În lecţiile precedente, ne-am familiarizat doar cu limbajul HTML şi cu proprietăţile sale. Am învățat cum să adăugăm conţinut în pagini, iar acum ne vom familiariza cu CSS, cu regulile sale de bază şi cu stilizarea elementelor. Până în momentul de faţă, am definit conţinuturile unei pagini HTML, însă, cu această ocazie nu am folosit atributele HTML pentru editarea culorilor, dimensiunilor şi aspectelor conţinuturilor propriu-zise, ci am specificat că pentru aceste scopuri folosim CSS, aşadar, începând cu această lecţie, lucrăm în paralel atât în HTML, cât şi în CSS. Mai exact, când, de exemplu, scriem un cod pentru un tabel, vom adăuga şi codul CSS pentru formatarea sa.

Înainte să începem cu scrierea primelor noastre coduri CSS, trebuie să ştim cum îi vom indica lui CSS un element HTML pe care vrem să-l schimbăm. În general, în aceste scopuri, folosim două atribute, ID şi CLASS.

Atributul ID

Fiecare element HTML poate să conţină un atribut ID. El se foloseşte pentru a determina un element în mod unic și concret. Aceasta înseamnă că pe o anumită pagină HTML poate exista doar un singur element cu o anumită valoare ID. Poate fi repetat într-un alt document.

Valoarea atributului ID poate să conţină litere/caractere de text (minuscule şi majuscule), anumite semne şi numere. Totuşi, trebuie să înceapă cu o literă. Se scrie ca orice alt atribut HTML în tag-ul HTML de început:

```
<h1 id="titlu_principal">Acesta este un titlu caruia ii este adaugat un ID</h1>
```

Astăzi, ID-ul se foloseşte cel mai des pentru stilizarea CSS şi pentru programarea JavaScript. CSS poate folosi ID-ul pentru a aplica o anumită regulă CSS doar pe un singur element (vom vedea mai târziu cum funcționează CSS). De asemenea, JavaScript foloseşte ID-ul într-un

mod similar, pentru a conecta anumite elemente ale paginii cu codul său.

Simpla adăugare a atributului ID pe un element nu va influenţa afişarea acestuia. Îi adaugă doar o etichetă, practic un nume pe care îl putem folosi mai târziu, dacă este necesar.

ID-ul se mai numeşte şi atribut global (*global attribute*), deoarece se poate aplica pe orice element.

Atributul CLASS

Atributul CLASS este foarte similar cu atributul ID menționat anterior, singura diferenţă constând în faptul că acesta poate apărea de mai multe ori pe aceeași pagină HTML. Uneori, este util să etichetăm mai multe elemente cu aceeaşi clasă (CLASS), pentru a le stiliza pe toate deodată cu CSS. Cu alte cuvinte, adăugăm o clasă pe mai multe elemente şi le stilizăm pe toate dintr-odată în CSS.

De asemenea, o altă diferenţă în raport cu ID-ul constă în faptul că un element poate conţine în acelaşi timp, mai multe clase. Le scriem pe toate într-un singur atribut CLASS, însă le separăm printr-un spaţiu gol. Mai jos, analizăm exemplul:

```
<p class="paragraf">Primul paragraf</p>
<p class="paragraf">Al doilea paragraf</p>
<p class="paragraf_separat">Al treilea paragraf</p>
<p class="paragraf">Al patrulea paragraf</p>
```

În acest exemplu, putem atribui aceeaşi regulă CSS tuturor celor patru paragrafe şi putem adăuga o altă descriere doar pentru al treilea paragraf (prin clasa „paragraf_separat”).

O întrebare care se pune frecvent este: Cum stabilim dacă vom folosi atributul ID sau atributul CLASS şi când trebuie să le folosim? Răspunsul nu poate fi simplu şi nici nu există doar unul corect. După cum am menţionat deja, atributele ID şi CLASS doar determină sau distanţează un element de mediul său. În diferite circumstanţe, adăugăm atribute diferite. De asemenea, diferiţi autori vor proceda diferit în situaţii identice, însă vor obţine probabil acelaşi rezultat. Pentru moment, nu vă bateți capul cu această problemă, **reţineţi doar regulile şi diferenţele dintre atributele ID şi CLASS**, iar totul vă va fi mult mai clar când vom folosi regulile CSS.

Ce este CSS?

CSS (*Cascading Style Sheets*) este un limbaj care se foloseşte pentru descrierea semanticii de prezentare a documentului scris într-un limbaj descriptiv (markup language). Cu alte cuvinte, acesta descrie, respectiv ordonează aspectul şi formatarea oricărui element de pe pagină. Acest lucru ne asigură, creând reguli **CSS, să stilizăm şi elemente HTML din alte documente**.

CSS a apărut mult mai târziu după HTML. La început, paginile HTML nu erau stilizate din punct de vedere vizual. Apoi, tag-urile HTML le-am folosit pentru stilizare, însă totul s-a dovedit a fi mult prea complicat şi ineficient. De aceea, a fost conceput limbajul CSS, care permite structurii HTML să afişeze conţinutul, iar CSS-ului să îl stilizeze. La început, mulţi autori au combinat stilizarea HTML şi CSS, însă în zilele noastre, când CSS a ajuns la maturitate, putem **respecta în totalitate regula de separare a conţinutului (HTML) de stilizare (CSS)**.

CSS ne permite să creăm niște reguli (*descriptions*), cu ajutorul cărora vom stiliza anumite elemente. De exemplu, cu CSS putem specifica faptul ca fundalul unei pagini să fie albastru, iar textul roșu cu font Arial. De asemenea, putem seta ca toate paragrafele să aibă diferite tipuri de margini etc.

În CSS, nu scriem tag-uri şi nici nu adăugăm conţinut HTML. CSS se foloseşte doar pentru ordonarea şi stilizarea conţinutului în HTML şi a altor documente.

CSS este separat de limbajul HTML şi de tag-urile sale. Dacă avem o pagină HTML, nu mai trebuie să corectăm codul respectiv pentru a introduce codul CSS. Pe acesta îl introducem fie într-un fişier separat (cu extensia .css), fie în partea head a documentului HTML. Mai târziu, vom explica în detaliu aspecte legate de cum şi unde se scrie CSS.

Cum funcţionează CSS-ul?

Pentru a înțelege cum funcționează limbajul CSS, trebuie să ne imaginăm că în jurul fiecărui element de pe pagină există un cadru (box) invizibil. Utilizând CSS, putem crea reguli, cu ajutorul cărora stilizăm cadrul respectiv, dar şi elementele din el.

Sintaxa **CSS** constă în **regula CSS** **(CSS Rule)**. Regulile CSS sunt alcătuite, în mod obligatoriu, din două părţi: selectorul **(selector)** şi **declaraţia** **(declaration)**.

Selectorul indică elementul (sau, dacă sunt mai multe, le separăm prin virgulă) la care se referă respectiva regulă CSS.

Declaraţia implementează stilizarea pentru elementul la care se referă regula CSS.

Când creăm o regulă CSS, plasăm selectorul, apoi introducem un spaţiu gol şi apoi punem paranteze acolade, după care introducem declaraţia, ca în exemplul de mai jos:

```
h1 { color:red; }
```

Exemplul codului de mai sus reprezintă o regulă CSS simplă, care

defineşte faptul că heading-ul documentului (h1) se afişează cu roşu. În imaginea de mai jos puteţi vedea aspectul înainte şi după aplicarea acestui cod CSS pe tag-ul h1.

Imaginea 10.1. Exemplu de titlu fără cod CSS (sus) şi cu cod CSS (jos)

Declaraţiile sunt, de asemenea, alcătuite din două părţi: proprietate **(property)** şi valoare **(value)**. Scriem proprietatea, apoi două puncte (:), apoi valoarea. În aceeaşi regulă, putem scrie mai multe declaraţii și trebuie doar să le separăm prin punct şi virgulă (;). Haideţi să vedem un exemplu.

Dacă vrem ca paragrafele paginii noastre să aibă fontul Tahoma şi dimensiunea de 12 px, vom scrie următoarele:

```
p {
 font-family:Tahoma;
 font-size:12px;
}
```

În imaginea de mai jos puteţi vedea aspectul înainte şi după aplicarea codului CSS pe tag-ul p.

Imaginea 10.2 Exemplu de titlu fără cod CSS (sus) şi cu cod CSS (jos)

În acest exemplu, am plasat selectorul **p**, ceea ce înseamnă că regula respectivă se referă doar la paragrafele din document (la tag-urile <p> din HTML). Am pus paranteze acolade şi între ele două declaraţii care selectează fontul (font family) şi dimensiunea textului (font size). Observaţi că între proprietate şi valoare în declaraţie stă întotdeauna semnul : (fără spaţii), iar după declaraţie stă întotdeauna semnul ; (care indică sfârşitul valorii).

Trebuie menţionat că noile rânduri şi spaţii nu au un rol important în limbajul CSS, însă trebuie respectată o anumită structură pentru a menține claritatea codului. De obicei, scriem selectorul şi paranteza deschisă pe un singur rând. Apoi, pe următoarele rânduri introducem toate declaraţiile (câte una pe fiecare rând), pentru ca la sfârşit să închidem paranteza pe noul rând, ca în exemplul de mai sus.

Selectorii CSS fac diferența între minuscule şi majuscule, aşadar trebuie să acordăm atenție acestui aspect.

Anumite proprietăţi pot avea mai multe valori (separate prin virgulă) sau mai multe grupuri de valori (separate prin spaţii goale) sau pot conţine un string (text). În acest caz, valoarea se scrie între ghilimele. Putem combina aceste metode de scriere. Haideţi să vedem exemple pentru astfel de cazuri:

```
p {
 font-family:"Times New Roman", Arial;
 margin:10px 0 0 0;
}
```

Înainte de a continua, reţineţi termenii menţionaţi până acum în cadrul acestei lecţii, deoarece vom reveni la ei. Prin urmare, **limbajul CSS este alcătuit din reguli. Fiecare regulă este alcătuită din selector şi declaraţie. Declaraţia este alcătuită din proprietate şi valoare.**

Modalitatea de scriere a unei reguli CSS

Am menţionat că CSS este separat de codul HTML, de aceea există trei opțiuni de plasare a acestuia, din care două se folosesc în mod activ.

CSS-ul extern

După cum îi spune și numele, CSS-ul extern se află în afara documentului HTML, într-un fişier separat. Procesul de salvare este identic celui din HTML, doar că fişierul se salvează în formatul .css.

Cea mai simplă versiune este cea în care avem un fişier HTML, iar lângă el, un fişier CSS. Este suficient să introducem un anumit tag în partea head a fişierului HTML, tag care va conecta fişierul CSS extern. Pe de altă parte, în fişierul CSS scriem doar regulile. În fişierele CSS, nu trebuie să apară nicio parte a limbajului HTML. Spre deosebire de fişierele HTML, fişierele CSS sunt alcătuite doar din reguli CSS.

Tag-ul pentru legarea regulilor CSS în partea head a paginii este **<link>**. La fel ca și alte tag-uri (de exemplu, meta tag-ul), acest tag nu are conţinut, fiind cu autoînchidere. Toate valorile se află în atributele sale de pe tag. În cazul tag-ului <link>, aceste atribute sunt:

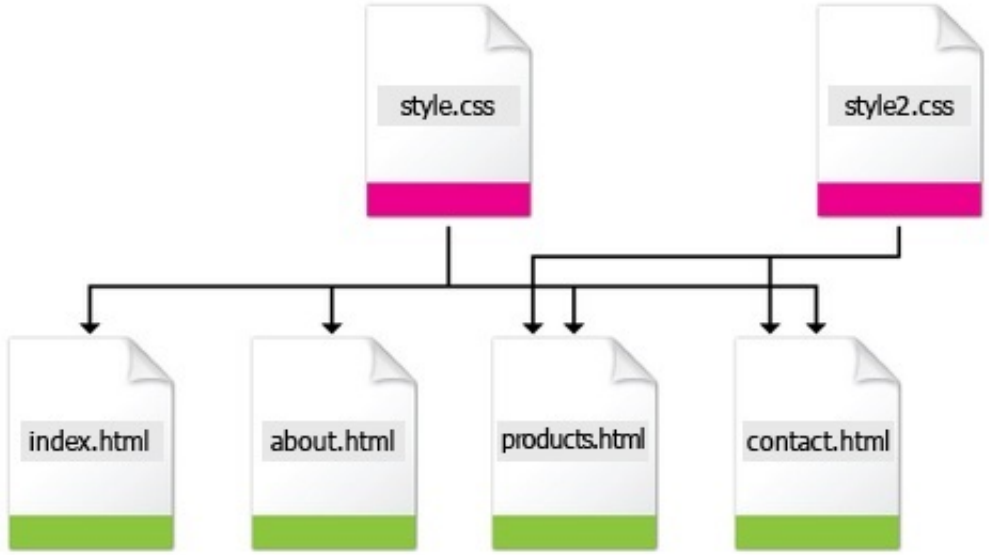
- **href** – Defineşte calea până la fişierul CSS extern. Poate fi relativă sau absolută. Vă recomandăm să setați căi relative şi să vă asigurați ca fişierele site-ului să fie întotdeauna împreună şi în aceleaşi relaţii. Am întâlnit deja acest atribut şi în tag-ul *img*, cu o funcţie similară (calea până la imagine).
- **type** – Defineşte tipul de document la care ne legăm, deoarece tag-ul <link> se poate folosi şi în alte scopuri, deşi, în ultimul timp, se foloseşte doar pentru CSS. Setăm: "*text/css*".
- **rel** – Defineşte relaţia dintre fişierul HTML şi cel legat, adică fişierul extern. Setăm: *„stylesheet".*

Iată un exemplu de fişier CSS extern legat, mai exact un exemplu de început de pagină HTML, care arată astfel:

```
<!DOCTYPE html>

<html>
<head>
  <title>CSS extern</title>
  <link href="css/stilizare.css" type="text/css" rel="stylesheet">
</head>
<body>
...
```

Nu uitați că acest tag se pune **întotdeauna** în cadrul **părţii head** a fişierului HTML şi, practic, este întotdeauna identic, numai calea până la fişier fiind diferită (în acest exemplu, fişierul nostru .css se găseşte în folderul css).



Imaginea 10.3. Pagini HTML şi fişiere CSS externe

Într-un singur document HTML, pot exista mai multe fişiere CSS legate[1]. În acest caz, vom adăuga câte un tag <link> pentru fiecare fişier CSS extern. Vă reamintim că un fişier CSS poate stoca un număr nelimitat de reguli, însă o parte dintre autori separă grupurile de reguli CSS în diferite fişiere, pentru o organizare mai uşoară.

Pe de altă parte, un fişier CSS poate fi apelat, încărcat de mai multe fişiere HTML diferite. Astfel, cu acelaşi fişier putem stiliza aceleaşi elemente de pe diferite pagini.

CSS-ul intern

Spre deosebire de CSS-ul extern, CSS-ul intern se află direct în documentul HTML, mai exact în partea head. În loc să punem un tag <link> ca mai sus, putem adăuga tag-ul <style>, iar în cadrul lui să scriem direct regulile CSS. Iată un exemplu:

```
<!DOCTYPE html>

<html>
```

```
<head>
<title>CSS intern</title>
<style type="text/css">
 p {
  font-family:Tahoma;
  font-size:12px;
}
</style>
</head>
<body>
...
```

În acest mod, practic am încorporat întregul fişier CSS în structura paginii HTML. În afara tag-ului <style>, se află o pagină HTML şi aici sunt valabile regulile ei. Pe de altă parte, în cadrul acestui tag se află regulile CSS şi sunt valabile regulile limbajului CSS.

Imaginea 10.4. CSS-ul intern încorporat în pagina HTML

Notă:

CSS-ul extern *vs.* CSS-ul intern

Se pune întrebarea ce CSS ar trebui să utilizăm, cel extern sau cel intern. Aproape întotdeauna **este mai practic să utilizăm CSS-ul extern**. Iată doar câteva motive:

- Putem folosi aceleaşi reguli CSS pentru mai multe pagini diferite, în loc să le repetăm în mod inutil;
- Deoarece nu există repetiții ale codului, paginile sunt mai ușoare și se încarcă mai repede (CSS se încarcă doar pe prima);
- Putem separa conţinutul de stilizare;
- Putem schimba regulile CSS într-un singur loc, ceea ce va influenţa imediat toate paginile legate;
- Navigăm mai uşor în codul CSS, care este separat de HTML.

Inline CSS

Am menţionat că există trei moduri de a scrie codul CSS. În afară de CSS-ul extern şi intern, există şi modul de scriere **inline**. Acesta se deosebeşte prin faptul că selectorul este omis, iar CSS-ul se scrie direct în tag-ul HTML, la fel ca atributul.

Acest mod de scriere este permis doar în **variantele stricte** ale documentelor și este o relicvă din trecut.

În CSS inline, adăugăm atributul *style* (pe tag-ul HTML pe care vrem să-l stilizăm) și în el scriem între ghilimele proprietăţile şi valorile, ca mai devreme.

Din cauza specificităţii acestei versiuni, regula pe care o introducem se referă doar la un singur element. De aceea, scrierea stilurilor inline nu este practică, iar modificările ulterioare sunt și mai dificile.

Vom analiza următorul exemplu:

```
...
<p>First paragraph</p>
<p style="font-family:Tahoma; font-size:12px;">My paragraph</p>
<p>Third paragraph</p>
...
```

Astfel, am stilizat un singur paragraf de text - al doilea. Dacă am fi vrut să le stilizăm pe toate, ar fi trebuit să le adăugăm și lor atributul style şi aceste proprietăţi şi valori. Este mai uşor să adăugăm CSS-ul extern sau intern şi să stilizăm toate paragrafele cu o singură regulă.

Poate că vă întrebați, în acest moment, cum putem stiliza doar un singur paragraf dacă asta vrem cu adevărat, respectiv să-l evidențiem pe unul dintre ele? În astfel de cazuri sau în cazuri similare, nu vom folosi stilizarea inline, ci vom folosi selectori mai complecşi decât simplul **p**, după cum vom vedea în continuare.

Tipurile de selectori CSS

Există diferite tipuri de selectori, cu ajutorul cărora putem ţinti aproape fiecare element de pe pagină. De noi depinde ce mod vom folosi în funcţie de situaţie. Interesant este faptul că nu există un singur mod adecvat, ci mai multe.

În paragraful de mai sus, vorbind despre selectori, am folosit cuvântul **a ţinti**. Cuvântul face parte din jargon, respectiv din vocabularul experților CSS. Deşi ar fi mai corect să spunem: „**această regulă stilizează elementul determinat de selectorul x**”, spunem mai degrabă: „**această regulă ţinteşte elementul x**”. Acest mod de exprimare a devenit o regulă nescrisă, deoarece pleacă de la faptul că selectorii separă doar anumite elemente dintr-o mulţime de alte elemente şi le ţinteşte doar pe acelea.

Analizaţi tabelul de mai jos, în care sunt prezentate tipurile de selectori.

Tipul de selector	Exemplu	Explicaţie

General (standard)	Body, p, h2	Ţinteşte <body>, precum şi toate elementele <p> și <h2> de pe pagină.
ID†	#exemplu	Ţinteşte elementul cu valoarea ID marcată.
CLASS‡	.exemplu	Ţinteşte elementul cu valoarea CLASS marcată.

Tabelul 10.1. Tipurile de selectori

†Notă: *Valoarea ID se pune în atributul ID al tag-ului HTML doar după nume, în timp ce în regula CSS se scrie cu prefixul #.*

‡Notă: *Valoarea CLASS se adaugă în atributul CLASS al tag-ului HTML doar după nume, în timp ce în regula CSS se scrie cu un prefix.*

Notă suplimentară pentru ID şi CLASS

Selectorii ID şi CLASS se pot combina cu alţi selectori. Facem aceasta scriindu-i împreună. De exemplu:

```
h1#exemplu {...}
```

ar ţinti doar tag-ul <h1>, care posedă valoarea ID *example*. Acelaşi lucru este valabil şi pentru CLASS, numai că în loc de # se scrie . (punct).

Notă:

În plus față de tipurile de bază ale selectorilor, există un selector universal care se caracterizează prin caracterul *. Acest selector trage toate elementele de pe pagină.

Tipuri de selectori avansați:

Tipul de selector	Exemplu	Explicaţie
Moștenitor (descendent)	h1 a	Ţinteşte toate elementele <a> aflate în cadrul <h1>, chiar dacă între ele există elemente.
Copil (child)	h1>a	Ţinteşte toate elementele <a> aflate în cadrul <h1>, însă doar acele elemente care se află direct în el (fără elemente între).
Rudă (sibling)	h1~p	Ţinteşte toate elementele <p> care vin după elementul <h1>.
Rudă directă (direct sibling)	h1+p	Ţinteşte doar un element <p> care se află imediat după <h1>.

Tabelul 10.2 Selectorii avansaţi

Pentru cei care vor mai multe detalii, la următorul link găsiţi o listă completă de selectori pe care îi puteţi folosi în standardul HTML5 şi CSS3:

https://www.w3.org/TR/selectors-3/

Selectorii complecşi

Haideţi să vedem ce reguli sunt valabile pentru selectori şi, în general, pentru regulile CSS.

Regula ordonării stilurilor CSS

Am văzut că CSS se poate scrie în trei moduri (ca fişier separat, ca secţiune head a documentului HTML sau ca inline al tag-ului), astfel încât se poate ajunge la suprapunerea aceloraşi valori CSS. De asemenea, în cadrul fişierelor CSS sau al secțiunii CSS din cadrul paginii se poate ajunge la suprapunere. În astfel de situaţii, există anumite reguli care se aplică în mod automat şi determină care regulă CSS va fi aplicată.

În cazul în care regulile CSS se suprapun, browser-ul va aplica regula de **proximitate** (engl. *proximity rule*), care specifică aplicarea regulii CSS care este **cea mai apropiată** de elementul-ţintă.

Practic, aceasta înseamnă că, în caz că există mai multe stilizări diferite pentru acelaşi element, browser-ul va folosi cel mai apropiat element de elementul respectiv, cu condiţia ca selectorul să fie identic. În caz că selectorii se deosebesc (dar ţintesc acelaşi element), este important care dintre aceştia este mai precis, iar apoi este importantă şi ordinea.

Haideţi să vedem în exemplu! Dacă în fişierul CSS am fi inserat următoarele:

```
p {
  color:#Foo; /*red color*/
}
```

în partea head a paginii am fi inserat:

```
p {
  color: #3Fo; /*green color*/
}
```

iar în tag-ul inline, culoarea albastră (blue color):

```
<p style="color:#00F;">Acesta este un text.</p>
```

este logic că textul nu poate fi negru (cum este în mod implicit), roşu, verde şi albastru în același timp. Browser-ul va seta culoarea albastră, deoarece această regulă CSS este **cea mai aproape** de element.

Este important să menţionăm și că, dacă avem două sau mai multe reguli CSS care se aplică la acelaşi element-ţintă, iar proprietăţile nu sunt în conflict, vor fi aplicate toate proprietăţile.

Să vedem câteva exemple în acest sens.

Dacă avem regula CSS:

```
p {
  color:#Foo;
}
```

dar şi o altă regulă:

```
p {
  font-family:Arial;
}
```

deşi se referă la acelaşi element, paragraful va fi formatat în culoarea roşie şi cu fontul Arial, deoarece acestea se completează.

Pe de altă parte, dacă avem:

```
p {
```

```
  color:#Foo;
}
```

dar şi

```
p {
  font-family:Arial;
  color:#3Fo;
}
```

atunci se iau **toate proprietăţile care nu se contrazic, plus una dintre cele aflate în conflict**. Paragraful va fi formatat cu fontul Arial, însă formatarea culorii (roşu sau verde) va depinde de regulile contradicției menţionate mai devreme.

Elementele-ţintă prin care atribuim proprietăţi pot fi mai complexe şi mai specifice, în funcţie de elementele aflate deasupra lor.

De exemplu:

```
p {
 color:red;
}
```

a stilizat toate elementele p, în timp ce următorul cod:

```
.first {
 color:blue;
}
```

a stilizat doar elementele p care au clasa (class) *first*. Dacă, de exemplu, avem 10 paragrafe pe pagină, toate vor obţine culoarea din primul selector (culoarea roşie), în timp ce fiecare paragraf, care are şi clasa *first*, va obţine culoarea albastră. Apoi, în acest caz, regula CSS cu clasă nu trebuie să se găsească după selectorul general pentru a funcţiona, deoarece clasa este întotdeauna mai importantă decât selectorul general. Acelaşi lucru se întâmplă şi cu valorile ID, care sunt întotdeauna mai importante decât clasele.

Haideţi să analizăm şi această situaţie. De exemplu:

```
h1 {
 color:#F90;
}
```

ar fi stilizat toate elementele h1, în timp ce următorul cod:

```
div h1 {
 color:#F90;
}
```

ar fi stilizat doar elementele h1 care se află în cadrul elementului div. Dacă, de exemplu, punem h1 direct în body, acesta nu este influenţat şi nici stilizat de această regulă, deoarece nu se află în cadrul niciunui div.

În continuare, putem defini şi mai precis raportul elementului pe care îl ţintim cu stilizarea. În exemplul anterior, am avut:

```
div h1 {
 color:#F90;
}
```

iar dacă scriem:

```
div.myClass h1 {
 color:#F90;
}
```

vom stiliza toate titlurile h1, care se află în elementul div cu clasa myClass.

Pe scurt, dacă introducem un spaţiu, ţintim elementul de dedesubt; dacă nu există niciun spaţiu, atunci ţintim elementul de la acelaşi nivel (elementul care îndeplineşte ambele condiţii).

Până acum aţi primit multe informaţii noi, cu care nu v-aţi mai întâlnit. După ce veți parcurge exemplele şi veți exersa puţin, reveniţi la această lecţie pentru a fixa detaliile, deoarece ele reprezintă baza limbajului CSS.

Proprietăţile disponibile în CSS

După cum ştim deja, în afară de selector, care face parte din regula CSS, trebuie să cunoaștem şi ce proprietăţi şi valori putem utiliza.

Normal că nu trebuie să învăţați pe de rost toate proprietăţile posibile, deoarece, pe de o parte, există prea multe, iar, pe de altă parte, nici nu e necesar. Oricum, vă stau întotdeauna la dispoziție și pe internet şi/sau în bibliografie[2]. Important este să știți cum şi unde să le aplicaţi.

Cele mai des folosite proprietăţi vor fi abordate pe parcusul cursului.

Afişarea în diferite browsere

Pe oricine întrebați, un dezvoltator web cu experienţă, un designer sau, în general, orice persoană cu experienţă în HTML şi CSS, dacă CSS se comportă identic în diferite browsere, răspunsul pe care-l veţi primi va fi unul negativ. Ca şi HTML, CSS este un document text care trebuie parsat şi afişat, iar toate acestea depind de multe detalii, în special de browser. A devenit deja celebră povestea browserului Internet Explorer şi (ne)respectarea standardelor general-acceptate. În continuare, Internet Explorer rămâne în urma celorlalte browsere moderne (Chrome, Opera, Firefox). Odată cu apariţia programului Microsoft Edge, s-au rezolvat majoritatea problemelor, însă, în continuare există diferenţe. Ceea ce trebuie noi, ca programatori web să facem, este să testăm site-urile noastre în diferite browsere, pentru a fi siguri că totul se afişează în mod normal. Nu este suficient să creăm un site în care totul se afişează perfect, de exemplu în browserul Chrome şi să ignorăm că totul arată greşit în Internet Explorer. Poate nu folosiţi Internet Explorer, dar mulți alții îl folosesc. Astăzi, în momentul scrierii acestui curs, tot mai mulţi autori au suspendat complet optimizarea pentru Internet Explorer, dar acest lucru ţine de decizia proprie şi de piaţa pentru care se proiectează site-ul.

Înainte de a începe să lucraţi la un site, trebuie să determinaţi (sau să ajungeţi la un acord cu cel care a comandat site-ul) ce browsere va susţine pentru a şti pe ce funcţii vă puteţi baza, deoarece unele versiuni mai vechi pur și simplu nu suportă funcţii mai noi.

Dacă vreţi să folosiţi o anumită proprietate, dar nu sunteţi siguri ce browsere o susţin, puteţi folosi pagini precum: https://caniuse.com/, pe care puteţi verifica ce browsere, începând de la care versiune, susţin proprietatea căutată. Desigur, mai târziu, în timpul cursului, vom lucra în detaliu cu majoritatea acestor proprietăţi, aşadar, abia atunci sau mai târziu, în timpul lucrului, veţi avea nevoie să folosiţi instrumente similare.

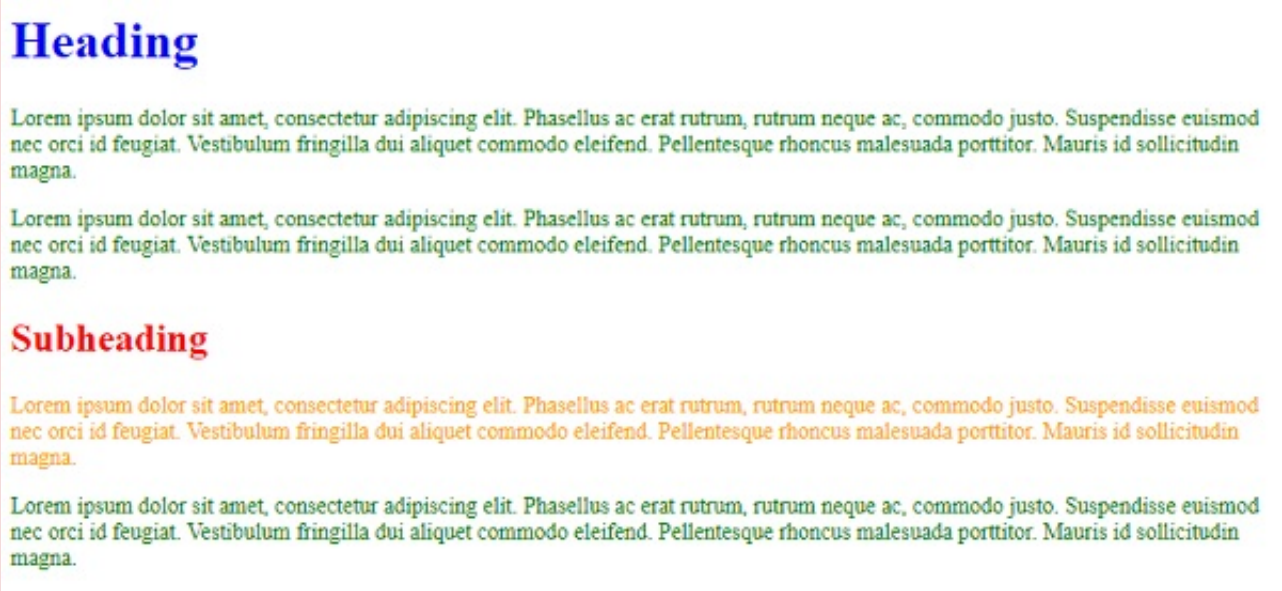
Întrebarea care rămâne este legată de modul în care putem soluţiona situaţia în care într-un browser ceva poate fi afișat normal, în timp ce

în altul nu. Vom începe verificarea, parcurgând următoarele etape.

- **Există o eroare în codul nostru HTML şi/sau CSS?** Dacă browser-ul găsește un tag deschis, tag-uri suprapuse, o regulă CSS scrisă incorect etc., el va încerca să presupună ce ar trebui să se afle aici şi atunci apar problemele. În majoritatea cazurilor, problema se află aici, iar nu în browser.
- **Browser-ul în care apare problema suportă ceea ce vrem să realizăm?** Am menționat deja acest detaliu în paragraful precedent. Nu toate browserele suportă totul. De exemplu, versiunea 6 de Internet Explorer nu suportă fişierele PNG transparente. În toate celelalte browsere, un astfel de PNG va fi afişat normal, dar excepţie face doar browserul de mai sus. Putem să înlocuim imaginile PNG cu altele diferite sau să ignorăm acest browser, însă aceasta este deja o chestiune legată de planificare.
- Dacă am trecut cu bine de cele două etape mai sus-menționate și problema noastră nu este legată de ele, atunci avem de-a face cu un CSS ***browser bug*** (o problemă de browser), respectiv ***browser quirk***. În acest caz, cel mai bine ar fi să căutăm un răspuns sau o soluţie pe internet, deoarece este probabil ca cineva să se fi confruntat înainte cu aceeaşi problemă sau cu una similară şi să fi împărtăşit soluţia cu alte persoane de pe internet.

Exerciţiu:

În cadrul acestui exerciţiu trebuie să scrieţi un cod CSS care va stiliza o pagină HTML aşa cum este prezentat în imaginea de mai jos:



Imaginea 10.5. Paginile HTML şi fişierele CSS externe

Notă:

Codul CSS se poate scrie în funcție de preferințe, fie ca un fişier inline (în tag-urile propriu-zise), ca fişier CSS intern (în tag-ul script HTML) sau ca fişier CSS extern. Ca întotdeauna, recomandăm să folosiţi fişierul CSS extern pe care îl veţi lega cu fişierul HTML.

Prezentarea formularului pe pagină depinde de browserul folosit şi de dimensiunea ferestrei, aşadar, soluţia voastră nu trebuie să fie identică.

Fişierul HTML propriu-zis cu text se poate prelua de la următorul link. Voi trebuie doar să scrieţi codul CSS pentru acest exerciţiu.

Dacă întâmpinaţi dificultăţi în rezolvarea acestui exerciţiu sau vreţi să verificaţi codul CSS, soluţia exerciţiului se poate prelua de la următorul link.

1. *Versiunile mai vechi de internet pot încărca până la 20 de fişiere CSS diferite într-un singur document. În noile versiuni, nu există limite*.

2. *Mai multe despre proprietăţile şi valorile CSS disponibile puteţi citi la următoarele adrese web*:
http://reference.sitepoint.com/css/propertyref sau
https://developer.mozilla.org/en-US/docs/CSS/CSS_Reference.
Bibliografia scrisă şi recomandată se află în cadrul cursului.