CSS - *Cascading Style Sheets*

http://www.marplo.net/curs_css/

Cuprins Lectii :

Curs C.S.S. (Cascading Style Sheets)

Introducere

Crearea paginilor HTML este un lucru relativ simplu, invatarea etichetelor HTML si crearea unor imagini ducand la realizarea de pagini web de o complexitate medie. Odata cu dezvoltarea internetului, site-urile au devenit din ce in ce mai complexe, cu un numar mai mare de pagini, cerintele privind grafica si elementele din pagina au devenit mai pretentioase si astfel proiectarea paginilor web a devenit o sarcina ceva mai dificila.

O problema importanta cand avem un site cu multe pagini este atunci cand dorim sa facem anumite schimbari in elementele pagini: fondul, grafica sau fontul textelor din pagini.

Prin utilizarea CSS (*Cascading Style Sheets*), in traducere "*foi de stil in cascada*", acest lucru nu mai este o problema, realizandu-se relativ usor, prin schimbarea sau adaugarea unor elemente in codul CSS, ne fiind nevoi sa lucram la fiecare pagina sau la fiecare element din pagina.

CSS se ocupa in general cu aspectul si controlul grafic al elementelor din pagina, cum ar fi: textul, imaginile, fondul, culorile si asezarea acestora in cadrul ferestrei paginii.

CSS foloseste stiluri, acestea inglobeaza, sub un anumit nume, atribute de formatare care se aplica asupra unui element individual din pagina, asupra unui grup de elemente sau la nivelul intregului document.

CSS functioneaza cu HTML, insa nu este HTML. El extinde functionalitatile HTML, permitand redefinirea etichetelor HTML existente.

Prin utilizarea CSS aspectul documentului pe ansamblu, sau a unui element individual din interiorul sau, poate fi controlat mult mai usor. Stilurile pot fi aplicate asupra unui element, a unui document sau chiar asupra unui intreg site web. Un dezavantaj ar fi ca unele navigatoare nu sunt compatibile CSS, astfel ca documentele HTML sunt afisate ca si cum CSS n-ar exista, dar cele mai cunoscute si utilizate browsere, cum ar fi: Mozilla Firefox, Internet Explorer, Opera, si altele, sunt compatibile CSS.

Acest curs prezinta elementele de baza CSS si modul de lucru cu "foile de stil", va invata cum sa adaugati si sa folositi CSS in paginile dv. web. Tot ce aveti nevoie este un editor de texte simplu, cum ar fi Notepad, si un navigator, de exemplu Mozilla Firefox, care este gratuity.

Pentru a invata cat mai bine elementele prezentate in aceste lectii, exersati personal fiecare exemplu si proprietate explicata.

Retineti ca acest curs reprezinta un punct de plecare, ne fiind prezentate aici toate proprietatile si elementele CSS. Pentru o cunoastere mai amanuntita a acestora, vizitati site-ul www.w3.org, la sectiunile despre CSS.

Lectia 1. Scrierea codului CSS

Codurile CSS pot fi scrise in *interiorul paginii* sau intr-un fisier extern cu *extensia".css"*. Codul CSS, ca forma generala, este alcatuit din: obiectul care va fi formatat. proprietatile acestuia si valoarea (sau atributele) fiecarei proprietati.Cand este adaugat in documentul HTML, trebuie scris in cadrul unui tag **<style>** in sectiunea HEAD a documentului HTML, dupa cum se vede in formula generala de mai jos;

```
<html> <head> <title>Titlul</title>
<style type="text/css">
   obiect_css {
                    proprietate:valoare;
                    alta_proprietate:valoare;
               }
</style> </head><body> ... Continut ... </body> </html>
```

- Observati sintaxa codului CSS. Perechile **proprietate:valoare** se scriu intre acoladele obiectului CSS pe care-l definesc, intre "proprietate" si "valoare" trebuie sa fie un caracter doua-puncte (**:**) iar la sfarsitul perechii se adauga un caracter punct-si-virgula (**;**)

- Cand este adaugat intr-un fisier extern ".css", codul CSS se scrie la fel, dar nu se mai adauga tag-ul <style>.

1.1. Obiectele (regulile) CSS.

Exista trei tipuri principale de obiecte CSS: selector, clasa si identificator.

1. Selectorul HTML se foloseste pentru a redefini modul de afisare a continutului etichetei HTML. Un selector HTML reprezinta partea etichetei HTML care indica navigatorului tipul de eticheta. Iata un exemplu:

h1 { font-family:"Arial"; font-size:15px; }

- Aici selectorul este "h1". Definirea unui selector HTML folosind CSS are ca rezultat redefinirea etichetei HTML. Selectorul si eticheta desi par identice, totusi nu sunt.

2. Clasa este un obiect care poate fi aplicat oricarei etichete HTML. O clasa trebuie creata in interiorul etichetei HTML inainte de a fi definita intr-un cod CSS. Crearea clasei in interiorul etichetei se face simplu, prin specificarea cuvantului **class** si numele clasei, ca in exemplu de mai jos:

<h1 class="nume_clasa"> Text </h1> unde "nume_clasa", poate fi orice nume dorim.

- Apoi in interiorul codului CSS, clasa trebuie definita prin adaugarea unui caracter punct (.) inaintea numelui clasei, ca in exemplu urmator: **.nume_clasa { font-family:"Arial"; font-size:15px; }**

Numele aceleasi clase poate fi atribuit mai multor etichete HTML din aceeasi pagina, si toate vor prelua acelasi stil css.

3. Identificatorul (ID) este asemanator cu *clasa*. Pot fi aplicate oricarei etichete HTML, dar spre deosebire de clase, numele unui identificator trebuie atribuit numai unei singure etichete HTML dintr-o pagina, pentru alta eticheta se adauga un ID cu nume diferit. Ca si clasa, identificatoru trebuie intai creat in interiorul etichetei HTML.

Modul de creare este simplu, prin specificarea cuvantului **id** si numele clasei, ca in exemplu de mai jos:

<h1 id="nume_id"> Text </h1> "nume_id", poate fi orice nume dorim.

In interiorul codului CSS, identificatorul este definit prin adaugarea unui caracter diez (#) inaintea numelui, ca in exemplu urmator: **#nume_id { font-family:"Arial"; font-size:15px; }**

In urmatoarea lectie va fi explicat mai amanuntit modul de definire si rolul acestor obiecte CSS .

1.2. Componentele unui obiect CSS

Obiectele CSS, indiferent de tipul lor, au in componenta urmatoarele elemente:

- **Selectorii** - identifica un obiect; pot fi selectori de etichete HTML, clase sau identificatori.
- **Proprietatile** - identifica o proprietate a obiectului; se refera in special la aspect.
- **Valorile** - sunt atributele unei proprietati; pot fi cuvinte cheie, valori numerice sau procentuale, tipul valorii depinzand de proprietate.

Sintaxa generala a unei reguli CSS este urmatoarea: **selector {proprietate: valoare; }**

Proprietatile si valorile constituie definitia regulii (obiectului) CSS. Acestea se regasesc sub forma de perechi,despartite de caracterul doua puncte ":", fiecare pereche se termina cu un caracter punct si virgula ";".

Etichetele HTML **nu** sunt sensibile la diferenta intre majuscule si litere mici. Dar odata cu aparitia limbajului XHTML, acesta face distinctie intre majuscule si minuscule, astfel ca toate etichetele si toti selectorii trebuie scrisi cu litere mici.

Lectia 2 - Crearea Foilor de Stil

2.1. Etichete HTML si stiluri CSS

CSS ofera posibilitatea de a schimba aspectul fiecarei etichete in parte, prin stabilirea unui anume stil in interiorul ei, cu atributul "**style**". Acest lucru este util mai ales pentru a anula alte stiluri ale elementului respectiv sau de a da elemente grafice de stil doar etichetei respective.

Sintaxa pentru definirea stilurilor in interiorul unei etichete HTML este urmatoarea:

<eticheta style="proprietate:valoare;"> text ... </eticheta>

CSS permite si definirea unor reguli de stil generale intr-o pagina web. Acest set de reguli trebuie scris in sectiunea de antet (head) a documentului, in cadrul tag-ului <style>.

Sintaxa pentru definirea CSS intr-un document HTML, in interiorul etichetei <head> </head>, este urmatoarea:

```
<style type="text/css">
selector_1 {proprietate1:valoare1; proprietate2:valoare2; ... }
...
selector_n {proprietate1:valoare1; proprietate2:valoare2; ... }
</style>
```

- Definirea tuturor stilurilor intr-o locatie comuna usureaza modificarea mai rapida a unei pagini.
Iata un exemplu practic de cod css:

```
<style type="text/css">
  h1 { font-family:'Arial'; font-size:15px; font-weight:bold; color:#1111ff; }
  p {font-family:'Arial'; font-size:12px; color:blue; }
</style>
```

Foile de stil pot fi utilizate nu doar la nivel de pagina web, ci si la nivel de intreg site. Astfel, trebuie creata o foaie de stil externa intr-un fisier separat, de preferat cu extensia ".css", care poate fi inclus in pagina HTML prin doua procedee: legatura sau import.

Crearea unei foaie de stil externe se face scriind codul CSS intr-un fisier text cu ajutorul unui editor de texte simplu (de exemplu Notepad in Windows), sau specializat in acest sens (precum Notepad++). In fisierul extern creat se adauga reguli CSS, fara insa ca aceste reguli sa fie incadrate in etichete STYLE. Dupa ce a fost creata foaia de stil externa, aceasta poate fi folosita de un document HTML utilizand urmatoarea sintaxa, in interiorul tag-ului <head>

<head> <link rel="stylesheet" href="nume_fisier.css" type="text/css"> </head>

- Eticheta LINK apare in antetul documentului (sectiunea head), iar atributele folosite transmit navigatorului tipul de legatura ("rel" – legatura cu o foaie de stil, "type" - tipul codului din fisier) si locatia inspre fisierul ce contine codul CSS ("href" – calea si numele complet al fisierului, inclusiv extensia).

O alta modalitate de utilizare a foilor de stil externe intr-un document HTML o reprezinta importul acestora folosind comanda **@import**. Sintaxa pentru importul unei foi de stil externe este urmatoarea:

<style type="text/css"> @import url(nume_fisier.css); </style>

Pentru a importa un fisier CSS extern se foloseste in cadrul sectiunii HEAD a documentului HTML eticheta STYLE. In cadrul acestei etichete este adaugata instructiunea "@import" de mai sus, unde "nume_fisier.css" reprezinta calea si numele fisierului ce contine regulile CSS definite.

Alaturi de instructiunea "@import", in cadrul etichetei STYLE, pot fi adaugate definiti si selectori suplimentari. Legatura la un fisier CSS extern sau importul acestuia intr-un document HTML are acelasi efect ca si cum stilurile incluse ar fi fost definire direct in eticheta STYLE din sectiunea HEAD a documentului HTML.

2.2. Definirea selectorilor

- Selectorii HTML pot fi definiti prin adaugarea unui numar de definitii compatibilie cu eticheta HTML la care se refera, avand urmatoarea forma generala: **selector_HTML { proprietate1:valoare1; proprietate2:valoare2; ... }**

Dupa redefinirea etichetei HTML, stilurile etichetelor respective din documentul HTML vor fi modificate automat. Prin redefinirea unei etichete, proprietatile prestabilite existente nu sunt anulate, ci se adauga altele noi.

Utilizarea selectorilor de tip clasa ofera posibilitatea configurarii unor stiluri care se pot aplica doar acelor elemente care sunt etichetate cu clasa respectiva. Sintaxa generala de definire a unei clase CSS este:

.nume_clasa { proprietate1:valoare1; proprietate2:valoare2; ... }

Exista cazul in care o clasa este asociata unui selector HTML, ceea ce inseamna ca acea clasa poate fi folosita doar cu eticheta HTML respectiva. Pentru a defini o clasa care sa afecteze in mod direct un anume selector HTML, se foloseste urmatoarea sintaxa: **selector_HTML .nume_clasa { proprietate1:valoare1; proprietate2:valoare2; ... }**

Selectoarele de clasa sunt acceptate de toate navigatoarele. Numele unei clase e recomandat sa fie diferit de cuvantele rezervate JavaScript.

Asemanator cu selectorii de clasa se definesc si identificatorii (id-ul). Acestia sunt folositi pentru crearea de stiluri care pot fi atribuite unei singure etichete HTML dintr-o pagina. Sintaxa generala de definire a unui identificator este:

#identificator { proprietate1:valoare1; proprietate2:valoare2; ... }

Identificatorii permit definirea unui element sub forma unui obiect, fiind folositi doar o singura data in cadrul unei pagini web pentru identificarea tag-ului respectiv, astfel poate fi manipulat si cu ajutorul functiilor JavaScript.
Numele unui identificator e recomandat sa fie diferit de cuvantele rezervate JavaScript.

3. Crearea etichetelor HTML personalizate

Majoritatea etichetelor HTML au unele proprietati prestabilite. Acestea fie raman asa cum sunt, fie pot fi redefinite. Exista insa cazuri in care se doreste crearea unor etichete personalizate, pornind de la zero. In acest caz se folosesc etichetele **** si ** **.
Eticheta **** nu are proprietati mostenite. Ea reprezinta doar o locatie vida care creaza o eticheta in linie.
Pentru a configura o eticheta in linie trebuie definita o clasa sau identificator care sa poata fi aplicat apoi unei etichete **.**
Iata un exemplu cu, forma generala, in care selectori sunt precedati de eticheta :

```
...
<span class="nume_clasa"> ... </span>
```

...
** ... **
...
** ... **
...

Acum iata cum pot fi definiti acestia in interiorul unei foi de stil:

.nume_clasa { proprietate1:valoare1; proprietate2:valoare2; ... }
#span1 { proprietate1:valoare1; proprietate2:valoare2; ... }
span .clasa_span { proprietate1:valoare1; proprietate2:valoare2; ... }

In momentul in care se doreste configurarea unui bloc separat de restul continutului unui document HTML, solutia este eticheta ** **. Aceasta creaza o zona proprie in pagina, cu linie noua atat deasupra sa cat si dedesubtul sau.

Pentru crearea etichetelor DIV se procedeaza la fel ca si in cazul etichetelor in linie SPAN, prin definirea mai intai a unei etichete de tip clasa sau identificator, urmata apoi de aplicarea ei asupra unei etichete .
Iata forma generala de aplicare a unei etichete intr-o pagina HTML :

...
** ... **
...
** ... **
...

Definirea acestor etichete intr-un cod CSS se face astfel:

div { proprietate1:valoare1; proprietate2:valoare2; ... }
.nume_clasa { proprietate1:valoare1; proprietate2:valoare2; ... }
#div1 { proprietate1:valoare1; proprietate2:valoare2; ... }

Regulile CSS pentru definirea etichetelor sau , pot fi plasate la fel ca si celelalte tipuri de selectoare, in sectiunea "head" a documentului in interiorul etichetei "style", sau intr-un fisier extern carev ulterior este legat sau importat in documentul HTML. *Mai multe lucruri despre DIV si SPAN puteti invata de aici -> DIV si SPAN*.

4. Definirea de reguli similare

Daca mai multi selectori folosesc aceleasi definitii css, acestia pot avea aceeasi lista de elemente, fiind scrisi separat prin virgule. Sintaxa generala pentru definirea unei liste cu mai multi selectori este urmatoarea:

selector1, selector2, ... { proprietate1:valoare1; proprietate2:valoare2; ... }

Impreuna cu selectorii pot fi de asemenea definiti identificatorii si clasele. Dezavantajul ar fi ca in momentul in care este modificata o valoare a unei proprietati incluse in definitie, valoarea respectiva se va modifica in toate etichetele reprezentate de acesti selectori.

5. Definirea etichetelor in context (imbricate)

Cand o eticheta este inconjurata de alte etichete (aflandu-se astfel una in interiorul alteia), spunem ca aceste etichete sunt imbricate. Eticheta exterioara se numeste, in acest caz, eticheta parinte, iar cea inferioara se numeste copil. Se pot crea reguli care revin numai etichetelor copil, sintaxa generala a unei astfel de reguli fiind:

selector1 selector2 ... { proprietate1:valoare1; proprietate2:valoare2; ... }

- Unde "selector1" este selectorul parinte iar "selector2" este selectorul copil.

Lista de selectori imbricati poate fi mai mare de doi, ultimul selector din lista este cel care primeste toate stilurile incluse in regula si in plus le mosteneste si pe cele ale parintilor.Ca si selectorii singuri, selectorii imbricati pot include in lista clase, identificatori sau selectori HTML.Toate etichetele HTML, cu exceptia etichetei BODY, au cel putin o eticheta parinte care le inconjoara.

De cele mai multe ori stilurile etichetelor din interior preiau stilurile etichetelor parinte (exista insa cazuri in care proprietatile nu sunt mostenite de etichetele imbricate). Acest mecanism se numeste mostenirea stilurilor.

Prin redefinirea unui selector, eticheta nu-si pierde proprietatile prestabilite, doar in cazul in care acestea sunt modificate. Astfel proprietatile mostenite pot fi anulate prin redefinirea acestora in lista de definitii a etichetei imbricate.

6. Cresterea prioritatii unei definitii

Valoarea **!important** adaugata unei definitii atribuie maximum de prioritate in determinarea ordinii unei executii.Valoarea **!important** trebuie plasata inaintea caracterului ';', ca in exempul urmator:

selector { proprietate1:valoare1; proprietate2:valoare2 !important; ... }

7. Determinarea ordinii executiei

Deoarece exista mai multe moduri de a aplica stilurile, pot apare situatii in care unei etichete sa-i fie aplicate mai multe stiluri. Foile de stil din doua sau mai multe surse, folosite simultan, pot cauza contradictii. De aceea exista cateva reguli care determina ordinea executiei (numita si cascada), acestea sunt:

1. Regulile CSS scrise in interiorul paginii HTML, in cadrul etichetei "style" din sectiunea "head" au o prioritate mai mare decat cele scrise intr-un fisier extern, iar regulile scrise in interiorul etichetelor au o prioritate mai mare decat cele din cadrul etichetei "style" din sectiunea "head".

2. Existenta atributului **!important** – confera prioritate maxima la afisare definitiei in care este utilizat.

3. **Sursa regulilor** – exista numeroase navigatoare care permit utilizatorului sa-si defineasca propriile foi de stil. Totusi, foile de stil ale autorului le anuleaza pe cele ale vizitatorului daca acestea din urma nu au valoarea "!important".

4. **Specificitate** – cu cat o regula dispune de mai multi selectori HTML, de clasa si de identificator, cu atat prioritatea ei creste. ID-urile au valoarea 100, clasele au valoarea 10, iar selectorii HTML au valoarea 1.

5. **Momentul aparitiei** – cu cat un stil apare mai tarziu, cu atat importanta lui este mai mare. Astfel, definitiile unei etichete copil au prioritatea mai mare si anuleaza toate stilurile precedente cu care intra in conflict.

8. Adaugarea comentariilor la CSS

Este bine ca uneori sa fie adaugate comentarii in codul CSS, mai ales in cazul fisierelor externe. Comentariile ajuta la intelegerea codului, facandu-se astfel cunoscut, pentru mai tarziu, rolu anumitor elemente din codul CSS.
Comentariile nu au nici un efect si pot fi plasate in jurul regulilor, fiind utile si in cazul navigatoarelor care nu suporta CSS.*Pentru a adauga comentarii in regulile de stil avem la dispozitie doua modalitati:*

// pe o linie sau / comentariu */ pe mai multe linii*

Lectia 3 - Configurarea fonturilor

Alegerea potrivita a fonturilor si folosirea acestora in cadrul paginilor web este importanta, poate atrage critici si comentarii din partea vizitatorilor, mai ales daca formatarea clasica prin scris normal, aldin sau cursiv nu este folosita corespunzator. CSS include facilitati de control asupra aspectului fonturilor, prin posibilitatea de a stabili familia de fonturi, atributele ingrosat si inclinat, dimensiunea literelor precum si spatiul dintre linii.
Exista cinci familii de fonturi de baza:

- **serif** – au un ornament (serif) plasat la terminatia literei, care ii ofera o distinctie speciala. Sunt folosite pentru tiparire, chiar daca textele sunt mai mari sau mai mici. Nu sunt indicate pentru afisarea textelor pe ecran.
- **sans serif** – sunt fonturi care nu folosesc serife, fiind indicate pentru continut text general.
- **monospace** – fonturile monospatiate pot avea serife, ele se deosebesc prin faptul ca fiecare litera ocupa aceeasi cantitate de spatiu. Sunt cele mai indicate pentru textele care trebuie citite cu exactitate, ca de exemplu liniile de program.
- **cursive** – imita scrisul de mana, intr-o maniera stilizata. Sunt indicate in scopuri decorative, nefiind recomandate pentru scrierea unor texte mai lungi.
- **fantasy** – nu se incadreaza in nici una dintre categoriile de mai sus, fiind fonturi care au un caracter predominant ornamental (reprezentand ilustratii sau pictograme).

1. Stabilirea fontului

Fontul folosit pentru afisarea unui text poate fi stabilit prin proprietatea "font".
Pentru definirea fontului in cadrul unei reguli, trebuie specificata, dupa selectorul din cadrul foii de stil, proprietatea **font-family** urmata de numele fontului sau a fonturilor (despartite prin virgula). Este bine ca numele fonturilor sa fie incadrate intre ghilimele simple sau duble, mai ales daca numele acestora contine spatii.
Forma generala fiind urmatoarea: **selector { font-family:"nume_font1", "nume_font2", ..., nume_generic }**

Dupa ultima virgula se pot folosi urmatoarele nume generice de fonturi: "serif", "sans-serif", "cursive", "monospace" sau "fantasy". Includerea unei asemenea valori este optionala, insa recomandata.
Iata un exemplu practic: **h1 { font-family:"Arial", "Helvetica", sans-serif }**

Cand este specificata o lista de fonturi, navigatorul incearca sa foloseasca primul font din lista, daca nu este gasit parcurge lista pana in momentul in care intalneste un font instalat. Daca nu exista fonturi echivalente, textul va fi afisat cu fontul prestabilit de browser. Daca este specificat un nume generic, textul va fi afisat cu un font apartinand aceluiasi stil, in cazul in care fonturile specificate in lista nu sunt disponibile.

Exista posibilitatea folosirii unei palete largi de fonturi in crearea paginilor web, nu doar a celor din lista limitata a fonturilor compatibile cu navigatoarele. Solutia consta in inglobarea fontului in cadrul paginii si trimiterea lui in mod automat in calculatorul vizitatorului, adica, descarcarea si folosirea lui automata.

Pentru a ingloba un font intr-o pagina si a-l descarca, se foloseste intructiunea **@font-face**. Aceasta trebuie sa includa proprietatea **font-family**, urmata de numele fontului si apoi trebuie indicata locatia pe server de unde poate fi descarcat fontul respectiv. Codul CSS este urmatorul:

@font-face { font-family:nume_font src: url(locatie_font); }

Nu poate fi ales orice font pentru a fi descarcat, acesta trebuie sa respecte un anume format. Totusi, problema este ca nu exista un format valabil pentru toate navigatoarele, de exemplu Internet Explorer foloseste fonturi in format "eof" (obtinute cu ajutorul unui program numit WEFT), iar Mozilla foloseste un format "otf".

2. Stabilirea dimensiunii fontului

Cu ajutorul CSS se poate stabili dimensiunea fontului folosind valori absolute (exprimate in diferite unitati de masura: pixeli, centimetri), procentuale, sau chiar relative.

Pentru a defini dimensiunea fontului in cadrul unei reguli trebuie specificata proprietatea **font-size** urmata de o valoare a dimensiunii care poate lua una din urmatoarele tipuri de valori:

- **unitate de masura** - exprimata in pixeli, puncte, inci sau centimetri
- **expresie absoluta** - xx-small, x-small, small, medium, large, x-large si xx-large
- expresiile **smaller** sau **larger**, ca raport cu elementul parinte
- **procent** - un numar exprimat in procente (cu %), care indica marimea textului in raport cu dimensiunea elementului parinte

Iata forma generala a codului CSS pentru stabilirea dimensiunii fontului: **selector { font-size:valoare }**

Este bine ca pentru foile de stil care formateaza iesirea la imprimanta sa se foloseasca ca unitati de masura punctele, cm sau mm. Pixelii reprezinta, probabil, cea mai des utilizata unitate de masura pentru stabilirea dimensiunii fontului. Pixelii nu reprezinta o masura foarte sigura, dar, de obicei, pentru afisarea pe ecran, constituie o masura mai sigura decat dimensiunea in puncte. Iata un exemplu practic exprimat in pixeli: **h1 { font-size:15px; }**

3. Text inclinat

Atributul "font-style" permite scrierea textelor inclinate in doua moduri: "cursiv" si "oblic". Notiunile pot fi confundate, insa cursivul se refera la versiunea unui font a carui caractere au o inclinare spre dreapta, iar oblicul este un font inclinat fortat spre dreapta.

Pentru a crea caractere cursive in cadrul unei reguli trebuie precizata proprietatea **font-style**, astfel:

selector { font-style:valoare }

Unde "valoare" poate fi unul din urmatoarele cuvinte: **normal**, **italic** sau **oblique**

Iata un exemplu: **h1 { font-style:italic; }**

Valoarea **normal** se poate folosi pentru a sterge formatarea cu caractere inclinate atunci cand aceasta este mostenita.

4. Grosimea fontului

Ingrosarea este una dintre metodele utilizate des pentru scoaterea in evidenta a unui text. CSS confera mai multa flexibilitate in utilizarea textului ingrosat decat elementul HTML <b>, prin posibilitatea introducerii unei valori proprietatii **font-weight**, dupa cum urmeaza:

- **bold** – scrie fontul folosind caractere ingrosate
- **bolder** - mareste grosimea fontului relativ la grosimea curenta
- **lighter** – micsoreaza grosimea fontului fata de grosimea curenta
- - o valoare cuprinsa intre **100** si **900** (multiplu de 100) – reprezinta valoarea ingrosarii, bazata pe versiunile alternative disponibile ale fontului
- **normal** – elimina formatarea cu caractere aldine (ingrosate)

In cazul folosiri valorii cuprinse intre 100 si 900, 400 reprezinta valoarea corespunzatoare textului normal, iar 700 valoarea ingrosarii pentru stilul aldin. Majoritatea fonturilor nu dispun de noua ponderi, astfel ca daca este specificata o valoare care nu este disponibila, va fi folosita o alta pondere.

Forma generala pentru stabilirea grosimii fontului este urmatoarea: **selector { font-weight:valoare }**

Iata un exemplu practic: **h1 { font- weight:bold; }**

5. Majuscule mici

Majusculele mici sunt folosite de obicei in cadrul titlurilor, ele fiind obtinute prin convertirea literelor mici in majuscule cu o dimensiune mai mica decat majusculelor normale. Astfel se obtin cuvinte in care toate literele sunt scrise cu majuscule insa prima litera este mai inalta decat celelalte.

Stabilirea unei reguli pentru majuscule mici se realizeaza prin utilizarea proprietatii **font-variant** cu valoarea **small-caps**. Prin folosirea valorii **normal**, vor fi anulate celelalte valori mostenite ale proprietatii **font-variant**.

Forma generala este urmatoarea: **selector { font-variant:valoare }**

Iata cum formatam un text dintr-o eticheta "h1" cu valoarea "small-caps": **h1 { font-variant:small-caps; }**

6. Valori multiple pentru font

Deseori este util sa fie precizate toate elementele fontului intr-o singura definitie.
Pentru a se realiza acest lucru este folosita proprietatea "font" urmata de valorile pentru celelalte proprietati, separate prin spatiu. *Sintaxa generala este urmatoarea:*

selector { font:val_font-family val_font-style val_font-variant val_font-weight val_font-size val_height }

Daca o anumita valoare din lista este omisa, navigatorul va folosi valoarea prestabilita. Iata si un exemplu practic de folosire a mai multor valori in aceeasi definitie: **h1 { font:"Arial",sans-serif italic small-caps bold 14pt; }**

Lectia 4 - Configurarea textului

Textele alcatuiesc o buna parte din paginile web. Metodele de afisare a textelor prin controlul nu numai a fontului, dimensiunii si culorilor, ci si prin alte elemente, pot imbunatati aspectul pagini si pot atrage atentia asupra anumitor elemente din text.

1. Spatiul intre litere

Spatierea se refera la cantitatea de spatiu dintre literele unui cuvant. Se poate aduaga sau reduce spatiul dintre litere folosind proprietatea **letter-spacing**, urmata de o valoare exprimata intr-o anumita unitate de masura, ce poate lua si valori negative. Sintaxa generala este urmatoarea: **selector { letter-spacing:valoare }**

Daca valoarea proprietatii este **normal** sau **0** atunci spatierea va fi fixata la valorile implicite (fara modificari suplimentare). Daca se utilizeaza o valoare exprimata in **em** atunci dimensiunea rezultata va fi calculata in raport cu cea a elementului parinte. Exemplu: **h1 { letter-spacing:1,5em; } (Puteti folosi si pixeli (px))**

2. Spatiul dintre cuvinte

CSS da posibilitatea modificarii spatiului intre cuvinte atat prin marire, cat si prin micsorare.
Spatiul dintre cuvinte poate fi modificat folosind proprietatea **word-spacing**, urmata de o valoare exprimata intr-o anumita unitate, fiind de obicei stabilita in pixeli. Pentru anularea spatierii intre cuvinte, valoarea proprietatii trebuie sa fie: **normal**. Sintaxa generala este urmatoarea: **selector { word-spacing:valoare }**

O valoare pozitiva semnifica un plus de spatiu adaugat la valoarea implicita, iar o valoare negativa reduce acest spatiu. Daca valoarea este egala cu 0 atunci spatiul dintre cuvinte este fixat la valoarea prestabilita, neproducand de fapt nici un efect, fiind similar cu valoarea normal. Exemplu: **h2 { word-spacing:8px; }**

3. Spatiul intre linii

Inaltimea randurilor se refera la spatiul dintre liniile paragrafului. Modificarea spatiului intre linii poate avea ca efect o mai buna si usoara citire a textului in cazul in care aceasta este mai mare (creaza o regiune cu spatiu intre liniile de text). Alteori, daca inaltimea este mai mica (folosita la randurile de titlu), poate conferi un aspect mai stilat.
Pentru a modifiica inaltimea randurilor se foloseste proprietatea **line-height**, urmata de o valoare care poate fi exprimata in trei moduri:

- un **numar** care inseamna multiplicarea dimensiunii fontului cu numarul respectiv pentru a obtine valoarea spatierii;
- o **valoare de tip absolut**, exprimata in pixeli, puncte sau orice alta unitate de masura folosita in CSS, care indiferent de dimensiunea stabilita a fontului, inaltimea liniei va avea o valoare exacta;
- un **procent**, care stabileste inaltimea liniei ca fiind egala cu un anumit procent din dimensiunea fontului folosit pentru scrierea textului.

Sintaxa generala pentru modificarea spatiului intre linii este: **selector { line-height:valoare }**

Iata un exemplu in care marim inaltimea liniei cu 50% din dimensiunea fontului: **h1 { line-height:150%; }**

Inaltimea liniei poate de asemenea fi definita si in cadrul proprietatii "font-size", prin introducerea caracterului "/" urmat de valoarea pentru inaltimea randului, imediat dupa dimensiunea fontului.
Exemplu: **h1 { font-size:12px/28px; }**

4. Marimea (tipul) literelor

Exista cazuri in care se doreste afisarea textului cu un anume tip de litere. Folosind CSS, se poate specifica explicit ca textul sa apara cu majuscule initiale, majuscule, minuscule, combinat sau asa cum a fost el scris.
Cu ajutorul proprietatii **text-transform** se poate controla marimea literelor din text, indiferent de modul cum a fost el scris initial.Pentru a schimba tipul literelor dintr-un text, proprietatea **text-transform** va fi urmata de o valoare care poate fi:

- **capitalize** – pentru ca primul caracter din fiecare cuvant sa fie scris cu majuscula;

- **uppercase** – pentru ca toate caracterele sa fie scrise cu majuscule;
- **lowercase** - pentru ca toate caracterele sa fie scrise cu litere mici;
- **none** – pentru ca textul sa fie afisat asa cum este.

Sintaxa generala este: **selector { text-transform:valoare }**

Iata un exemplu in care toate literele vor fi afisate cu majuscule: **h1 { text-transform:uppercase; }**

Proprietatea "text-transform" este utila atunci cand textele sunt create in mod dinamic (de exemplu, in cazul in care ele provin dintr-o baza de date sau form HTML).

5. Alinierea textului.

Un text poate fi aliniat de la marginea din stanga, din dreapta, centrat sau la ambele margini (stanga-dreapta). Proprietatea **text-align** ofera posibilitatea controlului asupra alinierii textului. Definirea alinierii se face specificand dupa proprietatea **text-align** una din valorile: **left**, **right**, **center**sau **justify**.

Forma generala este: **selector { text-align:valoare }**

Proprietatea poate fi aplicata numai elementelor la nivel de bloc (DIV, UL, Hx, p), valoarea sa implicita fiind in majoritatea cazurilor stabilita la valoarea "left". Iata un exemplu in care textul din eticheta "h2" este aliniat "stanga-dreapta": **h2 { text-align:justify; }**

- In cazul in care alinierea textului este "stanga-dreapta", spatierea cuvintelor si a literelor se schimba pentru a se obtine linii de aceeasi lungime.

6. Alinierea pe verticala a textului

Cu ajutorul proprietatii **vertical-align** se poate preciza pozitia unor elemente (in linie) in raport cu textul din jurul lor. Asta inseamna ca proprietatea poate fi folosita doar cu etichete in linie care nu determina un salt la linie noua, de obicei utilizandu-se cu eticheta "span".

Sintaxa pentru utilizarea proprietatii **vertical-align** este: **selector { vertical-align:valoare }**

Pentru definirea alinierii pe verticala a textului se foloseste una din urmatoarele optiuni:

- **super** – pentru scrierea textului in stil exponent, deasupra liniei de baza;
- **sub** - pentru scrierea textului in stil indice, sub linia de baza;
- **baseline** – pentru scrierea textului pe linia de baza;
- una din valorile: "top", "middle", "bottom", "text-top", "text-bottom" – pentru a alinia textul relativ la alinierea parintelui acestuia;
- valoare **procentuala** – care ridica sau coboara linia de baza a elementului proportional cu dimensiunea fontului elementului parinte. Iata un exemplu in care textul din eticheta va aparea ca exponent fata de textul din stanga-dreapta lui: **span { vertical-align:super; }**

7. Formarea paragrafelor

Cu ajutorul proprietatii **text-indent** se poate stabili ce spatiu suplimentar este inserat la inceputul primei linii de text dintr-un paragraf. Pentru aceasta, proprietatea **text-indent** este urmata de o valoare exprimata intr-o unitate de masura (pixeli sau em) sau in procente (proportional cu latimea paragrafului).

Sintaxa generala este: **selector { text-indent:valoare }**

Valorile pozitive determina o indentare tipica (spre dreapta), in timp ce valorile negative determina o indentare inspre argine, fiind necesar sau marirea umplerii (padding) sau marirea marginilor.

Iata un exemplu de cod CSS in care va fi determinat un spatiu de 20 pixeli la inceputul paragrafului din eticheta <p>:

p { text-indent:20px; }

Daca se utilizeaza o valoare exprimata in unitati "em", atunci rezultatul va fi relativ la dimensiunea fontului elementului asupra caruia se aplica. Pentru eliminarea decalarii se utilizeaza valoarea 0.

8. Aplicarea elementelor decorative

CSS permite ornarea textului prin scoaterea lui in evidenta in patru moduri: subliniere, supraliniere, taierea textului cu o linie sau crearea de texte care clipesc.

Aplicarea de elemente ornamentale unui text se realizeaza prin proprietatea **text-decoration**, urmata de una din valorile:

- **underline** – pentru a sublinia textul
- **overline** – pentru a trasa o linie deasupra textului
- **line-through** – pentru a taia textul cu o linie
- **blink** – pentru a face textul sa apara si dispara intermitent

Forma generala este : **selector { text-decoration:valoare }**

Pentru a elimina decoratiile se foloseste valoarea "none". Prin folosirea acestei valori se elimina si sublinierile link-urilor, chiar daca navigatorul are prestabilit sa le arate subliniat.
Iata un exemplu de cod in urma caruia link-urile nu vor mai fi afisate subliniat: **a:link { text-decoration:none; }**

Elementul decorativ "blink" a prezentat probleme, fiind de curand eliminat de unele navigatoarelor.

9. Spatiile albe

Intreruperile de linie si spatiile albe dintr-un document HTML sunt tratate ca un singur spatiu sau sunt ignorate, daca nu este utilizata eticheta <pre>. CSS permite afisarea acestor spatii, precum si a intreruperilor textului, utilizand proprietatea **white-space**, urmata de valoarea "**pre**". Forma generala este: **selector { white-space:valoare }**

In locul cuvantului "valoare" poate fi adaugat: "pre", "nowrap" sau "normal"; Utilizarea valorii "**nowrap**" impiedica trecerea la linia urmatoare. Valoarea "normal" permite navigatorului sa decida asupra modului de tratare a spatiilor. De obicei, acestea vor fi transformate intr-un singur spatiu. Iata un exemplu in urma caruia spatiile albe si trecerea la linia noua vor fi afisate asa cum au fost scrise: **p { white-space:pre; }**

Spre deosebire de eticheta <pre> care schimba fontul cu unul monospatiat, valoarea "pre" a proprietatii **white-space** nu are nici un efect asupra fontului elementului la care se aplica.

Lectia 5 - Configurari pentru culori si fundal

Daca HTML permitea configurarea unor culori si elemente grafice de fundal doar pentru anumite elemente, CSS permite definirea culorii si a fundalului pentru orice element din pagina web.

1. Configurarea fundalului

Se poate schimba fundalul (background) intregii pagini, sau pentru un anumit element din pagina, fie el de tip DIV, paragraf, imagine, tabel sau formular.

a) **Culoarea fundalului.** Pentru a schimba culoarea fundalului se utilizeaza proprietatea **background-color**, urmata de o valoare care reprezinta numele culorii sau valoarea RGB, exprimata in cod hexa.
Sintaxa generala este: **selector {background-color:valoare }**

Daca se foloseste valoarea "transparent" navigatorul va afisa culoarea prestabilita sau cea a elementului parinte.
Pentru schimbarea fundalului intregii pagini se aplica aceasta proprietate selectorului "body".
Iata un exemplu in care culoarea de fundal a paginii este schimbata in albastru: **body { background-color:#0000ff; }**

b) **Imaginea de fundal.**

Puteti utiliza pentru fundal si o imagine, pentru aceasta se foloseste proprietatea **background-image**, urmata de adresa URL a locatiei imaginii. Sintaxa generala este: **selector { background-image:url('adresa_URL'); }**

Unde "adresa_URL" este calea si numele imaginii care va fi folosita. Daca in locul adresei URL se foloseste valoarea "none", navigatorul nu va folosi nici o imagine. Iata un exemplu in care pentru fundalul paginii este folosita o imagine: **body { background-image:url('cale/imagine.gif'); }**

c) **Repetarea imaginii de fundal**

Pe langa posibilitatea afisarii unui element grafic ca fundal, CSS ofera si un mod de repetare a acestuia.
Pentru a repeta imaginea de fundal se foloseste proprietatea **background-repeat**, urmata de o valoare care poate avea una din urmatoarele optiuni:

- **repeat** – pentru a repeta imaginea pe toata suprafata elementului respectiv;
- **repeat-x** – pentru a repeta imaginea numai pe orizontala;
- **repeat-y** – pentru a repeta imaginea numai pe verticala;
- **no-repeat** – pentru a afisa imaginea o singura data, fara repetare.

Forma generala este: **selector { background-repeat:valoare }**

Iata un exemplu de cod in care imaginea folosita pentru fundalul paginii se va repeta pe verticala pe toata inaltimea documentului: **body { background-repeat:repeat-y; }**

d) **Controlul imaginii de fundal**

CSS poate defini modul de tratare a fundalului la derularea paginii, pentru aceasta se foloseste proprietatea **background-attachment** cu optiunea "**fixed**", pentru a lipi imaginea de fundal de fereastra navigatorului sau optiunea "**scroll**", pentru a permite derularea imaginii de fundal alaturi de elementul corespunzator atunci cand vizitatorul deruleaza pagina.Sintaxa generala este: **selector { background-attachment:valoare }**

Iata un exemplu practic aplicat imaginii folosita pentru fundalul paginii: body { background-attachment:scroll; }

e) **Pozitia imaginii de fundal**

Pentru a pozitiona imaginii de fundal in functie de coltul din stanga-sus al elementului se utilizeaza proprietatea **background-position** urmata de doua valori (coordonate x si y) separate prin spatiu, 'x' pentru pozitia orizontala si 'y' pentru cea verticala.Forma generala a acestei proprietati este: **selector { background-position:valoare }**

Coordonatele x si y pot fi exprimate ca valori absolute sau ca procente. Se pot folosi si cuvintele cheie: "**left**", "**center**" sau "**right**" pentru coordonata x; respectiv "**top**", "**center**" sau "**bottom**" pentru coordonata y.

Iata un exemplu in care imaginea de fundal a paginii este pozitionata la 10 pixeli (x si y) relativ la coltul din stanga-sus:

```
body { background-position:10px 10px; }
```

f) **Stabilirea simultana a proprietatilor fundalului**

Toate proprietatile fundalului pot fi configurate cu ajutorul proprietatii **background**. Se poate stabili simultan printr-o lista de valori proprietatile fundalului, enumerate in orice ordine si separate prin spatiu.

Cand se utilizeaza proprietatea **background**, nu este necesar sa fie specificate toate proprietatile, cele nespecificate fiind stabilite la valoare lor implicita.

Sintaxa generala este: **selector {background:val_bg-color val_bg-img val_bg-repet val_bg-attach val_bg-position }**

Valoarea implicita a proprietatii **background-color** este "transparent".

Valoarea implicita a proprietatii **background-image** este "none".

Valoarea implicita a proprietatii **background-attachment** este "scroll".

Valoarea implicita a proprietatii **background-position** este "top left".

Iata si un exemplu in care sunt aplicate mai multe proprietati pentru background, in aceeasi definitie:

```
div { background:#e8e8fe url('imagine.jpg') 50% repeat }
```

DIV-ul va avea culoarea de fndal "#e8e8fe", cu o imagine pozitionata la jumatatea distantei X, Y si care se repeta.

2. Stabilirea culorii din prim-plan.

Proprietatea **color** poate schimba culoare oricarui element, fie el text, linie orizontala sau element de tip formular.Sintaxa generala pentru utilizarea acestei proprietati este: **selector { color:valoare }**

Valoarea culorii poate fi:

- **numele culorii** - una din cele 16 culori predefinite (red, green, blue, white, silver, ...)
- **valoare in hexa** - sub forma #rrggbb
- **valoare RGB** - sub forma rgb(r, g, b), unde r, g, b pot fi reprezentate prin numere intregi cuprinse intre 0 si 255 sau procente de rosu, verde si albastru cu valori intre 0% si 100%.

Iata un exemplu de cod CSS in care culoarea textului din eticheta "h3" este definita verde:

```
h3 { color:green; }
```

Lectia 6 - Controale pentru chenare si afisarea elementelor

Indiferent daca o eticheta este de sine statatoare sau imbricata in alte etichete, aceasta poate fi tratata ca un element distinct pe ecran si poate fi controlata prin intermediul CSS.

Orice element creat intr-o foaie de stil este prezentat in propriul lui cadru.

Notiunea de element se refera la componentele unui document HTML, configurate prin intermediul etichetelor HTML.

Caseta (cadrul elementului) dispune de numeroase proprietati, intre care se numara: marginile, chenarul, completarea (umplerea, cunoscuta ca "padding"), latimea si inaltimea, care pot fi modificate cu ajutorul CSS.

Elementele HTML au patru laturi (sus, dreapta, jos si stanga) la care se pot aplica proprietatile CSS specifice, restul proprietatilor CSS (font, text, culoare, fundal) se aplica in continutului casetei. Continutul poate include texte, liste, formulare sau imagini.

1. Latimea si inaltimea unui element

Latimea si inaltimea elementelor pot fi specificate cu ajutorul proprietatilor **width** si **height**.

In mod prestabilit acestea sunt determinate automat de catre navigator, fiind egale cu necesarul afisarii intregului continut.

Pentru definirea latimii si inaltimii se folosesc urmatoarele tipuri de valori:

- o valoare de tip **numeric**, de obicei in pixeli;
- un **procent**, care stabileste o valoare proportionala in functie de cea a elementului parinte;
- valoarea **auto**, care foloseste latimea si inaltimea calculata de navigator, si care reprezinta cantitatea de spatiu maxim pe care o poate ocupa elementul pentru afisarea intregului continut.

Sintaxa generala pentru configurarea latimii si inaltimii este:

- **selector { width:valoare; height:valoare }**

Iata un exemplu in care elementul cu id="un_id" va avea latimea de 300 pixeli si inaltimea de 500 pixeli:
#un_id { width:300px; height:500px; }

2. Marginile unui element

Proprietatea **margin** permite stabilirea distantei dintre un element si alte elemente alaturate, prin specificarea unei valor pentru marginea din fiecare latura (sus, dreapta, jos, stanga).
Daca specificati cele 4 valori in aceeasi definitie, acestea reprezinta marginile elementului in urmatoarea ordine: sus, dreapta, jos, stanga.
Daca specificati doar o valoare, aceasta va fi aplicata tuturor marginilor.
In cazu in care specificati doua sau trei valori, atunci valorile care lipsesc sunt copiate dupa valorile marginilor opuse.
Sintaxa generala pentru configurarea marginilor unui selector este:

- **selector { margin:valoare/valori; }**

Tipul de valoare folosit poate avea urmatoarele optiuni:

- o valoare de tip **lungime (numerica)** – care poate fi si negativa;
- o valoare **procentuala (procent)** – creaza o margine proportionala cu latimea elementului parinte;
- valoarea **auto** – lasa controlul marginilor la decizia navigatorului.

Iata un exemplu in care etichetele DIV vor avea distanta pentru marginea de sus 10 pixeli, dreapta 5 pixeli, jos 8 pixeli si in stanga 15 pixeli: **div { margin:10px 5px 8px 15px; }**

Exista posibilitatea de a stabili valoarea marginii doar pentru o singura latura, fara a tine cont de celelalte margini. Pentru aceasta sunt folosite proprietatile **margin-top**, **margin-bottom**, **margin-left** si **margin-right** cu aceleasi valori ca si in cazul proprietatii **margin**, dupa cum puteti vedea mai jos.

selector { margin-top:valoare; }
selector { margin-bottom:valoare; }
selector { margin-left:valoare; }
selector { margin-right:valoare; }

3. Chenarul unui element

Proprietatea **border** permite stabilirea simultana a atributelor chenarului pentru toate cele patru laturi ale casetei. Aceasta foloseste 3 valori distincte, pentru a configura latimea (grosimea), stilul si culoarea liniei bordurii.
Sintaxa generala pentru definirea chenarului este: **selector { border:val_grosime val_stil val_culoare; }**

Unde
- prima valoare (val_grosime) reprezinta grosimea chenarului si poate fi una din urmatoarele tipuri:
- o valoare de tip lungime (valoarea 0 determina anularea afisarii chenarului);
- un cuvant cheie (**thin, medium, thick**) care caracterizeaza o dimensiune relativa.
- a doua valoare (val_stil) reprezinta numele stilului atribuit chenarului (valoarea "none" anuleaza afisarea chenarului);
- ultima valoare (val_culoare) reprezinta culoarea, exprimata in cod hexa sau valoare RGB. Iata un exemplu in care div-urile vor avea o bordura groasa de 1 pixel, solid si culoare albastra: div { border:1px solid blue; }

Pot fi folosite si proprietati separate pentru fiecare: "border-width", "border-style" si "border-color"; prezentate in continuare.Un atribut care poate fi folosit in configurarea chenarului este **border-width**, acesta furnizeaza o metoda rapida pentru stabilirea latimii celor patru laturi din jurul unei casete.
Sintaxa generala este: **selector { border-width:valoare; }**

Daca specificati toate cele patru valori, ele sunt aplicate in ordinea: sus, dreapta, jos, stanga. Daca specificati doar o valoare, ea va fi aplicata tuturor marginilor. Daca folositi doua sau trei valori, valorile care lipsesc sunt copiate din valorile marginilor opuse lor in caseta.Aspectul bordurii poate fi stabilit prin atributul **border-style**; stilul este aplicat celor patru margini si este definit astfel: **selector { border-style:valoare; }** Pentru specificarea valorii, pot fi folosite urmatoarele tipuri:

Valoare	Aspect
none (nici unul)	
dotted (linie punctata)	
dashed (linie intrerupta)	
solid (linie plina)	
double (linie dubla)	
groove (tridimensional brazdata)	
ridge (tridimensional stil creasta)	
inset (tridimensional inauntru)	
outset (tridimensional in afara)	

Ultimele patru valori sunt reprezentate tridimensional, iar in cazul in care ele nu sunt acceptate de catre navigator, inlocul lor se va folosi valoarea cu linie plina.

Un alt atribut care poate fi utilizat in configurarea chenarului este **border-color**, acesta stabileste culoarea pentru toate cele patru laturi ale chenarului si foloseste pentru valoarea ei un singur cuvant cheie exprimat in cod hexa, valoare RGB sau numele culorii. Sintaxa generala este:**selector { border-color:valoare; }**

Puteti stabili culoarea fiecarei margini si separat, folosind proprietatile:

border-top-color, **border-right-color**, **border-bottom-color** si **border-left-color**

Fiecare latura a chenarului poate avea valori configurate in mod separat (prin care se stabilesc stilul si culoarea fiecarei borduri din jurul elementului) dupa cum este prezentat mai jos:

selector { border-top:grosime stil culoare; }
selector { border-bottom:grosime stil culoare; }
selector { border-left:grosime stil culoare; }
selector { border-right:grosime stil culoare; }

4. Adaugarea spatiului in interior, in jurul elementului

Proprietatea **padding** adauga o cantitate de spatiu suplimentar in jurul continutului elementului, in interiorul chenarului, intre bordura si continut.Forma generala pentru utilizarea acestei proprietati este:

selector { padding: valoare/valori; }

Valoarea pentru completarea spatiului poate fi una din urmatoarele: o valoare de tip **lungime sau** o valoare **procentuala** – creaza umplerea in raport cu latimea elementului parinte.

Cand specificati toate cele patru valori, ele sunt aplicate in ordinea: sus, dreapta, jos, stanga.
Daca specificati doar o valoare, aceasta va fi aplicata tuturor celor patru directii.
Daca specificati doua sau trei valori, atunci valorile care lipsesc sunt copiate dupa valorile laturilor opuse.
Iata un exemplu in care etichetele DIV vor avea distanta padding: sus 4 pixeli, dreapta 2 pixeli, jos 3 pixeli si in stanga 2 pixeli: **div { padding:4px 2px 3px 2px; }**

Fiecare latura a chenarului poate avea proprietatea **padding** configurata separat:

selector { padding-top:valoare; }
selector { padding-bottom:valoare; }
selector { padding-left:valoare; }
selector { pading-right:valoare; }

5. Elemente float

CSS ofera posibilitatea de a "infasura" anumite elemente in jurul altora. Acest lucru se face prin intermediul proprietatii **float**, care determina o mutare fortata a elementului in directia data de valoarea mentionata, lasand loc liber in spatiul opus directiei, care va fi ocupat de urmatorul element din codul HTML.
Sintaxa generala pentru utilizarea acestei proprietati este: **selector { float:valoare; }**

Unde "vaoare" poate fi" **left**, **right** si **none;** Valoarea "none" permite elementului sa fie plasat acolo unde este posibil, iar celelalte valori forteaza plasarea elementului in stanga sau in dreapta ecranului, textul din exteriorul etichetei HTML respective va fi plasat in partea opusa, in jurul elementului.Iata un exemplu in care elementul cu id="un_id" va fi fortat sa fie pozitionat in dreapta celorlaltor elemente**: #un_id { float:right;** }

6. Anularea plasarii unui element in spatiul eliberat de "float"

Asemanator cu actiunea etichetei HTML
, care trece pe un nou rand elementele ce o preced, este si proprietatea **clear**. Cu ajutorul acesteia se poate anula ocuparea unui spatiu care a fost lasat liber de un element care utilizeaza proprietatea "float".Sintaxa generala pentru utilizarea proprietati **clear**: **selector { clear:valoare; }**

Unde "valoare" specifica latura in jurul careia este interzisa infasurarea textului si poate lua valoarea: **left**, **right** sau **both**. Se poate folosi si valoarea "none", ceea ce are ca efect anularea altor atribute ale proprietatii **clear**.
De exemplu, daca aveti un DIV pozitionat fortat in dreapta, un paragraful care urmeaza dupa el va fi afisat in stanga DIV-ului respectiv. Pentru a anula acest lucru, ca paragraful sa fie afisat sub DIV, se foloseste proprietatea "clear", precum in exemplul urmator**: #un_div { float:right; } p { clear:both; }**

7. Afisarea si ascunderea elementelor

Proprietatea **display** poate stabili daca un element va fi de tip block, incluzand linii noi deasupra si dedesubtul sau, daca este inclus in linie, daca este tratat sub forma de lista sau daca elementul este afisat sau nu.
Sintaxa generala pentru aceasta proprietate este: **selector { display:valoare; }**

Unde "valoare" poate fi una din urmatoarele optiuni:

- **list-item** – plaseaza in prima linie a textului un indicator pentru articole de tip lista, dar si un salt deasupra si dedesubtul articolului;
- **block** – defineste eticheta ca fiind de tip bloc si plaseaza un salt la linie noua deasupra si dedesubtul ei;
- **inline** – defineste eticheta ca o eticheta in linie si elimina caracterele de salt la linie noua;
- **none** – determina ascunderea elementului si neafisarea lui de catre navigator; codul acestuia este incarcat insa continutul sau nu e afisat in pagina.

Iata un exemplu in care elementele <li> sunt asezate in linie, iar un element cu id="un_id" va fi ascuns in pagina:

li { display:inline; } #un_id { display:none; }

Proprietatea **display** nu trebuie confundata cu **visibility**. Spre deosebire de **visibility** care nu afiseaza elementul dar lasa liber spatiu pentru element, sintaxa **display: none** elimina complet elementul din pagina.

Lectia 7 - Controale de pozitionare

Pozitionarea elementelor folosind CSS este mai precisa decat prin intermediul obiectelor grafice HTML sau a tabelelor, afisarea facandu-se mult mai rapid. Prin intermediul CSS este permisa pozitionarea exacta sau relativa a elementelor intr-o fereastra sau in raport cu alte elemente. Fereastra navigatorului este suprafata in care sunt afisate toate elementele. Ea poate fi redimensionata sau pozitionata pe ecran, sau poate fi divizata in alte ferestre prin intermediul cadrelor. Toatele elementele amplasate in fereastra sunt pozitionate relativ la coltul din stanga-sus.

1. Stabilirea modului de pozitionare

Prin utilizarea proprietatii **position** se poate specifica pozitia elementului in pagina web.
Un element poate avea una din urmatoarele valori de pozitionare: **static**, **relative**, **absolut** si **fixed**.
Tipul de pozitie indica navigatorului cum sa trateze amplasarea unui element intr-o fereastra.

a) Pozitionarea static

Valoarea initiala, prestabilita, a pozitionarii elementelor in fereastra este "static". Cand nu este specificata o pozitionare "relativa", "absoluta" sau "fixa"; elementele sunt dispuse unul dupa altul in interiorul documentului.Sintaxa pentru specificarea pozitionarii **static** este: **selector { position:static }** Un element static nu poate fi repozitionat in mod explicit.

b) Pozitionarea relativa

Un element cu pozitionare "relativa" este amplasat la locul sau in cadrul ferestrei sau a elementului parinte, in sensul ca el apare dupa toate elementele dinaintea sa, respectiv inaintea tuturor elementelor amplasate dupa el.
Sintaxa pentru specificarea pozitionarii **relative** este: **selector { position:relative }**
Elementele pozitionate relativ pot fi mutate din locatia lor folosind proprietatile "top" si "left" sau "bottom" si "right".

c) Pozitionarea absoluta

Pozitionarea absoluta are ca efect crearea unui element neafectat de restul documentului, plasarea lui in fereastra fiind facuta intr-o locatie precisa, definita prin intermediul coordonatelor **x** si **y**, indiferent de pozitiile altor elemente.
Sintaxa pentru specificarea pozitionarii **absolute** este: **selector { position:absolute }**
Originea (punctul de coordonate 0,0) este coltul din stanga-sus al ferestrei sau al obiectului in care este inclus elementul pozitionat absolut.

d) Pozitionarea fixa

Pozitionarea fixa a unui element este aproximativ la fel cu cea absoluta, cu diferenta ca la derularea paginii elementul fixat ramane pe pozitia lui initiala, fara a se derula.
Sintaxa pentru specificarea pozitionarii **fixed** este: **selector { position:fixed }**

2. Pozitionarea in raport cu latura de sus, respectiv stanga

Dupa stabilirea tipului de pozitionare, se poate determina distanta intre punctul de origine si laturile de sus si din stanga ale elementului sau parinte, folosind proprietatile **top** si **left**
Forma generala este: **selector { top:valoare; left:valoare; }**

- Unde "valoare" poate fi:
  - o valoare de tip numeric, care defineste distanta dintre laturile ferestrei sau a elementului parinte si laturile elementului;
  - o valoare procentuala, care semnifica deplasarea fata de laturile ferestrei sau a elementului parinte;

- valoarea auto, care permite navigatorului sa calculeze el insusi valoarea.

Pot fi utilizate proprietatile **top** si **left** sau **bottom** si **right**, de asemenea pot lua si valori negative.
In cazul in care este vorba despre elemente imbricate, acestea vor fi deplasate solidar cu elementul parinte daca acesta are *position:relative*. Iata un exemplu in care o eticheta <h1> este pozitionata la o distanta de 10 pixeli fata de marginile sus si stanga ale unui DIV in interiu caruia se afla:

```
<style type="text/css">
   div { position:relative; }
   h1 { top:10px; left:10px; }
</style>
<div>   <h1>Text...</h1> </div>
```

3. Pozitionarea in raport cu latura de jos, respectiv dreapta

In unele cazuri este necesara pozitionarea in raport cu laturile de jos, respectiv dreapta. In acest caz originea va fi coltul din dreapta-jos al ferestrei sau al elementului parinte.
Definirea marginilor fata de latura de jos, respectiv din dreapta se face cu ajutorul proprietatilor **bottom** si **right**; ele pot lua aceleasi valori ca si "left" si "top", de asemenea pot fi combinate cu acestea.
Forma generala este: **selector { bottom:valoare; right:valoare; }**

In cazul in care pentru acelasi element sunt stabilite atat marginile "top / left" cat si "bottom / right", rezultatul afisat depinde de navigator, dar in mod prestabilit se folosesc pozitiile **top** si **left**.

4. Pozitionarea in spatiul 3D

Elementele pot primi o a treia dimensiune, si anume asezarea lor in stiva, unele in raport cu altele.
Amplasarea se face in mod automat, incepand cu valoarea 0 si continuand prin incrementare cu o unitate, in ordinea aparitiei lor in documentul HTML si relativ la elementele parinte.
Pentru pozitionarea elementelor in stiva unele peste altele se foloseste proprietatea **z-index**. Valoarea acestei proprietati fiind relatia tridimensionala a elementului in raport cu alte elemente din fereastra.
Sintaxa generala pentru proprietatea **z-index** este: **selector { z-index:valoare; }**

Unde "valoare" este un numar intreg, pozitiv, 0 sau negativ.In cazul in care continutul elementelor se suprapune, elementul cu numarul de ordine mai mare apare deasupra elementului cu numar mai mic.Utilizarea unei valori negative determina amplasarea elementului dedesubtul parintelui sau cu atatea niveluri cu cate indica **indexul z**.

5. Includerea unui element absolut intr-un element relativ

Un element poate fi pozitionat exact in cadrul unei ferestre, sau el poate fi inclus intr-un element cu pozitionare relativa. In cazul includerii unui element pozitionat absolut intr-un element pozitionat relativ, elementul absolut este pozitionat folosind ca origine coltul din stanga-sus al elementului relativ.

Iata un exemplu de cod HTML in care o eticheta (cu class="absolut") este inclusa (imbricata) in alta eticheta (cu class="relativ"):

```
    <div class=”relativ”>
...
    <div class=”absolut”> ... </div>
...
</div>
```

Pentru a face o pozitionare absoluta a elementului inclus (care are class="absolut") in elementul parinte (care are class="relativ"), caruia ii definim o pozitionare relativa; scriem in foaia de stil urmatorul cod:

```
.relativ {position:relative; top:30px; left:50px; }
.absolut {position:absolute; top:15px; left:0px; }
```

6. Includerea unui element relativ intr-un element absolut

Cand un element pozitionat absolut este inclus intr-un element pozitionat relativ, primul foloseste ca origine coltul din stanga-sus al parintelui.
In cazul in care un element pozitionat relativ este plasat in interiorul unui element pozitionat absolut, acesta se va deplasa o data cu elementul absolut.
Iata un exemplu de cod HTML in care o eticheta (cu class="relativ") este inclusa (imbricata) in alta eticheta (cu class="absolut"):

```
    <div class=”absolut”>
...
```

 ...
...

Pentru a face o pozitionare relativa a elementului inclus (care are class="relativ") in elementul parinte (care are class="absolut"), caruia ii definim o pozitionare absoluta; adaugam in foaia de stil urmatorul cod:

.absolut {position:absolute; top:20px; left:25px; }
.relativ {position:relative; top:10px; left:5px; }

Lectia 8 - Configurari pentru vizibilitate si mouse

CSS ofera posibilitatea de a afisa sau ascunde unele elemente sau parti ale unor elemente.

1. Stabilirea vizibilitatii unui element

Proprietatea **visibility** poate controla faptul ca un element sa fie vizibil sau nu. Chiar daca elementul este invizibil, el va ocupa un spatiu liber in pagina, acolo unde ar fi trebuit sa fie afisat.

Sintaxa pentru folosirea acestei proprietati este urmatoarea:

- **selector { visibility:valoare }**

Unde "valoare" poate fi:

- **hidden** – ascunde elementul de la afisarea pe ecran;
- **visible** – afiseaza elementul;
- **inherit** – vizibilitatea elementului este mostenita de la elementul parinte.

Iata un exemplu in urma caruia textul din interiorul etichetei <h1> va fi invizibil: **h1 { visibility:hidden; }**

Pentru a ascunde complet afisarea unui element in pagina, se poate utiliza formula **display:none;**

2. Stabilirea suprafetei vizibile a unui element

Definirea suprafetei vizibile a unui element stabileste portiunea din elementul respectiv care este vizibila in fereastra navigatorului. Restul continutului acelui element nu dispare, ci este invizibil pentru vizitator.

Pentru a defini forma regiunii vizibile se foloseste proprietatea **clip** cu valoarea **rect** (definita prin patru valori separate prin spatii si incadrate de paranteze rotunde)

Forma generala este: **selector {clip:rect(val1 val2 val3 val4); }**

- Valorile construesc un patrulater, definesc distanta dintre coltul din stanga-sus al elementului si laturile de sus, dreapta, jos si stanga ale regiunii vizibile.

Iata un exemplu de cod CSS in care suprafata vizibila a unui element cu id="viz" va fi cea incadrata de valorile atributului **rect: #viz { clip: rect(25 300 125 100); }**

Daca se foloseste valoarea "auto", navigatorul calculeaza dimensiunea regiunii vizibile la 100%. Regiunile de vizibilitate pot avea doar forma dreptunghiulara.

3. Definirea pozitiei depasirii

Uneori elementele nu sunt in totalitate continute in casetele lor, caseta nefiind suficient de mare, astfel ca o parte a continutului nu este afisata sau depaseste marginea stabilita.

Proprietatea **overflow** permite tratarea continutului in exces, controland astfel modul de afisare sau nu al acesuia.

Forma generala a proprietatii **overflow** este:

- **selector { overflow:valoare; }**

- Unde "valoare" stabileste unde va fi plasata depasirea, folosind una din valorile:

- **visible** – extinde caseta elementului astfel incat sa incapa tot continutul sau, ignorand delimitarea suprafetei vizibile. Este optiunea prestabilita.
- **hidden** – ascunde continutul care nu incape in caseta elementului, si impiedica aparitia barei de derulare.
- **scroll** – adauga intotdeauna bare de derulare elementului, pentru ca sa se poata accesa tot continutul elementului.
- **auto** – barele de derulare apar doar atunci cand este necesar.

Iata un exemplu in care toate elementele cu class="extradim" vor avea bare de scroll:

.extradim { overflow:scroll; }

Pentru definirea proprietatii "overflow" doar pentru una din directii: orizontala sau verticala; adica bara de derulare sa fie disponibila sau nu doar pentru una din aceste directii, se poate folosi o alta varianta a acestei proprietati, si anume:

- **overflow-x:valoare** - pentru orizontala
- **overflow-y:valoare** - pentru verticala; Unde "valoare" poate fi una din valorile prezentate mai sus.

4. Aspectul indicatorului de mouse

In mod normal, aspectul indicatorului de mouse este determinat de browser. Navigatorul modifica indicatorul de mouse in functie de continutul deasupra caruia se afla acesta.
Uneori este dorita anularea sau completarea preferintelor navigatorului si configurarea unor aspecte personalizate.
Proprietatea **cursor** ajuta la stabilirea aspectului unui indicator de mouse.
Sintaxa este urmatoarea: **selector { cursor:valoare; }**

Unde "valoare" poate avea urmatoarele nume pentru indicatoarele de mouse:

Nume	Aspect
crosshair	+
hand	
pointer	
move	
n-resize	
ne-resize	
e-resize	
se-resize	
s-resize	
sw-resize	
w-resize	
nw-resize	
text	][
wait	
help	?

- Daca se foloseste valorea "auto", navigatorul decide asupra tipului de indicator folosit.

Lectia 9 – Pseudoclase

Pseudo-clasele permit adaugarea de stiluri CSS doar la anumite elemente ale aceluiasi selector, id sau clasa.
De exemplu, cand definiti un stil pentru un anumit tag HTML sau pentru o clasa, continutul din toate aceleasi tag-uri sau aceeasi clasa vor avea acel stil, iar daca doriti sa definiti un stil diferit pentru primul (sau ultimul) din acele tag-uri ori pentru primul continut definit de aceeasi clasa, puteti realiza asta folosind pseudo-clasele. De asemenea, acestea pot modifica stilul grafic al elementelor cand mouse-ul actioneaza asupra lor.
Pentru a functiona in Internet Explorer 7+ este necesar declararea <!DOCTYPE>, care se adauga la inceputul documentului HTML, mai multe detalii (in engleza) gasiti Aici: *http://www.w3schools.com/tags/tag_doctype.asp*
Sintaxa pentru utilizarea pseudo-clasei este urmatoarea:

element:pseudo-clasa { proprietate:valoare; }

- unde "element" este un selector, id sau clasa, iar "pseudo-clasa" este una din expresiile urmatoare:

- **active** - Adauga un stil unui element cand acesta este activat (*actionat prin click pe el*)
- **first-child** - Adauga un stil unui element care este primul din acel tip de elemente
- **focus** - Folosit pentru input-urile de formulare, le adauga un stil cand acestea sunt active (*dupa click si cursorul de text in ele*)
- **hover** - Adauga un stil unui element cand mouse-ul se afla deasupra lui
- **lang(cuvant)** - Adauga un stil unui element care are atributul *lang="cuvant"* (nu e suportat de Safari si IE mai mic de 8)
- **last-child** - Adauga un stil unui element care este ultimul din acel tip de elemente
- **link** - Adauga un stil unui link nevizitat
- **visited** - Adauga un stil unui link vizitat

Ca sa intelegeti mai bine cum functioneaza si ce fac pseudo-clasele, studiati exemplele care sunt prezentate in continuare.

1. Pseudo-clase cu selector

Selectoarele fac referire la tag-urile HTML pe care le denumesc (de ex.: **p** pentru <p>, **li** pentru <li>, **div** pentru , etc.).
In exemplul urmator este folosit "first-child" pentru paragraf:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html><head>
<title>Pseudo-clase - Ex. 1</title>
```

```
<style type="text/css">
p:first-child {
   color:blue;
}
</style>
</head><body>
<p>Un text din primul paragraf</p>
<p>Continut din al doilea paragraf</p>
</body></html>
```

- Rezultatul va fi urmatorul, primul paragraf va avea textul albastru.

Un text din primul paragraf

Continut din al doilea paragraf

Iata si un exemplu cu "hover" pentru

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html><head>
<title>Pseudo-clase - Ex. 2</title>
<style type="text/css">
li:hover {
   background-color:#88fe88;
}
</style>
</head>
<body>
<ul>
  <li>Text list 1 ...</li>
  <li>Text list 2 ...</li>
  <li>Text list 3 ...</li>
</ul>
</body></html>
```

- Cand pozitionati mouse-ul deasupra fiecarui LI, acesta va avea background verde, testati mai jos
  - Text list 1 ...
  - Text list 2 ...
  - Text list 3 ...

2. Pseudo-clase si clase

Trebuie stiut ca pseudo-clasele nu sunt acelasi lucru ca si clasele, acestea fiind cele care fac referire la valoarea atributului "class" si in CSS se adauga dupa un caracter punct (**.**).

Iata un exemplu in care sunt folosite pseudo-clasele "lang(cuvant)" (*pt. IE incepand cu versiunea 8*) sl "last-child" la o clasa ".test'

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html><head>
<title>Pseudo-clase - Ex. 3</title>
<style type="text/css">
.test:last-child {
   background-color:#88fe88;
}
.test:lang(da) {
   color:blue;
}
</style>
</head>
<body>
<ul>
  <li class="test">Text in list cu class 1 ...</li>
```

```
  <li class="test" lang="da">Text in list cu class 2 ...</li>
  <li class="test">Text in list cu class 3 ...</li>
</ul>
</body></html>
```

- Observati ca stilul definit pentru ".test:last-child" va fi aplicat doar ultimului element care are clasa "test", iar "lang(da)" se aplica numai elementului care pe langa *class="test"* are si *lang="da"*.

- Rezultatul este acesta:

- Text in list cu class 1 ...
- Text in list cu class 2 ...
- Text in list cu class 3 ...

Iata un alt exemplu in care sunt combinate o clasa (test), un selector (tag <i>) si "first-child":

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
<title>Pseudo-clase - Ex. 4</title>
<style type="text/css">
.test i:first-child {
   font-weight:bold
}
</style>
</head>
<body>
<p class="test"><i>Paragraf</i> cu mai multe tag-uri I: <i>text oblic</i>, alt test <i>oblic</i></p>
<p class="test">Alt paragraf: <i>sir inclinat si bold</i>, alt sir <i>italic</i></p>
</body></html>
```

- Stilul CSS va fi aplicat primelor tag-uri <i> din fiecare clasa ".test", dupa cu puteti vedea mai jos

Paragraf cu mai multe tag-uri I: *text oblic*, alt test *oblic*

Alt paragraf: ***sir inclinat si bold***, alt sir *italic*

- Daca doriti ca stilul sa fie aplicat tuturor etichetelor <i> din prima clasa, scrieti: **.clasa:first-child i**

- Daca vreti sa fie aplicat numai primului <i> din prima clasa, scrieti: **.clasa:first-child i:first-child**

3. Pseudo-clase cu id-uri si elemente de formular

In CSS, id-urile fac referire la valoarea atributului ID si se scriu dupa caracterul **#**.

Iata un exemplu in care este folosita pseudo-clasa "hover" impreuna cu un ID, iar "focus" impreuna cu o clasa (*focus nu functioneaza pe versiuni de IE mai mici de 8*)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html><head>
<title>Pseudo-clase - Ex. 5</title>
<style type="text/css">
#unid:hover {
   background-color:#dadafe;
}
.test:focus {
   background-color:#ededfe;
}
</style></head>
<body><form action="" method="post">
  <input type="text" class="test" /><br />
  <textarea cols="20" rows="5" class="test"></textarea><br />
  <input type="button" value="Buton" id="unid" />
</form></body></html>
```

- Stilul CSS definit in acest exemplu face ca atunci cand mouse-ul se afla deasupra butonului (care are id="unid") acesta sa isi schimbe culoarea de fundal, iar cand se da click pe zonele de text din formular (care au class="test"), la fel se modifica culoarea background.

Lectia 10 – Pseudo-elemente

Pseudo-elementele permit adaugarea de stiluri CSS anumitor parti din continutul unui element HTML.
De exemplu, cand definiti un stil pentru un anumit tag HTML (sau pentru o clasa), tot continutul incadrat de acel tag va avea acelasi stil, iar daca vreti sa adaugati un stil CSS diferit primei litere sau primlui rand dintr-un paragraf, se folosesc pseudo-elemente.
Sintaxa pentru utilizarea pseudo-elementelor este urmatoarea:

obiect_css:pseudo-element { proprietate:valoare; }

- unde "obiect_css" este un selector, id sau clasa, iar "pseudo-element" este una din expresiile urmatoare:

- **after** - Adauga continut dupa un element HTML (*nu e suportat de versiuni IE mai mici de 8*)
- **before** - Adauga continut inaintea unui element HTML (*nu e suportat de versiuni IE mai mici de 8*)
- **first-letter** - Adauga un stil css primului caracter dintr-un text
- **first-line** - Adauga un stil primei linii dintr-un text

Ca sa intelegeti mai bine cum functioneaza si ce fac pseudo-elementele, iata cateva exemple cu fiecare in parte.

1. after

Pentru a adauga un anumit continut (text, imagine, sunet .wav) prin CSS, se foloseste proprietatea **content** si valoarea ce reprezinta continutul respectiv: text se adauga intre ghilimele, iar imagine sau sunet .wav se adauga folosind ca valoare **url(adresa_fisier)**
In urmatorul exemplu va fi adaugat un continut text dupa fiecare element <h4>.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html><head>
<title>Pseudo-elemente - Ex. 1</title>
<style type="text/css">
h4:after {
   content:"MarPlo";
   color:blue;
}
</style></head>
<body>
<h4>Continutul din primul tag H4 ... </h4>
<h4>Un alt text din alt element H4</h4>
</body></html>
```

- Observati cum este adaugat cuvantul "MarPlo", imediat dupa ultimul caracter al fiecarui element <h4>, ca si cum ar face parte din textul initial, dar putand avea stil propriu.

In loc de selector puteti folosi si ID sau clasa.

2. Before este similar cu "after", continutul se adauga la fel, dar la inceputul elementului.
Pentru o mai mare difersitate, puteti combina pseudo-clasele cu pseudo-elemente folosind sintaxa:

obiect_css:pseudo-clasa:pseudo-element { proprietate:valoare; }

- *Pseudo-clasele sunt explicate in lectia anterioara.*

Ca sa vedeti efectul, iata un exemplu aplicat cu "before" acelorasi tag-uri H4 din exemplul precedent.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html><head>
<title>Pseudo-elemente - Ex. 2</title>
<style type="text/css">
h4:first-child:before {
   content:"MarPlo- ";
   color:blue;
```

```
}
</style></head>
<body>
<h4>Continutul din primul tag H4 ... </h4>
<h4>Un alt text din alt element H4</h4>
</body></html>
```

- De data aceasta continutul "MarPlo- " este adaugat imediat in fata si doar primului element <h4> (*precizat prin 'first-child'*

- Daca vreti sa adaugati in loc de continut text o imagine, de exemplu .gif, scrieti codul CSS astfel: **h4:first-child:before { content:url(imagine.gif); }**

3. First-letter aplica un stil CSS doar primului caracter dintr-un text.
In urmatorul exemplu este aplicat un stil CSS primului caracter din continutul fiecarui element definit printr-o clasa "test".

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html><head>
<title>Pseudo-elemente - Ex. 3</title>
<style type="text/css">
.test:first-letter {
   font-size:25px;
   color:red;
}
</style></head>
<body>
<p class="test">Textul din paragraf ...</p>
<div class="test">Alt continut int-un tag DIV, dar cu aceeasi clasa.</div>
</body></html>
```

- Dupa cum puteti vedea in rezultatul de mai jos, prima litera din continutul fiecarui tag ce are class="test" este de culoare rosie si marime 25px.

First-line aplica un stil CSS doar primei linii din continutul elementului HTML.Studiati exemplul urmator:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html><head>
<title>Pseudo-elemente - Ex. 4</title>
<style type="text/css">
.test:first-line {
   font-weight:bold;
   color:blue;
}
</style></head><body><p class="test">Paragraf cu mai multe linii<br />  A doua linie ...<br />
  Alta linie din acelasi paragraf.</p><div class="test">Continut pe doua linii intr-un tag DIV<br />
  A doua linie din DIV.</div></body></html>
```

- Rezultatul este urmatorul, comform codului CSS, prima linie din fiecare element cu class="test" are textul albastru si ingrosat.