

Capitolul I Limbajul HTML

1.1 Istorico

În urmă cu peste 30 de ani, la sfârșitul anilor '60, Departamentul Apărării din S.U.A. a pus bazele primei rețele de calculatoare destinață să acopere o largă întindere geografică, *ARPAnet* (*Advanced Research Projects Agency Network*). Scopul rețelei era schimbul de informații cu caracter științific și tehnic între participanții la diferite proiecte de cercetare din cadrul Departamentului Apărării și cercetători din instituții de învățământ superior implicate. Rezultatele obținute în cadrul acestui proiect au condus la definirea și apoi la standardizarea unor protocoale folosite la transferul de date prin rețelele de calculatoare (*TCP/IP - Transfer Control Protocol/Internet Protocol*, *HTTP - HyperText Transfer Protocol*, *FTP - File Transfer Protocol* și altele) și a determinat în anii '70 și '80 o extindere lentă dar continuă a rețelelor de calculatoare.

Înaintea exploziei informaționale din anii '90 au mai trebuit să apară două realizări importante:

1. Dezvoltarea limbajului *HTML*, propus de Dr. Tim Berners-Lee, într-un articol publicat în 1989. Articolul propunea folosirea pentru codificarea informației vehiculate prin rețelele de calculatoare a unui subset de marcaje definite în standardul *SGML* (*Standard Generalized Markup Language*, 1986). În esență *SGML* propune un set de marcaje care permit reconstituirea documentelor difuzate pe suport electronic.
2. Apariția în februarie 1992 a primei aplicații destinate navigării pe Internet, *Mosaic*. Aplicația a fost scrisă de studenți ai *NCSA* (*National Center of Supercomputing Applications*) din cadrul universității statului Illinois, USA.

Limbajul *HTML* a făcut obiectul unor standardizări succesive, prima versiune finalizată fiind 2.0, în 1995 (autori Tim Berners-Lee și Dan Connolly).

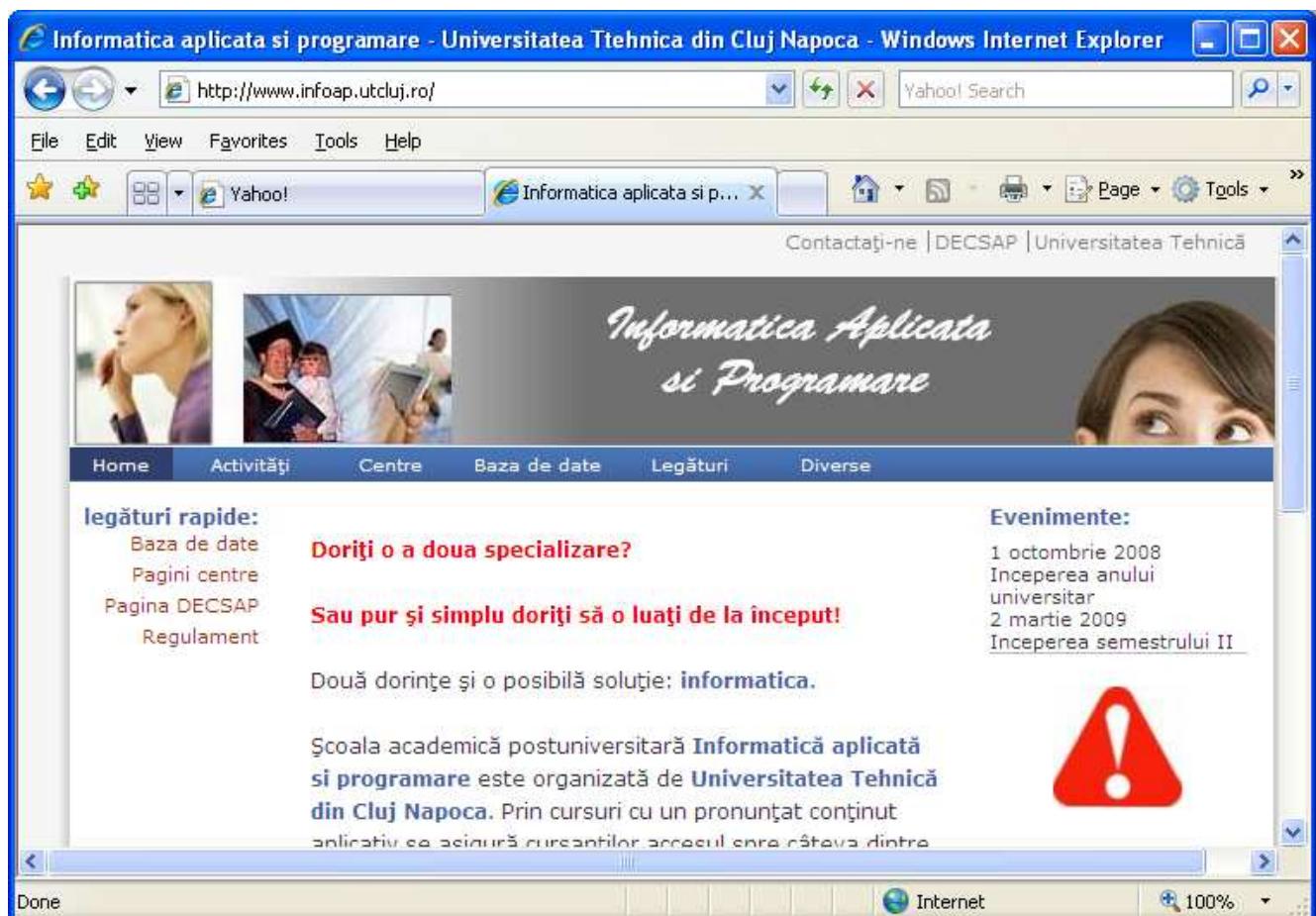
Importanța dezvoltării limbajului a condus la crearea *World Wide Web Consortium* (*W3C*), organizație care în 1997 a finalizat versiunea 3.2 care și în prezent este recunoscută ca standardul *HTML*.

Spre finele anilor '90 au apărut însă noi tehnologii, legate de realizarea paginilor *Web*: *Cascading Style Sheets (CSS)*, *Active Server Pages (ASP)*, ceea ce a determinat continuarea activității în domeniul standardizării a *W3C*, organizația realizând succesiv standardele 4.0 și 4.01. Ultima versiune este 4.1, actualmente fiind în curs de pregătire versiunea 5.

În viitorul previzibil, *HTML* va continua să reprezinte suportul pentru difuzarea de informații prin Internet.

1.2. Adresarea

Regăsirea unui fișier printre miliardele de fișiere existente în World Wide Web se bazează pe folosirea unui sistem unitar de adresare pus la punct odată cu internetul. Sistemul de adresare poartă numele de URL (*Uniform Resource Locator*). Astfel, dacă se afișează în fereastra unei aplicații de navigare pagina principală a saitului Școlii Academice Postuniversitare *Informatica Aplicată și Programare* aceasta ar putea arăta ca în figură.



Adresa paginii apare în partea superioară a ferestrei și este:

<http://www.infoap.utcluj.ro/>

Prima parte a adresei (http) precizează protocolul care trebuie folosit pentru exploatarea fișierului, în cazul dat *http* (*HyperText Transfer Protocol*) deoarece fișierul este în format *hypertext* și codificat în *html*.

A doua parte a adresei, *www.infoap.utcluj.ro* identifică un director de pe discul serverului care găzduiește saitul. Ea poate fi scrisă și folosind adresa fizică din Internet alocată serverului, care este formată din patru numere cuprinse fiecare între 0 și 255. De exemplu adresele <http://www.infoap.utcluj.ro/> și <http://193.226.7.133/infoap/> sunt echivalente dar a doua este mai greu de reținut. Punerea în corespondență a celor două modalități de indicare a directorului care

conține pagina web dorită se realizează de una dintre aplicațiile pentru DNS (*Domain Name System*) accesate de browser.

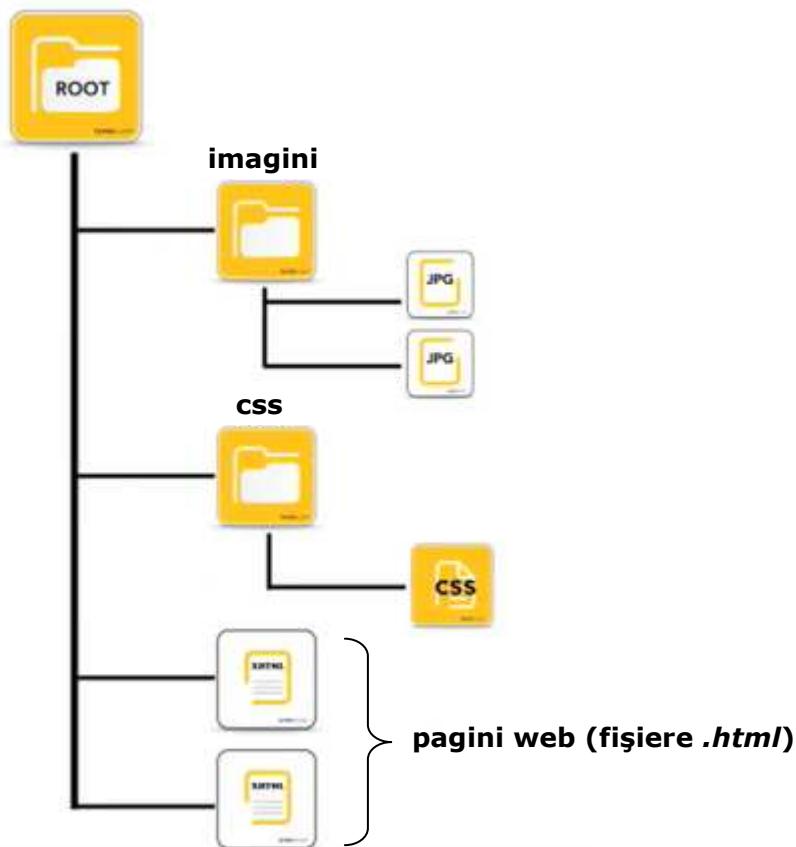
Dacă adresa indicată în browser nu mai conține și alte caractere, serverul de Web specificat va trimite solicitantului pagina principală a saitului, fișierul corespunzător fiind denumit de obicei *index.html*.

Adresa unui fișier din Internet poate fi însă mai complexă. Astfel adresa http://www.infoap.utcluj.ro/progr/bilet_ex.pdf conține după denumirea serverului o cale, */progr/* și denumirea fișierului referit, *bilet_ex.pdf*. Aceasta înseamnă că în directorul saitului există un subdirector (progr) în care este înregistrat fișierul *bilet_ex.pdf*.

Se poate deci scrie:

URL = protocol + server + cale + fișier

Crearea unui număr de subdirectoare derivate din directorul principal al saitului face întreținerea acestuia mult mai ușoară. Așa cum se va vedea în capitolele următoare, chiar și în cazul unui sait mic numărul de fișiere conținând imagini sau a altor tipuri de fișiere poate fi destul de mare, fiind indicată o structurare a lor ca în figură.



1.3 Principiile limbajului HTML

Codificarea informațiilor în HTML se bazează exclusiv pe informații de tip text (șiruri de caractere ASCII). Dacă informația propriuzisă este un text acest sistem de codificare este natural. Dacă însă informația este de altă natură (o imagine, o secvență video sau audio) documentul în format HTML va conține numele fișierului care conține informația respectivă.

Exemplu de fișier în format hypertext:

```

<HTML>
  <HEAD>
    <TITLE>Prima pagina Web</TITLE>
  </HEAD>
  <BODY bgcolor=yellow>
    <H1> NUMAI UNA </H1>
    <P>
      Pe umeri pletele-i curg rau <BR>
      Mladie ca un spic de grau, <BR>
      Cu sortul negru prins in brau, <BR>
      O pierd din ochi de draga. <BR>
      Si cand o vad, innebunesc; <BR>
      Si cand n-o vad, ma-nbolnavesc, <BR>
      Iar cand merg altii de-o petesc, <BR>
      Vin popi de ma dezleaga.
    </P>
    <IMG src="imagini/coșbuc.jpg" alt="George Coșbuc">
  </BODY>
</HTML>

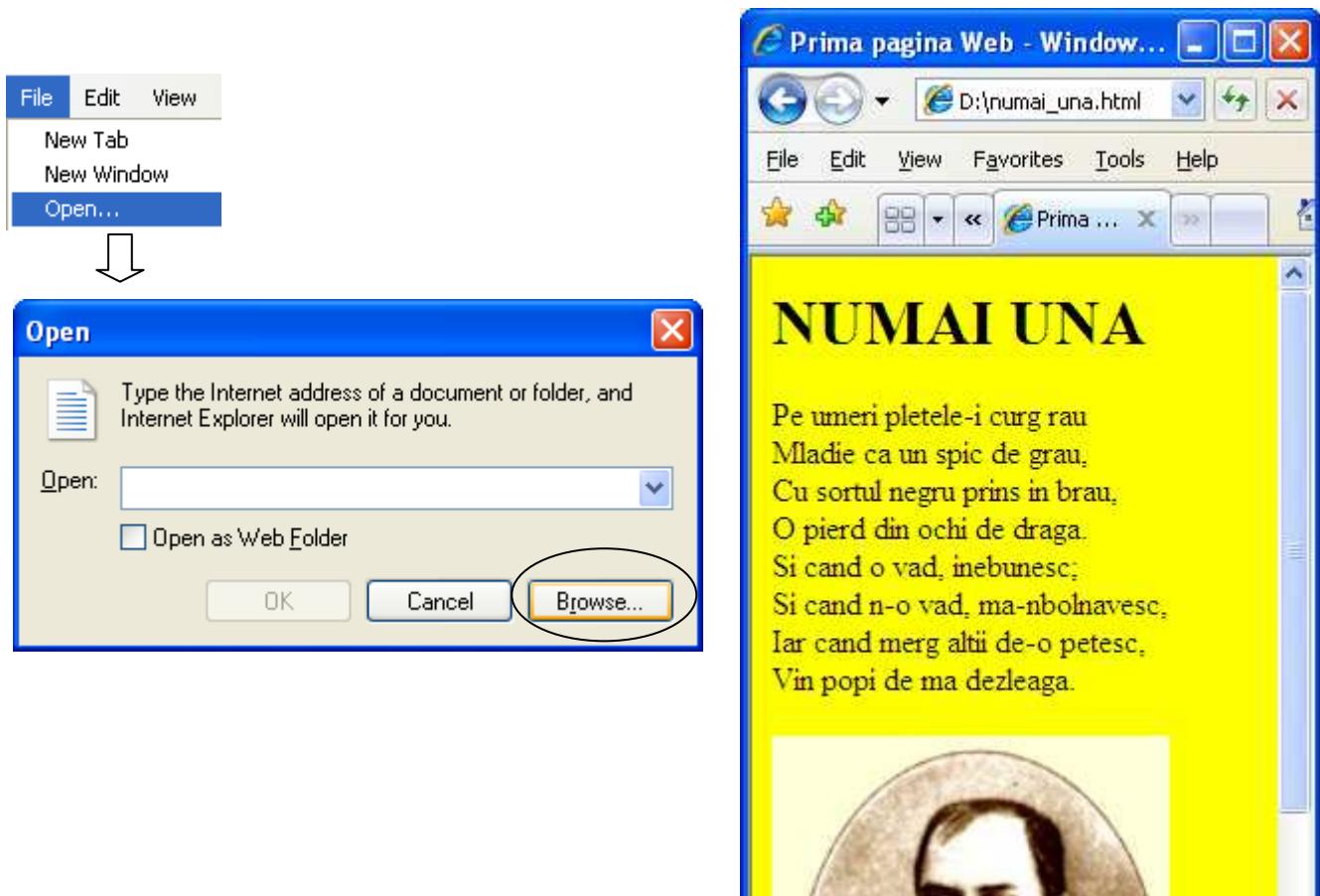
```

În fișierul dat ca exemplu <HTML> ... </HTML>, <HEAD> ... </HEAD>, <BODY> ... </BODY> și.a. sunt denumite *marcaje HTML*. Unele marcaje (<body> sau) conțin informații suplimentare ca valori ale unor *atribute* (*bgcolor* sau *src*). Analizând conținutul fișierului se observă că informația propriuzisă este încadrată între marcaje <H1> ... </H1> și <P> ... </P>. Marcajul <H1> delimită un titlu (eng. *heading*) iar <P> ... </P> delimită un paragraf (eng. *paragraph*).

În conținutul fișierului apar și marcaje
. Acestea indică trecerea la linie nouă (eng. *break*) și nu au pereche.

Pentru inserarea unei imagini s-a folosit tot un marcat fără pereche, . Acesta indică fișierul care conține imaginea. În exemplul dat atributul *src* are valoarea "imagine/coșbuc.jpg", deci fișierul *coșbuc.jpg* este situat în subdirectorul *imagine*.

După scrierea fișierului în Notepad și salvarea sub numele *numai_una.html* se poate deschide fișierul folosind de exemplu *Internet Explorer*.



1.4. XML, XHTML

Un limbaj similar limbajului HTML este XML. Spre deosebire de HTML, în XML sintaxa este mai riguroasă, marcajele apărând în mod obligatoriu în perechi. Semnificația lor este însă nedefinită, interpretarea căzând în sarcina aplicațiilor care exploatează respectivele fișiere.

Faptul că limbajele de programare actuale au clase care facilitează prelucrarea fișierelor în format XML a favorizat apariția unui limbaj nou de codificare a paginilor Web, XHTML.

Deosebirile majore dintre HTML și XHTML constau în faptul că marcajele și atributurile acestora sunt scrise cu litere mici, toate marcajele apar în perechi și valorile atributelor sunt obligatoriu cuprinse între ghilimele.

Observație: Pentru scrierea marcajelor din HTML care practic nu au nevoie de pereche se folosește o sintaxă aparte. Astfel, pentru trecerea la linie nouă în loc să se scrie `
</br>` se va scrie mai simplu `
`. La fel pentru inserarea imaginii se scrie ``.

În XHTML pagina Web analizată deja s-ar scrie astfel:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html lang="en">
```

```

<head>
    <title>Prima pagina Web</title>
    <style type="text/css">
        body { background-color: yellow; }
    </style>
</head>
<body>
    <h1> NUMAI UNA </h1>
    <p>
        Pe umeri pletele-i curg rau <br />
        Mladie ca un spic de grau, <br />
        Cu sortul negru prins in brau, <br />
        O pierd din ochi de draga. <br />
        Si cand o vad, innebunesc; <br />
        Si cand n-o vad, ma-nbolnavesc, <br />
        Iar cand merg altii de-o petesc, <br />
        Vin popi de ma dezleaga.
    </p>
    
</body> </html>

```

Fișierul astfel modificat se salvează sub același nume. Primele două linii din fișier impun browserului modul de tratare a informației conținute:

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

```

DTD (*Document Type Definition*) indică pe lângă natura informației conținute (XHTML 1.0) și setul de marcaje utilizate. În exemplul considerat s-a indicat modelul *Strict*. În loc de *Strict* se putea indica modelul *Transitional*, diferența dintre cele două fiind lista de marcaje și de atrbute utilizabile, mai restrânsă în cazul variantei *Strict*.

A doua linie face trimitere la o adresă din Internet. Fișierul referit conține definiția riguroasă a modelului *Strict*, respectiv modul în care trebuie interpretate marcajele și atrbutele acestora. În acest mod se dorește impunerea tratării uniforme, de către toate browserele, a conținutului paginilor Web.

O altă diferență notabilă a variantei XHTML față de HTML este adăugarea în secțiunea *<head>* a marcajului *<style>*. Aceasta conține indicații privind formatarea conținutului paginii web sub forma unor reguli. Pentru pagina considerată s-a inclus o singură regulă:

```

body { background-color: yellow; }

```

Ea impune pentru corpul paginii (secțiunea *<body>*) colorarea fundalului în galben. Caracteristic pentru XHTML este separarea informațiilor propriuizise de indicațiile de formatare. Astfel, în HTML pentru a scrie diferit o porțiune dintr-un paragraf am putea întâlni o succesiune de marcaje ca în exemplul următor:

```

<P>Poezia <I><B><FONT color=blue>Numai una...</FONT></B></I> a fost publicata in 1837. </P>

```

În XHTML același efect s-ar obține scriind:

```

<p>Poezia <span class="bib">Numai una...</span> a fost publicata in 1837. </p>

```

Marcajul ** face referință la clasa *bib*. Definirea ei se poate face prin adăugarea unei reguli suplimentare între *<style> ... </style>*:

```
<style type="text/css">
    body { background-color: yellow; }
    .bib {font-style: italic; font-weight: bold; color: blue}
</style>
```

A doua variantă de formatare, specifică limbajului XHTML este interesantă deoarece orice clasă astfel definită va putea fi referită în pagina Web ori de câte ori este nevoie. Așa cum se va vedea în continuare, ansamblul de reguli introdus prin `<style>` poate fi salvat într-un fișier separat, utilizarea regulilor definite putând fi imediat extinsă la toate paginile care alcătuiesc un sait Web dându-le un aspect unitar. De asemenea eliminarea din corpul paginii (secțiunea `<body>`) a detaliilor privind formatarea facilitează întreținerea conținutului paginilor.

1.3 Marcaje XHTML

1.3.1. Generalități

Având în vedere avantajele limbajului XHTML și tendința actuală de a se renunța la HTML în favoarea limbajului XHTML, în cele ce urmează vor fi prezentate doar cunoștințe de XHTML.

O pagină web este conținută între marcaje `<html> ... </html>` și este constituită de cele mai multe ori din două secțiuni, *head* și *body*.

Secțiunea *head* este delimitată prin `<head> ... </head>` iar secțiunea *body* este delimitată prin `<body> ... </body>`.

Practic majoritatea paginilor care vor fi scrise în cadrul cursului se vor încadra în structura următoare:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
    <head>
        ...(conținut head)
    </head>
    <body>
        ... (conținut body)
    </body>
</html>
```

1.3.2 Paragrafe

Ca și în cazul banal al unui document editat cu un procesor de texte, fiecare frază dintr-o pagină Web este conținută într-un paragraf. Pentru evidențierea unui paragraf se folosește perechea de marcaje `<p> ... </p>`.

Exemplu:

<p>Primul paragraf poate conține informații generale. Următoarele paragrafe vor aduce precizări suplimentare, în final mesajul transmis fiind limpede. </p>

Dispunerea conținutului paragrafului este lăsată la latitudinea aplicației de navigare. Dacă în cadrul unui paragraf trebuie totuși trecut forțat la linie nouă, în locul în care se dorește ruperea paragrafului se va folosi un marcat
.

Exemplu:

<p>

Pe umeri pletele-i curg rau

Mladie ca un spic de grau,

....

</p>

În unele cazuri se dorește ca browserul să nu separe pe linii consecutive două șiruri de caractere între care există spațiu. Este cazul numerelor de telefon de exemplu. Pentru astfel de situații se poate încadra șirul care nu trebuie separat între marcaje <nobr> ... </nobr>.

Exemplu:

Sunăți la <nobr> 0745 225236 </nobr> pentru informații suplimentare.

Pentru scrierea caracterelor specifice limbii române este necesară includerea în secțiunea *head* a marcajului:

<meta http-equiv="content-type content="text/html; charset=windows-1252" />

diacriticile putând fi apoi codificate folosind următoarele succesiuni de caractere:

Simbol	Cod	Simbol	Cod
Ș	Ş	Ț	Ţ
ș	ş	ț	ţ
Ă	Ă	Î	Î
ă	ă	î	î
Â	Â		
â	â		

Și pentru alte caractere speciale, de exemplu pentru cele folosite la scrierea marcajelor, trebuie folosite sevențe de caractere:

&nbsp -	pentru caracterul spațiu
< -	pentru <
> -	pentru >
& -	pentru &

Pentru a evidenția părți dintr-un paragraf se folosesc marcajele ... și Primul marcat este interpretat de browser ca

indicând scrierea înclinată (italic) iar al doilea este interpretat ca indicând o scriere cu caractere îngroșate (bold).

1.3.3 Titluri

Titlurile din paginile web sunt încadrate între marcaje `<h> ... </h>`. Ca și în cazul procesoarelor de texte se pot folosi marcaje diferite pentru titluri de diferite nivele : `h1, h2, ..., h6`. În lipsa unor specificații privind mărimea caracterelor, pentru nivelul 4 textul va fi afișat normal. Mărimea caracterelor pentru nivelele 5 și 6 va fi în acest caz mai mică decât cea normală iar pentru 1-3 va fi mai mare.

Exemplu:

```
<h2>Programul de lucru cu publicul </h2>
```

1.3.4 Imagini

Pentru includerea în pagina web a unei imagini se folosește marcajul ``. Acesta acceptă atributele `src` pentru indicarea fișierului care conține imaginea și `alt` pentru definirea textului care va apărea în locul imaginii dacă afișarea imaginilor în fereastra browserului este inactivată sau la plasarea cursorului mouse-ului deasupra imaginii.

Exemplu:

```

```

Din punctul de vedere al programului de navigare, o imagine este un simbol ca și oricare dintre caracterele afișate în pagină. Față de text, imaginile necesită un timp de încărcare mai îndelungat, ceea ce impune limitarea mărimii și a numărului de imagini din paginile web.

Imaginiile pot proveni din diferite surse : alte pagini web, scanate sau desenate cu ajutorul unei aplicații specializate.

Rezoluția imaginii, cuprinsă ușual între 72 *DPI* (*dots per inch*, puncte pe inci) și 600 *DPI*, influențează mărimea fișierului care conține imaginea. Monitoarele uzuale afișează punctele la 0.23-0.26 mm, deci au o rezoluție de aproximativ 96 *DPI*. Dacă imaginile nu trebuie ulterior imprimate și nu se dorește modificarea mărimii lor în momentul afișării în fereastra browserului, se recomandă scanarea folosind această rezoluție.

Imaginiile care trebuie afișate în paginile web sunt păstrate în fișiere separate. Deoarece memorarea informației grafice se poate realiza folosind o multitudine de tehnologii, prin convenție formatul fișierelor destinate afișării în cadrul paginilor web trebuie să fie *GIF* (*Graphics Interchange Format*, având extensia `.gif`, formatul suportând 256 culori), *JPEG* (*Joint Photographic Experts Group*, având extensia `.jpg`, formatul suportând $2^{24} = 16777216$ culori) sau *PNG* (*Portable Network Graphics*, variantă ameliorată a formatului *GIF* având extensia `.png`).

Dacă aceeași imagine este folosită pe mai multe pagini, programul de navigare o va încărca și memora temporar pe disc reducând timpii de încărcare a paginilor.

Adaptarea unei imagini la cerințele unui proiect presupune în principal decuparea zonei interesante și redimensionarea acesteia în corespondență cu spațiul alocat în pagină.

Există o multitudine de aplicații care permit manipularea imaginilor. Cele mai utilizate sunt următoarele: Adobe Photoshop, Corel Photo-Paint, Corel Paint Shop Pro, The GIMP, Inkscape, Pixel image editor, Paint.NET sau Xara.

Alegerea unei aplicații potrivite pentru realizarea activităților legate de web design este simplă dacă se urmărește minimizarea costurilor deoarece numai *Inkscape*, The Gimp și *Paint.NET* sunt gratuite. În cadrul cursului, prelucrările de imagini vor fi realizate cu *Inkscape* (<http://www.inkscape.org/>). Un tutorial succint din care se poate învăța folosirea aplicației poate fi accesat la adresa:

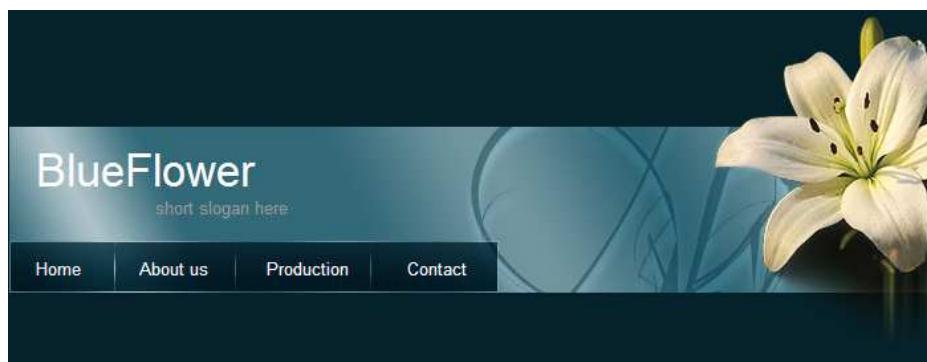
<http://en.flossmanuals.net/Inkscape/>.

1.3.5 Mic tutorial Inkscape

Din punct de vedere al realizării grafice, un sait trece prin două faze.

În prima fază se realizează un "layout". Practic este vorba despre un fișier în care informația este dispusă pe mai multe nivele, la bază fiind nivelul care conține un fundal. Nivelele conțin text, alte mici imagini (de exemplu fundalul unor butoane) sau dreptunghiuri pline care delimită exact zona în care va fi afișată o prezentare (video, succesiune de imagini etc.). Pentru această etapă se folosește o aplicație specializată pe informație grafică deoarece trebuie depasate și redimensionate imagini, trebuie create diferite efecte și construite imaginile care vor fi folosite ca fundal.

Exemplu:



După încheierea acestei etape rezultatul este prezentat firmei care a comandat lucrarea în vederea obținerii aprobării pentru continuare pe soluția propusă.

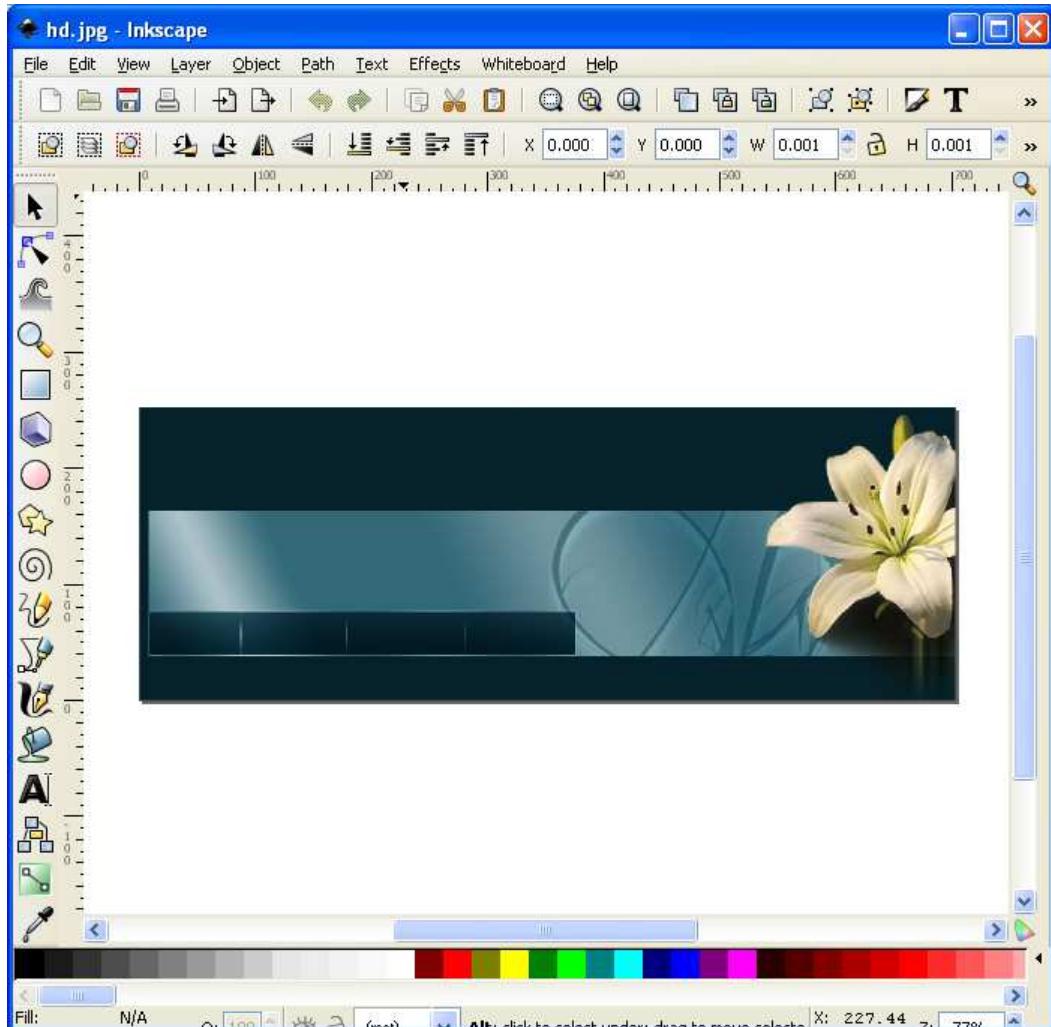
Recomandare: Pentru reușita unui proiect este bine să fie studiate cât mai multe realizări înrudite, o sursă importantă de inspirație pentru graficieni și nu numai fiind <http://www.templatemonster.com>.

A doua fază, de decupare (engl. *slicing*), constă în divizarea imaginii obținută ca fundal în prima etapă în imagini mai mici a căror alăturare permite refacerea fundalului inițial. Aceste imagini vor constitui fundalul unor blocuri care, în etapa de integrare, vor primi informația efectivă (text, imagini, player video, galerie de imagini în JavaScript sau Flash, etc.).

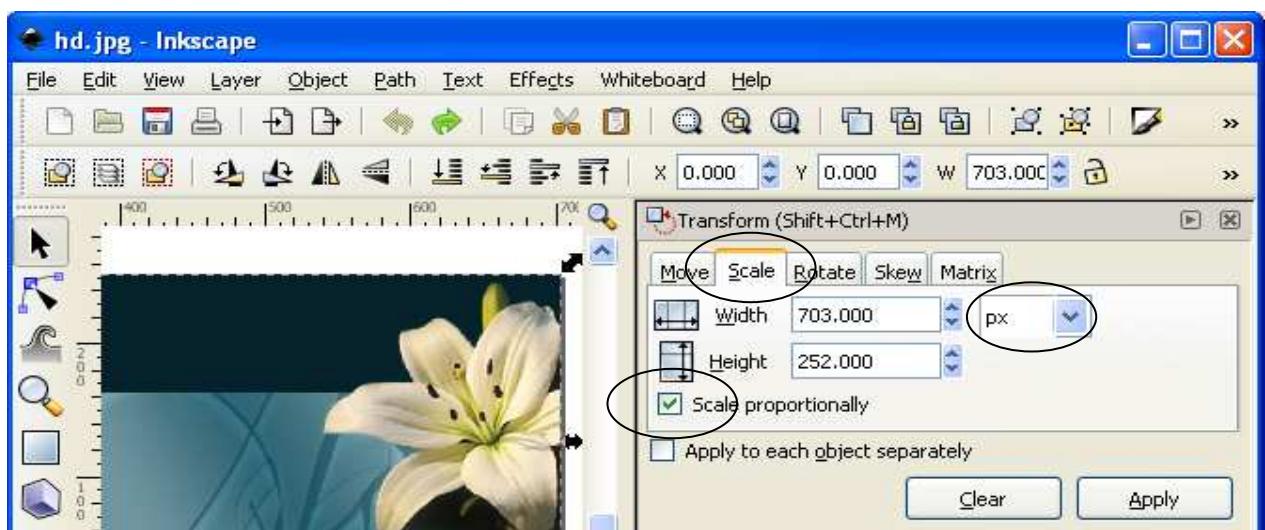
În această etapă se ascund informațiile inițial introduse în layout rămânând numai fundalul paginii. Apoi se decupează blocurile de imagine dorite și se salvează fiecare bloc într-un fișier separat. Aceste fișiere vor primi denumiri sugestive pentru a putea fi ușor identificate și integrate în blocurile care vor fi definite în etapa următoare (etapa zisă "*de integrare*").

a. Redimensionarea unei imagini

Se pornește aplicația *Inkscape*, se deschide fișierul care conține imaginea (*File / Open*) și se selectează fișierul dorit.



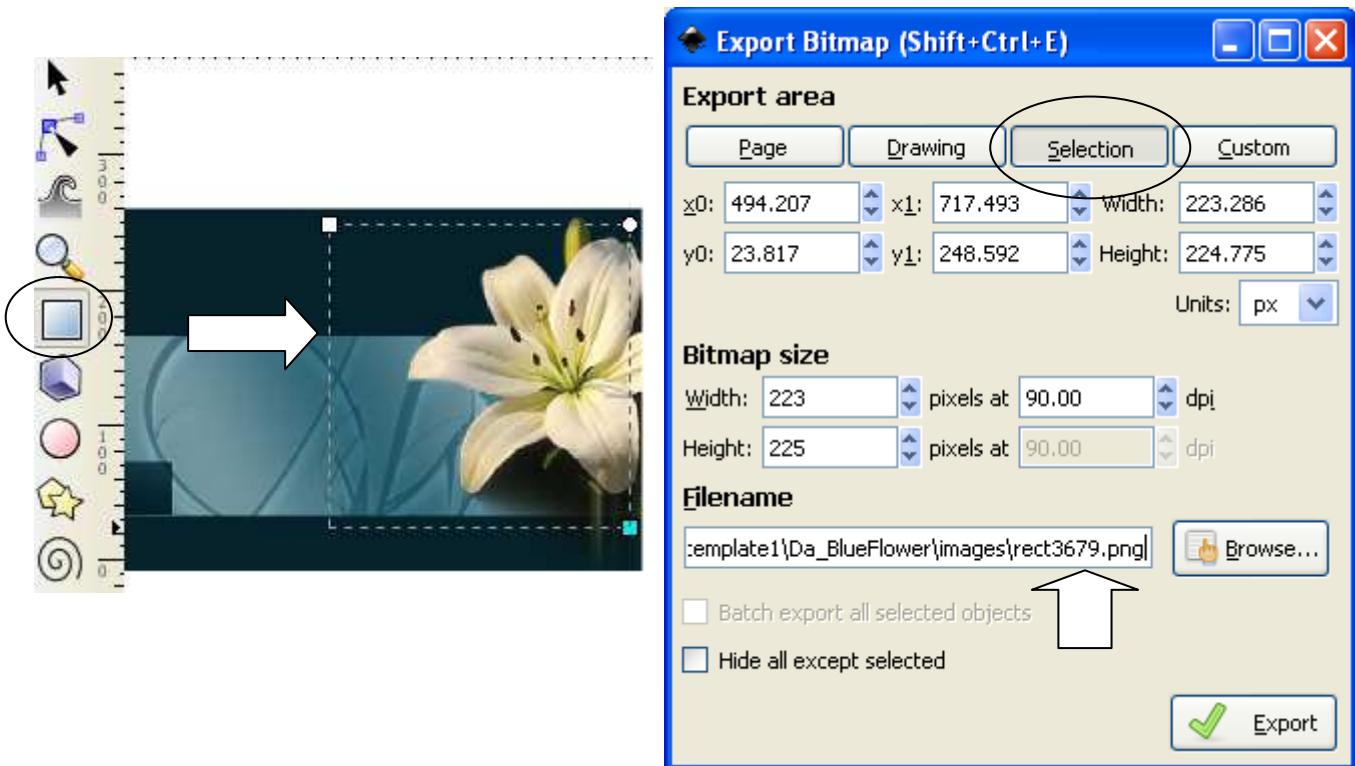
Dacă fișierul conține o imagine se selectează imaginea și apoi se selectează în meniu aplicației *Object / Transform*. Aplicația va afișa o fereastră în care se selectează tabul de selecție *Scale*. De obicei dimensiunile imaginilor sunt date în pixeli (punkte ecran) și din acest motiv se selectează ca și unitate de măsură *px*.



b. Decuparea unei zone dreptunghiulare

Dacă se dorește decuparea unei porțiuni dreptunghiulare dintr-o imagine, se va desena un dreptunghi care încadrează regiunea dorită și apoi se va selecta *File / Export Bitmap* și se va preciza fișierul în care se va păstra noua imagine. Fișierul va fi în format .png.

Exemplu:



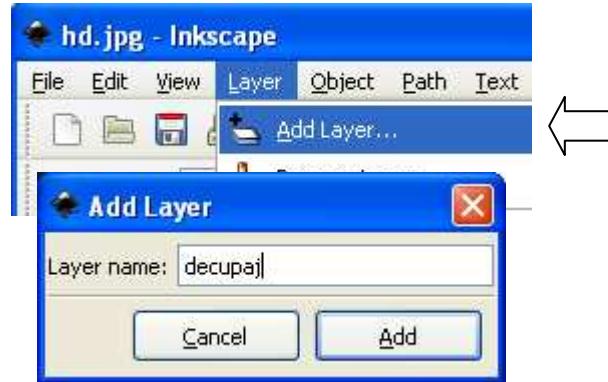
Numele fișierului .png (rect3679) poate fi editat. Numele inițial afișat a fost stabilit folosind denumirea internă a obiectului (dreptunghi) folosit la selecție.

c. Slicing (decuparea imaginii de fundal)

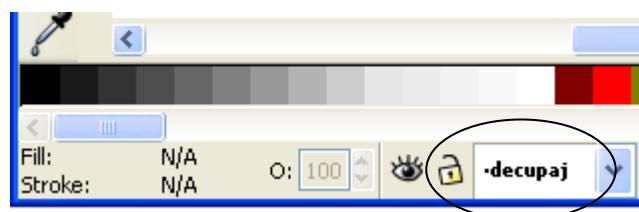
Începerea lucrului la un sait presupune frecvent activități de decupare a unei imagini pe care un grafician a plasat-o ca fundal al saitului. În urma decupării imaginii folosite ca fundal vor fi obținute mai multe imagini care vor fi ulterior aplicate ca fundal blocurilor care compun pagina Web și care vor conține informația propriu-zisă (text, alte imagini etc). Numele consacrat al acestei activități este *slicing* (*slice = felie*).

Deoarece în HTML blocurile care formează pagina sunt dreptunghiulare, ne interesează doar decuparea în această formă.

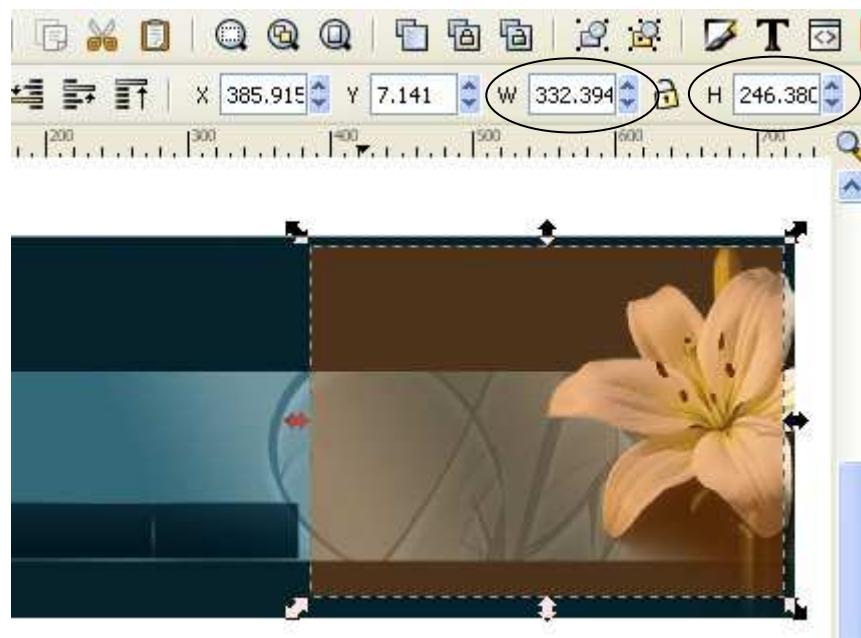
Pentru a realiza decuparea unei imagini în porțiuni mai mici se vor suprapune peste imagine o serie de dreptunghiuri care vor fi utilizate în final pentru tăierea porțiunilor dorite. Pentru a realiza toate dreptunghiurile înaintea decupării propriu-zise, activitatea va începe prin declararea unui nou nivel (*layer*) care va conține doar dreptunghiuri de selecție.



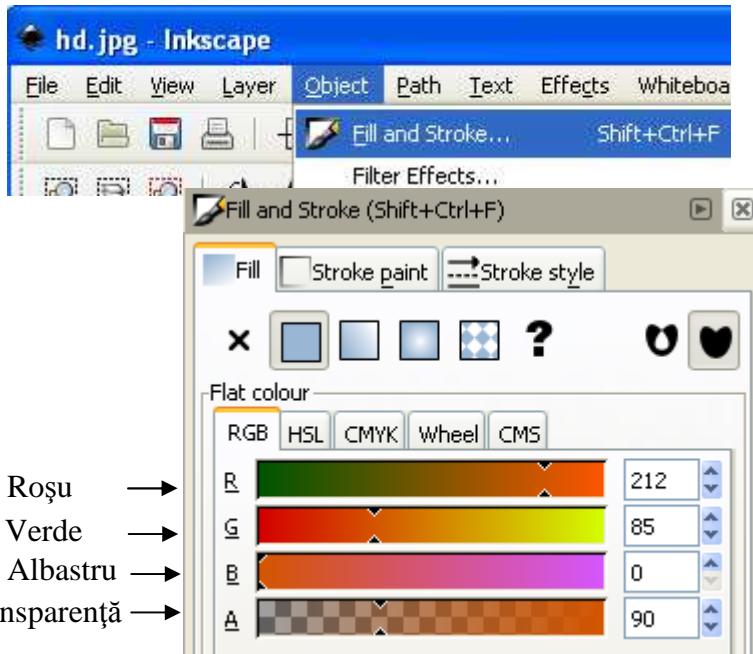
Noul nivel este imediat impus ca nivel curent. Numele nivelului curent este afișat permanent în partea de jos a ferestrei aplicației, în bara de stare.



În noul nivel se desenează un dreptunghi suprapus peste una dintre zonele care trebuie decupată. Dimensiunile dreptunghiului pot fi adaptate cu mouse-ul sau pot fi modificate direct, schimbând valorile afișate pe bara cu instrumente.



Pentru a modifica aspectul dreptunghiului (culoare, transparentă) se selectează în meniu *Object / Fill and Stroke* sau se apasă butonul de pe bara cu instrumente.

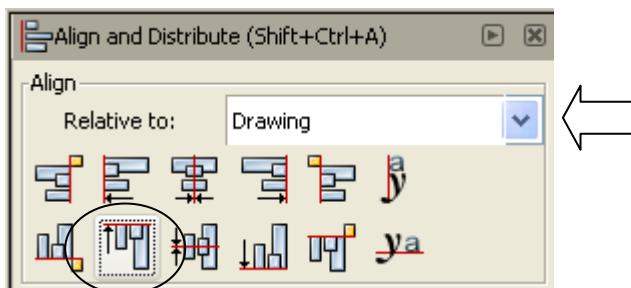
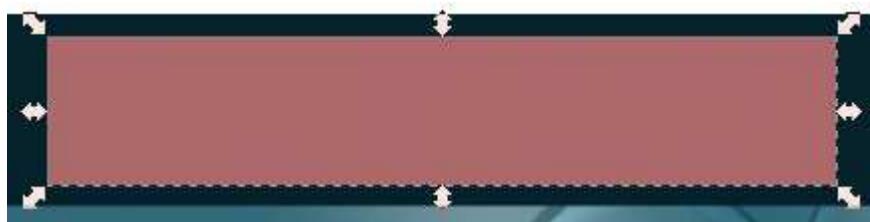


În procesul de decupare vor fi realizate evident mai multe dreptunghiuri. Din considerente legate de ușurința manipulării acestora este bine să fie declarate parțial transparente și colorate într-o culoare distinctă, deși înaintea folosirii lor pentru decuparea zonelor dorite acestea vor fi declarate din nou ca fiind complet transparente. Nivelul de transparentă se impune prin depalasarea ultimului cursor din fereastră, A (prescurtare de la *alpha channel*, denumirea consacrată a componentei transparentă). Transparentă poate varia între 0 (complet transparent) și 255 (complet opac).

O funcție interesantă a aplicației Inkscape este posibilitatea impunerii unei alinieri perfecte a unui dreptunghi astfel desenat față de oricare dintre laturile imaginii inițiale sau față de alt dreptunghi.

Pentru primul caz, alinierea față de o latură a imaginii inițiale, se afișează fereastra *Align and Distribute* (din meniu *Object / Align and Distribute* sau butonul  de pe bara cu instrumente) și se selectează dreptunghiul (butonul ).

Situată inițială:



Final:

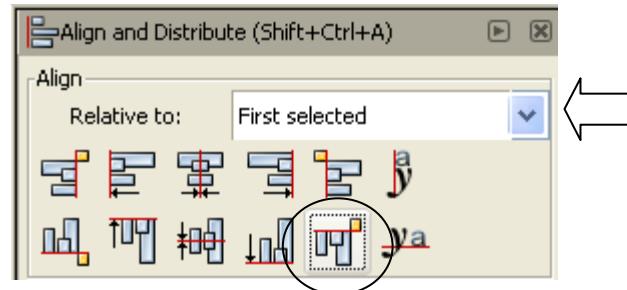


Pentru a realiza o aliniere perfectă a laturilor corespunzătoare în cazul în care sunt realizate mai multe dreptunghiuri de selecție, fereastra *Align and Distribute* conține butoane care permit poziționarea unui dreptunghi față de alt dreptunghi, construit anterior și care este identificat implicit ca referință. Pentru a realiza o astfel de aliniere se selectează cu mouse-ul ambele dreptunghiuri (dreptunghiul curent și referință) și se apasă butonul (*Align tops of objects to bottom of anchor*).

Situată inițială:



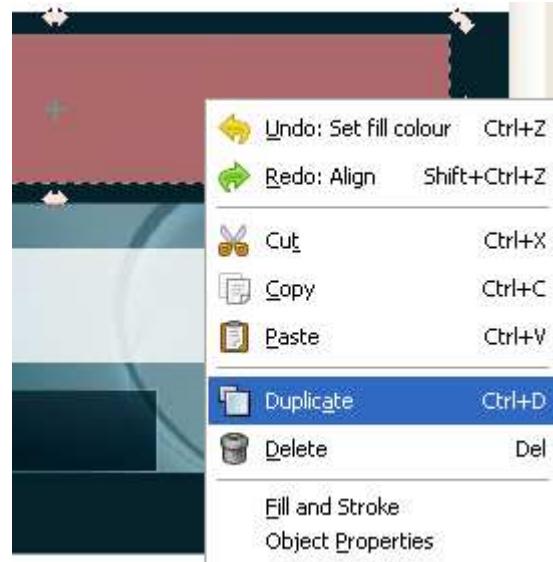
Comandă aliniere:



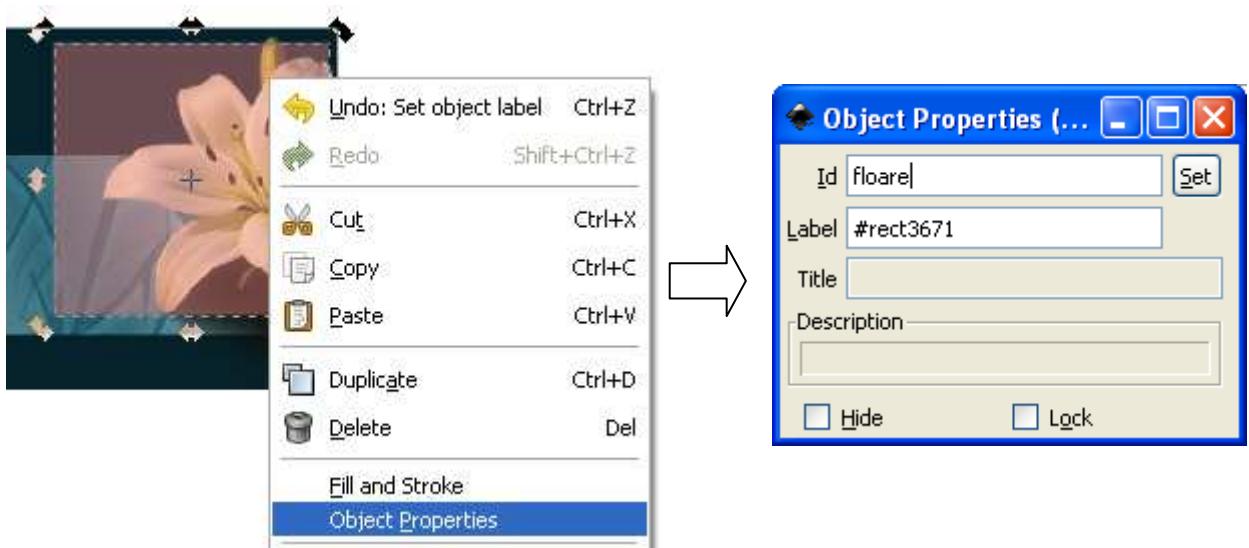
Final:



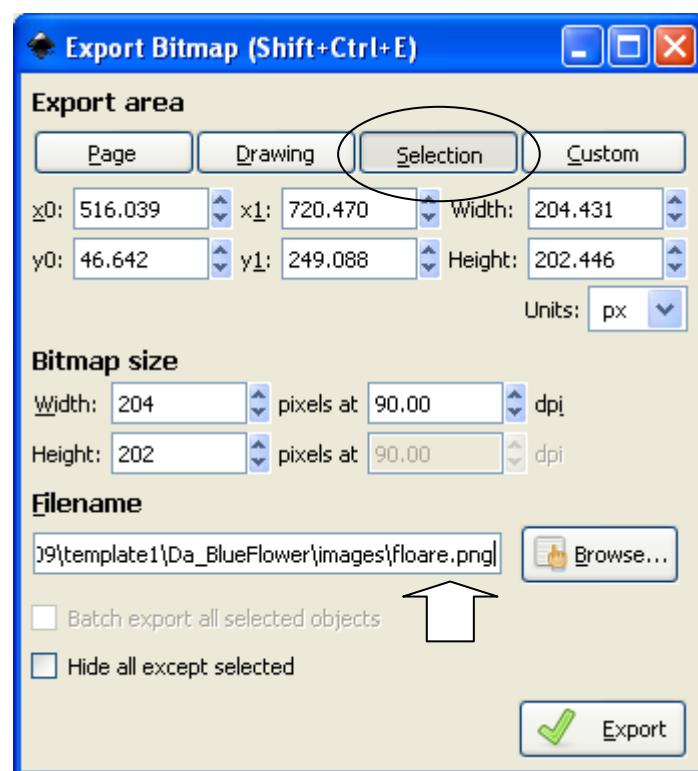
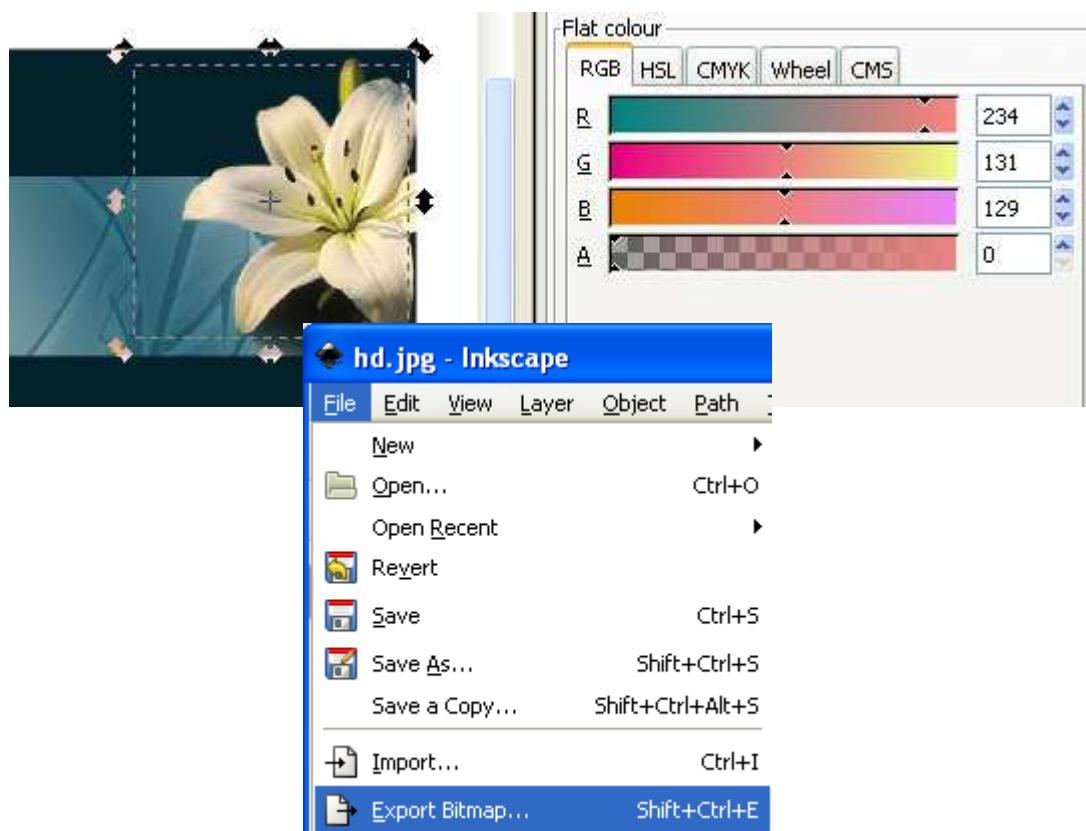
În procesul de decupare este necesară frecvent duplicarea unui dreptunghi deja realizat. Pentru aceasta se selectează cu butonul drept dreptunghiul realizat și apoi se selectează în meniul contextual afișat opțiunea *Duplicate*. Noul dreptunghi va fi realizat peste cel de referință, deci trebuie imediat deplasat cu mouse-ul și apoi redimensionat și aliniat. Dimensiunile finale ale dreptunghiurilor de selecție trebuie notate deoarece blocurile în care acestea vor fi plasate ca imagini de fundal trebuie să aibă aceleași dimensiuni.



În meniul contextual afișat la selectarea unui dreptunghi se află și opțiunea *Object properties*. În urma selectării acestei opțiuni se afișează o fereastră în care se poate introduce un nume pentru dreptunghiul selectat. Acest nume va fi păstrat și pentru fișierul care va conține porțiunea din imagine decupată cu ajutorul dreptunghiului respectiv.



Se reduce parametrul A la 0:



Se observă că numele fișierului coincide cu cel al dreptunghiului, formatul implicit fiind evident tot .png.

1.3.6 Liste

Marcajele HTML permit definirea mai multor tipuri de liste.

1.3.6.1 Lista ordonată

Lista ordonată este declarată prin marcajul `` (*ordered list*) și are rândurile numerotate. Fiecare linie este încadrată între marcaje ` ... `.

Exemplu

```
<ol>
  <li>Introducere </li>
  <li>Limbajul HTML </li>
  <li>Sistemul de operare UNIX </li>
  <li>Publicarea saitului pe un server UNIX </li>
</ol>
```

- 1. Introducere
- 2. Limbajul HTML
- 3. Sistemul de operare UNIX
- 4. Publicarea saitului pe un server UNIX

Numerotarea liniilor este automată. Pentru a începe numerotarea de la o valoare diferită de 1, marcajului `` i se va adăuga atributul `start=n`, n fiind numărul dorit pentru prima linie a listei:

```
<ol start="4">
  <li> Introducere </li>
  ...
</ol>
```

- 4. Introducere
- 5. Limbajul HTML
- 6. Sistemul de operare UNIX
- 7. Publicarea saitului pe un server UNIX

1.3.6.2 Lista neordonată

Lista neordonată este declarată prin marcajul `` (*unordered list*) și are rândurile precedate de puncte. Fiecare linie este încadrată de marcaje ` ... `.

Exemplu:

```
<ul>
  <li>Introducere </li>
  <li>Limbajul HTML </li>
  <li>Sistemul de operare UNIX </li>
  <li>Publicarea saitului pe un server UNIX </li>
</ul>
```

- Introducere
- Limbajul HTML
- Sistemul de operare UNIX
- Publicarea saitului pe un server UNIX

1.3.6.3 Lista conținând definiții

Lista conținând definiții este declarată prin marcajul `<dl>` (*definition list*) și are rândurile precedate de marcaje `<dt>` pentru titluri și `<dd>` pentru definiții.

```
<h2> Definitii</h2>
<dl>
  <dt> Pagină Web </dt>
    <dd> Fișier (document) aparținând WWW. </dd>
  <dt> Server Web </dt>
    <dd> Calculator care memorează pagini web și le pune la dispoziția utilizatorilor rețelei. </dd>
  <dt> Sait Web </dt>
    <dd> O colecție de pagini Web întreținută de o firmă, o instituție de învățământ, o agenție guvernamentală sau chiar de un individ. </dd>
</dl>
```

Definitii

Pagină Web

Fișier (document) aparținând WWW.

Server Web

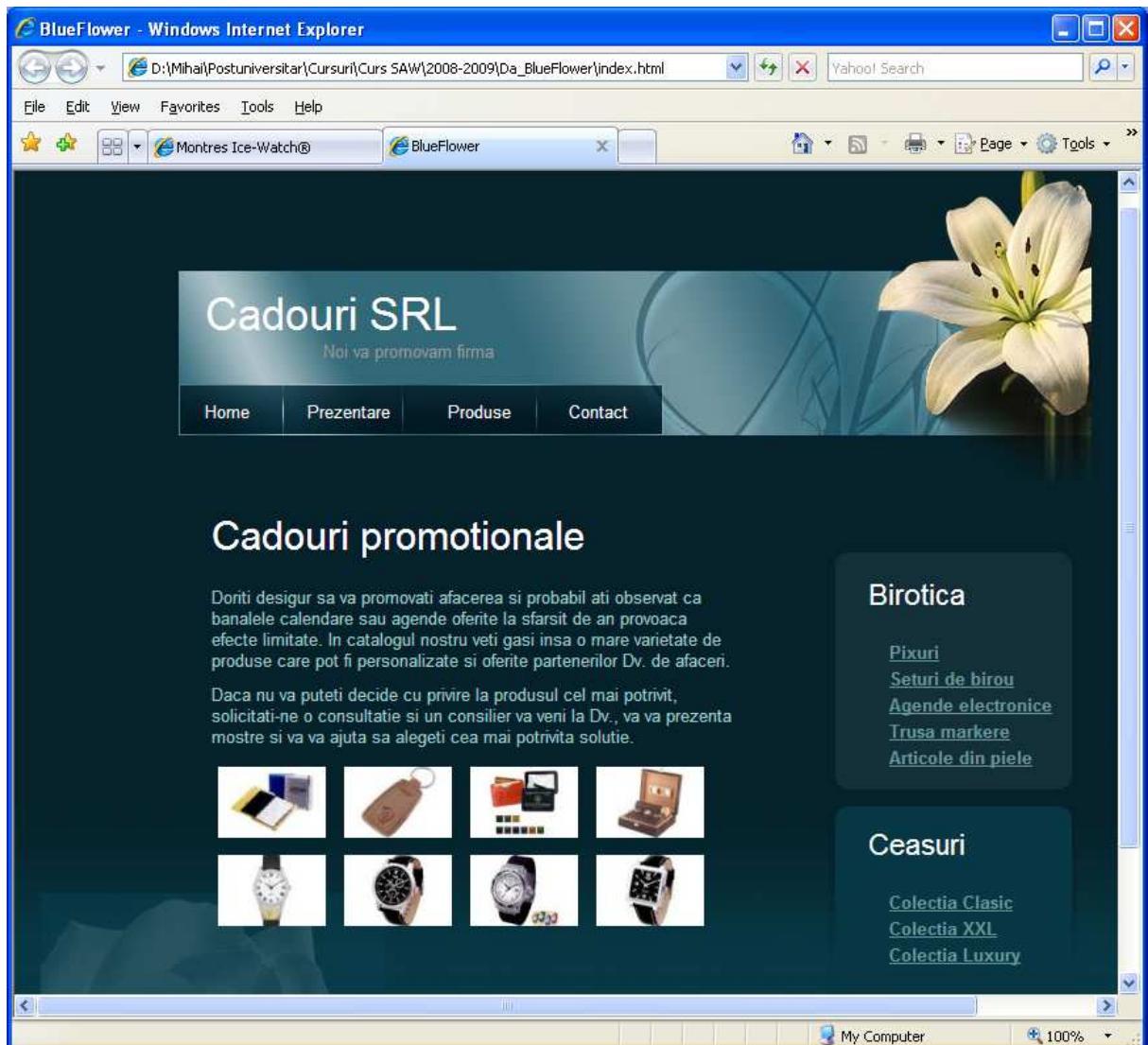
Calculator care memorează pagini web și le pune la dispoziția utilizatorilor rețelei.

Sait Web

O colecție de pagini Web întreținută de o firmă, o instituție de învățământ, o agenție guvernamentală sau chiar de un individ.

Exemplul 1:

Realizați prima pagină a saitului firmei *Cadouri SRL*:



Textul se va introduce conform specificațiilor de mai jos.

Cadouri SRL

Noi va promovam firma

<h1>

<p>

- Home
- Prezentare
- Produse
- Contact

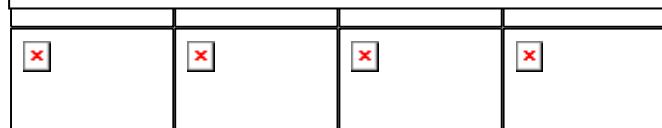
Cadouri promotionale

<h2>

Doriti desigur sa va promovati afacerea si probabil ati observat ca banalele calendare sau agende oferite la sfarsit de an provoaca efecte limitate. In catalogul nostru veti gasi insa o mare varietate de produse care pot fi

<p>

Daca nu va puteti decide cu privire la produsul cel mai potrivit, solicitati-ne o consultatie si un consilier va veni la Dv., va va prezenta mostre si va va ajuta sa alegeti cea mai potrivita solutie.



Birotica

<h2>

- Pixuri
- Seturi de birou
- Agende electronice
- Trusa markere
- Articole din piele

Ceasuri

<h2>

- Colectia Clasic
- Colectia XXL
- Colectia Luxury

[Home](#) | [Prezentare](#) | [Produse](#) | [Contact](#)

<p>

<p>

© 2008. Toate drepturile rezervate. Proiectat de INFOAP.

Rezolvare:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

```
<html>
<head>
<title>Cadouri SRL</title>
</head>
<body>
    <h1>Cadouri SRL</h1>
    <p>Noi va promovam firma</p>
    <ul>
        <li>Home</li>
        <li>Prezentare</li>
        <li>Produse</li>
```

```
<li>Contact</li>
</ul>
<h2>Cadouri promotionale</h2>
<p>Doriti desigur sa va promovati afacerea si probabil ati observat ca banalele calendare sau agende oferite la sfarsit de an provoaca efecte limitate. In catalogul nostru veti gasi insa o mare varietate de produse care pot fi personalizate si oferite partenerilor Dv. de afaceri. </p>
<p>Daca nu va puteti decide cu privire la produsul cel mai potrivit, solicitati-ne o consultatie si un consilier va veni la Dv., va va prezenta mostre si va va ajuta sa alegeti cea mai potrivita solutie.</p>



<br />




<h2>Birotonica</h2>
<ul>
<li>Pixuri </li>
<li>Seturi de birou </li>
<li>Agende electronice </li>
<li>Trusa markere </li>
<li>Articole din piele </li>
</ul>
<h2>Ceasuri</h2>
<ul>
<li>Colectia Clasic </li>
<li>Colectia XXL </li>
<li>Colectia Luxury </li>
</ul>
<p>Home | Prezentare | Produse | Contact</p>
<p>© 2009. Toate drepturile rezervate. Proiectat de INFOAP. </p>
</body>
</html>
```

1.3.7 Tabele

Tabelele sunt folosite în paginile web pentru prezentarea informațiilor organizate ca tabele. Tabelele pot servi și la poziționarea informației dintr-o pagină web, dar această soluție a fost practic abandonată odată cu extinderea utilizării CSS (Cascade Style Sheet).

Un tabel este format din *rânduri de celule*. Principalele marcaje folosite la descrierea unui tabel sunt:

`<table>...</table>` - pentru delimitarea tabelului;

`<tr>...</tr>` - pentru delimitarea unei linii (*table row*)

`<td>...</td>` - pentru delimitarea unei celule din cadrul tabelului (*table data*)

`<th>...</th>` - pentru celulele din capul tabelului (*table header* - determină scrierea implicită bold și centrat).

Formatul general al unui tabel este:

```

<table> (începe definirea tabelului)
<caption> titlul, dacă există </caption>
<tr> (începe definirea primei linii)
  <th> conținut </th> (prima celulă din linia 1, titlul coloanei)
  ...
  <th> conținut </th> (ultima celulă din linia 1, titlu)
</tr> (sfârșitul primei linii)
<tr> (începe definirea liniei 2)
  <td> conținutul primei celule </td>
  ...
  <td> conținutul ultimei celule </td>
</tr> (sfârșitul liniei 2)
.....
<tr> (începutul ultimei linii)
  <td> conținutul primei celule </td>
  ...
  <td> conținutul ultimei celule </td>
</tr> (sfârșitul ultimei linii)
</table> (sfârșitul tabelului)
```

Includerea unei imagini într-o celulă a tabelului se face normal, incluzând între marcajele `<td>...</td>` marcajul ``:

`<td></td>`

Exemplu:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
```

```

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
<head>
<title>Tabele</title>
</head>

<body>
<h1>Exemplu</h1>
<table>
<tr>
    <th>Produs</th>
    <th>Pret</th>
    <th>Cantitate disponibila</th>
</tr>
<tr>
    <td>Pantofi Lux</td>
    <td>156</td>
    <td>243</td>
</tr>
<tr>
    <td>Pantofi Carmens</td>
    <td>174</td>
    <td>189</td>
</tr>
</table>
</body>
</html>

```



Două atribute care pot fi incluse în marcajele `<td>` sau `<th>` permit unirea celulelor.

a. colspan="n" - permite unirea a *n* celule consecutive de pe aceeași linie.

Exemplu:

```
<td colspan="3">Zona de est</td>
```

b. rowspan="n" - permite unirea a *n* celule consecutive din cadrul aceleiași coloane.

Exemplu:

```
<td rowspan=3>Valori</td>
```

Pentru a nu greși la descriere, în cazul unui tabel având celule unite se recomandă să se pornească de la o reprezentare grafică a acestuia pe care să se indice celulele care trebuie efectiv descrise, ca în figură:

1	1		1
2	2	2	
	3	3	
	4		

```
<table>
<tr>
    <td> ... </td>
    <td colspan="2"> ... </td>
    <td> ... </td>
</tr>
<tr>
    <td rowspan="3"> ... </td>
    <td> ... </td>
    <td colspan="2"> ... </td>
</tr>
<tr>
    <td> ... </td>
    <td colspan="2" rowspan="2"> ... </td>
</tr>
<tr>
    <td> ... </td>
}> linia 1
}> linia 2
}> linia 3
}> linia 4
</table>
```

Aplicație:

Refațeți zona din pagina principală a saitului firmei *Cadouri SRL* astfel încât imaginile obiectelor oferite să fie conținute într-un tabel cu două linii.



Rezolvare:

```
<table>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</table>
```

Modificați tabelul realizat astfel încât să afișeze patru linii, ca mai jos:



<table>

```

<tr>
<td>Birotice</td>
<td colspan="3">Articole din piele</td>
</tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td colspan="4">Ceasuri</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</table>

```

1.3.8 Marcaje pentru definirea legăturilor

Una dintre caracteristicile care dau putere *web*-ului constă în posibilitatea includerii de referințe la alte pagini, din același sait sau din orice sait accesibil prin rețea și indicate prin adresă *URL*. Selectarea cu mouse-ul a unei referințe provoacă afișarea paginii referite.

Pentru includerea unei referințe se va folosi marcajul *<a>* (*anchor*).

1.3.8.1 Referințe selectate folosind șiruri de caractere

Se consideră o pagină web de intrare într-un sait destinat prezentării principalelor realizări artistice ale lui Michelangelo având conținutul următor:

```

<html>
<head><title>Referinte</title></head>
<body>
<h1>MICHELANGELO</h1>
<ol>
    <li><a href="Introd.html">INTRODUCTION</a></li>
    <li><a href="Early.html">EARLY LIFE IN FLORENCE</a></li>
    <li><a href="First.html">FIRST ROMAN SOJOURN</a></li>
    <li><a href="Retour.html">FIRST RETOURN TO FLORENCE</a></li>
    <li><a href="Sistine.html">THE SISTINE CHAPEL CEILING</a></li>
    <li><a href="Julill.html">THE TOMB OF JULIUS II</a></li>
    <li><a href="Laurent.html">THE LAURENTIAN LIBRARY</a></li>
    <li><a href="Medici.html">THE MEDICI TOMBS</a></li>

```

```

<li><a href="Last.html">THE LAST JUDGMENT</a></li>
<li><a href="Campido.html">THE CAMPIDOGLIO</a></li>
</ol>
<hr>
<a href="http://sunsite.unc.edu/wm/paint/auth/michelangelo/">
WebMuseum, Paris: Michelangelo</a><em>Biographical information and images of several of his works on the Web.</em>
</body>
</html>

```

MICHELANGELO

- I. [INTRODUCTION](#)
- II. [EARLY LIFE IN FLORENCE](#)
- III. [FIRST ROMAN SOJOURN](#)
- IV. [FIRST RETOURN TO FLORENCE](#)
- V. [THE SISTINE CHAPEL CEILING](#)
- VI. [THE TOMB OF JULIUS II](#)
- VII. [THE LAURENTIAN LIBRARY](#)
- VIII. [THE MEDICI TOMBS](#)
- IX. [THE LAST JUDGMENT](#)
- X. [THE CAMPIDOGLIO](#)

[WebMuseum, Paris: Michelangelo](#) Biographical information and images of several of his works on the Web.

1.3.8.2 Referințe selectate folosind imagini

În același fel în care sunt atașate şirurilor de caractere, referințele pot fi asociate și imaginilor dintr-o pagină web:

```
<a href="http://www.tigerworld.com"></a>
```

Imaginea folosită pentru specificarea referinței va fi implicit afișată într-un chenar a cărui culoare respectă aceleași reguli ca și culoarea în care ar apărea şirurile de caractere folosite pentru referințe.

Uneori este utilă plasarea unei imagini mici într-o pagină, având o legătură spre o imagine mărită. De exemplu o pagină din saitul despre Michelangelo poate avea în pagina despre Capela Sixtină (*Sistine.html*) câteva astfel de imagini:

```

<h1>V THE SISTINE CHAPEL CEILING</h1>
<p><a href="tavan1.jpg"></a> <a href="tavan2.jpg"></a>
<a href="tavan3.jpg"></a></p>
<p>Michelangelo was recalled to Rome by Pope Julius II in 1505 for two commissions. The most important one was for the
frescoes of the Sistine...

```

V THE SISTINE CHAPEL CEILING



Michelangelo was recalled to Rome by Pope Julius II in 1505 for two commissions. The most important one was for the frescoes of the Sistine Chapel ceiling. Working on scaffolding high above the chapel floor, Michelangelo painted, between 1508 and 1512, some of the finest pictorial images

1.3.8.3 Folosirea referințelor pentru navigarea în pagini lungi

Referințele pot servi și la navigarea în interiorul paginilor lungi. O referință destinată navigării într-o pagină lungă se scrie astfel:

```
<a href="#eticheta">... </a> sau  
<a href="nume.html#eticheta">...</a>
```

În al doilea caz, eticheta indicând poziția din care se va realiza afișarea este conținută în altă pagină web.

Punctele din care se dorește începerea afișării vor fi etichetate. Pentru declararea unei etichete se va folosi tot marcajul având un atribut *name* care indică numele etichetei. Exemplu :

```
<html><head><title>Airplanes</title></head>
<body>
<h1>AIRPLANE STRUCTURE</h1>
<p>Airplanes generally share the same basic configuration—each usually has a fuselage, <a href="#aripi">wings</a>, <a href="#coada">tail</A>, landing gear, and a set of specialized control surfaces mounted on the wings and tail.</p>
<h2>Fuselage</h2>
<p>The fuselage is the main cabin, or body of the airplane. Generally the fuselage has a cockpit section at the front end, where the pilot controls the airplane, and a cabin section. The cabin section may be designed to carry passengers, cargo, or both. In a military fighter plane, the fuselage may house the engines, fuel, electronics, and some weapons. In some of the sleekest of gliders and ultralight airplanes, the fuselage may be nothing more than a minimal structure connecting the wings, tail, cockpit, and engines.</p>
<h2><a name="aripi"></a> Wings </h2>
<p>All airplanes, by definition, have wings. Some, like the flying wings built by the Northrop-Grumman Corporation, based in the United States, are nearly all wing with a very small cockpit. Others have minimal wings, or wings that seem to be merely extensions of a blended, aerodynamic fuselage, such as the space shuttle.</p>
<h2><a name="coada"></a>Tail Assemby </h2>
```

<p>Most airplanes, except for flying wings, have a tail assembly attached to the rear of the fuselage, consisting of vertical and horizontal stabilizers, which look like small wings; a rudder; and elevators. The components of the tail assembly are collectively referred to as the empennage.</p>

Dacă se dorește trecerea la eticheta #aripi pornind dintr-o altă pagină web, referința ar fi trebuit scrisă astfel :

airplane wings

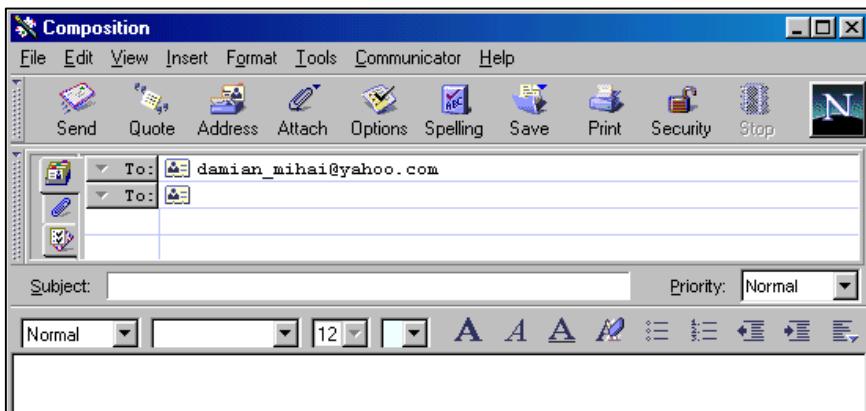
1.3.8.4 Folosirea referințelor pentru lansarea în execuție a unor aplicații

O referință poate declanșa execuția unei aplicații instalate pe calculatorul folosit la navigare. Dacă referința conține numele unui fișier (audio, video, prezentare multimedia etc.), aplicația de navigare realizează două acțiuni: lansează în execuție aplicația asociată implicit extensiei fișierului și transferă acesteia fișierul preluat prin rețea.

- a. **Lansarea în execuție a aplicației destinată trimiterii de mesaje prin e-mail.** Exemplu:

<p>Pentru orice informații suplimentare puteti scrie directorului scolii, conf.dr.ing. Mihai DAMIAN (damian_mihai@yahoo.com)</p>

Pentru orice informații suplimentare puteti scrie directorului scolii, conf.dr.ing. Mihai DAMIAN (damian_mihai@yahoo.com)



VARIANTĂ : folosirea unei imagini :

- b. **Redarea unei prezentări realizate în PowerPoint.** Prezentările multimedia realizate în *PowerPoint* și salvate în format *PowerPoint Show*, *.pps:, pot fi redate fie direct în fereastra programului de navigare (cazul aplicației Internet Explorer) fie în fereastra aplicației *PowerPoint*, dacă aplicația de navigare nu dispune de posibilitatea redării acestor fișiere. Exemplu:

```
<a href="prezentare.pps">Start prezentare PowerPoint Show</a>
```

- c. **Secvențe de film.** Pentru a reda o secvență de film este necesar ca formatul fișierului care conține filmul să fie compatibil cu o aplicație instalată pe calculator. De regulă pentru afișarea secvențelor video se folosește aplicația *Windows Media Player* (componentă Windows). Exemple:

```
<a href="film.wmv">Start film</a>
<a href="film.avi">Start film</a>
```

- d. **Secvențe sonore.** Redarea acestora se face similar celor video. Fișierele conținând secvență sonoră pot avea diferite extensii, cele mai întâlnite fiind .wav sau .mp3. Exemple:

```
<a href="sunet.wav">Start secvența sonora</a>
<a href="melodie.mp3">Start melodie mp3</a>
```

1.3.9 Hărți de imagini

Se poate crea o imagine având mai multe regiuni, fiecare fiind folosită pentru realizarea unei referințe. Zonele sensibile la selectarea cu mouse-ul pot fi rectangulare, poligonale sau circulare. Determinarea coordonatelor punctelor care definesc regiunile se poate realiza folosind Inkscape, care afișează în permanență coordonatele cursorului. Marcajele care se vor folosi în acest caz sunt *<map>* și *<area>*. Numele hărtii se declară cu ajutorul atributului *usemap="#nume"* introdus în cadrul marcajului **.

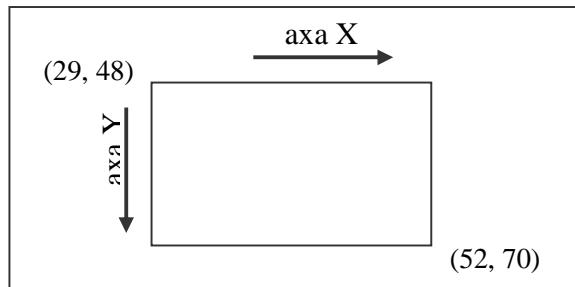
Exemplu:

```

<map id="depozite">
  <area shape="rect" coords="29,48,52,70" href="dep1.html" alt="" />
  <area shape="rect" coords="39,78,152,120" href="dep2.html" alt="" />
  <area shape="rect" coords="190,38,252,70" href="dep3.html" alt="" />
</map>
```

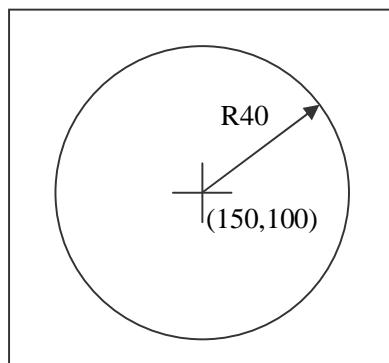
Pentru declararea zonelor sensibile la acțiunea mouse-ului se vor folosi atributele *shape* pentru a indica forma zonei și *coords* pentru a-i defini poziția și dimensiunile. Modul de folosire a acestor atrbute reiese din exemplele următoare:

a. Zonă de formă rectangulară (*shape="rect"*):



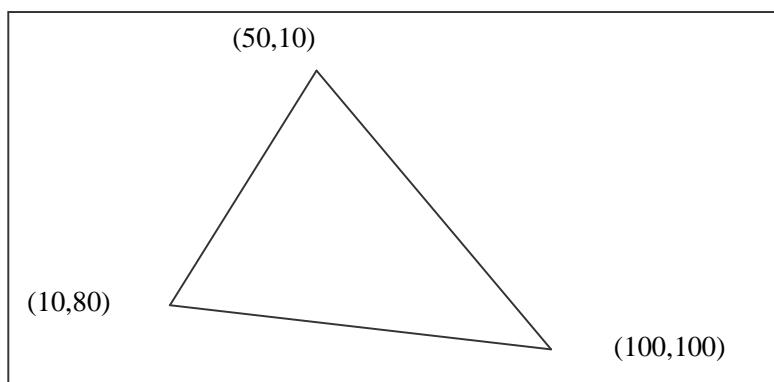
```
<area shape="rect" coords="29,48,52,70" href="dep1.html" alt="" />
```

b. Zonă de formă circulară (*shape="circle"*):



```
<area shape="circle" coords="150,100,40" href="departe.html" alt="" />
```

c. Zonă de formă poligonală (*shape="poly"*)



```
<area shape="poly" coords="10,80,50,10,100,100" href="cluj.html" alt="" />
```

Aplicație: Completăți prima pagină a saitului *Cadouri SRL* adăugând legături conform indicațiilor din figură:

Cadouri SRL

Noi va promovam firma

- [Home](#) → #
- [Prezentare](#) → prezentare.html
- [Produse](#) → produse.html
- [Contact](#) → mailto:cadouri_srl@gmail.com

Cadouri promotionale

Doriti desigur sa va promovati afacerea si probabil ati observat ca banalele calendare sau agende oferite la sfarsit de an provoaca efecte limitate. In catalogul nostru veti gasi insa o mare varietate de produse care pot fi personalizate si oferite partenerilor Dv. de afaceri.

Daca nu va puteti decide cu privire la produsul cel mai potrivit, solicitati-ne o consultatie si un consilier va veni la Dv., va va prezenta mostre si va va ajuta sa alegeti cea mai potrivita solutie.

Birotica Articole din piele



Birotica

- [Pixuri](#) → catalog.html#pix
- [Seturi de birou](#) → catalog.html#setb
- [Agende electronice](#) → catalog.html#agende
- [Trusa markere](#) → catalog.html#mark
- [Articole din piele](#) → catalog.html#piele

Ceasuri

- [Colectia Clasic](#) → catalog.html#clasic
- [Colectia XXL](#) → catalog.html#xxl
- [Colectia Luxury](#) → catalog.html#lux

[Home](#) | [Prezentare](#) | [Produse](#) | [Contact](#)

© 2008. Toate drepturile rezervate. Proiectat de INFOAP.

Rezolvare:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
<head>
  <title>Cadouri SRL</title>
</head>
<body>
  <h1>Cadouri SRL</h1>
  <p>Noi va promovam firma</p>
  <ul>
    <li><a href="#">Home</a></li>
    <li><a href="prezentare.html">Prezentare</a></li>
    <li><a href="produse.html">Produse</a></li>
    <li><a href="mailto:cadouri_srl@gmail.com">Contact</a></li>
  </ul>
  <h2>Cadouri promotionale</h2>
  <p>Doriti desigur sa va promovati afacerea si probabil ati observat ca banalele calendare sau agende oferite la sfarsit de an provoaca efecte limitate. In catalogul nostru veti gasi insa o mare varietate de produse care pot fi personalizate si oferite partenerilor Dv. de afaceri. </p>
  <p>Daca nu va puteti decide cu privire la produsul cel mai potrivit, solicitati-ne o consultatie si un consilier va veni la Dv., va va prezenta mostre si va va ajuta sa alegeti cea mai potrivita solutie.</p>
  <table>
    <tr>
      <td>Birotica</td>
      <td colspan="3">Articole din piele</td>
    </tr>
    <td></td>
    <td></td>
    <td></td>
    <td></td>
  </tr>
  <tr>
    <td colspan="4">Ceasuri</td>
  </tr>
  <tr>
    <td></td>
    <td></td>
    <td></td>
    <td></td>
  </tr>
  </table>
  <h2>Birotica</h2>
  <ul>
    <li><a href="catalog.html#pix">Pixuri </a></li>
    <li><a href="catalog.html#setb">Seturi de birou </a></li>
    <li><a href="catalog.html#agende">Agende electronice </a></li>
    <li><a href="catalog.html#mark">Trusa markere </a></li>
```

```
<li><a href="catalog.html#piele">Articole din piele </a></li>
</ul>
<h2>Ceasuri</h2>
<ul>
    <li><a href="catalog.html#clasic">Colectia Clasic </a></li>
    <li><a href="catalog.html#xxl">Colectia XXL </a></li>
    <li><a href="catalog.html#lux">Colectia Luxury </a></li>
</ul>
<a href="#">Home</a> | <a href="prezentare.html">Prezentare</a> | <a
href="produse.html">Produse</a> | <a
href="mailto:cadouri_srl@gmail.com">Contact</a>
<p id="copyright">© 2008. Toate drepturile rezervate. Proiectat de INFOAP. </p>
</body>
</html>
```

2.1 Aspecte generale

Evoluția web-ului a condus la apariția unui limbaj destinat specificării aspectului paginilor web, CSS (eng. *Cascading Style Sheets*). Folosind CSS se pot impune proprietățile fonturilor, culori, se poate controla poziția imaginilor sau a altor elemente care formează conținutul paginilor. Înaintea adoptării pe scară largă a cuplajului XHTML-CSS se încerca rezolvarea problemelor legate de aspectul paginilor web folosind marcaje HTML și atribute ale acestora dar neuniformitatea tratării acestora în diferite browsere a devenit o problemă a cărei adevărată soluție este reprezentată de CSS.

Pe lângă specificarea modului de afișare a informației din paginile web, CSS oferă posibilitatea impunerii acelorași caracterisici ansamblului de pagini care formează un sait.

CSS este suportat de toate browserele actuale.

Deoarece învățarea CSS presupune un mare număr de exerciții și experimente, este bine să se acceseze saitul <http://www.w3schools.com/css/>. Exercițiile din acest sait se realizează într-o aplicație integrată în browser care permite editarea regulilor CSS și vizualizarea imediată a efectelor.

2.2 Sintaxa CSS

Impunerea modului de afișare a informației în CSS se realizează prin reguli. O regulă indică modul în care trebuie afișat un element din pagină.

Sintaxa unei reguli este:

```
selector {proprietate:valoare;}
```

Exemplu:

```
body {
    background-color: black;
}
```

sau, mai compact:

```
body {background-color: black;}
```

Regulile CSS pot fi definite fie ca valoare a atributului *style* adăugat marcajelor HTML, fie pot fi incluse la începutul paginii, în secțiunea *head*, între marcaje *<style> ... </style>*. Prima variantă realizează o modificare locală, afectând doar informația afișată folosind marcajul modificat, în timp ce a doua se extinde la tot fișierul. În cele ce urmează se va folosi însă numai a doua variantă deoarece scopul utilizării soluției

XHTML+CSS în locul limbajului HTML este mai ales acela de a separa conținutul paginilor web de informațiile de formatare.

Exemple:

```
<head>
<title> Formatarea textului</title>
<style type="text/css">
  p {color: white; font-family: verdana,arial; font-size:14px; background-color: purple"}
</style>
</head>
<body>
...
<p>Text imprimat pe fond purpuriu.</p>
...
```

Rezultat:

Text imprimat pe fond purpuriu

```
<head><title>Exemplu mai complicat</title>
<style type="text/css">
```

```
h1 { font-size: x-large; color: red }
h2 { font-size: large; color: blue }
body {background-color: black;
      color: white;
    }

p {
  color: green;
  font-family: tahoma,arial,verdana;
  font-size: 12px;
}
```

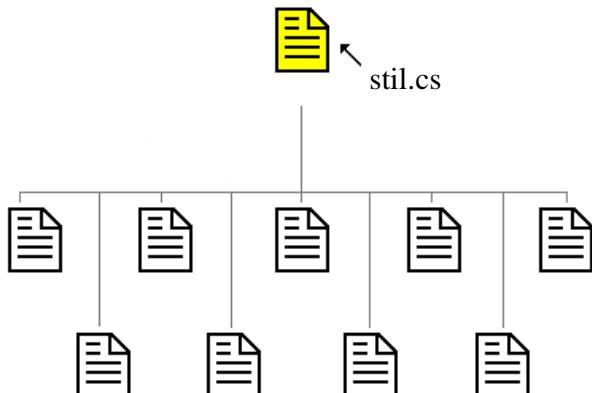
(stil.css)

```
</style>
</head>
```

Observație: Dacă zona de definire a stilurilor atașate diferitelor marcaje este salvată separat, într-un fișier denumit de exemplu **stil.css**, pentru a impune într-o pagină web a saitului aceleași modificări, se va include în secțiunea **<head>** a acesteia un marcat **<link>** în care se va specifica fișierul care conține definițiile de stiluri, ca în exemplul de mai jos:

```
<head>
<link rel="stylesheet" type="text/css" href="stil.css" />
</head>
<body>
...
```

Avantajul acestei soluții constă din faptul că aspectul paginilor sitului în care este inclus fișierul **stil.css** poate ușor modificat, prin simpla editare a regulilor conținute de acesta.



2.3 Selectorii

Pentru a impune cui se aplică o anumită regulă sau un set de reguli, în CSS se folosesc selectori. În regulile deja scrise, *body* sau *p* care precedau regulile scrise sunt selectori. Un selector poate fi deci numele unui marcat XHTM și în acest caz regulile care i-au fost atașate în zona de definire a stilurilor vor fi aplicate peste tot unde se folosește respectivul marcat. Dar CSS permite și declararea unor selectori noi care vor fi folosiți în conținutul paginii ca valori ale atributelor *class* sau *id*.

Posibilitatea definirii o singură dată a modului în care un marcat XHTM operează reprezintă un argument important în adoptarea CSS ca soluție pentru formatare. În paginile web anumite marcaje pot apărea de multe ori (*<p>* sau *<td>* de exemplu). Dacă de fiecare dată am modifica aspectul implicit prin atribute suplimentare fișierul rezultat ar fi mult mai mare și mai greu de pus la punct și de întreținut.

Exemplu fundamental:

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<head>
    <title>Exemplu de formatare paragraf</title>
    <style type="text/css">
        p {color: white; font-family: verdana,arial; background-color: purple;}
        /*
        Comentariu CSS pe mai multe linii
        */
        .verde {color: white; font-family: verdana,arial; background-color: green;}
        .albastru {color: white; font-family: verdana,arial; background-color: blue;}
        #col_stg {position: absolute; left:10px; top:50px; width:200px; background:#fff; border:1px solid #000;padding:8px;}
    </style>
</head>
<body>
    <p>Un paragraf cu fundal violet.</p>
    <p class="verde">Un paragraf verde.</p>
    <p class="albastru">Un paragraf albastru.</p>
    <div id="col_stg">Un element cu proprietăți speciale.</div>
</body>

```

```

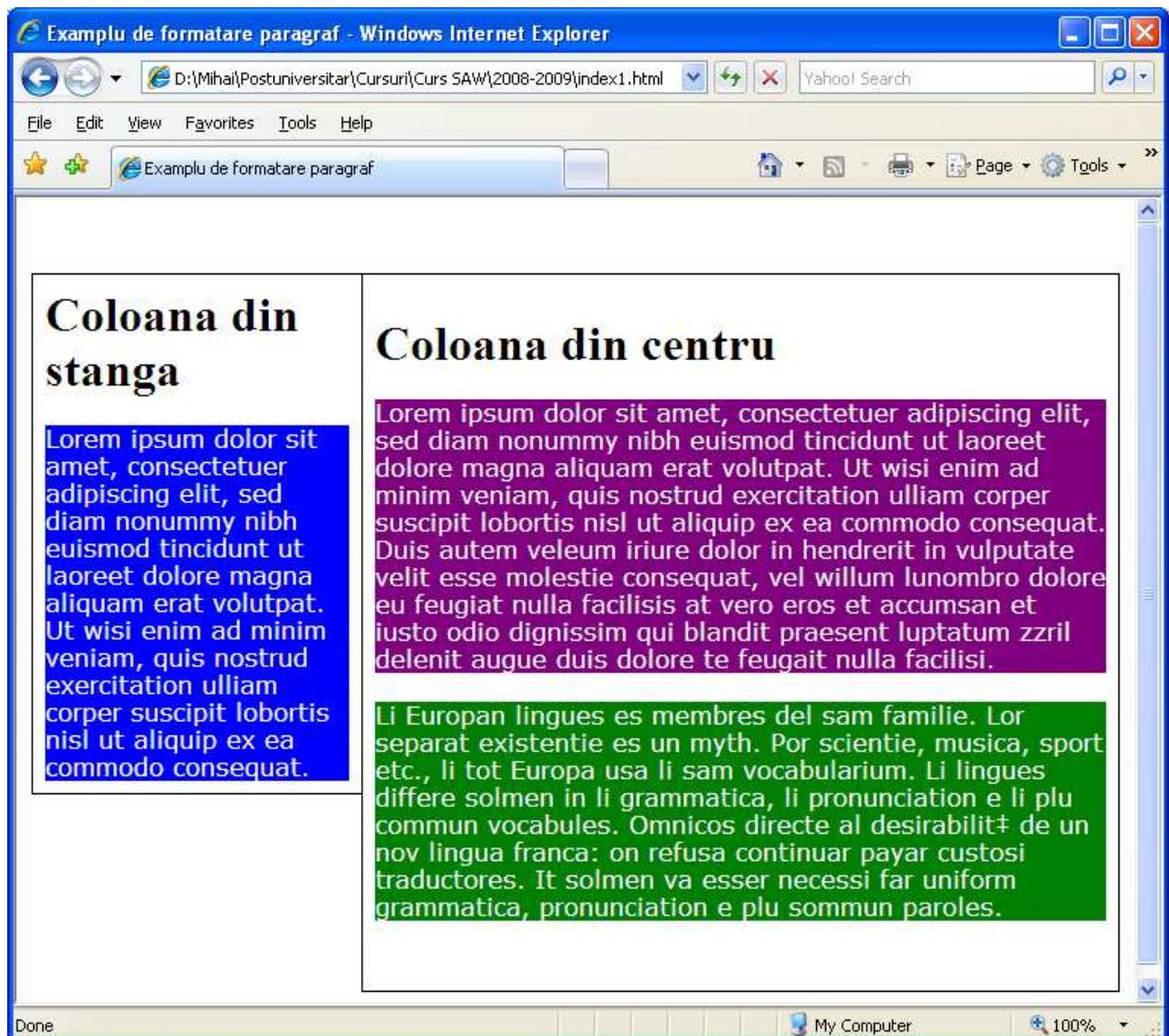
#col_centr {background:#fff; margin-left: 217px; margin-top:35px; border:1px solid
#000;padding:8px;}
</style>
</head>
<body>
<div id="col_stg">
    <h1>Coloana din stanga</h1>
    <p class="albastru">Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
    nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim
    ad minim veniam, quis nostrud exercitation ullam corpor suscipit lobortis nisl ut aliquip ex ea
    commodo consequat.
    </p>
</div>
<div id="col_centr">
    <h1>Coloana din centr</h1>
    <p>
        Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
        tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis
        nostrud exercitation ullam corpor suscipit lobortis nisl ut aliquip ex ea commodo consequat.
        Duis autem voleum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel
        willum lunombro dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio
        dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi.
    </p>
    <p class="verde">Li European lingues es membres del sam familie. Lor separat existentie es
        un myth. Por scientie, musica, sport etc., li tot Europa usa li sam vocabularium. Li lingues
        differe solmen in li grammatica, li pronunciation e li plu commun vocabules. Omnicos directe al
        desirabilit‡ de un nov lingua franca: on refusa continuar payar custosi traductores. It solmen
        va esser necessi far uniform grammatica, pronunciation e plu sommun paroles.
    </p>
    <br />
</div>
</body>
</html>

```

În definirea stilurilor s-au specificat caracteristicile a cinci selectori: *p*, *.verde*, *.albastru*, *#col_stg*, și *#col_centr*. Primul este un marcat XHTML, deci peste tot unde acest marcat va fi folosit va determina afişarea în modul specificat. Ceilalți doi selectori vor putea fi folosiți ca valori ale atributului *class*. Folosirea unor selectori suplimentari astfel definiți face posibil ca anumite paragrafe să fie afișate cu un stil diferit de cel obișnuit, definit folosind selectorul *p*.

Ultimii doi selectori sunt folosiți ca valori ai atributului ID. Spre deosebire de *class*, o valoare a atributului ID poate apărea într-o pagină o singură dată. În CSS atributul ID se folosește în principal pentru a specifica proprietățile blocurilor în care se divide o pagină. În exemplul dat selectorii *col_stg* și *col_centr* sunt folosiți în marcasele destinate divizării paginii în blocuri, *<div>*. Practic s-au definit două blocuri ale căror proprietăți au fost astfel stabilite încât pagina să conțină două coloane.

Rezultat:



Dacă un selector `.nume` definit de programator va fi folosit numai pentru un tip de marcaj html, la declarare se va scrie `marcaj.nume{...}`. Exemplu:

```
p.big{font-size: 2em;}  
...  
<p class="big">Scriere cu litere mari.</p>
```

Un grup de reguli poate fi aplicat mai multor selectori. În astfel de cazuri, înaintea grupului de reguli sunt scriși selectorii despărțiti prin virgule. Exemplu:

```
p, h1, h2, h3, h4, h5 {font-family: Arial; color: black; }
```

2.4 Culori

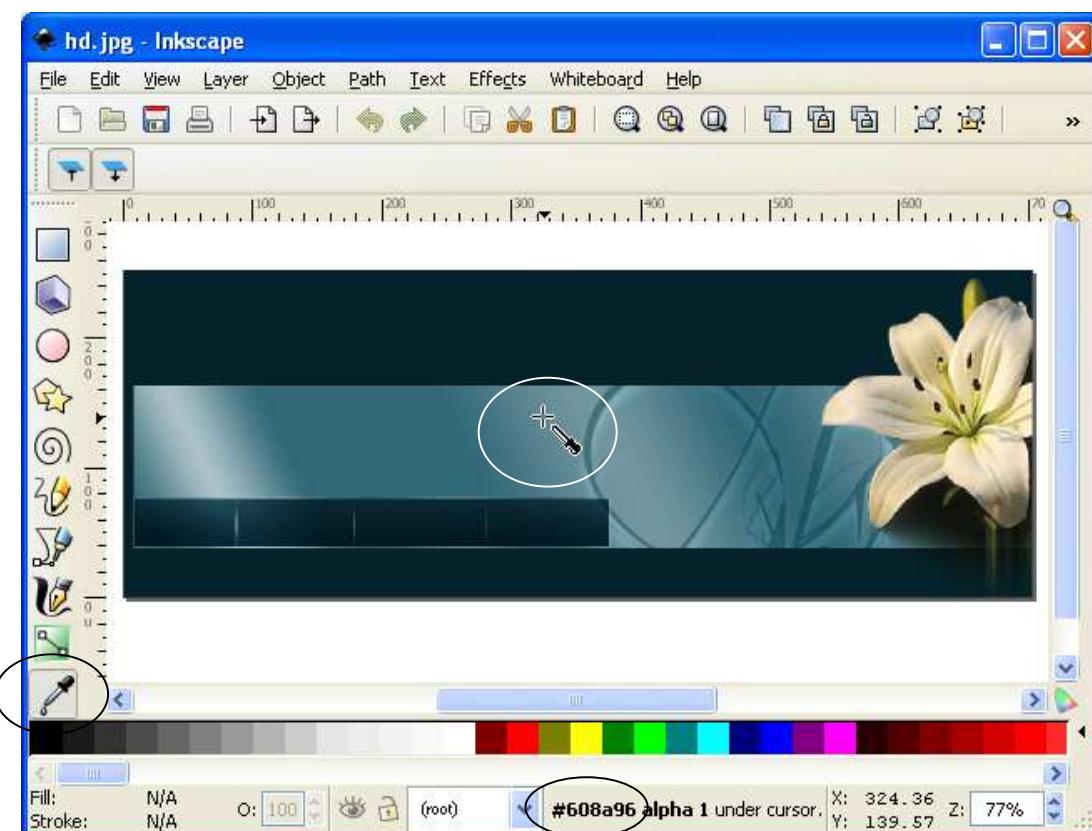
Proprietățile CSS pentru definirea culorilor sunt *color* pentru conținut și *background-color* pentru culoarea fundalului.

O culoare este indicată prin nume (pentru culorile obișnuite: red, yellow, blue, white etc.) sau este definită prin componente RGB (Red + Green + Blue, roșu + verde + albastru) care o compun. Fiecare dintre componentele culorii poate fi exprimată printr-un număr întreg cuprins între 0 și 255.

Definirea culorii poate fi realizată folosind codul hexazecimal sau zecimal. În cazul folosirii codului hexazecimal notația culorii începe cu caracterul '#' și are formatul *#rrggbb*. Exemplu: *#ff7c0d*. În cazul în care cele trei componente ale culorii sunt exprimate prin perechi de cifre identice se poate folosi o scriere prescurtată. Exemplu: *#ff00aa* se poate scrie *#f0a*.

Pentru a exprima în zecimal codul unei culori se va scrie *rgb(rrr, ggg, bbb)*. Exemplu: *color: rgb(62, 178, 12)*.

Pentru a obține codul culorii în hexazecimal se poate folosi Inkscape. Astfel pentru a prelua culoarea unui punct de pe o imagine se selectează  , se indică punctul și se citește codul de pe bara de stare a aplicației.



Aplicația afișează de asemenea o paletă de culori în partea de jos a ferestrei. Codul hexa al unei culori din această paletă este afișat automat la plasarea cursorului mouse-ului peste mostra de culoare afișată.

2.6 Unități de măsură

Unele dintre proprietățile elementelor conținute într-o pagină web necesită scrierea unei dimensiuni. Cu excepția valorilor nule, după dimensiune trebuie scrisă unitatea de măsură.

U.M.	Explicație
%	procent. Ex: p.big {line-height: 200%}
in	inch. Ex: h2 {margin: 0.5in;}
cm	centimetri. Ex. h2 {margin: 0.5cm;}
mm	milimetri
em	1 em este egal cu mărimea normală a fontului. Ex. h1 {font-size: 2em;} (identic cu font-size:200%)
pt	1pt=1/72in
pc	1pc=12px
px	pixeli. Exemplu: p {font-size: 12px}

Exemplu:

p {margin: 0 20px 0 20px}

2.7. Fonturi

În CSS fontul este definit prin denumire, dar, deoarece pe calculatorul pe care se afișează pagina web fontul indicat poate lipsi, după denumirea fontului (fonturilor) se precizează familia din care acesta face parte. Proprietatea utilizată este *font-family*.

Fontul	Familia
"Times New Roman", Gramond, Georgia	serif
Arial, Verdana, Helvetica	sans-serif
Courier, "Courier New"	monospace

Exemplu:

```
h1 {font-family: verdana, helvetica, sans-serif; }
h2 {font-family: "Times New Roman", serif; }
```

Majoritatea paginilor web folosesc *Verdana*.

Înclinarea caracterelor se stabilește cu ajutorul proprietății *font-style*, valorile posibile fiind *normal*, *italic* sau *oblique*.

Exemplu:

```
h2 {
    font-family: "Times New Roman", serif;
    font-style: italic;
}
```

Folosind proprietatea *font-variant* se poate impune scrierea cu majuscule mici (*font-variant: small-caps*) sau normal (*font-variant: normal*).

Scrierea *bold* se poate impune folosind *font-weight*. Proprietatea poate avea valorile *bold* sau *normal*.

Dimensiunea caracterelor poate fi impusă folosind proprietatea *font-size*. De obicei mărimea este definită în pixeli (px) sau procentual (%) sau em). Este recomandată varianta definirii procentuale deoarece în acest caz rămâne posibilă mărirea fontului folosind opțiunile programului de navigare. Varianta definirii mărimii caracterelor în unități absolute (px, in, cm, etc.) se aplică în cazurile în care modificarea mărimii fontului în momentul afișării ar afecta grav aspectul acesteia. De exemplu textul plasat deasupra unui buton desenat nu trebuie să poată fi mărit.

Exemplu:

```
h1 {font-family: arial, verdana, sans-serif;
    font-size: 150%; font-weight: bold}
h2 {font-family: "Times New Roman", serif;
    font-size: 120%; }
```

2.8. Formatarea textului

Formatarea textului se realizează cu un set de proprietăți care permit diverse tipuri de alinieri, indentări și spațieri.

Indentarea se realizează folosind proprietatea *text-indent*. Efectul indentării este decalarea poziției de început a primei linii a fiecărui paragraf cu o mărime impusă.

Exemplu:

```
p {
    text-indent: 50px;
}
```

Alinierarea textului se realizează folosind proprietatea *text-align*. Valorile posibile ale proprietății sunt: *left*, *right*, *center* sau *justify*.

Exemplu:

```
th {
    text-align: right;
}

td {
    text-align: center;
}

p {
    text-align: justify;
}
```

Sublinierea sau bararea textului se realizează folosind proprietatea *text-decoration*, având valorile posibile *none*, *underline* (subliniat), *overline* (barat), *line-through* (tăiat).

Exemplu:

```
h1 {
    text-decoration: underline;
}

h2 {
    text-decoration: overline;
}

h3 {
    text-decoration: line-through;
}
```

Modificarea distanței dintre caractere se realizează folosind proprietatea *letter-spacing*.

Exemplu:

```
h1 {
    letter-spacing: 6px;
}

p {
    letter-spacing: 3px;
}
```

Controlul scrierii cu majuscule se realizează folosind proprietatea *text-transform*. Valorile posibile sunt:

- *capitalize* - prima literă va fi majusculă
- *uppercase* - toate literele vor fi majuscule,
- *lowercase* - toate literele vor fi mici,

- *normal* - caracterele vor fi scrise normal.

2.9. Formatarea referințelor

Legăturile sunt evidențiate în HTML prin sublinierea sirului de caractere folosit ca suport și utilizarea unor culori distințe pentru legăturile nevizitate sau vizitate. În CSS aceste attribute implicate pot fi modificate.

Modificarea culorii, a fontului sau suprimarea sublinierii se realizează ca și pentru texte obișnuite.

Exemplu:

```
a {
  color: blue;
  text-decoration:none;
}
```

Pentru a modifica pe rând proprietățile sirului folosit pentru o referință se vor folosi *pseudoclaselor*:

- *a:link*, pentru legături nevizitate,
- *a:visited*, pentru legături vizitate,
- *a:active* pentru legături active sau,
- *a:hover*. pentru a modifica modul de afișare a sirului la plasarea cursorului mouse-ului deasupra lui.

Exemplu:

```
a:link {
  color: #6699CC;
}
a:visited {
  color: #660099;
}

a:active {
  background-color: #FFFF00;
}

a:hover {
  text-transform: uppercase;
  font-weight:bold;
  color:blue;
  background-color:yellow;
}
```

2.10. Colorarea scrisului și a fundalului

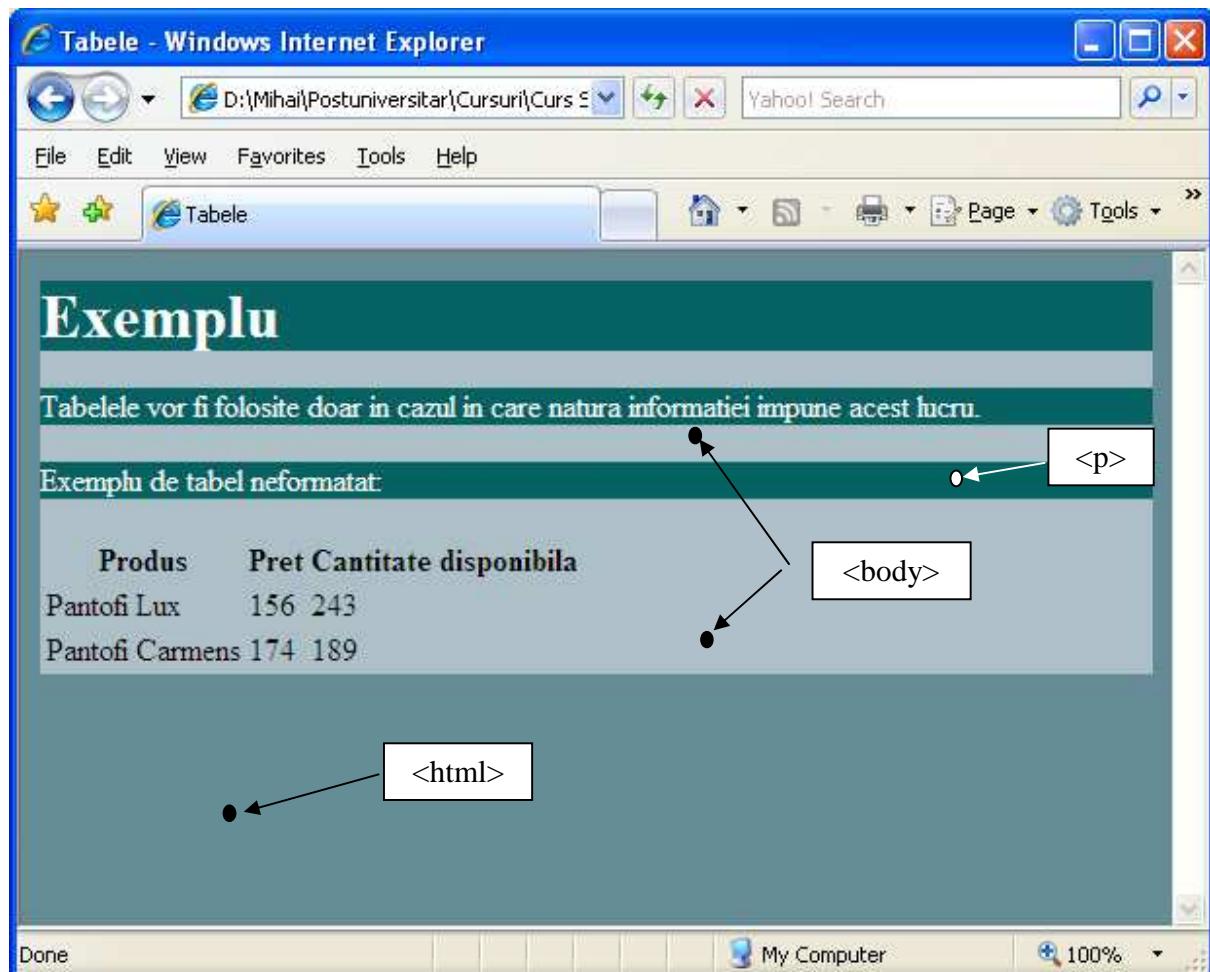
Culoarea folosită la scriere se definește prin proprietatea *color*. De exemplu dacă titlurile marcate cu *<h1>* trebuie scrise în roșu, aceasta se poate indica prin regula următoare:

```
h1 {color:#ff0000; }
(sau, mai simplu, h1 {color:red; })
```

Culoarea fundalului se impune prin proprietatea *background-color*. Modul în care dispune browserul diferențele blocuri care formează o pagină poate fi evidențiat colorând diferit fundalul acestora, ca în exemplul următor:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<title>Tabele</title>
<style type="text/css">
    html {background-color:#658c96}
    body {background-color:#adc0c7}
    p,h1 {color: white; background-color:#076363}
</style>
</head>

<body>
<h1>Exemplu</h1>
<p>Tabelele vor fi folosite doar în cazul în care natura informației impune acest lucru.</p>
<p>Exemplu de tabel neformatat:</p>
<table>
<tr>
    <th>Produs</th>
    <th>Preț</th>
    <th>Cantitate disponibilă</th>
</tr>
<tr>
    <td>Pantofi Lux</td>
    <td>156</td>
    <td>243</td>
</tr>
<tr>
    <td>Pantofi Carmens</td>
    <td>174</td>
    <td>189</td>
</tr>
</table>
</body>
</html>
```



Aplicație:

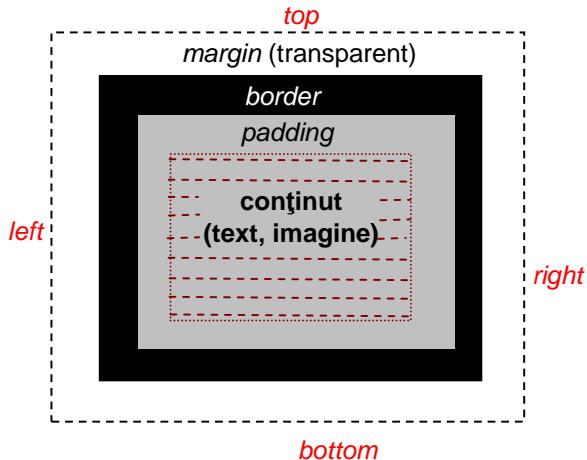
Completați prima pagină a saitului firmei *Cadouri SRL* cu următoarele reguli:

```
html {background-color:#05212c }
body {font-family:Verdana, Arial, sans-serif; }
p {color:#9cc; text-align: left;}
td {color:#fff; text-align: center;}
h1, h2 {font-size:30px; color:#fff; text-align: left;}
a {color:#9cc; text-decoration: none;}
a:hover {text-decoration: underline;}
```

2.11. Geometria unui bloc în CSS

Informația cu caracter grafic (text, imagine) conținută într-o pagină web este dispusă într-o serie de blocuri alăturate. Browser-ul generează aceste blocuri automat, la întâlnirea marcajelor care delimită informația: `<html>`, `<body>`, `<h1>...<h6>`, `<p>`, `<table>`, `<td>` etc.

Arhitectura unui bloc este prezentată în figura următoare:



Din schiță se observă că blocul este înconjurat de un contur transparent (*margin*) având rolul de a permite distanțarea blocurilor și eventual de un chenar (*border*). Între informația efectiv conținută și *border* este o distanță definită prin proprietatea *padding*. Fundalul este colorat în culoarea definită prin proprietatea *background-color* și se extinde până la *border* (include deci și zona definită prin *padding*).

Geometria implicită a blocurilor depinde de browser. Diferențele de la un browser la altul nu sunt mari dar pot fi deranjante.

Modul implicit în care *Internet Explorer* construiește și plasează blocurile poate fi evidențiat construind o mică pagină web și impunând culori diferite pentru fundalul diferitelor blocuri.

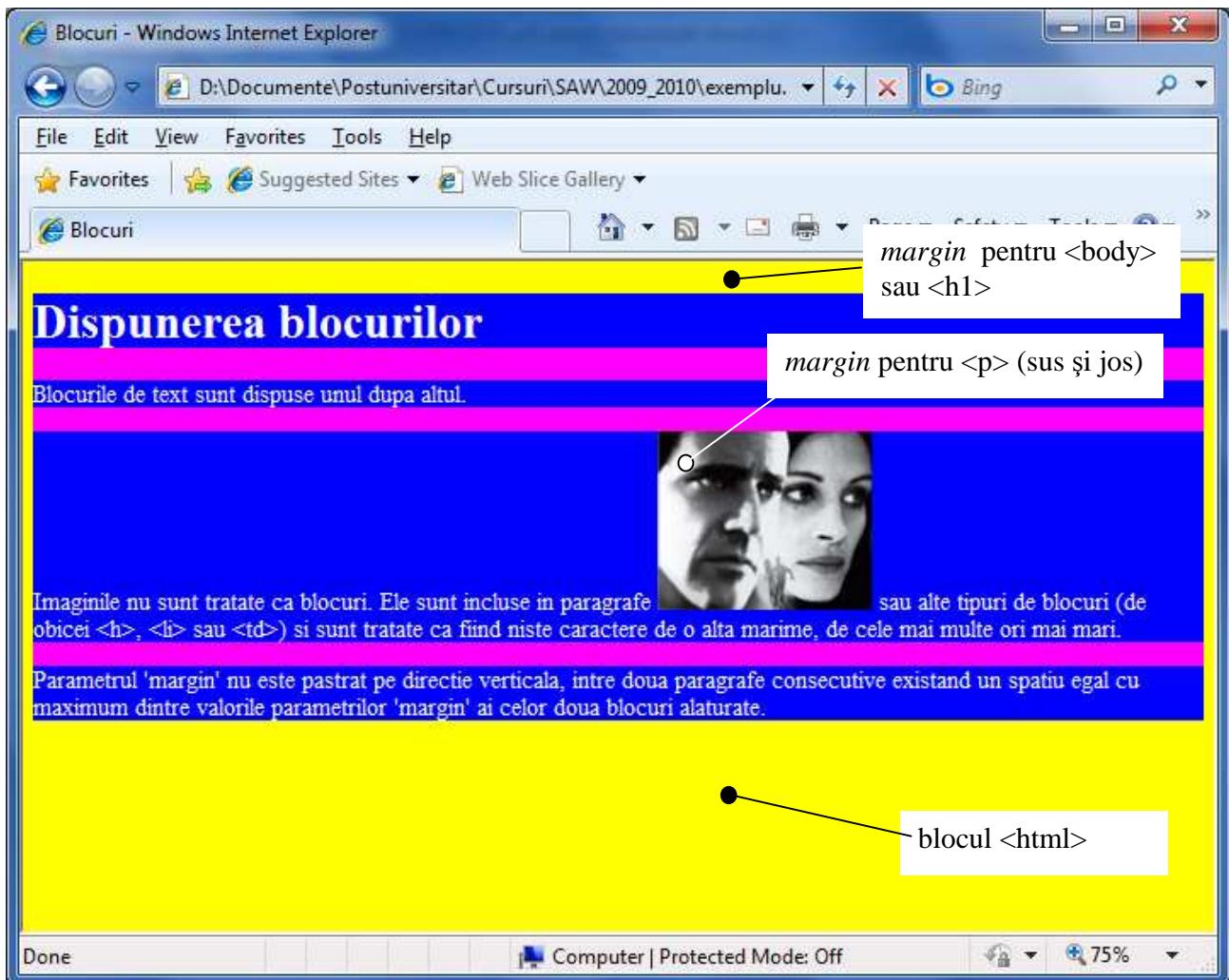
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<title>Blocuri</title>
<style type="text/css">
    html {background-color:yellow}
    body {background-color:magenta}
    p,h1 {color: white; background-color:blue}
</style></head>
<body>
    <h1>Dispunerea blocurilor</h1>
    <p>Blocurile de text sunt dispuse unul după altul.</p>
```

<p> Imaginile nu sunt tratate ca blocuri. Ele sunt incluse in paragrafe sau alte tipuri de blocuri (de obicei <h>, sau <td>) si sunt tratate ca fiind niste caractere de o alta marime, de cele mai multe ori mai mari.</p>

<p>Parametrul 'margin' nu este pastrat pe directie verticala, intre doua paragrafe consecutive existand un spatiu egal cu maximum dintre valorile parametrilor 'margin' ai celor doua blocuri alaturate.</p>

</body>
</html>

Rezultat:



Proprietățile care pot fi specificate pentru a defini geometria unui bloc sunt: *margin*, *border*, *padding*, *width* (lățimea) și *height* (înălțimea). Pentru *margin*, *border* și *padding* se pot indica dimensiuni diferite pentru fiecare dintre laturile blocului folosind sufixele *-top*, *-right*, *-bottom* și *-left*. Exemplu:

```
h1 {margin-left:120px; margin-right:12px; border:1px; border-color:blue;}
```

Pentru simplificarea scrierii dimensiunilor, dacă toate cele patru valori sunt identice se poate scrie *parametru:valoare*. Dacă valorile sunt diferite se pot scrie toate cele patru valori, una după alta, despărțite prin spațiu, ordinea fiind *top*, *right*, *bottom*, *left*. Exemplu:

```
p {margin:12px 0 14px 120px; padding:8px;}
```

Notă: Pentru valori nule dimensiunile pot fi omise.

Dacă se impune de exemplu *margin:0* pentru *body*, *p* și *h1* pagina precedentă ar fi afișată astfel:

The screenshot shows a Mozilla Firefox window with the title bar "Blocuri - Mozilla Firefox". The menu bar includes File, Edit, View, History, Bookmarks, Tools, and Help. The toolbar contains icons for Back, Forward, Stop, Home, and Search. The address bar shows "file:///D:/Mihai/Postuniversitar/Cursuri/Curs SA". The bookmarks bar has links to Most Visited, Customize Links, Free Hotmail, My Yahoo!, Windows Marketplace, Windows Media, Disable, Cookies, CSS, Forms, Images, Information, Miscellaneous, Outline, and Resize. The main content area displays the following text and image:

Dispunerea blocurilor

Blocurile de text sunt dispuse unul după altul.

Imaginele nu sunt tratate ca blocuri. Ele sunt incluse în paragrafe sau alte tipuri de blocuri (de obicei <h>, sau <td>) și sunt tratate ca fiind niște caractere de o altă marime, de cele mai multe ori mai mari.

Parametrul 'margin' nu este pastrat pe direcție verticală, între două paragrafe consecutive existând un spațiu egal cu maximum dintre valorile parametrilor 'margin' ai celor două blocuri alăturate.

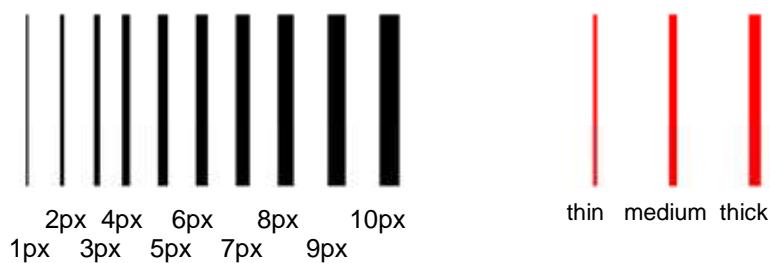
A large yellow rectangular area covers the bottom portion of the content area.

De altfel frecvent se impune pentru `<body>` proprietatea `margin=0` (implicit `margin` este de aproximativ 8px, depinzând de browser).

2.12. Încadrarea blocurilor

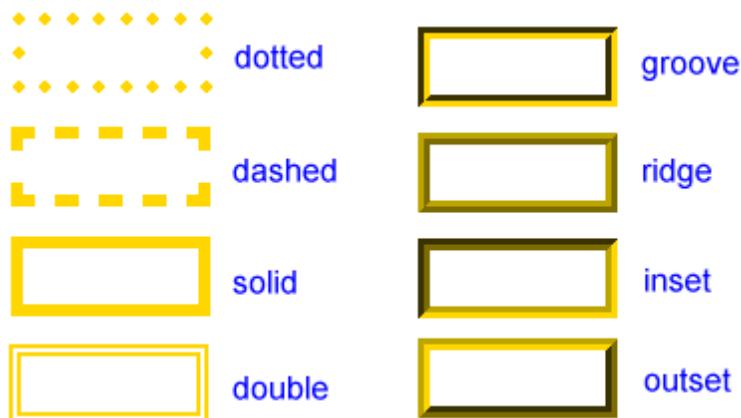
Blocurile pot fi încadrate cu un contur având caracteristicile precizate prin proprietățile `border-width`, `border-color` și `border-style`.

`border-width` permite specificarea grosimii liniei de încadrare în pixeli sau printr-una din valorile: `thin`, `medium` sau `thick`.



`border-color` permite specificarea culorii conturului de încadrare.

`border-style` permite precizarea aspectului conturului. valorile posibile sunt cele din figură.



Exemplu de definiții de contururi de încadrare:

```
h1 {
  border-width: thick;
  border-style: dotted;
  border-color: gold;
}
```

```

h2 {
    border-width: 20px;
    border-style: outset;
    border-color: red;
}

p {
    border-width: 1px;
    border-style: dashed;
    border-color: blue;
}

ul {
    border-width: thin;
    border-style: solid;
    border-color: orange;
}

```

Ca și în cazul proprietăților *margin* și *padding*, este posibilă specificarea unor caracteristici diferite pentru *top*, *right*, *bottom* și *left*. Pentru specificarea valorilor specifice vor fi folosite în acest caz denumirile *border-top*, *border-right*, *border-bottom* sau *border-left*, după caz.

Exemplu:

```

h1 {
    border-top-width: thick;
    border-top-style: solid;
    border-top-color: red;

    border-bottom-width: thick;
    border-bottom-style: solid;
    border-bottom-color: blue;

    border-right-width: thick;
    border-right-style: solid;
    border-right-color: green;

    border-left-width: thick;
    border-left-style: solid;
    border-left-color: orange;
}

```

Observație: Pentru *border*, CSS definește ordinea normală de definire a proprietăților conturului de încadrare ca fiind *border-width* -> *border-style* -> *border-color*. Pentru o scriere mai compactă se acceptă și o variantă prescurtată de definire:

border: 1px solid blue;

2.13. Parametrii *width* și *height*

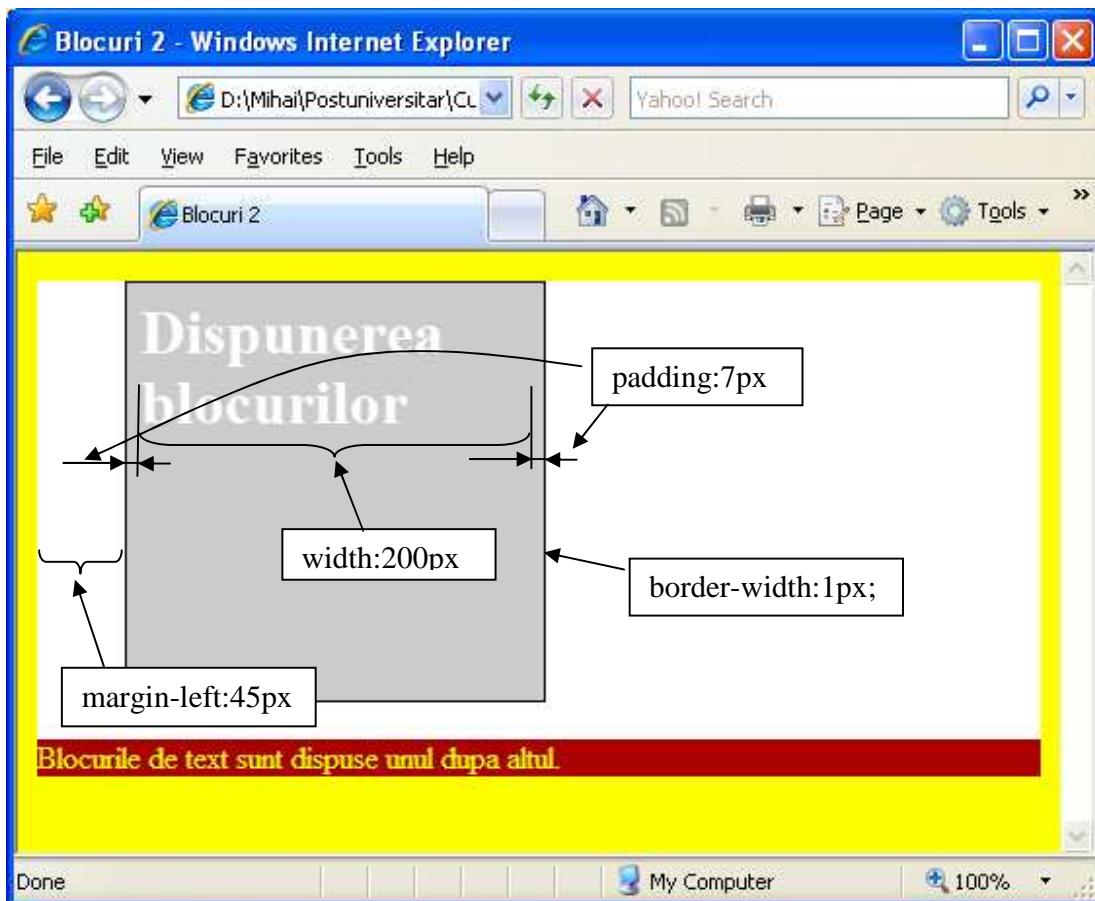
Standardul W3C permite precizarea mărimii zonei folosite pentru afișarea informației. Modelul definit în standard este cel din figură.



Dacă descrierea blocului include valori nenule pentru *padding*, *border* și *margin*, lățimea efectivă se calculează adunând parametrului *width* mărimile corespunzătoare ale celorlalte elemente ale blocului. Exemplu:

```
<html>
<head>
<title>Blocuri 2</title>
<style type="text/css">
    html {background-color:yellow}
    body {background-color:white}
    h1 {
        color: white;
        background-color:#ccc;
        margin-left:45px;
        margin-right:0;
        border-width:1px;
        border-style:solid;
        border-color:#000;
        padding:7px;
        width:200px;
        height:200px;
    }
    p {color: yellow; background-color:#a00;}
</style>
</head>

<body>
    <h1>Dispunerea blocurilor</h1>
    <p>Blocurile de text sunt dispuse unul după altul.</p>
</body>
</html>
```



Lățimea efectivă a blocului referit prin selectorul *h1* este:

$$L = 45 + 1 + 7 + 200 + 7 + 1 = 261\text{px}.$$

Proprietatea *overflow* permite specificarea modului de tratare a situației în care conținutul unui bloc nu începe în spațiul alocat. Situația apare în cazul blocurilor pentru care s-au specificat atât lățimea cât și înălțimea (*width* și *height*). Valorile posibile sunt *overflow :scroll* sau *overflow :hidden*. În prima variantă, dacă se plasează într-un bloc mai multă informație decât permit dimensiunile acestuia vor fi afișate automat cursoare pentru defilare (*scroll*).

2.14. Marcajele *<div>* și **

În paginile realizate până acum *<body>* conținea un număr de blocuri (definite prin selectorii *p*, *h1*, *h2* etc) dispuse unul după altul, de sus în jos. Selectorul *<body>* descrie deci un container care conține celealte blocuri care formează conținutul paginii.

La rândul său, *<body>* este conținut în *<html>*.

Standardul HTML oferă însă programatorilor posibilitatea definirii de containere suplimentare care vor conține fiecare un ansamblu de blocuri. Pentru definirea blocurilor având rol de container se folosesc marcaje `<div>` sau ``. Diferența dintre cele două este faptul că `` este de tip *inline*, el realizând o grupare în interiorul unui bloc, asemenea marcajelor `` sau ``.

Pentru o pagină obișnuită marcajul `<div>` va fi folosit de exemplu pentru a defini blocuri diferite pentru capul paginii, zona de picior și corp. Separarea în blocuri distincte a zonelor care se repetă pe toate paginile saitului este interesantă deoarece de cele mai multe ori problemele ridicate de descrierea conținutului acestora sunt mai complicate. În plus, în timp ce pentru capul și piciorul paginii fonturile sunt de regulă de dimensiuni fixe (exprimate în px), pentru conținutul paginii dimensiunile fonturilor vor fi exprimate în unități relative (em, %) astfel încât cititorul să poată modifica mărimea folosind facilitățile browser-ului (în Internet Explorer se selectează *View/Text Size/ Larger* sau *Largest*).

Exemplu fundamental:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<title>Blocuri DIV</title>
<style type="text/css">
    html {background-color:yellow;margin:0; padding:0;}
    body {margin:0; padding:0;}
    div {margin:0; padding:0;}
    p,h1 {color: white; background-color:#000080;
           margin:10px; padding:0; font-family:verdana, helvetica, sans-serif;}
    p {font-size:80%;}
    img {display: block; margin:10px;padding:0;float:left;}
    #continut {width:700px; background-color:#008080;
               text-align:left;margin:0 auto 0 auto;padding:0;}
    #cap {width:700px; height:87px; background-color:#ff8080;}
    #cap h1 {font-size: 20px;padding-top:30px;margin:0 10px 0 10px;}
    #cap p {font-size: 10px;padding:0;}

    #nav {width:700px; height:40px; background-color:#00aa00; }
    #nav p {margin:0 10px 0 10px;padding-top:10px; font-size: 10px }

    #colstg {width:200px; background-color:#ffffff;float: left;}
    #coldr {width:500px; background-color:#aaaaaa; float:left;}
    #picior {width:700px; height: 35px; background-color:#ff8080;
              clear: both;text-align: center;}
    #picior p {font-size:12px;margin:0 10px 0 10px;padding-top:10px;}
</style>
</head>
<body>
```

```

<div ID="continut">
  <div ID="cap">
    <h1>Dispunerea blocurilor</h1>
    <p>Blocurile de text sunt dispuse unul dupa altul.</p>
  </div>

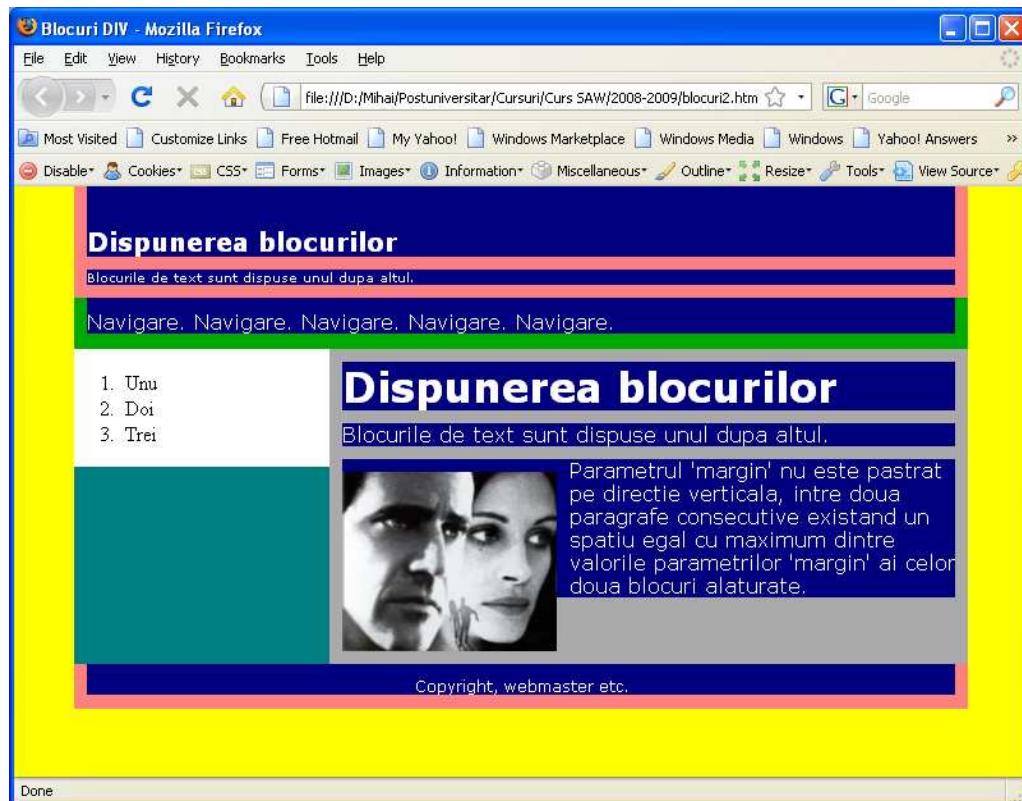
  <div ID="nav">
    <p>Navigare. Navigare. Navigare. Navigare. Navigare. </p>
  </div>

  <div ID="colstg">
    <ol>
      <li>Unu</li>
      <li>Doi</li>
      <li>Trei</li>
    </ol>
  </div>

  <div id="coldr">
    <h1>Dispunerea blocurilor</h1>
    <p>Blocurile de text sunt dispuse unul dupa altul.</p>
    
    <p>Parametrul 'margin' nu este pastrat pe directie verticala, intre doua paragrafe consecutive existand un spatiu egal cu maximum dintre valorile parametrilor 'margin' ai celor doua blocuri alaturate.</p>
  </div>

  <div id="picior">
    <p>Copyright, webmaster etc.</p>
  </div>
</div>
</body>
</html>

```



Spre deosebire de `<div>`, folosit la definirea de blocuri, marcajul `` permite modificarea proprietăților unei porțiuni din textul conținut într-un bloc.

Exemplu:

```
<style type="text/css">
    .profit {
        color:orange;
    }
    .paguba {
        color:blue;
    }
</style>
```

`<p>Munca in echipa poate creste eficienta fiecaruia. Munca de unul singur e paguboasa.</p>`

2.15 Imagini plasate pe fundal

CSS oferă o multitudine de proprietăți care pot restricționa modul de disponere a unei imagini pe fundalul unui bloc.

Exemplu:

Se dorește includerea unei imagini pe fundalul paginii. Imaginea va fi automat repetată până la umplerea spațiului disponibil.

```
<style type="text/css">
    body {
        background-color: #FFCC66;
        background-image: url("imagini/sigla.gif");
    }
</style>
```

Fișierul care conține imaginea va fi plasat în directorul *imagini* derivat din directorul care conține pagina web.

Repetarea imaginii poate fi controlată prin valorile proprietății *background-repeat*. Valorile posibile sunt:

Valoare	Semnificație
<i>background-repeat: repeat-x</i>	Repetare pe orizontală
<i>background-repeat: repeat-y</i>	Repetare pe verticală
<i>background-repeat: no-repeat</i>	Fără repetare

Exemplu:

```
<style type="text/css">
#header {
background-color: #FFCC66;
background-image: url("imagini/sigla.gif");
background-repeat: no-repeat;
}
</style>
```

În cazul în care imaginea dispusă pe fundal nu este repetată iar mărimea blocului diferă de mărimea imaginii se va preciza de regulă și culoarea fundalului. Alegerea corectă a culorii fundalului presupune de obicei determinarea cu Inkscape a culorii frontierei imaginii, trecerea de la imagine la zona alăturată putând fi practic imperceptibilă.

Comportamentul imaginii în fereastră la defilarea conținutului acesteia (*scroll*) este specificată prin proprietatea *background-attachment*. Valorile posibile ale acestei proprietăți sunt *scroll* sau *fixed*. Prima valoare specifică defilarea imaginii de fundal împreună cu conținutul paginii în timp ce a doua indică o poziție fixă a imaginii în raport cu fereastra aplicației, pagina defilând pe deasupra acesteia.

Exemplu:

```
<style type="text/css">
#continut {
background-color: #FFCC66;
background-image: url("imagini/sigla.gif");
background-repeat: no-repeat;
background-attachment: fixed;
}
</style>
```

Pozitia imaginii în fereastră este indicată folosind proprietatea *background-position*. Există trei moduri de a indica poziția imaginii de fundal:

- prin coordonate (de obicei pixeli):

background-position: 30px 45px;

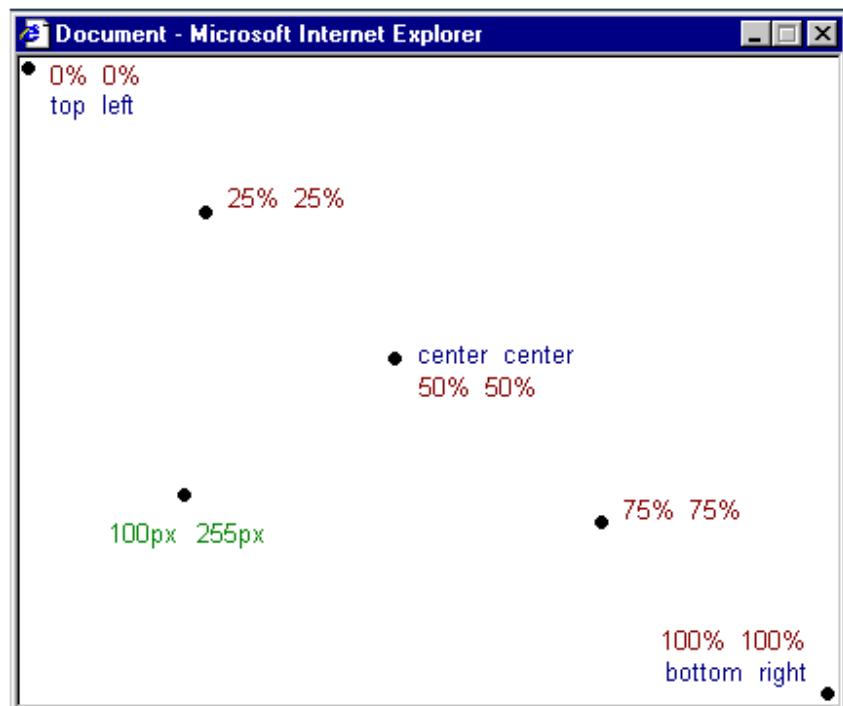
- prin procente din dimensiunile zonei afișabile:

background-position: 50% 35%;

- în raport cu laturile blocului. Valorile posibile sunt *top left*, *top right*, *bottom left*, *bottom right* sau *center center*.

Exemplu:

```
#continut {
background-color: #FFCC66;
background-image: url("butterfly.gif");
background-repeat: no-repeat;
background-attachment: fixed;
background-position: right bottom;
}
```



2.16. Blocuri flotante

În mod normal elementele conținute într-o pagină web sunt plasate de browser unul după altul, de sus în jos. Există situații în care se dorește dispunerea de la stânga la dreapta sau de la dreapta la stânga și în aceste cazuri se va folosi la descrierea blocurilor proprietatea *float*.

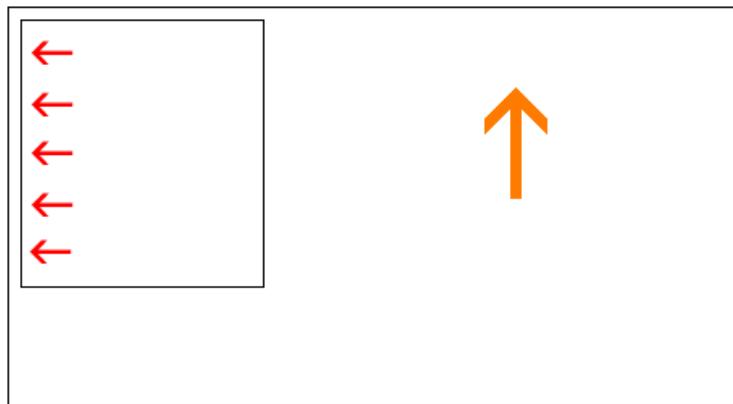
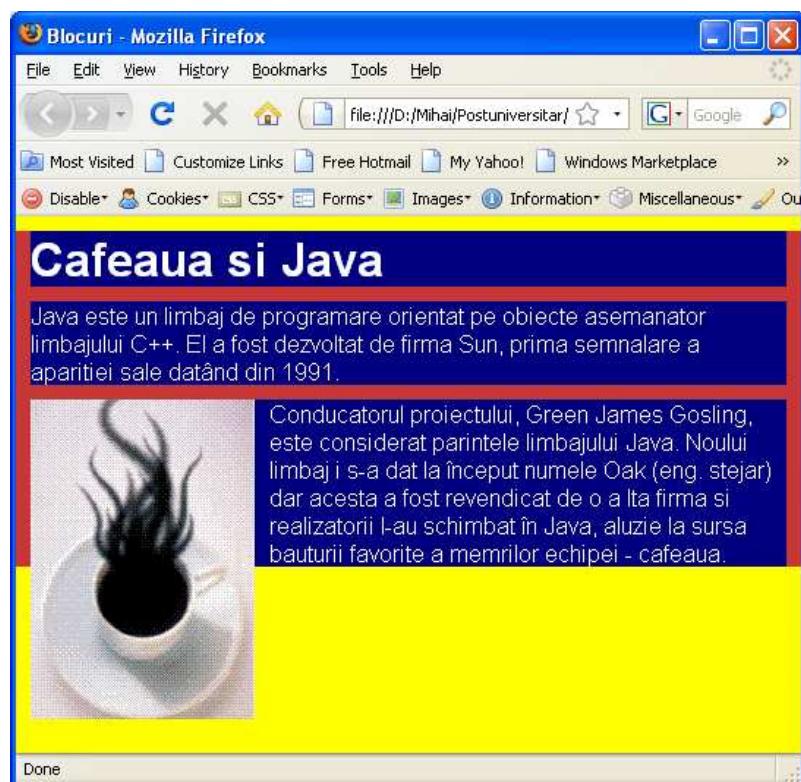


Fig. 2.9 Efectul proprietății *float : left*

Blocul declarat flotant va fi cadrat în sensul dat de valoarea proprietății (*float :left* sau *float :right*) iar restul conținutului paginii este aranjat astfel încât să ocupe spațiul astfel creat.

Exemplul 1.

Se dorește plasarea într-un bloc la stânga paginii a unei figuri și apoi scrierea unui text astfel încât acesta să încadreze blocul, ca în figura următoare.



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<title>Blocuri</title>
<style type="text/css">
html {background-color:yellow}
body {background-color:#c83737; margin:0;}
p,h1 {color: white; background-color:#000080; margin:10px; font-family: arial, verdana, sans-serif;}
img.stg{margin-right: 10px; float:left;}
</style>

<body>
<h1>Cafeaua si Java</h1>
<p>Java este un limbaj de programare orientat pe obiecte asemanator limbajului C++. El a fost dezvoltat de firma Sun, prima semnalare a aparitiei sale datand din 1991. </p>
<p> Conducatorul proiectului, Green James Gosling, este considerat parintele limbajului Java. Noului limbaj i s-a dat la inceput numele Oak (eng. stejar) dar acesta a fost revendicat de o alta firma si realizatorii l-au schimbat in Java, aluzie la sursa bauturii favorite a membrilor echipei - cafeaua. </p>
</body>
</html>
```

Exemplul 2.

Pentru evidențierea primului cuvânt al unui capitol se dorește plasarea primului caracter ca în figura de mai jos.

When you create an action, you automate a series of steps. The hardest part about creating a new action is figuring out what functions you want to automate. Think about steps that you carry out over and over, and whether you could be more productive if you had an action that could do them for you.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<style type="text/css">
.stg{
    margin-right: 10px;
    color: blue;
    font-family: arial, verdana, sans-serif;
    font-size: 300%;
    float:left;
}
p{
    color: #008080;
```

```

        font-family: arial, verdana, sans-serif;
    }
</style>
</head>
<body>
<p><span class="stg">W</span>hen you create an action, you automate a series of steps. The hardest part about creating a new action is figuring out what functions you want to automate. Think about steps that you carry out over and whether you could be more productive if you had an action that could do them for you.</p>
</body>
</html>

```

Proprietatea clear permite plasarea unui bloc sub blocurile flotante anterioare. Valorile posibile sunt *clear:left*, *clear:right* sau *clear:both*. Astfel la descrierea unui bloc care trebuie dispus sub un număr de blocuri care definesc un ansamblu de coloane (cazul blocului care conține piciorul paginii) folosind *clear* avem garantia că noul bloc nu va fi plasat peste unul dintre blocurile flotante anterioare.

Exemplul 3.



Java este un limbaj de programare orientat pe obiecte asemănător limbajului C++. El a fost dezvoltat de firma Sun, prima semnalare a apariției sale datând din 1991. Limbajul a fost dezvoltat în cadrul unui proiect denumit Green, obiectivul

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<html>
<style type="text/css">
    div.stg{margin-right: 10px; float:left; }
    .gata{ color: #008080; font-family: arial, verdana, sans-serif; clear:both; }
</style>

```

```

<body>
<div class="stg"></div>
<p class="gata">Java este un limbaj de programare orientat pe obiecte asem&atilde;n&atilde;tor limbajului C++. El a fost dezvoltat de firma Sun, prima semnalare a aparitiei sale dat&acirc;nd din 1991.
...
</p>
</body>
</html>

```

Exemplul 4.

Pentru scrierea conținutului unei pagini pe trei coloane s-au definit blocurile *col1*, *col2* și *col3* ca în listingul următor:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

```

<html>
<head>
<style type="text/css">
div {
    color: #008080; font-family: arial, verdana, sans-serif;
}

.col{
    width:31%; padding-right: 15px; float: left;
}
</style>
</head>
<body>
<h1>Limbajul Java</h1>
<div class="col">
<p>Java este un limbaj de . . .</p>          (prima coloană)
</div>
<div class="col">
<p>Java este un limbaj de . . .</p>          (a doua coloană)
</div>
<div class="col">
<p>Java este un limbaj de . . .</p>          (a treia coloană)
</div>
</body>
</html>

```

Limbajul JAVA

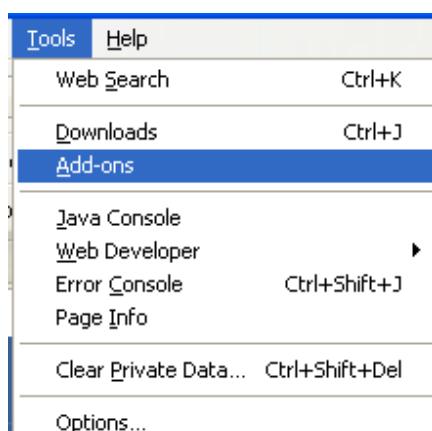
Java este un limbaj de programare orientat pe obiecte asemãnãtor limbajului C++. El a fost dezvoltat de firma Sun, prima semnalare a aparitiei sale datând din 1991. Limbajul a

Java este un limbaj de programare orientat pe obiecte asemãnãtor limbajului C++. El a fost dezvoltat de firma Sun, prima semnalare a aparitiei sale datând din 1991. Limbajul a

Java este un limbaj de programare orientat pe obiecte asemãnãtor limbajului C++. El a fost dezvoltat de firma Sun, prima semnalare a aparitiei sale datând din 1991. Limbajul a

Validarea paginilor web

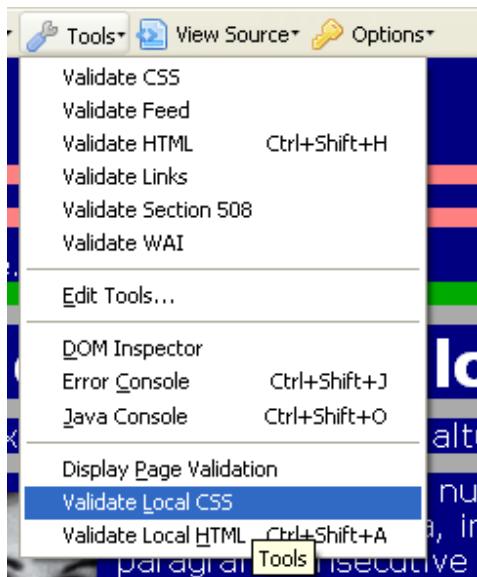
Pentru a avea garanția corectitudinii conținutului unei pagini web, acesta poate fi verificat cu ajutorul unei aplicații dezvoltate de W3C și disponibile online. O metodă ușoară de trimitere a paginii spre verificare presupune utilizarea ca browser fie a aplicației Internet Explorer versiunea 8 fie a aplicației Mozilla Firefox. În cazul folosirii aplicației Firefox acesteia trebuie să se adauge extensia *Web Developer*. Această extensie este destinată dezvoltatorilor de pagini web și oferă un ansamblu de instrumente foarte utile. Pentru adăugarea în Firefox a extensiei *Web Developer* se selectează în meniul aplicației *Tools / Add-ons*:



The screenshot shows the Mozilla Add-ons page for the 'Web Developer' extension. At the top, there's a logo for 'Web Developer' by Chris Pederick, a rating of 4.5 stars from 116 reviews, and 81,185 weekly downloads. Below this, there's a 'Web Development' category label and a brief description: 'Adds a menu and a toolbar with various web developer tools.' On the right side, there's a green 'Add to Firefox' button with a dropdown arrow. At the bottom right, it says 'Updated May 19, 2008'.

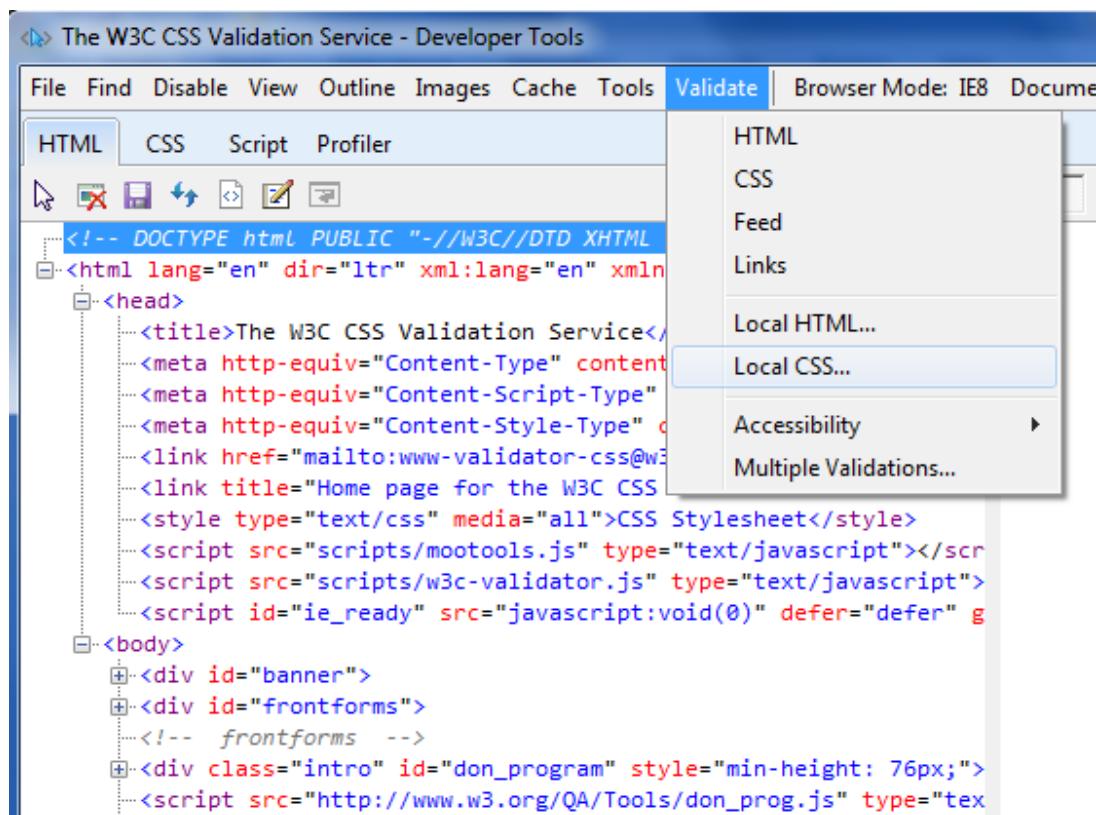
În urma instalării extensiei *Web Developer* aplicația va afișa o bară cu instrumente suplimentară.

Pentru validarea codului unei pagini web se deschide pagina în browser și se selectează pe noua bară cu instrumente *Tools / Validate Local CSS* sau *Tools / Validate Local HTML*:

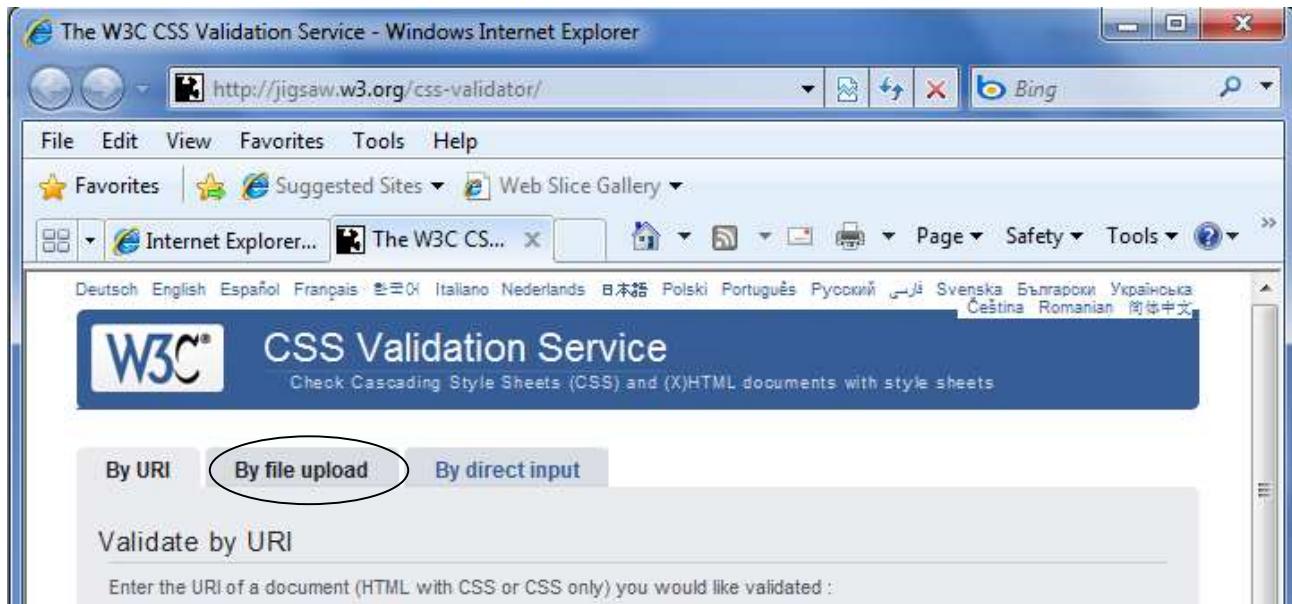


Aplicația va transmite prin Internet conținutul foii de stiluri spre o aplicație specializată (<http://jigsaw.w3.org/css-validator/validator>). Raportul prezentat de aplicație va conține erorile depistate și apoi va afișa regulile găsite ca fiind corecte.

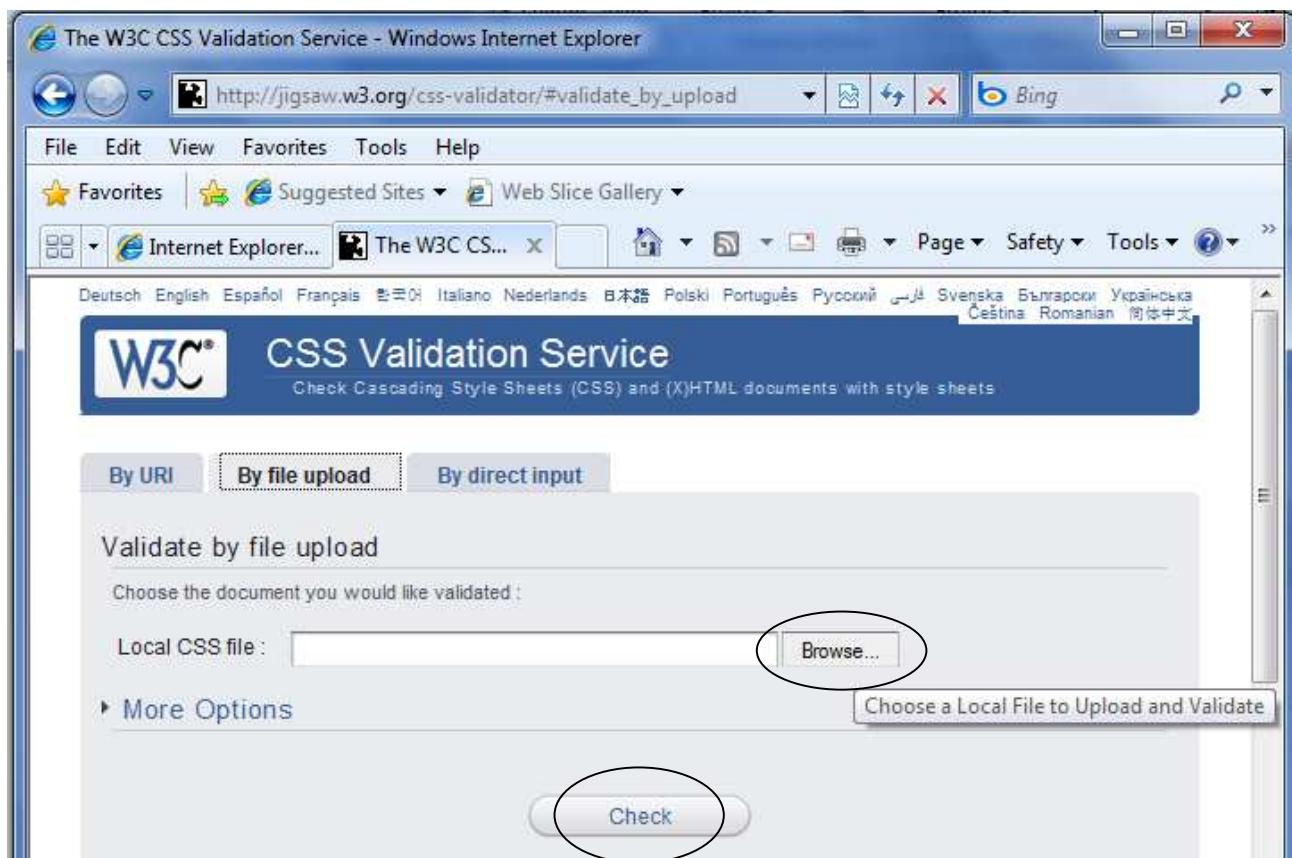
În Internet Explorer se va accesa Tools / Developer Tools și în fereastra afișată se va accesa Validate / Local CSS sau Local HTML.



Internet Explorer va accesa imediat saitul <http://jigsaw.w3.org/> și în fereastra afișată se va selecta opțiunea By file upload, ca în figură.



În fereastra afișată se va selecta fișierul de validat și se va apăsa *Check*.



2.17. Poziționarea blocurilor

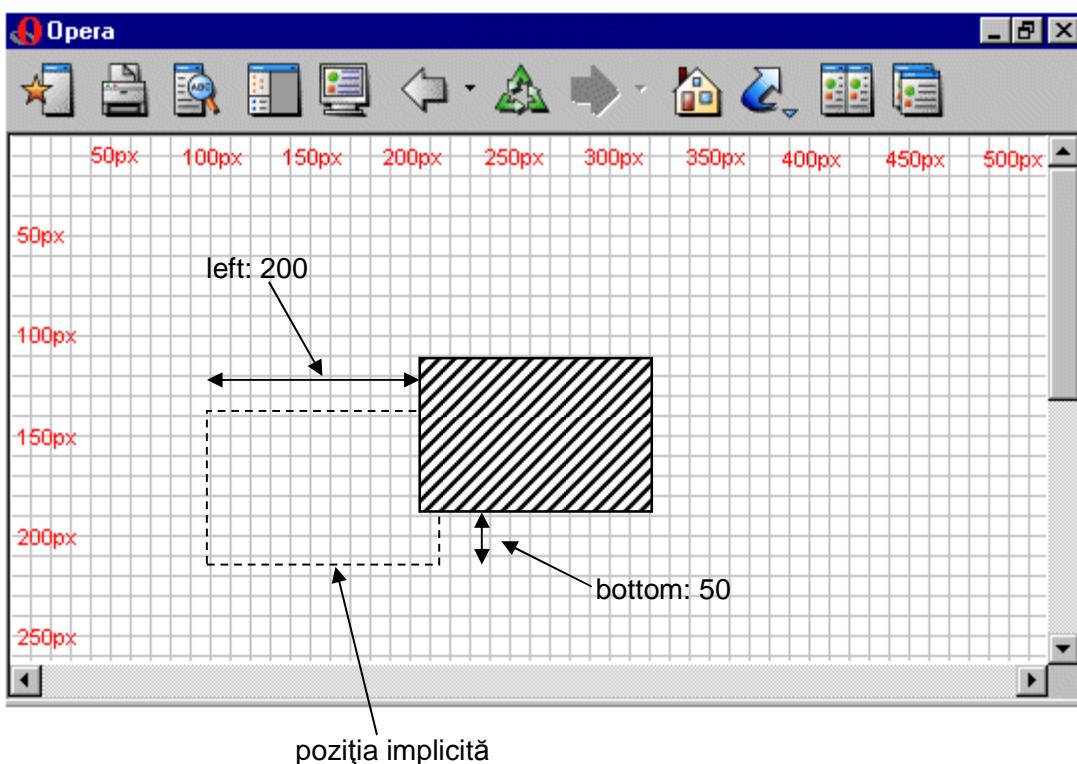
În CSS poziționarea a blocurilor se poate realiza folosind coordonate absolute sau relative. În mod corespunzător, proprietatea *position* care specifică modul de poziționare a unui bloc poate avea valoarea *relative* sau *absolute*.

Distanțele care precizează poziția unui bloc sunt specificate printr-o pereche de proprietăți din mulțimea *top*, *right*, *bottom* sau *left*.

În cazul poziționării relative poziția blocului este specificată în raport cu punctul în care blocul ar fi fost afișat în lipsa proprietății *position*.

Exemplu:

```
div.poz1 {
    position: relative;
    left: 200px;
    bottom: 50px;
}
```



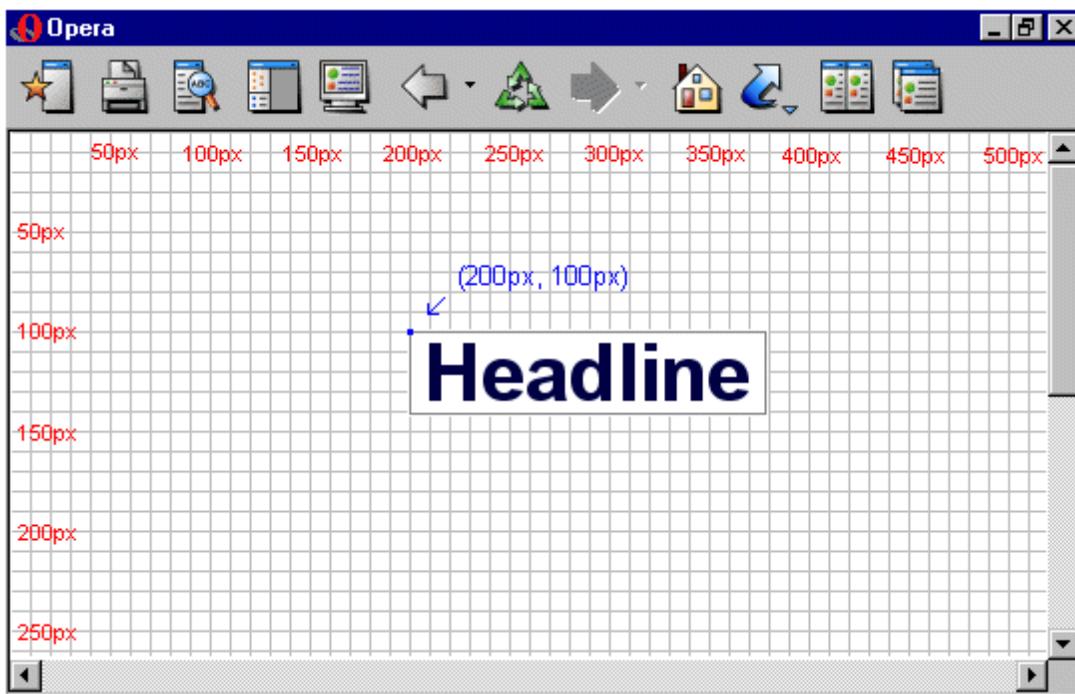
Deși conținutul blocului din exemplu va fi afișat în altă poziție, *browserul* va lăsa neocupată zona pe care acesta ar ocupa-o în mod normal !

Poziționarea absolută diferă fundamental de cea relativă. Astfel, pentru un bloc poziționat absolut *browser-ul* nu va rezerva loc în pagină. El trebuie să fie considerat ca aparținând unui alt nivel, plasat deasupra ferestrei browserului, deci un astfel de bloc se

poate suprapune peste un alt bloc din pagină. Astfel de blocuri pot fi folosite pentru a realiza de exemplu meniuri derulante care în stare desfășurată acoperă parțial conținutul paginii.

Pentru înțelegerea poziționării absolute se consideră regula următoare:

```
h1 {
    position: absolute;
    top: 100px;
    left: 200px;
}
```



Coordonatele indicate (`top:100px` și `left:200px`) au ca și punct de referință *colțul din stânga-sus al primului bloc pozitionat* în care noul bloc (`h1` în cazul dat) este conținut sau zona utilă a ferestrei browserului, ca în imagine, dacă un astfel de bloc nu există.

În consecință, pentru ca un bloc pozitionat absolut să aibă ca și punct de referință un bloc exterior (de obicei blocul care conține întreaga pagină), definiția blocului exterior trebuie să conțină proprietatea `position`:

```
#bloc_ext {
    width:800px;
    text-align:left;
    background-color:#91d38d;
    position:relative;
    margin:0 auto;
}
```

O astfel de definire a blocului exterior va determina de exemplu ca un bloc interior acestuia a cărui poziție este definită prin regula: `{position: absolute; top:0; left:0;}` să fie poziționat în colțul din stânga-sus al blocului exterior și nu în colțul ferestrei browserului.

Exemplu:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<title>Blocuri</title>
<style type="text/css">
html {background-color:yellow}
body {background-color:#c83737;}
* {padding:0; margin:0;}
#bloc_ext {
    position:relative;
    width:600px;
    text-align:left;
    background-color:silver;
    margin:0 auto;
}
p{color: white; background-color:#000080; margin:15px; font-family: arial, verdana, sans-serif;}
img.stg{margin-right: 10px; float:left;}
.suprapus {position:absolute; top:0; left:0; background-color:teal; margin:25px; color: white;
font-family: arial, verdana, sans-serif;
}
</style>

<body>
<div id="bloc_ext">
<h1 class="suprapus">Cafeaua și Java</h1>
<p>Java este un limbaj de programare orientat pe obiecte asemanator limbajului C++. El a fost dezvoltat de firma Sun, prima semnalare a apariției sale datând din 1991. </p>
<p>
Conducatorul proiectului, Green James Gosling, este considerat parintele limbajului Java. Noului limbaj i s-a dat la început numele Oak (eng. stejar) dar acesta a fost revendicat de o altă firmă și realizatorii l-au schimbat în Java, aluzie la sursa băuturii favorite a membrilor echipei - cafeaua.
</p>
</div>
</body>
</html>
```

Rezultat:



Se observă că blocul `<h1>` este plasat peste primul paragraf. Faptul că `<p>` este definit ca având `margin:15px` determină deplasarea în jos a blocului `body` a cărui frontieră `top` coincide cu frontieră primului paragraf. Această deplasare afectează evident și blocul `bloc_ext` inclus în `body`. Pentru a evita această deplasare primul paragraf ar trebui să aibă ca și `body`, proprietatea `margin:0`, deci el trebuie să fie definit ca aparținând unei clase aparte având această caracteristică.

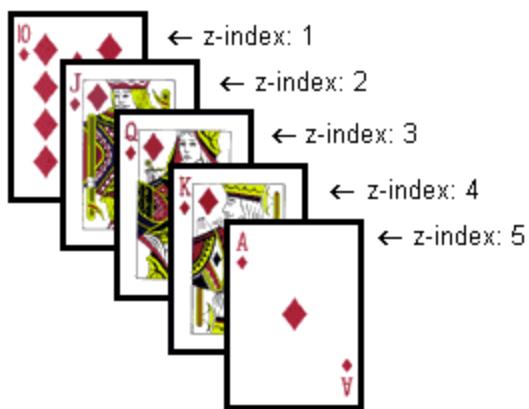
```
p.primul {margin-top:0;}  
...  
<p class="primul"> Java este un limbaj de programare orientat ... </p>
```



2.18. Suprapunerea blocurilor

Pozitionarea absolută a blocurilor permite plasarea acestora oriunde în planul xoy atașat zonei client a browserului. Dar CSS permite și specificarea poziției pe axa Oz (perpendiculară pe planul monitorului) și implicit suprapunerea blocurilor.

Un exemplu elementar este reprezentat în figura de mai jos. Cărțile de joc afișate sunt disponibile ca imagini individuale, dar trebuie afișate parțial suprapuse și într-o ordine precizată. Pentru a poziționa pe axa z un bloc se va folosi proprietatea *z-index* a acestuia. Modul de utilizare a acesteia decurge din exemplul considerat.



```

div.ten_of_diamonds {
    position: absolute;
    left: 100px;
    top: 100px;
    z-index: 1;
}

div.jack_of_diamonds {
    position: absolute;
    left: 115px;
    top: 115px;
    z-index: 2;
}

div.queen_of_diamonds {
    position: absolute;
    left: 130px;
    top: 130px;
    z-index: 3;
}

div.king_of_diamonds {
    position: absolute;
    left: 145px;
    top: 145px;
    z-index: 4;
}

div.ace_of_diamonds {
    position: absolute;
    left: 160px;
    top: 160px;
    z-index: 5;
}

```

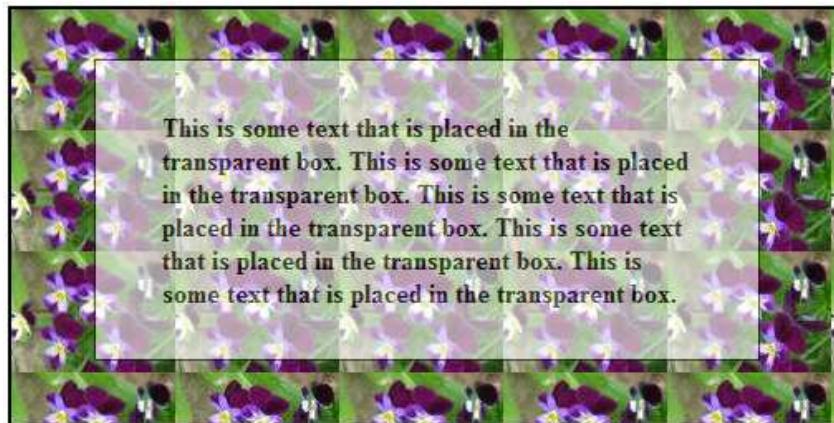
Proprietatea *z-index* poate avea și valori negative, situație în care blocurile sunt plasate sub planul paginii web. Planul ferestrei browserului are *z-index:0*.

2.19. Alte proprietăți ale blocurilor

Pe lângă cele prezentate, CSS oferă și alte proprietăți. În continuare vor fi prezentate două dintre ele, frecvent folosite de realizatorii de situri web.

Proprietatea *opacity* permite definirea unui efect de transparentă a unui bloc. Valoarea introdusa pentru această proprietate este un număr subunitar, valoarea însemnând 0 - lipsă transparentă, 1 – transparentă totală.

Exemplu :



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<style type="text/css">
div.background
{
width: 500px;
height: 250px;
background: url(flori.jpg) repeat;
border: 2px solid black;
}
div.transbox
{
width: 400px;
height: 180px;
margin: 30px 50px;
background-color: #ffffff;
border: 1px solid black;
/* pentru Internet Explorer */
filter:alpha(opacity=60);
/* CSS3 standard */
opacity:0.6;
}
div.transbox p
{
margin: 30px 40px;
font-weight: bold;
color: #000000;
}
</style>
</head>
<body>
<div class="background">
<div class="transbox">
<p>This is some text that is placed in the transparent box.
This is some text that is placed in the transparent box.
This is some text that is placed in the transparent box.

```

This is some text that is placed in the transparent box.
This is some text that is placed in the transparent box.

```
</p>
</div>
</div>
</body>
</html>
```

Proprietatea *visibility* permite definirea unor blocuri al căror conținut este afișat la producerea unui eveniment. Proprietatea *visibility* poate lua valorile *visible* sau *hidden*. Întrucât schimbarea valorii proprietății se realizează dinamic, folosind o secvență de cod programată în JavaScript, exemplificarea utilizării acestei proprietăți se va face la sfârșitul cursului, după introducerea unor noțiuni de programare în JavaScript.

2.20. Formatarea listelor

CSS pune la dispoziția utilizatorilor o serie de proprietăți care permit modificarea radicală a modului de afișare a unei liste. Folosind aceste proprietăți elementele unei liste pot fi ușor aranjate sub forma unui meniu folosit la navigarea în sait.

In HTML listele sunt plasate decalat față de marginea din stânga a blocului în care sunt inserate. Deoarece mărimea acestei decalări și modul de realizare (*padding* pentru Mozilla, Netscape, Safari sau *margin* pentru Internet Explorer, Opera), pentru a evita problemele de plasare a listelor, la specificarea regulilor de plasare a listelor (selectorul *ul*) proprietățile *padding* și *margin* trebuie impuse.

```
ul {margin: 0; padding: 0;}
```

Proprietatea *list-style-type* este folosită pentru a impune tipul de marcator folosit înaintea fiecărui element din listă.. Valorile posibile sunt *none* (suprimare marcatori), *disc*, *circle* sau *square* pentru liste neordonate (**) respectiv *decimal*, *lower-alpha*, *upper-alpha*, *lower-roman* sau *upper-roman* pentru liste ordonate (**).

Exemple :

```
<ul>
  <li>Unix</li>
  <li>Windows</li>
  <li>Linux</li>
</ul>
```

ul { list-style-type:disc ; }	<ul style="list-style-type: none"> • Unix • Windows • Linux
ul {list-style-type :circle ;}	<ul style="list-style-type: none"> ○ Unix ○ Windows ○ Linux
ul {list-style-type :square ;}	<ul style="list-style-type: none"> ■ Unix ■ Windows ■ Linux

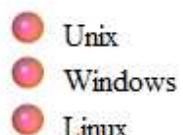
```
<ol>
  <li>Unix</li>
  <li>Windows</li>
  <li>Linux</li>
</ol>
```

ol {list-style-type :decimal ;}	1. Unix 2. Windows 3. Linux
ol {list-style-type :lower-alpha ;}	a. Unix b. Windows c. Linux
ol {list-style-type :upper-alpha ;}	A. Unix B. Windows C. Linux
ol {list-style-type :lower-roman ;}	i. Unix ii. Windows iii. Linux
ol {list-style-type :upper-roman ;}	I. Unix II. Windows III. Linux

Proprietatea *list-style-image* permite înlocuirea marcatorilor standard cu o imagine.

Exemplu de utilizare:

```
ul { list-style-image: url("imagini/bulina.jpg"); }
```



Proprietatea *display :inline* aplicată simultan selectorilor *ul* și *li* permite dispunerea elementelor liste pe orizontală.

```
ul,li {display: inline ;}
```

Exemplu:

Se dorește realizarea unei liste în care elementele să fie dispuse orizontal, fiecare element fiind în același timp și referință spre o altă pagină.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<title>Liste</title>
<style type="text/css">
ul {list-style-type :none; margin: 0; padding: 4px 20px 4px 40px; }
```

```

ul,li {display: inline;}
ul li a {font-family:arial, verdana, sans-serif; padding: 4px; background-color: teal;
          text-decoration:none; color:white; }
ul li a:hover {text-decoration:underline; color:yellow;}
</style>
</head>

<body>
<h1>Lista disputa orizontal</h1>
<ul>
<li><a href="#">Unix</a></li>
<li><a href="#">Windows</a></li>
<li><a href="#">Linux</a></li>
</ul>
</body>
</html>

```

Rezultat :

Unix
Windows
Linux

Dacă diferența evidentă de lățime a blocurilor afișate astfel constituie o problemă se poate modifica descrierea stilurilor astfel:

```

<style type="text/css">
ul {list-style-type: none; margin: 0; padding: 0;}
li {
  float:left;
  width:80px;
  padding: 4px;
  margin: 4px;
  background-color: teal;
  background-image:url("capat_lista.png");
  background-repeat: no-repeat;
  background-position: top-left;
  font-family: arial, verdana, sans-serif;
  text-align: center;
}
ul,li {display: inline;}
ul li a {color: white; text-decoration: none;}
ul li a:hover {color: yellow;text-decoration: underline; }
.decalat {margin-left:100px;}
</style>
</head>

<body>
<h1>Lista de prioritati</h1>
<ul>
  <li class="decalat"><a href="#">Unix</a></li>
  <li><a href="#">Windows</a></li>

```

```

<li><a href="#">Linux</a></li>
</ul>
</body>
</html>

```

Rezultat :



Notă : Deplasarea în direcție orizontală a listei s-a realizat prin impunerea clasei *.decalat* pentru primul element din listă. Clasa *.decalat* are proprietatea *margin-left: 100px*.

Pentru decuparea colțului blocului *li* care conține elementele listei s-a plasat pe fundalul acestuia imaginea (capat_lista.png).

2.21. Selectorii contextuali

În exemplele precedente apare un mod de a scrie selectorii diferit de cele folosite până în prezent. Exemplu:

```
#nav ul li a {color: white; text-decoration: none;}
```

Într-un astfel de caz selectorii trebuie citiți de la dreapta la stânga. În exemplul dat se precizează regulile de afișare a unei referințe (*<a>*) dar se dorește ca regulile să se aplice numai referințelor *<a>* incluse în blocuri **, incluse la rândul lor în liste neordonate (**) conținute în blocul identificat prin *#nav*.

Scrierea aceasta poate părea complicată dar este o soluție în a preciza regulile de afișare numai a unui anumit element din pagină.

Observație: Dacă în locul spațiului selectorii sunt despărțiti prin virgule se obține un grup de selectori. Regulile date după enumerarea selectorilor care formează grupul se aplică tuturor elementelor din grup. Exemplu:

```
p,h1,h2,h3 {color: white; text-decoration: none; margin: 10px; padding: 0;}
```

În cazul în care în grupul de selectori apar și selectori contextuali sintaxa este următoarea:

```
div#news h3, div#news ul {margin: 0 2em; }
```

2.22. Formatarea tabelelor

Un tabel trebuie să fie privit ca fiind un bloc (`<table>`) care conține un ansamblu de linii (blocuri `<tr>`), fiecare linie fiind compusă din blocuri elementare (`<th>` sau `<td>`). Descrierea fiecăreia dintre aceste blocuri se realizează folosind proprietățile folosite deja anterior pentru blocuri `<p>`, `<h>`, `<div>` etc.

Pentru ameliorarea aspectului unui tabel sunt esențiale culorile aplicate pe fundal, aspectul liniilor despărțitoare și distanțele dintre conținuturile celulelor și liniile despărțitoare.

Exemplu :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<title>Tabele</title>
<style type="text/css">
    table {border-collapse: collapse; border: 4px ridge blue; }
    th {background-color:#bbbbbb; border: 1px solid blue; width:80px; padding:4px;}
    td {border: 1px solid blue; text-align: center; padding:4px;}
</style>
</head>

<body>
<h1>Tabele</h1>
<table>
    <tr>
        <th>Nr. crt</th> <th>Valori</th> <th>2008</th> <th>2009</th>
    </tr>
    <tr>
        <td>1</td> <td>Manopera</td> <td>1.232</td> <td>1.230</td>
    </tr>
    <tr>
        <td>2</td> <td>Materiale</td> <td>1.678</td> <td>2.01</td>
    </tr>
</table>
</body>
</html>
```

Nr. crt	Valori	2008	2009
1	Manopera	1.232	1.230
2	Materiale	1.678	2.01

În plus față de proprietățile cunoscute deja și aplicabile oricărui bloc, la formatarea tabelelor intervin câteva proprietăți suplimentare :

Proprietate	Valori, explicație
<i>border-collapse</i>	<i>collapse / separate</i> Impune unirea / separarea frontierelor celulelor.
<i>border-spacing</i>	<i>dimensiune</i> Aplicabil dacă border-collapse:separate.
<i>caption-side</i>	<i>top / bottom / left</i> Specifică poziția titlului tabelului (definit prin <caption> ... </caption>)

Capitolul III Aplicații WEB

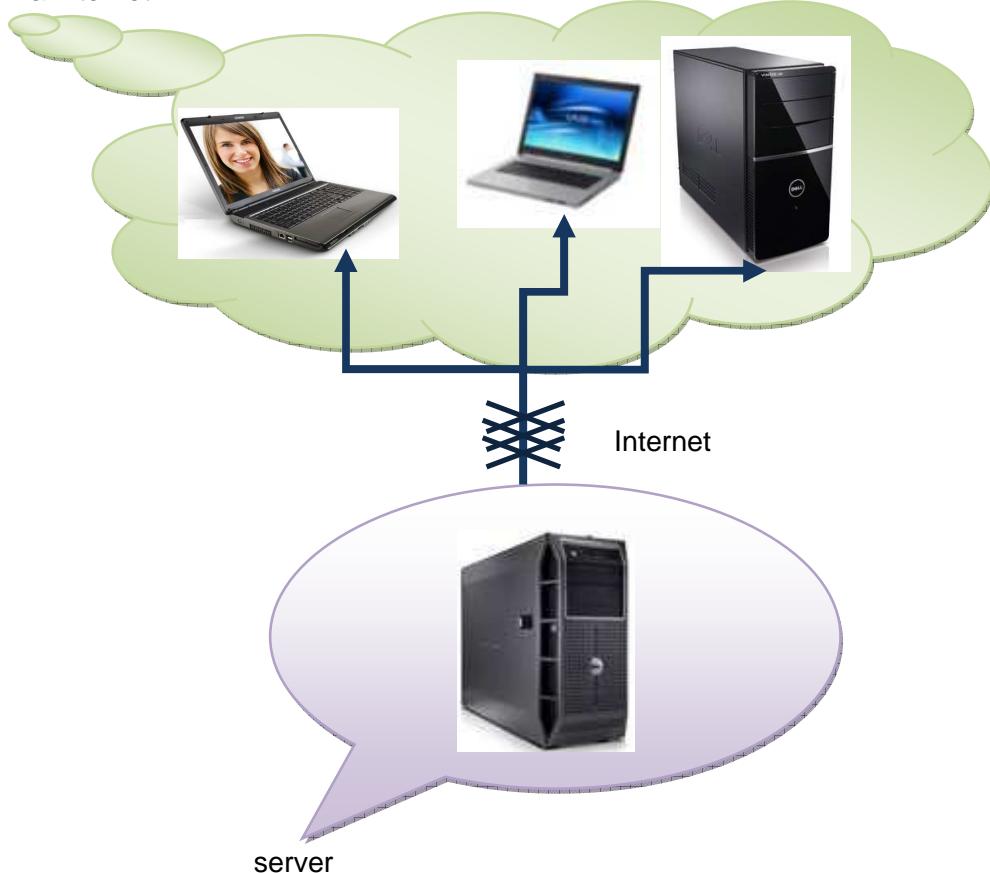
3.1 Generalități

În ingineria software o aplicație web este definită ca fiind o aplicație care este accesată prin intermediul unui browser web. Spre deosebire de aplicațiile obișnuite, care pot fi executate numai după obținerea unui kit de instalare și instalarea pe calculator, aplicațiile web pot fi executate imediat ce interfața expusă de acestea s-a încărcat în fereastra browserului. Această caracteristică precum și accesibilitatea, dată de folosirea rețelei Internet pentru a accesa aplicațiile, au asigurat succesul rapid al acestei tehnologii.

Componentele software ale unei aplicații web sunt păstrate și executate pe un calculator conectat la Internet care are rol de server. Unele componente sunt specifice acelei aplicații iar altele sunt aplicații generale, partajate de mai multe aplicații web. În a doua categorie intră de exemplu serverele de baze de date care au rolul de a păstra datele diferitelor aplicații.

calculatoare conectate

la Internet



Stațiile client accesează aplicațiile web prin intermediul browserelor, deci aplicațiile web trebuie să transmită stațiilor client informații codificate în *html*.

Pe server rulează permanent o aplicație de tip server care realizează trimiterea prin rețea a paginilor web cerute de stațiile client. Cea mai răspândită aplicație de acest tip este *Apache* (www.apache.org) , care este și gratuită.

Pe lângă trimitera paginilor web, Apache poate realiza și lansarea în execuție a diferitelor componente ale unei aplicații web.

Notă: Atenție la termenul *server!* Când se referă la un calculator, acesta are niște caracteristici aparte respectiv este conectat într-o rețea și pe el rulează continuu un ansamblu de aplicații care nu aparțin unui utilizator anume ci așteaptă comenzi care vin prin intermediul rețelei de la aplicații client. Dacă termenul se referă la un program, termenul indică o aplicație care rulează continuu și așteaptă ca o altă aplicație, denumită *aplicație client*, să-i transmită o comandă. De cele mai multe ori aplicațiile de tip server rulează pe servere iar aplicațiile client rulează pe calculatoare conectate la rețea.

3.2 Marcaje pentru realizarea formularelor

Cunoștințele de *html* prezентate deja permit afișarea în fereastra browser-ului a secvențelor de text și a imaginilor. În cele ce urmează va fi prezentată descrierea în *html* a controalelor windows de bază: casete de text, butoane de diferite feluri, controale cu listă etc. destinate introducerii datelor care urmează să fie prelucrate într-o aplicație web.

Marcajul `<form>... </form>` servește la gruparea controalelor Windows destinate trimiterii de informații spre o aplicație web. În cele ce urmează zona din pagina web conținută între marcaje `<form> ... </form>` va fi denumită *formular*.

Marcajul `<form>` are două atrbute obligatorii :

- *method* definește modul în care sunt transmise datele serverului. Atributul poate lua două valori: *method="post"* dacă datele provenite din controalele Windows vor fi citite de aplicația web ca și fișier standard de intrare (*stdin* în limbajul C); *method="get"* dacă datele din controalele Windows ale formularului vor fi transmise aplicației web printr-un sir de caractere.
- *action* indică denumirea componentei software a aplicației web care va prelucra datele trimise. Această componentă software este lansată în execuție de aplicația de tip server pentru web (Apache de exemplu).

Pentru scrierea componentelor aplicațiilor web în ultimii ani s-au dezvoltat diferite limbiage, cele mai răspândite fiind limbiagile PHP și PERL. Ambele sunt limbiage interpretate (execuția instrucțiunilor are loc într-un interpretor de comenzi).

Deoarece o componentă software asociată unui formular poate conține comenzi destinate altor aplicații de pe server, ea are caracteristicile unui *script*. Din acest motiv secvențele de cod scrise de exemplu în PHP în care se vor realiza prelucrările datelor din formulare sunt denumite *scripturi PHP*.

Exemplu :

```
<form method="post" action="programe/calcul.php">
```

Marcajul <input> definește caracteristicile unui câmp al formularului. Atributele marcajului <INPUT> sunt :

type=tip definește tipul controlului Windows

name=nume definește numele simbolic al valorii câmpului

value=valoare definește conținutul prestabilit al câmpului

size=valoare definește lungimea în caractere a unui control de tip *text* sau *password*

checked se folosește pentru un buton radio dintr-un grup de butoane sau pentru o casetă de validare pentru a impune starea *selectat* ;

maxlength=m numărul maxim de caractere acceptate într-un control de tip *text* sau *password*.

Pentru modificarea aspectului unui control <input> se poate adăuga stilurilor definite o regulă suplimentară folosind selectorul *input*. În cazuri mai deosebite de formatare se pot defini clase care pot fi asociate controalelor prin adăugarea atributului *class*. Formularele prezentate în continuare vor conține modificări ale modului normal de afișare a controalelor Windows folosind stiluri.

HTML 4 permite definirea următoarelor tipuri de câmpuri: *button*, *checkbox*, *file*, *hidden*, *image*, *password*, *radio*, *reset*, *submit* și *text*.

- a. **Tipul text** permite definirea unui câmp pentru introducerea unui sir de caractere :

```
<input type="text" name="cod_numeric" />
```

Adăugarea atributului *maxlength* permite limitarea lungimii sirului de caractere permis.

- b. **Tipul password** (parolă) permite introducerea unei parole. Caracterele introduse nu vor fi afișate, în câmp apărând doar caractere * :

```
<input type="password" name="parola" />
```

- c. **Tipul checkbox** (casetă de validare) permite desenarea unei casete izolate de validare:

```
<input type="checkbox" name="clientnou" checked value="da" />
```

De regulă caseta de validare este inclusă între marcaje <label> ... </label> pentru a ataşa casetei un sir de caractere.

<label> <input type="checkbox" name="clientnou" value="da" checked /> **Client nou** </label>

Client nou

Efectul validării casetei va fi includerea în sirul de caractere trimis spre server a sirului "clientnou=da".

- d. **Tipul radio** permite descrierea unui grup de butoane radio. Gruparea se realizează prin atribuirea aceluiași nume tuturor controalelor care formează grupul. Pentru a adăuga în dreptul fiecărui buton un sir de caractere (etichetă) descrierea controlului va fi inclusă între marcaje <label> ... </label>. Exemplu:

<label><input type="radio" name="semestrul" checked value="0" /> I (octombrie-februarie) </label>
<label><input type="radio" name="semestrul" value="1" /> II (martie-iunie)</label>

- e. **Tipul buton submit** are ca și efect transmiterea spre server a valorilor introduse în câmpurile formularului. Un formular poate conține mai multe butoane de tip *submit*. Exemplu:

<input type="submit" name="gata" value ="OK" />

La apăsarea butonului din exemplu, în sirul de caractere trimis scriptului asociat formularului va fi inclusă secvența "gata=OK".

Exemplu de pagină web care conține un formular:

Email:	 mdamian@google.com
Password:	*****
Remember Me:	<input checked="" type="checkbox"/>
Login	

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>Exemplu</title>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
<style type="text/css">
```

```

div#login {
    margin: 0 0 0 100px; color: #000;
    font-family: Verdana, Arial, Helvetica, sans-serif;
    width: 300px;
}

div#login input {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    font-size: 12px;
    width: 160px;
    padding: 4px 4px 4px 4px;
    background-color: #666699;
    color:white;
}

div#login .casetta1 {
    background-image:url("om.jpg");
    background-position:top left;
    background-repeat:no-repeat;
    padding: 4px 4px 4px 30px;
}

div#login .simplu {
    padding: 4px 30px 4px 4px;
}

div#login .buton { width:80px; }

div#login table {
    width: 100%;
    border: 2px solid rgb(215,215,215);
    border-collapse: collapse;
    background-color: #cccccc;
}

div#login td {padding:5px; border: 1px solid #ffffff; }

.centrat {text-align:center;}
}

</style>
</head>
<body>
<h1>Formulare</h1>
<div id="login">
<form method="post" action="programme/log.php">
<table>
<tr>
<td>Email:</td><td><input type="text" name="eml" class="casetta1" /></td>
</tr>
<tr>
<td>Password:</td><td><input type="password" name="passwd" class="simplu" /></td>
</tr>
<tr>
<td colspan="2"><label>Remember Me:</label><input type="checkbox" name="rmbm" checked="checked" value="true" /></td>

```

```

</tr>
<tr>
<td colspan="2" class="centrat"><input type="submit" name="gata" value ="Login" class="buton" /></td>
</tr>
</table></form>
</div>
</body>
</html>

```

Pentru a simplifica alinierea informațiilor din formular, conținutul acestuia a fost plasat într-un tabel.

- f. **Tipul buton reset** determină crearea unui buton a cărui acționare permite utilizatorului să steargă valorile din toate câmpurile formularului.

```
<input type ="reset" value="Anulare" />
```

- g. **Tipul image** permite includerea în formular a unei imagini având la selectare un comportament similar unui buton de tip *submit*:

```
<input type="image" src="imagini/poza.gif" alt="Validare" />
```

- h. **Tipul hidden** permite transmiterea unor informații spre server fără ca utilizatorul să le vadă pe ecran. De regulă câmpurile *hidden* servesc la realizarea unei sesiuni de lucru. Aceasta presupune înlățuirea coerentă a unui ansamblu de pagini, de regulă realizate dinamic de scripturile aplicației web. O astfel de aplicație va memora în atributele *value* din câmpuri *hidden* informații necesare următorului script lansat în execuție ca urmare a apăsării unui buton de tip *submit*.

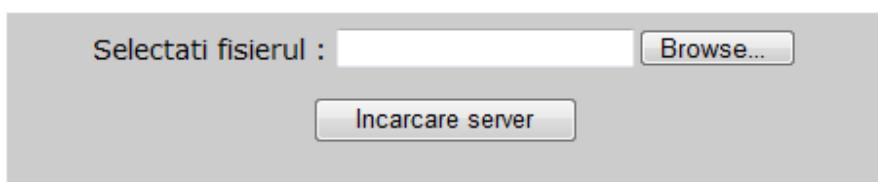
- i. **Tipul file** permite încărcarea unui fișier pe server. Controlul de tip *file* afișează o casetă de text și un buton pe care scrie *Browse*. După ce se apasă butonul *Browse* și se selectează fișierul numele acestuia va fi automat afișat în caseta de text. Apăsarea butonului de tip *submit* al formularului determină transferul fișierului pe server, într-un director temporar. Scriptul asociat formularului va realiza mutarea fișierului din directorul temporar într-un director al saitului, eventual după o validare în care se verifică tipul și dimensiunea fișierului.

Exemplu :

```

<form enctype="multipart/form-data" action="prelucrez.php" method="post">
  <input type="hidden" name="MAX_FILE_SIZE" value="30000">
  <p>Selectati fisierul : <input type="file" name="fisier" /></p>
  <input type="submit" value="Incarcare server">
</form>

```

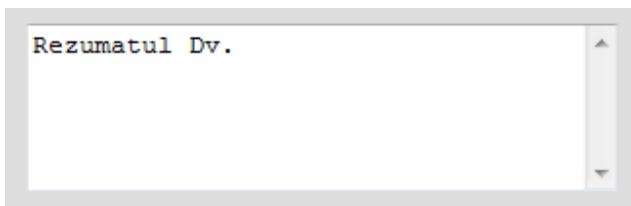


Atributul `enctype` indică natura informației trimise spre server. Valoarea implicită a acestui parametru este `enctype="application/x-www-form-urlencoded"` și corespunde formularelor obișnuite care trimit spre server un sir de caractere. În cazul transmiterii unui fișier atributul trebuie schimbat în `enctype="multipart/form-data"`.

Fișierul astfel transmis poate fi de orice tip. Pe lângă controlul de tip `file` formularul poate conține și alte controale conținând diverse informații care trebuie trimise împreună cu fișierul.

Marcajul `<textarea>` permite definirea unei zone în care se poate scrie un text mai lung, dispus pe mai multe linii. În acest mod se definesc de exemplu zonele destinate scrierii unor mesaje, impresii etc. Marcajul `<textarea>` are trei attribute: `name`, `rows` (câte linii) și `cols` (câte coloane).

```
<textarea name="rezumat" rows="5" cols="35">Rezumatul Dv.</textarea>
```

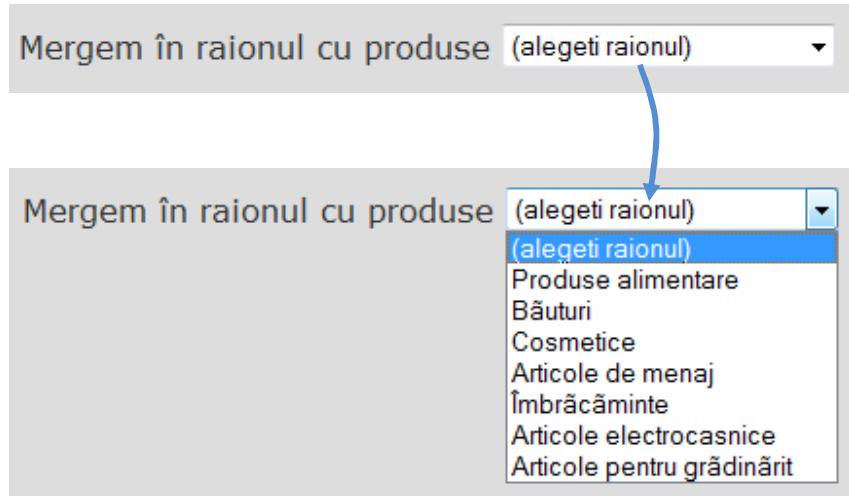


Marcajul `<select>` permite definirea controalelor complexe de tip text plus listă (*combo box*). Caseta de tip combobox afișează o valoare. Selectarea acesteia sau a săgeții din dreapta ei cu mouse-ul va provoca afișarea listei asociate, vizitatorul paginii putând selecta linia dorită.

Exemplu :

Mergem în raionul cu produse

```
<select name="raion">
  <option value="niciunde" selected>(alegeti raionul)
  <option value="alimente">Produse alimentare
  <option value="bautura">B&atilde;uturi
  <option value="cosmetice">Cosmetice
  <option value="menaj">Articole de menaj
  <option value="imbrac">&Icirc;mbr&atilde;c&atilde;minte
  <option value="electro">Articole electrocasnice
  <option value="gradina">Articole pentru gr&atilde;din&atilde;rit
</select>
```



În exemplul dat, dacă se selectează raionul *Produse alimentare*, sirul de caractere trimis spre server în urma apăsării butonului de tip *submit* va conține secvența "raion=alimente".

În cazul în care trebuie realizat un control de tip listă derulantă, care de regulă permite selectarea mai multor opțiuni, se va utiliza atributul *size* care precizează numărul de opțiuni vizibile în cadrul controlului și *multiple*, în cazul în care controlul admite selectare multiplă. Pentru selectarea multiplă se va ține tasta "Ctrl" apăsată.

Exemplu:

```
<p>Ce marcă de mașină ati dori să aveți (puteti selecta mai multe)?</p>
<p><select name="cars" multiple size="4">
  <option value="dacia"> Dacia
  <option value="renault"> Renault
  <option value="vw"> Volkswagen
  <option value="audi"> Audi
  <option value="opel"> Opel
  <option value="fiat"> Fiat
  <option value="seat"> Seat
</select> </p>
```

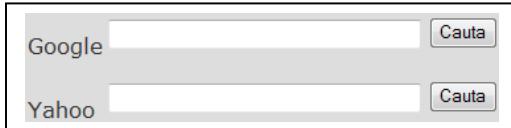
Atributul *size="4"* provoacă afișarea pe ecran a 4 dintre opțiunile indicate. Dacă atributul *size* lipsește, caseta va conține toate valorile din lista de opțiuni.

Formularele prezentate au atributul *method="post"*. Atributul *method="get"* oferă posibilitatea lansării unor scripturi aflate pe servere de căutare, de exemplu Google sau Yahoo.

Exemplu :

<http://www.google.ro/search?hl=ro&q=xhtml+tutorial>

```
<table>
<tr>
<td> Google </td>
<td><form action="http://www.google.ro/search" method="get">
    <input type="hidden" value="ro" name="hl" />
    <input type="text" size="35" name="q" />
    <input type="submit" value="Cauta" /></form>
</td>
</tr>
<tr>
<td> Yahoo </td>
<td><form action="http://www.yahoo.com/search" method="get">
    <input type="text" size="35" name="p" />
    <input type="submit" value="Cauta" /></form>
</td>
</tr>
</table>
```



Întrucât şirul de caractere care trebuie trimis diferă pentru serverele de căutare menŃionate, este esenŃială transcrierea corectă a formularelор.

3.3 XAMPP

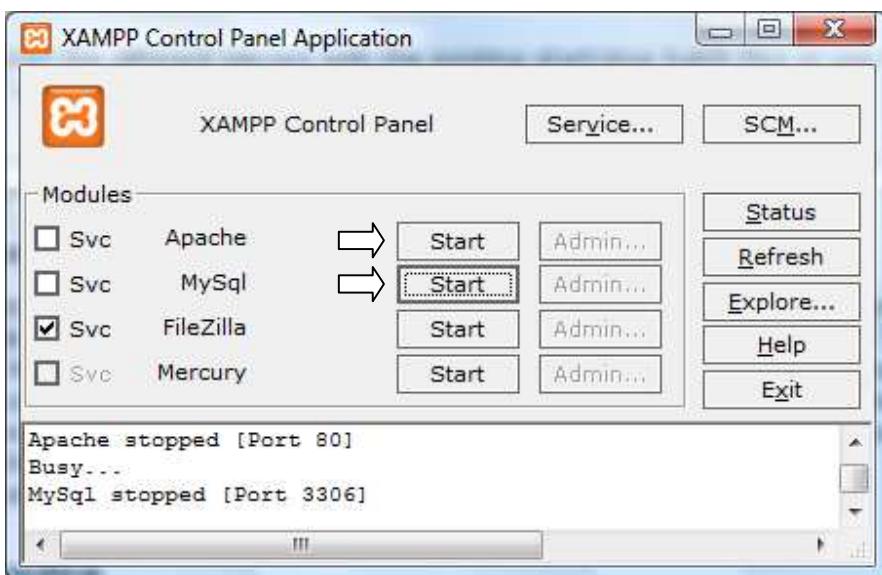
XAMPP este numele dat unui pachet de aplicații care constituie infrastructura software necesară găzduirii saiturilor web: server de web (*Apache*), server de baze de date (*MySQL*), interpretoare pentru scripturi scrise în limbajele PHP și PERL.

Pentru a putea fi folosit pe servere care găzduiesc saituri, după instalarea pachetului de aplicații trebuie operate unele configurații, mai ales pentru ameliorarea securității.

XAMPP a fost creat însă pentru a pune la dispoziția dezvoltatorilor un instrument eficient de testare. Odată instalat pe calculatorul propriu, pachetul de aplicații va face ca acesta să aibă comportamentul unui server, permitând testarea aplicațiilor scrise fără a intra în conflict cu firma care va găzdui în final produsul software realizat.

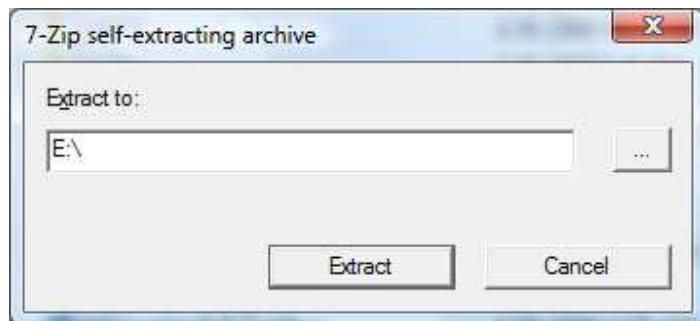
XAMPP (<http://www.apachefriends.org/en/xampp-windows.html>) are și o variantă « lite », XAMPP Lite, care poate fi utilizată imediat după descărcare și dezarchivare, fără a fi necesară instalarea.

După instalare, prezența pe calculator a pachetului de aplicații se manifestă prin aplicația XAMPP Control Panel Application.

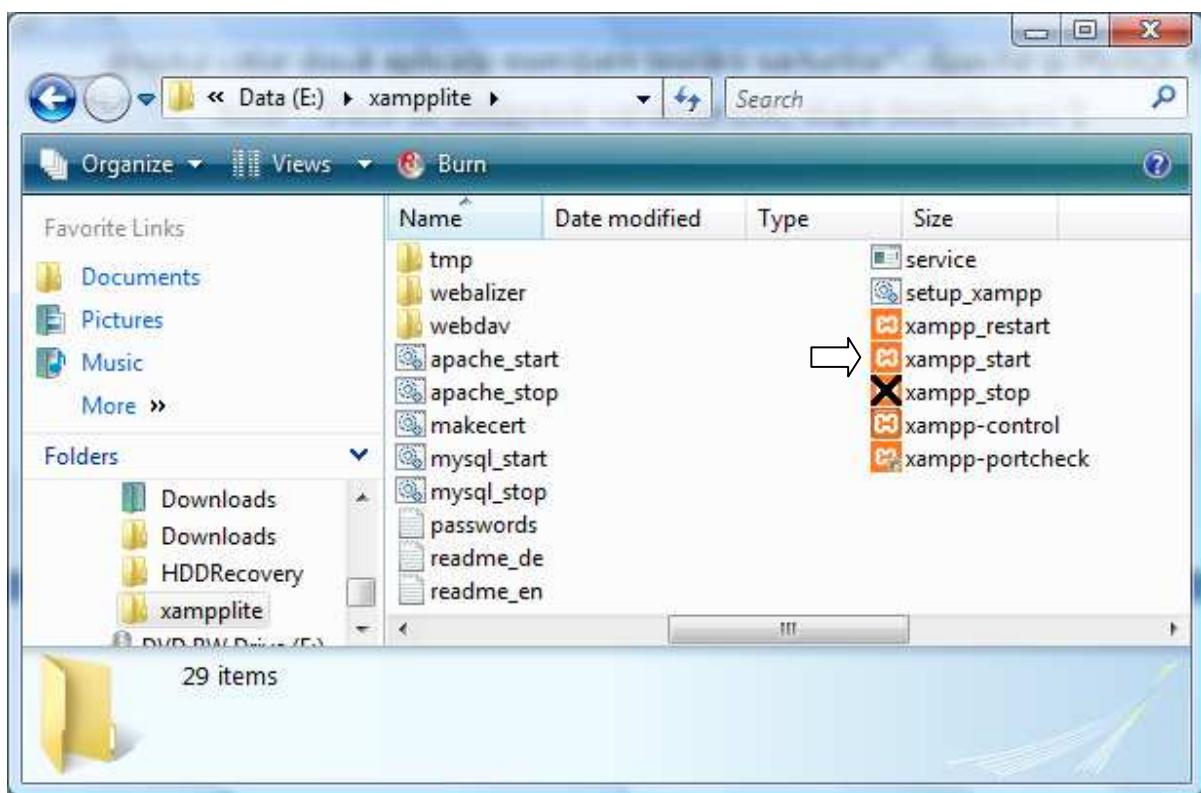


După lansarea în execuție a acestei aplicații se vor apăsa butoanele *Start* din dreptul celor două aplicații esențiale testării saiturilor : *Apache* și *MySQL*.

Notă : Dacă se utilizează varianta XAMPP Lite, se dezarchivează arhiva descărcată din Internet în rădăcina discului care va fi folosit pentru testarea saiturilor (E:\ de exemplu) :

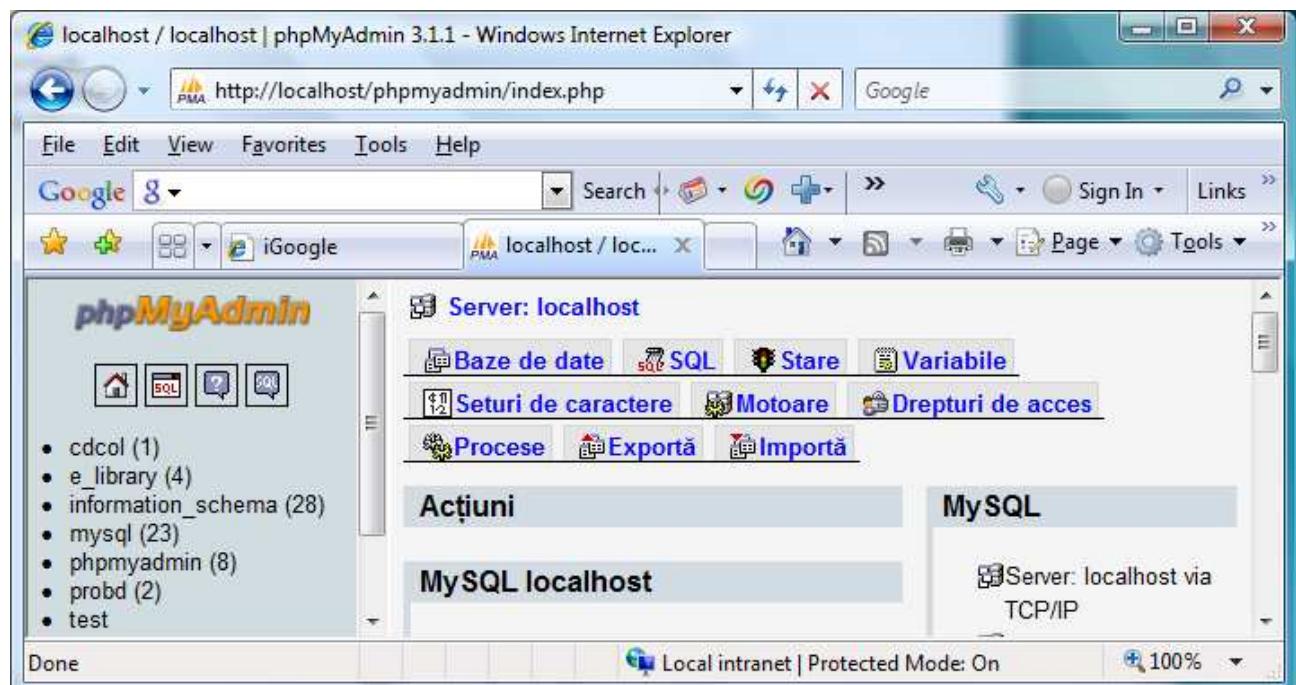


În urma desarhivării, pe discul selectat va fi adăugat directorul *xampplite*. În această variantă pornirea aplicațiilor Apache și MySQL necesare testării saiturilor se realizează selectând *xampp_start* din directorul *xampplite*.



Pentru oprirea aplicațiilor se va selecta *xampp_stop*.

Verificarea funcționării aplicațiilor necesare pentru testarea saiturilor se poate realiza tastând în browser adresa aplicației web integrate în XAMPP și destinate administrării serverului de baze de date MySQL: <http://localhost/phpmyadmin/>.



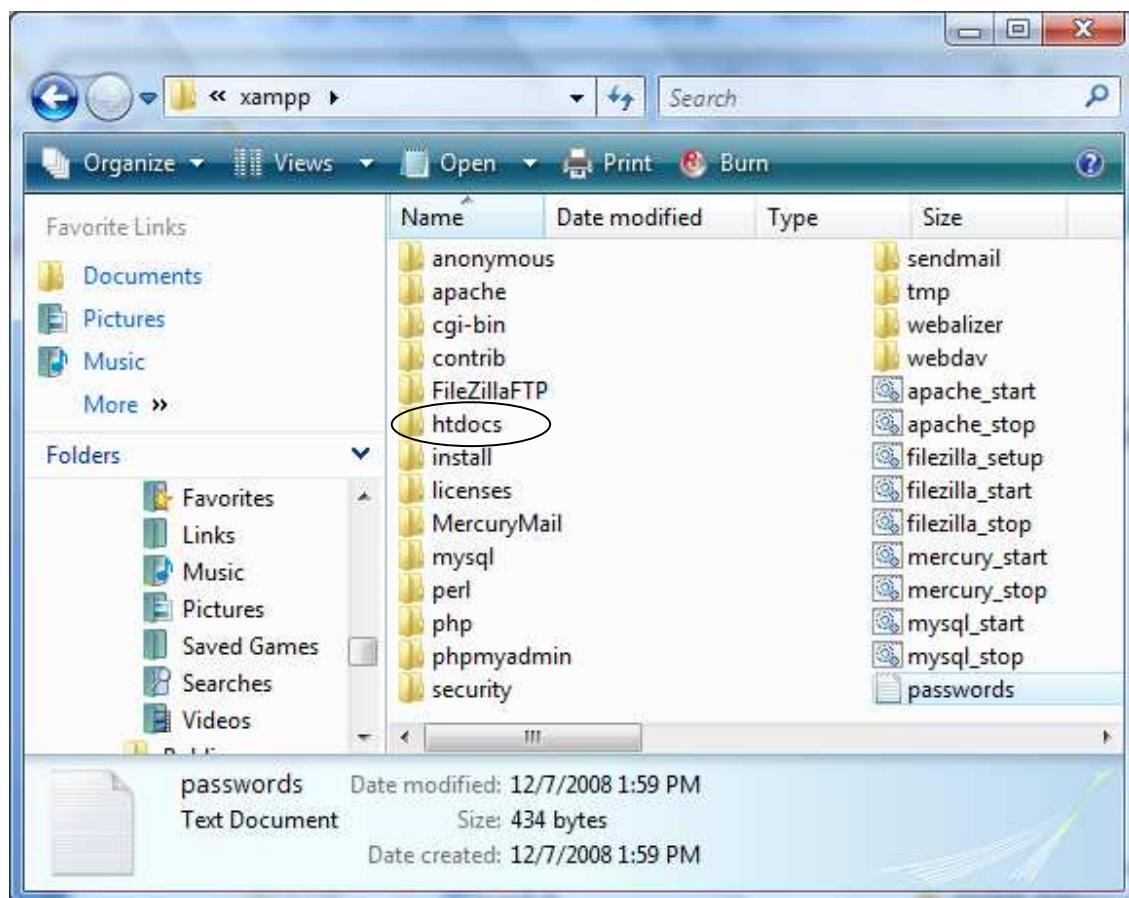
3.4 Apache

Apache (<http://www.apache.org/>) este o aplicație de tip server pentru web. O aplicație de acest tip este un program reactiv. El rulează în continuu pe calculatorul destinat păstrării unuia sau mai multor saituri web și așteaptă cereri din partea unei aplicații client (*Internet Explorer, Mozilla Firefox, Opera etc.*).

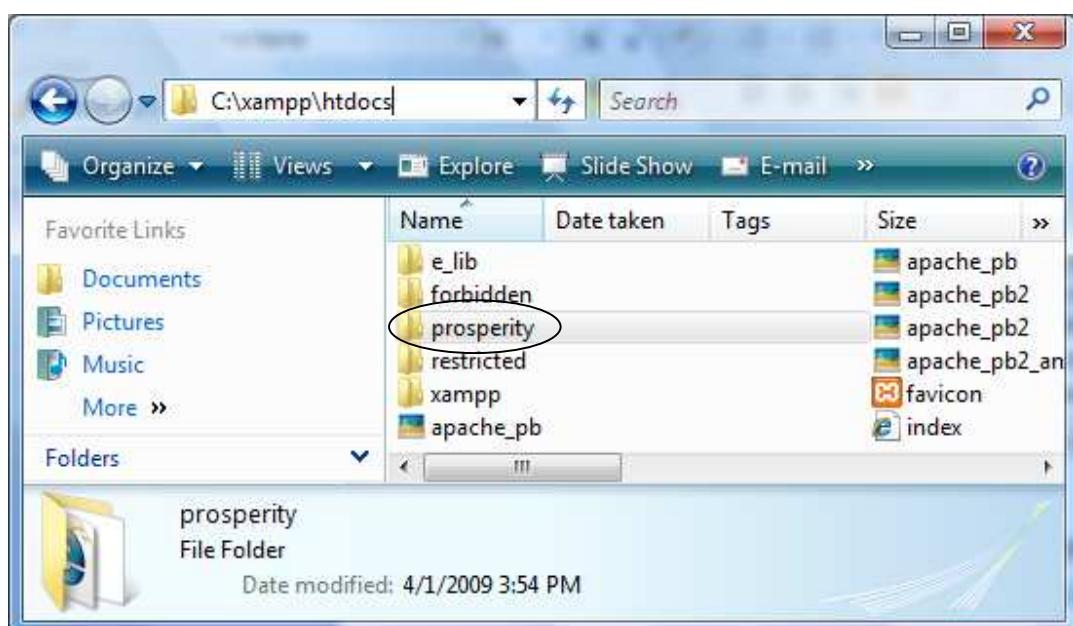
Ca aplicație, serverul pentru web acceseează un ansamblu de fișiere dispuse pe discul calculatorului pe care acesta este instalat. Dacă o aplicație client solicită un fișier existent, serverul pentru web îl va furniza respectând regulile unui protocol precizat în cererea clientului (*http*).

Paginile web sunt de regulă documente multimedia conținând text, imagini, sunet, animații. Ele sunt stocate pe disc în mai multe fișiere, fiecare fișier conținând informații de un anumit tip. Corespunzător, serverul pentru web va primi din partea aplicației client o suită de cereri de transfer de fișiere începând cu fișierul principal. Asamblarea documentului din fragmentele primite cade în sarcina aplicației client.

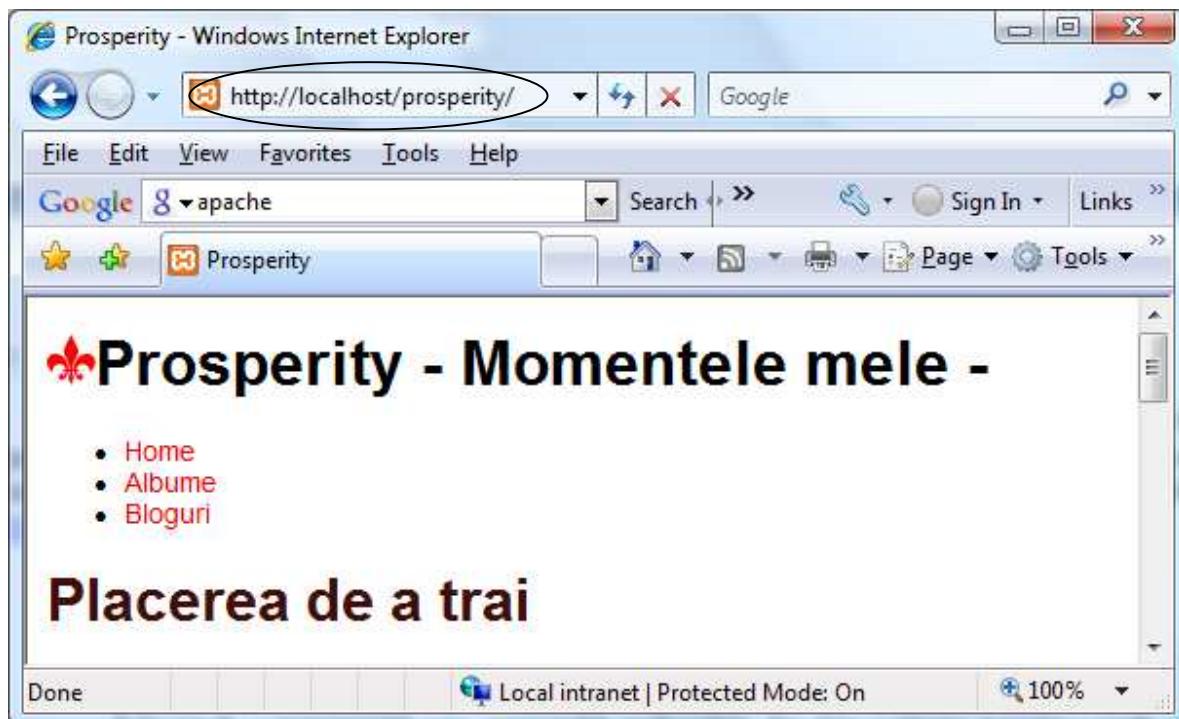
Pe serverele pentru Internet funcționând sub UNIX sau Linux, Apache este configurat astfel încât rădăcina structurii arborescente de directoare care găzduiesc saituri diferă de cea creată în urma instalării XAMPP. În cele ce urmează se va exemplifica modul de realizare a unei aplicații pentru web respectând structura de directoare din imagine, creată prin instalarea XAMPP sau disponibilă în directorul creat după dezarhivarea fișierului conținând *XAMP Lite*.



După instalarea pachetului de aplicații XAMPP (sau XAMPP Lite) se va copia în directorul **xamplite\htdocs** directorul care conține saitul.



Se vor porni apoi aplicațiile conținute în XAMPLITE (`xampp_start`) și se va tasta în caseta de text pentru adresa a browserului adresa saitului. Aceasta este <http://localhost/numesite/>, deci, pentru exemplul dat, <http://localhost/prosperity/>.



În lipsa numelui fișierului solicitat, serverul Apache va transmite browserului fișierul *index.html*.

3.5 MySQL

MySQL este un sistem de gestiune a bazelor de date relațional produs de compania suedeză MySQL AB (preluată în 2008 de Sun Microsystems) și distribuit sub Licența Publică Generală GNU. Este cel mai popular SGBD open-source și o componentă principală a pachetului de aplicații XAMPP.

Deși este folosit foarte des împreună cu limbajul PHP, cu MySQL se pot construi aplicații în orice limbaj major C, C++, C#, Borland Delphi, Java, Perl, sau Python.

Licența GNU GPL nu permite încorporarea MySQL în aplicații comerciale; cei care doresc să facă acest lucru pot achiziționa, contra cost, o licență comercială de la compania producătoare, Sun Microsystems.

Pentru a administra bazele de date MySQL se poate folosi modul *linie de comandă* sau aplicația gratuită, scrisă în PHP, *phpMyAdmin*.

MySQL poate fi rulat pe multe dintre platformele software existente: AIX, FreeBSD, GNU/Linux, Mac OS X, NetBSD, Solaris, SunOS, Windows 9x/NT/2000/XP/Vista.

În cele ce urmează MySQL va fi componenta aplicațiilor web destinață păstrării datelor aplicației. Deși limbajul PHP în care vor fi scrise secvențele de cod care prelucrează informațiile aplicației web are funcții destinate scrierii de fișiere pe disc, de regulă se apelează la varianta mult mai sigură a folosirii serverului de baze de date MySQL.

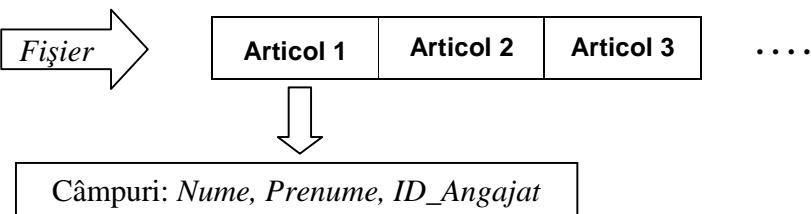
3.5.1 Date și tabele

O aplicație pentru web prelucrează date de diferite naturi: siruri de caractere, valorile numerice, date calendaristice etc. Exemple : denumiri de produse, nume de utilizatori sau de clienți, parole, adrese de e-mail, prețuri, cantități, date de livrare sau de încheiere a unei comenzi, etc. Evident pot apărea informații și de alte naturi : imagini, fișiere în format .pdf, filme, fișiere audio .mp3 etc.

Informațiile păstrate într-o bază de date relatională sunt dispuse într-un ansamblu de fișiere. Din considerente legate de creșterea vitezei de accesare a informațiilor, fișierele sunt formate din articole având aceeași structură.

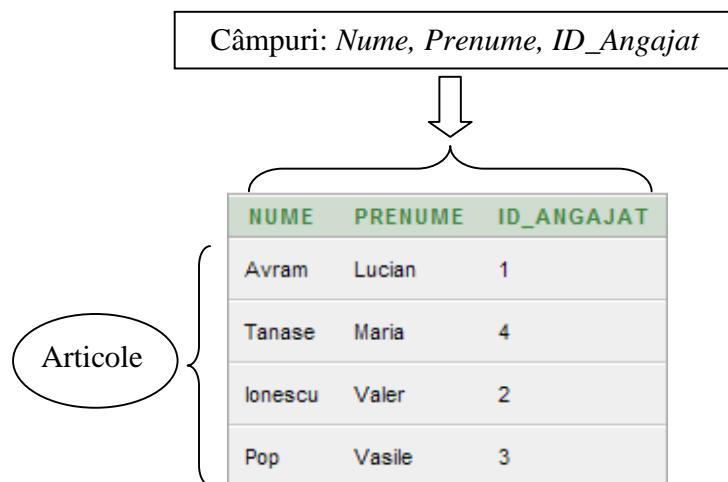
Structura articolelor unui fișier de date este definită la crearea sa, prin precizarea câmpurilor pe care le va conține fiecare articol.

Exemplu:



Un câmp dintr-un articol conține o informație elementară. Fiecare câmp are un nume, conține un tip de informație (sir de caractere, număr, dată calendaristică, fișier etc.), are o lungime de reprezentare a informației și, În cazul pentru câmpurilor numerice, are precizat numărul de zecimale.

Datorită faptului că formatul articolului este fix, frecvent se folosește pentru fișier o reprezentare tabelară și chiar se folosește pentru fișierele de date denumirea de "**tabele**".

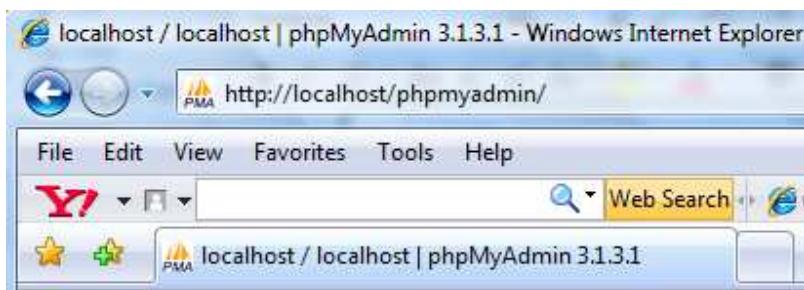


3.5.2 SQL

O aplicație de tip server este un program care așteaptă comenzi din partea unor aplicații client. În cazul serverelor de baze de date relaționale (Oracle, MySQL, IBM DB2, Microsoft Access, OpenOffice Base etc.) se folosește limbajul SQL (Structured Query Language). Deoarece prezentarea acestui limbaj depășește cadrul acestui curs, în cele ce urmează comenzile SQL vor fi construite folosind interfața grafică oferită de PHPMyAdmin, o aplicație integrată în XAMPP.

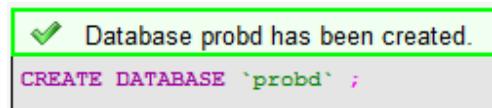
a. **Crearea unei baze de date** - comanda *create database*

Pentru deschiderea aplicației phpMyAdmin se pornește XAMPPLITE  și se tastează în caseta de text pentru adrese a aplicației Internet Explorer adresa <http://localhost/phpmyadmin/>



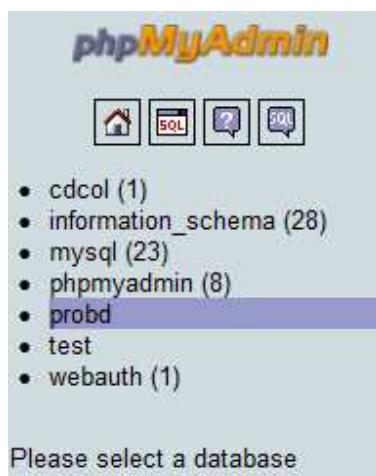
În fereastra afișată se va tasta numele noii baze de date (*probd*) și se va apăsa butonul *Create*:

phpMyAdmin va crea comanda SQL de creare a unei baze de date și o va trimite serverului MySQL. *phpMyAdmin* va afișa de fiecare dată comenziile pe care le-a trimis serverului de baze de date MySQL.



Odată creată, baza de date *prodb* va fi *bază de date curentă*. Comenziile ulterioare, de exemplu cele prin care se vor crea tabelele (fișierele) acesteia, se vor adresa acesteia.

La următoarele porniri ale aplicației *phpMyAdmin* intrarea în baza de date *prodb* se va realiza prin selectarea acesteia în arborele afișat în panoul din stânga al aplicației.



b. ***Crearea tabelelor bazei de date*** - comanda *create table*

După crearea bazei de date *prodb* se vor crea succesiv tabelele acesteia.

- ***Crearea tabelului utilizatori***

Tabelul *utilizatori* va conține datele de identificare a utilizatorilor autorizați să opereze în baza de date. Acest tabel este prezent în toate bazele de date accesibile prin Internet.

Crearea unui tabel se realizează concomitent cu definirea structurii sale.

The screenshot shows the phpMyAdmin interface for creating a new table. On the left, there is a sidebar with a dropdown menu set to "probd" and a link "probd (0)". The main area displays the message "No tables found in database." Below this, there is a form titled "Create new table on database probd". The form has two input fields: "Name: utilizatori" and "Number of fields: 3". There is also a "Go" button at the bottom right of the form.

Tabelul utilizatori are 3 câmpuri (engl. *fields*).

În vederea creării comenzi `create table`, *phpMyAdmin* va cere introducerea numelui și tipului fiecărui dintre câmpurile tabelului.

Field	ID_Utilizator	Nume	Parola
Type <small>?</small>	INT	VARCHAR	VARCHAR
Length/Values <small>1</small>		20	20
Default <small>2</small>	None	None	None
Collation			
Attributes			
Null	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Index	PRIMARY	INDEX	---
AUTO_INCREMENT	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Câmpurile tabelului Utilizatori sunt *ID_Utilizator* (*INT* - valoare întreagă), *Nume* (VARCHAR lungime maximă 20 caractere) și *Parola* (VARCHAR - lungime maximă tot 20 caractere). Tipul *VARCHAR* se folosește pentru siruri de caractere de lungime variabilă.

Pentru câmpul *ID_Utilizator* s-a mai precizat că este *PRIMARY* și s-a selectat caseta *AUTO_INCREMENT*. Prima caracteristică, *PRIMARY* indică faptul că *ID_Utilizator* este **cheie primară**. Cheia primară a unui tabel din baza de date are valori distincte pentru toate articolele din tabel și are rolul de identificator unic al acestora. Dacă ulterior se va dori suprimarea unui articol se va putea selecta articolul indicându-i cheia primară.

Proprietatea *AUTO_INCREMENT* indică modul de formare a valorilor câmpului *ID_Utilizator*. Valoarea acestuia pentru un nou articol care urmează să fie adăugat se calculează automat, prin incrementarea valorii corespunzătoare ultimului articol prezent în tabel.

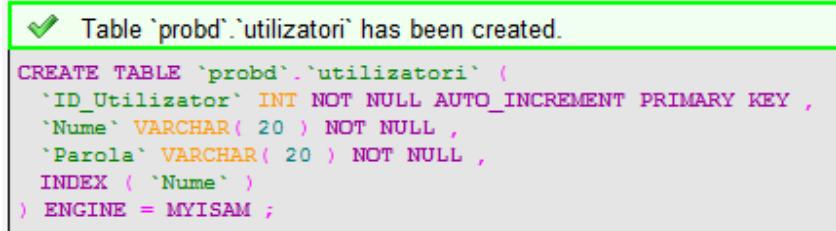
Câmpul *Nume* s-a definit ca fiind *INDEX*. Impunerea ca index a unui câmp facilitează căutările în tabel efectuate după valori ale respectivului câmp. În cazul dat, serverul MySQL va ataşa tabelului *Utilizatori* un fișier ale cărui înregistrări vor conține perechi de valori *Nume* - *adresă articol pe disc*. În noul fișier înregistrările vor fi păstrate ordonate crescător după valorile câmpului *Nume*. La căutarea după *Nume* a unui articol din tabelul *Utilizatori* se va căuta mai întâi în fișierul asociat tabelului adresă articolului căutat după care se va extrage direct articolul dorit.

Notă: Spre deosebire de *PRIMARY*, *INDEX* nu înseamnă valori distincte. Câmpul *Nume* permite introducerea același nume de mai multe ori. În acest caz, în fișierul asociat tabelului *Utilizatori* vor fi înregistrate grupat mai multe perechi *Nume* - *adresă* conținând același nume. La căutarea unui nume care apare repetat, serverul MySQL va furniza tot grupul de articole corespunzând criteriului de căutare impus.

Crearea tabelului se realizează prin apăsarea butonului

Save

phpMyAdmin va afișa și comanda *CREATE TABLE* trimisă serverului MySQL.



```
✓ Table `probd`.`utilizatori` has been created.

CREATE TABLE `probd`.`utilizatori` (
  `ID_Utilizator` INT NOT NULL AUTO_INCREMENT PRIMARY KEY ,
  `Nume` VARCHAR( 20 ) NOT NULL ,
  `Parola` VARCHAR( 20 ) NOT NULL ,
  INDEX ( `Nume` )
) ENGINE = MYISAM ;
```

- **Crearea tabelului categorii**

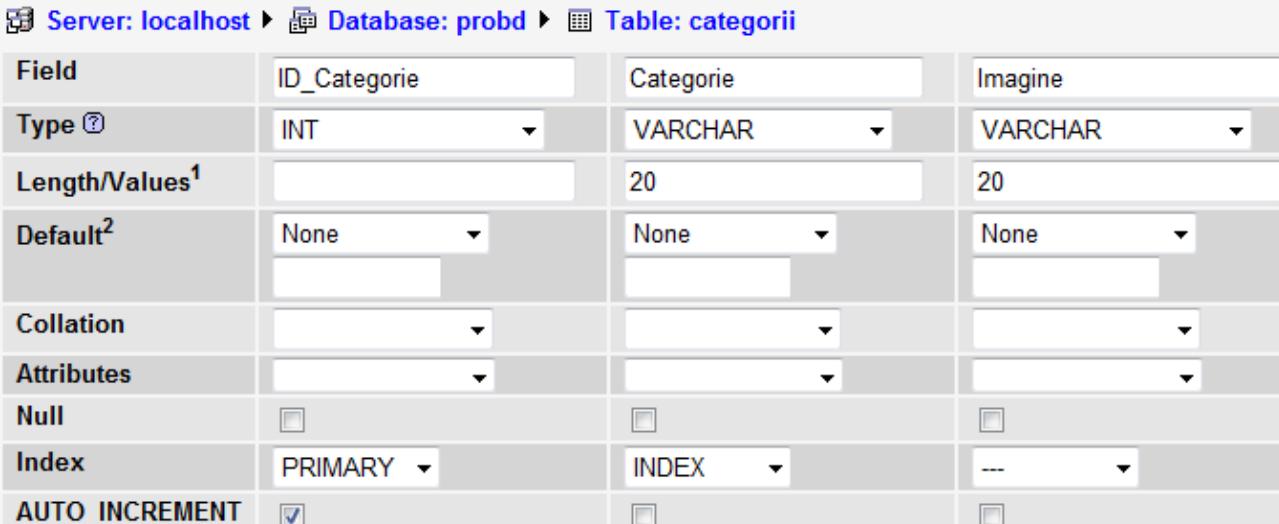
Baza de date *prodb* va conține printre altele un tabel cu poze. Pentru afișarea acestora se va declara în prealabil tabelul *categorii*. Rolul acestuia va fi acela de a permite ulterior selectarea și afișarea grupată a pozelor aparținând aceleiași categorii.

Tabelul *categorii* are trei câmpuri, *ID_Categorie* fiind declarat cheie primară.



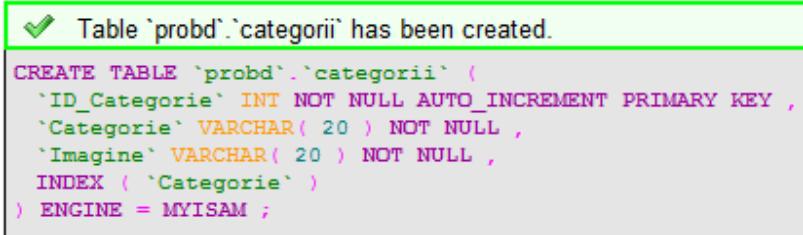
Create new table on database **probd**

Name: **categorii** Number of fields: **3**



Server: localhost > Database: probd > Table: categorii

Field	ID_Categorie	Categorie	Imagine
Type	INT	VARCHAR	VARCHAR
Length/Values ¹		20	20
Default ²	None	None	None
Collation			
Attributes			
Null	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Index	PRIMARY	INDEX	---
AUTO_INCREMENT	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>



```
✓ Table `probd`.`categorii` has been created.

CREATE TABLE `probd`.`categorii` (
  `ID_Categorie` INT NOT NULL AUTO_INCREMENT PRIMARY KEY ,
  `Categorie` VARCHAR( 20 ) NOT NULL ,
  `Imagine` VARCHAR( 20 ) NOT NULL ,
  INDEX ( `Categorie` )
) ENGINE = MYISAM ;
```

Notă: Câmpul *Imagine* va conține denumirea unui fișier reprezentativ pentru fiecare categorie. Acesta va permite scrierea unui script PHP care va genera pagina care prezintă categoriile de poze din baza de date. Dacă acest script va fi scris, adăugarea unei noi categorii nu va necesita modificarea unui cod *html* ci doar adăugarea unui nou articol în tabelul *Categorii*.

- **Crearea tabelului *imagini***

Server: localhost ▶ Database: probd ▶ Table: imagini			
Field	Type ②	Length/Values ¹	Index
ID_Poza	INT		PRIMARY ▾
Fisier	VARCHAR	50	---
ID_Categorie	INT		INDEX ▾
Comentariu	VARCHAR	100	---

✓ Table 'probd'.`imagini` has been created.

```
CREATE TABLE `probd`.`imagini` (
  `ID_Poza` INT NOT NULL ,
  `Fisier` VARCHAR( 50 ) NOT NULL ,
  `ID_Categorie` INT NOT NULL ,
  `Comentariu` VARCHAR( 100 ) NOT NULL ,
  PRIMARY KEY ( `ID_Poza` ),
  INDEX ( `ID_Categorie` )
) ENGINE = MYISAM ;
```

c. **Inserarea datelor în tabele** - comanda *INSERT*

Odată definite tablele, baza de date poate fi exploarată. Spre deosebire de operațiile realizate deja, respectiv crearea bazei de date și a tabelelor acesteia, inserarea, corectarea și regăsirea datelor se realizează de regulă de către scripturile aplicației Web. Totuși setul de date de testare necesar în etapa de construire a aplicației Web se realizează tot cu phpMyAdmin.

Comanda *insert* are formatul:

INSERT INTO tabel (camp1, camp2 ...) VALUES (v1, v2, ...)

Dacă toate câmpurile unui tabel primesc valori și valorile sunt dispuse în ordinea în care au apărut la declararea tabelului, comanda *INSERT* se poate scrie prescurtat:

INSERT INTO tabel VALUES (v1, v2, ...)

Fiecare dintre valorile *v1, v2, ...* va fi încadrată între caractere ' (apostrof).

Formatul implicit al valorilor de tip dată calendaristică este *an-luna zi*: Exemplu:

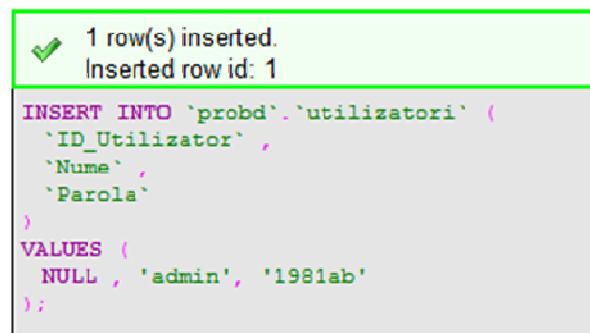
INSERT INTO tabtest (nrcrt, data_cal) VALUES ('1', '2009-04-22');

Exemplu: Inserarea unui articol în tabelul *Utilizatori* folosind phpMyAdmin :

Se selectează în panoul din stânga al aplicației phpMyAdmin baza de date și apoi tabelul (*utilizatori*). În panoul central se selectează tabul *Insert*:

Field	Type	Function	Null	Value
ID_Utilizator	int(11)			
Nume	varchar(20)			admin
Parola	varchar(20)			1981ab

Notă: Deși nu s-a specificat o valoare pentru *ID_Utilizator*, faptul că acest câmp a fost declarat la crearea tabelului ca fiind de tip *autoincrement* face ca aplicația să introducă automat o valoare. Următorul articol va avea *ID_Utilizator* = 2 și aşa mai departe.



d. **Selectarea datelor din tabele** - comanda *SELECT*

Comanda SELECT crează o mulțime de selecție care va conține datele din baza de date care satisfac eventualele condiții impuse.

Sintaxa comenzi este:

SELECT coloane FROM tabel

Exemplu:

SELECT nume, parola FROM utilizatori;

			nume	parola
<input type="checkbox"/>			admin	1981ab
<input type="checkbox"/>			Pop Vasile	123^\$q
<input type="checkbox"/>			INFOAP	1209re

*SELECT * FROM utilizatori;*

	ID_Utilizator	Nume	Parola
<input type="checkbox"/>	1	admin	1981ab
<input type="checkbox"/>	2	Pop Vasile	123^\$q
<input type="checkbox"/>	3	INFOAP	1209re

Caracterul '*' folosit în locul numelor de coloane determină includerea în mulțimea de selecție a tuturor coloanelor tabelului.

Clauza *WHERE* adăugată frazei *SELECT* permite filtrarea datelor incluse în mulțimea de selecție.

Exemplu:

SELECT nume, parola FROM utilizatori WHERE id_utilizator > 1;

	nume	parola
<input type="checkbox"/>	Pop Vasile	123^\$q
<input type="checkbox"/>	INFOAP	1209re

La scrierea condițiilor sirurile de caractere constante vor fi încadrate între caractere ' (apostrof) iar valorile numerice vor fi scrisa fără apostroafe.

La scrierea condițiilor introduse prin clauza *WHERE* pot fi folosiți următorii operatori:

=	egal
<>	diferit
>	mai mare
<	mai mic
>=	mai mare sau egal
<=	mai mic sau egal
BETWEEN	cuprins între două valori
LIKE	căutare după un şablon
IN	inclus într-o mulțime precizată

Pentru scrierea condițiilor complexe se pot folosi operatorii AND (și), OR (sau) și NOT (negație).

Exemple:

```
SELECT * FROM utilizatori WHERE id_utilizator BETWEEN 1 AND 3;
```

```
SELECT * FROM utilizatori WHERE nume LIKE 'Po%';
```

	ID_Utilizator	Nume	Parola
	2	Pop Vasile	123^\$q

Caracterul generic '%' ține locul oricărui grup de caractere. Astfel scrisă clauza WHERE permite selectarea tuturor articolelor în care câmpul *nume* începe cu 'Po'.

Comanda *SELECT* poate fi scrisă astfel încât mulțimea de selecție să conțină câmpuri provenind din mai multe tabele. În acest caz clauza *WHERE* va conține și relațiile dintre tabele folosite.

Exemplu:

```
SELECT categorii.denumire, imagini.fisier, imagini.comentariu FROM categorii, imagini WHERE categorii.id_categorie = imagini.id_categorie AND categorii.id_categorie=3;
```

Clauza *ORDER BY* adăugată frazei *SELECT* permite crearea unei mulțimi de selecție ordonată după valorile unuia sau mai multe câmpuri.

Exemple:

```
SELECT * FROM utilizatori ORDER BY nume;
```

```
SELECT * FROM imagini WHERE imagini.id_categorie < 3 ORDER BY fisier DESC;
```

În al doilea exemplu clauza *DESC* indică faptul că se dorește o ordonare descrescătoare. Pentru o ordonare crescătoare se poate scrie *ASC*. În lipsa precizării modului de ordonare, aceasta se va realiza crescător.

e. **Modificarea datelor din tabele** - comanda *UPDATE*

Comanda UPDATE permite atât modificarea structurii tabelelor bazei de date cât și modificarea valorilor conținute de acestea. Deoarece modificarea structurii tabelelor se realizează mai elegant folosind phpMyAdmin, în cele ce urmează comanda va fi folosită numai pentru editarea datelor din tabele.

Sintaxa comenzi este următoarea:

UPDATE tabel SET câmp=valoare WHERE condiție;

Exemplu:

UPDATE utilizatori SET parola='12345\$' WHERE id_utilizator=2;

f. **Ștergerea datelor din tabele** - comanda *DELETE*

Comanda DELETE permite ștergerea înregistrărilor din tabele.

Sintaxa comenzi este:

DELETE FROM tabel WHERE condiție;

Exemple:

DELETE FROM utilizatori WHERE id_utilizator=3;

Pentru a șterge toate datele dintr-un tabel comanda DELETE se scrie fără clauza WHERE.

DELETE FROM utilizatori;

3.6 PHP

PHP este un limbaj de programare destinat scrierii componentelor unei aplicații Web. Limbajul PHP reprezintă o bună soluție pentru scrierea scripturilor care prelucrează datele din formulare sau pentru generarea automată a unor pagini Web folosind date păstrate pe server în baze de date.

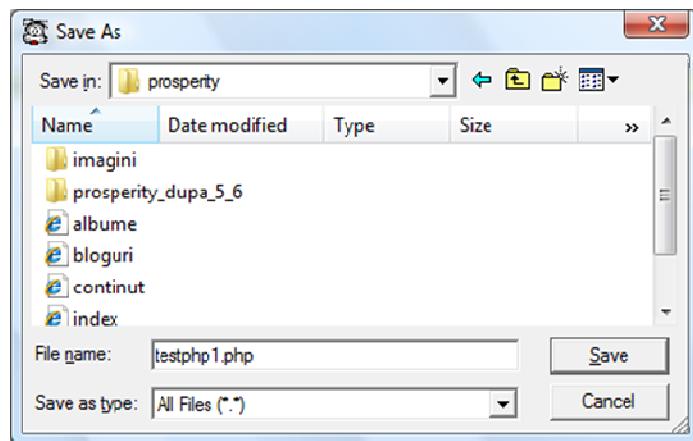
Limbajul PHP și interpretorul de comenzi aferent sunt gratuite. Interpretorul de comenzi este conținut în pachetul de aplicații XAMPP și este instalat automat împreună cu celelalte componente ale pachetului.

Un fișier conținând cod scris în PHP poate conține pe lângă secvențele de cod și marcaje HTML cunoscute. Lansarea în execuție a interpretorului PHP se realizează de către serverul de Web (Apache, IIS etc.) dacă fișierul cerut de browser are extensia *.php*. Interpretorul va trimite browserului atât codul HTML găsit în fișier cât și rezultatele executării secvențelor programate în PHP.

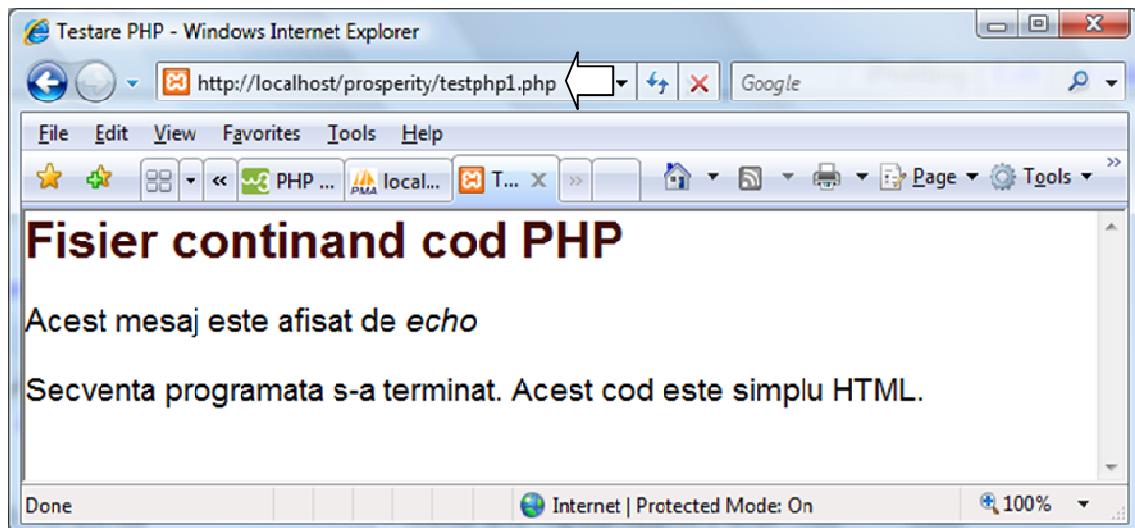
Exemplu:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>Testare PHP</title>
<link rel="stylesheet" type="text/css" href="stil.css" />
</head>
<body>
<h2>Fisier continand cod PHP</h2>
<?php
    echo "<p>Acet mesaj este afisat de <em>echo</em></p>";
?>
<p>Secventa programata s-a terminat. Acet cod este simplu HTML.</p>
</body>
</html>
```

Secvența scrisă se salvează într-un director accesibil serverului de Web. Dacă se testează folosind XAMPP fișierul trebuie să fie plasat în directorul *htdocs* sau într-un director derivat din acesta.



Pentru testare se va porni *Internet Explorer* și se va tasta adresa fișierului în caseta de text destinată adreselor.



În fișier s-a evidențiat secvența scrisă în PHP. Instrucțiunea `echo` realizează trimiterea spre browser a sirului de caractere dat ca argument. Sirul de caractere este încadrat între ghilimele.

Exemplul 2.

În acest exemplu se realizează o pagină web conținând un formular (`greutate.html`) și se scrie scriptul care prelucrează datele din formular (`testphp2.php`) indicat de argumentul `action` din marcapajul `<form>`.

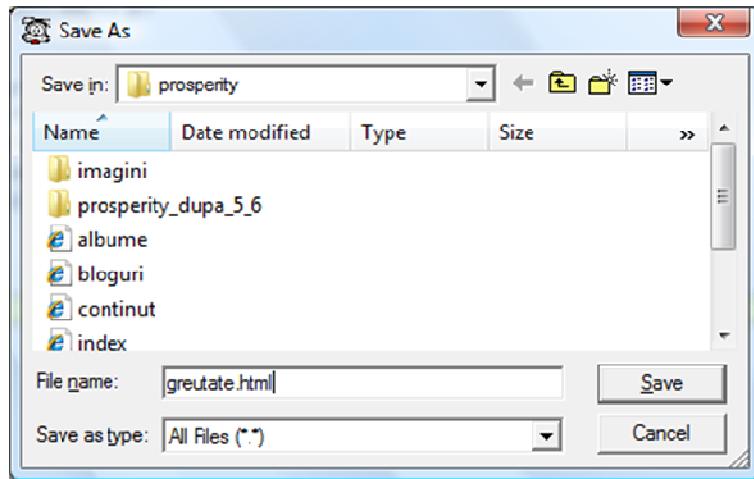
greutate.html:

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>Calculul greutății</title>
<link rel="stylesheet" type="text/css" href="stil.css" />
</head>
<body>
<h2>Calculul greutății in PHP</h2>
<p>Completați formularul și apăsați butonul <em>Calcul</em></p>
<form action="testphp2.php" method="post">
<table>
<tr><td>Inaltimea [cm]: </td><td> <input type="text" name="inaltime" /></td></tr>
<tr><td>Varsta [ani]: </td><td> <input type="text" name="varsta" /></td></tr>
<tr><td colspan="2"><label>Femeie<input type="checkbox" name="sex" value="f" /></label></td></tr>
<tr><td colspan="2"><input type="submit" value="Calculeaza" /></td></tr>
</table>
</form>

```

```
</body>
</html>
```



testphp2.php:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>Calculul greutatii</title>
<link rel="stylesheet" type="text/css" href="stil.css" />
</head>
<body>
<h2>Calculul greutatii in PHP</h2>
<p>Greutatea Dv. ar trebui sa fie
<?php
    $v = $_REQUEST["varsta"];
    $h = $_REQUEST["inaltime"];
    $s = $_REQUEST["sex"];
    $g = 50 + ($h-150)*0.75+($v-20)*0.25;
    if($s == "f")
        $g = $g * 0.9;
    echo "aproximativ $g";
?>
kg.</p>
</body>
</html>
```

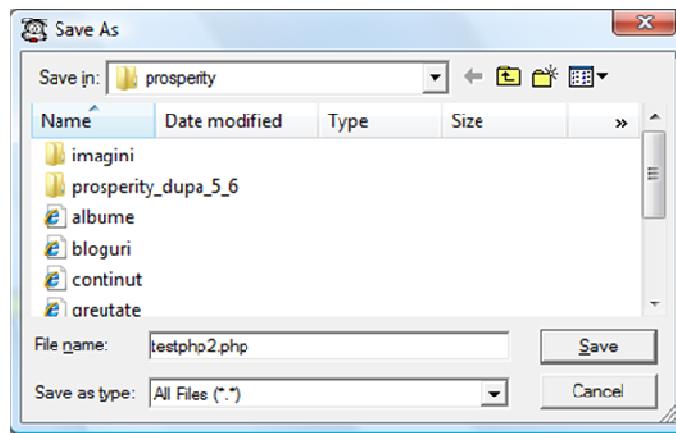
Spre deosebire de primul script PHP, în acest exemplu se operează cu câteva variabile: \$v, \$h, \$s, \$g și \$_REQUEST. În PHP numele variabilelor încep cu caracterul '\$'.

`$_REQUEST` este un sir care contine valorile din campurile formularului asociat. Pentru a extrage o valoare se foloseste expresia `$_REQUEST["camp"]`, in care *camp* este valoarea proprietatii *name* a controlului Windows a carui valoare dorim sa o preluam.

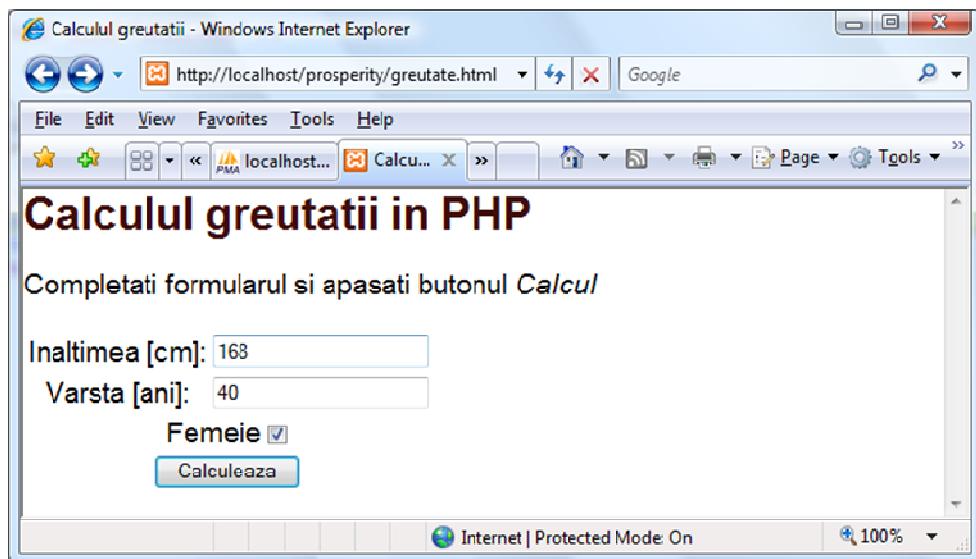
Exemplu:

```
$h = $_REQUEST["inaltime"];
```

Scriptul PHP se salveaza in acelasi director in care s-a salvat *greutate.html*.

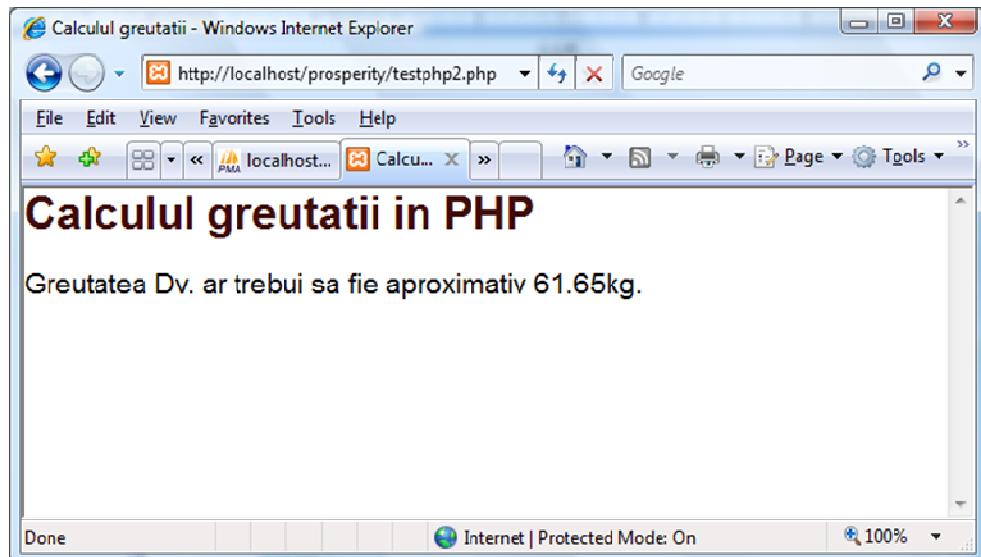


Pentru testarea aplicatiei se introduce adresa paginii *greutate.html* si se apasă butonul *Calculeaza*.



Apasarea butonului determină trimitera spre serverul de Web a cererii de executare a scriptului *testphp2.php* împreună cu un sir de caractere conținând valorile

din câmpurile formularului. Serverul de Web va lansa în execuție scriptul solicitat și în fereastra browser-ului vor fi afișate rezultatele.



Notă: În cazul formularelor pentru care *method="post"*, la apăsarea butonului de tip *submit* aplicația de navigare trimite spre Internet cererea de executare a scriptului PHP după care trimite scriptului un sir de caractere având structura:

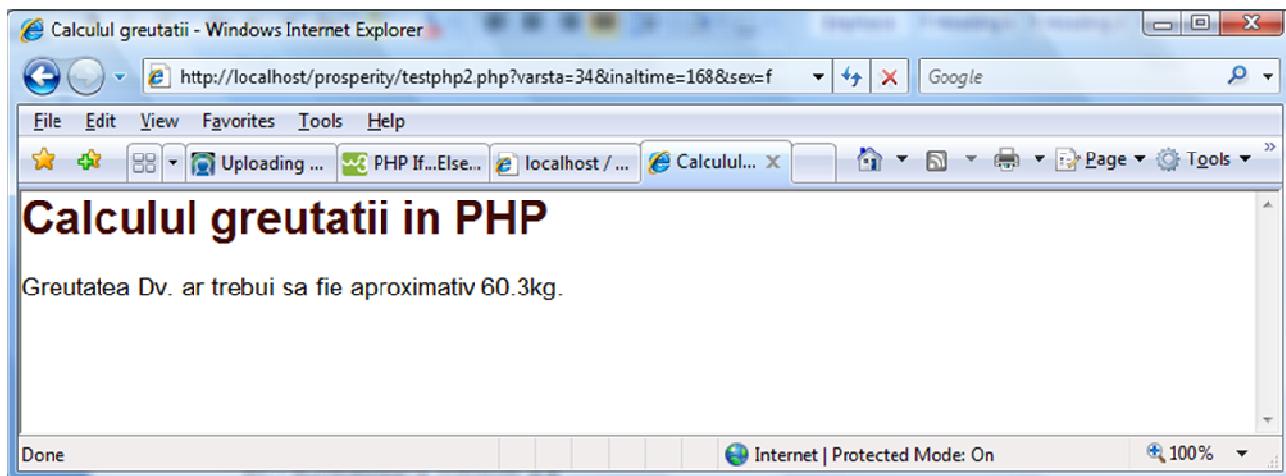
numecontrol1=valoare1&numecontrol2=valoare2& numecontrol3=valoare3 ...

În cazul formularelor pentru care *method="get"*, la apăsarea butonului de tip *submit* aplicația de navigare trimite spre Internet o cerere având structura:

nume_script? numecontrol1=valoare1&numecontrol2=valoare2& ...

Scripturile PHP care extrag datele primite folosind sirul `$_REQUEST` se scriu la fel indiferent de modul de trimis de către browser a parametrilor asupra cărora trebuie să opereze. Astfel scriptul *testphp2.php* poate fi lansat în execuție fără a folosi formularul din *greutate.html* scriind în caseta de adrese a browser-ului :

`http://localhost/prosperity/testphp2.php?varsta=34&inaltime=168&sex=f`



Acest mod de scriere poate servi la lansarea în execuție a unui ansamblu de scripturi folosind în locul formularelor simple legături:

E exemplu :

```
<ul>
<li><a href="scriptphp1.php?opt=1">Calcul pentru opțiunea 1</li>
<li><a href="scriptphp1.php?opt=2">Calcul pentru opțiunea 2</li>
<li><a href="scriptphp1.php?opt=3">Calcul pentru opțiunea 3</li>
</ul>
```

3.6.1 Elementele de bază ale limbajului PHP

Variabile

În PHP numele unei variabile începe cu caracterul "\$". Variabilele nu trebuie declarate în prealabil. Crearea unei variabile și stabilirea tipului acesteia se realizează automat, în momentul în care acesteia i se atribuie o valoare. Unele variabile sunt create automat de interpretorul PHP, ca în cazul scripturilor care prelucrează datele conținute în formulare (variabila \$_REQUEST).

În PHP variabilele pot fi de tip *integer*, *double*, *string*, *boolean* (logic) sau *array* (șir). Exemple :

```
$a = "acesta este un sir de caractere";           // $a este un sir de caractere
$b = 3;                                         // $b este un întreg
$c = 4.12;                                       // $c este un nr. real, dublă precizie
$d = "2";                                         // $d este un sir de caractere
$e = $b + $d;                                     // $e = 5, deci $d a fost tratat ca întreg.
$f = TRUE;                                        // $f este o variabilă logică (boolean). Se poate scrie cu majuscule sau normal.
```

```
$g = false; // idem.
$luni = array("ian", "feb", "mar", "apr"); // $luni este un sir de valori
```

Şiruri de caractere

Şirurile de caractere sunt delimitate prin caractere " (ghilimele). Dacă în sir există ghilimele, caractere \$ sau caractere \ (backslash), pentru a fi interpretate ca atare acestea vor fi precedate de \.

Exemplu:

```
$a = "New York";
$mesaj = "Excursie la $a";
```

Variabilele prezente în siruri delimitate prin ghilimele vor fi înlocuite prin valoarea lor, deci *\$mesaj* = "Excursie la New York".

Caracterul "." (punct) este în PHP operatorul de concatenare.

Exemplu:

```
<?php
$a=1;
$b=3.14159;
$c=TRUE;
$d=array("ian", "feb", "mar");
echo "Tipul variabilei '$a' : " . gettype($a). "<br>\n";
echo "Tipul variabilei '$b' : " . gettype($b). "<br>\n";
echo "Tipul variabilei '$c' : " . gettype($c). "<br>\n";
echo "Tipul variabilei '$d' : " . gettype($d). "<br>\n";
?>
```

Un sir poate fi delimitat și prin caractere ' (apostrof) dar în acest caz variabilele incluse în sir nu vor mai fi înlocuite prin valoarea lor.

Şiruri de valori

Un sir poate fi declarat folosind funcția *array()*:

```
$luni = array("ianuarie", "februarie", "martie", "aprilie");
$luni[4] = "mai";
$luni[5] = "iunie";
```

```
$luni[] = "iulie";
echo "Luna a doua este $luni[1].";
```

Deoarece primul element dintr-un sir are indicele 0, efectul comenzii `echo` va fi imprimarea sirului "*Luna a doua este februarie.*" Atribuirea `$luni[] = "iulie";` este corecta, elementul care va primi valoare fiind luni[6].

Şiruri asociative

Elementele dintr-un sir asociativ sunt perechi de forma (*cheie => valoare*).

Un script scris in PHP asociat unui formular primeşte într-un astfel de sir, denumit `$_REQUEST`, valorile din câmpurile formularului. Scripturile PHP încep de obicei cu o secvenţă de cod care preia valorile din sirul `$_REQUEST`.

Exemplu:

```
$eml = strip_tags($_REQUEST["e_mail"]);
```

Notă: Funcția `strip_tags()` are rolul de a înlătura din sirul dat ca argument eventualele marcaje HTML sau scripturi PHP. Această măsură de precauție blochează tentativele celor care urmăresc preluarea de către script și apoi afișarea de informații nedorite.

Exemplu de formular care permite încărcarea unui fișier pe server:

Fisierul formpoze.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>Upload poze</title>
</head>
<body>
<h1>Incarc poze pe server</h1>
<form enctype="multipart/form-data" action="upld.php" method="post">
<table>
<tr>
<td>Categoria: </td><td><select name="categoria">
<option value="niciuna" selected>(Selectati categoria)
<option value="1">Austria
<option value="2">Franta
<option value="3">Elvetia
<option value="4">Ungaria
<option value="5">Grecia
</select>
```

```

</td>
</tr>
<tr>Selectati fisierul : </td><td><input type="file" name="fisier" /></td></tr>
<tr><td colspan="2"><input type="submit" value="Incarcare server"></td></tr>
</table>
</form>
</body>
</html>

```

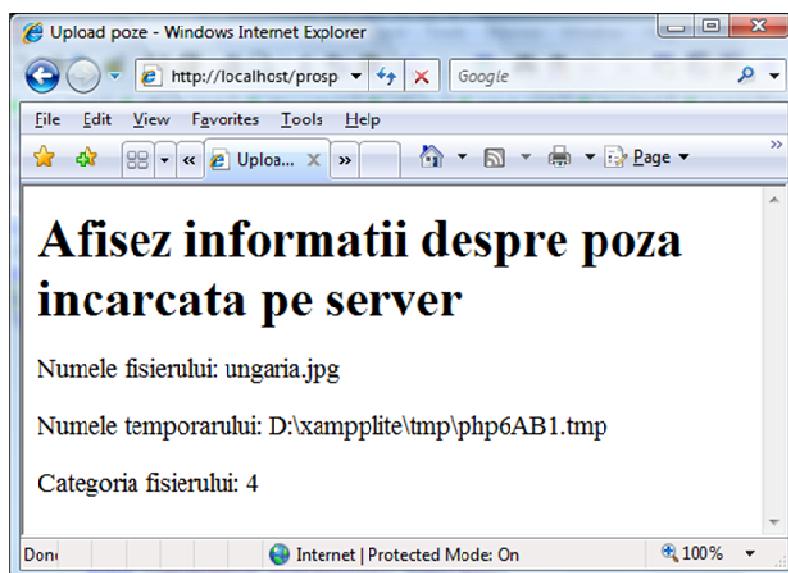
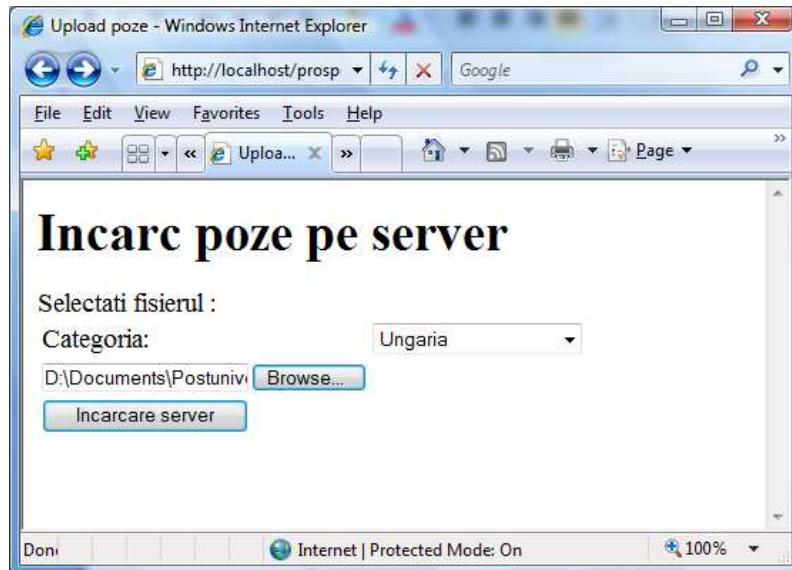
Fisierul upld.php

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>Upload poze</title>
</head>
<body>
<h1>Afisez informatii despre poza incarcata pe server</h1>
<?php
    if ($_FILES["fisier"]["error"] > 0)
    {
        echo "Error: " . $_FILES["fisier"]["error"] . "<br />";
        exit;
    }
    else
    {
        $categ = $_REQUEST["categoria"];
        $nm = $_FILES["fisier"]["name"];
        $nmtmp = $_FILES["fisier"]["tmp_name"];
        echo "<p>Numele fisierului: $nm</p>";
        echo "<p>Numele temporarului: $nmtmp</p>";
        echo "<p>Categoria fisierului: $categ</p>";
        $cale = "poze/$nm";
        $rezultat = move_uploaded_file($nmtmp, $cale);
        if (!$rezultat) {
            echo "Eroare la incarcarea fisierului";
            exit;
        }
    }
?></body>
</html>

```

Exemplu de utilizare a aplicației:



Notă: Încărcarea unui fișier pe server se realizează în directorul pentru fișiere temporare **XAMPPLITE/tmp**. Scriptul PHP asociat formularului de încărcare trebuie să mute fișierul din acest director într-un director al aplicației folosind funcția PHP **move_uploaded_file()**.

Accesul la caracteristicile fișierului se realizează prin intermediul matricii **\$_FILES**.

<code>\$_FILES["nume_control"]["name"]</code>	numele original (pe calc. client)
<code>\$_FILES["nume_control"]["tmp_name"]</code>	numele temporar (pe server)
<code>\$_FILES["nume_control"]["type"]</code>	tipul fișierului (
<code>\$_FILES["nume_control"]["size"]</code>	mărimea în octeți
<code>\$_FILES["nume_control"]["error"]</code>	<i>true</i> dacă încărcarea nu a reușit

Dat fiind faptul că numărul de fișiere care pot fi încărcate pe server poate fi mare, scriptul ar trebui să se continue cu memorarea datelor fișierului într-o bază de date.

Funcții PHP pentru testarea variabilelor

Rezultatul apelului unei astfel de funcții poate fi "FALSE" (fals) sau "TRUE" (adevărat).

- Funcția `isset($var)` verifică dacă variabili \$var indicată ca argument își are atribuită o valoare.
- Funcția `empty($var)` are valoarea "TRUE" (adevărat) dacă variabila indicată ca argument este neinitializată, este un sir de caractere de lungime 0 sau are valoarea întreagă 0.

Exemplu:

```
if (empty($nume_client)) {
    echo "Va rog sa completati numele dv.!";
    exit;
}
```

- Funcțiile `is_int($var)`, `is_double($var)`, `is_string($var)`, `is_array($var)`, și `is_bool($var)` testează tipul variabilei indicate ca argument.

Modificarea tipului variabilelor

Modificarea tipului unei variabile se poate face în 3 feluri:

1. Folosind operatori de transtipaj, ca în C:

```
$a = 1;
$b = (string) $a;
```

Operatorii posibili sunt (*integer*), (*double*), (*array*) și (*string*).

2. Folosind funcția `settype()`:

```
$a = 1;
settype($a, "string");
```

3. Folosind funcțiile *intval()*, *doubleval()* sau *stringval()*. Aceste funcții nu modifică tipul variabilei introduse ca argument dar returnează o valoare aparținând tipului cerut:

```
$a = "12";
$b = intval($a) * 2;
```

3.6.2 Instrucțiunile limbajului PHP

- **Instrucțiunea if**

Ca și în alte limbaje, instrucțiunea *if* are două forme:

```
if (condiție)
{
    acțiuni pentru condiție = adevărat (TRUE)
}
```

respectiv

```
if (condiție)
{
    acțiuni pentru condiție = adevărat (TRUE)
}
else
{
    acțiuni pentru condiție = fals (FALSE)
}
```

Condiția poate fi o expresie logică, scrisă folosind operatorii relaționali uzuali: *<*, *<=*, *>*, *>=*, *==*, *!=* (sau *<>*, diferit), respectiv *&&* (dar și *and*, operatorul *și*), *||* (dar și *or*, operatorul *sau*), *!* (negație). Cele două variante de operatori logici *"și"* respectiv *"sau"* diferă prin ordinea de evaluare a operanzilor.

Ca și în alte limbaje, *condiție* poate fi o expresie aritmetică, o valoare 0 a acesteia fiind interpretată ca fals (*FALSE*) iar o valoare diferită de 0 ca adevărat (*TRUE*).

- **Instructiunea if ... elseif**

Ca și în C, și în PHP pot fi scrise construcții de forma:

```
if (conditie_1)
{
    Acțiuni pentru condiție_1 = adevărat
} elseif (conditie_2)
{
    Acțiuni pentru condiție_2 = adevărat
} elseif (conditie_3)
{
    Acțiuni pentru condiție_3 = adevărat
} else {
    Acțiuni pentru restul situațiilor
}
```

Ultimul **else** poate lipsi.

- **Instructiunea switch ... case**

Pentru cazul în care într-un program ramificarea depinde de valoarea unei variabile, structura switch ... case este mai potrivită. Ea are sintaxa:

```
switch ($var)
{
    case v1:
        acțiuni;
        break;

    case v2:
        acțiuni;
        break;
    case v3:
        acțiuni;
        break;
}
```

Exemplu:

```
switch ($submit)
{
    case "insert":
        // instructiuni pt. inserare in baza de date
        break;
    case "update":
```

```

    // instructiuni pt. actualizarea bazei de date
case "display":
    // instructiuni pt. imprimare
    break;
}

```

Porțiunea din script din exemplu realizează o acțiune comandată prin apăsarea unui buton de tip *submit*. În funcție de butonul apăsat, valoarea transmisă scriptului diferă. În cazul *"update"* instrucțiunea *"break"* lipsește, deci după actualizare se va executa codul specific imprimării (*\$submit = "display"*).

- **Instrucțiunea while**

Instrucțiunea *while* indică repetarea unei acțiuni atâtă vreme cât o condiție este adevărată. Sintaxa instrucțiunii este:

```

while (condiție)
{
    acțiune
}

```

Exemplu:

```

$comanda = "select nume, prenume from utilizatori";
$rezultat = mysql_query($comanda) or die(mysql_error());
while ($row = mysql_fetch_array($rezultat))
{
    echo $row["nume"] . " " . $row["prenume"] . "<br>\n";
}

```

Această structură de cod este specifică activității de recuperare de informații dintr-o bază de date.

- **Instrucțiunea do ... while**

Instrucțiunea *do ... while* indică repetarea unei acțiuni atâtă vreme cât o condiție este adevărată. Spre deosebire de *while*, condiția este testată după executarea cel puțin o dată a acțiunii. Sintaxa instrucțiunii este:

```

do
{
    acțiune
}
while (condiție);

```

- **Instructiunea for**

Instructiunea for are sintaxa următoare:

```
for(exp1; exp2; exp3)
{
    acțiune
}
```

Exemplu:

```
<html>
<?php
    $factorial = 1;
    for ($i=1; $i<7; $i++)
    {
        echo $i . " factorial = " . $factorial . "<br>\n";
        $factorial *= $i; // atribuire combinata, ca în C.
    }
?>
</html>
```

- **Instructiunea foreach**

Instructiunea *foreach* se folosește exclusiv pentru prelucrarea elementelor sirurilor. Sintaxa instructiunii este:

```
foreach ($nume_sir as $variabila)
{
    acțiune pentru valoarea curentă
}
```

Exemplu:

```
$nume = array("Ion", "Maria", "George");
foreach ($nume as $membru)
{
    echo "$membru este invitat la cina.<br>\n";
}
```

- **Instrucțiunile *break* și *continue***

Ca și în C, *break* îintrerupe un ciclu sau un *switch*. Înstrucțiunea care urmează după *break* este cea care urmează după instrucțiunea *switch* sau după ciclul care conține *break*.

Instrucțiunea *continue* este folosită tot în interiorul ciclurilor și comandă reluarea imediată a ciclului îintrerupând secvența de cod cuprinsă între locul în care *continue* este inserat și sfârșitul ciclului.

Exemplu:

```
<html>
<?php
$director = opendir ('c:/apache/htdocs/postuniv/');
echo "fisierele din directorul $director sunt:<br>\n";
while ($fisier = readdir ($director))
{
    if (is_dir ($fisier)) { continue; }
    echo "$fisier <br> \n";
}
closedir ($director);
?>
</html>
```

Funcțiile *opendir()* respectiv *closedir()* deschid respectiv închid un director iar funcția *readdir()* citește o intrare din acest director. Tipul intrării (fișier sau director) este testat prin apelul funcției *is_dir()* și în caz pozitiv, intrarea este ignorată (se execută instrucțiunea *continue*).

- **Instrucțiunea *exit***

Instrucțiunea *exit* oprește imediat scriptul PHP. Același efect îl are funcția *die()*.

Exemplu:

```
die("Ati completat gresit formularul!");
```

3.6.3 Funcții

Într-o aplicație scrisă în PHP pot fi întâlnite două tipuri de funcții: funcții predefinite, aparținând limbajului și funcții definite de programator pentru a evita scrierea repetată a unor secvențe de cod sau pentru a ușura înțelegerea codului.

Definirea unei funcții

Sintaxa unei funcții scrise în PHP este următoarea:

```
function nume([listă_parametri])
{
    corpul_funcției
    [return expresie;]
}
```

Lista parametrilor transmiși unei funcții poate lipsi. Dacă funcția nu returnează nimic (lipsește instrucțiunea *return*) funcția nu poate figura într-o expresie.

Exemplu:

```
function calcul ($a=0, $b=0)
{
    $c = $a + $b;
    return $c;
}
```

Includerea în lista de parametri a atribuirilor `$a=0, $b=0` asigură valori inițiale variabilelor `$a` și `$b`, în cazul în care apelul s-a realizat incorect (cu valori neinitializate de exemplu). Apelul funcției se poate face astfel:

```
$rez = calcul(12, 223);
```

Valoarea returnată de o funcție poate apartine oricărui tip: *string, array, integer, double* etc.

Vizibilitatea variabilelor

Utilizarea funcțiilor pune și în PHP probleme de vizibilitate a variabilelor. În PHP o variabilă declarată în afara oricărei funcții este o *variabilă globală* iar o variabilă

declarată într-o funcție este o *variabilă locală*. Spre deosebire de alte limbaje, pentru a fi vizibile într-o funcție variabilele globale trebuie declarate folosind declarația **global**.

Exemplu:

```
function suma ($b)
{
    global $a;
    $c = $a + $b;
    return $c;
}

$a = 100;
echo suma(12);
```

Funcții predefinite

PHP este deosebit de bogat în funcții. O listă completă poate fi găsită la adresa <http://www.php.net/manual>.

Funcții pentru prelucrarea sirurilor de caractere

În domeniul prelucrării sirurilor de caractere, PHP oferă o mare varietate de funcții predefinite, peste 70 la număr, ceea ce face posibilă realizarea în acest limbaj a celor mai multe dintre prelucrările posibile în alte limbaje. Fiind orientat pe tratarea informațiilor conținute în pagini web, PHP pune la dispoziție și o serie de funcții specifice, deosebit de utile în practică.

- *explode()* – realizează fragmentarea unui sir de caractere folosind separatorii coținuți într-un alt sir. Rezultatul va fi un sir de elemente. Prototipul funcției este:

```
string[] explode(string separatori, string sir_de_prelucrat, limita)
```

Exemplu de utilizare:

```
$elemente = explode(',', $cumparaturi); // separatorul este virgula
foreach ($elemente as $articol)
{
    ... // prelucrez elementul curent, $articol
}
```

Parametrul *limita* este optional. El are rolul de a permite limitarea numărului de elemente din sirul creat.

- *strip_tags()* – realizează înlăturarea dintr-un sir de caractere a tuturor marcajelor HTML sau PHP, cu excepția unora specificate în mod explicit. Prototipul funcției este:

```
string strip_tags (string sir [, string marcaje_permise]).
```

Funcția permite înlăturarea pericolului includerii de către un utilizator într-o casetă de text sau o zonă de text, a unor marcaje HTML sau scripturi PHP nedorite.

Exemplu de utilizare:

```
$sir_bun = strip_tags($sir, "<em><strong>");
```

- *addslashes()* – modifică un sir de caractere dat adăugând caractere "\\" (backslash) în fața unor caractere ca: " (ghilimele), ' (apostrof), \ (backslash). Prototipul funcției este:

```
string addslashes (string sir).
```

Funcția este gândită să ajute la formarea sirurilor de caractere care sunt inserate în câmpuri aparținând unor tabele dintr-o bază de date, ca în exemplul următor:

```
$str1 = "let's see";
$str1 = addslashes($str1);
$rez = mysql_query("insert into continut (continut) values ('$str1')");
```

Sirurile de caractere provenind din câmpurile unui formular nu necesită o tratare folosind această funcție deoarece ele sunt în mod automat modificate și transmise în acest fel.

- *stripslashes()* – este funcția opusă funcției addslashes(). Ea suprimă caracterele '\\' adăugate ca urmare a apelării funcției addslashes(). Prototipul funcției este:

```
string stripslashes (string sir).
```

- *str_replace()* – realizează căutarea într-un sir de caractere a unui subșir și înlocuirea sa cu un alt sir. Prototipul funcției este:

```
string str_replace (string sir_cautat, string sir_substitutie, string sir_de_prelucrat).
```

Exemplu:

```
$sir = "Cei patru evanghelisti erau trei";
$sir1 = str_replace("trei", "doi", $sir);
```

- *substr_replace()* – permite înlocuirea unei secvențe de caractere dintr-un sir cu un alt sir, dat ca argument. Prototipul funcției este:

string **substr_replace** (string sir_de_prelucrat, string sir_substitutie, int start [, int lungime]).

Parametrii *start* și *lungime* definesc poziția în sirul de prelucrat respectiv lungimea subșirului care va fi înlocuit. Dacă ultimul parametru lipsește, vor fi înlocuite toate caracterele rămase până la sfârșitul sirului.

Exemplu:

```
$sir = "Vara aceasta mergem in Turcia.";
$sir1 = substr_replace($sir, "Italia.", 23);
```

- *strcmp()* – este funcția de comparare a două siruri. Prototipul funcției este:

int **strcmp** (string sir1, string sir2).

Funcția returnează o valoare pozitivă dacă primul sir este mai mare, 0 dacă sirurile sunt identice și o valoare negativă dacă primul sir este mai mic.

- *strlen()* – returnează lungimea sirului dat ca argument. Prototipul funcției este:

int **strlen** (string sir).

- *strpos()* – furnizează poziția sirului dat ca al doilea argument în sirul dat ca prim argument. Prototipul funcției este:

int **strpos** (string sir_dat, string sir_cautat [, int offset]).

Funcția returnează o valoare pozitivă dacă sirul este găsit și FALSE dacă sirul căutat nu există. Dacă parametrul *offset* este definit, acesta reprezintă poziția în sirul dat de unde începe căutarea.

Exemplu:

```
$sir = "Vara aceasta mergem la munte.";
$p = strpos($sir, " ");
```

Variabila \$p va avea valoarea 4, deoarece pe poziția 4 se află primul spațiu.

- *strrpos()* – operează asemănător cu *strpos()* dar funcția va returna poziția ultimei apariții în șirul dat a șirului dat ca al doilea argument. Prototipul funcției este:

```
int strrpos (string sir_dat, string sir_cautat).
```

- *substr()* – returnează o porțiune dintr-un șir indicată prin două valori numerice. Prototipul funcției este:

```
string substr (string sir_dat, int start [, int lungime]).
```

Dacă parametrul lungime lipsește, subșirul returnat va conține toate caracterele de la *start* până la sfârșitul șirului.

Exemplu:

```
<?
$nume = "carte.pdf";
$lung = strrpos($nume, ".");
$pdf = substr ($nume, 0, $lung);
echo $pdf; // Va afisa "carte"
?>
```

- *strrev()* – inversează ordinea caracterelor dintr-un șir dat. Prototipul funcției este:

```
string strrev (string sir_dat).
```

- *strtolower()* – transformă toate majusculele din șirul dat în litere mici. Prototipul funcției este:

```
string strtolower (string sir_dat).
```

- *strtoupper()* – transformă toate literele din șirul dat în majuscule. Prototipul funcției este:

```
string strtoupper (string sir_dat).
```

- *strtr()* – este o funcție care permite înlocuirea în sirul dat ca prim argument a caracterelor din sirul dat ca al doilea argument cu caracterele corespunzătoare din sirul dat ca al treilea argument. Prototipul funcției este:

```
string strtr (string sir_dat, string car_de_inlocuit, string car_de_substitutie).
```

Exemplu:

```
$sir = "sir de caractere de tratat";
$sir1 = strtr($sir, "ca", "qw");
```

În exemplul dat, toate caracterele "c" vor fi înlocuite cu "q" și toate caracterele "a" vor fi înlocuite cu "w".

- *trim()* – este o funcție care înlătură toate spațiile de la începutul și sfârșitul sirului dat ca argument, inclusiv caracterele return, LF (line feed) sau tab. Prototipul funcției este:

```
string trim (strin sir_dat).
```

Funcții pentru accesarea unei baze de date

PHP dispune de funcții care pot asigura exploatarea unui mare număr de servere de baze de date. În cele ce urmează vor fi prezentate doar funcțiile care servesc la exploatarea unei baze de date MySQL.

- *mysql_connect()* și *mysql_select_db()* – realizează o conexiune cu serverul MySQL. Prototipurile funcțiilor sunt:

```
int mysql_connect (string host, string user, string password).
int mysql_select_db(string database [, int conect])
```

De regulă apelul funcției *mysql_connect()* este urmat de apelul funcției *mysql_select_db()*. Dacă se recuperează valoarea întreagă returnată de *mysql_connect()*, ea poate fi folosită ca al doilea parametru în *mysql_select_db()*, permitând astfel ca într-un script să se realizeze conexiuni la mai multe servere de baze de date. Pentru conectarea la mai multe baze de date, valorile întregi și distințe returnate de funcțiile *mysql_select_db()* apelate vor fi folosite ca și parametri în funcții ca *mysql_query()* , *mysql_insert_id()*, *mysql_affected_rows()* etc., apelate ulterior.

Exemplu:

```
mysql_connect("localhost", "root", "") or die (mysql_error());
mysql_select_db("studenti") or die (mysql_error());
```

Dacă serverul de baze de date MySQL rulează pe un alt calculator din rețea, având adresa *postuniv.east.utcluj.ro*, conectarea la aceeași bază de date ar putea fi realizată scriind:

```
mysql_connect ("postuniv.east.utcluj.ro", "mdamian", "aq^123") or die (mysql_error());
mysql_select_db ("studenti") or die (mysql_error());
```

De cele mai multe ori scripturile PHP ale unei aplicații Web se conectează la o singură bază de date. Programatorii își scriu atunci un fișier PHP care realizează conexiunea și care va fi ulterior inclus în toate scripturile aplicației.

Exemplu:

```
<?php
$gazda = "localhost"; //locația serverului MySQL
$utilizator = "root"; //numele utilizatorului bazei
$parola = ""; //parola
$nume_baza = "prosperity"; //numele bazei de date
mysql_connect($gazda, $utilizator, $parola) or die (mysql_error());
mysql_select_db($nume_baza) or die (mysql_error());
?>
```

Dacă scriptul scris s-a salvat cu numele *conectare.php*, inserarea sa în scripturile aplicației se realizează printr-o comandă *include*:

```
<?php include "conectare.php"; ?>
```

- *mysql_query()* – este funcția cea mai frecvent utilizată când este exploatață o bază de date. Prototipul funcției este :

```
int mysql_query (string fraza_sql [, int link]);
```

Această funcție trimite o comandă serverului MySQL (de obicei *insert*, *select*, *delete* sau *update*). Funcția nu returnează rezultatul comenzi SQL ci o valoare care va fi folosită în continuare ca parametru în apelurile funcțiilor care realizează extragerea rezultatului comenzi. Dacă s-a introdus o comandă *select*, rezultatul este o mulțime de selecție, rândurile acesteia putând fi ulterior prelucrate folosind *mysql_fetch_array()*.

De regulă apelul funcției *mysql_query()* este urmat de construcția *or die(mysql_error())*, pentru a primi în caz de eșec un mesaj privind cauza acestuia.

- `mysql_insert_id()` – este o funcție care se poate apela după o comandă `insert`, dacă tabelul în care s-a adăugat un rând are o cheie primară de tip `autoincrement`. Funcția va returna valoarea cheii primare corespunzând rândului inserat. Prototipul funcției este:

```
int mysql_insert_id ([int link]).
```

Exemplu:

Inserarea unui articol în baza de date

Se dorește realizarea unui formular și a unui script PHP care realizează inserarea unui articol în tabelul `imagini`. Structura acestuia este:

Field	Type
<u>ID_Poza</u>	int(11)
<u>Fisier</u>	varchar(50)
<u>ID_Categorie</u>	int(11)
<u>Comentariu</u>	varchar(100)

Pentru a nu putea avea mai multe fișiere cu același nume, fiecare fișier încărcat va fi redenumit, noul nume fiind obținut după formula `ID_categorie + id_imagine`. Câmpul `id_inagine` fiind de tip `autoincrement`, în acest mod se va obține un nume unic.

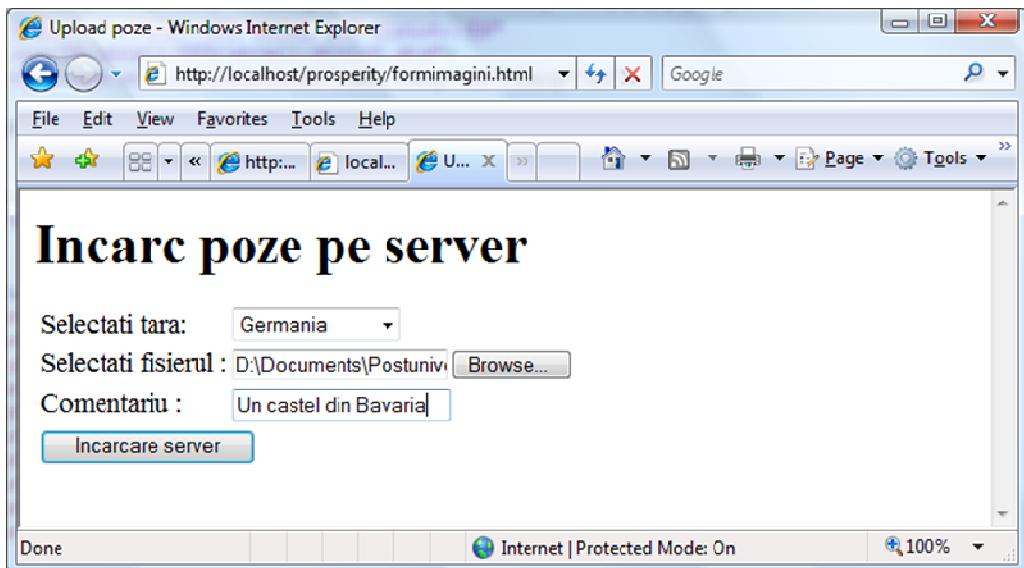
formimagini.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>Upload poze</title>
</head>
<body>
<h1>Incarc poze pe server</h1>
<form enctype="multipart/form-data" action="upimagini.php" method="post">
<table>
<tr>
<td>Selectati tara: </td><td><select name="id_tara">
<option value="0" selected>(Selectati tara)
<option value="1">Austria
<option value="2">Elvetia
<option value="3">Franta
<option value="4">Germania
<option value="5">Grecia
```

```

<option value="6">Italia
<option value="7">Spania
<option value="8">Turcia
<option value="9">Ungaria
</select>
</td>
</tr>
<tr><td>Selectati fisierul : </td><td><input type="file" name="fisier" /></td></tr>
<tr><td>Comentariu : </td><td><input type="text" name="comentariu" maxlength="98" /></td></tr>
<tr><td colspan="2"><input type="submit" value="Incarcare server"></td></tr>
</table>
</form>
</body>
</html>

```



În controlul de tip *select* având numele *id_tara* s-au adăugat țările din tabelul *tari*, valorile introduse prin atributul *value* fiind valorile câmpului *id_tara* din tabelul *tari*.

upimagini.php

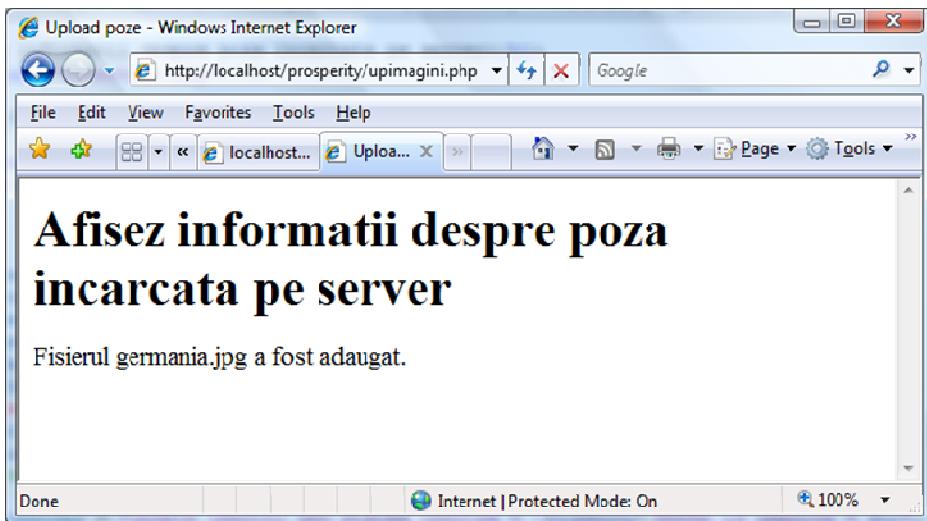
<i>id_tara</i>	<i>tara</i>	<i>imagine_tara</i>
1	Austria	austria.jpg
2	Elvetia	elvetica.jpg
3	Franta	franta.jpg
4	Germania	germania.jpg
5	Grecia	grecia.jpg
6	Italia	italia.jpg
7	Spania	spania.jpg
8	Turcia	turcia.jpg
9	Ungaria	ungaria.jpg

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>Upload poze</title>
</head>
<body>
<h1>Afisez informatii despre poza incarcata pe server</h1>
<?php include "conectare.php"; ?>
<?php
    if ($_FILES["file"]["error"] > 0)
    {
        echo "Error: " . $_FILES["fisier"]["error"] . "<br />";
        exit;
    }
    else
    {
        $tara = $_REQUEST["id_tara"];
        $coment = $_REQUEST["comentariu"];
        $nm = $_FILES["fisier"]["name"];
        $nmtmp = $_FILES["fisier"]["tmp_name"];
        $cale = "poze/a.jpg";
        $comanda1 = "insert into imagini values ('null','a.jpg','$tara','$coment')";
        mysql_query ($comanda1) or die(mysql_error());
        // Schimb numele a.jpg. Noul nume foloseste codul tarii si cheia primara
        // Preiau cheia primara
        $nr = mysql_insert_id();
        $numenou = (string)$tara."_".(string)$nr.".jpg";
        $comanda2 = "update imagini set fisier_imagine = '$numenou' WHERE id_imagine = $nr";
        mysql_query ($comanda2) or die(mysql_error());
        $cale = "poze/".$numenou;
        $rezultat = move_uploaded_file($nmtmp, $cale);
        if ($rezultat)
            echo "<p>Fisierul $nm a fost adaugat.</p>";
        else
            die("<p>Eroare la incarcarea fisierului</p>");
    }
?>
</body>
</html>

```

Rezultatul încărcării pe server:



- `mysql_fetch_array()`- permite preluarea unui rând dintr-o mulțime de selecție realizată ca urmare a unei interogări. Valorile sunt memorate într-un sir asociativ în care numele cheilor coincid cu numele câmpurilor. Prototipul funcției este :

array `mysql_fetch_array` (int rezultat).

Exemplu:

Afișarea conținutului unui tabel al bazei de date

tabtari.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>Tabelul tari</title>
</head>
<body>
<h1>Afisez informațile din tabelul <em>tari</em></h1>
<?php include "conectare.php"; ?>
<?php
$comanda = "select * from tari order by tara";
$rezultat = mysql_query ($comanda) or die(mysql_error());
echo "<table>";

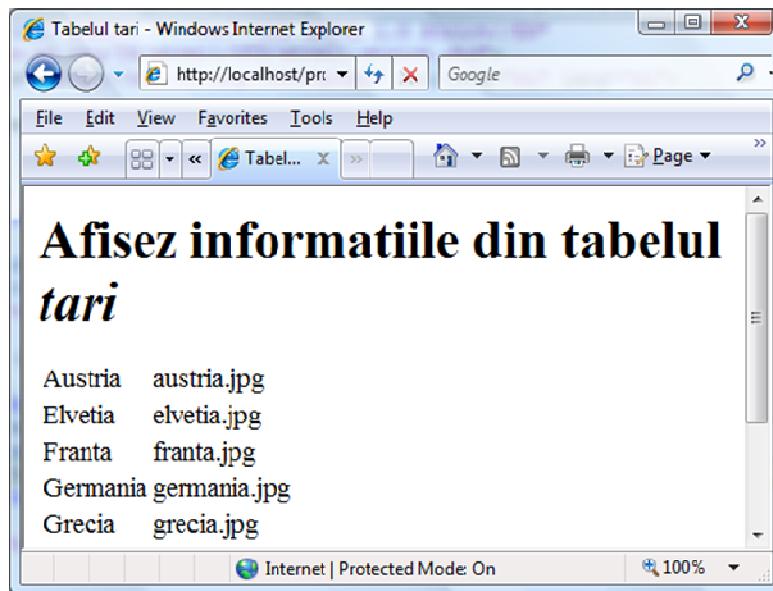
while ($linie = mysql_fetch_array($rezultat))
{
    echo "<tr>";
    $tar = $linie["tara"];
    $img = $linie["imagine_tara"];
    echo "<td>$tar</td><td>$img</td>";
```

```

        echo "</tr>";
    }
    echo "</table>";
?
</body>
</html>

```

Rezultat:



- `mysql_num_rows()` - returnează numărul de linii dintr-o mulțime de selecție creată ca urmare a execuțării unei comenzi SQL `select`. Prototipul funcției este:

```
int mysql_num_rows (int rezultat);
```

Exemplu:

```

$query = "select * from utilizatori";
$rez = mysql_query($query) or die (mysql_error());
if (mysql_num_rows($rez) == 0)
{
    echo "Tabelul utilizatori nu contine inregistrari";
} else {
    // se prelucrează rezultatele
}

```

- *mysql_affected_rows()* – returnează numărul de linii afectate de o comandă *update*, *insert* sau *delete*. Prototipul funcției este:

```
int mysql_affected_rows ([int link]).
```

Exemplu:

```
$query = "delete from utilizatori where id_utilizator = $v";
$rez = mysql_query($query) or die (mysql_error());
$sterse = mysql_affected_rows();
if ($deleted == 0)
{
    echo "Nimic nu s-a sters!";
} else {
    echo "Ati sters $sterse linii din tabelul utilizatori.";
}
```

- *mysql_error()* – afișează un mesaj care indică natura erorii care s-a produs în timpul execuției unei comenzi SQL. Prototipul funcției sunt:

```
string mysql_error ( [int link] ).
```

Alte funcții importante

- *date()* – returnează data și ora curentă. Prototipul ei este:

```
string date (string format [int timp])
```

Dacă este prezent al doilea argument, funcția va forma din valoarea acestuia un sir de caractere reprezentând o dată, conform formatului indicat ca prim argument.

Exemplu:

```
echo date("F d, Y g:i a");
```

Potrivit documentației, poate imprima ceva de forma "May 12, 2002 19:21 pm".

Caracterele care pot fi folosite în sirul *format* sunt:

Indicator	Semnificație
a	am sau pm
A	AM sau PM
d	ziua din lună (01 la 31)
D	ziua din săptămână, 3 litere, în engleză (ex. Fri)
F	luna, text lung, engleză (ex. January)
g	ora, 0, 1 ... 12
G	ora, 0 la 24
h	ora, 01, 02, ... 12
H	ora, 01 ... 24
i	minutul, 00 la 59
I (L mic)	ziua din săptămână, text lung (ex. Friday)
m	luna în cifre (01 la 12)
M	luna, text 3 caractere (ex. Jan)
n	luna, fără zerouri în fața lunilor dintr-o cifră
s	secunda, 00 la 59
S	sufixul pt. numerale ordinale, engl. (ex. th, nd)
w	ziua din săptămână, numeric, 0 la 6
Y	anul, patru poziții zecimale
z	ziua din an, numeric, 0 la 365

- *mkttime()* – crează o dată. Prototipul ei este:

```
int mkttime (int ora, int minutul, int secunda, int luna, int ziua, int anul)
```

Exemplu:

```
$anul = 2002;
$luna = 5;
$ziua = 25;
echo date ("I F d, Y", mkttime(0,0,0, $luna, $ziua+30, $anul));
```

va imprima "Monday June 24, 2002". Funcția permite calcule cu zile și ore, ca în exemplu.

- *mail()* – permite trimiterea unui e-mail dintr-un script. Prototipul ei este:

```
bool mail(string to, string subject, string message [, string additional_info]).
```

Exemplu:

```
$mesaj = "Comanda dv. a fost înregistrată și va fi tratată cu prioritate";
$adr = "JohnDoe@somwhere.net";
mail($adr, "Subject Confirmare", $mesaj,
     "From: International Trade\r\nReply-to: numelemeu@yahoo.com");
```

- *readfile()* – realizează deschiderea unui fișier și trimiterea direct spre browser.

Exemplu:

Trimiterea spre browser a unui fișier .txt:

```
<?php
echo "Continutul fisierului date.txt \n";
echo "<pre>";
readfile("date.txt");
echo "</pre>";
?>
```

Notă: Marcajul *<pre>* impune respectarea formatării textului din fișier, mai precis respectarea caracterelor "LF" (sfârșit de linie) conținute în acesta. În lipsa acestui marcaj informația din fișierul *date.txt* va apărea pe linii de lungime impusă de browser.

- *header()* – permite trimiterea unui antet. Antetul unei pagini web cuprinde un ansamblu de informații având un format predefinit și este trimis de serverul de web browserului *înaintea informației propriu-zise*. Antetul precizează, printre altele, tipul de informații care vor urma să fie trimise în continuare. Parametrul din antet care precizează natura informațiilor care urmează să fie transmise în continuare este *"Content-type"*. Exemple:

- Content-type : text/html - fișier în format text sau html;
- Content-type : application/xhtml+xml - fișier în format .xhtml;
- Content-type : application/pdf - fișier în format .pdf;
- Content-type : image/jpeg - fișier în format .jpg;
- Content-type : image/png - fișier în format .png;
- Content-type : application/zip - fișier în format .zip

Exemplu:

```
<?php
header('Content-type: application/pdf');
header('Content-Disposition: inline; filename="articol.pdf"');
readfile('math12934.pdf');
?>
```

Observații: 1. Caracterele trimise de funcția *header()* vor fi *primele trimise spre browser*. Aceasta este o condiție uneori dificil de verificat mai ales dacă înainte de apelul funcției *header()* sunt inserate scripturi prin *include()* sau *require()* care ar putea conține comenzi care trimit informații spre browser. Practic se va verifica dacă există comenzi *echo* care s-ar putea executa înaintea apelului funcției *header()*.

2. *Content-Disposition* poate lua valorile *inline* sau *attachment*. În primul caz browserul va încerca să afișeze fișierul în fereastra sa iar dacă tipul fișierului nu permite afișarea, va afișa ferestra de dialog care permite salvarea acestuia pe disc. În cazul al doilea, se afișează direct fereastra de dialog pentru salvarea pe disc a fișierului.



Exemple suplimentare:

1. Pagină web care apelează un script PHP în vederea transferului de pe server a unui fișier în format .pdf:

```
<html>
<head>
<title>Afisez un fisier .pdf</title>
</head>
<body>
Afiseaza <a href="pdf.php?afisez=carte14">cartea</a>
</body>
</html>
```

```
<?php
$pdf = $_REQUEST["afisez"].".pdf";
header('Content-type: application/pdf');
header('Content-Disposition: attachment; filename='.$pdf);
readfile($pdf);
?>
```

Dacă primul fișier este tot un script PHP, numele fișierului de afișat (*carte14*) poate proveni dintr-un articol al unui tabel al unei baze de date.

2. Trimiterea unui fișier .gif:

```
<?php
$file = 'monkey.gif';

if (file_exists($file)) {
    header('Content-Description: File Transfer');
    header('Content-Type: application/octet-stream');
    header('Content-Disposition: attachment; filename='.$file);
    header('Content-Transfer-Encoding: binary');
    header('Expires: 0');
    header('Cache-Control: must-revalidate, post-check=0, pre-check=0');
    header('Pragma: public');
    header('Content-Length: ' . filesize($file));
    ob_clean();
    flush();
    readfile($file);
    exit;
}
?>
```

Notă: În acest ultim exemplu antetul trimis spre browser este complet, cu toate informațiile posibile.

3.6.4 Sesiuni

Aplicațiile din categoria serverelor Web nu permit transferul unor informații de la o pagină vizitată la alta. Un astfel de server se limitează la a trimite spre client (browser) fișierul cerut sau lansează în execuție o aplicație, ca în cazul scripturilor asociate formularelor de exemplu.

Există cazuri în care este necesar ca în timpul navigării prin sait anumite informații să fie "transportate" de la o pagină la alta. Un exemplu clasic este procesul de cumpărare dintr-un magazin virtual. După cumpărarea unui articol dintr-o pagină se trece la o altă pagină de unde se dorește un alt articol și aşa mai departe. Articolele deja selectate în vederea cumpărării trebuie memorate până la definitivarea comenzi.

Limbajul PHP permite diverse soluții pentru această problemă, în cele ce urmează fiind prezentat lucrul cu *sesiuni* (eng. *sessions*). O sesiune inițiată într-o pagină va fi menținută deschisă în timpul navigării și va fi închisă doar atunci când nu mai este necesară. În cazul unui magazin virtual închiderea sesiunii se va realiza după definitivarea comenzi de cumpărare.

Pe toată durata existenței unei sesiuni, în cadrul acesteia poate fi definit un set de variabile care vor păstra informațiile dorite. De exemplu, în cazul cumpărării dintr-un magazin virtual, în cadrul sesiunii deschise se poate utiliza o variabilă de tip sir (array) sau un sir de caractere care va conține codurile articolelor selectate la un moment dat.

Funcții pentru gestiunea sesiunilor

- *session_start()* – realizează inițierea unei sesiuni sau reluarea unei sesiuni deja inițiate.

Datorită acestei funcții duble, toate scripturile PHP din zona din sait în care operăm cu sesiuni vor începe cu *session_start()*. Apelul va fi plasat înaintea oricărui alt cod PHP sau HTML. Exemplu:

```
<?
session_start();
>
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<head>
    <title>Documentatie</title>
    <LINK REL="STYLESHEET" TYPE="text/css" HREF="stil.css">
    . . . . .
```

În cazul inițierii unei sesiuni, `session_start()` va genera un cod unic, accesibil ulterior prin variabila `$PHPSESSID`. Concomitent, pe discul serverului este scris un fișier care va păstra variabilele sesiunii respective.

Scriptul PHP trimite automat browserului valoarea variabilei `$PHPSESSID` iar browserul o memorează sub forma unui *cookie* (secvență de cod păstrată de browser pe disc, într-un fișier). Dacă browserul are dezafectată această facilitate (în IE8 se accesează *Tools / Internet Options / Privacy / Advanced*) acest mecanism nu mai poate funcționa. În exemplele următoare se va da și soluția ocolirii acestui mecanism.

Odată stabilit codul sesiunii, următoarele apeluri ale funcției `session_start()` vor realiza doar reluarea sesiunii deja începute.

Memorarea unei valori specifică unei sesiuni începute se realizează folosind sirul asociativ `$_SESSION`. Exemplu:

```
<?
session_start();
...
$_SESSION["cos_cump"] = $id_produs;
...
```

- `isset()` – permite testarea unei variabile a sesiunii. Funcția returnează `true` sau `false` după cum variabila testată este inițializată sau nu.

Exemplu:

```
<?
session_start();
if(isset($_SESSION['cos_cump']))
{
    $cos = $_SESSION['cos_cump'];
    $cos = $cos . ",".$id_produs; // adaug un produs in cos
    $_SESSION['cos_cump'] = $cos;
} else
{
    $_SESSION["cos_cump"] = $id_produs;
}
...
```

- `unset()` – realizează anularea unei variabile a sesiunii. Exemplu:

```
<?
session_start()
...
unset($_SESSION['cos_cump']); // golesc cosul
...
```

- `session_destroy()` – realizează distrugerea unei sesiuni. Următorul apel al funcției `session_start()` va genera un alt cod de sesiune. În cazul magazinelor virtuale, `session_destroy()` este apelată după încheierea tranzacției (ieșire de la casa magazinului virtual).

Notă: Dacă o tranzacție nu este încheiată prin ieșire pe la casă, sesiunea rămâne deschisă un timp dat de o valoare din fișierul de configurare a PHP (`php.ini`). În fișierul de configurare al PHP apare linia:

```
session.gc_maxlifetime = 1440
```

Aceasta stabilește timpului de menținere a unei sesiuni la 1440 s (24 minute). După trecerea acestui timp sesiunea rămasă deschisă va fi închisă și fișierul aferent de pe discul serverului vor fi automat distruse. Există posibilitatea modificării duratei de menținere deschisă a unei sesiuni prin apelul funcției `ini_set()`, ca în exemplul de mai jos:

```
<?php
    ini_set("session.gc_maxlifetime", "7200");
?>
```

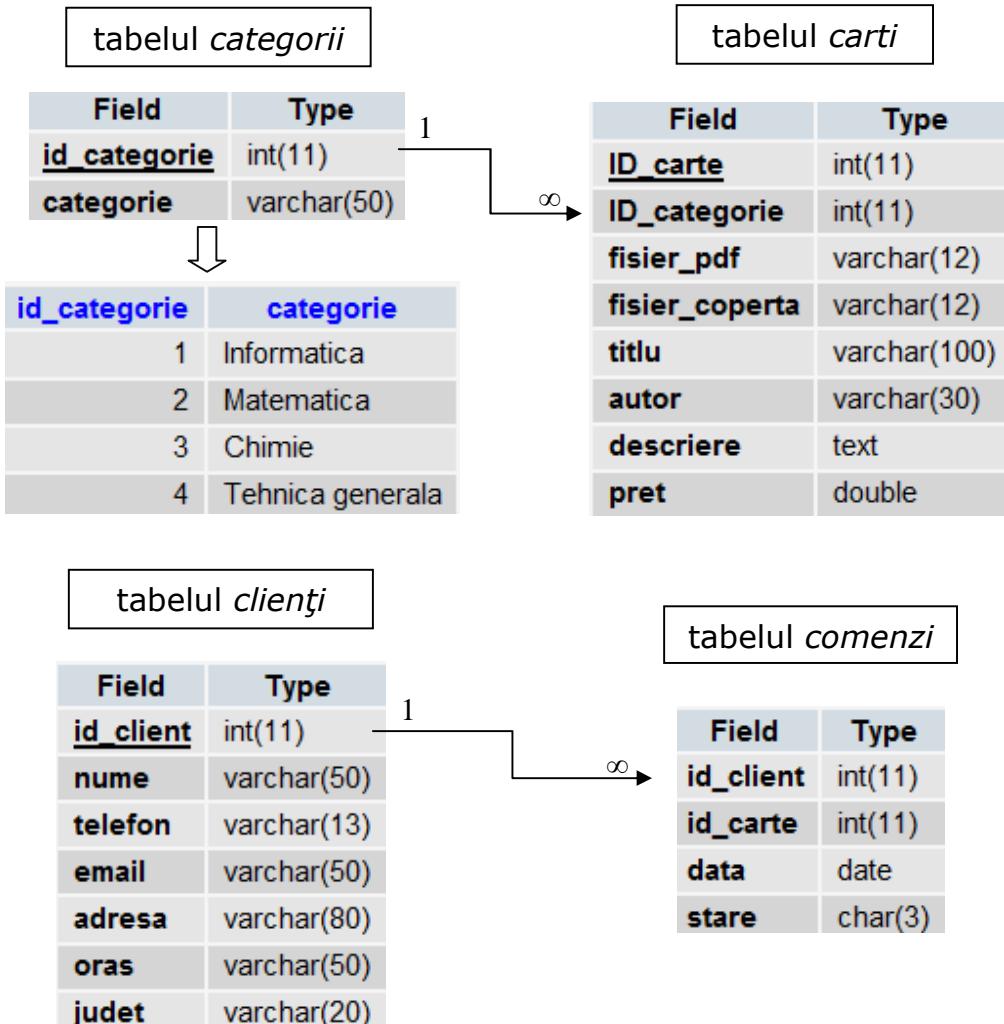
Scriptul care conține acest apel va modifica durata de menținere activă a sesiunii deschise la două ore (7200 s).

Exemplu fundamental:

Se dorește realizarea unui magazin virtual care vinde cărți în format electronic. Se dorește asigurarea următoarelor funcții:

- adăugarea în coșul de cumpărături a unei cărți prin selectare cu mouse-ul ;
- afișarea permanentă a numărului de cărți selectate în vederea cumpărării;
- afișarea coșului de cumpărături;
- cumpărarea propriu-zisă constând din afișarea unui formular pentru preluarea datelor cumpărătorului. Butonul de tip *Submit* va realiza copierea datelor cumpărătorului în fișierul pentru clienți și memorarea articolelor din coșul de cumpărături în tabelul *comenzi*.

A. Crearea bazei de date "libraria"



B. Fișierul index.php

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <title>Carti</title>
  <LINK REL="STYLESHEET" TYPE="text/css" HREF="stil.css">
</head>
<body>
<div id="bloc_ext">
  <div id="header"><h1>Magazinul de carti</h1>
  <hr></div>

```

```

<div id="colstg">
    <h3>Categorii</h3>
    <p>Selectati categoria dorita. </p>
    <?php
        include 'colstg.php';
    ?>
</div>
<div id="main">
    <h3>Carti in categoria selectata</h3>
</div>
</div>
</body>
</html>

```

Fișierul *colstg.php* afișează conținutul coloanei din stânga (din blocul *colstg*):

```

<?php
mysql_connect(localhost, "root", "") or die ("Nu ma pot conecta la server");
mysql_select_db("libraria") or die ("Nu pot selecta baza de date carti");
// Afisez categoriile
$cda_SQL="SELECT * FROM categorii";
$rezultat=mysql_query($cda_SQL) or die(mysql_error());
echo "<ul>";
while ($row = mysql_fetch_array($rezultat)) {
    echo '<li><a href="caut.php?id='.$row['id_categorie'].'">'.$row['categorie'].'</a></li>';
}
echo "</ul>";
?>

```

Rezultat:



C. Scriptul de afișare a cărților dintr-o anumită categorie (caut.php)

Apelul se face prin selectarea unei intrări din lista de categorii afișată de *colstg.php*. În momentul apelului, scriptului i se transmite codul categoriei. Pentru prima categorie apelul va fi de exemplu *caut.php?id=1*.

```
<?
session_start();
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
 "http://www.w3.org/TR/html4/strict.dtd">
<head>
    <title>Carti</title>
    <LINK REL="STYLESHEET" TYPE="text/css" HREF="stil.css">
</head>
<body>
<div id="bloc_ext">
    <div id="header"><h1>Magazinul de carti</h1>
    <hr>
    </div>
    <div id="colstg">
        <h3>Categorii</h3>
        <p>Selectati categoria dorita. </p>
        <?php
            include 'nrart.php';
            include 'colstg.php';
        ?>
    </div>

    <div id="main">
        <h2>Carti in categoria
        <?php
        // Preiau codul categoriei
        $id = $_REQUEST['id'];
        // Caut in tabelul categorii numele categoriei
        $comanda="SELECT * FROM categorii where id_categoria = $id";
        $rez=mysql_query($comanda) or die(mysql_error());
        $linie = mysql_fetch_array($rez);
        $categ = $linie['categorie'];
        echo $categ. "</h2>";
        echo "<P>Selectati cartea dorita.</p>";
        // Selectez cartile din categoria $id
        $c_da="SELECT * FROM carti where id_categoria = $id";
        $rezultat=mysql_query($c_da) or die(mysql_error());

        $i = 1; // $i este un contor
        while ($linie = mysql_fetch_array($rezultat)) {
            echo "<p>$i'. ' .
                '</p><p> '$linie['titlu'] . '</p>'";
            // Creez referinta pentru apelarea scriptului cump.php
            echo '<p><a href="cump.php?id='.$id.'&id_carte='.$linie['ID_carte'].'"';
            echo "'>Cumpar cartea!</a></p><hr>';
        }
    </div>
</body>
```

```

        $i++;
    }
    echo numar_articole();
?>
</div>
</div>
</body>
</html>

```

Rezultat:

Magazinul de carti

Categorii	
Selectati categoria dorita.	<p>1. Design Patterns: Elements of Reusable Object-Oriented Software Cumpar cartea!</p> <hr/> <p>2. High Performance MySQL Cumpar cartea!</p>
Aveti 1 articole in cosul Dv.	

Fișierul *nrart.php* conține funcția *numar_articole()*. Această funcție returnează un mesaj care conține numărul de articole din coșul cu cumpărături. Mesajul conține și o referință spre scriptul *cos.php* care afișează conținutul coșului cu cumpărături.

```
<?php
function numar_articole()
{
    if(isset($_SESSION['cos_cumparaturi']))
    {
        $cos = $_SESSION['cos_cumparaturi'];
        $articole = explode(',',$cos);
        $nr = count($articole);
        return '<p>Aveti '.$nr.'<a href="'.$cos.'.php"> articole in cosul Dv. </a></p>';
    }
    else
        return '<p>Cosul cu cumparaturi este gol.</p>';
}
?>
```

Selectarea řirului de caractere "*Cumpar cartea*" declanșează executarea scriptului *cump.php*. Acest script adaugă o carte în cosul cu cumpărături. Scriptul cuprinde și o secvență de verificare a existenței prealabile a cărții în coș. În cazul în care cartea era deja în coș adăugarea nu mai are loc. După adăugare se apelează funcția *header()* pentru a reveni la *caut.php*. Pentru ca scriptul *caut.php* să afișeze cărțile din aceeași categorie, scriptul *cump.php* este apelat în *caut.php* cu doi parametri, după modelul: *cump.php?id=1&id_carte=12*.

```
<?php
session_start();
// Adaug cartea in cos
$idcateg = $_REQUEST['id'];
$idcarte = $_REQUEST['id_carte'];
$gasit = false;
if(isset($_SESSION['cos_cumparaturi']))
{
    $cos = $_SESSION['cos_cumparaturi'];
    // Verific daca a mai fost adaugata anterior
    $articole = explode(',',$cos);
    foreach ($articole as $item) {
        if($item == $idcarte)
        {
            $gasit = true;
            break;
        }
    }
    if(!$gasit)
        $cos = $cos.','.$idcarte;
}
else
{
    $cos = $idcarte;
}
$_SESSION['cos_cumparaturi'] = $cos;
header('Location: caut.php?id='.$idcateg);
?>
```

Notă: Scriptul anterior realizează un ansamblu de prelucrări după care apelează *header()* pentru a trece la un alt script care va afișa un anumit conținut pe ecranul browserului. *Principalele scripturi ale unei aplicații web fie produc conținutul afișat de browser, fie realizează o prelucrare de date încheiată cu trecerea la un script din prima categorie.*

D. Scriptul de afișare a coșului cu cumpărături (cos.php)

Scriptul afișează în blocul main un tabel conținând cărțile selectate și prețurile acestora. Sub tabel apar trei opțiuni: *Cumpăr cărțile* (se apelează scriptul *cumpăr.php*), *Inapoi în magazin* (se revine la *index.php*) și *Golesc coșul* (se apelează scriptul *golesc.php*).

```
<?php
session_start();
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<head>
    <title>Carti</title>
    <LINK REL="STYLESHEET" TYPE="text/css" HREF="stil.css">
</head>
<body>
<div id="bloc_ext">
    <div id="header"><h1>Magazinul de carti</h1>
    <hr></div>
    <div id="colstg">
        <h3>Categorii</h3>
        <p>Selectati categoria dorita. </p>
        <?php
        include 'colstg.php';
        ?>
    </div>

    <div id="main">
        <h2>Cosul cu cumparaturi</h2>
        <?php
        // Parcurg cosul cu cumparaturi
        $cos = $_SESSION['cos_cumparaturi'];
        if (!$cos) {
            echo '<p>Cosul cu cumparaturi este gol.</p>';
        } else {
            $vtotal = 0; // Pentru calculul valorii totale a cartilor din cos
            $articole = explode(',',$cos);
            echo '<table class="tcentrat">';
            foreach ($articole as $item) {
                // Caut titlul in baza de date dupa $item
                $c_da="SELECT * FROM carti where id_carte = $item";
                $rezultat=mysql_query($c_da) or die(mysql_error());
                $linie = mysql_fetch_array($rezultat);
```

```

echo "<tr><td>".$linie['titlu']. '</td><td><nobr>' . $linie['pret']. 
' lei</nobr></td></tr>'; 
        $vtotal += (double)$linie['pret'];
    }
    echo "</table>";
}
echo "<p>Cartile costa in total ".$vtotal." lei.</p>";
?>

<br /><br />
<p class="centrat"><a href="cumpar.php">Cumpar cartile</a> &nbsp; | &nbsp; <a href="index.php">Inapoi in magazin</a> &nbsp; | &nbsp;
<a href="golesc.php">Golesc cosul</a></p>
</div>
</div>
</body>
</html>

```

Design Patterns: Elements of Reusable Object-Oriented Software	84.2 lei
--	----------

E. Scriptul pentru golirea coșului cu cumpărături (golesc.php):

```

<?php
session_start();
unset($_SESSION['cos_cumparaturi']);
// Merg la index.php

```

```
header('Location: index.php');
?>
```

F. Scriptul de finalizare a cumpărăturii (cumpar.php)

Scriptul cumpar.php afișează într-un tabel un ansamblu de controale care permit preluarea datelor clientului. Trimiterea conținutului coșului spre scriptul care prelucrează datele din formular (*comand.php*) se realizează prin intermediul unui câmp ascuns (*hidden*) deoarece scripturile asociate formularelor nu li se transmit valorile memorate în sesiunea deschisă.

```
<?php
session_start();
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<head>
    <title>Carti</title>
    <LINK REL="STYLESHEET" TYPE="text/css" HREF="stil.css">
</head>
<body>
<div id="bloc_ext">
    <div id="header"><h1>Magazinul de carti</h1>
    <hr></div>
    <div id="colstg">
        <h3>Categorii</h3>
        <p>Selectati categoria dorita. </p>
        <?php
            include 'colstg.php';
        ?>
    </div>

    <div id="main">
        <h2>Informatii client</h2>
        <form method="post" action="comand.php">
        <?php
            // Mut intr-un camp ascuns (hidden) continutul cosului
            echo '<input type="hidden" name="coscump" '.
'value="'. $_SESSION['cos_cumparaturi']. '">';
            // Golesc cosul
            unset($_SESSION['cos_cumparaturi']);
        ?>
        <table class="tcentralat">
            <tr><td>Numele si prenumele: </td><td>
<input type="text" name="numepren" /></td></tr>
            <tr><td>Telefon (preferabil mobil): </td><td>
<input type="text" name="tel" /></td></tr>
            <tr><td>E-mail: </td><td><input type="text" name="email" /></td></tr>
            <tr><td>Adresa: </td><td><textarea name="adr" rows="2" cols="30">
</textarea></td></tr>
            <tr><td>Localitatea: </td><td><input type="text" name="loc" /></td></tr>
            <tr><td>Judet (sector): </td><td>
```

```

<input type="text" name="jud" /></td></tr>
<tr><td colspan="2" class="centrat"><input type="submit" name="cump"
value="Cumpar!" /> &nbsp; <input type="submit" name="abnd" value="Abandon!"
/></td></tr>
</table>
</form>
</div>
</div>
</body>
</html>

```

Carti - Windows Internet Explorer

http://localhost/carti/cumpar.php

File Edit View Favorites Tools Help

Favorites Suggested Sites Web Slice Gallery

Carti

Magazinul de carti

Categorii	Informatii client
Selectati categoria dorita.	Numele si prenumele: <input type="text"/> Telefon (preferabil mobil): <input type="text"/> E-mail: <input type="text"/> Adresa: <input type="text"/> Localitatea: <input type="text"/> Judet (sector): <input type="text"/>
	<input type="button" value="Cumpar!"/> <input type="button" value="Abandon!"/>

Done Local intranet | Protected Mode: Off 100%

G. Scriptul de adăugare a comenzii în baza de date (comand.php).

Scriptul comand.php adaugă clientul în tabelul clienti, preia valoarea curentă a cheii primare din tabelul clienti și scrie înregistrările corespunzătoare din tabelul comenzi, respectiv codul clientului, codul cartii, data comenzii și impune starea acesteia (CN - comandă neînratată). După trimiterea cărților spre cumpărător, operatorul va interveni asupra liniilor corespunzând cărților trimise prin modificarea stării (CN ar putea trece în R - rezolvată).

```
<?php
$cump = $_REQUEST['cump'];
if($cump)
{
    $cos = $_REQUEST['coscump'];
    $nume = $_REQUEST["numepren"];
    $telefon = $_REQUEST["tel"];
    $email = $_REQUEST["email"];
    $adresa = $_REQUEST["adr"];
    $oras = $_REQUEST["loc"];
    $judet = $_REQUEST["jud"];
    // Apelam strip_tags() pentru a evita marcaje nepermise
    $cos = strip_tags($cos);
    $nume = strip_tags($nume);
    $telefon = strip_tags($telefon);
    $email = strip_tags($email);
    $adresa = strip_tags($adresa);
    $oras = strip_tags($oras);
    $judet = strip_tags($judet);
    // Adaug clientul în tabelul clienti
    mysql_connect(localhost, "root", "") or die ("Nu ma pot conecta la server");
    mysql_select_db("libraria") or die ("Nu pot selecta baza de date carti");
    $c_da = "insert into clienti values(NULL,'$nume', '$telefon', '$email', '$adresa', '$oras',
'$judet')";
    mysql_query($c_da) or die(mysql_error());
    // Preiau id_client
    $idcli = mysql_insert_id();
    // Inserez articole în tabelul comenzi
    $articole = explode(',',$cos);
    foreach ($articole as $item) {
        // Adaug $item în tabelul comenzi
        $data = date('Y-m-d'); // data în format aaaa-ll-dd
        $c_da1="INSERT INTO COMENZI VALUES ('$idcli', '$item', '$data', 'CN')";
        // CN = comanda nouă
        mysql_query($c_da1) or die(mysql_error());
    }
    header("location:mesaj.php");
}
else
    header("location:index.php");
?>
```

Scriptul *mesaj.php* se obține din *index.php* modificând blocul *main* ca mai jos:

```
<div id="main">
    <h3>Comanda preluata</h3>
    <p>Un operator va va contacta telefonic in vaderea confirmarii comenzi.</p>
    <p>Va multumim!</p>
</div>
```

Observație: Mecanismul de transmitere a datelor de la o pagină la alta descriș poate servi de exemplu la menținerea autentificării unui operator care administrează un site. Astfel, după completarea formularului de logare (nume, parolă) se execută un prim script care inițiază o sesiune. În variabilele acestei sesiuni acest script va adăuga o variabilă care ar putea primi valoarea *true* sau *1*. Toate scripturile lansate în continuare vor verifica imediat după *session_start()* dacă variabila respectivă există. În cazul în care variabila nu există se va afișa imediat pagina de logare. Exemplu:

The image shows a simple login interface with a blue border. At the top center, it says "Member Login". Below that, there are two text input fields: one for "Username" and one for "Password". At the bottom center is a "Login" button.

```
<form method="post" action="veriflogin.php">
    <table border="0" align="center" cellpadding="10" cellspacing="1">
        <tr><td colspan=2 align="center"><b>Member Login </b></td>      </tr>
        <tr><td width="70">Username : </td>
            <td width="100"><input name="myusername" type="text" id="myusername" size=12></td></tr>
        <tr><td>Password : </td><td><input name="mypassword" type="password" id="mypassword" size=12></td></tr>
        <tr><td colspan=2 align="center"><input type="submit" name="Submit" value="Login"></td></tr>
    </table>
</form>
```

Fișierul **veriflogin.php**:

```
<?php
session_start();
mysql_connect(localhost, "root", "") or die ('Eroare la conectare la server');
mysql_select_db("carti") or die ('Eroare la conectare la baza de date');
```

```

// Preiau myusername si mypassword
$utilizator=$_REQUEST['myusername'];
$parola=$_REQUEST['mypassword'];

// Pentru a proteja scriptul de caractere ilegale, apelez strip_tags si mysql_real_escape_string
$utilizator = strip_tags($utilizator);
$parola = strip_tags($parola);
$utilizator = mysql_real_escape_string($utilizator);
$parola = mysql_real_escape_string($parola);

$comanda="SELECT * FROM clienti WHERE nume='$utilizator' and parola='$parola'";
$rezultat=mysql_query($comanda);

// mysql_num_rows numara liniile dintr-o multime de selectie
$count=mysql_num_rows($rezultat);
// Daca s-a gasit un client care are $utilizator si $parola, multimea de sel. trebuie sa contina o
linie

if($count==1){
    // Inregistrez in sesiune variabila autentificat si continui cu "login_success.php"
    $_SESSION['autentificat'] = true;
    header("location:login_success.php");
}
else {
    echo "Eroare nume sau parola!";
}
?>

```

Toate scripturile apelate în continuare vor verifica existența variabilei de sesiune "*autentificat*". De exemplu scriptul *login_succes.php* va începe astfel:

```

<?
session_start();
if(!$_SESSION['autentificat']){
    header("location:index.php");
}
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<head>
...

```

3.7 Publicarea saitului

Odată încheiată testarea folosind XAMPP a unui sait, următorul pas este publicarea acestuia.

3.7.1 Găzduirea saitului

Un sait web poate fi găzduit pe un server aparținând unei firme specializate. Furnizorii de servicii Internet oferă de obicei această facilitate dar există zeci de firme care și-au făcut din găzduirea saiturilor web o afacere.

Exemple: <http://rohost.com/>, <http://www.redhost.ro/>, <http://gazduire.domeniu-ro.ro>, <http://www.exclusivehosting.net/> etc.

O soluție alternativă este punerea în funcțiune a unui server propriu, dar pentru saituri personale sau ale unor firme mici, cu trafic redus, soluția nu este avantajoasă. Costul serverului și costul energiei consumate sunt cu siguranță mult superioare costurilor antrenate de folosirea unei firme specializate.

Stabilirea numelui domeniului se realizează folosind serviciul *Whois...* oferit de *Romanian Top Level Domain* () organismul abilitat să superviseze în Romania domeniile având sufixul *.ro*.

The screenshot shows a web page titled "Romania Top Level Domain". Below it, a sub-section is titled "WHOIS.ROTLD.RO - server WHOIS pentru domeniile .ro". A text input field contains placeholder text: "Introduceti <name>.ro, <name>.com.ro, <name>.org.ro, <name>.tm.ro, <name>.nt.ro, <name>.nom.ro, <name>.info.ro, <name>.rec.ro, <name>.arts.ro, <name>.firm.ro, <name>.store.ro, <name>.www.ro pentru detaliile despre un domeniu:". Below this is a search bar with the text "Cautare in baza de date Whois: montana.ro" and a blue "Enter" button.

Dacă verificarea relevă existența unui domeniu având numele specificat trebuie găsită o altă soluție.

The web gateway to the ROTLD *whois* database

```
% whois.rotld.ro :  
%  
% Rights restricted by copyright.  
%  
% Specifically, this data MAY ONLY be used for Internet operational  
% purposes. It may not be used for targeted advertising or any  
% other purpose.  
%  
% Este INTERZISA folosirea datelor de pe acest server in oricare  
% alt scop decat operarea retelei. In special este INTERZISA  
% folosirea lor in scopuri publicitare.  
  
domain-name: montana.ro  
description: Poplinks srl  
description: Oltului 5, bloc 12, ap 15  
description: Brasov  
description: RO
```

Dacă s-a reușit identificarea numelui potrivit, se selectează în pagina RoTLD legătura *Inregistrare domeniu nou* și se urmează pașii ceruți de formularele afișate:

RoTLD > Domenii .ro > Inregistrare Domeniu Nou

Formularul de inregistrare pentru domenii .ro

Acest formular este pentru inregistrarea unui domeniu nou cu urmatoarele ierarhii: **.ro, .com.ro, .org.ro, .tm.ro, .nt.ro, .nom.ro, .info.ro, .rec.ro, .arts.ro, .firm.ro, .store.ro and .www.ro.**

Introduceti numele domeniului pe care doriti să-l înregistriati, selectati subdomeniul si apoi apăsați pe "Enter".

e-jucu	. ro	▼	Enter
e.g. mycompany		e.g. .org.ro	

Prin formularul de înregistrare se vor cere o serie de informații despre persoana sau firma care înregistrează domeniul. În partea sa finală formularul cere precizarea numelui și a adresei IP a cel puțin unui server conectat la Internet pe care rulează o aplicație de tip server DNS (*Domain Name Service*). Aceste informații vor fi obținute de la firma care va găzdui saitul sau de la furnizorul de servicii Internet, dacă saitul este găzduit pe un server propriu.

Servere DNS

DNS Primar (Nume Host si adresa IP):	<input type="text"/>	(*)
DNS Secundar (Nume Host si adresa IP):	<input type="text"/>	
DNS 3 (Nume Host si adresa IP):	<input type="text"/>	
DNS 4 (Nume Host si adresa IP):	<input type="text"/>	
DNS 5 (Nume Host si adresa IP):	<input type="text"/>	

După trimiterea prin Internet a cererii de înregistrare va urma emiterea de către RoTLD a unei facturi proforma pentru suma de 51.26\$ + TVA corespunzătoare procesării cererii, se va realiza plata și în final se va primi factura originală și contractul care va confirma înregistrarea domeniului cerut.

Pașii următori înregistrării numelui de domeniu depind de soluția de găzduire. Dacă găzduirea se realizează pe un server al unei firme specializate se va încheia cu firma un contract după care se va transfera saitul pe serverul firmei. Transferul se va realiza folosind aplicația pusă la dispoziție de firma gazdă.

Dacă saitul este găzduit pe un server propriu se va comunica furnizorului de servicii Internet numele domeniului înregistrat pentru a crea în serverul său DNS (rulând pe serverul a cărui adresă IP a fost comunicată la RoTLD!) o legătură între numele domeniului și IP-ul calculatorului care va găzdui efectiv saitul.

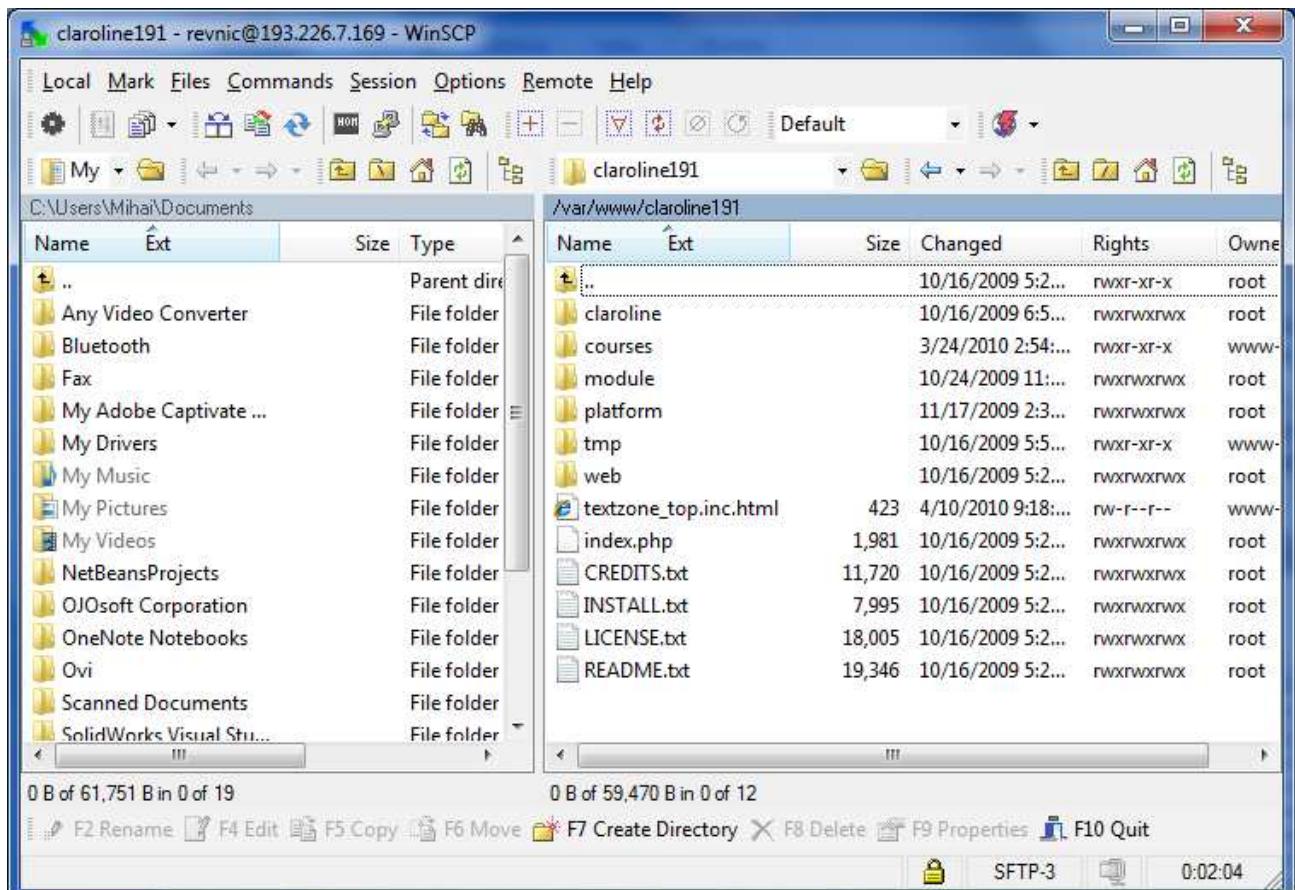
3.7.2 Găzduirea saitului web pe serverul propriu

Datorită stabilității și siguranței în funcționare majoritatea serverelor pentru Internet funcționează sub UNIX sau Linux, aplicația de tip server pentru web cea mai folosită fiind *Apache*.

Pentru regăsirea fișierelor solicitate prin Internet de către aplicații client, pe un server funcționând sub Linux (varianta Ubuntu) fișierele saiturilor sunt dispuse în directoare prestabilite, de exemplu :

- paginile web în directorul `/var/www` sau în directoare derivate din acesta,
- programele executabile (altele decât scripturile scrise în PHP!) asociate formularelor sau aplicațiile care sunt lansate pe server ca urmare a selectării unei legături) în `/usr/lib/cgi-bin` sau într-un director derivat,
- fișierele accesibile aplicațiilor de tip server de fișiere transferate folosind protocolul *ftp* sunt dispuse în directorul `/home/ftp` sau într-un director derivat.

Pentru a putea separa fișierele diferitelor saituri înregistrate pe discul serverului se vor crea directoare derivate din directoarele menționate, în care fișierele vor fi plasate după aceleași reguli (documentele în format hipertext în directoare derivate din `/var/www`, etc.).



Dacă o cerere de pagină web care parvine serverului nu specifică numele fișierului care trebuie transmis, serverul va trimite din directorul precizat fișierul *index.html*.

Dacă o cerere de document în format hipertext conține numai numele serverului pe care sunt plasate fișierele unui sait, serverul Apache va furniza de exemplu documentul /var/www/index.html.

<http://postuniv.east.utcluj.ro/> → /var/www/index.html
<http://postuniv.east.utcluj.ro/sitecj/mon/> → /var/www/sitecj/mon/index.html

Există însă posibilitatea legării numelui unui domeniu de un alt director de pe server. Pentru aceasta va fi editat fișierul de configurare a serverului *Apache*, *httpd.conf*. Astfel în secțiunea *virtual hosts* a acestuia se va adăuga blocul:

```
<VirtualHost *>
ServerName www.domeniu.ro
DocumentRoot /var/www/director_sit
</VirtualHost>
```

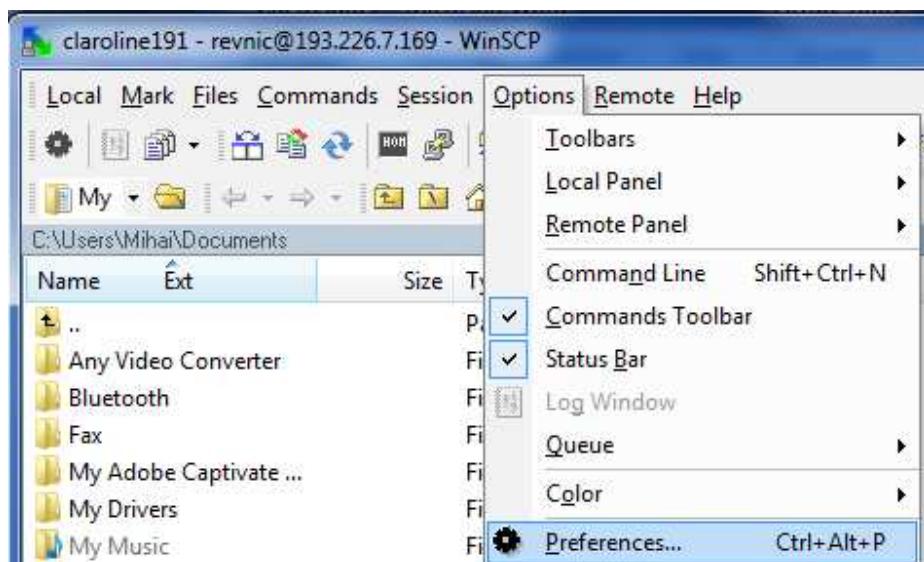
3.7.3 Utilitare: Total Commander și WinSCP

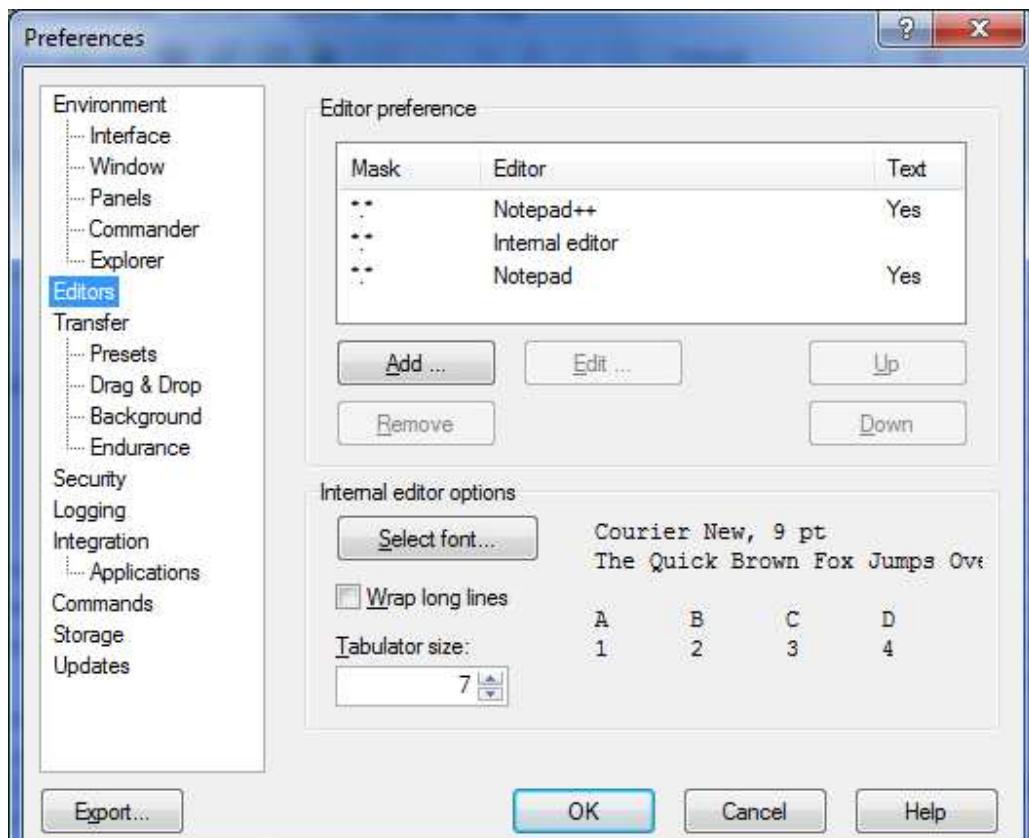
Total Commander sau *WinSCP* sunt aplicații similare unui vechi utilitar, *Norton Commander*. Aceste aplicații afișează la pornire două panouri, facilitând transferul fișierelor dintr-un director într-altul.

Transferul fișierelor unui sait web de pe calculatorul pe care s-a realizat saitul pe serverul care îl va găzdui se poate realiza folosind protocolul FTP (*File Transfer Protocol*). *Total Commander* sau *WinSCP* permit transferul fișierelor spre server folosind acest protocol, dar înainte de a realiza transferul trebuie realizată o conexiune la serverul care va găzdui saitul. Configurarea conexiunii presupune introducerea unor parametri: numele serverului, numele utilizatorului, parola și eventual directorul al căruia conținut trebuie afișat.

După conectare, unul dintre panouri afișează conținutul unui director de pe discul serverului, iar celălat panou afișează un director al calculatorul de pe care se transferă fișierele. După selectarea fișierelor de transferat se apasă tasta F5.

Modificarea conținutului unui fișier de pe server se poate face prin deschiderea acestuia într-o aplicație rulând pe calculatorul personal, modificare și apoi transfer înapoi pe server. Activarea editorului se realizează selectând (F4 Edit) sau cu un dublu clic pe fișier. În exemplul dat în continuare s-a folosit editorul **Notepad++**. Aceasta a fost adăugat în lista de editoare utilizabile în WinSCP accesând în meniul aplicației *Options / Preferences / Editors*.





3.7.4 Publicarea saitului creat

Pentru publicarea (transferul) unui sait pe un server conectat la Internet sunt necesari pașii următori :

1. Crearea de către administratorul serverului (conectat ca *root*) a structurii de directoare necesare :

- un director derivat din */var/www/* pentru *index.html* și alte fișiere de tip hipertext, imagini, sunet, animație, film;
- un director purtând de regulă același nume și derivat din */usr/lib/cgi-bin* ;
- un director purtând de regulă același nume și derivat din */home/ftp* dacă în pagini există referințe spre fișiere care vor fi transferate folosind protocolul *ftp* ;

2. Modificarea de către administrator (*root*) a proprietarului acestor directoare folosind comanda UNIX *chown* :

```
# chown proprietar director[director1...]
```

3. Transferul de către proprietar a fișierelor sitului. Transferul se poate face cu ajutorul unui utilitar (WinSCP, *Windows Commander*, *WS_FTP*, *FTP Explorer* etc). După conectarea la server se vor selecta pe rând directoarele create și se vor transfera fișierele corespunzătoare.

4. Dacă saitul conține scripturi sau alte aplicații care se vor executa pe server și care necesită recompilare, se va lansa utilitarul PuTTY și se vor crea executabile. Drepturile de acces la directorul în care aplicațiile vor fi plasate (derivat din */usr/lib/cgi-bin*) vor fi modificate folosind comanda UNIX *chmod* :

```
$ chmod 755 director
```

Dacă scriptul CGI trebuie să scrie într-un director, drepturile de acces trebuie să fie impuse cu comanda: \$ chmod 777 *director*.

Ultimul pas este editarea fișierului de configurare a serverului Apache, *httpd.conf* pentru a realiza asocierea numelui domeniului cu directorul care conține fișierele saitului.

3.7.5 Ameliorarea vizibilității sitului

Odată realizat și publicat, un sit web este accesibil din întreaga rețea. Pentru a fi ușor de localizat de către potențialii cititori se pot lua niște măsuri suplimentare :

1. Identificarea unui ansamblu de cuvinte cheie definitorii pentru sait și adăugarea lor în secțiunea header.

Pentru a stabili corect cuvintele cheie se pot accesa saituri specializate ca [Wordtracker](#) sau [Keyword Discovery](#). Includerea în sait a cuvintelor cheie se realizează adăugând în secțiunea *<head>* marcaje *<meta>*. Aplicațiile de căutare folosesc informațiile precizate folosind atributul *name* și *contents* din cadrul acestui maraj. Atributul *name* poate lua două valori : *name=description* și *name=keywords*. În funcție de valoarea alesă, atributul *contents* va conține un sir de caractere precizând conținutul site-ului respectiv cuvinte cheie definitorii pentru subiectul tratat.

Exemplu:

```
<html>
<head>
    <meta name="keywords" content="Apple Computer, Power Macintosh, PowerBook, AppleWorks, WebObjects, iMovie, QuickTime, Desktop Movies, Software, Operating Systems, Mac OS, iMac, iBook">
    <meta name="Description" content="Visit www.apple.com for the latest news, the hottest products, and technical support resources from Apple Computer, Inc.">
    <meta name="Date.Modified" content="20102104">
    <title>Apple computers</title>
</head>
```

Important: Datorită modului în care aplicațiile de căutare stabilesc relevanța unui sait în raport cu un cuvânt cheie, aceleași cuvinte cheie trebuie să fie folosite în titlul paginilor (`<title> ... </title>`) și în paginile saitului, în special în titluri și subtitluri sau ca valori ale atributului "alt" din marcajele ``.

2. Actualizarea frecventă a saitului.

Aplicațiile de căutare vor lăsa la urma listei afișate ca urmare a unei căutări fișierele vechi.

- 3. Luarea tuturor măsurilor posibile pentru informarea potențialilor beneficiari:** anunțuri, afișe, antetele documentelor firmei, menționarea sitului pe cărțile de vizită etc.
- 4. Înregistrarea sitului la principalele aplicații de căutare** disponibile, dacă acestea oferă această facilitate: *Google, Yahoo, AltaVista, Lycos, Webcrawler*, etc.

3.7.6 Securitatea în Internet

Calculatoarele conectate în rețea oferă posibilitatea transferului autorizat sau neautorizat de fișiere. Problemele de securitate a datelor în cazul în care un calculator este conectat în rețea implică securizarea tuturor factorilor care participă la funcționarea acestuia :

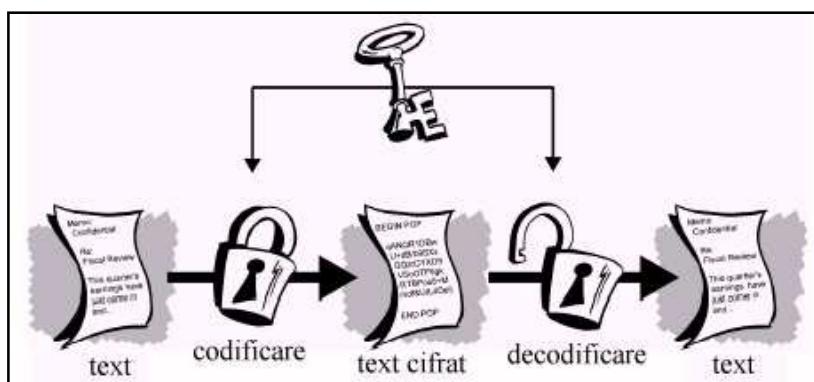
- *Aplicația client (navigator);*
- *Sistemul de operare al clientului;*
- *Rețeaua clientului;*
- *Rețeaua Internet;*
- *Rețeaua locală a serverului;*
- *Sistemul de operare al serverului;*
- *Programul server web.*

Principalul mod de pătrundere într-un astfel de sistem rămâne cunoasterea parolei unuia dintre utilizatorii autorizați să folosească serverul pe care sunt stocate datele de protejat. În lipsa unei parole se poate încerca găsirea parolei folosind programe care accesază serverul repetat, folosind ca și parole cuvinte din dicționare. Pentru a îngreuna pătrunderea pe server pe această cale este necesară utilizarea obligatorie în construirea parolelor a cifrelor sau a unor caractere speciale și limitarea drastică a numărului de utilizatori autorizați ai calculatorului.

3.7.7 Criptarea informațiilor

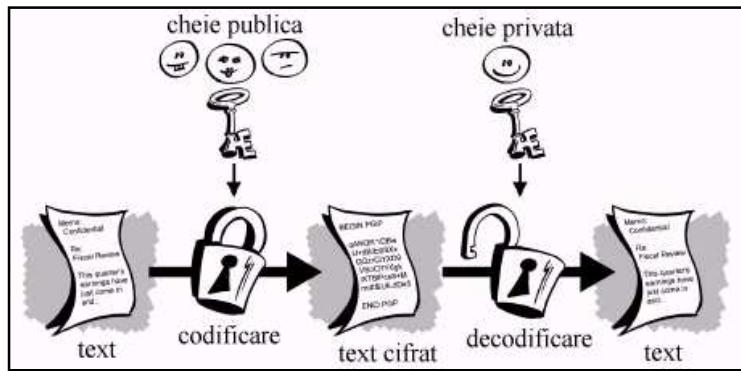
Chiar dacă sunt interceptate, datele vehiculate prin Internet pot fi neutilizabile pentru cel ce le-a preluat fraudulos dacă înaintea trimiterii au fost criptate.

Criptarea are la bază utilizarea *cheilor de criptare*. Acestea sunt secvențe de caractere care, combinate cu textul supus criptării fac imposibilă redarea în clar a acestuia fără folosirea unei chei de decriptare. Sistemele informatice care deservesc sisteme închise (armată ...) folosesc pentru codificare o singură cheie de criptare :



În sistemele deschise se utilizează de obicei o pereche de chei: o cheie publică și o cheie privată (secretă). Cheia publică este utilizată pentru criptare iar cheia privată pentru decriptarea mesajelor primite. În cazul interceptării unui mesaj criptat, sistemul de criptare nu permite găsirea cheii private chiar dacă se cunoaște cheia publică.

Generarea perechii de chei se realizează de către o aplicație specializată și după generare, pentru a garanta autenticitatea celui care transmite datele, se contactează o autoritate de certificare (*Certification Authority, CA*) căreia i se transmite cheia publică. Autoritatea de certificare emite un certificat de securitate (*Security Certificate*) având rolul de a lega cheia publică de proprietarul acesteia. Certificatul este un fișier care identifică o persoană sau o organizație și obținerea lui implică plata unei taxe. Cea mai cunoscută autoritate de certificare este *Verisign* (www.verisign.com).



3.7.8 Semnătura electronică

Pentru a autentifica expeditorul unui mesaj trimis prin e-mail se poate ataşa mesajului o semnătură electronică. Aceasta se obţine în doi paşi: se aplică mesajului un algoritm matematic (algoritm de semnare) prin care din mesaj se obţine un şir unic şi caracteristic de octeţi iar apoi acest şir este criptat folosind cheia privată a expeditorului. Codul obţinut va putea fi prelucrat la destinaţie folosind cheia publică a expeditorului şi comparat apoi cu un nou şir obţinut la destinaţie prin aplicarea unui algoritm asupra mesajului recepţionat. În acest fel se poate stabili dacă mesajul a fost transmis fără a se interveni asupra sa, autentificându-se în acelaşi timp şi expeditorul. Codul ataşat prin metoda descrisă mai sus unui mesaj poartă numele de *semnătură electronică*. În fiecare ţară operează legi care reglementează utilizarea semnăturii electronice.

