|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Visualisering  V3-M1-F-2020 | | | |  |
| **Emne:** | SASS/SCSS | **Opgave:** | Matematiske operatorer |
| **Formål:** | * Jeg kan anvende matematiske operatorer i SCSS | | |
| * Jeg kan anvende pseudoklasser | | |
| * Jeg kan anvende nested SCSS | | |

**Opgave 4 – SCSS og matematiske operatorer**

I denne opgave skal du bruge matematiske operatorer til at lave en generisk kode, som kan anvendes til at lave et billede grid, hvor du selv bestemmer, hvor mange kolonner, du vil have i dit grid, og så udregner SCSS automatisk størrelsen på billederne og margenen.

1. Opret en ny index.html fil og en tilhørende style.scss
2. Start med at linke til dit stylesheet. Husk, at du skal linke til style.css
3. I HTML filen skal der øverst være en h1 med indholdet ”Image grid”
4. Herunder opretter du en div med klassen ”imagegrid”
5. I din imagegrid div skal du oprette en ny div med klassen ”item”
6. I din item div skal du oprette et billede
   1. Brug placeholder.com til at indsætte et placeholder billede, som er 700 x 700 px stort. Det kan du gøre ved at sætte følgende som image source:  
      ”https://via.placeholder.com/700x700”
   2. Angiv teksten ”thumbnail” som alternativ tekst
7. Kopier og indsæt din item div (med billedet i), så du ender med at have 13 item div’er lige efter hinanden
8. I din style.scss skal du starte med at lave en CSS reset (sæt margin og padding til 0 på dit root element)
9. Find en eller to fonte, som du vil bruge til din side på google-fonts og importer dem ind i din .scss fil (jeg har valgt at bruge PT Sans Narrow til min tekst og Ubuntu til mine overskrifter, dem må du også gerne bruge, hvis du vil)
10. Læg din font til almindelig tekst i variablen $font-main og læg din font til overskrifter i variablen $font-headings
11. Henvis til dine variabler i hhv. dit body tag og dit h1 tag
12. Opret derudover to variabler til farver. Variablen $color-bg skal indeholde farven hsl(40, 43%, 82%) og variablen $color-heading skal indeholde farven hsl(220, 43%, 30%)
13. Tildel din body baggrundsfarven vha $color-bg variablen og tildel din skriftfarve i h1’ere farve vha variablen $color-heading
14. Giv din body en padding på 1rem på alle sider
15. Giv din h1 en padding på 2rem i bunden
16. Du skal nu lave en mixin, som kan udregne størrelsen på billederne og margenerne afhængigt af, hvor mange kolonner, der er, og hvor stor en margin, man vil have
    1. Giv din mixin navnet imagegrid og lav plads til at der kan overføres to argumenter til mixin’en: $columns og $margin
    2. Først skal vi udregne bredden på hvert billede i vores imagegrid:  
       Bredden på hvert billede er lig med den fulde bredde minus den margen, der skal være imellem billederne divideret med antal billeder i kolonnen – se illustrationen nedenfor.  
         
         
         
         
         
         
         
       Prøv at se, om du selv kan lave den udregning, der skal til for at finde bredden på hvert billede, som du skal sætte propertyen width lig med.  
       Giv dig selv et begrænset tidsrum til at lave udregningen, så den bliver generisk og kan bruges til alle overførte argumenter i $columns og $margin. Hvis ikke du har fundet den indenfor dit afsatte tidsrum, så gå ind og få løsningen ved Hanne (dette er ikke et matematik-kursus men et scss-kursus 😉)

$margin

Bredde af hele elementet: 100%

* 1. Vi skal nu sætte margenen til højre og i bunden til at være den overførte værdi fra variablen $margin. Dette skal gøres på alle vores elementer med klassen ”item”. Til det skal vi bruge pseudoklassen :nth-element på det overliggende element. Når vi vil have vores :nth-element pseudoklasse til at slå igennem på alle elementer, overfører vi blot argumentet n til :nth-element (:nth-element(n))
  2. Det eneste element, som vi ikke vil have margenen til at slå igennem på, er det sidste element i hver kolonne. Det sidste element i hver kolonne er elementet, som svarer til værdien i variablen $columns. Man kan i SCSS overføre variabler til fx pseudoelementer ved at anvende interpolation, som skrives #{$variabelnavn}. Så du skal igen kalde :nth-element pseudoklassen på det overliggende element men denne gang med argumentet #{$columns}, så du på det sidste element i hver række, sætter både højre margen og bund margen til 0

1. Du skal nu style på imagegrid klassen, hvor du bl.a. skal kalde din mixin
2. Først skal du oprette din item klasse som nested element. Her sætter du først elementet til at floate left og derefter kalder du din mixin. Start med at teste den med 5 kolonner og 1% i margen
3. I din imagegrid klasse skal du også style på img selectoren som nested element. Lav afrundede kanter på 15px og sæt max bredden til 100%, så du er sikker på, at billedet altid udfylder hele det overliggende element
4. Gem din SCSS fil og tjek siden i en browser
5. Eksperimentér med forskellig antal kolonner og forskellig margen
6. Din mixin kan også bruges til at lave forskellig opsætning i et responsivt design
   1. Opret en mediaquery til desktop visning, som gælder for viewports, der er bredere end 600x
   2. Når du er i desktop visning, skal din mixin kaldes med 7 kolonner og 2% margen
   3. Når du er i mobil visning, skal din mixin kaldes med 3 kolonner og 1% margen
   4. Gem din scss fil og kontrollér i en browser, at det virker både med 7 kolonner og 3 kolonner (resize din browser for at teste om det virker)