egenskaber for ul og li i din nav
- Fjern stregen under links i nav (text-decoration:none;)
- Leg med forskellige egenskaber for a i nav

Til slut vil vi gerne have der sker noget med vores links når vi kører musen over dem. Det gør man ved at tilføje en pseudoklasse til sit element i CSS.

Man laver en ny selector f.eks. a:hover ved et link eller img:hover hvis det er et billede. Så kan man sætte nye værdier for -baggrund eller –tekst farve.

Hover udtales ”håvver” og ikke ”huver”

Opgave 4f:

- Lav en hover på links i navigationen
- Lav en anden hover på links i main.

Der er flere opgaver med opbygning og styling af navigation i det kompendie som hedder ”HTML og CSS navigation_listemenu opgaver”