**Date generale**

* **Forma generala:**

Selector{

Proprietate1 : valoare1;

Proprietate2: valoare2;

}

* **Selector** – identifica elemental HTML caruia i se aplica stilul dat
* **Proprietati** – elemente de aspect ce vrem sa le modificam
* **Valori** **ale proprietatilor** – noile valori pe care vrem sa le atribuim proprietatilor respective.

**Moduri de aplicare a modelelor de stiluri(CSS)**

1. **Inline –** direct in eticheta HTML

se foloseste style la un element intre <>

<element style=”proprietate1:valoare1; proprietate2:valoare2;”></element>

<h4 style=”color:#fdfdfd; margin-left:15px;”></h4>

font-family:’times new roman’;

Se pun ‘ ’ pentru a arata ca cuvintele date trebuie luate impreuna, caci browserul cand vede un spatiu liber, crede ca am terminat de dat valoarea.

1. **Intern -** in antetul (header-ul) fisierului

Se pune in <head> css </head> intre <style type=”text/css”> </style>

<head>

<style type=”text/css”>

h2{

color:red;

}

</style>

</head>

CSS din head se va folosi pentru mai multe elemente din pagina, adica toate vor folosi acest stil. <h2></h2> va folosi utomat ce este mai sus. De asta, nu e tehnica buna.

1. **Extern –** proprietatile si valorile pentru stiluri sunt specificate intr-un fisier extern, .css. Putem specifica diferite stiluri pentru diferite pagini sau pentru diferite elemente.

Pentru adauga o foaie de stil CSS, in <head> </head> scriem:

**<link href=”fisier.css” rel=”stylesheet” type=”text/css”>**

**rel** – specifica relatia dintre documentul apelat si cel legat.

**type** – specifica tipul documentului legat, adica ce fel de resursa este. CSS nu este suficietn!

**href** – adresa fisierului CSS

1. **Toate cele 3 de sus**

Mereu se va lua drept efect cel care e ultimul. Insa daca de exemplu, le folosim pe toate 3 metode de sus, si fiecare specifica un anumit stil diferit pentru un element, de exemplu pentru body folosim inline background-color, intern color si extern padding, vor fi folosite toate 3.

**Reguli**

* Daca elementul are mai multe cuvinte, se folosesc “” sau ‘’

font-family:”times new roman”;

* **Comentariu**: /\* comentariu \*/
* **Putem grupa elemente cu aceleasi proprietati**:

h1, h2, h3, p{

…

}

* A nu se confunda stilurile cu atributele!

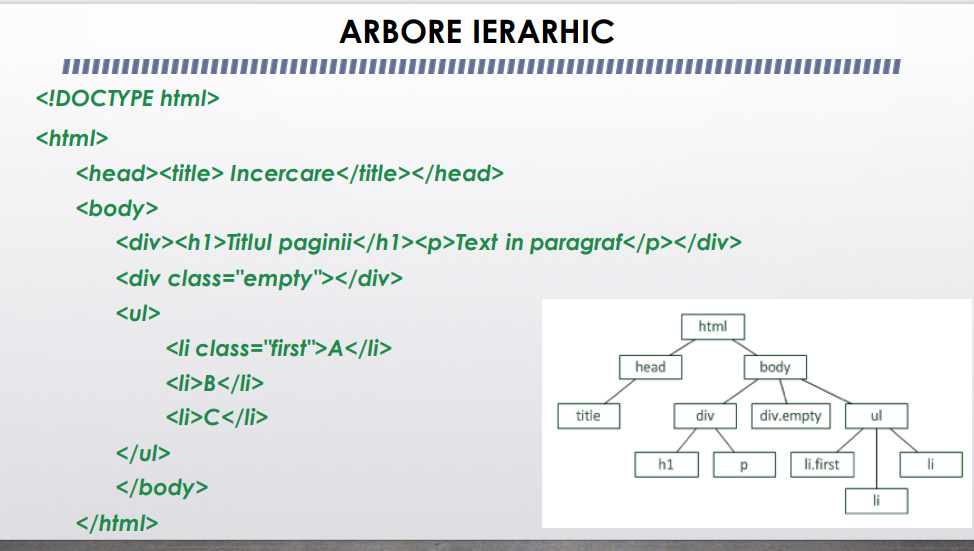
Stilurile mereu au forma <p style=”color: red;”> text </p>

Atributele CSS <p align=”center”; background; red;”> text</p>

Totusi, ele pot fi identice si ca attribute si ca stiluri.

**Mostenire**

* Este un mecanism prin care valorile proprietatilor unui element parinte sunt transmise urmasilor.
* Stilurile atribuite unui element sunt mostenite de toti urmasii daca ele nu sunt redefinite de ei.
* Unele proprietai nu pot fi mostenite, caci unele elemente pur si simplu nu le au, ca background, border, padding, margin, width, height, position etc.



<body>

<br> un spatiu liber <br> 2 spatii libere <br> 3 spatii libere </br> </br> </br>

</body>

**Stiluri**

* **font-family: font1,font2…;** - se specifica mai multe fonturi cand nu suntem siguri ca browserul il are instalat pe primul. Daca nu, se afiseaza urmatorul si tot asa.
* **font-size: marime;**

Mai poate lua si marimile:

xx-small – text foarte mic

x-small – text mai mic

small – text mic

medium – text mediu

large – text larg

larger – text mai larg

**Marimea exprimata in cifre nu este in pixeli!**

Marimea mai poate fi exprimata si in em.

1em = 16px;

Marimea poate fi exprimata si in %, unde 100% este text de marime obisnuita.

* **font-style**: normal|intalic|oblique(ca italic, dar mai putin recunoscut de browsere)
* **font-weight**: de la 100 la 900, sau:

bold – 700

bolder – 900

inhering

lighter – 100

normal – 400

* **font-variant**: permite convertirea literelor in litere mici. Nu toate fonturile suporta asta!
* small-caps
* normal
* inhering

Putem folosi stilul font: cu mai multe attribute:

**font: style variant weight size font-family;**

Trebuie folosite toate!

* **text-align**: right/end | left/auto | justify(redimensioneaza spatiile libere din text odata cu redimensionarea paginii)
* **text**-**decoration**:

none(nu apare nimic peste/sub/in text, nici macar la linkuri nu apare sublineirea)

overline – linie deasupra

line-through – lini in text

underline – linise sub text

blink – eliminat din HTML

text-decoration poate fi si combianat, de ex:

text-decoration: overline underline line-through

* **text-transform:** transforma literele din text in mari sau mici

uppercase – toate literele sunt mari

lowercase – toate sunt mici

capitalize – fiecare cuvant incepe cu litera mare

* **text-indent: nrpx** - muta textul mai la dreapta de bordure
* **text**-**shadow**: orizontal(px) vertical(px) rafinare(px) culoare

orizontal/vertical – distanta de la inceputul textului si al umbrei. Un numar pozitiv indreapta umbra spre dreapta/jos, unul negative spre stanga/sus

rafinare – cat de dispersata e umbra

* **letter-spacing**: px;

mareste sau micsoreaza spatiul dintre caractere. – il micsoreaza

* **word-spacing: px;**

mareste sau micsoreaza spatiul dintre cuvinte. – il micsoreaza

* **direction: de la spre**;

ltr(left to right) – textul incepe sa fie scris din dreapta si merge spre stanga

rtl(right to left) – textul incepe de la dreapta si merge spre stanga

* **margin** – arata distanta la care se va situa backgroundul elementului de alte elemente din exteriorul backgroundului lui. Putem avea si **margin left/top/bottom/right**
* **Padding** – arata distanta la care se va situa elementul de backgroundul lui. Orice element are un background, si deci putem seta o distanta fata de el. Putem avea si **padding left/right/top/bottom. Nu accepta valori negative!**

De exemplu, daca avem un paragraph, el are un background si textul e plasat in el. Padding ar deplasa textul in interiorul backgroundului, iar margin ar deplasa insusi backgroundul fata de alte elemente.

* **Background-color**
* **Background-image: url(“adresa”)** – browserul va repeat imaginea pe intreaga suprafata, pentru a crea un background.
* **Background-repeat: repeat-x/y/no-repeat**
* **Background-attachment:**

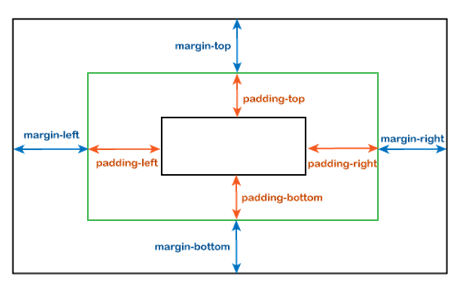
**Scroll –** se misca imagine odata ce se da mai in jos in pagina

**Fixed –** imaginea nu se misca din loc

* **Background-position: top left, top center …**

Putem folosi toate atributele de mai sus, ce tin de background, in unul, exact in ordinea de mai sus: **background: .. .. … ..;**

* **Background-size**



**Putem combina toate 4(left/righ/buttom/top) in una:**

**Margin: 10px 2px 30px 50px;**

* **Border-collapse = collapse/separate;**

Acesta determina cum vor fi liniile celulelor. Separte inseamna ca fiecare are liniile ei(contur), iar collapse toate sunt impreunate

* **Border-style = stil** creaza un border in jurul backgroundului. Poate lua multe attribute [CSS Borders (w3schools.com)](https://www.w3schools.com/css/css_border.asp)
* **Contet=”text”/url(“”) –** adauga ceva propriu zis pe ecran, un text,imagine etc.

**Stiluri pentru liste**

1. Pentru a stabili care va fi elementul de aliniere(cerc, numar etc.) folosim in CSS **list-style-type: tip;**

De ex:

Ul{

List-style-type: circle;

}

* Liste ordonate pot avea urmatoarele tipuri de ordonare;
* Decimal – numere incepand cu 1
* Lower-alpha – litere mici
* Upper-alpha – litere mari
* Lower-latin – litere mici
* Upper-latin – litere mari
* Lower-roman – romane mici
* Upper-roman – romane mari
* None - gol
* Liste neordonte:
* Circle
* Square
* Disc
* None – gol

1. **List-style-position**: alinierea unei liste.

Outside – e default. Elementele sunt inafara listei.

Inside – elementele se vor deplasa putin in dreapta, cu tot cu elementul de aliniere. Asta inseamna ca elementele se afla in lista.

1. **List-style-image: url(image.png)**

Foloseste o imagine in loc de cerc/numar etc.

Trebuie mereu de plasat inaintea acestuia **list-style-type**, deoarece in caz ca imaginea nu merge, sa se foloseasca altceva.

Se poate folosi si combinatia:

List-style-type:

List-style-position:

List-style-image:

Sau mai pe scurt: list-stile: tipul positia url(imagine.png)

**Elemente de aranjare**

* margin – distanța între bordură (border) și marginea (margin) paginii
* border – marginea (margin) și cutia care încadrează conținutul (content)
* padding – distanța între bordură (border) și conținut (content)
* content – conținutul (content) propriu-zis (text, imagini, etc

**Stiluri:**

* **border-style**: none, solid, dotted, dashed(din segmente), double, groove(putin transparent), ridge(accentuat la hotar), inset(accentuat in sus sis tanga), outset(accentuat in jos si dreapta).
* **Border-top/right/bottom/left-style:**

Poate fi scrisa doar una si combinate toate 4 in ordinea de sus.

Daca se scriu 2 valori, prima e pentru cea de sus/jos, a doua stanga/dreapta

O valoare e pentru toate

3 valori : prima sus, a doua pentru dreapta/stanga si a 3 pentru jos.

* **Border-width:** e groimea**,** nu merge fara border-style. Unitatile sunt thin,px sau thick
* **Border-color: sus,dreapta,jos,stange**
* **Border:** border-width, border-style, border-color;(forma scurta)
* **Border-right: width,style,color(**forma scurta)
* **Border-radius:** face rotunde colturile. Putem folosi si border-top-radius sau border-top-left-radius etc.

border-radius: 25px 50px 30px 5px; (1 - stânga-sus, 2 - dreapta-sus, 3 - dreapta-jos și 4 - stânga-jos)

border-radius: 25px 50px 30px; (1 - stânga-sus, 2 – dreapta-sus și stânga-jos, 3 – dreapta-jos)

border-radius: 25px 50px; (1 - stânga-sus și dreapta-jos, 2 - t dreapta-sus și stânga-jos)

border-radius: 25px; (pentru toate colțurile).

* **Outline** are tot ce are border si se comporta ca el, doar ca el e desenat in jurul lui border.
* **Margin –** stabileste distanta intre bordura si pagina.

**Margin: auto;** browserul stabileste automat dimensiunile asa ca elementul sa fie la mijloc,pe orizontala.

margin: 30px 20px 30px 50px; (sus, dreapta, stanga, jos)

margin: 30px 20px 50px; (sus, dreapta si stanga, jos)

margin: 30px 20px (sus si jos, dreapta si stanga)

margin: 30px (sus jos dreapta si stanga - o spatiere de 30 de pixeli de jur imprejurul bordurii)

Putem avea margin-top/bottom/right/left

* **Padding: -** distanta intre continut si chenar. Are tote atributele si caracteristiile lui padding.
* **Box-shadow**: adauga o umbra.

**None –** fara

**Inside –** in interior

**Deplasare-**x – intinde umbra orizotal. O valoare pozitiva e la dreapta, una negative la stanga

**Deplasare-**y: pozitiv sus, negative jos

**Degradare:** gradul de dispersie, de intindere

**Extensie:** pozitiv o intinde, negative o comprima

**Color:**

Box-shadow: horizontal, vertical, dispersie, degradare, culoare

* **Position –** setam pozitia unui element

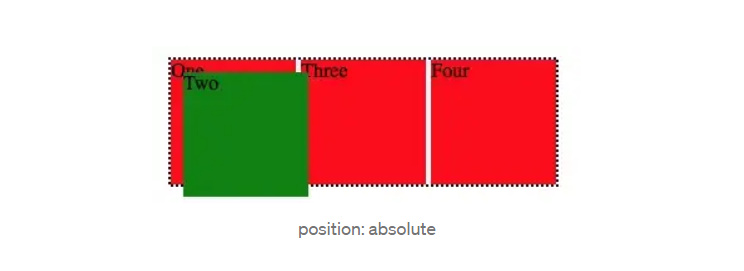
**Static –** e default

**Fixed –** are mereu pozitie fixa, indifferent de cum ne miscam in pagina

In mod nomal, elementul se asupra caruia folosim position apare peste celelalte, dar folosind width facem ca celelalte sa apara peste el, anume asta si facand width, aratand cati pixeli se pot suprapune pest eel din alte elemente.

**Relative – Permite sa modificam pozitia obiectului din cea pe care ar fi trebuit sa o aiba cum vrem, si poate atinge alte obiecte fara sa le afecteze.Chiar de e mutat, locul unde el a stat initial se pastreaza si mutarea lui incepe din pozitia initiala.** se combina cu elementele **Left:** right: top: bottom: si e ca static, doar ca apare in urma altor elemente si nu le poate influenta pozitia. **Ne permite sa mutam cum vrem obiectul cu selectorii Left, Right, Bottom, Top, chiar daca nimeresc peste alte structuri din pagina, si se muta in raport cu pagina.**

**Absolute –**se pozitioneaza anume in raport cu blocul din care provine(<div> de ex), nu cu pagina.Spatiul destinat lui in bloc dispare si incepe sa fie pozitionat din punctul 0 0 al blocului parinte.. Tot poate avea si left: right: top: bottom:



**Float si Clear**

* **Float**: right/left;

Acest identificator face ca un element sa permita alinierea sau plutirea altor elemente in partea stanga/dreapta a acestuia. De exemplu, daca punem o imagine si apoi un text dupa ea, imaginea va crea un rand mare, si daca punem text dupa ea, textul va fi langa ea, dar jos. Cu float, textul va incepe de sus de langa imagine chiar. **Float e doar pentru alinierea orizontala, asa cum textul si asa poate sta deasupra sau dedesuptul imaginii.**





Float-left: imaginea va fi in stanga textului, si elementele plutesc in dreapta ei.

Float-right: imaginea va fi in dreapta textului, si elementele plutesc in stanga ei.



Cu float pot fi folosite si alte elemente inafara de imagini.

Float poate fi aplicat si listelor.

* **Clear: left/right/both** – acest atribut e folosit pentru a indica ca un element nu poate pluti langa un element flotant. De exemplu, daca imaginea e flotanta si punem un paragraf dupa ea, paragraful va fi amplasat langa imagine, dar daca nu vrem ca acest paragraf sa fie amplasat langa, ci asa cum se face in mod normal, dupa ea, folosim clear, care nu permite plutirea.

Left – nu se permite aranjarea langa o imagine ce este in partea stanga,adica are float left

Right – nu se permite aranjarea langa o imagine ce este in partea dreapta,adica are float right

**Overflow**

Permite crearea unei bare de derulare atunci când conținutul elementului este prea mare pentru zona specificată



* Visible – valoarea default, se afiseaza tot elementul chiar de nu incape
* Hidden – elementul care iese din border nu este afisat.
* Scroll – se adauga bare de defilare a elementul, ca e prea mare sau nu
* Auto – se adauga bare de defilare doar daca elementul e prea mare
* Inherit – mosteneste de la parinte

**Z-index**

Permite crearea unei ordini de a insera obiectele unul peste altul. Daca un obiect are z-index -1 si altul 1, cel ce are 1 va fi afisat peste cel ce are -1. Merge doar la obiectele ce au position alternative sau absolute. Elementele cu z-index mai mare sunt situate peste cele cu index mai mic. Valoarea default e 0

In mod default, obiectele sunt afisate in ordinea lor din cod. Daca mai intai e pusa o imagine si dupa ea e scris codul unui paragraf, e logic ca paragraful va fi afisat peste imagine.

**Rect**

Permite afisarea unei bucati din imagine.

React: rect(x1,x2,y2,y1)

**Display**

**Display** controleaza cum vor fi afisate elementele intr-un bloc.

* Inline – toate elementele vor fi afisate unul langa altul. Daca un div are display: inline; atunci, de exemplu, paragrafele nu vor mai avea cate o linie inainte si dupa ele.
* Block - toate elementele vor fi blocuri ca si <p>, adica cu o linie pana la

**Machetarea WEB responsiva**

Este un principiu bazat pe crearea unei pagini in care aspectul e bazat pe CSS pe baza marimii ferestrei, ceea ce permite un design comod pentru orice dimensiune de ecran.

**Instrumente CSS pentru proiectare responsive**

* **Interogari media** – permite determinarea in CSS a conditiilor pentru aplicarea anumitor stiluri pentru diferite ecrane.

Atunci când un site fără un design responsive este vizualizat pe un dispozitiv mobil, browserul îl plasează complet în lățimea ecranului și reduce dimensiunea paginii.

Dacă designul site-ului este responsive, trebuie să informați suplimentar browserul despre aceasta adăugând linia:

**<meta name="viewport" content="width=device-width, initial-scale=1.0" />**

- meta tag-ul spune browserului că lățimea width trebuie să fie egală cu lățimea ecranului browserului, care la rândul său este egală cu lățimea ecranului dispozitivului de pe care este vizualizată pagina web. Initial-scale controlează nivelul de zoom la încărcarea unei pagini.

**Puncte de control** – puncte in care pagina ia alte stiluri. De ex, daca lungimea < de 668px, se vor folosi anumite stilui, daca > alte siluri si etc. 668 e punct de control.

Utilizam, deobice, 3 puncte de control:

* Smartphone
* Tableta
* monitor

Sintaxa CSS:

**@media not|only mediatype and (mediafeature and|or|not mediafeature) {CSS-Code;}**

* **not** inversează semnificația unei interogări media.
* **only**: împiedică browserele mai vechi care nu acceptă interogările media cu caracteristici media să aplice stilurile specificate. Nu are niciun efect asupra browserelor moderne.
* **and**: combină o caracteristică media cu un tip media sau alte caracteristici media

Pot fi aplicate foi de stil diferite pentru diferite medii, cum ar fi:

<link rel="stylesheet" **media="screen and (min-width: 1200px)"** href=“largescreen.css">

< link rel="stylesheet" **media="screen and (max-width: 600px)"** href="smallscreen.css">

Tipuri de media:

**all** - Implicit, folositpentru toate dispozitivele de tip media.

**print** - pentru imprimante.

**screen** - pentru monitoare, tablete, telefoane, etc.

**speech** - pentru cititoarele de ecran care „citesc” pagina cu voce tare.

**Valori media**:

**height** - înălțimea ferestrei

**max-height** - înălțimea maximă a zonei de afișare, cum ar fi o fereastră de browser **min-height** - înălțimea minimă a zonei de afișare, cum ar fi o fereastră de browser **width** - lățimea ferestrei

**max-width** - lățimea maximă a zonei de afișare, cum ar fi o fereastră de browser **min-width** - lățimea minimă a zonei de afișare, cum ar fi o fereastră de browser **orientation** - orientarea ferestrei (modul landscape sau portrait

@media (min-width: 576px) {

.m576 {

width: 100%; }

h1 {

font-size: 24px;

}

p {

color:blue;

}

... }

* **Griduri Fluide** - marcaj care se poate extinde și contracta (împreună cu conținutul), adaptându-se la ecranul unui dispozitiv specific

box-sizing: border-box;

Rezolva problema care apare la folosirea lui padding, ce face ca dimeniunea blocului sa creasca cand obiectul se misca.

**Word Spacing**

Word-spacing: Xpx;

Seteaza dimensiunea intre cuvinte.

**White-space**

* **Normal –** valoare default
* **Nowrap –** textul va fi mereu scris intr-o singura linie, si daca nu incape in border, iese din el si daca nu incape in pagina, o intinde mai tare
* **Pre –** e ca si <pre>, adica adauga linii noi cand in HTML textul are linii noi. Iese din border daca nu incape.
* **Pre-line –** daca textul nu incape in border, trece la linia urmatoare. Transforma mai multe spatii libere in unul
* **Pre-warp –** exact ca pre-line, dar afiseaza toate spatiile libere.
* **Initial -**