**Cascading Style Sheets**

CSS /Cascading Style sheet er det kodesprog man bruger til at style sin HTML dokumenter. Dette gør arbejdet nemmere at skille ad og arbejde uafhængigt af hinanden.

Stylingen af HTML foregår langt de fleste gange via stylesheet reference indlejring. Og er det denne der gør at man har nemmere ved at skulle skille tingene ad. Man kan også embedded via <style> tagget og den sidst og mindst brugte er via inline.

De forskellige måder at style på har deres egne positive samt negative sider.

Inline er typisk set ned på, da det for det første bryder god kodeskik, men også ødelægger hele pointen i at have layout og struktur separat. Udover det kan det også være svært at style hele siden, hvis HTML inline stylingen kommer i konflikt med CSS stylingen.

Forskellen på embedded og inline, er at <style> tagget er også i HTML’en, men det befinder sig I <head> tagget i stedet og har kun effekt på det dokument den befinder sig i. Og det er lidt et problem hvis man har flere sider der skal styles. Og så skal det derved kopieres ind I hver enkel head på hver enkel side.

Stylesheet tilgangen er den anbefalede/Mest udbredte måde at style sin HTML via CSS. Det gør arbejdet at skulle style en hjemmeside med flere undersider på engang. Det gøres ved at benytte et href link I HTML’en med en attribut i form af ”stylesheet” og på den måde definere medietypen.

**CCS SYNTAX/SYNTAKS**

CSS Syntax er som HTML bygget op omkring simple regler. De starter med en selector der indikere hvad der præcis skal styles. De mest tilbagevendende er Element der beskriver en type af selector f.eks <p> som er en simpel selector. Der findes også .class som beskriver og rammer alle der har samme klasse. Det kunne eksempelvis være .template så class=”template” ville blive berørt. #id bruges til at tage et specifikt element med det pågældende ID.

Når seletor er valgt, skal man bruge curly brackets/Turborg klammer { til at starte med og } til at

slutte den.

Inde i selve klammerne kan man derefter tilføle egenskaber og tilføje værdier til disse.

De skal adskilles med et kolon som bliver referet til som property/Value separartor og afsluttes altid med et semikolon/declaration speperator. Man kan sagtens have flere deklarationer under samme selector, og derfor er det vigtigt at huske en separator.

**[Billede CSS Syntax]**

**Cascading**

Cascading eller som navnet angives på dansk kaskade er betegnelsen for at noget passere ned af, i en række af andre. Nemmeste måde at sige det på er at det er en nedarvning. Reglerne bestemmer hvilke deklarationer der er prioriteret over andre.

For det meste er det soruce order / Kilde rækkefølgen. Dvs den sidste deklaration er den der har effekt. Det betyder at hvis 2 deklarationer der rammer det samme element, vil det være den sidste af de 2 der tager prioritet.

**[Billede Cascading]**

På det viste eksempel ville det betyde at selvom de begge rammer h1 klassifiseringen, er det kun den sidste der har effekt, og derfor ville denne være sort. Hvis vi byttede rundt på gul og sort, ville teksten istedet være gul.

**CSS Grid**

Grid er nok et af de mere udbredte typer af styling strukturen som bliver brugt idag. Det har gjort opsætning og design af websites mange gange lettere at overskue, da det som navnet indikere er et grid/gitter. Man kan indsætte elementer på kryds og tværs i hhv kolonner og rækker og styre hvor indhold skal sidde på samme tid.

Man bruger grid ved at aktivere det i CSS’en med ”display:grid” imens det befinder sig I deklarationen på et container element. Dette vil gøre at alt I containeren bliver til grid-items. Næste del af processen tilføjer man typisk grid-template-colums. Størrelsen på disse bestemmes med FR enheden som står for fractions/brøkdele.

For at positionere ting i ens grid kræver det at hver enkel ting der skal påvirkes, skal have et unikt grid-area id. De kan hedde hvadsomhelst, dog anbefales det at navngive dem så det giver mening for dig selv, men ikke mindst andre der også skal ind og kode videre senere hen.

Visuelt eksempel er taget fra vores Onepage Projekt vi arbejde på i November 2018. Hvor faktisk havde undladt god kodeskik, men det var en del af vores fleksion over projektet efterfølgende.

**[Billede Grid area]**

Når alt derefter er indelt med et grid-area id kan de placeres i ens grid/gitter. Opstillingen der er vist her er ligedes også fra tidligere nævnte Onepage projekt

**[Billede CSS grid display]**

Måden vi brugte ovenstående grid og grid-areas var for at have en tekst samt billede ved siden af hinanden og derefter centreret billede nedenunder som vist her

**[Billede SS1 + SS2]**