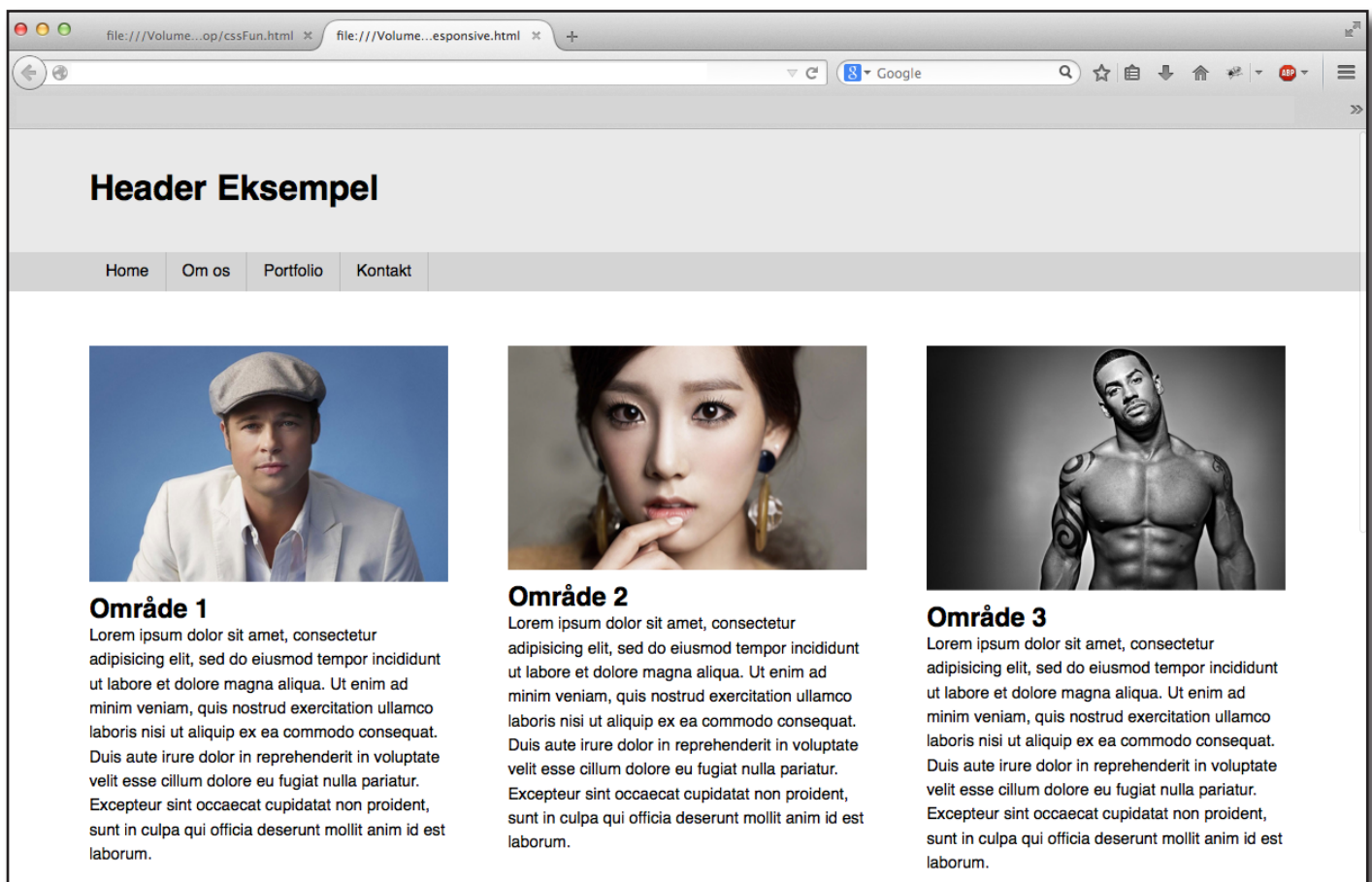


Intro til responsive design



Vi skal starte med at arbejde med Responsive webdesign.

Et mere teknisk udtryk vil være at vi skal arbejde med en introduktion til:
"Media Queries"

Media Queries giver os muligheden for at bruge flere CSS dokumenter på samme side afhængig af skærmstørrelsen.

Lad os komme igang!

Du har modtaget en mappe som hedder "Intro_Responsive" Den indeholder:

-----#.#-----#.#-----#.#-----#.#-----#.#-----#.#-----#.#-----#.#-----

responsive.html

--CSS-- »

main.css

responsive.css

--images-- »

boy01.jpg

boy02.jpg

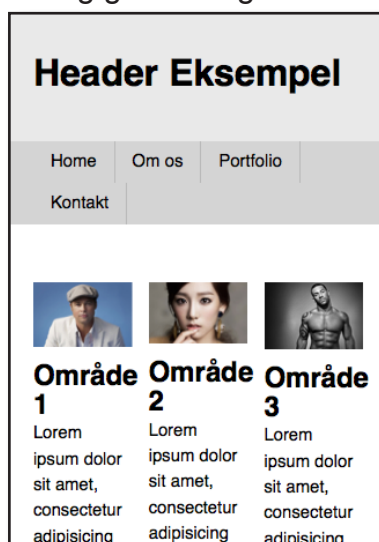
girl01.jpg

-----#.#-----#.#-----#.#-----#.#-----#.#-----#.#-----#.#-----

1

Start med at åbne "responsive.html" i din fortrukne html editor. Lav et preview i browseren, og som du kan se så er det en almindelig fungerende hjemmeside.

Hvis du gør skærmen mindre, så tilretter kolonnerne sig godt nok efter skærmen. Det ser bare ikke særlig godt ud og vil absolut ikke fungerer på en tablet eller smartPhone.



2

På billedet herunder ser du den HTML der skalber siden. Det er en simpel side, og som du kan se så er der allerede linket til et CSS dokument.

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">

  <title></title>

  <meta name="viewport" content="width=device-width">

  <!-- css -->
  <link rel="stylesheet" href="css/main.css">

</head>
```

3

Du skal nu lave et link til et ny CSS dokumen som kan hjælpe os med at gøre designet bedre på mindre skærme. CSS dokumentet skal dog først blive aktivt når vi specielt ønsker det, og det gør vi ved at kalde den skærmstørrelse som aktiverer CSS dokumentet.

```
<!-- css -->
<link rel="stylesheet" href="css/main.css">
<link rel="stylesheet" href="css/responsive.css" media="screen and (max-width: 900px)">
```

Vi kalder på et nyt CSS dokument som du navngiver "responsive.css"

Nu bruger du attributten **media=" "** og giver den værdien som på billedet.

Max-width:900px gør at CSS dokumentet kun er aktivt når skærmen er mindre end 900px i bredden.

I din CSS mappe laver du en ny fil og navngiver den "responsive.css"

Åben den i din editor, så du er klar til at lave responsive design.

4

I dit nye CSS dokument laver du en værdi for "body" så farven på teksten bliver rød.

```
body{  
  color: red;  
}
```

Gem! Se resultatet i browseren når du har genindlæst dokumentet.
Gør dit browser vindue mindre end 900px i bredden os se resultatet.

Header Eksempel

Home Om os Portfolio Kontakt



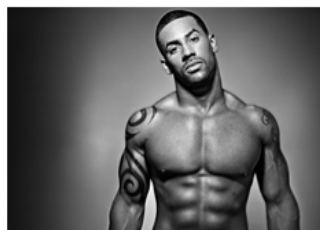
Område 1

Lorem ipsum dolor sit amet,
consectetur adipisicing elit, sed do
eiusmod tempor incididunt ut



Område 2

Lorem ipsum dolor sit amet,
consectetur adipisicing elit, sed do
eiusmod tempor incididunt ut



Område 3

Lorem ipsum dolor sit amet,
consectetur adipisicing elit, sed do
eiusmod tempor incididunt ut

Det er selvfølgelig ikke et resultat vi skal bruge til noget, men kun en måde at se vi har forbindelse til CSS dokumentet og at vores begrænsninger på skærmstørrelsen virker.

Så sætningen i CSS dokumentet er ret vigtig. Tænk på det som et spørgsmål til browseren.
HVIS skærmen har en **max-width** på **900px** (altså mindre end 900px) så skal CSS dokumentet aktiveres ELLERS skal CSS dokumentet være inaktiv.

5

Lad os så komme igang med at skrive noget vi kan bruge.

Det første vi skal er at lave top sektionen om til 2 kolonner i stedet for 3, når skærmen er mindre end 900px.

Lad os tage et kig på hvad det er der laver 3 kolonner for os i HTML'en

```
<!-- one-third -->
<div class="one-third mobile-collapse">
  
  <h2>Område 1</h2>
  <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed
    aute irure dolor in reprehenderit in voluptate velit esse cillum
  </div><!-- one-third -->
```

Som du kan se så er det en almindelig div med en **class="one-third"**
Lad os se på hvad den gør i "main.css"

```
div.one-third {
  width: 30%;
  float: left;
  margin-right: 5%;
}

div.one-third-last {
  margin: 0;
}
```

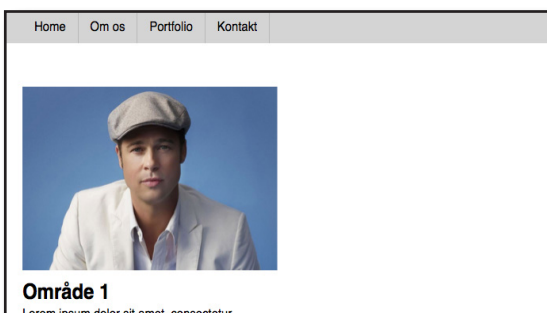
Som du kan se så opdeler den bare siden med 3 kolonner på hver **30%**
Egentlig skal de være **33%**, men vi vil også gerne have lidt luft mellem dem så de er **30%** med en **margin-left** på **5%**. Vi skal selvfølgelig altid forsøge at ramme **100%** i bredden, derfor har den sidste en class med **margin: 0**; Når de er sat til "**float: left;**" så flyder de pænt på plads ved siden af hinanden.

6

For at få 2 kolonner skal vi altså lave om i % som styrer one-third.
Derfor skriver vi i vores nye "responsive.css" (HUSK! slet body farven, det var kun en test)

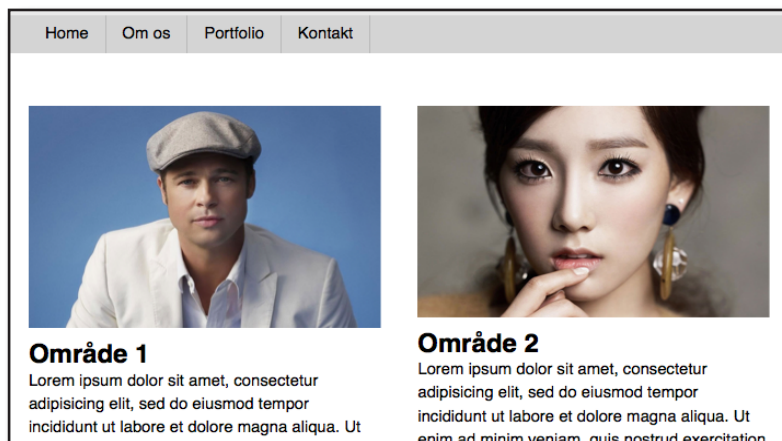
```
div.one-third {
  width: 47.5%;
}
```

Hvis du ser det i browseren, så kan du se at de 3 kolonner er væk, men der er et lille problem.



Kolonne nummer 2 har også en margin så den skal fjernes.

```
div.one-third{
  width: 47.5%;
}
div.one-third-second{
  margin: 0;
}
```



Så passer det med 2 kolonner i bredden. Kolonne nummer 3 ser bare ikke så fornuftig ud, så den skal vi have gjort noget ved.

```
div.one-third-last{
  clear: both;
  float: none;
  width: auto;
  padding: 20px 0 0 0;
}
```

Vi giver den **clear: both;** det holder float på plads ovenover den, så fjerner vi **float:** fra den selv, gør den til fuld bredde med **width:auto;** og giver lidt plads ovenover med **padding:**



Når billederne sidder ordenligt i designet, så er det ikke noget de gør af sig selv. Det er et lille stykke kode i "main.css" der sørger for det.

```
img {  
  max-width: 100%;  
  height: auto;  
  margin: 0 0 10px 0;  
}
```

Billederne får en **max-width:100%**; så de altid bliver indenfor vores design, højden bliver styret automatisk i forhold til bredden med **height:auto**;

Prøv at udkommenterer koden her og se resultatet på dit design. --

Som du kan se, så bruger jeg store billeder for at de kan bruges responsivt.

7

Nu har vi et Tablet-design færdigt, men vi skal også bruge et design med kun en enkelt kolonne, som egner sig bedre til Smartphones.

Jeg har vist dig en måde at inkluderer et nyt css dokument på. Her er så en anden:

```
@media screen and (max-width: 500px){  
  
}
```

Vi skriver simpelthen en media Querie i css dokumentet, hvor vi kalder at den skal virke på alle skærme der max har en bredde på 500px eller mindre.

Den kode vi skriver indenfor denne querie er først aktiv når reglen er opfyldt.

så du skriver:

```
@media screen and (max-width: 500px) {  
  
  div.mobile-collapse {  
    width: auto;  
    float: none;  
    margin-right: 0;  
  }  
  
}
```

Refresh din browser, flyt størrelsen ned under 500px i bredden og se effekten.

Vi har her brugt en hjælper-class som er givet til alle div's der skal have specielle indstillinger med kun en kolonne. Du kan finde dem i HTML dokumentet og se hvor de virker.

8

Vi har flere muligheder for at gøre siden rigtig responsive i designet. F.eks. faktaboksene i bunden af designet. Den blå og de to gule. Den blå er vigtig information, men de gule er ikke så vigtige, så dem kan vi godt fjerne når designet kun er på en kolonne. De har en **class** ved navn **hide-mobile** som ikke har været aktiv endnu, for de må jo gerne være med når der er plads til det, men nu kan vi skrive **display:none**; og fjerne dem ved skærmbredde under **500px**.

```
@media screen and (max-width: 500px) {  
  
  div.mobile-collapse {  
    width: auto;  
    float: none;  
    margin-right: 0;  
  }  
  .hide-mobile{  
    display: none;  
  }  
}
```

9

Nu skal vi lige kigge lidt på navigationen.

Navigationen følger ikke særlig godt med når skærmen bliver mindre, vi skal ikke ret langt ned før de sidste menupunkt går på anden linje og det ser ikke så heldigt ud.

Så et sted i responsive.css hvor vi kalder elementer under 900px kalder vi en anden bredde på menuen.

```
div.nav ul li{  
  width: 25%;  
  border-bottom: 1px solid #bababa;  
}
```

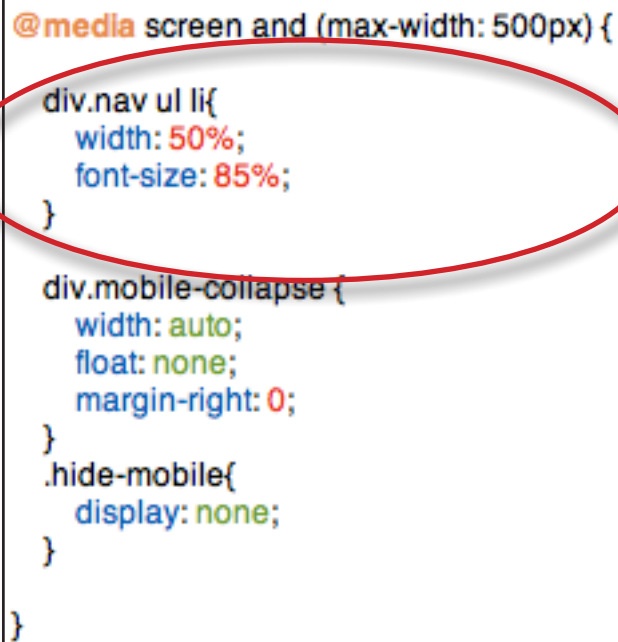
Med **width:25%**; sikre vi os at der kun er 4 menupunkter på hver linje. Og for at adskille dem fra hinanden så får de en **border-bottom**:

Det ser rigtigt ud nu, men når skærmen bliver mindre så kommer der rod igen. så det må vi ordne på ny.

Så vi skriver lidt kode til at styre menuen når vi kommer under 620px i bredden

```
@media screen and (max-width: 620px){  
  div.nav ul li{  
    width: 33%;  
  }  
}
```


Nu har vi 3 menupunkter på hver linje og mangler kun at ordne menuen når vi når under 500px i skærmbredden.



```
@media screen and (max-width: 500px) {  
  div.nav ul li {  
    width: 50%;  
    font-size: 85%;  
  }  
  
  div.mobile-collapse {  
    width: auto;  
    float: none;  
    margin-right: 0;  
  }  
  .hide-mobile {  
    display: none;  
  }  
}
```

Nu laver vi kun plads til 2 menupunkter på hver linje, og gør fonten en lille smule mindre.

Og nu da navigationen er på plads, så er arbejdet vist færdigt. Du har skrevet et responsivt web-design.

```
1  div.one-third{
2      width: 47.5%;
3  }
4  div.one-third-second{
5      margin: 0;
6  }
7
8  div.one-third-last{
9      clear: both;
10     float: none;
11     width: auto;
12     padding: 20px 0 0 0;
13 }
14
15 div.nav ul li{
16     width: 25%;
17     border-bottom: 1px solid #bababa;
18 }
19
20 @media screen and (max-width: 620px){
21     div.nav ul li{
22         width: 33%;
23     }
24 }
25
26 @media screen and (max-width: 500px) {
27
28     div.nav ul li{
29         width: 50%;
30         font-size: 85%;
31     }
32
33     div.mobile-collapse {
34         width: auto;
35         float: none;
36         margin-right: 0;
37     }
38     .hide-mobile{
39         display: none;
40     }
41
42 }
43
```

Responsive.css