

## 2. FUNDAMENTE TEORETICE ALE PAGINILOR WEB – LIMBAJUL HTML

### 2.1. Scurt istoric al apariției Internetului și funcționarea sa. Rețeaua WWW.

#### Conceptul de Hipertext.

Istoria Internetului începe cu anul 1968, când guvernul S.U.A. intenționa să interconecteze universitățile, departamentele militare și de apărare ale țării, astfel încât ele să coopereze în cadrul unor proiecte de cercetare comune. Astfel, s-a format o agenție numită *Advanced Research Projects Agency (ARPA)*. Una din cheile proiectului punea în discuție faptul că, stocarea tuturor informațiilor pe un singur calculator nu ar fi fost deloc sigură, fie din cauză că acesta ar putea fi țintă vulnerabilă a unui eventual atac, fie pur și simplu din cauză că acestea ar putea fi pierdute în cazul unei defecțiuni tehnice majore. O metodă de a face față unei asemenea situații ar fi de a copia și distribui informațiile pe mai multe calculatoare, în întreaga țară, folosind o rețea.

În 1975, câteva dintre limbajele sau protocoalele pe care calculatoarele le foloseau pentru a comunica între ele s-au standardizat. Majoritatea universităților importante și a departamentelor de apărare din S.U.A. s-au legat împreună într-o rețea numită *DARPANET*, toate calculatoarele folosind același protocol pe care astăzi îl cunoaștem sub denumirea de TCP/IP. Rețeaua, cu timpul, a fost înlocuită de mai multe rețele, care astăzi împânzesc globul pământesc.

Începând cu anul 1980, mai multe colegii și universități au fost conectate la Internet. Acest lucru a permis universităților să-și împartă informații despre cercetările lor, programe și știri recente. În anii '90 Internetul s-a deschis și în scopuri comerciale. În curând, multe alte căi de utilizare a informațiilor transmise prin intermediul acestei gigantice rețele au fost dezvoltate.

În prezent, este posibil să folosești Internetul pentru a trimite scrisori electronice pe întregul glob în doar câteva secunde. Poți căuta informații despre orice subiect dorești. Expresia „World Wide Web” (*WWW*) definește o colecție de documente care se întinde în câteva sute de milioane de calculatoare.

Principiul de bază al funcționării Internetului constă în faptul că două sau mai multe calculatoare pot comunica între ele. Pentru ca acest lucru să fie posibil este necesar să existe un „protocol”, adică un ansamblu de norme care trebuie respectate de calculatoare (deci de programele care rulează pe ele) pentru ca schimbul de date să poată avea loc.

Normele se referă la:

- găsirea calculatorului destinatar al transferului de date;
- transmiterea efectivă a datelor;
- modalități prin care expeditorul comunică faptul că au fost transmise toate datele, iar destinatarul comunică faptul că le-a recepționat;

- compresia datelor: prin aplicarea anumitor algoritmi matematici, datele care urmează să fie expediate sunt prelucrate de așa natură, încât să fie memorate prin utilizarea unui spațiu cât mai mic de memorie. Prin urmare, transmiterea lor durează mai puțin. Invers, la destinație sunt decompresate prin utilizarea acelorași algoritmi matematici;

- identificarea erorilor care pot interveni în transmiterea datelor: și aici există mai mulți algoritmi care permit identificarea și corectarea erorilor.

Standardul care s-a impus în ceea ce privește Internetul, constă în protocolul TCP/IP. Numele este de fapt, numele comun al unei familii de protocoale utilizate pentru transferul datelor în rețea. Orice calculator conectat la Internet are o adresă, numită adresă IP (Internet Protocol Address). O adresă IP este alcătuită din 4 numere între 0 și 255, prin urmare o astfel de adresă ocupă 4 octeți. Cum transmiterea datelor la un moment dat se face între două calculatoare, datele se transmit de la o adresă IP la alta.

*Protocolul IP* (Internet Protocol) reglementează transmiterea datelor de la o adresă IP la alta. Datele sunt transmise divizate în pachete. În acest fel, se preîntâmpină monopolizarea transmisiei în rețea doar de către un singur utilizator.

*Protocolul TCP* (Transmission Control Protocol): de la plecare, un program TCP împarte informația de transmis în mai multe pachete IP. Acestea sunt transmise la destinație prin intermediul rețelei. O dată ajunse la destinație, un alt program TCP assemblează și aranjează în ordinea corectă pachetele IP de date primite. Firește, din cauza unor probleme hardware, unele pachete se pot pierde pe drum. Protocolul TCP se ocupă și de acest lucru. Astfel, când împachetează datele într-un plic „IP”, protocolul TCP al expeditorului adaugă și un număr (numit sumă de control) care va permite destinatarului să se asigure de faptul că datele primite sunt corecte. Receptorul recalculează suma de control și o compară cu cea transmisă de emițător. Dacă ele nu sunt identice, înseamnă că a apărut o eroare în timpul transmisiei, motiv pentru care protocolul TCP anulează acel pachet, cerând retransmiterea sa.

**Bazele World Wide Web (WWW)** au fost puse în 1989 la Centrul European de Cercetări Nucleare (CERN) în Geneva (Elveția). Propunerea inițială de creare a unei colecții de documente având legături între ele a fost făcută de Tim Berners-Lee în martie 1989. Această propunere a apărut în urma problemelor de comunicare pe care le întâmpinau echipele de cercetători ce foloseau centrul, chiar și folosind poșta electronică.

Primul server web folosit de Tim Berners-Lee a apărut nu mult înainte de decembrie 1991, când s-a făcut prima lui demonstrație publică. Studiul a fost continuat prin apariția primei aplicații grafice Mosaic, în februarie 1993, realizată de cercetătorul Marc Andreessen de la centrul universitar National Center for Supercomputing Applications (NCSA) din orașul Urbana-Champaign din statul federal Illinois, SUA. Ulterior WWW-ul a evoluat până la ceea ce este astăzi, un serviciu integrativ și multimedial, având ca suport fizic Internetul.

Practic, WWW este un sistem de documente și informații de tip hipertext legate ele între ele, care pot fi accesate prin rețeaua mondială de Internet. Documentele, care rezidă în diferite locații pe diverse calculatoare-server, pot fi regăsite cu ajutorul unei adrese unice. Hipertextul este prelucrat cu un ajutorul unui program de navigare în web numit *browser* care descarcă paginile web de pe un server web și le afișează pe un terminal.

**Prin conceptul de hipertext** se înțelege o formă de document electronic, o metodă de organizare a informațiilor în care datele sunt memorate într-o rețea de noduri și legături, putând fi accesată prin intermediul programelor de navigare interactivă, și manipulată de un editor structural. Conceptul de bază în definirea hipertextului este "legătura" (link-ul), fie în cadrul aceluiași document, fie către alt document. Legătura de tip link permite organizarea neliniară a informațiilor. Un sistem hipertext permite autorului său să creeze așa-numite "noduri", să le lege între ele, iar unui cititor navigarea de la un nod la altul. Astfel un nod reprezintă un concept putând conține orice fel de informație: text, grafică, imagini, animații, sunete, etc. Nodul sursă al unei legături se numește "referință" iar cel destinație "referent" sau ancoră, punctele de legătură din respectivele noduri fiind marcate. Activarea marcajelor unei legături duce la vizualizarea nodurilor. Asocierea cu unele elemente mediale a dus la extinderea noțiunii de hipertext către "hipermedii".

## **2.2. Despre website-uri.**

Noțiunea de website (sau pur și simplu site, ori „site web”) desemnează o grupă de pagini web multimediale (conținând texte, imagini fixe, imagini mișcătoare și chiar sunete), accesibile în Internet în principiu oricui, de obicei pe o temă anume, și care sunt conectate între ele prin așa-numite hyperlinkuri. Diversele situri web pot fi oferite de către o companie, un proiect, o rețea de utilizatori, o persoană particulară, o administrație publică și multe altele.

Pentru crearea paginilor web s-a impus limbajul HTML (**H**yper**T**ext **M**arkup **L**anguage) – un limbaj de marcare, al cărui scop constă în prezentarea într-un anumit format a informațiilor: paragrafe, tabele, fonturi, culori, ș.a.m.d.

Calculatorul pe care se găsește site-ul se numește „server”, iar calculatoarele care accesează conținutul site-ului se numesc „client”.

Orice calculator client trebuie să dispună de un program specializat, numit „browser”, cu ajutorul căruia să se poată interpreta și deci vizualiza fișierele HTML.

Pe server trebuie să se găsească un program care răspunde cererilor browser-ului aflat pe calculatorul client. Cererea efectuată de către browser și răspunsul server-ului se fac prin respectarea unui anumit protocol. Acest protocol se numește HTTP (**H**yper**T**ext **T**ransfer **P**rotocol).

### **2.3. HTML standard – limbaj descriptiv al unei pagini WEB.**

HTML este un limbaj de marcare orientat către prezentarea documentelor text pe o singura pagină.

Utilizând un software de redare specializat, numit agent utilizator HTML (cel mai bun exemplu de astfel de software fiind browserul web) HTML furnizează mijloacele prin care conținutul unui document poate fi adnotat cu diverse tipuri de metadata și indicații de redare. Indicațiile de redare pot varia de la decorațiuni minore ale textului (cum ar fi specificarea faptului că un anumit cuvânt trebuie subliniat sau că o imagine trebuie introdusă) până la scripturi sofisticate, hărți de imagini și formulare. Metadatale pot include informații despre titlul și autorul documentului, informații structurale despre cum este împărțit documentul în diferite segmente, paragrafe, liste, titluri etc. și informații cruciale care permit ca documentul să poată fi legat de alte documente pentru a forma astfel hiperlink-uri.

HTML este un format text proiectat pentru a putea fi citit și editat de oameni utilizând un editor de text simplu. Totuși scrierea și modificarea paginilor în acest fel solicită cunoștințe solide de HTML și este consumatoare de timp. Editoarele grafice cum ar fi Macromedia Dreamweaver sau Microsoft FrontPage permit ca paginile web să fie tratate asemănător cu documentele Word, dar cu observația că aceste programe generează un cod HTML care este de multe ori de proastă calitate.

HTML se poate genera direct utilizând tehnologii de codare din partea serverului cum ar fi PHP, JSP sau ASP.

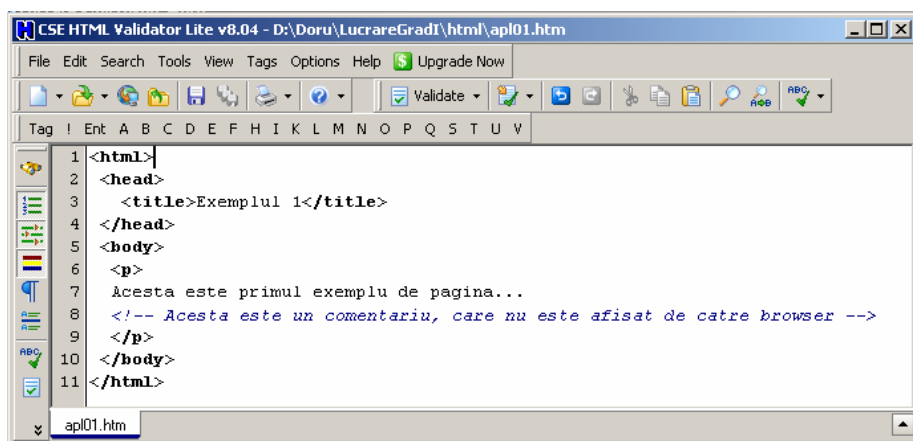
### 2.3.1 Scrierea de cod HTML. Editoare specializate și validatoare HTML.

Crearea unui fișier HTML este foarte simplă, putând fi făcută cu ajutorul oricărui editor de text. Totuși, pentru a avea un control ridicat asupra corectitudinii codului scris, este recomandat să utilizăm un editor specializat, care să pună în evidență diversele elemente de marcare (TAG-uri, numite și „elemente” sau „etichete”) sau, mai mult, să poată verifica și detecta erorile.

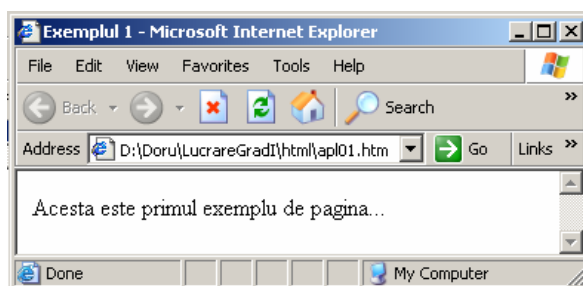
Din categoria editoarelor care pun în evidență diferitele elemente face parte editorul *Notepad++*, iar din categoria validatoarelor face parte *CSE HTML Validator Lite*, ambele fiind gratuite și putând fi descărcate de pe Internet.

### 2.3.2. Structura de bază a unei pagini.

Structura de bază a unei pagini HTML este următoarea:



Iată și modul în care pagina de mai sus este vizualizată în Internet Explorer:



Din analiza exemplului observăm că:

- O pagină începe cu tag-ul **<HTML>** și se termină cu tag-ul **</HTML>**;
- O pagină conține un antet (HEAD) și corpul propriu-zis (BODY);
- Antetul este cuprins între etichetele **<HEAD>** și **</HEAD>**;
- Corpul este cuprins între etichetele **<BODY>** și **</BODY>**;

- Opțional, antetul poate conține titlul paginii, cuprins între tag-urile **<TITLE>** și **</TITLE>**. Titlul apare pe bara de titlu a ferestrei afișate în browser.
- Corpul poate conține texte și/sau imagini. În exemplu, pagina conține textul „Acesta este primul exemplu de pagina...”
- Comentariile, care nu sunt afișate de către browser, pot fi scrise între tag-urile **<!--** și **-->**.
- Numele tag-urilor nu sunt case sensitive, deci pot fi scrise atât cu litere mici cât și cu litere mari. În continuare, pentru a le pune în evidență, le vom scrie cu litere mari.

### 2.3.3. Paragrafe. Atribute ale unui tag.

În general, textele conținute de o pagină se pot găsi în mai multe paragrafe. Un paragraf se introduce între tag-urile **<P>** ... **</P>**.

La afișare, două paragrafe consecutive vor fi separate printr-o linie goală.

Tag-ul **</P>** poate lipsi; un nou paragraf poate fi detectat prin tag-ul **<P>**.

În cadrul unui fișier HTML, Enter-ul nu are nici un efect. De asemenea, dacă două cuvinte ale unui paragraf sunt separate prin mai multe spații sau alte caractere albe (enter-uri, tab-uri), browser-ul afișează doar un singur spațiu.

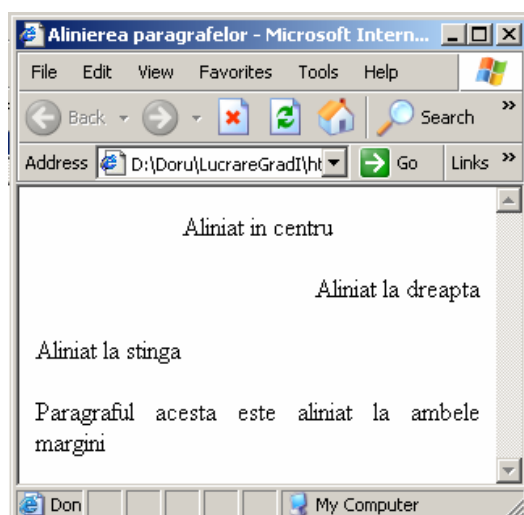
Majoritatea tag-urilor li se pot specifica atribute. Acestea determină comportamentul mai amănunțit al elementului respectiv.

Un atribut se specifică înainte de închiderea parantezei unghiulare a tag-ului (**>**) prin **nume\_atribut="valoare"**.

În cazul paragrafului, atributul **align** controlează alinierea textului din cadrul paragrafului. Dacă acest atribut nu este prezent, alinierea este făcută în mod implicit la stânga. Acest atribut poate lua una dintre valorile **center**, **left**, **right**, **justify**, ca în exemplul de mai jos:

```
<HTML>
<HEAD>
  <TITLE>Alinierea paragrafelor</TITLE>
</HEAD>
<BODY>
  <P align="center">Aliniat in centru</P>
  <P align="right">Aliniat la dreapta</P>
  <P align="left">Aliniat la stinga</P>
  <P align="justify">Paragraful acesta este aliniat la ambele margini</P>
</BODY>
</HTML>
```

Iată pagina al cărei cod tocmai a fost prezentat, vizualizată în Internet Explorer:



#### 2.3.4. Elemente care permit formatarea textului.

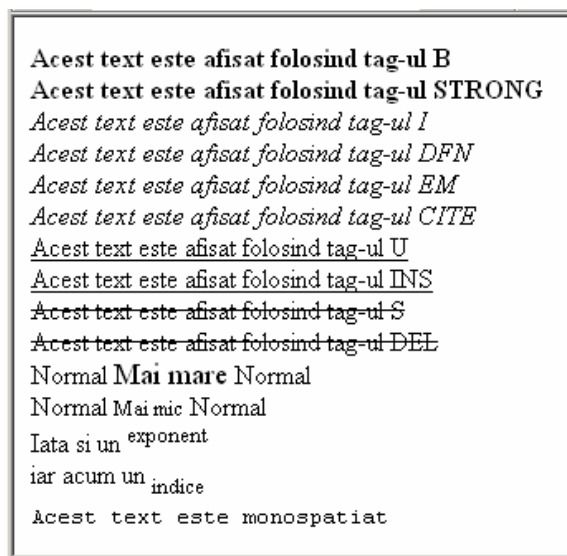
- **<BR>** : Are ca efect forțarea afișării a ceea ce urmează pe rândul următor. Acest tag nu creează un nou paragraf (să ne reamintim că între două paragrafe este automat lăsată o linie vidă)
- **<B>...</B>** : Are rolul de a afișa bold (îngroșat) textul cuprins între cele două tag-uri ale sale. Un tag sinonim al lui **<B>** este: **<STRONG>...</STRONG>**
- **<I>...</I>** : Are rolul de a afișa italic (înclinat) textul cuprins între cele două tag-uri ale sale. Tag-uri sinonime ale lui **<I>** sunt: **<EM>...</EM>**, **<DFN>...</DFN>**, **<CITE>...</CITE>**.
- **<U>...</U>** : Are rolul de a afișa subliniat textul cuprins între cele două tag-uri ale sale. Un tag sinonim al lui **<U>** este: **<INS>...</INS>**
- **<S>...</S>** : Are rolul de a afișa tăiat (cu o linie orizontală) textul cuprins între cele două tag-uri ale sale. Un tag sinonim al lui **<S>** este: **<DEL>...</DEL>**
- **<BIG>...</BIG>** : Are rolul de a afișa textul cuprins între cele două tag-uri ale sale mai mare decât textul în care este cuprins.
- **<SMALL>...</SMALL>** : Are rolul de a afișa textul cuprins între cele două tag-uri ale sale mai mic decât textul în care este cuprins.
- **<SUP>...</SUP>** : Are rolul de a afișa textul cuprins între cele două tag-uri ale sale mai sus (ca o putere)
- **<SUB>...</SUB>** : Are rolul de a afișa textul cuprins între cele două tag-uri ale sale mai sus (ca un indice)

- **<TT>...</TT>** : Are rolul de a afișa textul cuprins între cele două tag-uri ale sale mai sus monospațiat (toate caracterele ocupă aceeași lungime – practic, se folosește fontul Courier New)

În cod-ul HTML de mai jos găsiți toate aceste tag-uri exemplificate:

```
<HTML>
<HEAD>
  <TITLE>Formatarea textului</TITLE>
</HEAD>
<BODY>
  <P>
    <B>Acest text este afisat folosind tag-ul B</B> <BR>
    <STRONG>Acest text este afisat folosind tag-ul STRONG</STRONG> <BR>
    <I>Acest text este afisat folosind tag-ul I</I> <BR>
    <DFN>Acest text este afisat folosind tag-ul DFN</DFN> <BR>
    <EM>Acest text este afisat folosind tag-ul EM</EM> <BR>
    <U>Acest text este afisat folosind tag-ul U</U> <BR>
    <INS>Acest text este afisat folosind tag-ul INS</INS> <BR>
    <S>Acest text este afisat folosind tag-ul S</S> <BR>
    <DEL>Acest text este afisat folosind tag-ul DEL</DEL> <BR>
    Normal <BIG>Mai mare</BIG> Normal <BR>
    Normal <SMALL>Mai mic</SMALL> Normal <BR>
    Iata si un <SUP>exponent</SUP> <BR>
    iar acum un <SUB>indice</SUB> <BR>
    <TT>Acest text este monospațiat</TT>
  </P>
</BODY>
</HTML>
```

Acest cod vizualizat în browser arată în felul următor:



- Pentru scrierea titlurilor se utilizează tag-urile **<H1>...<H1>**, **<H2>...<H2>**, . . . , **<H6>...<H6>**. Practic, în funcție de numărul de după H mărimea fontului diferă (**<H1>** utilizează fontul de dimensiune maximă, **<H6>** fontul de dimensiune minimă) iar textul care apare între tag-uri este scris îngroșat (bold).

- Pentru stabilirea font-ului se folosește tag-ul **<FONT>...<FONT>**. Atributele acestuia sunt:

- **face** indică numele font-ului
- **size** indică mărimea (trebuie să fie un număr cuprins între 1 și 7. Implicit este 3)



- **color** permite specificarea culorii. Aceasta se specifică fie prin intermediul constantelor predefinite ale HTML-ului (numele englezesc al culorii) fie prin componentele sale de Roșu, Verde și Albastru exprimate în hexazecimal, de forma #RRGGBB (vom detalia aceste constante de culoare ceva mai încolo).

Iată un exemplu de utilizare al lor:

```
<HTML>
<HEAD>
  <TITLE>Exemplificare titluri si font</TITLE>
</HEAD>
<BODY>
  <P>
    <H1>Acesta este un titlu de tip H1</H1>
    <H2>Acesta este un titlu de tip H2</H2>
    <H3>Iar acesta este un titlu de tip H3</H3>
    <FONT face="arial" color="blue" size="4">
      Acest text este scris cu fontul Arial, albastru, dimensiune 4
    </FONT><BR>
    Iar acest text este scris normal<BR>
  </P>
  <P>
    Iata si culorile cucubeului, scrise cu font-ul Comic Sans MS,
    bold, dimensiune 7:<br>
    <B>
      <FONT face="Comic Sans MS" size="7" color="red">R</FONT>
      <FONT face="Comic Sans MS" size="7" color="orange">O</FONT>
      <FONT face="Comic Sans MS" size="7" color="yellow">G</FONT>
      <FONT face="Comic Sans MS" size="7" color="green">V</FONT>
      <FONT face="Comic Sans MS" size="7" color="blue">A</FONT>
      <FONT face="Comic Sans MS" size="7" color="darkblue">I</FONT>
      <FONT face="Comic Sans MS" size="7" color="magenta">V</FONT>
    </B>
  </P>
</BODY>
</HTML>
```

Vizualizat în browser:



Așa cum am văzut, dacă în cadrul unui text din cadrul documentului HTML apare un grup de mai multe spații, în browser va fi afișat doar unul singur. Dacă dorim forțarea afișării unui spațiu, se folosește identificatorul special **&nbsp;** (ultimul caracter, ”;”, face parte din identificator)

### 2.3.5. Liste.

Acestea permit ca anumite enunțuri (texte, elemente) să fie numerotate sau marcate într-un anumit fel. O astfel de organizare poartă numele de liste.

În HTML distingem 3 feluri de liste:

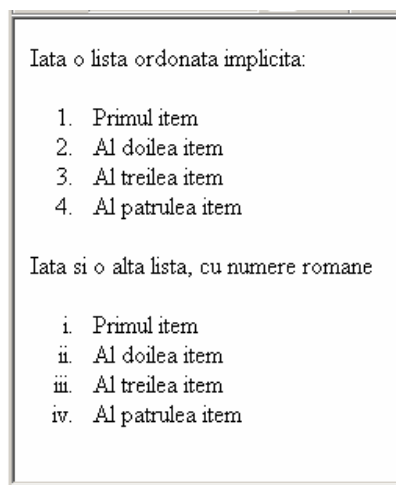
- **Liste ordonate (Ordered Lists):** sunt liste în care elementele sunt numerotate.

Inserarea lor în cadrul documentului HTML se face prin tag-urile `<OL>...</OL>`, elementele (itemii) lor fiind introduse între aceste două tag-uri prin `<LI>...</LI>` (tag-ul de sfârșit nefiind obligatoriu). Implicit, numerotarea se face cu numere arabe (1, 2, 3, ...). Ea poate fi modificată prin folosirea atributului `type` în cadrul tag-ului `OL`. Acesta poate lua una dintre valorile:

- **a** : numerotarea se va face cu litere mici (a, b, c, ...)
- **A** : numerotarea se va face cu litere mari (A, B, C, ...)
- **i** : numerotarea se va face cu numere romane mici (i, ii, iii, iv ...)
- **I** : numerotarea se va face cu numere romane mari (I, II, III, IV, ...)
- **1** : (implicit) numerotarea se va face cu numere arabe obișnuite (1, 2, 3, ...)

Iată un exemplu de cod și vizualizarea sa în browser

```
<HTML>
<HEAD>
  <TITLE>Liste</TITLE>
</HEAD>
<BODY>
  <P>
    Iata o lista ordonata implicita:
  <OL>
    <LI>Primul item</LI>
    <LI>Al doilea item</LI>
    <LI>Al treilea item</LI>
    <LI>Al patrulea item</LI>
  </OL>
  Iata si o alta lista, cu numere romane
  <OL type="i">
    <LI>Primul item</LI>
    <LI>Al doilea item</LI>
    <LI>Al treilea item</LI>
    <LI>Al patrulea item</LI>
  </OL>
  </P>
</BODY>
</HTML>
```



• **Liste neordonate (Unordered Lists):** sunt liste în care elementele nu sunt numerotate, ci în dreptul fiecăruia este afișat un marcator.

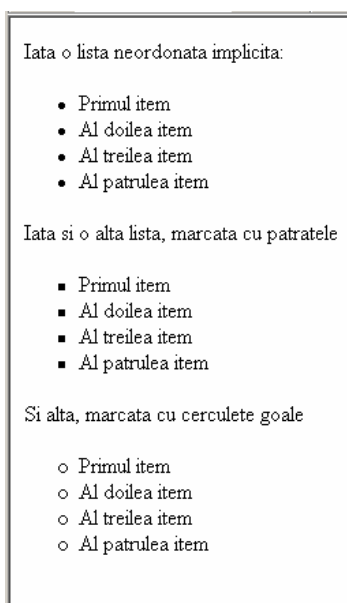
Inserarea lor în cadrul documentului HTML se face prin tag-urile `<UL>...</UL>`, elementele (itemii) lor fiind introduse între aceste două tag-uri prin `<LI>...</LI>` (tag-ul de sfârșit nefiind obligatoriu).

Implicit, marcarea lor se face prin cerceulețe pline. Ea poate fi modificată prin folosirea atributului **type** în cadrul tag-ului **UL**. Acesta poate lua una dintre valorile:

- **disc** : marcarea se face cu cerceulețe pline (implicit)
- **square** : marcarea se face cu pătrățele
- **circle** : marcarea se face cu cerceulețe goale

Iată un exemplu de cod și vizualizarea sa în browser:

```
Iata o lista neordonata implicita:
<UL>
  <LI>Primul item</LI>
  <LI>Al doilea item</LI>
  <LI>Al treilea item</LI>
  <LI>Al patrulea item</LI>
</UL>
Iata si o alta lista, marcata cu patratele
<UL type="square">
  <LI>Primul item</LI>
  <LI>Al doilea item</LI>
  <LI>Al treilea item</LI>
  <LI>Al patrulea item</LI>
</UL>
Si alta, marcata cu cerceulete goale
<UL type="circle">
  <LI>Primul item</LI>
  <LI>Al doilea item</LI>
  <LI>Al treilea item</LI>
  <LI>Al patrulea item</LI>
</UL>
```



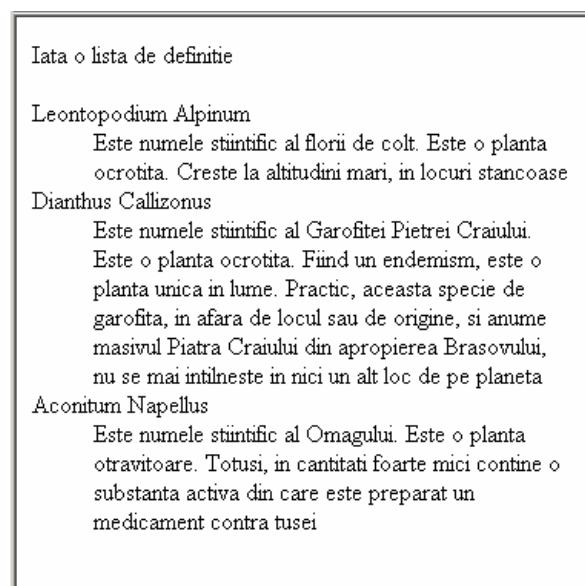
- **Liste de definiție (Definition Lists)**: au rolul de a descrie o listă de definiții.

Inserarea lor în cadrul documentului HTML se face prin tag-urile **<DL>...</DL>**. Elementele lor sunt de două tipuri:

- Termenul care este definit: este introdus între tag-urile **<DT>...</DT>** (tag-ul de sfârșit nefiind obligatoriu).
- Definiția propriu-zisă: este introdusă între tag-urile **<DD>...</DD>** (tag-ul de sfârșit nefiind obligatoriu).

Iată un exemplu de cod și vizualizarea sa în browser:

```
Iata o lista de definitie:
<DL>
  <DT>Leontopodium Alpinum</DT>
  <DD>Este numele stiintific al florii de colt. Este o planta ocrotita. Creste la altitudini mari, in locuri stancoase</DD>
  <DT>Dianthus Callizonus</DT>
  <DD>Este numele stiintific al Garofitei Pietrei Craiului. Este o planta ocrotita. Fiind un endemism, este o planta unica in lume. Practic, aceasta specie de garofita, in afara de locul sau de origine, si anume masivul Piatra Craiului din apropierea Brasovului, nu se mai intilnește in nici un alt loc de pe planeta</DD>
  <DT>Aconitum Napellus</DT>
  <DD>Este numele stiintific al Omagului. Este o planta otravitoare. Totusi, in cantitati foarte mici contine o substanta activa din care este preparat un medicament contra tusei</DD>
</DL>
```



## 2.3.6. Imagini.

Tag-ul utilizat pentru inserarea unei imagini în documentul HTML este **<IMG>**. Forma generală a acestui element este **<IMG attribute>**. Acest tag nu are și formă de închidere.

Atributele sale sunt:

- **src** identifică fișierul efectiv de pe disc, ce conține imaginea respectivă. Dacă imaginea se află în directorul curent, se specifică doar numele și extensia sa. Dacă se află într-un subdirector, acesta se specifică înaintea numelui și extensiei imaginii, separat prin caracterul /. Imaginile recunoscute de majoritatea browser-elor internet sunt de tip .jpg, .gif, .png

- **align** specifică tipul de aliniere al imaginii în raport cu textul în cadrul căruia se află. Acesta poate lua una dintre valorile următoare:

- **right** : imaginea se aliniază în dreapta, iar textul care urmează este scris în locul rămas liber, în stânga acesteia;

- **left** : imaginea se aliniază în stânga, iar textul care urmează este scris în locul rămas liber, în dreapta acesteia;

- **top** : doar latura de sus a imaginii se aliniază cu rândul de text în cadrul căruia se află; următorul rând de text va fi afișat după imagine, ocupând întreaga lățime a ecranului;

- **middle** : rândul de text în cadrul căruia se află imaginea se aliniază la jumătatea înălțimii acesteia; următorul rând de text va fi afișat după imagine, ocupând întreaga lățime a ecranului;

- **bottom** : doar latura de jos a imaginii se aliniază cu rândul de text în cadrul căruia se află; următorul rând de text va fi afișat după imagine, ocupând întreaga lățime a ecranului;

- Dacă dorim întreruperea unei alinieri de imagine de tip right sau left înainte ca textul să fi umplut spațiul liber din stânga, respectiv dreapta acesteia, putem folosi tag-ul br, căruia îi adăugăm unul dintre atributele **clear="left"** sau **clear="right"** sau **clear="all"**, după caz.

- atributul **alt="text"** permite specificarea unui text alternativ ce va fi afișat fie dacă menținem cursorul de mouse asupra imaginii, fie în locul imaginii propriu-zise, în cazul în care imaginea nu poate fi încărcată din cauza unei probleme de conexiune.

Iată câteva exemple, cu tot cu vizualizarea lor în browser:

1) Exemplu la folosirea atributului **align="right"** și a atributului

**alt="text"** :

```
<P>Acest text este asezat inaintea imaginii<br>
  <IMG SRC="dog.jpg" align="right" alt="catelus">
  In schimb, acest text este aliniat in stinga imaginii,
  deoarece am folosit atributul align="right" in momentul
  in care am inserat imaginea in pagina noastra web prin
  intermediul tag-ului src.
</P>
```



## 2) Exemplu la folosirea opțiunii **align="right"** împreună cu tag-ul **<br clear="right">** :

```
<P>
Acest text este asezat inaintea imaginii<br>
<IMG SRC="dog.jpg" align="right" alt="catelus">
Acest text, aliniat in stinga imaginii, il
intrerupem fortat AICI
<BR clear="right">
In acest fel, restul textului se va alinia
in mod obisnuit, sub imagine, restul spatiului
din stinga raminand liber.
</P>
```



## 3) Exemplu la folosirea opțiunii **align="top"** :

```
<P>
Acest text este asezat inaintea imaginii <br>
Se observa ca
<IMG SRC="dog.jpg" align="top" alt="catelus">
doar primul rind al textului este aliniat cu
latura de sus a imaginii, restul textului
fiind afisat dupa imagine
</P>
```



## 4) Exemplu la folosirea opțiunii **align="middle"** :

```
<P>
Acest text este asezat inaintea imaginii <br>
Se observa ca
<IMG SRC="dog.jpg" align="middle" alt="catelus">
doar primul rind al textului este aliniat la
jumatatea inaltimii imaginii, restul textului
fiind afisat dupa imagine
</P>
```



## 5) Exemplu la folosirea opțiunii **align="bottom"** :

```
<P>
Acest text este asezat inaintea imaginii <br>
Se observa ca
<IMG SRC="dog.jpg" align="top" alt="catelus">
doar primul rind al textului este aliniat cu
latura de jos a imaginii, restul textului
fiind afisat dupa imagine
</P>
```



• attributele **height** și **width** permit specificarea altor dimensiuni pentru imagine, decât cele reale ale acesteia. Evident, dacă dimensiunile nu sunt proporționale cu cele reale, imaginea va fi deformată. Totodată, dacă specificăm dimensiuni mai mari decât cele reale, imaginea se va vedea mai puțin clar. În realitate, imaginea este transferată de pe server la dimensiunile sale originale, redimensionarea având loc doar la nivelul calculatorului pe care este vizualizată pagina.

Iată un exemplu de folosire al celor două tag-uri, și vizualizarea acestui exemplu în browser:

```
<P>
  Imaginea originala are dimensiunile 200x150:
  <BR>
  <IMG src="dog.jpg">
  <BR>
  Iată-o redimensionată proporțional la 100x75:
  <BR>
  <IMG src="dog.jpg" width="100" height="75">
  <BR>
  Iată-o și deformată:<BR>
  <IMG src="dog.jpg" width="50" height="100">
  sau
  <IMG src="dog.jpg" width="150" height="50">
  <BR>
</P>
```



- atributul **border** permite stabilirea grosimii unui chenar care va înconjura poza. Implicit, valoarea acestui atribut este "0", ceea ce înseamnă că imaginea nu este înconjurată de chenar:

```
<P>
  Imaginea este înconjurată
  de un chenar
  de dimensiune 10<br>
  <IMG src="dog.jpg" border="10">
</P>
```



- attributele **hspace="nr.pixeli"** respectiv **vspace="nr.pixeli"** permit stabilirea distanței minime care separă imaginea de celelalte obiecte pe verticală, respectiv pe orizontală:

```
<P>
  Iată o aliniere a imaginii
  de tip "right", aliniere
  <IMG src="dog.jpg" align="right">
  în cadrul căreia nu am modificat
  nici unul dintre cele două
  attribute care controlează
  spațiarea dintre imagine
  și restul elementelor, pe
  orizontală respectiv pe
  verticală
  <BR clear="all"><BR>
  Iată acum o aliniere a imaginii
  tot de tip "right", aliniere
  <IMG src="dog.jpg" align="right"
    hspace="15" vspace="20">
  în cadrul căreia am modificat
  ambele attribute care controlează
  spațiarea dintre imagine
  și restul elementelor, stabilind
  valorile de 20 pe verticală
  respectiv de 15 pe orizontală
  <BR clear="all">
</P>
```







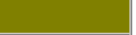
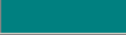












### 2.3.7. Specificarea culorilor în HTML.

O serie de elemente din HTML permit utilizarea de atribute de culoare. Acestea pot fi specificate în două moduri:

- prin constanta HTML ce reprezintă numele culorii (în engleză, bineînțeles). Există 216 astfel de constante recunoscute de majoritatea browser-elor. Ne vom limita în a le enumera doar pe cele 16 care sunt considerate de bază, exemplificându-le pe fiecare:

| Numele culorii | Aspectul sau  | Numele culorii | Aspectul sau  | Numele culorii | Aspectul sau   | Numele culorii | Aspectul sau  |
|----------------|---|----------------|---|----------------|--|----------------|---|
| aqua           |  | gray           |  | navy           |  | silver         |  |
| black          |  | green          |  | olive          |  | teal           |  |
| blue           |  | lime           |  | purple         |  | white          |  |
| fuchsia        |  | maroon         |  | red            |  | yellow         |  |




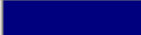




O serie dintre culori (însă nu toate) au și constante în variantele „dark” (î închis) respectiv „light” (deschis). De exemplu: darkred sau lightblue.

- prin constanta de tip RGB (Red, Green, Blue):


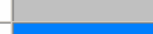



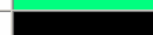



Principiul de bază al redării electronice ale unei imagini în culori se bazează pe amestecarea în proporții diferite ale culorilor Roșu, Verde și Albastru. În acest mod, se poate obține orice culoare se dorește. În cazul culorilor pe care le poate reda un browser HTML, fiecare dintre aceste componente de culoare poate avea 256 de stări posibile: de la 0, care înseamnă că respectiva culoare lipsește cu desăvârșire, până la 255, care înseamnă că respectiva culoare este folosită la intensitatea maximă. În acest fel, prin amestecuri diferite, putem obține  $256^3$ , deci aproximativ 16 milioane de nuanțe diferite.

Componentele de culoare în HTML se specifică folosind numere hexazecimale. Astfel, fiecare dintre numerele dintre 0 și 255 se codifică în hexazecimal printr-un număr între 00 și FF. Constanta HTML pentru specificarea unei culori are forma generală **#RRGGBB**, în care **RR**, **GG** respectiv **BB** reprezintă câte un număr hexazecimal cuprins între 00 și FF.

Iată câteva exemple de culori obținute folosind constante de forma celei de mai sus:

| Numele culorii                          | Aspectul sau  | Numele culorii                             | Aspectul sau  |
|---|---|--|---|
| #FF0000 (roșu de intensitate maxima)    |  | #0000FF (albastru de intensitate maxima)   |  |
| #7F0000 (roșu de intensitate jumătate)  |  | #00007F (albastru de intensitate jumătate) |  |
| #00FF00 (verde de intensitate maxima)   |  | #FFFF00 (roșu și verde ambele la maxim)    |  |
| #007F00 (verde de intensitate jumătate) |  | #FF00FF (roșu și albastru ambele la maxim) |  |

| Numele culorii                                | Aspectul sau  | Numele culorii                                 | Aspectul sau  |
|---|---|--|---|
| #00FFFF (roșu și albastru ambele la maxim)    |  | #FF007F (roșu la maxim, albastru la jumătate)  |  |
| #7FFF00 (roșu la jumătate, verde la maxim)    |  | #007FFF (verde la jumătate, albastru la maxim) |  |
| #FF7F00 (roșu la maxim, verde la jumătate)    |  | #00FF7F (verde la maxim, albastru la jumătate) |  |
| #7F00FF (roșu la jumătate, albastru la maxim) |  | #000000 (toate culorile sunt stinse)           |  |
|   |   | #FFFFFF (toate culorile sunt aprinse la maxim) |  |

### 2.3.8. Tabele.

Tabelele reprezintă un element foarte important al unei pagini web. În foarte multe cazuri, tabele cu chenare invizibile sunt folosite ca și „schelet” al paginii, pentru a putea realiza alinieri complexe ale elementelor acesteia.

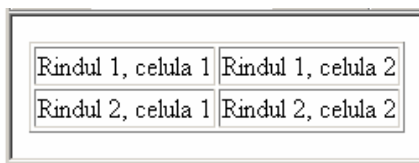
Tag-ul pentru descrierea unui tabel este `<TABLE>...</TABLE>`. În cadrul acestora trebuie descrise liniile (rândurile) tabelului, în cadrul fiecărui rând trebuind descrise celulele acestuia.

Descrierea unui rând se face între tag-urile `<TR>...</TR>`. La rândul lor, celulele din cadrul rândului se descriu între `<TD>...</TD>`. Atît tag-ul `</TR>` cât și tag-ul `</TD>` pot fi omise.

Un prim atribut al tag-ului `<TABLE>` este `border="grosime_pixel"`. Dacă acest atribut este omis, tabelul va avea un chenar invizibil. Dacă se specifică doar atributul, omițând grosimea, aceasta va fi luată, implicit, ca fiind 1.

Iată un exemplu de cod pentru definirea unui tabel:

```
<TABLE border>
  <TR>
    <TD>Rindul 1, celula 1
    <TD>Rindul 1, celula 2
  <TR>
    <TD>Rindul 2, celula 1
    <TD>Rindul 2, celula 2
</TABLE>
```



|                    |                    |
|--------------------|--------------------|
| Rindul 1, celula 1 | Rindul 1, celula 2 |
| Rindul 2, celula 1 | Rindul 2, celula 2 |

#### Atribute ale tag-ului `<TABLE>`

- `cellpadding="nr_pixel"` permite stabilirea unui spațiu care va fi lăsat, în fiecare celulă a tabelului, între conținutul celulei și marginile acesteia. Dacă nu se specifică acest atribut, el este în mod implicit considerat 0
- `cellspacing="nr_pixel"` permite stabilirea spațiului care va fi lăsat între chenarele celulelor vecine în tabel (și inclusiv între ele și chenarul exterior al tabelului). Dacă nu se specifică acest atribut, el este în mod implicit considerat 2.

Conținutul unei celule poate fi cât se poate de general: de la text și imagini până la alte tabele (se pot deci construi chiar și tabele imbricate), ca în exemplul următor:



```

<TABLE border="1" cellspacing="4" cellpadding="5">
  <TR>
    <TD>
      Poza cu catelus<BR>
      <IMG src="dog.jpg">
    <TD>
      Tabel cu baieti
      <TABLE border cellspacing="0">
        <TR><TD>Mihai
        <TR><TD>Costel
        <TR><TD>Alin
      </TABLE>
    <TD>
      Tabel cu fete
      <TABLE border cellspacing="0">
        <TR><TD>Mihaela
        <TR><TD>Costina
        <TR><TD>Alina
      </TABLE>
    </TD>
  </TR>
</TABLE>

```



- **width="lățime"** poate stabili cât de lat să fie tabelul. Lățimea poate fi dată în procente, caz în care se va calcula ca și procent din lățimea ferestrei browser-ului (ex: **width="50%"**) sau în pixeli (ex: **width="500"**);

- **height="înălțime"** poate stabili cât de înalt să fie tabelul. Lățimea poate fi dată, la fel ca și în cazul atributului width, în procente sau în pixeli;

- **align** determină alinierea tabelului în pagină. Poate la una dintre valorile **left**, **right** sau **center**. Dacă, pe lângă tabel, mai scriem și text, acesta se va poziționa față de tabel în același mod în care se poziționează și față de imagini;

- **bgcolor="culoare"** permite stabilirea culorii de fundal a tuturor celulelor tabelului;

- **bordercolor="culoare"** permite stabilirea culorii chenarului (deopotrivă cel interior cât și cel exterior)

### Atribute ale tag-ului <TR>

- **align** determină, pentru toate celulele de pe linie, modul alinierii conținutului pe orizontală, în interiorul celulelor. Poate la una dintre valorile **left**, **right**, **center** sau **justify**;
- **valign** determină, pentru toate celulele de pe linie, modul alinierii conținutului pe verticală, în interiorul celulelor. Poate la una dintre valorile **top**, **bottom** sau **middle**;
- **bgcolor** determină, pentru toate celulele de pe linia respectivă, culoarea de fundal.

### Atribute ale tag-ului <TD>

- **width** și **height** determină, pentru celula respectivă, lățimea și înălțimea. Poate fi dată în procente sau pixeli. Dacă e specificată în procente, se va lua din lățimea, respectiv înălțimea

tabelului. Modificarea lăţimii şi a înălţimii unei celule va avea efect şi asupra celorlalte celule, pentru ca tabelul să fie aliniat;

- **align** şi **valign** stabilesc, la fel ca şi în cazul lui **<TR>**, modul în care este aliniat conţinutul în interiorul celulei, pe orizontală respectiv pe verticală, fiind prioritare faţă de alinierea la nivel de linie

- **colspan="n"** stabileşte întinderea celulei respective în dreapta cu **n** coloane (echivalentul operaţiei Merge Cells din Word, în cazul în care unim celule adiacente pe orizontală);

- **rowspan="n"** stabileşte întinderea celulei respective în jos cu **n** linii (echivalentul operaţiei Merge Cells din Word, în cazul în care unim celule adiacente pe verticală);

- **bgcolor** determină, pentru celula respectivă, culoarea de fundal. Evident, este prioritară faţă de acelaşi atribut la nivel de linie.

Exemplu:

```
<TABLE border="1" cellspacing="0" cellpadding="5">
  <TR bgcolor="#c0c0ff">
    <TD>Ziua
    <TD>09h00 - 11h00
    <TD>11h00 - 13h00
    <TD>13h00 - 15h00
  <TR bgcolor="yellow" align="center">
    <TD align="left"><B>Luni</B>
    <TD colspan="2">Mecanica
    <TD bgcolor="#ffd0d0">Termodinamica
  <TR bgcolor="yellow" align="center">
    <TD align="left"><B>Marti</B>
    <TD>Electrostatica
    <TD>Optica
    <TD>Atomica
  <TR bgcolor="yellow" align="center">
    <TD align="left"><B>Miercuri</B>
    <TD rowspan="2" bgcolor="#ffd0d0">Termodinamica
    <TD>Optica
    <TD>Electrostatica
  <TR bgcolor="yellow" align="center">
    <TD align="left"><B>Joi</B>
    <TD>Mecanica
    <TD>Optica
</TABLE>
```

| Ziua     | 09h00 - 11h00  | 11h00 - 13h00 | 13h00 - 15h00  |
|----------|----------------|---------------|----------------|
| Luni     | Mecanica       |               | Termodinamica  |
| Marti    | Electrostatica | Optica        | Atomica        |
| Miercuri | Termodinamica  | Optica        | Electrostatica |
| Joi      |                | Mecanica      | Optica         |

- Tag-ul **<TH>...</TH>** poate înlocui **<TD>...</TD>**. Atributele sunt aceleaşi. Singura diferenţă este că textele de după tag-ul **<TH>** sunt, în mod implicit, tipărite îngroşat (Bold) iar alinierea lor se face pe centru;

- Tag-ul **<CAPTION>...</CAPTION>** permite scrierea unui titlu pentru tabel. Acest tag trebuie să se găsească imediat după **</TABLE>**. Acest tag suportă atributul **align**. Acesta poate lua una dintre valorile: **left** (titlul va fi poziţionat în stânga sus), **right** (poziţionare dreapta sus), **top** (poziţionare pe centru sus), **bottom** (poziţionare pe centru jos);

Exemplu:

```
<H3>Colegiul National "Andrei Saguna"</H3>
<TABLE border="1" cellspacing="0"
cellpadding="5" align="left">
  <CAPTION align="bottom">
    Scorul pe echipe</CAPTION>
  <TR><TH>Echipa<TH>Punctaj
  <TR><TD>clasa a 9-a A<TD align="right">87
  <TR><TD>clasa a 10-a B<TD align="right">80
  <TR><TD>clasa a 12-a B<TD align="right">91
</TABLE> <FONT color="blue">
Colegiul National "Andrei Saguna"
Colegiul National "Andrei Saguna"
Colegiul National "Andrei Saguna"
Colegiul National "Andrei Saguna"
Colegiul National "Andrei Saguna"
Colegiul National "Andrei Saguna"
Colegiul National "Andrei Saguna"
Colegiul National "Andrei Saguna"
Colegiul National "Andrei Saguna"
Colegiul National "Andrei Saguna"
</FONT>
```

| Colegiul National "Andrei Saguna" |         |   |
|-----------------------------------|---------|---|
| Echipa                            | Punctaj | Colegiul National "Andrei Saguna" Colegiul National "Andrei Saguna" Colegiul National "Andrei Saguna" Colegiul National "Andrei Saguna" Colegiul National "Andrei Saguna" Colegiul National "Andrei Saguna" Colegiul National "Andrei Saguna" Colegiul National "Andrei Saguna" Colegiul National "Andrei Saguna" Colegiul National "Andrei Saguna" |
| clasa a 9-a A                     | 87      |   |
| clasa a 10-a B                    | 80      |   |
| clasa a 12-a B                    | 91      |   |
| Scorul pe echipe                  |         |   |

### 2.3.9. Legături (link-uri).

Așa cum am văzut în partea introductivă a acestui capitol, noțiunea de [www](#) este strâns legată de documentele de tip hipertext.

Tot ceea ce am prezentat din limbajul HTML până în momentul de față, reprezintă doar partea descriptivă a acestuia, cu ajutorul căreia putem crea un conținut static.

Link-urile reprezintă mecanismul prin care:

- putem face ca un vizitator al paginii, prin executarea unui click, să poată accesa o altă pagină, la care dorim să-i creăm posibilitatea unui acces rapid și, dacă acesta dorește, să poată reveni în pagina inițială prin apăsarea butonului **Back** al browser-ului de Internet;
- putem face ca un vizitator al paginii noastre să primească un anumit fișier, de orice tip, care se găsește pe site-ul nostru (download);
- putem face ca un vizitator al paginii noastre să poată asculta un mesaj sonor sau chiar să poată viziona un film;
- putem ca, printr-un click, să putem vizualiza o pagină (inclusiv cea curentă) doar dintr-un anumit loc, fără a folosi barele de derulare;
- putem ca, prin accesarea unui click, cel care vizitează pagina să ne poată trimite un e-mail.

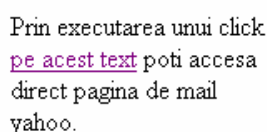
Pentru toate acestea, vom folosi tag-ul **<A>...</A>**, numit și Ancoră.

## Ancore de legătură către alte pagini

Acestea permit ca un anumit element din document să conțină legătura către o altă pagină. Elementul care face legătura este de obicei un text sau o imagine. De regulă, elementul legat își schimbă aspectul față de cel clasic: textul va fi subliniat și colorat altfel, iar imaginea va avea un chenar colorat. În momentul în care ducem cursorul deasupra elementului legat, acesta capătă forma unei mâini, indicându-ne astfel că este vorba de un link pe care îl putem utiliza. Printr-un simplu click, accesăm pagina către care este făcută legătura.

Acest tip de legătură se realizează practic folosind atributul **href**, ca în exemplul de mai jos (a se remarca modul în care, elementul legat, este inclus între tag-urile **<A href=...** și **</A>**):

```
<p> Prin executarea unui click  
<A href="http://mail.yahoo.com">  
pe acest text </A>  
poti accesa direct pagina de mail yahoo.  
</p>
```



După cum se observă, atributul **href** primește adresa completă a paginii către care dorim să facem legătura. Dacă e vorba de un fișier local, din același director cu pagina din care facem legătura, la **href** este suficient să scriem numele și extensia (de ex: **href="pagina.htm"**)

## Ancore de legătură către fișiere (pentru download)

Se realizează în mod analog cu cele către alte pagini, la atributul href trebuind specificat adresa fișierului respectiv (dacă este în același director cu pagina din care facem legătura, e suficient să-i scriem numele și extensia).

Ex: în cazul în care fișierul pentru download este local:

```
Pentru download arhiva executa un click  
<A href="arhiva.zip">aici</A>
```

Ex: în cazul în care fișierul pentru download se află la altă adresă:

```
Pentru a descarca subiectele de bacalaureat la disciplina informatica, da un click  
<A href="http://subiecte2008.edu.ro/bacalaureat/subiecte/E_informatica_c.zip">aici</A>
```

Legăturile către fișiere de tip sunet sau film se fac absolut în aceeași manieră. În funcție de extensia lor (.wav, .mid, .mp3, .avi) în momentul executării unui click asupra obiectului care face legătura către ele, acestea vor fi deschise automat către browser cu programul corespunzător.

## Legături relative la conținutul documentului (paginii)

Sunt acele ancore care permit accesarea directă a unei pagini web într-un anumit loc, fără a mai folosi barele de derulare pentru a ajunge în acel loc.

Pentru aceasta, locul respectiv trebuie marcat. Acest lucru se face tot cu ajutorul tag-ului **<a>**, însă folosind atributul **id**, care va denumi locul respectiv printr-un identificator, ca în exemplul de mai jos (a se observa că între tag-ul de deschidere și cel de închidere putem să nu punem nici un element):

```
<A id="capitolul2"></A>
```

Accesarea directă a acestui loc cu ajutorul unui link se poate face astfel:

a) Din interiorul aceleiași pagini: specificând la atributul **href** identificatorul respectiv (cel de la **id**) înainte de care se pune de caracterul **#**, ca în exemplul următor:

```
<A href="#capitolul2">Salt direct la capitolul 2</A>
```

b) Din altă pagină: specificând la atributul **href** adresa paginii accesate (a fișierului html) urmată de caracterul **#**, ca în exemplul următor:

```
<A href="http://www.myserver.ro/document.html#capitolul2">Deschide documentul extern,  
direct la capitolul 2</A>
```

## Ancoră de legătură pentru trimiterea unui e-mail

Acestea permit ca, atunci când o persoană ne vizitează site-ul, dacă dorește, să ne poată trimite un e-mail făcând un simplu click pe legătura respectivă. Totuși, pentru ca acest lucru să fie funcțional, cel care vizitează site-ul trebuie să aibă configurat pe calculatorul său un client de e-mail (cel mai frecvent este Outlook Express).

Iată un exemplu pentru o astfel de ancoră:

```
<ADDRESS>
```

```
Click <A href="mailto:somebody@someserver.com">aici</A> pentru a trimite un e-mail
```

```
</ADDRESS>
```

(tag-ul **<ADDRESS>...</ADDRESS>** nu face altceva decât să afișeze textul din cadrul său italic)

După cum se observă, pentru trimiterea unui e-mail, după atributul **href** trebuie specificat **mailto**: urmat de adresa de e-mail a destinatarului.

### 2.3.10. Elemente de structură (HTML, HEAD, BODY).

După cum am văzut în partea introductivă, orice document html este cuprins între tag-urile **<HTML>** și **</HTML>**. El este alcătuit dintr-un unic antet (**HEAD**) și un unic corp (**BODY**). Aceste 3 elemente au rolul de a defini structura documentului. Din acest motiv ele se mai numesc și elemente de structură.

Tag-ul **BODY** poate conține următoarele atribute:

- **background="fișier\_imagine"** permite specificarea unei imagini de fundal. Aceasta se va repeta atât pe orizontală cât și pe verticală, până când se acoperă întreaga suprafață necesară corpului;
- **bgcolor="culoare"** permite specificarea unei culori de fond;
- **text="culoare"** permite specificarea culorii întregului text cuprins în pagină;
- **link="culoare"** permite specificarea culorii unui link nevizitat;
- **alink="culoare"** permite specificarea culorii unui link activ; un link este considerat activ în timpul vizitării și imediat după aceasta;
- **vlink="culoare"** permite specificarea culorii unui link vizitat, care nu mai este activ.

### Conținutul secțiunii <HEAD>

În cadrul acestei secțiuni putem întâlni diverse alte tag-uri. Despre tag-ul **<TITLE>** am discutat deja, el permițând scrierea unui titlu pentru pagină.

În afară de acestea, vom aminti încă alte 3 tag-uri:

- **<BASE>** permite stabilirea unei adrese de bază pentru resurse. Acest tag se folosește în special atunci când resursele (sau, în fine, o mare parte a acestora) se găsesc în alt director decât cel în care se află documentul curent. În acest fel, folosirea fișierelor din directorul specificat în **BASE** se poate face direct prin numele și extensia lor. Specificarea se face prin:

**<BASE href="adresa resurse">**

- **<META>** este folosit pentru a furniza informații motoarelor de căutare. Unele dintre acestea vizitează doar antetul pentru a obține informații. Informațiile conținute de acest element nu sunt afișate de browser, însă este important să îl folosim pentru ca informațiile conținute în site-ul nostru să fie accesibile. Locul tag-ului **<META>** este în antet (**<HEAD>**).

Atributele tag-ului **<META>** sunt **name** și **content**. Folosirea lor este ceva mai particulară, rezultând din exemplele următoare:

- pentru a specifica autorul unui document:

```
<META name="Author" content="Prenume NUME">
```

- pentru a specifica titlul unui document:

```
<META name="TITLE" content="Metode de programare">
```

- pentru a preciza cuvintele cheie după care să fie regăsit site-ul:

```
<META name="KEYWORDS" content="backtracking, divide et impera, greedy, programare dinamica">
```

- pentru a specifica limba în care este scris site-ul:

```
<META name="LANGUAGE" content="RO">
```

Există și alte atribute ale elementului META, însă cele două deja prezentate sunt suficiente.

- **<STYLE>** este utilizat pentru introducerea stilurilor. Acestea permit stabilirea mai amănunțită a modului în care apar, implicit, diferitele elemente din document. Valorile se trec între **<STYLE>...</STYLE>**.

Exemplu:

```
<STYLE>
P {font-family:"Comic Sans MS"; font-size:14pt;}
</STYLE>
```

Prin specificarea lui P înainte de paranteza acoladă, stabilim ca modul implicit de afișare al paragrafelor (să ne reamintim că **<P>** este tag-ul pentru paragraf) să fie cel descris între parantezele acolade, deci, în cazul exemplului de față font-ul folosit să fie "Comic Sans MS", iar dimensiunea caracterelor să fie de 14.

- **<SCRIPT>** este utilizat pentru introducerea anumitor secvențe de program în cadrul paginilor web. Există mai multe limbaje (numite de scriptare) care permite scrierea acestor secvențe, cum ar fi JavaScript, VBscript. Specificarea limbajului în care este codat scriptul se face cu ajutorul atributului **language**, ca în exemplul de mai jos:

```
<SCRIPT language="JavaScript">
function calcul()
{ s=0; for(i=1;i<=10;i++)
      s+=i;
  alert("suma nr. de la 1 la 10 este: "+s);}
</SCRIPT>
...
<BODY onload="calcul();" > ... </BODY>
```

Acest exemplu definește în antetul paginii o funcție JavaScript capabilă să calculeze suma numerelor de la 1 la 10 într-o variabilă s și-apoi să afișeze valoarea obținută prin intermediul unei ferestre de dialog. Funcția este apelată automat (atributul **onload**) la încărcarea paginii.

### 2.3.11. Pagini cu cadre (FRAMESET, FRAME, IFRAME).

Utilizarea frame-urilor permite ca, în cadrul aceleiași ferestre ale browser-ului să fie afișate simultan mai multe documente HTML (sau alte resurse).

Tag-ul **<FRAMESET>** are rolul de a împărți fereastra în mai multe cadre. În fișierul HTML, el înlocuiește tag-ul **<BODY>**. Iată câteva atribute ale lui **FRAMESET**:

- **rows** – descrie liniile în care este împărțită secțiunea **FRAMESET** respectivă
- **cols** – descrie coloanele în care este împărțită secțiunea **FRAMESET** respectivă

descrierile pentru **rows**, respectiv **cols**, pot fi de forma:

```
<FRAMESET rows="30%, 50%, 20%">
  <FRAME ...>
  <FRAME ...>
  <FRAME ...>
</FRAMESET>
```

în acest exemplu, se definesc 3 cadre orizontale (linii) de înălțimi 30%, 50% respectiv 20% din înălțimea ferestrei.

Un alt exemplu, în care înălțimea cadrelor este definită proporțional:

```
<FRAMESET rows="3*, 1*, 2*">...
```

aici se definesc 3 cadre orizontale, proporționale cu 3, 1 și 2 dintr-o înălțime de  $3+2+1=6$  (deci cadrele vor fi  $3/6$ ,  $1/6$  respectiv  $2/6$  din înălțimea ferestrei)

Un alt exemplu, în care înălțimea cadrelor este definită în pixeli:

```
<FRAMESET rows="100, 200, *">...
```

aici se definesc trei frame-uri: unul de înălțime de 100 de pixeli, altul de 200 de pixeli, al treilea fiind alocat cu spațiul rămas.

Tag-ul **<NOFRAMES>...</NOFRAMES>** reprezintă conținutul care va fi afișat unui vizitator, în cazul în care browser-ul său nu poate afișa cadre (în prezent, nu prea mai este cazul unor asemenea browsere).

Fiecare tag **<FRAMESET>...</FRAMESET>** trebuie ca, după definirea aspectului (cu ajutorul unuia dintre atributele **cols** sau **rows**) să conțină descrierile fiecăruia dintre cadrele definite. Acest lucru se face cu ajutorul tag-ului **<FRAME>** prin intermediul atributelor:

- **src** – adresa fișierului HTML sau a imaginii care se va încărca inițial în cadru;
- **marginheight** – marginile (în pixeli sau procent) față de partea de sus și cea de jos;
- **marginwidth** – marginile (în pixeli sau procent) față de partea din stânga și din dreapta;



- **frameborder** – poate lua valorile **1** (implicită), care înseamnă că acest cadru este separat de celelalte printr-un chenar, respectiv **0**, care înseamnă că acest cadru nu mai este separat de celelalte printr-un chenar.

- **scrolling** – tratează afișarea barei de scroll (derulare). Poate lua trei valori:

  - auto** – valoarea implicită. Bara de scroll este prezentă numai dacă este cazul

  - yes** – bara de scroll este totdeauna prezentă

  - no** – bara de scroll nu va fi niciodată afișată

- **noresize** – dacă atributul acesta este prezent (el se folosește fără a i se atribui nici o valoare) atunci vizitatorului paginii nu i se va permite să redimensioneze cadrul. Prezența acestui atribut pentru un cadru nu permite nici redimensionarea cadrelor vecine.

- **name** – este un atribut foarte important. Prin intermediul său va putea fi identificat frame-ul respectiv. Acest lucru este foarte important, deoarece dintr-un cadru se poate comanda conținutul oricărui alt cadru.

Deschiderea unei pagini într-un anumit cadru, prin intermediul ancorelor, se poate specifica prin folosirea atributului **target="nume cadru"** imediat după folosirea atributului **href** în cadrul tag-ului **<A href="..." .. >**.

Iată un exemplu prin care definim o pagină cu două frame-uri verticale. Frame-ul din stânga va conține numele a 3 zile ale săptămânii (pe limba română). Accesarea fiecăruia va produce deschiderea în frame-ul drept a unei pagini care va conține traducerea numelui zilei respective în 4 limbi.

În total vom avea de construit 5 fișiere:

- un fișier pentru pagina inițială, cea care definește scheletul paginii cu frame-uri
- un fișier cu cele 3 zile ale săptămânii, pe fiecare dintre ele fiind pus câte un hyperlink care va deschide traducerea numelui său în celălalt frame
- 3 fișiere cu traducerilor numelor zilelor în 4 limbi străine.

Pagina inițială: **pagframe.html**:

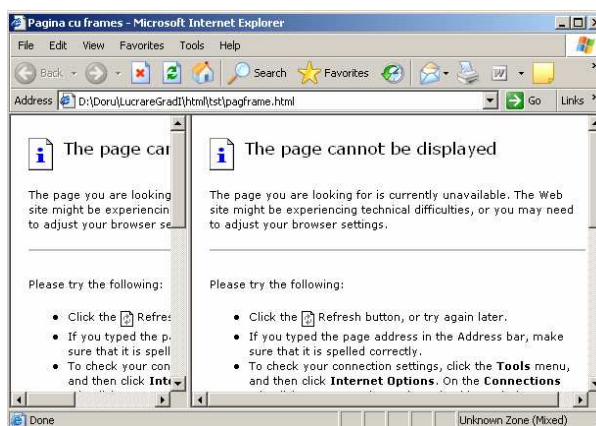
```
<HTML>
<HEAD>
  <TITLE>Pagina cu frames</TITLE>
</HEAD>
<FRAMESET cols="30%,*">
  <FRAME name="stinga" src="zile.html" noresize>
  <FRAME name="dreapta" src="luni.html">
</FRAMESET>
<NOFRAMES>
  Browser-ul tau nu este capabil sa afiseze pagini cu frame-uri
</NOFRAMES>
</HTML>
```

De remarcat faptul că această fișier HTML nu conține decât scheletul cadrelor, ele urmând a fi populate inițial, după cum remarcăți din codul sursă, cu fișierele zile.html pentru primul cadru (cel din stânga) respectiv cu fișierul luni.html pentru cel de-al doilea cadru.

Observați modul în care au fost definite cadrele în cadrul tag-ului **FRAMESET**: **cols="30%,\*"**. Acest lucru semnifică prezența a două cadre verticale (coloane) dintre care primul va ocupa 30% din lățimea ferestrei, iar al doilea restul (lucru semnatificat de caracterul \* care închide șirul de definiție al cadrelor).

De asemenea, atributul **noresize** în cadrul primului tag **FRAME** împiedică redimensionarea cadrelor de către utilizator. În cazul în care acest atribut nu ar fi fost prezent, utilizatorul, printr-un simplu „drag and drop” ar fi putut trage bara care separa cele două frame-uri, dându-i orice poziție ar fi dorit.

Dacă încărcăm în browser-ul de internet documentul creat în acest stadiu, fără ca pe disc să existe vreunul dintre celelalte patru fișiere planificate, am obține următorul rezultat:



Acesta era și de așteptat, de altfel, deoarece el demonstrează existența frame-urilor și lipsa conținutului.

Iată și conținutul celorlalte fișiere, pe care le vom pune în același director cu documentul de mai sus (în dreptul fiecăruia vom arăta și vizualizarea sa în browser):

Fișierul **zile.html**:

```
<HTML>
<HEAD> <TITLE>Zilele</TITLE> </HEAD>
<BODY>
<br>
<A href="luni.html" target="dreapta">Luni</A><br><br>
<A href="marti.html" target="dreapta">Marti</A><br><br>
<A href="miercuri.html" target="dreapta">Miercuri</A><br><br>
</BODY>
</HTML>
```



De remarcat modul în care am realizat link-urile asupra celor 3 cuvinte: folosind și atributul **target** în cadrul ancorei (**<A ...>**) am specificat browser-ului ca paginile respective să fie deschise în cadrul frame-ului al cărui nume apare după **target**.

#### Fișierul **luni.html**:

```
<HTML><BODY>  
<H2>Luni</H2>  
FR: Lundi<BR>  
IT: Lunedì<BR>  
GE: Montag<BR>  
EN: Monday<BR>  
</BODY></HTML>
```



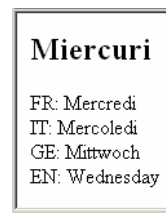
#### Fișierul **marti.html**:

```
<HTML><BODY>  
<H2>Marti</H2>  
FR: Mardi<BR>  
IT: Martedì<BR>  
GE: Dienstag<BR>  
EN: Tuesday<BR>  
</BODY></HTML>
```



#### Fișierul **miercuri.html**:

```
<HTML><BODY>  
<H2>Marti</H2>  
FR: Mardi<BR>  
IT: Martedì<BR>  
GE: Dienstag<BR>  
EN: Tuesday<BR>  
</BODY></HTML>
```



Iată cum arată vizualizarea finală în browser (după crearea celor 4 fișiere de mai sus):



Evident, la efectuarea unui click asupra legăturilor (luni, marti, miercuri) din partea stângă, se va produce deschiderea paginii corespunzătoare în frame-ul drept.

Tag-ul **<IFRAME>** este un element care nu a fost prezent în primele versiuni ale limbajului HTML, ci a apărut ceva mai nou. Actualmente, folosirea sa este preferată de majoritatea celor care programează pagini web, deoarece se comportă ceva mai flexibil decât cadrele clasice. Totodată, motoarele de căutare nu indexează conținutul paginilor cu frame-uri obișnuite, pe când cele care conțin iframe-uri sunt indexate.

Prin intermediul său, este permisă crearea unui cadru în corpul unui documente HTML, cadrul care se comportă asemănător unei imagini.

Atributele lui **IFRAME** sunt:

- **name** – la fel ca și la **FRAME**, acest atribut permite identificarea **IFRAME**-ului (pentru a putea comanda conținutul său din orice link)
- **height**, **width** înălțimea, respectiv lățimea. Pot fi specificate atât în pixeli, cât și în procente, relativ la dimensiunile ferestrei browser-ului
- **frameborder** – poate lua valoarea **0** sau **1**, la fel ca la elementul **FRAME**
- **src** – adresa resursei care va fi încărcată inițial în **IFRAME**
- **marginwidth**, **marginheight**, **scrolling** – la fel ca și la **FRAME**
- **align** – poate lua una dintre valorile **left**, **right**, **top**, **bottom**, **middle**, comportându-se întocmai ca și în cazul imaginilor

Iată reluarea aceleiași idei structurale ca și la aplicația de dinainte (cu frame-uri clasice) însă folosind un element de tipul **IFRAME**. Fișierele **luni.html**, **marti.html** respectiv **miercuri.html** le păstrăm nemodificate. Practic, mai creăm doar un singur fișier HTML, cu conținutul următor, și avem grijă să copiem în același director și cele 3 fișiere de mai sus:

```
<HTML>
<HEAD><TITLE>Elementul IFRAME</TITLE></HEAD>
<BODY>
  <IFRAME name="cadru" width="140"
    height="160" align="right" src="luni.html">
  </IFRAME>
  <BR>
  <A href="luni.html" target="cadru">
    Luni</A><BR><BR>
  <A href="marti.html" target="cadru">
    Marti</A><BR><BR>
  <A href="miercuri.html" target="cadru">
    Miercuri</A><BR><BR>
</BODY>
</HTML>
```



### 2.3.12. Bare de separare (HR).

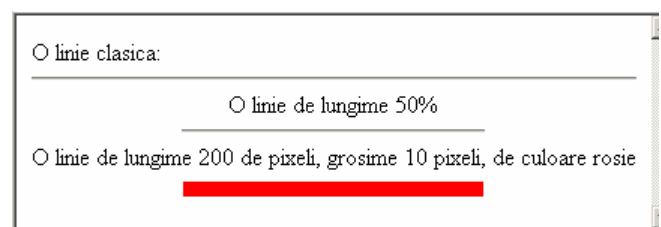
Bara de separare, al cărei tag este **<HR>** reprezintă un element decorativ. De obicei se folosește pentru a separa anumite secțiuni ale paginii web.

Atributele sale sunt:

- **width** – permite specificarea lungimii sale. Poate fi dată în pixeli sau în procente. Dacă acest atribut lipsește, atunci lungimea sa va fi maximă (din marginea stângă și până marginea dreaptă a ferestrei);
- **size** – permite specificarea înălțimii barei. Se specifică în pixeli;
- **color** – permite specificarea culorii sale.

Exemplu:

```
<BODY>
O linie clasica:
<HR>
<CENTER>O linie de lungime 50%</CENTER>
<HR width="50%">
<CENTER>O linie de lungime 200 de pixeli,
grosime 10 pixeli, de culoare rosie</CENTER>
<HR width="200" size="10" color="red">
</BODY>
```



### 2.3.13. Formulare.

Formularele sunt elemente ale limbajului HTML. Ele reprezintă o grupare de componente care permit trimiterea de date și de comenzi către un server. Acesta trebuie să fie mai mult decât un clasic server HTTP, trebuind să aibă instalată și o componentă capabilă de a răspunde comenzilor și a prelucra datele. Cea mai populară astfel de componentă, foarte larg utilizată în ultimii 10 ani în programarea pe Internet este limbajul PHP, de care ne vom ocupa pe larg în capitolul al III-lea al acestei lucrări.

Pentru moment ne vom concentra asupra componentelor unui formular și a aspectului acestora.

Un formular este descris prin intermediul tag-ului `<FORM>...</FORM>`. Atributele acestuia sunt:

- **action="adresa"** – acest atribut specifică adresa script-ului care se va ocupa de a răspunde la comenzi și de a prelucra datele.

- **method** – acest atribut specifică modul în care datele vor fi transmise către server.

Distinge, două valori pe care le poate lua acest atribut, și anume:

- **get** – datele sunt „la vedere” – acest lucru înseamnă că, în momentul trimiterii lor către server, ele vor apărea scrise în clar, în bara de adresă, într-un anumit format standard. De exemplu, dacă formularul trimite către pagina `test.php` o variabilă a care este egală cu 5, în bara de adresă a browser-ului ne va apărea `http://.../test.php?a=5`. Un dezavantaj major al acestei metode de trimitere a datelor este că volumul acestora este limitat (datorită șirului de caractere din adresă, care este limitat în cazul fiecărui browser).

- **post** – datele nu mai apar în mod explicit utilizatorului. Totuși, ele nu sunt criptate – practic, un program răufăcător le poate intercepta.

Pe lângă componentele specifice, un formular poate conține orice fel de alte elemente valide de HTML – tabele, imagini, text, bare de separare ...

În continuare vom prezenta câteva din componentele unui formular, prin intermediul cărora utilizatorul poate introduce date și trimite apoi aceste date către server. Un atribut foarte important al oricăruia dintre aceste componente este **name**, deoarece prin intermediul său, server-ul care va primi datele va ști despre care dintre controale este vorba.

## Câmpuri text

Permit utilizatorului să introducă date într-un câmp de tip edit (pe o singură linie).

Aceste se specifică prin tag-ul

```
<INPUT type="text" ...>
```

Atributele sale sunt:

- **size** – specifică lățimea (în număr aprox. de caractere) câmpului text; Dacă acest parametru este omis, este implicit considerat ca fiind 20;
- **maxlength** – specifică numărul maxim de caractere ce pot fi scrise în câmpul text. Acest atribut poate primi o valoare mai mare decât cea scrisă la size, caz în care, textul va defila în control (stânga dreapta) în cazul în care scriem mai multe caractere decât câte încap în porțiunea vizibilă. Omiterea acestui atribut va permite introducerea unui număr foarte mare de caractere (limita diferă de la un browser la altul);
- **name** – numele câmpului text (prin care server-ul va identifica acest câmp, pentru a prelua datele din el);
- **value** – poate specifica o valoare care să fie inițial (la încărcarea paginii) deja scrisă în cadrul controlului. Dacă oțitem acest atribut, câmpul text va fi gol.

## Butoane de tip „submit”

Aceasta componenta se prezintă sub forma unui buton. Prin apăsarea sa are loc trimiterea tuturor datelor din formular către script-ul de pe server-ul care le va prelucra.

Un control de tip submit se specifică prin tag-ul:

```
<INPUT type="submit" ...>
```

Atributele sale sunt:

- **name** – numele de identificare a componentei. Putem omite acest atribut. El se folosește în cazul în care același formular dorim să-i atașăm mai multe butoane de acest tip, iar apăsarea fiecăruia să producă o acțiune diferită;
- **value** – textul care va fi scris pe buton. De altfel, aceasta va fi și valoarea pe care server-ul o va primi pentru acest control.

## Câmpuri de tip password

Se comportă identic cu câmpurile de tip text. Singura deosebire este că, la scrierea de text în ele, acesta nu va fi vizibil, ci în locul caracterelor introduse se vor afișa asterisc-uri. Totodată, textul dintr-un astfel de control nu poate fi luat cu copy/paste.

Controalele de acest fel se specifică prin tag-ul:

**<INPUT type="password" ...>**

Atributele sunt identice cu cele de la **<INPUT type="text" ...>**

## Câmpuri de tip butoane radio

Sunt controalele care permit ca, dintr-o serie de opțiuni posibile, utilizatorul să aleagă una singură.

Controalele de acest fel se specifică prin tag-ul:

**<INPUT type="radio" ...>**

Atributele sale sunt:

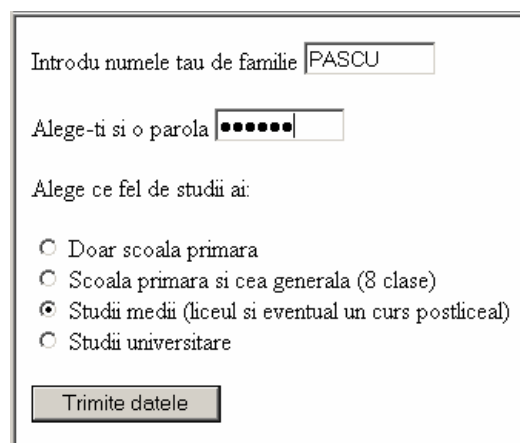
- **name** – numele de identificare al componentei. Este obligatoriu ca toate butoanele care aparțin aceluiași grup (deci seria de opțiuni din care trebuie aleasă doar una singură posibilă) să poarte același nume de identificare;
- **value** – valoarea pe care o va întoarce butonul respectiv, dacă el a fost cel ales;
- **checked** – dacă acest atribut este prezent, butonul respectiv va fi ales în mod implicit, la încărcarea paginii. Este recomandabil ca, dintre toate butoanele care aparțin aceluiași grup, exact unul singur să conțină acest atribut.

Iată și un exemplu care combină controalele prezentate până acum:

```
<FORM action="test.php" method="post">
  Introdu numele tau de familie
  <INPUT type="text" size="10" maxlength="20"
        name="numele">

  <BR><BR>
  Alege-ti si o parola
  <INPUT type="password" size="10" maxlength="20"
        name="parola">

  <BR><BR>
  Alege ce fel de studii ai:<BR><BR>
  <INPUT type="radio" name="studii" value="scprim">
  Doar scoala primara<BR>
  <INPUT type="radio" name="studii" value="8clase">
  Scoala primara si cea generala (8 clase)<BR>
  <INPUT type="radio" name="studii" value="medii"
        checked>
  Studii medii (liceul si eventual un curs postliceal)
  <BR>
  <INPUT type="radio" name="studii" value="univ">
  Studii universitare<BR><BR>
  <INPUT type="submit" value="Trimite datele">
</FORM>
```



The screenshot shows a web form with the following elements:

- A text input field labeled "Introdu numele tau de familie" containing the text "PASCU".
- A password input field labeled "Alege-ti si o parola" with masked characters (dots).
- A section titled "Alege ce fel de studii ai:" containing four radio button options:
  - ☐ Doar scoala primara
  - ☐ Scoala primara si cea generala (8 clase)
  - ☒ Studii medii (liceul si eventual un curs postliceal)
  - ☐ Studii universitare
- A "Trimite datele" button at the bottom.

## Câmpuri de tip checkbox

Sunt controale care permit bifarea sau ștergerea bifării unei căsuțe. Din punct de vedere practic, ele permit utilizatorului să marcheze una, nici una, sau mai multe opțiuni.

Controalele de acest fel se specifică prin tag-ul:

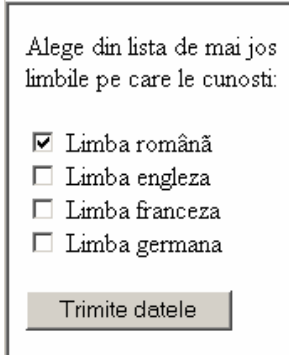
**<INPUT type="checkbox" ...>**

Atributele sale sunt:

- **name** – numele de identificare al componentei;
- **value** – valoarea pe care o va întoarce controlul respectiv;
- **checked** – dacă acest atribut este prezent, atunci controlul va fi bifat la încărcarea paginii.

Exemplu:

```
<FORM action="test.php" method="post">
Alege din lista de mai jos limbile
pe care le cunosti:<BR><BR>
<INPUT type="checkbox" name="rom" value="1" checked>
Limba română<BR>
<INPUT type="checkbox" name="eng" value="2">
Limba engleza<BR>
<INPUT type="checkbox" name="fr" value="3">
Limba franceza<BR>
<INPUT type="checkbox" name="germ" value="4">
Limba germana<BR><BR>
<INPUT type="submit" value="Trimite datele">
</FORM>
```



## Câmpuri ascunse (de tip hidden)

Aceste componente permit trimiterea de valori către server (o dată ce butonul submit a fost apăsat) fără ca acestea să fie vizibile în cadrul form-ului. Practic, aceste componente se specifică doar în cadrul codului HTML:

**<INPUT type="hidden" name="nume" value="value">**

Așa cum se observă în tag-ul de mai sus, cu ajutorul atributului **name** specificăm numele controlului, iar cu ajutorul atributului **value** specificăm valoarea care va fi trimisă către server.

## Controlul de tip TEXTAREA

Este o componentă care se utilizează pentru a introduce un text mai lung, care se poate întinde pe mai multe linii.

Tag-ul său este: **<TEXTAREA>...</TEXTAREA>**.



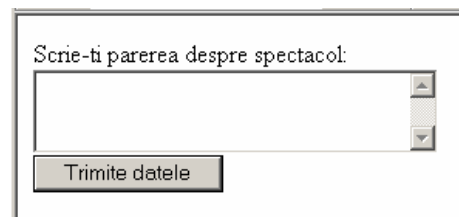
Atributele sale sunt:

- **name** – numele de identificare al componentei;
- **rows** – numărul de linii pe care se întinde componenta (implicit 2);
- **cols** – numărul de coloane pe care se întinde componenta (implicit 20);

Dacă dorim ca la încărcarea paginii să ne apară un text deja scris în cadrul controlului, acest text se va scrie între tag-ul de deschidere și cel de închidere al lui **TEXTAREA**.

Exemplu:

```
<FORM action="test.php" method="post">  
Scrie-ti parerea despre spectacol:<BR>  
<TEXTAREA rows="3" cols="30" name="parerea"></TEXTAREA>  
<BR>  
<INPUT type="submit" value="Trimite datele">  
</FORM>
```



## Controlul de tip SELECT

Acest control este utilizat pentru afișarea unei liste din care utilizatorul poate să aleagă unul sau mai multe opțiuni.

Tag-ul prin care se utilizează această componentă este **<SELECT>...</SELECT>**.

Atributele sale sunt:

- **name** – numele de identificare al componentei;
- **multiple** – dacă acest atribut este prezent, utilizatorul poate alege mai multe opțiuni din listă, ținând apăsată tasta control sau shift în timp ce dă click pe acestea.
- **size** – numărul de opțiuni care sunt afișate. Implicit este 1, în cazul listelor care nu sunt de tip multiple. În acest caz, lista se prezintă sub forma unei liste de tip drop-down;

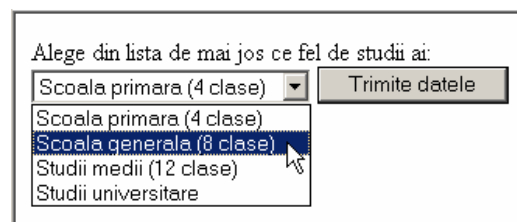
Fiecare opțiune din listă se specifică printr-un tag **<OPTION>...</OPTION>**. Acestea, la rândul lor, au următoarele atribute:

- **value** – reprezintă valoarea care va fi întoarsă de controlul SELECT în cazul în care va fi selectată opțiunea respectivă;
- **selected** – dacă acest atribut este prezent, opțiunea respectivă va fi selectată la încărcarea paginii

Textul efectiv al opțiunii se scrie între tag-ul de deschidere și cel de închidere. Tag-ul de închidere este opțional, el putând fi omis.

Iată un exemplu de folosire al controlului de tip select:

```
<FORM action="test.php" method="post">
Alege din lista de mai jos ce fel de studii ai:
<BR>
<SELECT name="studii">
<OPTION value="prim">Scoala primara (4 clase)
<OPTION value="gen">Scoala generala (8 clase)
<OPTION value="lic" selected>Studii medii (12 clase)
<OPTION value="univ">Studii universitare
</SELECT>
<INPUT type="submit" value="Trimite datele">
</FORM>
```



## 2.4. Extinderi ale limbajului HTML standard: HTML dinamic, script-uri.

Deși HTML-ul clasic permite redactarea unor documente hypertext de un nivel foarte înalt și elaborat, o dată cu evoluția limbajelor de programare vizuale, a început să devină mai puțin atractiv decât a fost la început.

Din acest motiv, a fost pus la punct ceea ce numim DHTML (Dynamic HTML) – care nu este un limbaj în sine, ci un termen prin care sunt desemnate tehnicile utilizate pentru a face paginile web cât mai dinamice și cât mai interactive.

Pe lângă HTML-ul propriu-zis, noile unelte recunoscute de browser-ele din ultima generație sunt CSS (Cascading Style Sheets), JavaScript și DOM (Document Object Model).

Scopul acestei lucrări nu este studiul amănunțit al acestora, de aceea le vom trece doar în revistă, folosind mici exemple comentate pentru fiecare dintre ele.

### 2.4.1. CSS (Cascading Style Sheets).

Noțiunea de **stil** este, pentru un document HTML, asemănătoare cu formatarea documentului, spre exemplu, pentru un document Word. Exisă o mulțime de atribute prin care se pot stabili font-urile, caracteristicile de aliniere a textului, forma elementului, culorile de fond și ale literelor, marginile, poziția elementelor, etc.

Pentru a putea gestiona cât mai eficient stilurile, a fost pus la dispoziția programatorilor de pagini web un limbaj prin care se poate realiza acest lucru. Acest limbaj este cunoscut sub numele de **CSS** (actualmente, vorbim de versiunea **CSS2**).

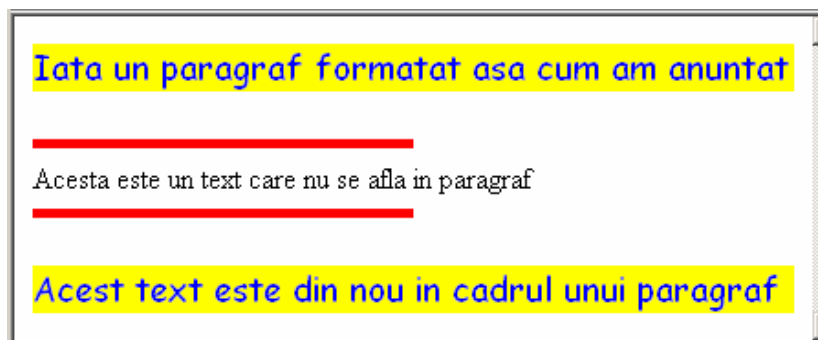
Legătura dintre HTML și CSS se realizează prin intermediul tag-ului **<STYLE>...</STYLE>** care trebuie așezat între **<HEAD>** și **</HEAD>**.

În cadrul tag-ului **STYLE** vom stabili modul în care dorim să arate elementele paginii. Fiecare element al HTML-ului pe care l-am studiat este identificat, în cadrul CSS-ului de tag-ul care îl gestionează. Spre exemplu, identificatorul CSS pentru paragrafe este P, pentru table este TABLE, pentru imagini este IMG, ș.a.m.d.

Folosind acești identificatori în cadrul unui **<STYLE>...</STYLE>**, putem face ca toate elementele de același fel din cadrul unui document să arate la fel. Astfel suntem scutiți de a scrie o grămadă de cod care s-ar repeta în cazul fiecărui element de același fel.

De exemplu, dacă dorim ca, în cadrul paginii noastre, absolut toate paragrafele să fie scrise cu fontul Comic Sans MS, caractere de 14, culoare albastră, pe fond galben, în loc de a scrie acești parametri la fiecare paragraf din document, este suficient să definim următorul **STYLE**:

```
<HTML>
<HEAD>
<TITLE>Utilizare STYLE in HEAD</TITLE>
<STYLE>
P {
    background:yellow;
    color:blue;
    font-family:"Comic Sans MS";
    font-size:14pt;
}
HR {
    text-align:left;
    width:50%;
    height:5px;
    color:red;
}
</STYLE>
</HEAD>
<BODY>
<P>Iata un paragraf formatat asa cum am anuntat</P>
<HR>
Acesta este un text care nu se afla in paragraf
<HR>
<P>Acest text este din nou in cadrul unui paragraf</P>
</BODY>
</HTML>
```



De remarcat faptul că ambele paragrafe, și de asemenea ambele linii orizontale (**HR**) din cadrul lui **BODY** nu conțin nici un fel de referință de formatare. Cu toate acestea, definițiile lui **P** și ale lui **HR** în cadrul lui **STYLE** au „predefinit” modul în care vor arăta toate paragrafele respectiv toate liniile orizontale ale documentului.

Sintaxa definiției este de felul următor: Se începe cu identificatorul elementului dorit a fi formatat (în cazul nostru **P** – tag-ul pentru paragraf, respectiv **HR**) între acolade trecându-se specificatorii de format (aceștia țin de limbajul CSS) doriți a fi modificați. În cazul de față, avem de-a face cu:

**background** = culoarea de fundal;

**color** = culoarea scrisului;

**font-family** = numele font-ului;

**font-size** = dimensiunea caracterelor;

**text-align** = alinierea în cadrul unui text;

**width** = lățimea;

**height** = înălțimea.

O altă formă de utilizarea a CSS-ului constă în definirea stilurilor cu ajutorul unor identificatori proprii, care se pot aplica ulterior unui anumit paragraf. În acest caz, în cadrul unui style putem defini proprii identificatori, precedându-i de caracterul #. Aplicarea ulterioară a lor asupra unui element, se face specificând un nou atribut, și anume `id="identificator"` unde identificator este cel propriu, definit în cadrul lui **STYLE** (cel precedat de #)

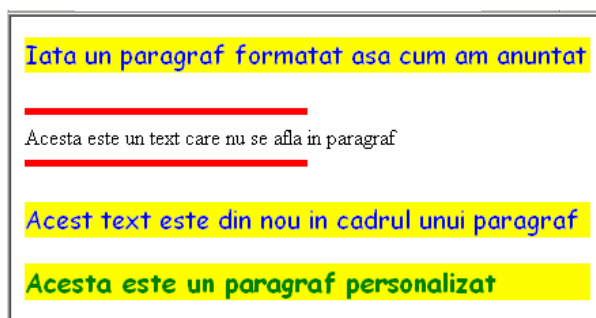
Ex: dacă inserăm în codul de mai sus secvența următoare (tot în cadrul lui **STYLE**, după definiția lui HR, adică cea scrisă cu **roșu închis**):

```
#alt_paragraf {  
    color:green;  
    font-weight:bold;  
}
```

iar înainte de **</BODY>** mai inserăm următorul paragraf:

```
<P id="alt_paragraf">Acesta este un paragraf personalizat</P>
```

vom obține următorul rezultat:



Remarcați faptul că au fost aplicați doar cei doi specificatori de format definiți în noul stil, și anume culoarea fontului și faptul că scrisul este bold. Celelalte caracteristici (font-ul și culoarea galbenă de fundal) au rămas cele definite tot în **STYLE**, în cadrul lui **P**.

În loc de a defini stilurile în cadrul antetului (**HEAD**), așa cum am arătat mai sus, ele pot fi scrise separat, într-un fișier text cu extensia .css, exact în aceeași manieră în care le-am fi scris între cele două tag-uri prezentate, **<STYLE>...</STYLE>**.

Includerea efectivă a acestui fișier în cadrul HTML-ului se face tot în secțiunea **<HEAD>**, prin intermediul următorului tag:

```
<LINK rel="stylesheet" type="text/css" href="fisier_stil.css">
```

Iată un exemplu:

1) Conținutul fișierului css, pe care l-am numit **stil.css**:

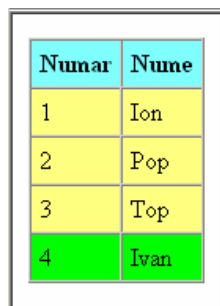
```
TABLE {  
    border-width:2px;  
    border-style:ridge;  
    border-collapse:collapse;  
}  
  
TD {  
    border-style:ridge;  
    border-width:2px;  
    padding:5px;  
}  
  
TH {  
    border-style:ridge;  
    border-width:2px;  
    background:#7ffffff;  
    padding:5px;  
}  
  
TR {  
    background:#ffff7f;  
}  
  
#TR1 {  
    background:#00ff00;  
}
```

După cum se observă, am definit în cadrul său formatele implicite pentru un tabel, rândurile și celulele sale (**TABLE**, **TR**, **TD**, **TH**) precum și un identificator propriu, **#TR1**.

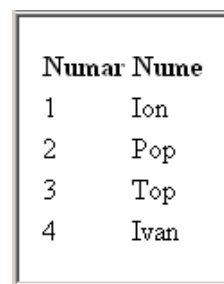
Iată și fișierul HTML care va folosi acest .css:

```
<HTML>  
<HEAD>  
    <TITLE>Utilizare css</TITLE>  
    <LINK rel="stylesheet" type="text/css" href="stil.css">  
</HEAD>  
<BODY>  
<TABLE>  
    <TR><TH>Numar<TH>Nume  
    <TR><TD>1<TD>Ion  
    <TR><TD>2<TD>Pop  
    <TR><TD>3<TD>Top  
    <TR id="TR1"><TD>4<TD>Ivan  
</TABLE>  
</BODY>  
</HTML>
```

Iată, în continuare, în partea stângă, cum arată HTML-ul, datorită includerii acestui fișier CSS, iar în partea dreaptă cum ar fi arătat același HTML, fără a specifica nici un fel de format în CSS:



| Numar | Nume |
|-------|------|
| 1     | Ion  |
| 2     | Pop  |
| 3     | Top  |
| 4     | Ivan |



| Numar | Nume |
|-------|------|
| 1     | Ion  |
| 2     | Pop  |
| 3     | Top  |
| 4     | Ivan |

## 2.4.2. JavaScript.

JavaScript este un limbaj de programare orientat pe obiecte. În ciuda numelui și a unor oarecare similarități în sintaxă, între JavaScript și Java nu există nici o legătură.

JavaScript are o sintaxă apropiată de cea a C-ului; din acest motiv un programator care cunoaște C poate cu ușurință să învețe JavaScript.

Deși acest limbaj are o plajă mai largă de extindere, cel mai des întâlnit este în scriptarea paginilor web. Programatorii web pot îngloba în paginile HTML script-uri pentru diverse activități, cum ar fi verificare datelor introduse de utilizatori, sau crearea de meniuri ori de alte efecte animate.

Browser-ele rețin în memorie o reprezentare a paginii web sub forma unui arbore de obiecte, punând aceste obiecte la dispoziția JavaScript-ului, care le poate citi și manipula. Acest arbore de obiecte, de care ne vom ocupa în paragraful următor, poartă numele de DOM (Document Object Model).

Pentru moment, vom da câteva exemple comentate de script-uri JavaScript, care nu folosesc DOM (pentru familiarizarea cu sintaxa), în cadrul unor documente HTML.

### 1) Calculul sumei cifrelor unui număr natural:

```
<HTML><HEAD>
<TITLE>JavaScript</TITLE>
</HEAD>
<BODY><HR>
<!--Vom scrie secventa de cod direct in cadrul paginii.
A se remarca faptul ca, va aparea mai intii primul HR,
se va rula codul din tag-ul SCRIPT iar apoi va aparea
cel de-al doilea HR-->
<SCRIPT language="JavaScript">
v_text=prompt("Introdu un numar intreg cu maxim 9 cifre:", "");
//functia prompt deschide o fereastră de dialog prin intermediul
//careia utilizatorul poate sa introduca date de tip string. Al doilea parametru
//(șirul vid "") reprezintă valoarea care se va găsi implicit scrisă în fereastră
//de dialog. Evident, dacă nu dorim nici o valoare implicită, se folosește șirul vid ("")
//String-ul obtinut l-am atribuit variabilei v_text
nr=parseInt(v_text); //am facut conversia de la variabila text
//la un numar intreg
s=0; //in s calculam suma cifrelor lui nr
do//procedam intocmai ca in limbajul C:
{
    r=nr%10;
    s+=r;
    nr=parseInt(nr/10); //in JavaScript impartirea NU mai respecta
    //regulile din C, deoarece operatorul / face impartire cu
    //zecimale. Pentru a obtine citul intreg, am facut conversia la
    //intreg cu acelasi parseInt
}while(nr);
alert("Suma cifrelor este "+String(s));
//functia alert(mesaj_de_tip_string) produce afisarea unei ferestre
//de dialog ce contine mesajul respectiv. A se remarca modul in care
//am concatenat mesajul cu valoarea variabilei s, convertita la string
//cu ajutorul functiei String.
</SCRIPT>
<HR>
</BODY></HTML>
```

A se remarca locul în care am pus script-ul (în cadrul paginii). În exemplele următoare nu vom mai da tot codul, ci doar secvența efectivă a script-ului.

## 2) Sortarea unui șir de numere:

```
<SCRIPT language="JavaScript">
v_text=prompt("Introdu un sir de numere pe care le separi prin spatii:", "");
x=v_text.split(" "); //functia split, aplicata lui v_text (cu parametrul " ")
//va extrage substringurile din v_text care sunt separate de spatii si va crea
//un sir de string-uri, pe care i-l atribuie variabilei x. Acestea vor fi
//x[0], x[1], ... Numarul total de elemente din sirul x se obtine prin x.length
n=x.length; //obtinem acest numar in variabila n
for(i=0; i<n; i++)
    x[i]=parseInt(x[i]); //in acest fel transformam toate componentele sirului
    //x din string-uri in intregi. In C acest lucru nu ar fi fost posibil.
//acum sortam sirul obtinut:
for(i=0; i<n-1; i++)
    for(j=i+1; j<n; j++)
        if(x[i]>x[j])
            { aux=x[i]; x[i]=x[j]; x[j]=aux; }
//si afisam sirul final. Pentru asta, formam tot mesajul de afisat intr-un string
s="Iata sirul final, sortat:\n";
for(i=0; i<n; i++)
    s=s+String(x[i])+" ";
alert(s);
</SCRIPT>
```

## 3) Descompunerea unui număr în factori primi:

```
<SCRIPT language="JavaScript">
v_text=prompt("Introdu nr. intreg pe care doresti sa-l descompui in factori primi:", "");
s="Iata descompunerea in factori primi:\n";
//pregatim string-ul in care vom afisa rezultatul final, pentru ca la acest string
//vom tot concatena noile date obtinute
n=parseInt(v_text);
f=2;
while(n!=1)
{
    p=0;
    while(n%f==0)
    {
        n=parseInt(n/f);
        p++;
    }
    if(p)
        s+="Factor="+String(f)+" putere="+String(p)+"\n";
    //fiecare nou factor si putere obtinute le concatenam la stringul
    //care va fi in final afisat
    f++;
}
alert(s);
</SCRIPT>
```

### 2.4.3. DOM (Document Object Model).

DOM reprezintă o interfață independentă față de orice limbaj de programare și platformă, care permite programelor informatice și script-urilor să aibă acces sau să actualizeze conținutul, structura sau stilurile unui document. Documentul poate fi apoi prelucrat, iar rezultatele acestor prelucrări pot fi reincorporate în document atunci când acesta este prezentat.

Înainte de standardizarea DOM-ului, fiecare navigator dispunea de propriul său model. Dacă limbajul de bază destinat manipulării documentelor web a fost repede standardizat în jurul lui JavaScript, nu același lucru se poate spune și despre funcțiile specifice de utilizat și maniera de a parcurge documentul. Cele două mari browser-e care s-au impus (Netscape Navigator și Internet Explorer) denumeau în moduri diferite o serie de componente. În practică, acest lucru obliga programatorul să scrie cel puțin două versiuni ale fiecărui script, dacă dorea ca site-ul său să fie accesibil pentru cât mai multă lume.

Prima încercare de standardizare (DOM 1) a avut loc de-abia în 1998. Ultimul nivel de standardizare (DOM 3) a avut loc în 2004.

Din punct de vedere dinamizării paginilor web, limbajul JavaScript reprezintă doar o unealtă de lucru (ați remarcat în paragraful anterior similitudinea dintre acesta și limbajul C). Pentru ca limbajul JavaScript să acționeze asupra conținutului paginii, ei bine, acest lucru îl face tocmai prin intermediul DOM.

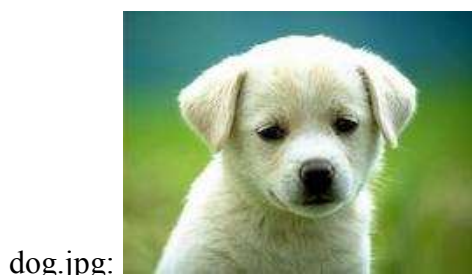
Prin intermediul DOM putem accesa orice obiect al paginii web și îl putem face să se comporte exact în felul în care dorim noi.

Ca și în cazul programării vizuale, DOM permite interceptarea anumitor evenimente (poziția mouse-ului, click-uri, etc.) și tratarea lor diferențiată.

Vom da în continuare, câteva exemple comentate, în care operăm cu JavaScript + DOM.

1) Schimbarea unei imagini atunci când trecem cu cursorul peste ea: Pentru aceasta aplicație avem nevoie de două fișiere imagine, care să fie, de preferabil, identice ca și dimensiuni, și obținute una dintr-alta printr-un procedeu de genul negativ sau trecere la alb-negru.

Iată fișierele imagine pe care am testat script-ul următor:





```

<HTML><HEAD>
<TITLE>Schimbare de imagini
</TITLE>
<SCRIPT language="JavaScript">

function schimba_negativ()
{
  //in momentul apelului, aceasta functie obtine
  //in variabila dp o referinta catre obiectul img
  //din pagina principala, gratie id-ului sau, si
  //anume 'poza'
  dp=document.getElementById("poza");
  //dupa care imaginea sursa a sa este schimbata,
  //folosind imaginea din fisierul dogneg.jpg (cea negativa)
  dp.src="dogneg.jpg";
}

function revine_normal()
{
  //exact la fel ca functia precedenta, insa
  //se foloseste alta imagine, si anume cea initiala,
  //dog.jpg
  dp=document.getElementById("poza");
  dp.src="dog.jpg";
}

</SCRIPT>
</HEAD>
<BODY>
<!--elementului img ii stabilim id-ul 'poza'
pentru a-l putea folosi apoi in cadrul script-ului
de asemenea, programam ca elementul img sa reactioneze
la cele doua evenimente:
- onmouseover (cind mouse-ul intra deasupra imaginii)
se va apela functzia care schimba imaginea originala cu
cea pe negativ
- onmouseout (cind mouse-ul iese de deasupra imaginii)
se va apela functzia care pune la loc imaginea originala-->
  <IMG src="dog.jpg" id="poza"
    onmouseover="schimba_negativ();"
    onmouseout="revine_normal();">
</BODY>
</HTML>

```

2) Schimbarea culorii de fundal a unui tabel, culoare pe care o compunem cu ajutorul a 3 valori pentru componentele R, G, B (între 0 și 255) pe care le scriem în niște zone text. Cele 3 valori le vom valida:

```

<HTML><HEAD><TITLE>Exemplu de JavaScript</TITLE>
<SCRIPT language="JavaScript">
function toHex(numar)
{
  //aceasta functie converteste numarul parametru din zecimal
  //in hexazecimal. Ne bazam ca este cuprins intre 0 si 255
  c1=parseInt(numar/16);
  c2=numar%16;
  //in c1 si c2 am obtinut cele 2 cifre hexazecimale
  if(c1>9) c1=String.fromCharCode(65+c1-10);
  //daca c1 este mai mare decit 9, o inlocuim cu litera corespunzatoare (A=10, B=11, ...,
  // F=15) adunind la codul ASCII al lui A (65) diferenta corespunzatoare. Conversia, in
  //JavaScript, de la codul ASCII la caracter se face prin
  //String.fromCharCode(codul ascii al caracterului)
  if(c2>9) c2=String.fromCharCode(65+c2-10);
  return String(c1)+String(c2);
}

function rgb(red,green,blue)
{
  //aceasta functie genereaza constanta HTML de tip culoare plecind de la valorile lui
  //red, green, blue, numere cuprinse intre 0 shi 255.Ea se foloseste de functia de mai sus
  return "#"+toHex(red)+toHex(green)+toHex(blue);
}

```

```

function coloreaza()
{
    //aceasta functie se apeleaza la apasarea butonului definit in cadrul lui BODY. In primul
    //rind testam daca valorile sunt intregi. Observati ca am folosit identificatorii dati la
    //atributul name pentru a extrage valorile din cimpurile text. Practic, prin r, g si b
    //accesam obiectele (din DOM) care se ocupa de cimpurile text. In DOM, cimpul value ne
    //intoarce taman valoarea scrisa in acestea
    nr=parseInt(r.value);ng=parseInt(g.value); nb=parseInt(b.value);
    if(nr!=r.value)//daca valoarea convertita la intreg nu coincide
    //cu cea neconvertita, inseamna ca nu este inteaga, deci dam un mesaj
        {alert("Valoarea lui r nu este corecta!");
        return;}//si iesim fortat (ca in C) cu return
    //procedam analog pentru celelalte doua
    if(ng!=g.value)
        {alert("Valoarea lui g nu este corecta!");return;}
    if(nb!=b.value)
        {alert("Valoarea lui b nu este corecta!");return;}
    //acum verificam sa fie cuprinse intre 0 si 255
    if(nr<0||nr>255)
        { alert('Valoarea lui r nu este cuprinsa intre 0 si 255');return;}
    if(ng<0||ng>255)
        { alert('Valoarea lui g nu este cuprinsa intre 0 si 255');return;}
    if(nb<0||nb>255)
        { alert('Valoarea lui b nu este cuprinsa intre 0 si 255');return;}
    //in fine, daca am trecut de aceste filtre, valorile lui r, g si b sunt corecte
    // si putem, in fine, stabili culoarea de fundal a celui alt tabel (caruia i-am dat
    //id-ul tabel) la cea pe care o obtinem din combinatia r, g, b introdusa.
    tbl=document.getElementById("tabel");
    //document este obiectul de baza al lui DOM. Functzia de mai sus, getElementById ne
    //intoarce o variabila prin intermediul careia putem accesa obiectul cu id-ul respectiv
    tbl.style.backgroundColor=rgb(nr,ng,nb);
    //apoi, prin intermediul variabilei intoarse, si anume tbl,
    //stabilim culoarea de fundal a tabelului. Pentru intoarcerea culorii
    //in formatul recunoscut de HTML, adica #RRGGBB apelam functia
    //rgb scrisa tot de noi, mai sus
}
</SCRIPT>

<BODY>
Introdu componentele de culoare (numere intre 0 si 255):<BR><BR>
<!--In tabelul de mai jos am folosit 3 input type="text" fara a ne afla
intr-un form. Este posibil si asa ceva, deoarece continutul lor
il vom prelua cu ajutorul unui script JavaScript. In cadrul acelui script
ne vom folosi de aceste controale prin intermediul atributului name
pe care l-am stabilit, deci r, g si b-->
<TABLE border="1" cellspacing="0" cellpadding="5">
    <TR><TD>Rosu<TD>
        <INPUT type="text" name="r" maxlength="3" size="3">
    <TR><TD>Verde<TD>
        <INPUT type="text" name="g" maxlength="3" size="3">
    <TR><TD>Albastru<TD>
        <INPUT type="text" name="b" maxlength="3" size="3">
</TABLE><BR>

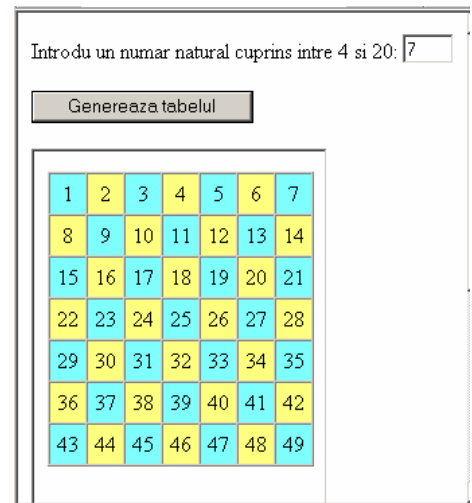
<!--Mai jos am folosit o componenta de tip BUTTON. Acestei componente i-am folosit
atributul onclick. Acestui atribut ii specificam practic ce functie JavaScript trebuie
apelata
in momentul in care se da click pe buton -->
<BUTTON onclick="coloreaza();">Testeaza</BUTTON>
<BR><BR>
<!--acestui tabel i-am utilizat atributul id, pentru a-l putea mai usor accesa
prin modelul DOM in cadrul codului JavaScript-->
<TABLE width="50%" height="50%" id="tabel" border="1">
<TR><TD align="center" valign="middle">TEST AREA
</TABLE>
</BODY></HTML>

```

3) În aplicația următoare, prin intermediul unui control de tip `input type="text"` vom cere utilizatorului să introducă un număr  $x$  între 4 și 20. Pe baza acestui număr (pe care-l validăm) vom genera, într-un element de tipul `iframe`, un tabel cu  $x$  linii și  $x$  coloane, în care punem numerele de la 1 la  $x^2$  și ale cărei celule le colorăm alternativ, la fel ca pe o tablă de șah.

Pe lângă codul sursă am pus și o captură a ferestrei, în urma rulării cu  $n=7$ :

```
<HTML><HEAD><SCRIPT language="JavaScript">
function genereaza()
{
    nr=parseInt(n.value);
    if(nr!=n.value)//verificam daca in n este un numar intreg
    {alert('Numarul introdus nu este intreg');return;}
    else if (nr<4||nr>20)//verificam si daca este intre 4 si 20
    {alert('Numarul trebuie sa fie intre 4 si 20');
        return;}
    d=document.getElementById("ifr").contentWindow.document;
    //obtinem in variabila d referinta DOM catre documentul din iframe
    d.open();//deschidem acest document pentru rescriere
    d.write('<TABLE border="1" cellspacing="0" cellpadding="5">');
    k=0;//si generam, prin script, in cadrul sau, codul HTML
    //care creeaza tabelul anuntat
    for(i=1;i<=nr;i++)
    {
        d.write('<TR>');
        for(j=1;j<=nr;j++)
        {
            d.write('<TD align="center" ');
            if((i+j)%2)//in functie de paritatea lui i+j
            //coloram intr-un fel sau intr-altul fundalul celulei
            d.write('bgcolor="#ffff7f">');
            else
            d.write('bgcolor="#7fffff">');
            ++k;
            d.write(String(k));
        }
        d.write('</TR>');
    }
    d.write('</TABLE>');
    d.close();
}
</SCRIPT></HEAD><BODY>
Introdu un numar natural cuprins intre 4 si 20:
<!--prin intermediul input type="text" scriem o valoare
care este apoi preluata de JavaScript. Acesta are numele
"n" -->
<INPUT type="text" name="n" size="2" maxlength="2">
<BR><BR>
<!--prin intermediul metodei "onclick()" a butonului
apelam functia care genereaza codul HTML al tabelului
in documentul din iframe-->
<BUTTON onclick="genereaza();">Genereaza tabelul</BUTTON>
<BR><BR>
<IFRAME name="ifr" width="70%" height="500">
</IFRAME><HR></BODY></HTML>
```



De reținut din această ultimă parte, că script-urile, deși reprezintă o automatizare și dinamizare foarte importantă a unei pagini web, nu sunt rulate pe server-ul HTML (de altfel, până în momentul de față am lucrat cu toate fișiere în mod local, ele fiind deschise automat de către browser-ul de internet) ci ele sunt rulate de către browser pe calculatorul clientului care accesează pagina ce le conține.