



Radu Crețulescu

Daniel Morariu

Dezvoltarea aplicațiilor WEB

Dezvoltarea aplicațiilor WEB

Acest material a fost realizat cadrul proiectului „Burse Universitare în România prin Sprijin European pentru Doctoranzi și Post-doctoranzi (BURSE DOC - POSTDOC)”

Cod Contract: **POSDRU/159/1.5/S/133255**



Editura Universității "Lucian Blaga" din Sibiu
ISBN 978-606-12-0995-8

2015

Cuprins

1	Introducere.....	5
1.1	Să ne lămurim.....	6
1.1.1	La început a fost WEB-ul: fals.....	6
1.1.2	WEB-ul era la sistemul de operare: fals	6
1.1.3	Sistemul de operare era WEB-ul: fals	6
1.2	Când a început WEB-ul?	7
1.3	La început a fost Internetul: aproape corect.....	9
2	HTML – HTML5	14
2.1	WWW (World Wide Web)?.....	14
2.2	Pagini Web	15
2.3	Adresa unei pagini de Web.....	15
2.4	Cum aduce browserul paginile web?	17
2.5	Cum afișează browser-ul paginile web?	19
2.6	Ce este un fișier HTML?	20
2.7	Structura standard a unui document html5	21
2.7.1	Primul cod html	21
2.7.2	Explicarea structurii standard:	24
2.8	Tag-uri html	26
2.8.1	DOM – Document Object Model	27
2.8.2	Observații generale	29

2.9	Marcarea corectă a textului	31
2.9.1	Marcarea metadatelor	32
2.9.2	Delimitarea logică a conținutului	34
2.9.3	Tag-uri pentru marcarea structurală a conținutului textului.....	40
2.9.4	Tabele	44
2.9.5	Tag-uri nestructurale pentru marcarea conținutului textului.....	58
2.9.6	Taguri pentru imagini și multimedia	67
2.9.7	Formulare	79
2.9.8	Taguri depreciate	96
2.10	Culori HTML.....	100
2.11	Poziționarea conținutului unei pagini html...	102
2.12	Marcajul semantic.....	102
3	CSS.....	108
3.1	Ce este CSS?	108
3.2	Versiuni CSS	109
3.3	Declararea stilului inline în html.....	110
3.4	Foaia de stil internă	111
3.5	Foaia de stil externă.....	112
3.5.1	Utilizarea tagului <link>.....	113
3.5.2	Utilizarea instrucțiunii @import.....	114

3.6	Ordinea de cascadata a stilurilor.	115
3.7	Sintaxa CSS	116
3.7.1	Tipuri de selectori	118
3.7.2	<i>Proprietățile CSS pentru background.....</i>	<i>143</i>
3.7.3	Proprietățile CSS pentru text.....	147
3.7.4	Proprietățile CSS pentru fonturi	149
3.7.5	Proprietățile CSS pentru tabele	153
4	PHP	156
4.1	Sintaxa php	157
4.2	Afișarea de text în php.....	159
4.2.1	Afișarea cu ajutorul lui “echo“	159
4.2.2	Elemente de formatare	161
4.2.3	Afișarea elementelor html cu ajutorul lui php 163	
4.2.4	Afișarea caracterelor speciale	164
4.3	Variabile în PHP.....	166
4.3.1	Declararea variabilelor	167
4.3.2	Atribuirea de valori unei variabile.....	167
4.3.3	Șiruri de caractere	167
4.3.4	Tipuri de variabile	169
4.3.5	Array-uri și Array asociativ	174

4.4	Operatori PHP.....	175
4.4.1	Operatori aritmetici.....	175
4.4.2	Operatori de atribuire	176
4.4.3	Operatori de comparare.....	176
4.4.4	Operatori logici	177
4.5	Instrucțiuni php.....	177
4.5.1	Instrucțiuni condiționale.....	177
4.5.2	Instrucțiuni repetitive	180
4.6	Preluarea datelor dintr-un formular	185
5	MySQL.....	187
5.1	Conectarea la MySQL folosind funcții PHP	187
5.2	Comenzi SQL	189
5.2.1	Comanda SELECT.....	189
5.2.2	Comanda INSERT	194
5.2.3	Comanda UPDATE.....	196
5.2.4	Comanda DELETE	197
6	Anexa 1 – Exemplu <div>	199
7	Bibliografie	211

1 Introducere

*La început era Cuvântul și Cuvântul era întors către
Dumnezeu și Dumnezeu era Cuvântul(Ioan 1)*

Pentru o carte care se vrea în esență despre internet și despre web este cel puțin ciudat să folosesc un astfel de motto. Totuși pentru mine *Cuvântul* este important. Și întreb: Cum era reprezentat *Cuvântul*, <16 biți> (Și mai complicat: ce era înainte de cuvânt?). Pentru că „WEB”-ul este un cuvânt am putea încerca să parafrăzăm:

*La început a fost WEB-ul și WEB-ul era la sistemul de operare și
sistemul de operare era WEB-ul.* (1)

Dacă motto-ul inițial poate fi considerat un șir de adevăruri din motiv că nu poate fi demonstrat contrariul sau nu poate fi găsit un contraexemplu, afirmația (1) este un șir de falsuri:

1.1 Să ne lămurim...

1.1.1 La început a fost WEB-ul: fals

Pentru a demonstra anumite afirmații mulți autori se bazează pe un istoric de evenimente, pe istorie. Problema este că anumite istorii se bazează pe istorioare transmise prin viu grai din moși-strămoși și sunt apoi considerate adevăruri incontestabile. În cazul WEB-ului “moși-strămoșii” din fericire sunt cei care acum au venerabila vârstă de 47 -50 de ani și care au fost acolo și au văzut începutul.

1.1.2 WEB-ul era la sistemul de operare: fals

Evident că WEB-ul nu poate fi la sistemul de operare deoarece conform definiției Webul este un sistem hipermedia distribuit. Asta înseamnă că este al tuturor și elementele acestui sistem – fișiere html, poze, sunete, videouri - se găsesc pe diferite servere în toată lumea.

1.1.3 Sistemul de operare era WEB-ul: fals

Deocamdată sistemul de operare nu poate fi WEB-ul, dar cine știe? Dacă ne gândim numai la aplicațiile în CLOUD care deja sunt uzuale și care datorită vitezelor de transmisie în rețea mari sunt rapide poate să apară în curând un sistem de operare direct în WEB. Asta ar fi bine dar și rău. Bine, pentru că nu ar mai fi problema update-urilor, copiilor ilegale, rău pentru că cineva ar fi șef peste tot și ar putea controla accesul la resurse. Și asta ar fi răul cel mai mic...

1.2 Când a început WEB-ul?



Fig. 1. Logo-ul pentru Web

Răspunsul este simplu: la sfârșitul anului 1990 s-a făcut prima propunere pentru un sistem de tip WEB de către Tim Berners Lee și Robert Cailliau în lucrarea „*WorldWideWeb: Proposal for a HyperText Project*” [1]. Totuși ideea lui Tim Berners Lee de a implementa un „sistem de management al informației” prezentată la CERN în Elveția, nu este nouă, ea a fost prezentată tot de Tim Berners Lee și într-o lucrare anterioară „*Information Management: A Proposal*” [2].

Din aceste motive fondatorul World Wide Web-ului (WWW), care astăzi este cel mai important serviciu care utilizează Internet-ul, este considerat cercetătorul britanic Tim Berners-Lee, care în cadrul CERN, la Laboratorul European de Cercetări Nucleare din Geneva, în 1989 a propus o soluție de

dezvoltare pentru un schimb unitar de informații între calculatoare, folosind Internetul. Scopul lui Berners Lee a fost de a crea un sistem în care oamenii de știință din întreaga lume să poată să-și comunice concluziile lor de cercetare într-un mod simplu. Acest lucru a fost posibil pentru început numai în format text.

Sistemul dezvoltat de Tim Berners Lee a pornit de la ideea mașinii *Memex* a lui Vannevar Bush [3] și pe de altă parte de la proiectul *Xanadu* a lui Ted Nelson [4]. În acest proiect, Nelson își imagina o „mașină de literatură”, care să stocheze toată literatura lumii și care era interconectată prin intermediul unor link-uri pe bază de cuvinte. El a inventat termenul "Hypertext" în 1965 și astfel a influențat considerabil activitatea lui Tim Berners-Lee. Acesta a dezvoltat WWW-ul ca un sistem neliniar unde documentele nu mai erau privite a fișiere contigue ci ca obiecte eterogene conectate la nivel logic prin hiperlinkuri.

Tim Berners-Lee nu a patentat niciodată ideile și implementarea tehnică punând WWW la dispoziția întregii lumi.

„Web-ul este mai degrabă o realizare socială decât una tehnică. L-am inventat dintr-un motiv comun de a ajuta oamenii să colaboreze dar nu ca jucărie tehnică” (Tim Berners Lee)

Tim Berners Lee este în prezent președintele World Wide Web Consortium (W3C – <http://w3c.org>), un forum deschis pentru organizații și companii care a însoțit dezvoltarea ulterioară a WWW și este responsabil pentru standardele

existente din WEB.

1.3 La început a fost Internetul: aproape corect

Istoricul Internet-ului

Când ne referim la istorie atunci automat ne gândim la ere și secole. Pentru Internet folosirea în acest context a cuvântului istorie este cam exagerat. Internetul are aproximativ 50 de ani deci nu a ieșit nici măcar la pensie. Succint câteva repere „istorice” despre evoluția internetului [6]:

Utilizarea militară (La început a fost armata americană)

În timpul războiului rece, la sfârșitul anilor 1960, Departamentul american de apărare ca răspuns la lansarea pe orbită a primului satelit artificial rusesc, Sputnik-ul (4.10.1957), a dezvoltat pe o rețea descentralizată de calculatoare, cu scopul de a crea o structură în care datele sunt distribuite astfel încât chiar și după un potențial atac rețeaua să permită o comunicare completă, fără întreruperi între toate bazele militare. Această descentralizare a rețelei este una dintre caracteristicile esențiale ale Internet-ului actual.

1969 - Departamentul de apărare a dezvoltat rețeaua ARPA-NET, care era formată din 4 calculatoare. Un an mai târziu rețeaua a ajuns la 40 de calculatoare interconectate.

Datorită interesului militar pentru descentralizarea rețelei a fost permis accesul și instituțiilor științifice.

Utilizarea științifică (Și apoi știința)

La începutul anilor 1970 ARPA NET-ul a fost folosit de către principalele instituții științifice din Statele Unite.

Oameni de știință, foloseau rețeaua în special pentru a schimba fișiere și mai puțin în realizarea unei descentralizări a conținutului prin seturi de date identice pe fiecare server.

1971 – Ray Tomlinson trimite primul mail prin rețeaua ARPANET

1974 - Protocolul TCP/IP

Bob Kahn și Vinton Cerf publică „A Protocol for Packet Network Intercommunication” [5] în care se prezintă pentru prima dată protocolul TCP. La rețea sunt conectate primele servere din afara SUA (Hawaii, Norvegia, Marea Britanie) Interconectarea unor sisteme informatice diferite a dus la elaborarea unui protocol de comunicații care pute fi folosit de orice sistem și astfel în 1978 s-a lansat protocolul de rețea TCP/IP utilizat și în zilele noastre.

1980 – ARPANET-ul se prăbușește datorită unui virus

1984 – MILNET - Datorită utilizării tot mai intense a ARPA-NET-ului de către instituțiile civile, instituțiile militare s-au văzut nevoite să-și creeze o rețea proprie MILNET.

1979 DARPANET - INTERNET

Militarii s-au retras din ARPA-NET și au lăsat rețeaua pentru uz civil. National Science Foundation (NSF) a creat un sistem de conectare a tuturor centrelor de cercetare importante. Rețelele de calculatoare mici sau calculatoare individuale ale unor instituții mai mici s-au putut conecta astfel la ARPA-NET. Astfel, a apărut treptat rețeaua de rețele. Acest lucru a evoluat atât de rapid astfel încât după 1980 se vorbește despre Internet ca fiind rețeaua rețelelor.

1984

Se introduce Domain Name System (DNS)

1989

Numărul host-urilor trece de 100,000

1990

ARPANET-nu mai există

Apare World Wide Web (WWW)

Tim Berners Lee lansează ideea unui browser și a unui limbaj care să simplifice schimbul de informații. Astfel pune bazele World Wide Web-ului care a devenit un serviciu printre multe altele care utilizează rețeaua Internet.

S-a creat consorțiul W3 ca organism independent, al cărui scop a fost și este stabilirea și menținerea standardelor în Web



World Wide Web

The WorldWideWeb (W3) is a wide-area [hypermedia](#) information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an [executive summary](#) of the project, [Mailing lists](#), [Policy](#),

[What's out there?](#)

Pointers to the world's online information, [subjects](#), [W3 servers](#), etc.

[Help](#)

on the browser you are using

[Software Products](#)

A list of W3 project components and their current state. (e.g. [Line Mode](#), [X11 Viola](#), [NeXTStep](#), [Servers](#), [Tools](#), [Mail robot](#), [Library](#))

[Technical](#)

Details of protocols, formats, program internals etc

[Bibliography](#)

Paper documentation on W3 and references.

[People](#)

A list of some people involved in the project.

[History](#)

A summary of the history of the project.

[How can I help?](#)

If you would like to support the web..

[Getting code](#)

Getting the code by [anonymous FTP](#), etc.

**Fig. 2. Prima pagină de web preluată de la
<http://info.cern.ch/hypertext/WWW/TheProject.html>**

1992

România se conectează la nodul din Viena prin ICI (Institutul Național de Cercetare-Dezvoltare în Informatică). În mai 1992, ICI începe, sub coordonarea CNI (Consiliul Național de Integritate), demersurile la RIPE (Regional Internet Registries) și apoi la IANA (Internet Assigned Numbers Authority) pentru recunoașterea și înregistrarea top level domain-ului ".ro". Primul site românesc a fost: ici.ro

1993

Web-ul cum îl cunoaștem noi astăzi a devenit popular deoarece în acel an Marc Andreessen, co-fondatorul firmei Netscape a implementat primul browser web – Mosaic -care funcționa în mod grafic. A fost primul browser care putea afișa

text și imagini în aceeași fereastră windows. Atunci datorită monitoarelor care puteau folosi doar 8 biți de culoare au putut fi folosite doar 216 culori – celebrele culori *web safe*.

Azi

În ziua de azi ne este foarte greu să concepem că am putea trăi fără să ne conectăm la "net". Zilele trecute m-am trezit că fiica mea de 10 ani nu mai poate să comunice cu prietenii ei din cauză că accesul la aplicația "WhatsApp" instalată pe telefonul mobil nu mai era posibil. Un părinte deosebit de rău și neînțelegător a setat routerul să oprească pentru o anumită adresă de mac accesul la rețeaua „internet” din cauză de depășire a orei 21.30.

2 HTML – HTML5

2.1 WWW (World Wide Web)?

World Wide Web-ul sau pe scurt WWW este un serviciu care folosește rețeaua INTERNET. Dacă dorim să definim acest serviciu atunci o posibilă definiție ar putea fi următoarea:

WWW este un sistem hipermedia distribuit care folosește rețeaua INTERNET. WEB-ul reprezintă totalitatea site-urilor/ documentelor și informațiilor de tip hipertext, legate între ele, care pot fi accesate prin rețeaua mondială de internet.

De ce distribuit?

Pentru că informațiile se găsesc în paginile web care sunt stocate pe diferite servere din lumea întreagă.

De ce hipermedia?

Pentru că informațiile sunt de tip text, imagini, sunete, filme (poate în curând și mirosuri) și se pot întâlni simultan toate pe o pagină de web.

2.2 Pagini Web

Paginile web nu sunt altceva decât niște fișiere text care conțin anumite marcate (numite tag-uri) și pe care browser-ele le recunosc și vor afișa conținutul fișierului text conform cu aceste marcate. Fișierele text sunt salvate pe anumite servere (calculatoare speciale) care au instalate servicii de web. Serviciile web nu sunt altceva decât aplicații care răspund la anumite cereri venite din internet pe anumite porturi (adrese) ale calculatorului server. De obicei pentru un serviciu web se folosește în mod tradițional portul 80 sau 8080. Cele mai cunoscute aplicații pentru servere web sunt: Apache, Tomcat, Microsoft Internet Information Services (IIS), Nginx, lighttpd, Jigsaw (W3C's Server), Klone, Abyss, Oracle Web Tier, X5 (Xitami) etc. În comunicarea cu serverul se utilizează un protocolul de comunicație numit **HTTP (HyperText Transfer Protocol)**.

2.3 Adresa unei pagini de Web

Pentru a accesa o pagină de web avem nevoie de o adresă a paginii pe care o specificăm prin intermediul unui browser (aplicație care rulează pe calculatorul utilizatorului). Structura unei adrese este de forma:

protocol://**host**/**cale**/**nume_fișier**

Ex.:

http://inginerie.ulbsibiu.ro/orar/afis_no_print.php

Adresa globală a unei pagini de web se mai numește și URL care este prescurtarea de la Uniform Resource Locator. Prima parte este denumită identificator de protocol (protocolul care este folosit pentru a accesa o resursă) iar partea a doua se numește numele resursei și specifică o adresă de IP (Internet Protocol) sau numele de domeniu unde se găsește respectiva resursă. Denumirea protocolului este delimitat de numele resursei prin simbolul „:/"

De exemplu cele două URL-uri de mai jos pointează spre două fișiere diferite din domeniul inginerie.ulbsibiu.ro. Primul specifică faptul că se va utiliza protocolul FTP (File Transfer Protocol – se va aduce local fisierul specificat) iar al doilea specifică o pagină web care va fi afișată în browser și care pentru a o accesa se va folosi protocolul HTTP:

<ftp://inginerie.ulbsibiu.ro/note.xls>

<http://inginerie.ulbsibiu.ro/index.htm>

Un URL este un tip de URI (Uniform Resource Identifier)[7] care este un termen generic pentru toate tipurile de nume și adrese care se referă la obiecte din World Wide Web.

Termenul „adresă web” este sinonim cu un URL care folosește protocolul HTTP sau HTTPS (**HyperText Transfer Protocol/Secure**).

2.4 Cum aduce browserul paginile web?

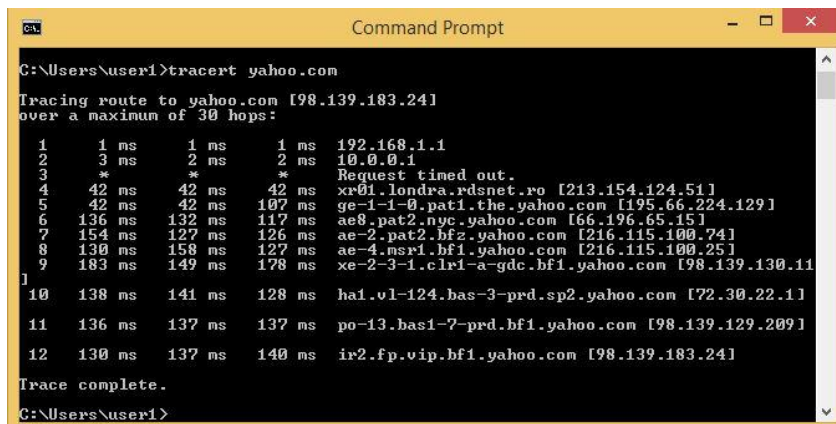
Web-ul fiind un serviciu care utilizează rețeaua INTERNET trebuie să-i respecte regulile de adresare. În internet pentru a realiza comunicarea dintre calculatoarele utilizatorilor și servere sau între servere fiecare calculator are o adresă unică care este reprezentată de un număr. Fiind utilizat în protocolul TCP/IP de comunicare adresa se numește adresă de IP (Internet Protocol). Adresele de IP sunt împărțite în mai multe clase. Nu vom intra în detalierea adreselor de IP. Standardul actual în adresarea IPv4 este un număr format din 4 grupuri de numere fiecare grup conținând numere cuprinse între 0 și 255.

Exemplu: 98.139.183.24 este adresa de IP a lui *yahoo.com*. Dacă introducem în browser această adresă browserul va deschide pagina yahoo.com. Evident că în timp această adresă poate să fie schimbată.

Pentru a afla adresa de IP a unui domeniu se poate executa din line de comandă instrucțiunea *tracert nume_domeniu* iar ultima adresă de IP afișată va fi chiar adresa de IP a serverului care deservește acel domeniu, ultima adresă scrisă este adresa serverului căutat.

Teoretic pentru a accesa un server avem nevoie de adresa sa de IP. Adresele de IP sunt greu de reținut iar o navigare bazată pe adrese de IP ar face viața utilizatorilor de web foarte grea. Din fericire există un sistem de nume numit DNS (Domain Name

System) care are la bază niște tabele în care se memorează adresele de IP și numele de domeniu asociate acestor adrese. Astfel putem naviga foarte ușor utilizând doar nume de domeniu.



```
C:\Users\user1>tracert yahoo.com

Tracing route to yahoo.com [98.139.183.241]
over a maximum of 30 hops:

  1  1 ms    1 ms    1 ms    192.168.1.1
  2  3 ms    2 ms    2 ms    10.0.0.1
  3  *        *        *        Request timed out.
  4  42 ms   42 ms   42 ms   xro1.londra.rdsnet.ro [213.154.124.51]
  5  42 ms   42 ms  107 ms   ge-1-1-0.pat1.the.yahoo.com [195.66.224.129]
  6  136 ms  132 ms  117 ms   ae8.pat2.nyc.yahoo.com [66.196.65.15]
  7  154 ms  127 ms  126 ms   ae-2.pat2.bf2.yahoo.com [216.115.100.74]
  8  130 ms  158 ms  127 ms   ae-4.msr1.bf1.yahoo.com [216.115.100.25]
  9  183 ms  149 ms  178 ms   xe-2-3-1.clr1-a-gdc.bf1.yahoo.com [98.139.130.11]
 10  138 ms  141 ms  128 ms   ha1.v1-124.bas-3-prd.sp2.yahoo.com [72.30.22.1]
 11  136 ms  137 ms  137 ms   po-13.bas1-7-prd.bf1.yahoo.com [98.139.129.209]
 12  130 ms  137 ms  140 ms   ir2.fp.vip.bf1.yahoo.com [98.139.183.241]

Trace complete.
C:\Users\user1>
```

Fig. 3. Tracert la yahoo.com

Practic pentru a accesa adresa unei pagini de WEB browserul accesează cu adresa de web serverul DNS și va obține o adresă de IP iar apoi browserul face cererea către adresa de IP respectivă. Dacă la adresa de IP se găsește un server WEB atunci acesta îi va trimite browserului resursa solicitată.

Schema de funcționare este prezentată în fig. 4.

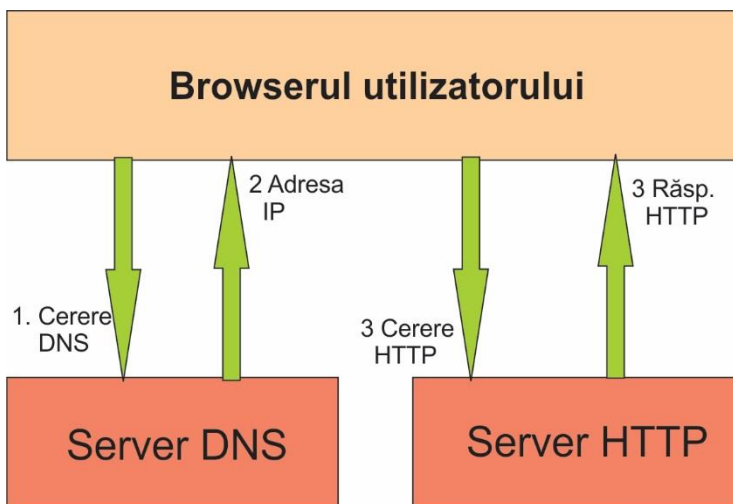


Fig. 4. Pașii pentru accesarea unei pagini web

2.5 Cum afișează browser-ul paginile web?

Toate paginile web sunt fișiere text care conțin **marcatori sau tag-uri (pentru afișaj)**. Acești marcatori reprezintă pentru browser comenzi de afișaj și care nu trebuie compilate. În cazul în care apare un tag pe care browserul nu-l recunoaște, acesta va fi ignorat de către browser. De obicei, tagurile sunt perechi de nume încadrate între caracterele „<>” și delimitează o bucată de text.

Exemplu pentru un tag:

<p>Acesta este un paragraf </p>.

Perechea `<p>` `</p>` reprezintă o pereche de tag-uri din limbajul html. Regulile complete pentru sintaxa html5 sunt specificate în [8].

2.6 Ce este un fișier HTML?

HTML este prescurtarea pentru **H**yper **T**ext **M**arkup **L**anguage. Un fișier HTML este un fișier text care conține **tag-uri**. Pentru a putea fi afișat corect de către browser fișierul trebuie să aibă extensia *htm* sau *html*. Un fișier html poate fi editat în orice editor simplu de text cum ar fi „Notepad” sau „Notepad++”. Bineînțeles există și variante comerciale de editoare de tipul WYSIWYG (What You See Is What You Get) care simplifică mult modul de scriere a acestor fișiere, dintre acestea amintim : Adobe Dreamweaver, Adobe Muse, Amaya, BlueGriffon, CKEditor, EZGenerator, FirstPage, Freeway, Froala Editor, Google Web Designer, Hyper Publish, Jimdo, KompoZer, Macaw Web Editor, Maqetta, Microsoft Expression Web, Microsoft SharePoint Designer, Microsoft Visual Studio, Microsoft Visual Web Developer Express, Microsoft Publisher, Microsoft WebMatrix, NetObjects Fusion, OpenBEXI, openElement, Opera Dragonfly, Quanta Plus, RapidWeaver, Sandvox, SeaMonkey Composer, Silex website builder, SiteGrinder, SnapEditor, Sparkle - Pro Visual Web Design, TinyMCE, tkWWW, TOWeb, UltraEdit, Webflow, WebPlus, ș.a.

De asemenea multe alte editoare de text au posibilitatea de a salva conținutul în format html. Totuși nu recomandăm folosirea acelor editoare deoarece fișierele rezultate au

dimensiuni mult mai mari față de documente realizate clasic.

2.7 Structura standard a unui document html5

Standardizarea în web este tot mai importantă și va deveni obligatorie dacă ne dorim saltul calitativ spre noul web 3.0 – numit și web-ul semantic, un web în care calculatoarele „înțeleg” conținutul paginilor web pe care le afișează și structura internetului este altfel organizată. Deocamdată standardul obligatoriu este la nivelul web 2.0 – web-ul în care utilizatorii pot interacționa cu paginile web putând să modifice conținutul acestora. Un exemplu clasic de pagina web 2.0 este *facebook.com*.

HTML5 este un standard care a fost lansat la 28.10.2014 de către W3C [9]. Deocamdată acest standard nu este obligatoriu să fie respectat de toate paginile web. Se lucrează deja la standardele HTML5.1 și HTML5.2

2.7.1 Primul cod html

Putem porni fără teama la scrierea fișierelor html pentru că niciodată nu veți vedea fereastra urât colorată a compilatorului care te anunță că ai ceva erori și chiar mai multe „warning-uri” umbrită de privirea dojenitoare a profesorului care îți dă de înțeles că iarăși ai uitat „un punct și virgulă”. Aici dacă există greseli în cod browserul nu va executa codul

respectiv, pur și simplu fără să îți spună exact unde este greșala.

În continuare vom prezenta comparativ structura standard a unui document în standardul XHTML1.0 și în noul standard HTML5.

Standardul XHTML 1.0

```
<!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">

<head>
  <title>Titlul paginii afisat pe fereastra
  browserului</title>
</head>
<body>
  <p> Buna ziua lume!</p>
</body>
</html>
```

Standardul HTML5

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title> Titlul paginii afisat pe
```

```
fereastra browserului </title>
</head>
<body>
    <p> Buna ziua lume!</p>
</body>
</html>
```

Structura unui document html conține două părți importante:

1. Zona delimitată de tag-urile `<head>` `</head>`. Aici se pun informații despre document și care nu vor fi afișate în fereastra browser-ului. De exemplu titlul documentului este delimitat de tagurile `<title>` `</title>` acest text fiind afișat **pe** fereastra browser-ului, ca fiind numele paginii accesate. Tot aici se pun informații legate de setul de caractere utilizat, linkuri către fișiere de stil și fișiere javascript (despre care vom vorbi puțin mai târziu) în general numite meta-informații.

2. Zona delimitată de tag-urile `<body>` `</body>` . Aici se pun informații care se vor afișa în fereastra browser-ului conform cu tag-urile existente.

Se observă o evidentă simplificare a structurii în standardul HTML5 față de standardul XHTML 1.0

ATENȚIE: Nu lăsați un spațiu după caracterul „<” pentru că browser-ul nu va mai recunoaște tag-ul.

Dacă salvăm structura HTML5 prezentată mai sus într-un fișier cu numele "hello.htm" browser-ul ar trebui să afișeze:

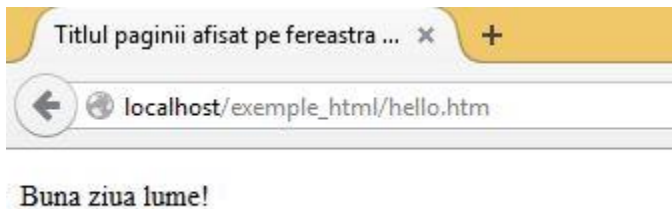


Fig. 5. Structura standard afișată în browser

2.7.2 Explicarea structurii standard:

Înainte de a începe observăm că avem mai multe **perechi** de tag-uri. Primul marchează începutul unui bloc de text iar tag-ul care începe cu „</” și are același nume cu tag-ul de deschidere marchează terminarea blocului de text.

Primul tag dintr-un document HTML este un comentariu.

```
<!DOCTYPE html>
```

Acest comentariu este important pentru standardizarea documentelor html. Pentru web-ul semantic, la care se lucrează momentan nefiind descris în totalitate, este deosebit de importantă respectarea standardelor. Motoarele de căutare inteligente vor „ști” semnificația tag-urilor iar astfel rezultatele căutărilor vor fi mult mai precise. Comentariul din exemplul de mai sus specifică faptul că documentul html care urmează este scris conform standardului HTML5.

Tag-ul `<html>`. Acesta are rolul de a anunța browser-ul

că urmează un document HTML. La sfârșitul documentului avem tag-ul de închidere `</html>` acesta atenționând browser-ul ca a ajuns la sfârșitul documentului HTML.

Exemplu 2:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title> Titlul paginii afisat pe fereastra
      browserului </title>
  </head>
  <body>
    <h1> Bună ziua lume!</h1>
    <p>Acesta e un paragraf.</p>
    <p>In limba engleză un paragraf se
      marcheaza printr-un rând liber lăsat în
      față.</p>
    <p>Acesta e un alt paragraf.</p>

  </body>
</html>
```

Browser-ul va afișa:

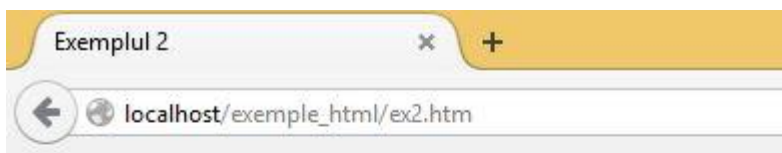


Fig. 6. Exemplul 2

În exemplele de cod html ce vor urma nu se va reproduce în totalitate structura standard a unui document html 5 fără să mai scriem explicit acest lucru

2.8 Tag-uri html

Pentru a marca corect un document HTML trebuie să respectăm ideea separării formei de conținut. Această separare este foarte importantă deoarece motoarele de căutare (roboții de căutare automată ai acestora numiți și agenți de căutare) vor indexa mai corect și mai repede documente care conțin tag-uri de delimitare structurală. Un robot de căutare nu va fi impresionat dacă textul este scris cu roz pe fundal verde și nu va

ține cont de acest aspect în stabilirea „page ranking-ului” pentru o anumită pagină. Page rank-ul este o modalitatea de a specifica importanța unei pagini, modalitate care este folosită actualmente de motoarele de căutare pentru a ordona paginile rezultate.

2.8.1 DOM – Document Object Model

Comparativ cu textul simplu dintr-un document, o pagină web conține pe lângă text și imagini și o structură. Structura de bază a unei pagini web este dată de DOM (Document Object Model). DOM este o interfață independentă de platformă și limbaj, care permite programelor și script-urilor să acceseze dinamic și să actualizeze conținutul, structura și stilul documentului web [10].

Structura DOM a unei pagini web este o structură arborescentă, în care fiecare tag (etichetă) HTML în pagina corespunde unui nod în arborele DOM. Pagina web poate fi delimitată (marcată) de tag-uri predefinite structural. Tag-uri utile ar fi <p> (paragraf), <table> (tabel), (element de lista), <h1>, ..., <h6> (titlu sau heading-uri) etc.

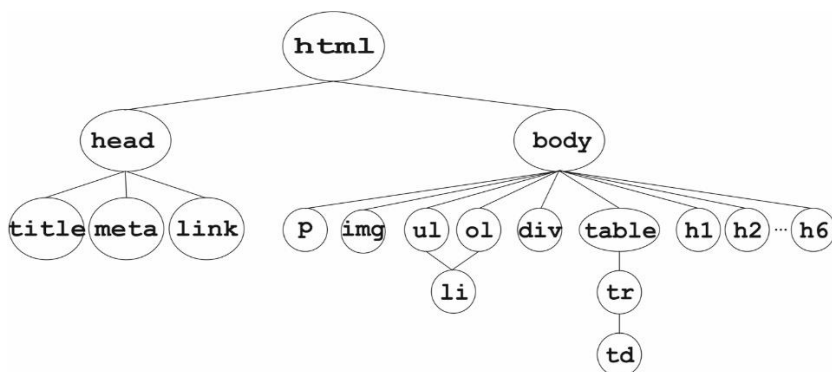


Fig. 7. Structura și taguri conținute într-o pagină web

Astfel, structura DOM poate fi folosită pentru a facilita extragerea de informații. Din păcate, datorită flexibilității de sintaxă HTML, multe pagini web nu respectă specificațiile W3C privind structura HTML. Cel mai „nociv” standard utilizat în web a fost standardul HTML 3.2 care a permis amestecarea „forme” cu cea a „conținutului” fapt ce a dus la îngreunarea indexării paginilor web. În prezent, standardul XHTML₁ și mai noul HTML₅ încearcă o separare clară a conținutului fișierului de formatarea pentru afișare prin utilizarea unor stiluri declarate în antetele paginilor sau în fișiere separate cu ajutorul sintaxei CSS (Cascade Style Sheets). Mai mult decât atât, arborele DOM a fost inițial introdus pentru afișarea documentului în browser și nu ca și o descriere a structurii paginii Web. De exemplu, chiar dacă două noduri în arbore DOM au același părinte, cele două noduri ar putea conține informații care le-ar lega de alte noduri, mai mult decât de nodul părinte.

2.8.2 Observații generale

Elementele HTML:

Elementele html sunt cuvinte care încep cu caracterul < și se încheie cu caracterul >. Este important să nu se insereze caracterul spațiu în cadrul unui tag. Cum s-a putut observa în secțiunile anterioare tagurile sunt în pereche, ele delimitând un document text astfel ca un browser să îl poată afișa.

Exemplu:

Perechea de taguri

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

delimitează un document html.

Recomandare:

W3C recomanda utilizarea literelor mici în scrierea tag-urilor, aceasta fiind recomandarea pentru standardul HTML 4. Pentru cei ce se pregătesc să scrie tag-uri conform standardului XHTML 1.0 și HTML5 utilizarea literelor mici este obligatorie.

Atributele tag-urilor

Tag-urile pot primi atribute. Acestea sunt cuvinte care se despart prin caracterul spațiu de numele tagului urmate de caracterul „=” și o valoare pentru acel atribut. În standardul XHTML1.0 s-a acordat o mare atenție valorii atributului care trebuia obligatoriu scrisă între ghilimele. În cazul standardului

HTML5 s-a renunțat la această cerință dacă valoarea este un singur șir de caractere. Dacă valoarea este compusă din mai multe șiruri de caractere atunci valoarea atributului se specifică obligatoriu utilizând ghilimelele.

```
<nume_tag atr=ceva> bla, bla </nume_tag>  
<nume_tag atr="ceva ceva"> bla, bla </nume_tag>
```

În exemplul de mai sus `atr` este atributul tag-ului `nume_tag`. Aceste atribute pot aduce informații suplimentare despre elementele HTML. Un tag HTML poate să primească mai multe atribute aceste fiind separate între ele prin caracterul spațiu.

Exemplu care nu se va folosi:

```
<body bgcolor="red">
```

În exemplul de mai sus tag-ul `<body>` a primit atributul `"bgcolor"` adică se specifică browser-ului că fundalul paginii este roșu. Deoarece în HTML5 se acordă o importanță deosebită separării formei de conținut aceste atribute de formatare nu se mai folosesc. Formatare unui document HTML5 se va face utilizând sintaxa CSS care va fi prezentată în capitolul 3.

La noi valoarea atributului **bgcolor** este **red**.

Observație:

Întotdeauna atributul pentru tag se adaugă doar tag-ului

de deschidere. Tagul de închidere nu conține atribute Adică:

```
<body bgcolor="red">. Bla.. Bla</body>
```

Comentarii în HTML

Pentru a insera un comentariu într-un fișier html se folosește tag-ul „<!---->”

```
<!--Aici punem textul care comentează ceva și nu va  
fi afișat în fereastra browser-ului-->
```

Observație:

Important este doar semnul exclamării și este necesar doar după deschiderea parantezei unghiulare! Comentariul nu este afișat de către browser, dar câteodată este bine să-l folosim pentru lizibilitatea codului.

Vizualizarea sursei unui document HTML

Acest lucru se poate face în doua moduri:

1. Din meniul browser-ului utilizând meniul **View -> Source**
2. Cu click dreapta în fereastra browser-ului și apoi **View page Source**

2.9 Marcarea corectă a textului

Dacă în standardul HTML3.2 în care s-au amestecat tag-urile pentru delimitarea structurală cu tag-urile care aranjau textul, în standardul HTML5 se acordă o atenție sporită marării

textului conform cu conținutul său. Adică dacă într-o pagina expresia „Capitolul 1” este titlu atunci ea va fi marcată utilizând marcasele `<h1>` `</h1>` pentru că aceste tag-uri delimitează un titlu. Mulți începători folosesc această pereche de tag-uri pentru a scrie un text cu caractere mai mari. În general pentru formatarea unui text se utilizează sintaxa CSS care va fi prezentată în capitolul următor.

În continuare vom prezenta tag-urile de bază pentru a delimita structural un document html precum și noile taguri HTML5 care introduc mai multe restricții în marcarea textului. Vom prezenta sintaxa la taguri pentru HTML5 deoarece este mai restrictivă și se urmărește în domeniu impunerea acestui standard.

2.9.1 Marcarea metadatelor

Vom prezenta tabelar tagurile utilizate în HTML5 și descrise în [10] pentru marcarea metadatelor și care se utilizează între tag-urile de `<head>` `</head>`

Tag	Descriere
<code><head></code>	Delimitează zona unui document html destinată metadatelor. Atenție însă dacă se pune în această zonă un tag de afișare cum ar fi <code><p></code> <code></p></code> browser-ul îl va afișa dar acest lucru nu respectă standardul. Exemplu <pre><head> <title>Titlul documentului</title> </head></pre>

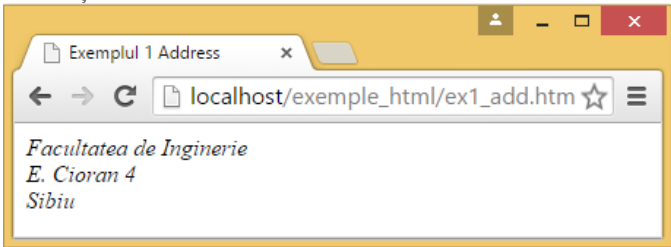
Tag	Descriere
<base>	Specifică URL-ul de bază pentru toate URL-urile relative cuprinse într-un document. Poate exista numai un singur element <base> într-un document. Exemplu: <pre><base href="http://www.csac.ro/test.html"></pre>
<link>	Specifică legătura dintre documentul curent și o resursă externă. Acest element este cel mai utilizat pentru a include foi de stil CSS în documentul html curent. Exemplu: <pre><link href="stil.css" rel="stylesheet"></pre>
<meta>	Descrie informații care nu pot fi reprezentate de către unul dintre tag-urile <base>, <link>, <script>, <style> sau <title> Exemplu: <pre><meta charset="utf-8"> <!-- Redirecționează browserul după 5 secunde--> <meta http-equiv="refresh" content="5;url=http://csac.ulbsibiu.ro/"></pre>
<style>	Conține informații de stil pentru un document, sau o parte a documentului. În mod implicit instrucțiunile de stil sunt scrise utilizând sintaxa CSS.

Tag	Descriere
	Exemplu: <pre><style type="text/css"> body { color:#00FF00; } </style></pre>
<script>	Delimitează scripturi care se execută la nivel de browser. Cele mai des utilizate scripturi sunt Javascript și VisualBasicScript. <pre><script src="nume_fisier.js"></script></pre>
<title>	Definește titlul documentului, afișat în bara de titlu a browser-ului. Acesta poate să conțină numai text. Exemplu: <pre><title>Titlul documentului fara alte taguri</title></pre>

2.9.2 Delimitarea logică a conținutului

Această categorie de tag-uri se utilizează pentru organizarea conținutului paginii permițând să separați conținutul documentului în bucăți logice. Se pot crea un antet, o zonă pentru meniul de navigare, un subsol al paginii, etc.

Tag	Descriere
<body>	Delimitează conținutul unui document HTML. Poate exista numai un singur element <body>într-

Tag	Descriere
	un document. Textul din body este afișat în fereastra browser-ului. Doar în această secțiune se pune conținut destinat afișării
<address>	Se utilizează pentru a marca o adresă de contact pentru un <article>sau <body>părinte; Exemplu: <pre><address> Facultatea de Inginerie
 E. Cioran 4
 Sibiu
 </address></pre> Va afișa:  The screenshot shows a web browser window with the title 'Exemplul 1 Address'. The address bar shows 'localhost/exemple_html/ex1_add.htm'. The rendered content is: <i>Facultatea de Inginerie</i>, <i>E. Cioran 4</i>, <i>Sibiu</i>. Fig. 8. Adresă
<article>	Acest tag este nou apărut în HTML5 și delimitează o zonă de text independent care ar putea fi de exemplu un post de forum, un articol de ziar, o intrare de blog, sau orice alt element independent de conținut. Fiecare tag <article>ar trebui să conțină un element de titlu marcat cu tag-uri h1-h6. Exemplu: <pre><article></pre>

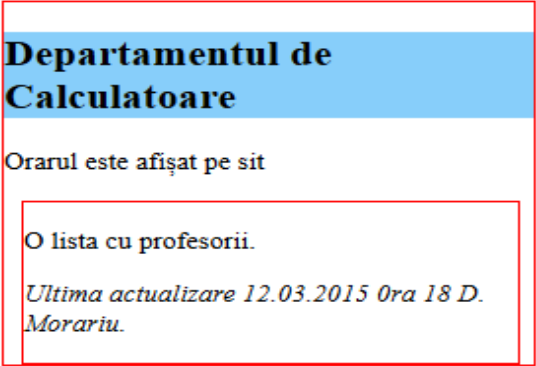
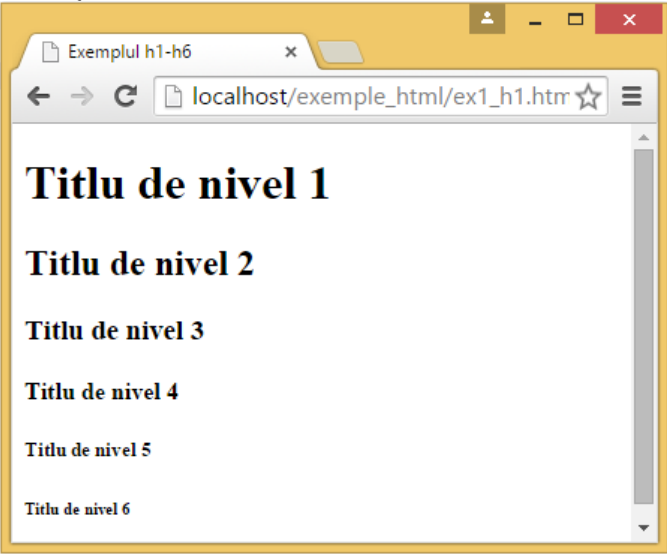
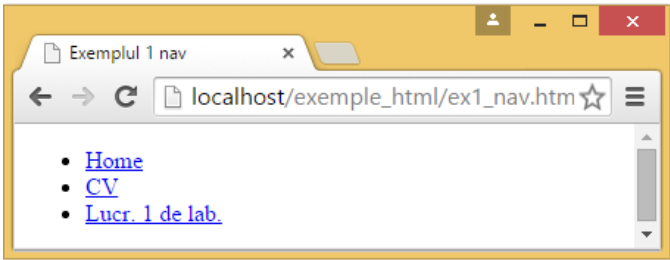
Tag	Descriere
	<pre> <header> <h2>Departamentul de Calculatoare</h2> </header> <section class="sef"> <p>Orarul este afișat pe site</p> </section> <section class="personal"> <article class="didactic"> <p>O lista cu profesorii.</p> <footer> <p>Ultima actualizare <time datetime="2015-03-12 18:00">12.03.2015 Ora 18</time> D. Morariu. </p> </footer> </article> </section> </pre>  The screenshot displays the rendered HTML code. It features a blue header bar with the text "Departamentul de Calculatoare". Below this, a section titled "Orarul este afișat pe sit" is shown. Inside this section, there is a box containing the text "O lista cu profesorii." and a footer that reads "Ultima actualizare 12.03.2015 Ora 18 D. Morariu.".

Fig. 9. Output utilizând tagul <article>din HTML5

Tag	Descriere
<footer>	Acest tag este nou apărut în HTML5 și delimitează o zonă care reprezintă un text scris în partea de jos a unei pagini . Acesta conține de obicei informații despre autor, copyright, dată. Exemplu: <pre><footer> <p>Ultima actualizare <time datetime="2015-03-12 18:00">12.03.2015 Ora 18</time> D. Morariu. </p> </footer></pre>
<header>	Acest tag este nou apărut în HTML5 și delimitează un grup de informații introductive sau de navigare. Poate conține informații cum ar fi un logo, un formular de căutare, etc. Exemplu: <pre><header> o poză cu logoul firmei </header></pre>
<h1>, <h2>, <h3>, <h4>, <h5>, <h6>	„Titlurile” dintr-un document html sunt definite folosind tag-urile <h1> până la <h6>,. <h1> definește cel mai important titlu iar <h6> definește cel mai puțin important titlu. Exemplu: <pre><h1>Titlu de nivel 1</h1> <h2>Titlu de nivel 2</h2> <h3>Titlu de nivel 3</h3></pre>

Tag	Descriere
	<pre> <h4>Titlu de nivel 4</h4> <h5>Titlu de nivel 5</h5> <h6>Titlu de nivel 6</h6> </pre> Va afișa:  Fig. 10. h1-h6
<hgroup>	Acest tag este nou apărut în HTML5 și definește un titlu unic pentru secțiunea căreia îi aparține. Exemplu: <pre> <hgroup> <h1>Titlul principal</h1> <h2>Titlul secundar</h2> </hgroup> </pre>

Tag	Descriere
	Titlul principal Titlul secundar
<nav>	Acest tag este nou apărut în HTML5 și definește o secțiune a paginii care conține link-uri către alte pagini, un meniu de navigare pentru un site. Exemplu: <pre> <nav> Home CV Lucr. 1 de lab. </nav> </pre>  The screenshot shows a web browser window with the title 'Exemplul 1 nav'. The address bar shows 'localhost/exemple_html/ex1_nav.htm'. The main content area displays a bulleted list of three links: 'Home', 'CV', and 'Lucr. 1 de lab.', all underlined and in blue text. Fig. 11. Listă neordonată
<section>	Acest tag este nou apărut în HTML5 și reprezintă o secțiune generică a unui document

2.9.3 Tag-uri pentru marcarea structurală a conținutului textului

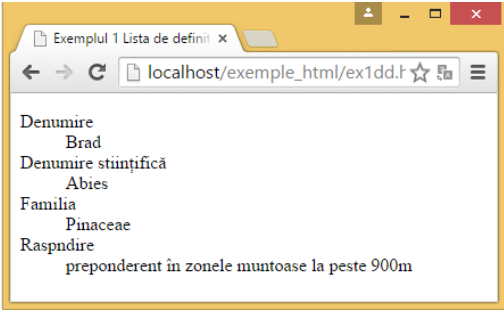
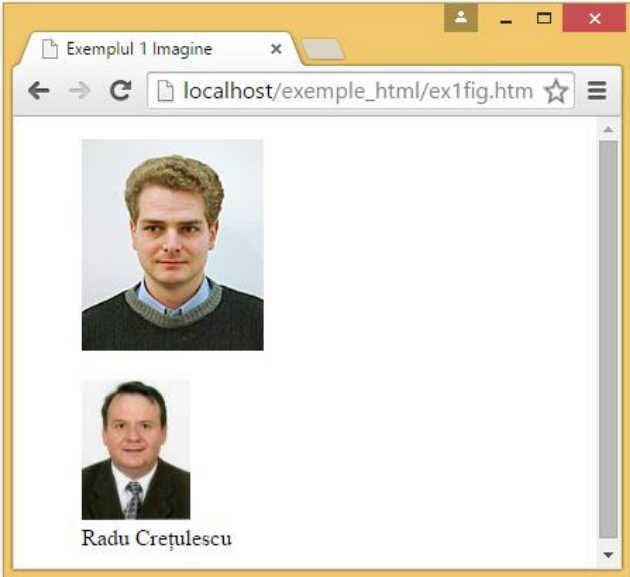
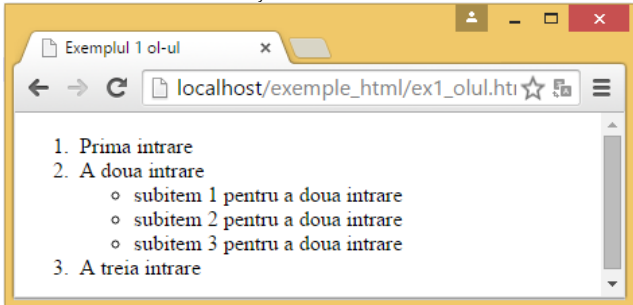
Tag	Descriere
<dl>	Delimitează o listă de definiții și este o lista de termeni și explicarea acestora. O lista de definiții începe cu tag-ul <dl>. Fiecare definiție a unei liste de definiții începe cu tag-ul <dt> care conține caracteristica urmat de tag-ul <dd> care conține definiția. Exemplu: <pre><dl> <dt>Denumire</dt> <dd>Brad</dd> <dt>Denumire stiintifica</dt> <dd>Abies</dd> <dt>Familia</dt> <dd>Pinaceae</dd> <dt>Raspndire</dt> <dd>preponderent în zonele muntoase la peste 900m</dd> </dl></pre>  The screenshot shows a web browser window with the title 'Exemplul 1 Lista de definitii'. The address bar shows 'localhost/example_html/ex1dd.t'. The page content displays the rendered HTML: 'Denumire' followed by 'Brad' on the next line, 'Denumire stiintifica' followed by 'Abies' on the next line, 'Familia' followed by 'Pinaceae' on the next line, and 'Raspndire' followed by 'preponderent în zonele muntoase la peste 900m' on the next line.

Fig. 12. Listă de definiții

Tag	Descriere
<dt>	Delimitează un termen dintr-o listă de definiții specificată prin<dl> Elementul <dt> poate fi doar copil al lui <dl> urmat de obicei de elementul <dd>.
<dd>	Delimitează descrierea unui termen dintr-o listă de definiții <dl> Acest element poate fi doar copil al lui <dt>
<div>	Acest tag este un container generic pentru părți de conținut. De obicei se utilizează dacă nu se folosesc alte tag-uri <header> sau <nav> Exemplu: <pre><div> <p>Un paragraf</p> </div></pre>
<figure>	De obicei acest element delimitează o figură care poate să fie o imagine, o poză etc. Poate să conțină și tagul <figcaption> Exemplu: <pre><figure> </figure></pre> <pre><!--Imagine cu figcaption --></pre>

Tag	Descriere
	<pre data-bbox="337 268 852 483"><figure> <figcaption>Radu Crețulescu </figcaption> </figure></pre>  Fig. 13. <code>img</code>, <code>figcaption</code> figure
<figcaption>	Acest tag este nou apărut în HTML5 și delimitează o legendă asociată unei ilustrații și are ca părinte tag-ul <figure>. Tag-ul <figcaption> poate fi primul sau ultimul element în interiorul unui bloc de tip <figure>. De

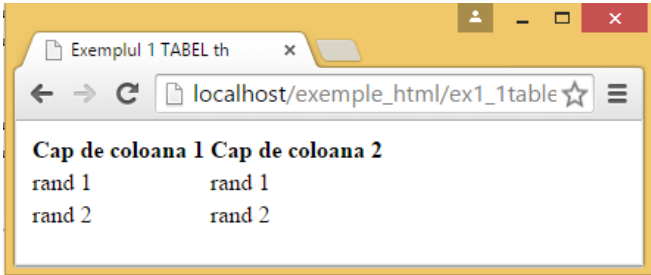
Tag	Descriere
	asemenea, elementul <figcaption> este opțional;
<main>	Elementul <main> poate fi utiliza pentru a delimita conținutul principal al unei pagini de web. Acest conținut ar trebui să fie unic pe o pagină de web.
	Este prescurtarea pentru „list item” și este utilizat pentru a delimita elementul unei liste. Listele pot fi ordonate sau neordonate . Se utilizează și pentru realizarea meniurilor de navigare deoarece elementele unei liste pot fi ușor manipulate cu ajutorul CSS
 	Marchează o listă ordonată. Marchează o listă neordonată Există situații când este necesară combinarea celor două tipuri de liste. De exemplu o listă ordonată care să conțină subliste neordonate: Exemplu: <pre> Prima intrare A doua intrare subitem 1 pentru a doua intrare subitem 2 pentru a doua intrare subitem 3 pentru a doua intrare </pre>

Tag	Descriere
	<pre> A treia intrare </pre> În browser se va afișa:  Fig. 14. Liste ordonate combinate cu subliste neordonate
<p>	Acest tag delimitează un paragraf

2.9.4 Tabele

Tabelul este un element care ne permite să realizăm ușor o structură într-o pagină de web. Poate să conțină orice alt element html. Un browser afișează tabel parcurgându-l linie cu linie. Din păcate nu există posibilitatea de a specifica desenarea unui tabel de 5 linii și 4 coloane de la început. Tabelul va rezulta din interpretarea tagurilor <tr> și <td> și doar la sfârșit putem vedea dimensiunea tabelului. Exemplele pentru această secțiune apar după definirea tuturor tagurilor.

Tag	Descriere
<table>	Este tagul care marchează un tablou unidimensional sau bidimensional.
<caption>	Delimitează titlul unui table. Acest element este întotdeauna primul element al tag-ului <table> și poate fi formata prin CSS.
<col>	Marchează prezența unei coloane într-un tabel
<colgroup>	Definește un grup de coloane într-un tabel
<tbody>	Definește corpul tabelului și este obligatoriu la descrierea unui table. În cazul în care se omite, browser-ul îl consider automat înainte de primul tag <tr>
<td>	Delimitează conținutul unei celule din tabel
<tfoot>	Delimitează footer-ul unui tabel
<th>	Definește celule ce marchează antetul (capul) unui tabel. <pre> <table> <tr> <th>Cap de coloana 1</th> <th>Cap de coloana 2</th> </tr> <tr> <td>rand 1 A</td> <td>rand 1 B</td> </tr> </pre>

Tag	Descriere
	<pre> <tr> <td>rand 2 C</td> <td>rand 2 D</td> </tr> </table> </pre>  Fig. 15. Utilizarea th Se observă că celule delimitate cu tagul <th> sunt scrise îngroșat de către browser. Aceste formatări pot fi modificate cu ajutorul sintaxei CSS iar folosirea acestor tag-uri se face doar pentru a delimita capătul de tabel.
<thead>	Definește un set de rânduri care fac parte din antetul tabelului.
<tr>	Delimitează conținutul unui rând într-un tabel .

În exemplele ce urmează din motive de simplificare nu vom mai afișam întreaga structură standard a documentului html. Practic

exemplele pot fi puse între tag-urile de <body> ca să funcționeze.

Exemplu 1: - marcarea a unui tabel simplu cu două rânduri și pe fiecare rând câte 2 celule.

```
<table>
<tr>
    <td>rand 1, celula 1</td>
    <td>rand 1, celula 2</td>
</tr>
<tr>
    <td>rand 2, celula 1</td>
    <td>rand 2, celula 2</td>
</tr>
</table>
```

Outputul este următorul:

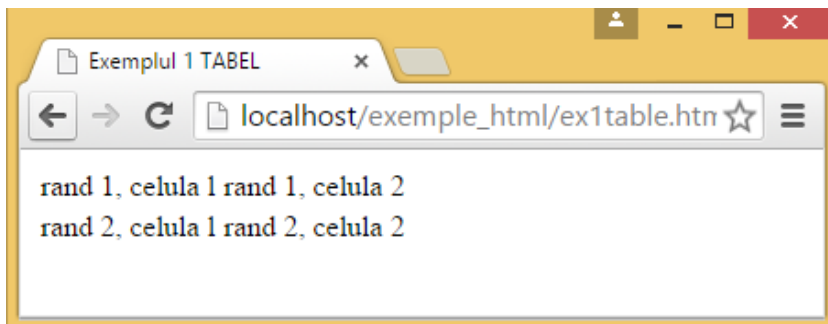


Fig. 16. Tabel simplu

Observație:

Browser-ul nu va afișa liniile care delimitează celulele tabelului.

Pentru a vizualiza liniile se pot folosi unele atribute pentru tag-ul `<table>`. Deoarece în standardul HTML5 se pune accentul foarte tare pe separarea formei de conținut toate formatările – inclusiv cele pentru tabele – se vor face utilizând sintaxa CSS care o vom detalia în capitolul 3.

Totuși pentru a atrage atenția cititorului asupra acestor atribute le vom prezenta pe scurt precum și efectele lor în modul de afișare.

☛ Atributul *align*:

Utilizarea atributului *align* în cadrul tag-ului `<table>`: atributul este separat de un spațiu de denumirea tag-ului și valoare atributului se specifică între ghilimele. În standardul HTML5 s-a renunțat la aceste ghilimele.

`<table align=left>` - înseamnă că browser-ul va alinia tabelul la stânga față de elementul în care este conținut tabelul. Atributul *align* poate să mai ia valorile *left*, *center* sau *right*.

Exemplu:

Alinierea unui tabel în centrul paginii:

```
<table align=center>
  <tr>
    <td>Mama</td>
    <td>Tata</td>
  </tr>
```

```
<tr>
  <td>Ion</td>
  <td>Maria</td>
</tr>
</table>
```

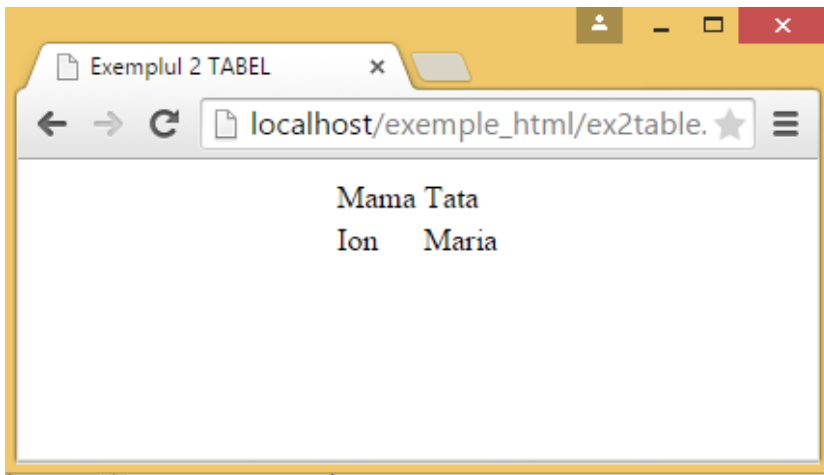


Fig. 17. Tabel aliniat la centrul paginii

☛ Atributul *bgcolor*:

Prin acest atribut se poate seta culoarea de fundal a tabelului. Culorile în HTML respectă modelul RGB și se formează din culorile roșu (Red), verde (Green) și albastru (Blue). Pentru fiecare culoare avem la dispoziție 256 de nuanțe codificate în cod HEXA cu valori de la 00 la FF. Astfel avem la dispoziție $256 \times 256 \times 256$ de culori adică 16.777.216 culori.

În „antichitate”, când plăcile video nu suportau decât 256 de

culori pentru WEB s-au inventat culorile „WEBSAFE” adică culorile care se compuneau utilizând nuanțe codificate doar cu valorile 00, 33, 66, 99, CC și FF adică $6*6*6 = 216$ culori.

În tabelul următor prezentăm câteva nume de culori, pe care browserele le cunosc, și care pot fi utilizate în locul valorilor HEXA corespunzătoare:

	black = "#000000"		green = "#008000"
	silver = "#CoCoCo"		lime = "#00FF00"
	gray = "#808080"		olive = "#808000"
	white = "#FFFFFF"		yellow = "#FFFF00"
	maroon = "#800000"		navy = "#000080"
	red = "#FF0000"		blue = "#0000FF"
	purple = "#800080"		teal = "#008080"
	fuchsia = "#FF00FF"		aqua = "#00FFFF"

Fig. 18. Tabel – denumiri culori

Exemplul:

Adăugare de culoare pe fundalul tabelului:

```
<body>
<table align=center bgcolor=red>
  <tr>
    <td>Mama</td>
    <td>Tata</td>
  </tr>
  <tr>
    <td>Ion</td>
    <td>Maria</td>
```

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

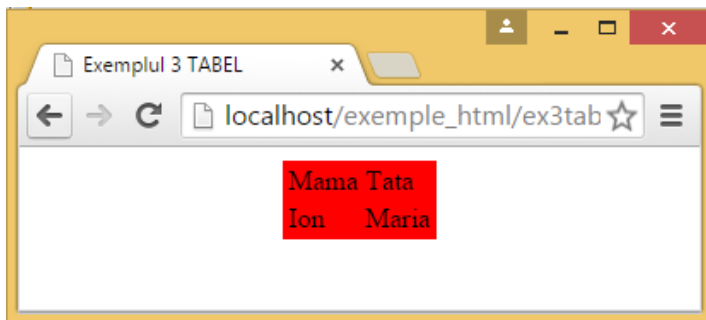


Fig. 19. Tabel cu fundal roșu

☛ Atributul *border*:

Acest atribut stabilește grosimea conturului tabelului.

```
<body>
<table align=center bgcolor=red bgcolor=1>
  <tr>
    <td>Mama</td>
    <td>Tata</td>
  </tr>
  <tr>
    <td>Ion</td>
    <td>Maria</td>
  </tr>
</table>
```

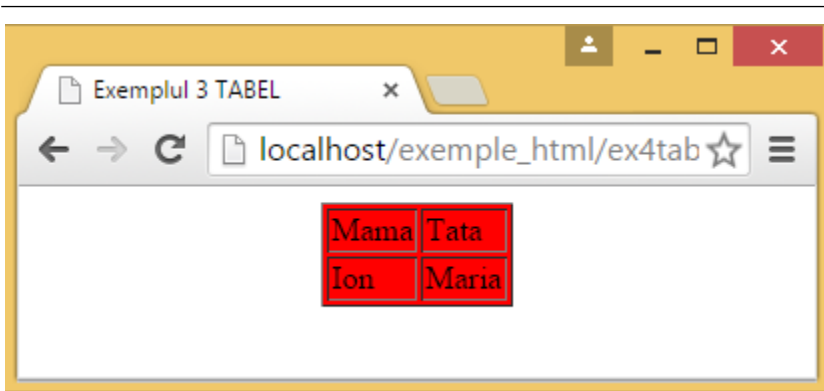


Fig. 20. Tabel cu atribut border=1

☞ Atributele *rowspan*, *colspan*:

Atributele *rowspan*, *colspan* unesc conținutul celulelor. Dacă se utilizează atributul *rowspan* atunci conținutul celulei se va întinde peste atâtea rânduri din tabel cât este specificată valoarea atributului. În cazul atributului *colspan* celula se va întinde peste mai multe coloane.

Exemplu:

```
<table border=1> <caption> Tabel cu
colspan</caption>
  <tr>
    <th>Nume</th>
    <th colspan=2>telefon</th>
  </tr>
  <tr>
    <td>Stefan Cel Mare</td>
    <td>0230.414055</td>
    <td>230.4141195</td>
```

```

        </tr>
</table>
<table border=1>
<caption> Tabel cu rowspan</caption>
    <tr>
        <th>Nume:</th>
        <td>Stefan Cel Mare</td>
    </tr>
    <tr>
        <th rowspan=2>telefon:</th>
        <td>0230.414055</td>
    </tr>
    <tr>
        <td>230.4141195</td>
    </tr>
</table>

```

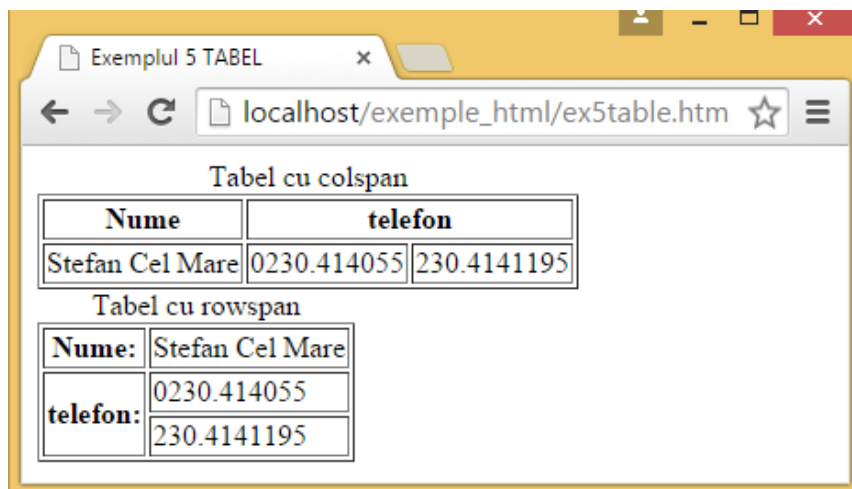


Fig. 21. Utilizarea atributelor *colspan* și *rowspan*

☛ Atributele de *cellspacing* și *cellpadding*

Aceste atribute se folosesc pentru a stabili distanța între celulele unui tabel (*cellspacing*) și distanța textului de la marginea celulei unui tabel (*cellpadding*).

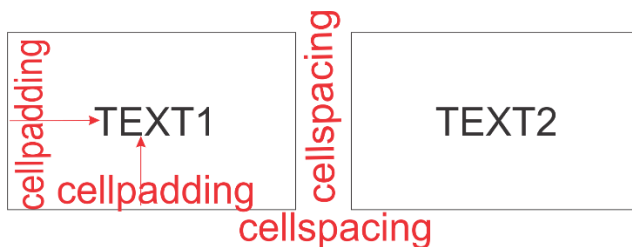


Fig. 22. **Cellspacing - cellpadding**

Vom demonstra în capitolul 3 utilizând proprietăți CSS formatarea tabelului în acest sens. Practic aceste atribute devin inutile în standardul HTML5.

Exemplu: Crearea cu ajutorul atributului de *cellpadding* de spațiu între text și marginea celulei

```
<table border=1 cellpadding=0><caption> Exemplu de  
table fara cellpadding</caption>  
<tr>  
    <th>Cap de coloana 1</th>  
    <th>Cap de coloana 2</th> </tr>  
<tr>  
    <td>rand 1,</td>  
    <td>rand 1, </td>  
</tr>  
<tr>
```

```
        <td>rand 2,</td>
        <td>&nbsp;</td>
</tr>
</table>

<table border=1 cellpadding=10><caption> Exemplu de
table cu cellpadding</caption>
<tr>
        <th>Cap de coloana 1</th>
        <th>Cap de coloana 2</th>
</tr>
<tr>
        <td>rand 1, celula 1</td>
        <td>rand 1, celula 2</td>
</tr>
<tr>
        <td>rand 2, celula 1</td>
        <td>&nbsp;</td>
</tr>
</table>
```

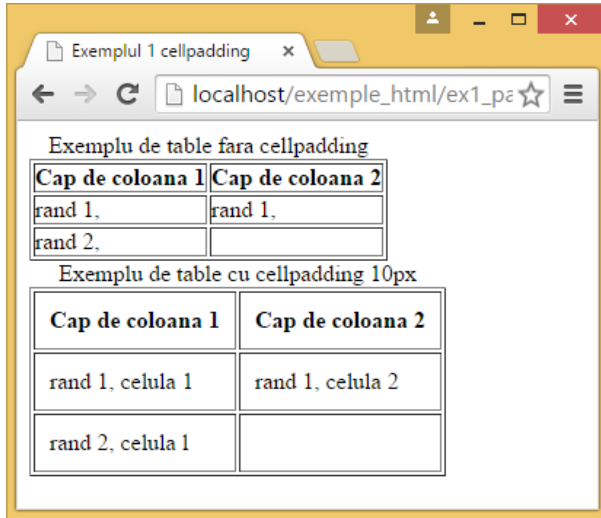


Fig. 23. Utilizarea atributului *Cellpadding*

Exemplu: Crearea cu ajutorul atributului de *cellspacing* de spațiu între celule unui tabel

```
<table border=1 cellspacing=0><caption> Exemplu de
table cu cellspacing=0px</caption>
<tr>
    <th>Cap de coloana 1</th>
    <th>Cap de coloana 2</th>
</tr>
<tr>
    <td>rand 1, celula 1</td>
    <td>rand 1, celula 2</td>
</tr>
<tr>
    <td>rand 2, celula 1</td>
    <td>&nbsp;</td>
```

```
</tr>
</table>
<table border=1 cellspacing=10><caption> Exemplu de
table cu cellspacing=10px</caption>
<tr>
    <th>Cap de coloana 1</th>
    <th>Cap de coloana 2</th>
</tr>
<tr>
    <td>rand 1, celula 1</td>
    <td>rand 1, celula 2</td>
</tr>
<tr>
    <td>rand 2, celula 1</td>
    <td>&nbsp;</td>
</tr>
</table>
```

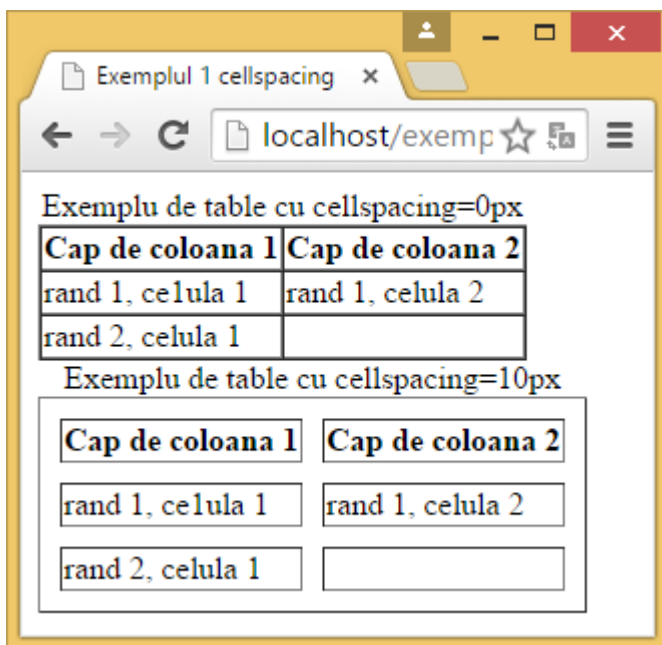
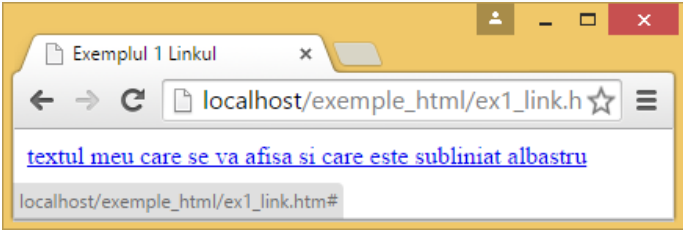


Fig. 24. Utilizarea atributului *Cellspacing*

2.9.5 Tag-uri nestructurale pentru marcare conținutului textului

Element	Descriere
<a>	Tagul <a> definește un link sau o ancoră într-un text. Acest tag a făcut practic posibilă parcurgerea neliniară a unui text. O ancora poate indica spre orice resursă din web: o pagina web, o imagine, un fișier de sunet, un film etc. Tag-ul <a> poate primi atribute.

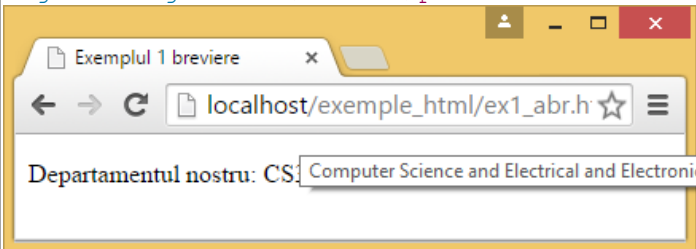
Element	Descriere
	Atributul <i>href</i> În cazul acesta ancora va crea o legătură spre un alt document. Sintaxa pentru crearea unei legături: <pre>textul meu care se va afișa si care este subliniat albastru</pre>  Fig. 25. Linkul Tag-ul <a> primește ca atribut pe <i>href</i> care ia ca valoare adresa resursei spre care ancora face legătura. În cazul în care URL-ul fișierului spre care va pointa linkul nu poate fi numit se utilizează caracterul #. Acest tag este foarte important în stabilirea importanței paginilor de Web de către motoarele de căutare (Search Engine Optimization – SEO). O pagina de web va primi un scor mai bun dacă toate linkurile sunt funcționale. Din acest motiv se folosește caracterul # dacă nu se știe url. Cuvintele scrise între cele două tag-uri <a> vor fi afișate de către browser ca și un hiperlink. Exemplu: <pre> Site-ul Catedrei de Calculatoare si Automatizari</pre>

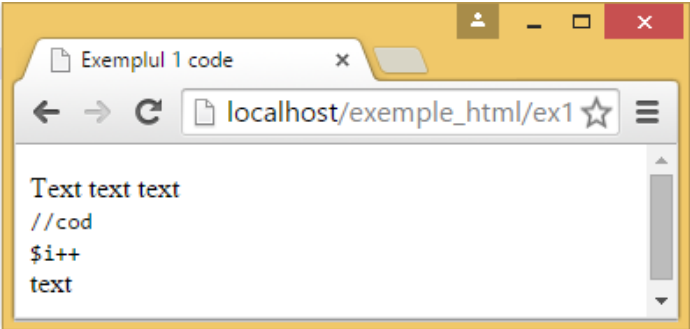
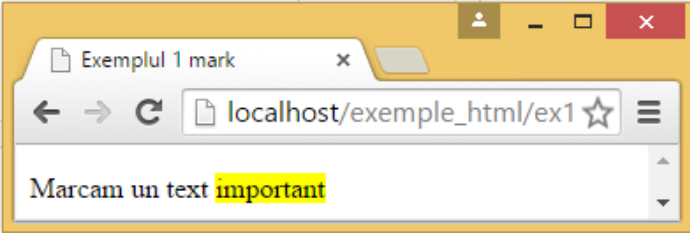
Element	Descriere
	Observație: Acesta este un link extern site-ului și de aceea trebuie trecut la atributul <i>href</i> întreg url-ul inclusiv protocolul http. Atributul <i>target</i> Acest atribut se folosește pentru a defini „locul” unde se va deschide documentul spre care se face legătura. Dacă acest atribut nu se specifică atunci pagina se va deschide în fereastra curentă a browserului. Pentru deschiderea documentului spre care se face legătura într-o fereastră nouă se va folosi atributul <i>target</i> cu valoarea <i>_blank</i> Exemplu: <pre>Site-ul Catedrei de Calculatoare si Automatizari</pre> Atributul <i>name</i> Atributul name este utilizat pentru a crea o ancora cu nume. Aceste ancure sunt foarte des utilizate în cadrul documentelor mari. Pentru a ajunge la un anumit punct din document acela se marchează ca fiind o ancora cu nume. Având această ancoră din orice loc al documentului se poate “sări” direct la secțiunea din document marcată de ea. Acest lucru ușurează enorm navigarea prin documente mari. Sintaxa pentru crearea unei ancure cu nume este: <pre>textul de afisat</pre> Exemplu: <pre>Capitolul 1</pre>

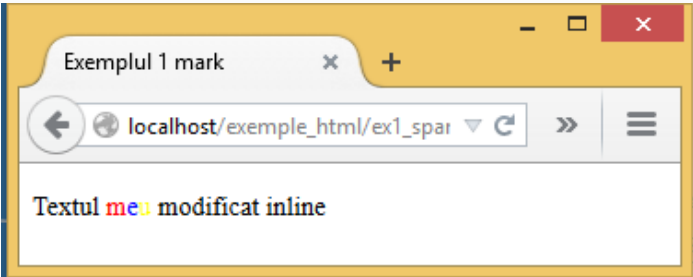
Element	Descriere
	Observație: O ancoră cu nume nu se va afișa de către browser într-un mod anume. Pentru a face o legătură spre o ancoră cu nume se adaugă la sfârșitul numelui documentului spre care se face legătura semnul # urmat de numele ancorei. Exemplu: Pas 1. În documentul <i>ex1.htm</i> avem mai multe denumiri de capitole. Pentru a sări mai ușor la începutul fiecărui capitol definim câte o ancora cu nume: <pre> Capitolul 1 <p> In capitolul 1 povestim basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si intamplari </p> Capitolul 2 <p> In capitolul 2 povestim iarasi basme si intamplari basme si intamplari basme si intamplari basme si intamplari basme si..... </p></pre> Pas 2. Definim legătura direct spre o ancora cu nume. Presupunem ca avem un document web care conține cuprinsul... <pre><body> <p> Cuprinsul</p></pre>

Element	Descriere
	<pre>Capitolul 1
 Capitolul 2
 </body> </html></pre> Observație: Dacă se dorește saltul în cadrul aceluiași document atunci la definirea legăturii se poate utiliza doar numele ancorei precedat de # fără a mai specifica numele fișierului. Exemplu: <pre>Capitolul 1
</pre> Observație generală: Întotdeauna este bine să adăugați la sfârșitul url-ului un / când introduceți adresa spre un subfolder: <i>Exemplu corect:</i> <pre>href=www.csac.ulbsibiu.ro/orar/</pre> Daca nu puneți și caracterul / la sfârșit atunci browserul va face două cereri: prima în care va adăuga automat caracterul / iar a doua când va face cererea cu / adăugat. <i>Exemplu incorect:</i> <pre>href=www.csac.ulbsibiu.ro/orar</pre> Exemple diverse: 1. Crearea unei legături interne sau externe site-ului <pre><html> <body> <p>acest text este un</pre>

Element	Descriere
	link spre pagina ex13.htm </p> <p> acest text este un link la pagina catedrei si se deschide intr-o fereastra noua </p> </body> </html> 2. Crearea unei legături printr-o poză <html> <body> <p>utilizam o poza ca link: </p> </body> </html> 3. Un link spre o adresă de mail <html> <body> <p>un link catre o adresa de mail - daca aveti un client de mail instalat la click pe link-ul următor se deschide fereastra clientului de mail: trimite-ma </p> </body> </html> Observație: Acest exemplu este destul de periculos dacă ținem cont că programele automate (mail-harvester) consideră ca și adrese exacte adresele de mail specificate în acest fel. Vom deveni foarte repede ținta spammer-ilor!!! Este de evita scrierea unei adrese de mail într-o pagină de web. Există varianta plasării unei poze cu textul adresei sau descrierea adrese fără a utiliza caracterele specifice unei adrese de mail:

Element	Descriere
	Exemplu: daniel [dot] morariu [at] ulbsibiu [dot] ro
<abbr>	Delmitează o abreviere. Dacă se folosește atunci trebuie să primească obligatoriu atributul <code>title</code> Exemplu: <pre><p>Departamentul nostru: <abbr title="Computer Science and Electrical and Electronical Engineering">CS3E</abbr></p></pre>  The screenshot shows a web browser window titled 'Exemplul 1 brevier'. The address bar shows 'localhost/exemple_html/ex1_abr.h'. The page content displays 'Departamentul nostru: CS' followed by a tooltip that reads 'Computer Science and Electrical and Electroni'. Fig. 26. Utilizarea abbr afișează un hint în fereastra browserului
 	Tagul
 este un tag care nu are pereche și marchează saltul la rând nou în fereastra browserului. Acest element este important deoarece browserul nu ține cont de enter-urile tastate în codul sursă
<code>	Tagul <code> delimitează un fragment de cod. Browserul îl va afișa utilizând fontul monospace setat. Exemplu: <pre><p>Text text text
 <code>//cod
 \$i++</code>
text </p></pre>

Element	Descriere
	 Fig. 27. Utilizarea tagului code
<dfn>	Tagul <dfn>delimitează o definiție
	Tagul (prescurtarea de la Emphasis Element) scoate în evidență un text. De obicei textul este afișat de către browser cu caracter italic.
<kbd>	Tagul <kbd> marchează un input de utilizator inline afișat de browser cu fontul monospace.
<mark>	Tagul <mark> delimitează text care se dorește a fi scos în evidență <pre><p>Marcam un text <mark>important</mark></p></pre>  Fig. 28. Utilizarea tagului mark
<q>	Acest tag marchează o citare inline. Pentru citări mai lungi se recomandă folosirea tagurilor <blockquote>

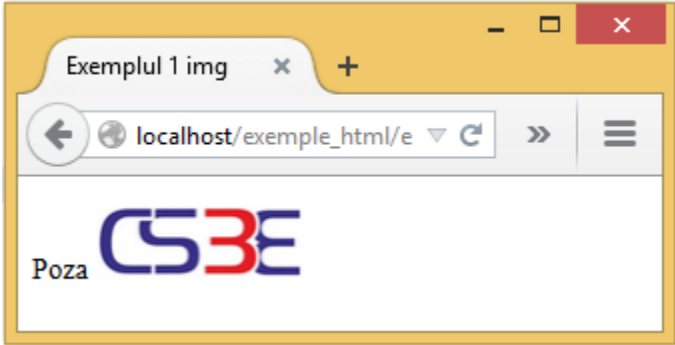
Element	Descriere
<small>	Marchează un text care va fi afișat cu o dimensiune a textului mai mică față de mărimea normală. De exemplu dacă textul este scris cu font-size: <i>small</i> atunci marcajul cu <small> va afișa textul cu font-size: x-small. Atenție pentru HTML5 acest tag se folosește pentru a marca textul scris mic, de exemplu: copyright-ul ...
	Tagul delimitează inline o bucată de text. De obicei se folosește împreună cu sintaxa CSS pentru formătări. Exemplu: <pre><p> Textul m e >u modificat inline</p></pre>  Fig. 29. Utilizare span
	Acest element marchează text care va fi afișat îngroșat.
<sub>	Marchează text care va fi afișat ca index – subscript.
<sup>	Marchează text care va fi afișat ca putere – superscript.
<time>	Tagul <time> marchează o oră sau o dată din

Element	Descriere
	calendar

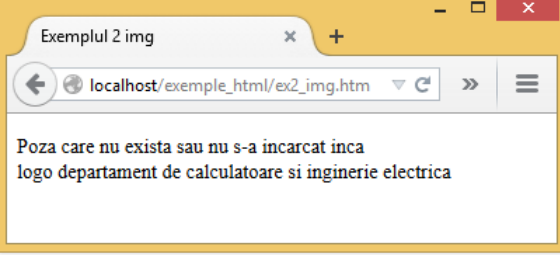
2.9.6 Taguri pentru imagini și multimedia

HTML5 permite inserarea de diverse imagini, video, sunet ...

Element	Descriere
	Tagul inserează o imagine în locul unde browserul întâlnește acest tag. Acest tag nu are un tag de închidere și primește obligatoriu două atribute: <i>src</i> = "url (adresa) fișerului sursă"- care conține o imagine (de obicei de tip <i>jpg</i>, <i>png</i>, <i>gif</i> ...) <i>alt</i> = "textul alternativ" De obicei acest text este afișat în fereastra browserului în cazul în care browserul nu poate afișa imaginea. Acest atribut are și un alt rol mult mai important, acela în indexarea corectă de către motoarele de căutare a pozei respective. Atributul src: Exemplu: <pre><p>Poza </p></pre>

Element	Descriere
	 Fig. 30. Utilizarea tagului img În acest caz fișierul „cs3e-logo.jpg” trebuie să fie salvat în același director cu fișierul curent html care face referire la acest fișier. De obicei pentru a organiza eficient structura site-ului imaginile utilizate în cadrul site-ului vor fi salvate într-un director (<i>img</i> de exemplu) separat care este un subdirector al directorului curent. Exemplu de mai sus devine atunci: <pre><p>Poza </p></pre> Atributul alt Acest atribut se folosește pentru a afișa un text alternativ pentru o imagine. Adică în cazul în

Element	Descriere
	care browserul nu poate afișa imaginea atunci în spațiul rezervat imaginii apare textul specificat de atributul <i>alt</i>. Foarte important!!! Fără acest atribut validatorul de cod html nu va valida documentul. Exemplele de mai sus care afișează imagini nu vor fi validate de validatorul de pagini web care se găsește la adresa http://validator.w3c.org. Exemplu de mai sus cu atributul <i>alt</i>: <pre> <!DOCTYPE html> <html> <head> <title> Exemplul 2 img</title> </head> <body> <p>Poza care nu exista sau nu s-a incarcacat inca
 </p> </body> </html> </pre>

Element	Descriere
	 Fig. 31. Utilizarea atributului alt. Afișare dacă fișierul cu imaginea nu este găsit Atributele <i>width</i> și <i>height</i> Aceste atribute se folosesc pentru stabilirea dimensiunii pozei. De obicei aceste atribute primesc valori numerice exprimate în pixeli. Cu toate că la prima vedere par interesante aceste două atribute modifică doar dimensiunea de afișare a pozei iar dimensiunea fizică a fișierului rămâne aceeași. Adică chiar dacă redimensionăm o poză care a fost preluată de la un aparat foto de 6 Megapixeli dimensiunea pozei pe care o va încărca browserul (o va salva mai întâi în <i>Temporary Internet Folder</i>) va rămâne tot de 3Mb. La o viteză scăzută a internetului acest lucru se poate observa prin efectul de cortina. Recomandăm cu căldură modificarea pozelor cu ajutorul programelor speciale de

Element	Descriere
	prelucrat de imagini și dimensionarea acestora la dimensiunile necesare pe site. Exemplu: <pre> <!DOCTYPE html> <html> <head> <title> Exemplul 3 img</title> </head> <body> <p> </p> <p> </p> <p> </p> <p> Atentie: putem micsora sau mari imaginile cu cele doua attribute width si height dar calitatea pozei are de suferit. Asa ca marimea pozei trebuie sa fie stabilita din programul de editare al pozelor!</p> <p> </p> </body> </html> </pre>

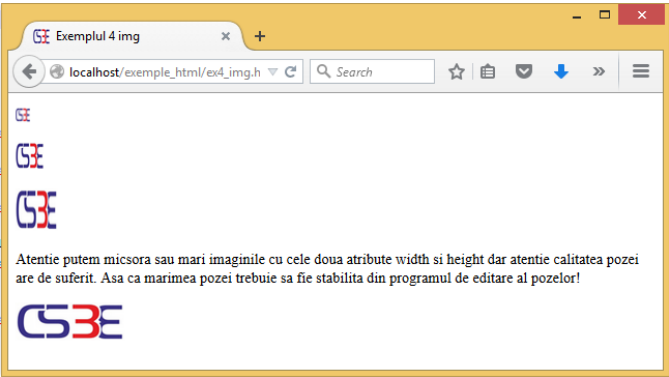


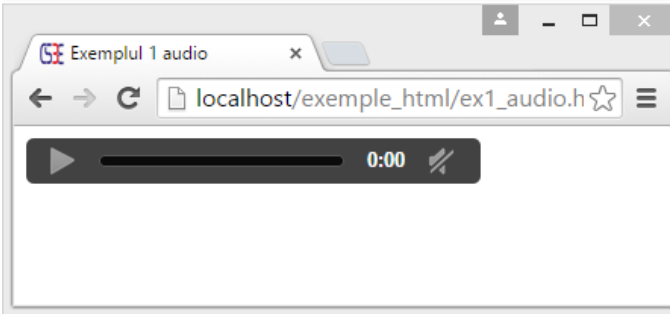
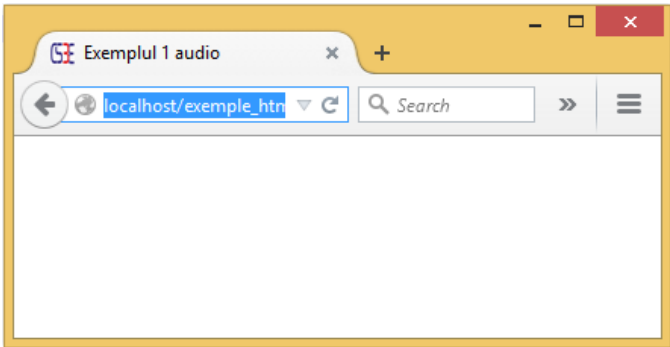
Fig. 32. Utilizarea atributelor width și height

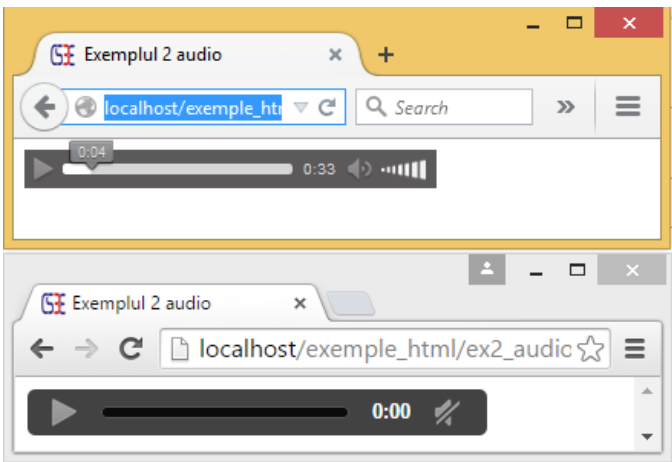
Browserele moderne au posibilitatea de a afișa pe fereastra browserului un logo mic denumit icon.

Pentru a afișa un astfel de logo se creează o imagine de 16x16 sau 32X32 de pixeli și se afișează utilizând sintaxa `<link>`:

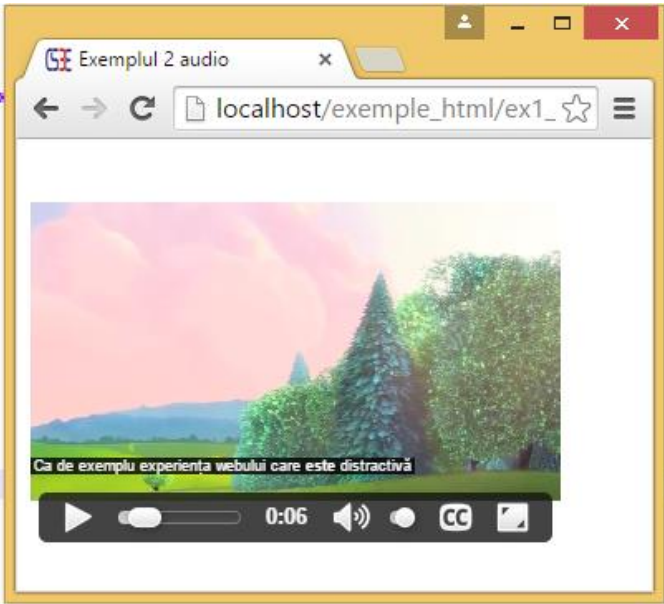
```
<head>
<title> Exemplul 4 img</title>
<link rel="icon" href="img/favicon.png"
sizes="16x16 32x32" type="image/png">
</head>
```

Element	Descriere
	 Fig. 33. Utilizare icon în titlu În versiuni mai vechi apare și atributul de <i>align</i> folosit în poziționarea pozei față de text. Acest atribut este depreciat și se folosește în locul lui sintaxa CSS cu proprietatea <i>vertical-align</i>.
<area>	Tagul <code><area></code> definește o zonă de hot-spot (o zonă pe care se poate face click) pe o imagine care poate fi apoi asociată unui link.
<audio>	Tagul <code><audio></code> se utilizează pentru a include un fișier audio în pagina de web. Exemplu: <pre data-bbox="277 1225 946 1257"><audio src="test.mp3" controls> </audio></pre> Va produce în Mozilla și Chrome afișări diferite:

Element	Descriere												
	 Fig. 34. Utilizarea tagului audio Problema este datorată suportului oferit de diferitele browsere pentru anumite tipuri de fișiere audio. În tabelul de mai jos sunt prezentate fișierele suportate de diversele browsere <table><tr><th>Browser</th><th>.mp3</th><th>.wav</th><th>.ogg</th></tr><tr><td>Mozilla Firefox 3.6</td><td></td><td>✓</td><td>✓</td></tr><tr><td>Opera 10.63</td><td></td><td>✓</td><td>✓</td></tr></table>	Browser	.mp3	.wav	.ogg	Mozilla Firefox 3.6		✓	✓	Opera 10.63		✓	✓
Browser	.mp3	.wav	.ogg										
Mozilla Firefox 3.6		✓	✓										
Opera 10.63		✓	✓										

Element	Descriere												
	<table><tr><td>Google Chrome 8.0</td><td>✓</td><td>✓</td><td>✓</td></tr><tr><td>Apple Safari 5.0.3</td><td>✓</td><td>✓</td><td></td></tr><tr><td>Microsoft IE 9</td><td>✓</td><td>✓</td><td></td></tr></table> Rezolvarea este simplă pentru că utilizând tagul <code><audio></code> putem specifica mai multe surse iar browserul o va alege pe cea suportată de el. <pre><audio controls> <source src="test.mp3" type="audio/mpeg"> <source src="test.ogg" type="audio/ogg"> </audio></pre>  Fig. 35. Utilizarea tagului source	Google Chrome 8.0	✓	✓	✓	Apple Safari 5.0.3	✓	✓		Microsoft IE 9	✓	✓	
Google Chrome 8.0	✓	✓	✓										
Apple Safari 5.0.3	✓	✓											
Microsoft IE 9	✓	✓											
<code><video></code>	Tagul <code><video></code> este utilizat pentru includerea în pagina web a conținutului video. Acest tag poate conține ca și tagul <code><audio></code> mai multe fișiere sursă												

Element	Descriere																								
	care pot fi alese în funcție de browser. Acest tag este suportat de toate browserele moderne începând cu IE9. <table><tr><th>Browser</th><th>MP4</th><th>WebM</th><th>Ogg</th></tr><tr><td>Internet Explorer 9</td><td>✓</td><td>-</td><td>-</td></tr><tr><td>Chrome</td><td>✓</td><td>✓</td><td>✓</td></tr><tr><td>Firefox</td><td>✓</td><td>✓</td><td>✓</td></tr><tr><td>Safari</td><td>✓</td><td>-</td><td>-</td></tr><tr><td>Opera(from Opera 25)</td><td>✓</td><td>✓</td><td>✓</td></tr></table> Exemplu: <pre><video width="320" height="240" controls> <source src="prezentare.mp4" type="video/mp4"> <source src="prezentare.ogg" type="video/ogg"> <track src="subtitles_en.vtt" kind="subtitles" srclang="ro" label="Romana"> <track src="subtitles_de.vtt" kind="subtitles" srclang="de" label="Deutsch"> </video></pre>	Browser	MP4	WebM	Ogg	Internet Explorer 9	✓	-	-	Chrome	✓	✓	✓	Firefox	✓	✓	✓	Safari	✓	-	-	Opera(from Opera 25)	✓	✓	✓
Browser	MP4	WebM	Ogg																						
Internet Explorer 9	✓	-	-																						
Chrome	✓	✓	✓																						
Firefox	✓	✓	✓																						
Safari	✓	-	-																						
Opera(from Opera 25)	✓	✓	✓																						

Element	Descriere
	 Fig. 36. Utilizarea tagului video Tagul <track> permite încărcarea unui fișier cu subtitrare de tip VTT. Un fișier "VTT" conține text care se va afișa peste imaginea video în timp se aceasta se derulează în browser Exemplu fișier .vtt <pre> WEBVTT FILE 1 00:00:03.500 --> 00:00:05.000 vertical:lr align:start Toată lumea dorește tot de la viață 2 </pre>

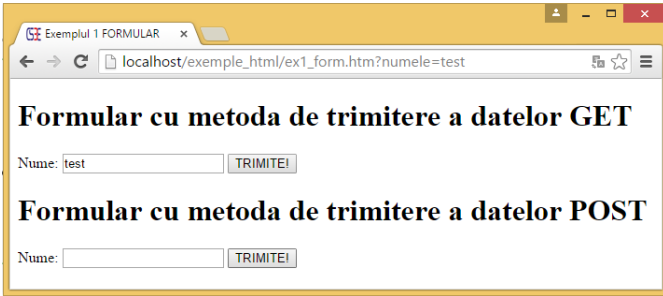
Element	Descriere
	00:00:06.000 --> 00:00:09.000 align:start Ca de exemplu experiența webului care este distractivă 3 00:00:10.500 --> 00:00:15.000 align:start CÂTEODATĂ
<map>	Acest tag este folosit pentru a crea pe un „image map” o zonă care poate fi utilizată ca link. Exemplu: <pre> <!DOCTYPE html> <html> <head> <title> Exemplul 1 map</title> <link rel="icon" href="img/favicon.png" sizes="16x16 32x32" type="image/png"> </head> <body> <!-- Creaza maparea --> <map name="html"> <area shape="circle" coords="50,7,10" href="ex2_map.htm" alt="exemplu map" target="_self" /> </map> </body> </html> </pre> Practic pe imaginea descrisă la <code>src="img/csac-logo.jpg"</code> și care primește atributul <code>usemap="#html</code> se aplică o hartă cu o zonă pe care se poate face click descrisă de tagul area.

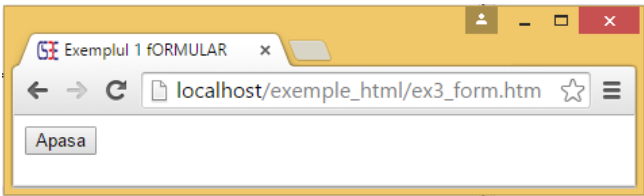
Element	Descriere
<track>	Tagul <track> se utilizează pentru a adăuga subtitluri playerului video primind ca sursă fișiere de tip WebVTT (.vtt) – Web Video Text Tracks.

2.9.7 Formulare

Formularele sunt foarte importante în realizarea interfețelor web pentru diverse aplicații.

Element	Descriere
<form>	Tagul <form> delimitează un formular într-un document web. Practic doar prin intermediul formularelor pot fi trimise date către aplicații care se execută la server. Există 2 metode de a trimite datele: POST și GET POST: datele din formular sunt incluse în corpul formularului și trimise la server. GET: datele din formular sunt incluse în URI cu un "?" ca separator, iar URI rezultat este trimis la server. Această metodă se poate folosi când datele trimise nu sunt importante și conțin numai caractere ASCII. Exemplu: <pre><!DOCTYPE html> <html> <head> <title> Exemplul 1 FORMULAR</title></pre>

Element	Descriere
	<pre> <link rel="icon" href="img/favicon.png" sizes="16x16 32x32" type="image/png"> </head> <body> <h1>Formular cu metoda de trimitere a datelor GET</h1> <form action="ex1_form.htm" method="get"> <label for="nume">Nume:</label> <input id="nume" type="text" name="numele"> <input type="submit" value="TRIMITE!"> </form> <h1>Formular cu metoda de trimitere a datelor POST</h1> <form action="ex1_form.htm" method="post"> <label for="nume">Nume:</label> <input id="nume" type="text" name="numele"> <input type="submit" value="TRIMITE!"> </form> </body> </html> </pre> 

Element	Descriere
	Fig. 37. Formulare – metode de trimitere a datelor În exemplul de mai sus datele au fost trimise cu metoda <i>Get</i> și observăm linkul format <code>http://localhost/exemple_html/ex1_form.htm?numele=test</code>
<code><button></code>	Tagul <code><button></code> definește un buton pe care utilizatorul poate da click pentru a trimite datele de pe formular. Poate să primească atributele <code>formaction</code> și <code>formmethod</code> pentru a trimite date la alt URI decât cel specificat la atributul <code>action</code> al tagului <code><form></code>. Exemplu: <pre><form action="ex2_form.htm" method="post"> <button name="button">Apasa</button> </form></pre>  The screenshot shows a web browser window with the title 'Exemplul 1 FORMULAR'. The address bar displays 'localhost/exemple_html/ex3_form.htm'. The page content includes a single button labeled 'Apasa'. Fig. 38. Button
<code><datalist></code>	Tagul <code><datalist></code> conține un set de taguri <code><option></code> care reprezintă valori disponibile pentru alte elemente. Putem specifica astfel o listă din care utilizatorul poate alege un element.

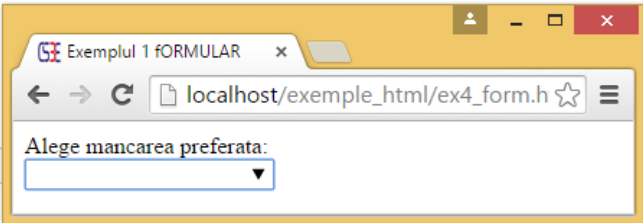
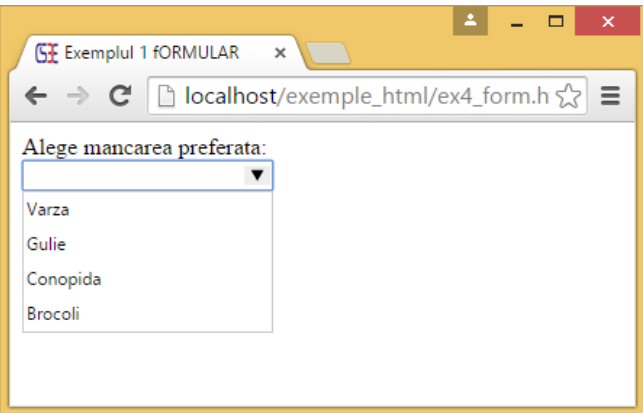
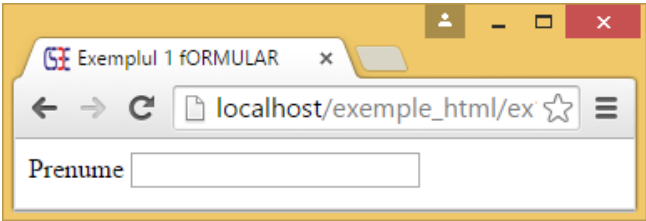
Element	Descriere
	<pre> <form action="ex2_form.htm" method="post"> <p>Alege mancarea preferata:</p> <input list="mancare" name="papa"/> <datalist id="mancare"> <option value="Varza"> <option value="Gulie"> <option value="Conopida"> <option value="Brocoli"> </datalist> </pre>  

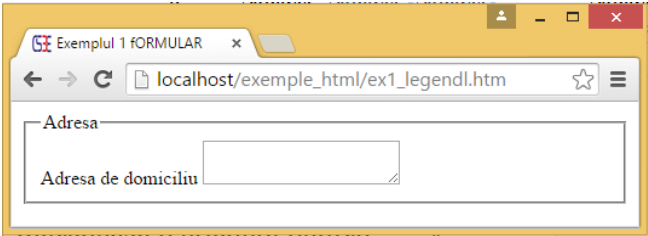
Fig. 39. Datalist

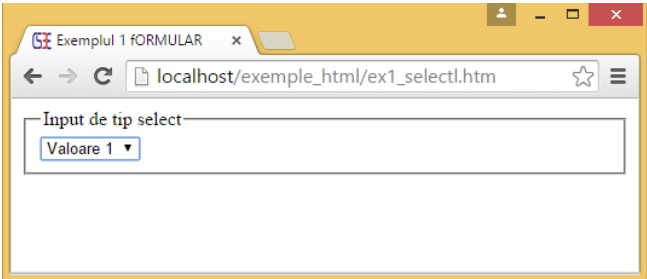
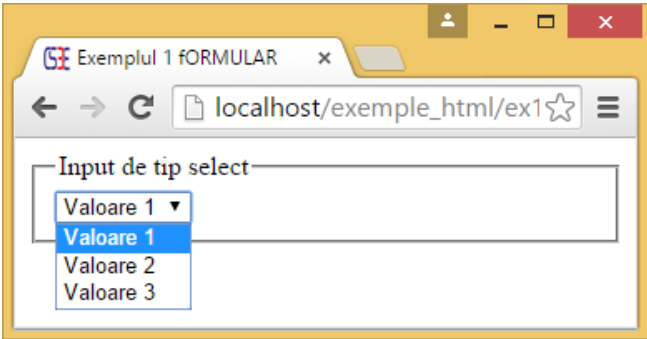
Element	Descriere
<fieldset>	Tagul <fieldset> este utilizat pentru a grupa mai multe elemente de pe un formular web și permite alegerea unui singur element din grup. Exemplu: <pre> <!DOCTYPE html> <html> <head> <title> Exemplul 1 FORMULAR</title> <link rel="icon" href="img/favicon.png" sizes="16x16 32x32" type="image/png"> </head> <body> <form action="ex2_form.htm" method="post"> <fieldset> <legend>Alege!!</legend> <input type="radio" name="radio" id="radio" value="ciocolata"> <label for="radio">Ciocolata</label> <input type="radio" name="radio" id="radio" value="cafea"> <label for="radio">Cafea</label> </fieldset> <input type="submit" value="Trimite!" </form> </body> </html> </pre> 

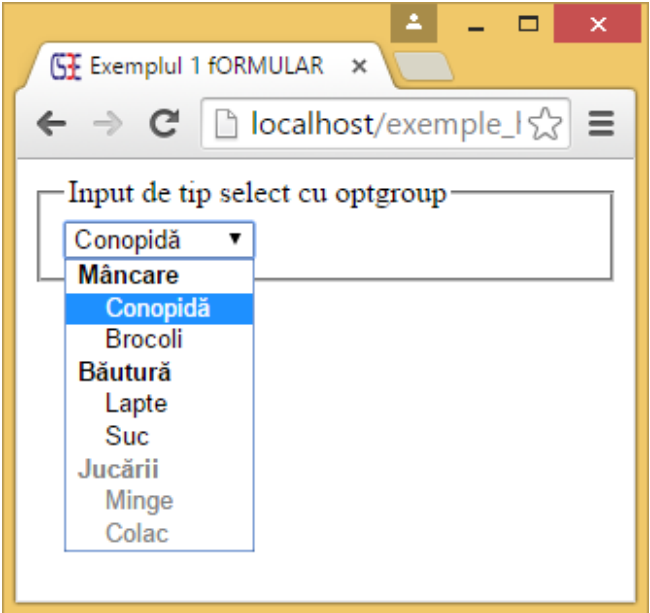
Element	Descriere
	Fig. 40. Fieldset cu legend
<input>	Tagul <input> este folosit în crearea de elemente de preluare a datelor furnizate de user. Forma input-ului diferă în funcție de atributul <code>type</code> utilizat. Vom prezenta în continuare valorile pentru atributul <code>type</code> în ordine alfabetică. Anumite tipuri de input nu sunt suportate de toate browserele. Din acest punct de vedere Chrome oferă suport pentru majoritatea tipurilor de input: • <code>button</code>: un buton cu nici un comportament implicit. • <code>checkbox</code>: o căsuță de selectare. Trebuie să utilizați atributul <i>value</i> pentru a defini valoarea prezentată de acest input. Utilizarea atributului <i>checked</i> indică faptul că acest element este selectat • <code>color</code>: specificarea unei culori ; • <code>date</code>: introducere dată (an, lună, zi); • <code>datetime</code>: introducere data și oră (oră, minute, secunde) fus orar UTC; • <code>datetime-local</code>: introducere de data și timp fără fus orar; • <code>email</code>: introducerea unei adrese de e-mail Acest tip este validat dacă conține în șir o adresă de mail validă; • <code>file</code>: introducere de fișier, utilizat în formulare pentru alegerea unui fișier pentru upload pe server;

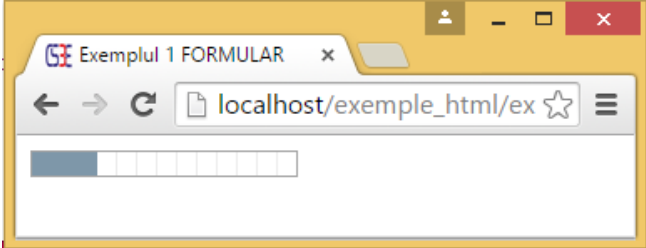
Element	Descriere
	• <code>hidden</code>: tip ascuns dar valoarea este trimisă; • <code>image</code>: un buton graphic. Trebuie specificat atributul <code>src</code> și <code>alt</code> pentru poza care se va utiliza; • <code>month</code>: introducerea lunii și anului fără fus orar; • <code>number</code>: introducerea unei valori de tip float; • <code>password</code>: introducerea unei parole. Literele vor fi înlocuite cu caracterul <code>*</code>; • <code>radio</code>: radio button. Trebuie specificat atributul <code>value</code> pentru a defini valoarea pentru opțiunea aleasă. Atributul <code>checked</code> indică elementul selectat implicit. Doar un singur element dintr-un grup de radiobutton-uri cu același atribut <code>name</code> poate fi selectat la un moment dat; • <code>range</code>: introducerea unui număr a cărei valoare nu este exactă. Dacă nu se specifică altfel valorile implicite pentru attributele pentru acest tip sunt: ◦ <code>min</code>: 0 ◦ <code>max</code>: 100 ◦ <code>value</code>: $\min + (\max - \min) / 2$, or <code>min</code> dacă <code>max</code> este mai mic ca <code>min</code> ◦ <code>step</code>: 1; • <code>reset</code>: un buton care resetează toate inputurile formularului; • <code>search</code>: un câmp pentru introducerea

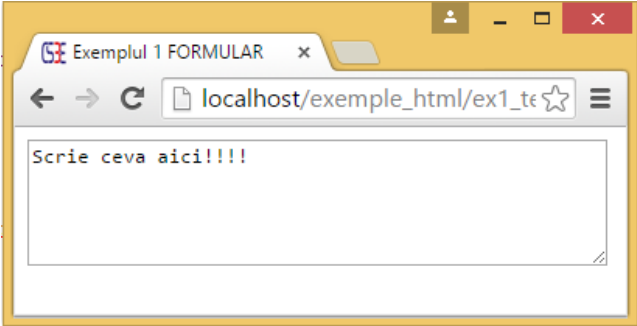
Element	Descriere
	șirurilor de căutare; enter-urile sunt automat eliminate; • <code>submit</code>: un buton care trimite valorile introduse în formular; • <code>tel</code>: introducerea unui număr în format număr de telefon; • <code>text</code>: câmp pentru introducerea de text pe o singură linie; • <code>time</code>: introducerea orei fără fus orar; • <code>url</code>: introducerea unui URL. Acest tip este validat dacă conține un șir vid sau un URL valid; Exemplu mai interesant este prezentat la sfârșitul acestei secțiuni
<code><label></code>	Tagul <code><label></code> este o etichetă pentru un alt element de pe interfață. De obicei se utilizează împreună cu atributul <code>for</code>. Exemplu: <pre><label for="prenume">Prenume</label> <input id="prenume" name="prenume" type="text" ></pre>  The screenshot shows a web browser window titled 'Exemplul 1 FORMULAR'. The address bar shows 'localhost/exemple_html/ex'. The page content displays the text 'Prenume' followed by a text input field.

Element	Descriere
	Fig. 41. Label
<legend>	Tagul <legend> marchează o etichetă pentru tagul <fieldset>. Exemplu: <pre> <fieldset> <legend>Adresa</legend> <label for="adresa">Adresa de domiciliu</label> <textarea id="adresa" name="adresa" rows="2" required></textarea> </fieldset> </pre>  Fig. 42. Legenda
<select>	Tagul <select> este un element de input care oferă utilizatorului o listă de valori predefinite. Elementele din listă se marchează cu tagul <option>. Acest tag va utiliza atributul value pentru a stabili valoarea pentru selecție Exemplu: <pre> <select name="sel"> </pre>

Element	Descriere
	<pre> <option value="1">Valoare 1</option> <option value="2">Valoare 2</option> <option value="3">Valoare 3</option> </select> </pre>   Fig. 43. Select
<optgroup>	Tagul <optgroup> grupează în cadrul elementului <select> opțiunile. Exemplu: <pre> <select> <optgroup label="Mâncare"> <option value="M_c">Conopidă</option> <option value="M_b">Brocoli</option> </pre>

Element	Descriere
	<pre> </optgroup> <optgroup label="Băutură"> <option value="B_l">Lapte</option> <option value="B_s">Suc</option> </optgroup> <optgroup label="Jucării" disabled> <option>Minge</option> <option>Colac</option> </optgroup> </select> </pre>  Fig. 44. Optgroup În acest exemplu se observă că a al treilea grup de opțiuni a fost dezactivat și browserul nu va permite accesarea de către utilizator.
<option>	Tagul <option> delimitează opțiunile pentru un

Element	Descriere
	tag <select>, <optgroup> sau <datalist> Tagul a fost folosit deja în exemplele precedente.
<output>	Tagul <output> delimitează un rezultat al unui calcul sau a unei acțiuni a utilizatorului. Acest tag se regăsește în exemplu de la finalul acestei secțiuni
<progress>	Tagul <progress> va afișa o bară de progres pentru completarea unei acțiuni. Pentru utilizarea acestui tag se folosește limbajul de script Javascript. Exemplu: <pre><progress id="progressbar" value="25" max="100"></progress></pre>  Fig. 45. Progress Animarea acestui control cu ajutorul librăriei jQuery o vom prezenta în capitolul Javascript.
<textarea>	Tagul <textarea> delimitează un input pe mai

Element	Descriere
	multe linii. Exemplu: <pre><textarea name="texmutiplu" rows="5" cols="50">Scrie ceva aici!!!!</textarea></pre>  The screenshot shows a web browser window titled 'Exemplul 1 FORMULAR'. The address bar shows 'localhost/example_html/ex1_te'. The main content area displays a text area with the text 'Scrie ceva aici!!!!'. Fig. 46. Textarea

Exemplu de formular deocamdată neformatat:

```
<form id="register"
oninput="output.value=preg.value">
<p>Inscrie-te la concursul de gatit!</p>
  <fieldset>
    <legend>Informații personale</legend>
    <div>
      <label>Prenume
      <input id="prenume" name="prenume"
type="text" placeholder="Un singur prenume" required
autofocus>
      </label>
    </div>
    <div>
      <label>Nume
```

```

        <input id="nume" name="nume" type="text"
placeholder="Numele de familie" required autofocus>
    </label>
</div>
<div>
    <label>Data nasterii
        <input id="datan" name="datan" type="date"
required>
    </label>
</div>
<div>
    <label>Email
        <input id="email" name="email" type="email"
placeholder="baburiba@test.ro" required>
    </label>
</div>
<div>
    <label>Site-ul personal
        <input id="url" name="url" type="url"
placeholder="http://csac.ulbsibiu.ro">
    </label>
</div>
<div>
    <label>Telefon
        <input id="tel" name="tel" type="tel"
placeholder="Ex. +40 7123 000000" required>
    </label>
</div>
<div>
    <label>Mărima cuțitului
        <input id="cutit" name="cutit"
type="number" min="15" max="30" step="0.5"
value="20">
    </label>
</div>
<div>
    <label>Nivelul de pregătire (1 mic - 100
maxim)
        <input id="preg" name="preg"
type="range" min="1" max="100" value="0">
        <output name="output" for="preg">

```

```

onforminput="value = preg.value;">0</output>
        </label>
    </div>
</fieldset>
<fieldset>
    <legend>Adresa</legend>
    <div>
        <label>Adresa
        <textarea id="adresa" name="adresa" rows=3
required></textarea>
        </label>
    </div>
    <div>
        <label>Cod poș tal
        <input id="codp" name="codp" type="text"
required>
        </label>
    </div>
    <div>
        <label>Student la:
        <input id="specializare" name="specializare"
list="num-spec" type="text" required>
        <datalist id="num-spec">
            <option label="Calculatoare"
value="C"></option>
            <option label="Tehnologia informației"
value="TI"></option>
            <option label="Ingineria Sistemelor
Multimedia" value="ISM"></option>
            <option label="Agronomie"
value="A"></option>
        </datalist>
        </label>
    </div>
</fieldset>
<fieldset>
    <legend>Detalii card</legend>
    <fieldset>
        <legend>Tip card </legend>
        <div class="card">
            <input id="visa" name="cardtype"

```

Exemplul 1 FORMULAR

localhost/example_html/ex6_form.htm

Inscrie-te la concursul de gatit!

Informații personale

Prenume

Un singur prenume

Nume

Numele de familie

Data nasterii

mm/dd/yyyy

Email

baburiba@test.ro

Situl personal

http://csac.ulbsibiu.ro

Telefon

Ex. +40 7123 000000

Mărimea cuțitului

20

Nivelul de pregătire (1 mic - 100 maxim)

25

Adresa

Adresa

Cod poștal

Student la:

Detalii card

Tip card

☐ VISA ☐ Mastercard

Nume

Numele cum apare pe card

Numar card

SCod de securitate

Data de expirare

Inscrie-te!!!!!!!!

Sterge

95

Fig. 47. Formular

2.9.8 Taguri depreciate

În continuare vom prezenta o listă de taguri și atribute ale unor taguri care sunt depreciate pentru standardulHTML5.

Taguri:

Tagul	Descriere
<acronym>	acronim
<applet>	aplet
<basefont>	definește fontul de bază al paginii
<big>	text scris mare
<center>	text centrat
<dir>	listă din director
	modalitate de afișare a textului
<frame>	definește un cadru - frame
<frameset>	set de cadre
<isindex>	input
<noframes>	secțiune fără frame-uri
<s>	definește text tăiat
<strike>	definește text tăiat
<tt>	definește text e tip teletype
<u>	definește text subliniatt

Attribute: în coloana stângă afișăm denumirea tagului iar în coloana dreaptă attributele care au fost eliminate pentru aceste taguri:

Tagul	Attribute eliminate
link, a	rev
link și a	charset
a	shape
a	coords
img și iframe.	longdesc
link	target
area	nohref
head	profile
html	version
img	name
meta	scheme
object	archive
object	classid
object	codebase
object	codetype

Tagul	Atribute eliminate
object	declare
object	standby
param	valuetype
param	type
td și tr	axis
td i tr	abbr
td	scope
caption, iframe, img, input, object, legend, table, hr, div, h1, h2, h3, h4, h5, h6, p, col, colgroup, tbody, td, tfoot, th, thead și tr.	align
body	alink
body	link
body	vlink
body	text
body	background
table, tr, td, th și body.	bgcolor
table și object.	border
table	cellpadding
table	cellspacing

Tagul	Atribute eliminate
col, colgroup, tbody, td, tfoot, th, thead și tr.	char
col, colgroup, tbody, td, tfoot, th, thead și tr.	charoff
br	clear
dl, menu, ol și ul.	compact
table	frame
dl, menu, ol și ul.	compact
table	frame
iframe	frameborder
img și object.	hspace
img și object.	vspace
iframe	marginheight
iframe	marginwidth
hr	noshade
td și th	nowrap
table	rules
iframe	scrolling
hr	size
li, ol și ul.	type
col, colgroup, tbody, td,	valign

Tagul	Atribute eliminate
tfoot, th, thead și tr	
hr, table, td, th, col, colgroup și pre.	width

2.10 Culori HTML

Pentru textul sau fundalul de pe paginile web culorile se obțin utilizând modelul RGB (RED, GREEN, BLUE)

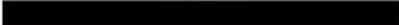
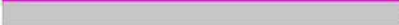
Valorile culorilor

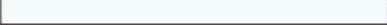
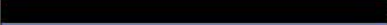
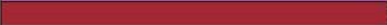
Culorile sunt definite cu ajutorul notației hexazecimale pentru fiecare dintre cele trei culori de baza: roșu, verde, albastru. Cea mai mica valoare care se poate da unei culori este #00 iar cea mai mare este #FF. Pentru proprietățile CSS culorile se pot specifica în mai multe moduri. Cele mai uzuale sunt culori specificate în cod HEXA sau direct în model RGB. În modelul HEXA codul culorii începe cu caracterul # urmat de 3 grupuri a câte 2 caractere fiecare pentru nuanțele celor 3 culori: roșu, verde și albastru.

Numele culorilor

Există aproximativ 500 de nume de culori recunoscute de majoritatea browserelor. De exemplu: *aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, yellow* etc.

Exemple de culori

	Color HEX	Color RGB
	#000000	rgb(0,0,0)
	#FF0000	rgb(255,0,0)
	#00FF00	rgb(0,255,0)
	#0000FF	rgb(0,0,255)
	#FFFF00	rgb(255,255,0)
	#00FFFF	rgb(0,255,255)
	#FF00FF	rgb(255,0,255)
	#C0C0C0	rgb(192,192,192)
	#FFFFFF	rgb(255,255,255)

Culoare	Color HEX	Nume
	#F0F8FF	AliceBlue
	#FAEBD7	AntiqueWhite
	#7FFFD4	Aquamarine
	#000000	Black
	#0000FF	Blue
	#8A2BE2	BlueViolet
	#A52A2A	Brown

Daca am dori să folosim toate nuanțele putem să alegem din 16 milioane de culori (256x256x256) disponibile.

663300	663333	663366	663399	6633CC	6633FF
666600	666633	666666	666699	6666CC	6666FF
669900	669933	669966	669999	6699CC	6699FF
66CC00	66CC33	66CC66	66CC99	66CCCC	66CCFF
66FF00	66FF33	66FF66	66FF99	66FFCC	66FFFF
990000	990033	990066	990099	9900CC	9900FF
993300	993333	993366	993399	9933CC	9933FF
996600	996633	996666	996699	9966CC	9966FF
999900	999933	999966	999999	9999CC	9999FF
99CC00	99CC33	99CC66	99CC99	99CCCC	99CCFF
99FF00	99FF33	99FF66	99FF99	99FFCC	99FFFF
CC0000	CC0033	CC0066	CC0099	CC00CC	CC00FF
CC3300	CC3333	CC3366	CC3399	CC33CC	CC33FF
CC6600	CC6633	CC6666	CC6699	CC66CC	CC66FF
CC9900	CC9933	CC9966	CC9999	CC99CC	CC99FF
CCCC00	CCCC33	CCCC66	CCCC99	CCCCCC	CCCCFF
CCFF00	CCFF33	CCFF66	CCFF99	CCFFCC	CCFFFF
FF0000	FF0033	FF0066	FF0099	FF00CC	FF00FF
FF3300	FF3333	FF3366	FF3399	FF33CC	FF33FF
FF6600	FF6633	FF6666	FF6699	FF66CC	FF66FF
FF9900	FF9933	FF9966	FF9999	FF99CC	FF99FF
FFCC00	FFCC33	FFCC66	FFCC99	FFCCCC	FFCCFF
FFFF00	FFFF33	FFFF66	FFFF99	FFFFCC	FFFFFF

2.11 Poziționarea conținutului unei pagini html

O pagină web este formată din mai multe elemente - logo, text informativ, imagini, hyperlinkuri, structura de navigație etc. În principiu „în antichitatea Webului” poziționarea textului și a imaginilor a fost efectuată utilizând tabele, apoi cadre. Varianta modernă consideră fiecare element structural component al unei pagini de web ca fiind o diviziune separată și delimitată de tagul `<div>`. Acest element oferă o flexibilitate sporită în adaptarea paginilor web la noile medii de afișare.

2.12 Marcajul semantic

HTML5 oferă un set de taguri speciale care permit structurarea logică a informațiilor pentru paginile web. Aceste elemente sunt adesea denumite Semantic Markup întrucât ele

transmit sensul și scopul lor în mod clar browser-ului.

Dezvoltatorii web utilizează frecvent `<div>` elementul la aspectul paginile lor web. Cu toate acestea, elementul `<div>` în sine nu transmite ceea ce reprezintă într-o pagină web. Un tag `<div>` poate încheia un meniu de navigație sau poate găzdui o listă de blog, astfel folosind doar `<div>` nu transmitem informații semantice despre tipul de conținut marcat. De obicei, clasele CSS aplicate elementului `<div>` dezvăluie unele informații despre scopul propus.

HTML5 include un set de elemente de marcaje care transmit unele aspecte semantice legate de conținut atât browser-ului cât și motorului de căutare. Aceste noi elemente au nume semnificative, astfel încât doar citindu-le oferă o idee clara despre conținutul pe care îl marchează. Mai jos (aceasta nu este o listă exhaustivă) le prezentăm pe cele mai utilizate:

- `<header>`
- `<footer>`
- `<section>`
- `<article>`
- `<aside>`
- `<nav>`

Un exemplu schematic al structurării unei pagini de web utilizând aceste taguri este prezentat în figura următoare:

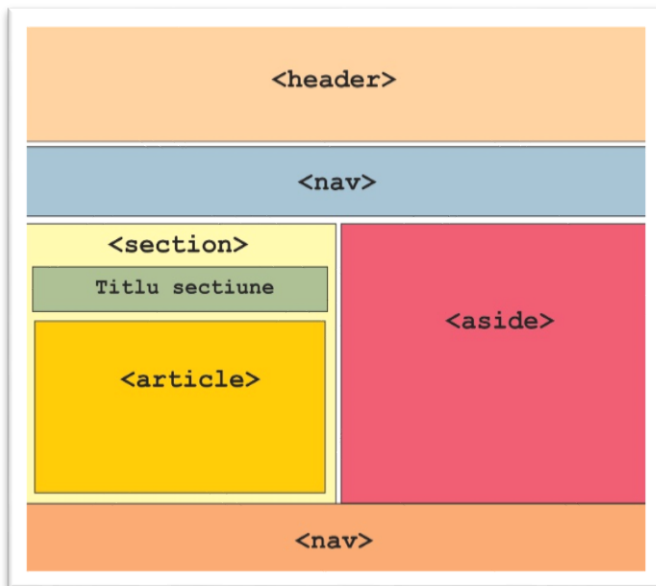


Fig. 48. Structurarea unei pagini Web cu taguri semantice

Totuși pentru a obține un template corespunzător aceste taguri trebuie formate cu ajutorul sintaxei CSS.

Exemplu:

```
<!DOCTYPE html>
<html>
<head>
<title> Exemplul 1 FORMULAR</title>
<link rel="icon" href="img/favicon.png" sizes="16x16
32x32" type="image/png">
<link rel="stylesheet" href="css/stil.css"
type="text/css">
<meta charset="utf-8">
</head>
<body>
```

```

<header>
  <h1>Acesta este un Antet</h1>
</header>
<nav>
  <ul>
    <li><a href="#">Home</a></li>
    <li><a href="#">Note</a></li>
    <li><a href="#">Studenti</a></li>
  </ul>
</nav>
<section>
  <h1>Acesta este antetul secțiunii</h1>
  <article>
    <h1>Acesta este antetul pentru article</h1>
    <p>
      Test Test Test Test Test Test Test Test
      Test Test Test Test Test Test Test
    </p>
  </article>
  <p>
    Test Test Test Test Test Test Test Test
    Test Test Test Test Test Test Test Test
  </p>
</section>
<aside>
  <figure>
    
    <figcaption>Logo-ul CSAC</figcaption>
  </figure>
  <p>
    Test Test Test Test Test Test Test Test
    Test Test Test Test Test Test Test Test
  </p>
</aside>
<footer>
  RC+DM &copy; 2015.
</footer>
</body>
</html>

```



Fig. 49. Structura unei pagini utilizând marcatori semantici

Incluzând fișierul de stil css vom avea următorul output:



Fig. 50. Poziționarea conținutului delimitat cu marcatori semantici

3 CSS

3.1 Ce este CSS?

Pe scurt:

- **CSS** este prescurtarea de la **Cascading Style Sheets** (foi de stiluri cascadeate);
- Stilurile definesc modul de afișare a elementelor HTML;
- Stilurile sunt conținute în foi de stiluri - **Style Sheets**;
- Stilurile au fost adăugate în HTML pentru a rezolva o problemă și anume separarea formei de conținut;
- Foile de stil externe sunt salvate în fișiere separate **CSS**;
- Definirea multiplă de stiluri se va **cascada** într-una singură.

Problema pe care o rezolvă utilizarea stilurilor de editare este faptul că se separă conținutul unui document HTML, marcat prin diverse tag-uri care îl împart în paragrafe, tabele, titluri etc., de forma sa.

Pentru motoarele de căutare actuale și chiar pentru web-ul semantic viitor această separare este esențială. Astfel deja în standardul XHTML nu se mai acceptă tag-uri de formatare a

textului cum ar fi cu attributele sale de size, color,....

Problema formătărilor „externe” a unui document HTML a fost rezolvată cu ajutorul CSS-urilor care sunt recunoscute deja de către Netscape Navigator 4 și IE 4.

CSS-urile salvează foarte multă muncă. Dacă doriți să modificați modul cum arată un site întreg este de ajuns să se modifice fișierul CSS corespunzător. Utilizarea CSS a devenit tot mai importantă după apariția standardului HTML4.01 și mai ales după standardul XHTML1.0 când separarea formei de conținut a devenit o cerință clară a paginilor moderne de HTML. Practic HTML5 nu există fără utilizarea CSS.

CSS este folosit pentru a defini modul în care conținutul este afișat ca și structură și ca formă: diferite fonturi, culori, margini, fundaluri etc.

3.2 Versiuni CSS

La momentul actual, există două versiuni majore CSS în uz: CSS 2.1 și CSS 3.0. Vom prezenta în continuare cele mai des utilizate caracteristici ale celor 2 versiuni. La urma urmei, de îndată ce o anumită caracteristică CSS este acceptată și implementată în browsere, nu mai este important din care versiune a venit respectiva proprietate de la CSS 2.1 sau CSS 3.0.

În capitolele anterioare am menționat faptul că în standardele HTML 4 și XHTML 1.0 se recomandă iar în standardul HTML5 se impune formatarea textului unui

document html în afara sa cu ajutorul unei „foi de stil”. Declararea unui stil se poate face în trei moduri:

- în cadrul instrucțiunii html
- în foaie internă
- în foaie externă documentului HTML

3.3 Declararea stilului inline în html

Cel mai simplu mod de a include CSS într-o pagină HTML este de a încorpora sintaxa CSS în interiorul atributului *style* al unui element HTML. Proprietățile CSS integrate într-un atribut de stil se aplică numai pentru elementul HTML în care sunt incluse. Atributul *style* poate conține în acest caz orice valoare a unei proprietăți CSS.

```
<p style="color:red"> Propozitia mea. </p>
```

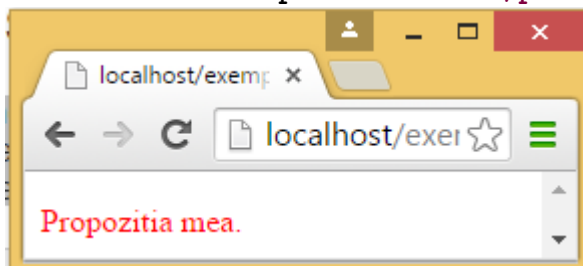


Fig. 51. Declarea inline a stilurilor

În exemplul de mai sus paragraful în care apare atributul *style* va fi afișat cu culoarea roșie. Dacă doriți utilizarea mai multor proprietăți CSS atunci „instrucțiunile” se separă prin „;”

```
<p style="color:red;font-size:large;font-family:Arial;"> Propozitia mea. </p>
```

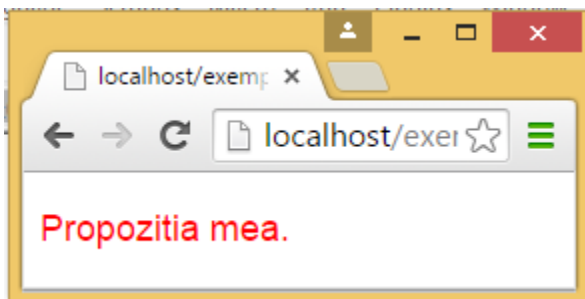


Fig. 52. Aplicarea mai multor proprietăți CSS

3.4 Foaia de stil internă

Acest tip de foaie de stil se folosește atunci când stilul de formatare se va utiliza doar în cadrul unui singur document HTML. Ca și în cazul foii de stil externe definirea unei foi de stil interne se face în secțiunea de head cu ajutorul tag-ului `<style>`.

Exemplu

```
<!DOCTYPE html>
<html>
<head>
  <style type="text/css">
    body {background-color: #ff0000}
    h1 {font-face: Verdana}
    p {margin-left: 30px}
  </style>
</head>

<body>
  <h1> Titlul important</h1>
  <p> Un paragraf care va avea o margine de 30px din
stanga</p>
</body>
```

</html>



Fig. 53. Declararea stilurilor în interiorul tagurilor <style>

3.5 Foaia de stil externă

Utilizarea unei foi de stil externe (un fișier css) este utilă atunci când se dorește modificare formatării unui site întreg. Practic orice modificare a unui stil în fișierul CSS duce la modificarea formatării paginilor site-ului cu condiția ca în paginile HTML ale site-ului să fie inclusă o legătură către foaia de stil CSS modificată.

Există două modalități de a include un fișier de stil:

- Utilizând tagul <link> în zona delimitată de tagul <head>;
- Utilizând instrucțiunea @import în zona delimitată de tagul <style>.

3.5.1 Utilizarea tagului <link>

Exemplu:

```
<!DOCTYPE html>
<html>

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

<body>
  <div> Formataam un div! </div>
  <span> Formataam un span! </span>
</body>
</html>
```

Tagul `<link>` primește ca atribut `href="css/exemplu.css"` care este calea și denumirea fișierului CSS. Marele avantaj al acestei metode este reutilizabilitatea fișierului de stil și modificarea foarte simplă a stilurilor. Un fișier HTML poate să conțină linkuri spre mai multe fișiere CSS. Fișierul CSS este un fișier text care conține o sintaxă specifică.

Exemplu de fișier CSS:

```
div {
  border: 1px dashed black;
}
span {
  border: 1px solid #ff0000;
}
```

Fără includerea fișierului de stil browserul va afișa conținutul exemplului de mai sus astfel:

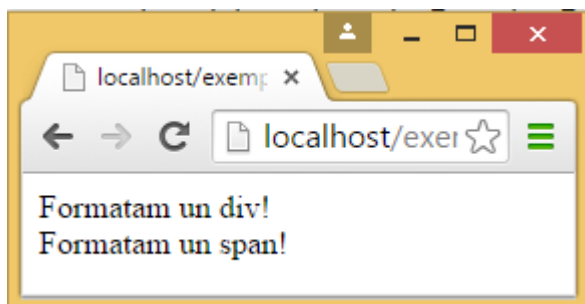


Fig. 54. Fără fișier CSS inclus

După includerea fișierului de stil browser-ul afișează:

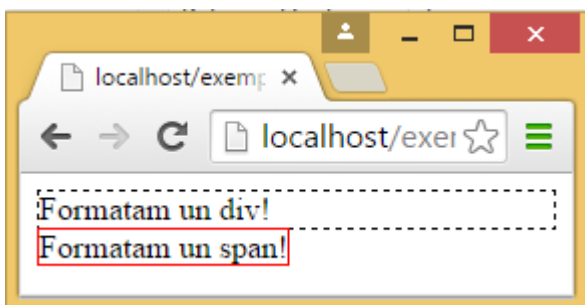


Fig. 55. Fișier cu CSS inclus

3.5.2 Utilizarea instrucțiunii @import

Un fișier CSS extern poate fi importat în cadrul fișierului HTML utilizând în interiorul tagurilor `<style>` instrucțiunea `@import`.

Un exemplu de utilizare:

```
<!DOCTYPE html>
<html>
<head>
  <style>
    @import
url('http://webspace.ulbsibiu.ro/radu.kretzulescu/cs
s/exemplu.css');
  </style>
</head>
<body>
  <div> Formatam un div! </div>
  <span> Formatam un span! </span>
</body>
</html>
```

Aici este importat un fișier CSS care se găsește la o adresă URL care este externă site-ului curent.

3.6 Ordinea de cascada a stilurilor.

La începutul acestui capitol am prezentat mai multe modalități de a declara un stil: în cadrul unui tag, în partea de <head> a documentului, într-un fișier extern CSS sau chiar în mai multe fișiere externe. Dacă la un moment dat se folosesc toate aceste moduri de declarat stiluri, acestea se vor cascada într-un stil nou, “virtual” după următoarea regulă - unde 1 are cea mai mică prioritate iar 4 cea mai mare:

1. Setări implicite ale browser-ului

-
2. Foaie de stil externă - fișier CSS
 3. Foaie de stil internă - declarată în interiorul tag-ului <head>
 4. Declaraire în interiorul elementului HTML

Practic o declaraire de stil în cadrul unui element HTML va suprascrie orice stil declarat în partea de <head> și orice stil declarat într-un fișier extern CSS.

În continuare ne vom ocupa de sintaxa declarării unui stil și de tipurile de stiluri

3.7 Sintaxa CSS

Sintaxa CSS este alcătuită din 3 părți:

- **un selector**
- **o proprietate**
- **o valoare atribuită proprietății**

Mod de redactare:

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

Selectorul este de obicei chiar elementul/tag-ul HTML pe care dorim să-l definim sau un șir de caractere ales de noi

Proprietatea este atributul elementului HTML pe care

dorim să-l modificăm: culoare, mărimea caracterului, margini, borduri,

Valoarea proprietății este chiar valoarea pe care dorim să o atribuim atributului elementului HTML

Trebuie să reținem faptul că în sintaxă proprietatea este separată de valoare prin “două puncte”. Dacă o valoare este compusă din mai multe cuvinte atunci în cadrul sintaxei acestea se pun între ghilimele. Dacă se dorește stabilirea mai multor proprietăți pentru un singur selector acestea se vor separa cu “punct și virgulă”

Exemple:

Ex1:

```
body {color: red}
```

Exemplul 1 stabilește pentru tag-ul <body> atributul color cu valoarea red adică textul delimitat de acest tag se va scrie cu culoarea roșie.;

Ex2

```
p {font-family: "sans serif"}
```

Exemplul 2 stabilește pentru tag-ul de paragraf <p> atributul font-family cu valoarea “sans serif” adică textul delimitat de acest tag va fi scris utilizând fontul “sans serif”;

Ex3

```
p{    font-family: verdana;
      color: green;
      text-align: right }
```

Exemplul 3 stabilește mai multe proprietăți pentru tag-ul `<p>`. Observăm că acestea sunt separate de „;”.

3.7.1 Tipuri de selectori

Există mai multe tipuri de selectori:

1. Selector universal
2. Selector HTML
3. Selector de tip Class
4. Selector ID
5. Selector atribut
6. Pseudo clase
7. Pseudo elemente
8. Gruparea selectorilor
9. Selector nod copil
10. Selector dependent de context
11. Selector noduri adiacente
12. Selector noduri cu părinte comun

3.7.1.1 *Selector universal* `*{}`

Selectorul universal `*{}` este folosit pentru a selecta toate elementele. Acesta este marcat cu `*`.

Exemplu:

```
* {
font-size: 25px;
}
```

Acest selector formatează toate elementele HTML

stabilind pentru acestea proprietatea *font-size* dimensiunea fontului de 25px.

3.7.1.2 *Selector HTML*

Selectorii de acest tip au numele tagurilor HTML folosite în marcarea unui document HTML de exemplu p, h1, div,... .

De exemplu:

```
p {  
  font-size: 25px;  
}
```

Utilizând această regulă tot textul marcat cu tagul <p> va fi afișat cu dimensiunea caracterului de 25px.

3.7.1.3 *Selector de tip Class*

.ClassSelector {propietate:valoare;}

Acest tip de selector este un selector specific și se va aplica doar tagurilor care primesc ca și valoare pentru atributul class numele selectorului. Denumirea selectorului este una aleasă de programator și care începe cu caracterul „.”

Exemplu:

```
.tilup {  
  font-family:arial;  
  font-size:14px;  
  color:red;  
}
```

Pentru a aplica acest stil unui tag trebuie să fie specificat atributul `class="titlup"`.

Exemplu:

```
<p class="titlup"> Text important</p>  
<p> Un paragraf care va avea o margine de 30px din  
stanga</p>
```

În exemplul de mai sus primul paragraf primind atributul `class` cu valoarea `titlup` va fi formatat cu font Arial, dimensiune 14px și culoarea fontului roșu. Al doilea paragraf va fi afișat cu setările implicite ale browserului.

Alte exemple de aplicare a selectorului de tip `class`:

Dacă am dori să avem două tipuri de paragrafe: unul aliniat la centru iar celălalt aliniat la dreapta am putea proceda astfel:

Pas 1

Ne declarăm ambele stiluri cu ajutorul unui selector de tip clasă astfel:

```
.dreapta {text-align: right}  
.centrat {text-align: center}
```

Aceste clase (le vom numi astfel în continuare) vor trebui aplicate ca și atribute paragrafelor. Afișarea ar arăta astfel:

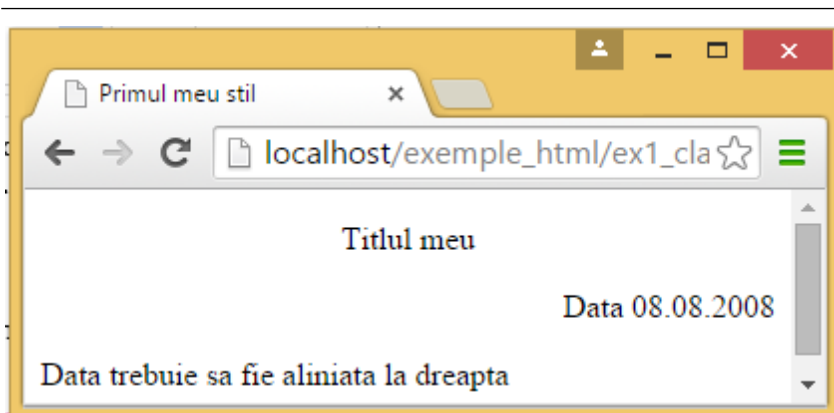


Fig. 56. Aplicarea selectorului de tip class

Codul sursă ar arăta astfel:

```
<!DOCTYPE html>
<html>
<head>
<title>Primul meu stil</title>
<style type="text/css">
    .centrat {text-align:center}
    .dreapta{text-align:right}
</style>
</head>
<body>
<p class="centrat">Titlul meu</p>
<p class="dreapta">Data 08.08.2008 </p>
<p>Data trebuie sa fie aliniata la dreapta</p>
</body>
</html>
```

Ceea ce este foarte important este faptul că ulterior se mai pot face modificări iar acestea se vor face în toate paragrafele care au ca și atribut clasa respectivă. ATENȚIE: Un element HTML poate primi un singur atribut de tip class.

3.7.1.4 Selector ID

#IDSelector {proprietate:valoare;}

Selectorul de tip ID este precedat de caracterul „#” și se aplică tagurilor folosind atributul id. Mulți programatori de pagini de web îl folosesc ca și pe cel de tip class. Totuși între cei doi selectori există o diferență. Dacă selectorul de tip class poate fi folosit de multe ori în pagina de web, selectorul de tip ID trebuie să fie unic, trebuie să fie un identificator unic pentru un element din pagină. De obicei selectorul de tip ID se folosește în special la identificarea diviziunilor într-o pagină de web.

Exemplu:

```
#Elementul1 {  
    border: 1px solid blue;  
}
```

Exemplu de utilizare în fișier HTML:

```
<!DOCTYPE html>  
<html>  
<head>  
<title>Selector class</title>  
<style type="text/css">  
#Elementul1 {  
    border: 1px solid blue;  
}  
</style>  
</head>  
<body>  
<div id="Elementul1">Titlul meu</div>  
</body>  
</html>
```

Va afișa:

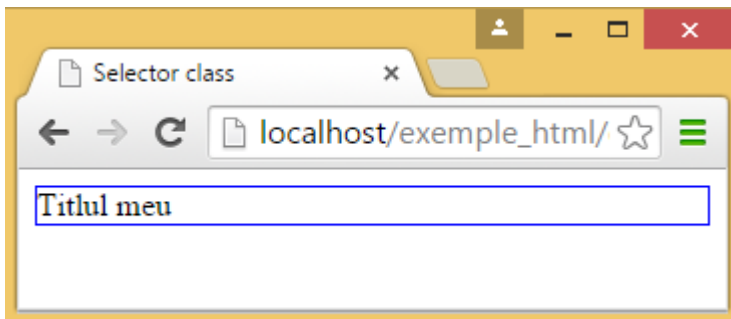


Fig. 57. Utilizare selector ID

3.7.1.5 Selector atribut

[atribut]{proprietate:valoare;}

Acest selector este un selector care identifică elementele HTML după atributele pe care le au. Pentru a aplica acest selector în taguri nu trebuie făcute alte modificări.

Exemplu:

```
[href] {  
    color: red;  
    font-size: 30px;  
}
```

Tagul/tagurile care conține atributul *href* va fi formatat conform proprietăților specificate.

```
<!DOCTYPE html>  
<html>  
<head>
```

```

<meta charset="utf-8">
<title>Selector class</title>
<style type="text/css">
[href] {
    color: red;
    font-size:30px;
}
</style>
</head>
<body>
<a href="http://csac.ulbsibiu.ro"> Departamentul de
Calculatoare și Inginerie Electrică</a>

<a name="profil">Cadre didactice:</a>
<br>
</body>
</html>

```

Va afișa:

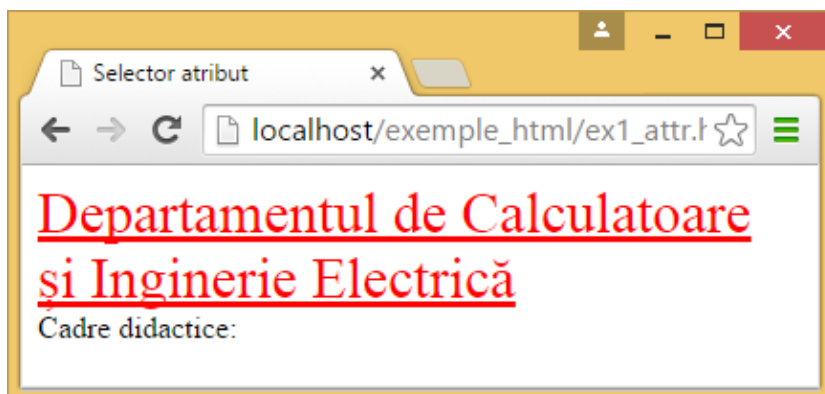


Fig. 58. Utilizare selectorului atribut

În continuare vom prezenta câteva variante de a utiliza

mai specific ”selector atribut”.

3.7.1.5.1 Atribut egal cu ...

Un element HTML poate fi selectat pe baza valorii atributului ca de exemplu:

```
[href="http://csac.ulbsibiu.ro"] {  
text-decoration: none;
```

În acest caz toate tagurile a căror atribut href sunt egale cu `http://csac.ulbsibiu.ro` vor fi scrise nesubliniat.

3.7.1.5.2 Atribut care începe cu ...

Cu acest selector poate fi ales un tag care are un atribut a cărui valoare începe cu un anumit șir de caractere. Astfel în locul utilizării semnului `=` se va folosi `^=`.

Exemplu:

```
<style type="text/css">  
[href^="http://"] {  
                                color: blue;  
                                font-size: 30px;  
                                }  
</style>  
</head>  
<body>  
<a href="http://csac.ulbsibiu.ro"> Departamentul de  
Calculatoare și Inginerie Electrică</a>  
<a href="http:// ulbsibiu.ro"> Universitatea  
ULSB</a>
```

În acest exemplu toate tagurile HTML care conțin atributul href și care începe cu `http://` vor fi afișate conform

proprietăților.

3.7.1.5.3 Atribut care se termină cu ...

Pentru a selecta un atribut care se termină cu un anumit șir de caractere se folosește \$= între numele atributului și valoare

Exemplu:

```
<style>
  [src$="t.jpg"] {
    font-size: 18px;
  }
</style>


```

În acest caz se vor selecta tagurile ale căror valori ale atributelor src se termină în t.jpg

3.7.1.5.4 Atribut care conține ...

Pentru identificarea unui tag a cărui valoare pentru un anumit atribut conține un șir de caractere se folosește *=

Exemplu:

```
[src*="radu"] {
  font-size: 18px;
}
```

Acest selector va modifica doar textul cuprins între acele taguri care conțin atributul src și în valoarea cărora se găsește șirul radu.

3.7.1.5.5 Atributul conține cuvântul ...

Exemplu:

```
[atr_meu~="mere"] {  
font-size: 18px;  
}
```

în secvența de cod:

```
<div atr_meu="meremere"></div>  
<div atr_meu ="doua mere"></div>
```

va alege al doilea div.

3.7.1.5.6 Atribut care începe cu codul de limbă ...

Acest selector este de obicei utilizat pentru a selecta taguri cu un anumit cod de limbă:

```
<p lang="en" > Text englez</p>  
<p lang="en-UK"> Text engleza britanica</p>  
<p lang="de"> text in germana</p>
```

Pentru a selecta elementele HTML care la codul de limbă au valoarea en sau en- folosim selectorul atribut astfel:

```
[lang|="en"] {  
font-size: 18px;  
}
```

3.7.1.6 Pseudo clase

Ce sunt astea 2 cuvinte

3.7.1.6.1 Pseudoclasa: `a:link`, `a:visited`, `a:hover`, `a:active`

Aceste clase se numesc pseudo clase pentru că proprietățile se referă la același tag aflat în ipostaze diferite. De exemplu linkul poate fi afișat când nu a fost accesat cu o anumită formatare, după ce a fost accesat cu alta, când mouse-ul se află deasupra link-ului cu alta.

`a:link`

Definește stilul pentru link-uri nevizitate;

`a:visited`

Definește stilul pentru link-uri vizitate;

`a:active`

Definește stilul pentru link-uri active.

Link-ul devine activ la click;

`a:hover`

Definește stilul pentru link-ul când mouse-ul se află deasupra sa.

Exemplu:

```
<style type="text/css">
```

```
a:link {text-decoration: none}
```

```
a:visited {text-decoration: none}
a:active {text-decoration: none}
a:hover {text-decoration: underline overline;color:
red;}

</style>
```

La începuturile css-ului doar tagul <a> a avut aceste 4 pseudo clase. În CSS3 și alte taguri acceptă aceste pseudo clase. Cu ajutorul acestor pseudo clase se pot realiza foarte ușor meniuri de navigare astfel încât putem renunța la meniurile tip flash sau javascript care sunt ignorate de către motoarele de căutare.

3.7.1.6.2 Pseudoclasa: first-child, :last-child

Scrie un rand despre asta

Exemplu:

```
<style>
p:first-child {
    background-color:red;
}
p:last-child {
    background-color:green;
}
</style>

<div>
    <p>Primul nod copil al tagului div</p>
    <p>Al doilea nod copil al tagului div</p>
    <p>Al treilea nod copil al tagului div</p>
```

```
</div>
```

În acest exemplu primul paragraf va fi afișat cu culoarea roșie iar al ultimul paragraf va fi afișat cu culoarea verde.

3.7.1.6.3 Pseudoclasa: nth-child()

Scrie un rand despre asta

Exemplu:

```
<style>
p:nth-child(2n) {
    background-color: yellow;
}</style>
```

Acest selector va alege tagurile copil pare.

```
<div>
  <p>unu</p>
  <p>doi</p>
  <p>trei</p>
  <p>patru</p>
  <p>cinci</p>
  <p>sase</p>
</div>
```

Va afișa:

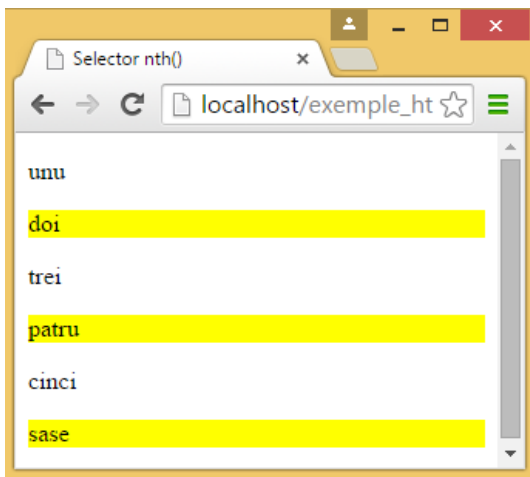


Fig. 59. Utilizarea pseudo clasei :nth-child()

3.7.1.6.4 Pseudoclasa: first-of-type, :last-of-type

Se utilizează când dorim să accesăm un element din structura documentului de un anumit tip. Se exemplul primul paragraf sau ultimul paragraf.

Exemplu:

```
<style>
p:first-of-type {
    background-color: #ffff00;
}
p:last-of-type {
    background-color: #00ff00;
}

<div>
    <h2>primul h2</h2>
    <p>primul p</p>
```

```
<h2>al doilea h2</h2>
<p>al doilea p</p>

<h2>al treilea h2</h2>
<p>al treilea p</p>
</div>
```

Va afișa:

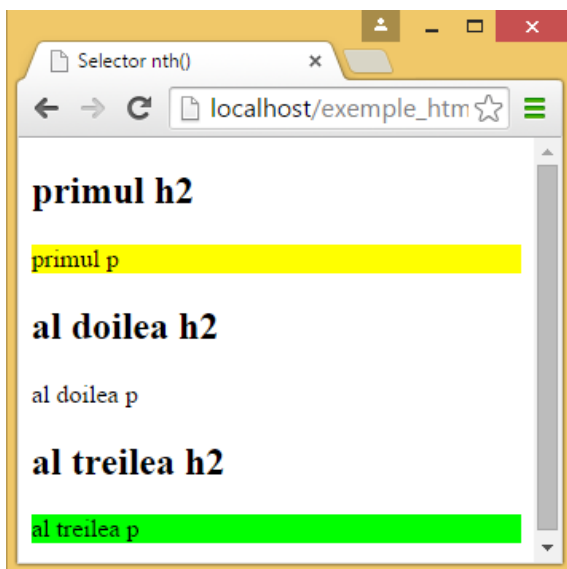


Fig. 60. Utilizarea :first-of-type, :last-of-type

3.7.1.6.5 Pseudoclasa: nth-of-type()

Se poate alege elementul n de un anumit tip. De exemplu toate paragrafele pare (al 2-lea, al 4-lea...)

Exemplu:

```
p:nth-of-type(2n) {  
    background-color: #ffff00;  
<div>  
    <p>unu</p>  
    <p>doi</p>  
    <p>trei</p>  
    <p>patru</p>  
    <p>cinci</p>  
    <p>sase</p>  
</div>
```

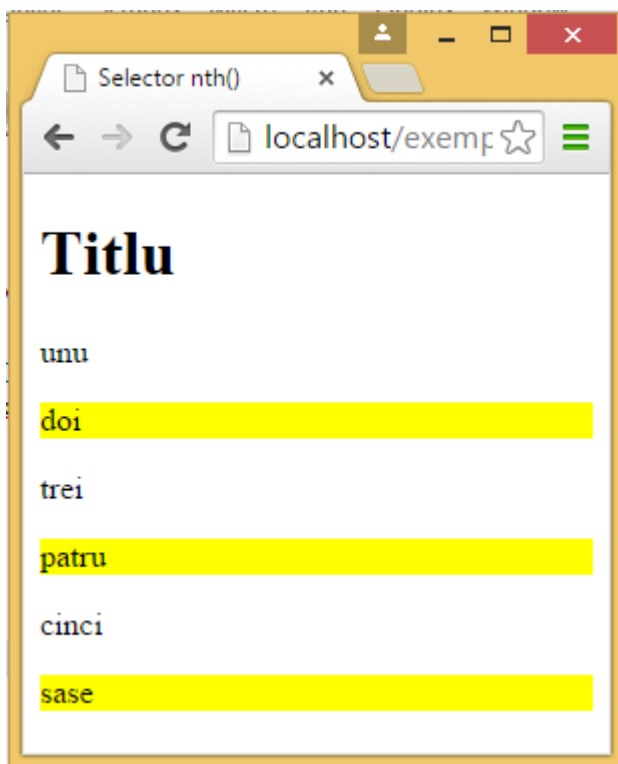


Fig. 61. nth-of-type()

3.7.1.6.6 Pseudoclasa: nth-last-child()

Similar cu :nth-child().

3.7.1.6.7 Pseudoclasa: nth-last-of-type()

Similar cu :nth-of-type () cu diferența ...că se aplică la ultimul.....

3.7.1.6.8 Pseudoclasa: only-child

Pseudo clasa :only-child selectează toate tagurile care sunt unici copii ai părintelui.

3.7.1.6.9 Pseudoclasa: only-of-type()

Pseudo clasa *only-of-type()* va selecta unicul copil de un anumit tip al unui tag părinte.

Exemplu:

```
p:only-of-type {  
    background: #ff0000;  
}  
div{border:dashed 1px red;  
    }  
  
<div>  
    <p>Un paragraf.</p>
```

```
</div>
<div> <p>Alt paragraf.</p>
      <p>Inca un paragraf.</p>
</div>
```

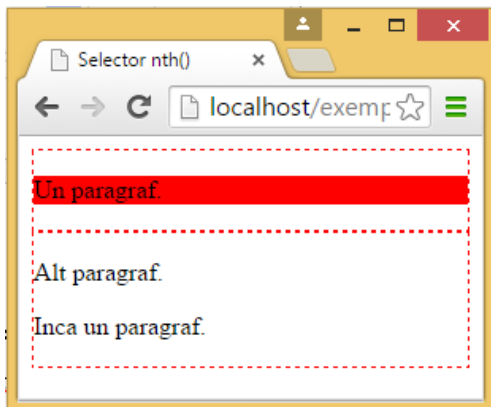


Fig. 62. :only-of-type

3.7.1.6.10 Pseudoclasa: empty

Selectează tagurile care nu delimitează nimic.

3.7.1.6.11 Pseudoclasa: not()

Se folosește ca operator de negare pentru regulile stabilite al tuturor celorlalte pseudo clase. De exemplu regula stabilită mai jos NU se aplică primului copil paragraf.

Exemplu:

```
p:not(:first-child) {
  border-bottom: 1px solid #ff0000;
}
```

3.7.1.6.12 Pseudoclasa: checked

Selectează elementele care au caracteristica checked selectată

Exemplu:

```
input[type="checkbox"] :checked {  
}
```

Selectează elementele input de tip checkbox care au fost bifate (care au ca valoarea checked).

3.7.1.6.13 Pseudoclasa: enabled, :disabled

Selectorul :enabled se utilizează pentru a aplica un stil unui element activ (utilizat în principal pe elemente de formular).

Exemplu:

```
input:enabled {  
    border: 1px dashed #6666ff;  
}  
input:disabled {  
    border: 1px dotted #666666;  
}
```

Selectează toate elementele input care sunt fie enabled fie disabled.

3.7.1.7 Pseudo elemente

Pseudo-elementele sunt identificatori care se aplică

textului delimitat de tag-uri.

3.7.1.7.1 Pseudo element: ::first-letter

Va selecta prima literă din textul unui tag care va fi afișat în fereastra browserului conform cu proprietatea specificată.

Exemplu:

```
p::first-letter {  
    font-size : 30px;  
    color:red;  
}
```

Dacă sintaxa este aplicată la codul

```
<p>Un paragraf.</p>  
<p>Alt paragraf.</p><p>Inca un paragraf.</p>
```

Va afișa:

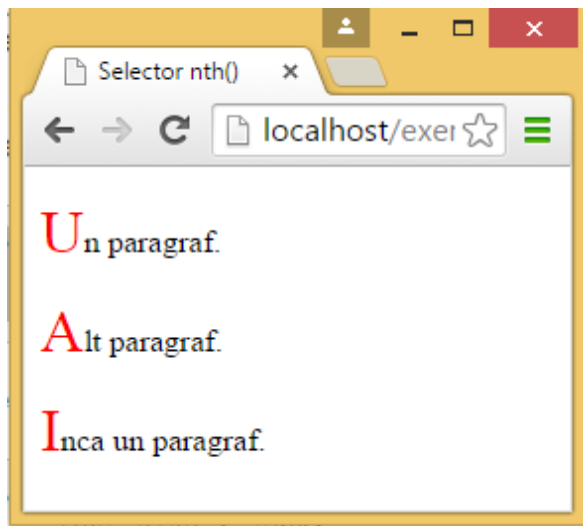


Fig. 63. ::first-letter

3.7.1.7.2 Pseudo-element: ::first-line

Va selecta prima linie din textul unui tag care va fi afișat în fereastra browserului conform cu proprietatea.

Exemplu:

```
p::first-line {  
    font-size : 20px;  
    background-color:red;  
}
```

<p>Text Text Text.Text Text Text.Text Text Text.Text
Text Text.Text Text Text.Text Text Text.</p>
<p>Alt paragraf.</p><p>Inca un paragraf.</p>

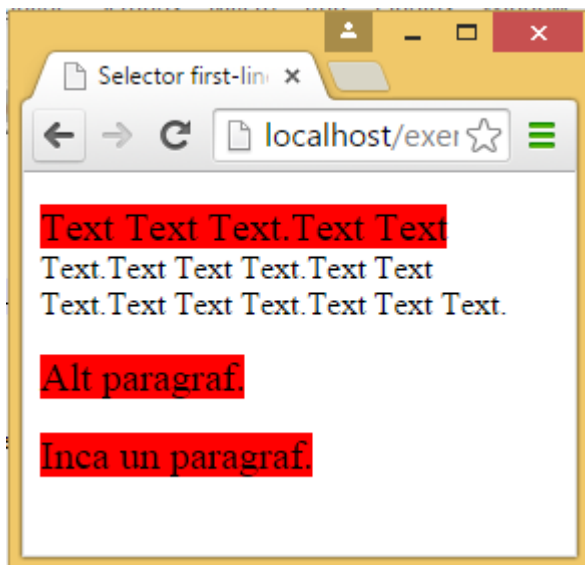


Fig. 64. ::first-line

3.7.1.7.3 Pseudo-element: ::before, ::after

Pseudo elementele CSS ::before și::after vor selecta un prim sau ultim copil virtual al elementului HTML selectat. Acest lucru este în mod normal folosit pentru a insera conținut suplimentar (text sau HTML).

Exemplu:

```
#Idul::after {  
    content : "--Aici am generat conținut cu  
ajutorul CSS";  
}
```

```
<div id="Idul">  
    Text din div  
</div>
```

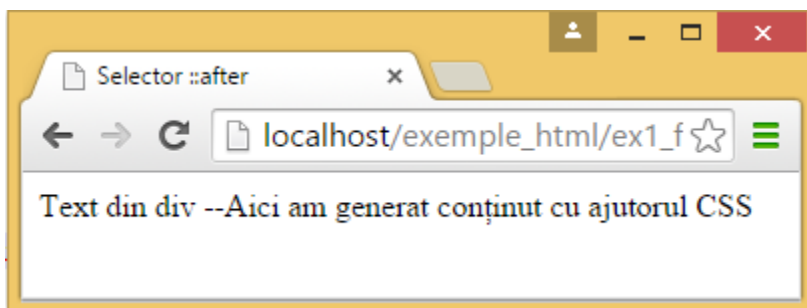


Fig. 65. ::after

Utilizarea pseudo elementului ::before este similară doar că va insera înainte de elementul selectat.

3.7.1.8 Gruparea selectorilor

Selectorii pot fi grupații, separarea lor se face cu ajutorul caracterului „,” virgulă.

Exemplu:

```
h1,h2,h3 {  
    color: blue  
}
```

Acest stil stabilește faptul că textul delimitat de cele 3 tag-uri <h1>,<h2> și <h3> se vor scrie cu culoare albastră.

3.7.1.9 Selector nod copil

Selectorul copil este folosit pentru a selecta toate elementele care sunt elemente copil ale unor taguri părinți

Exemplu:

```
li>a {  
    font-size: 18px;  
    color:red;  
}
```

Acest selector va selecta toate elementele a care sunt elemente copil ale elementelor li din următorul cod:

```
<a href="#">Linkul nu va fi selectat</a>  
  
<ul>  
  <li><a href="#">Aici modific linkul</a></li>  
</ul>
```

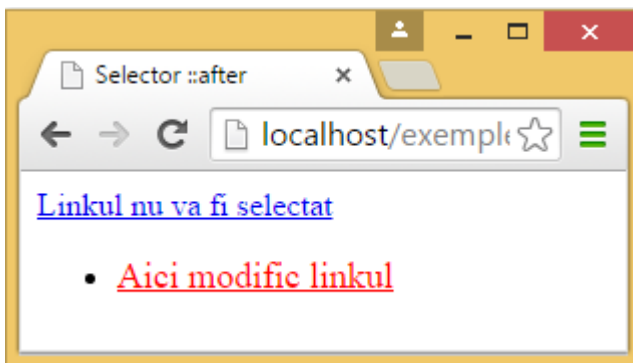


Fig. 66. Selector copil

Pot fi folosite mai multe niveluri de tip selector copil.

Exemplu:

```
ol>li>a {  
    font-family: verdana;  
}
```

În acest caz doar elementele `a` care sunt copii ale elementului `li` care la rândul său este copii al unui tag `ol` vor fi afișate cu font `verdana`.

3.7.1.10 Selector dependent de context

Selectorul dependent de context sau descendent este folosit pentru a selecta elemente care sunt descendenți ai altor elemente. Elementele selectate nu trebuie să fie elemente copil imediate ale tagului părinte specificat. Ele trebuie doar să fie

imbricate în interiorul unui tag strămoș undeva.

Exemplu

```
ol a {  
    font-size : 18px;  
}
```

Selectorul dependent de context este marcat prin utilizarea caracterului „spațiu” între selectori. În exemplul de mai sus folosind selectorul descendent am putut renunța la scrierea tuturor copiilor până la tagul a

3.7.1.11 Selector noduri adiacente

Acest selector este folosit în selectarea unor taguri adiacente de același nivel față de tagul părinte.

Exemplu:

```
<h1>Titlu</h1>  
<p>... </p>  
<p>... </p>  
<table>...</table>  
<table>...</table>
```

În exemplul prezentat taguri adiacente sunt tagurile <h1> și <p> pentru că sunt copii de același nivel și între ele nu mai sunt alte taguri. Pentru a le modifica poate fi folosit selectorul adiacent:

```
h1+p {  
  font-size : 20px;  
}
```

Similar am putea alege tagul <p> urmat de <table>:

```
p+table {  
  font-size : 20px;  
}
```

3.7.2 *Proprietățile CSS pentru background*

Aceste proprietăți definesc modul de afișaj al fundalului unui element html.

În tabelul de mai jos în coloana a doua sunt trecute cu *italic* valorile explicite care pot fi atribuite proprietăților din prima coloană.

Proprietate	Valoare	Scurtă descriere
background	<i>background-color</i> <i>background-image</i> <i>background-repeat</i> <i>background-attachment</i> <i>background-position</i>	Proprietate cu care se pot seta toate valorile de background;

background-attachment	scroll fixed	Stabilește faptul că o poză de pe fundal rămâne fixă sau se mișcă odată cu conținutul;
background-color	<i>color-rgb color-hex color- nume transparent</i>	Setează culoarea de fundal a unui element HTML;
background-image	<i>url() none</i>	Setează o imagine pe fundalul elementului HTML;
background-position	<i>top left top center top right center left center center center right bottom left bottom center bottom right x-% y-% x-pos y-pos</i>	Setează locul de unde începe poziționarea pozei de pe fundal;
background-repeat	<i>repeat repeat-x repeat-y no-repeat</i>	Setează modul cum/dacă se repetă o imagine pe fundal;

Example:

1. Stabilirea unor fundaluri.....

Codul sursă:

```
<!DOCTYPE html>
<html>
<head>
```

```
<meta charset="utf-8">
<title>Background</title>

<style type="text/css">
body{background-color:#CCCC00}
.centrat {text-align:center}
p.dreapta{text-align:right}
h1{background-color:#3300FF}
</style>

</head>
<body>
<p class="centrat">Titilul meu</p>
<p class="dreapta">Data 08.08.2008 </p>
<p>care trebuie sa fie aliniata la dreapta
</p>
<h1 class="centrat">un bla bla declarat cu h1 scris
centra</h1>
</body>

</html>
```

În browser arată astfel: (nu am ales culori care să se “potrivească”).

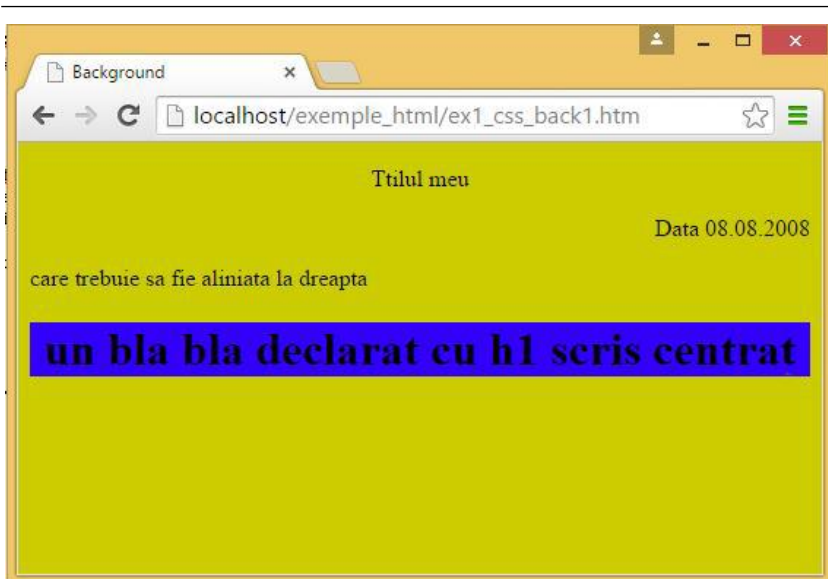


Fig. 67. Proprietatea background

2. Imagine pe fundal și repetarea ei doar pe axa x

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Background</title>
<style type="text/css">
    body {
        background-image:url (img/smiley.png) ;
        background-repeat: repeat-x
    }
</style>
</head>
<body>
<p> Text peste fundal</p>
</body>
</html>
```

Arată astfel:

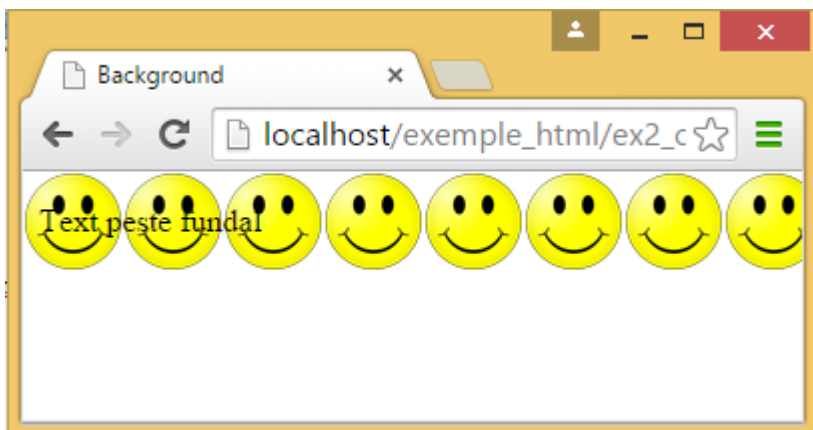


Fig. 68. Imagine pe fundal și repetarea ei doar pe axa x

3.7.3 Proprietățile CSS pentru text

Cu aceste proprietăți se poate modifica: modul cum se afișează textul, distanța dintre caractere, dintre cuvinte, etc.

Proprietate	Valori	Descriere
color	<i>color</i>	Setează culoarea unui text
direction	<i>ltr rtl</i>	Setează direcția de scriere <i>ltr</i> =de la stânga la dreapta <i>rtl</i> =de la dreapta la stânga
letter-spacing	<i>normal length</i>	Mărește sau micșorează spațiul dintre litere

text-align	<i>left right center justify</i>	Aliniază textul dintr-un element html
text-decoration	<i>none underline overline line-through blink</i>	Adaugă „decorări” textului
text-indent	<i>Lungime px %</i>	Indentează prima linie dintr-un element HTML
text-transform	<i>none capitalize uppercase lowercase</i>	Controlează modul de afișare a literelor unui text: cu MAJUSCULE
word-spacing	<i>normal length</i>	Mărește sau micșorează spațiul dintre cuvinte

Exemplu:

1. Mărirea spațiului dintre litere: în pixeli (sau cm sau pt)

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Background</title>
<style type="text/css">
    h1 {letter-spacing:10px}
    h2 {letter-spacing:1cm}
</style>
```

```

</head>
<body>
    <h1>Primul titlu</h1>
    <h2>Al doilea titlu</h2>
</body>
</html>

```

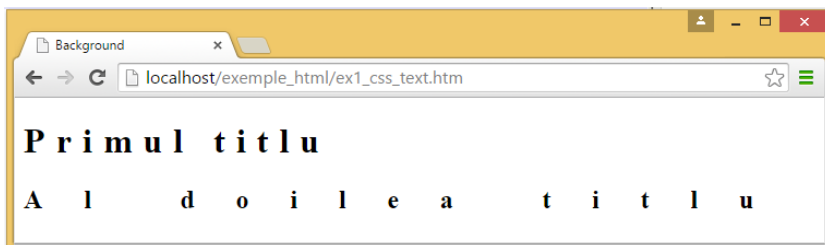


Fig. 69. Utilizarea proprietății letter-spacing

3.7.4 Proprietățile CSS pentru fonturi

Proprietate	Valori	Descriere
-------------	--------	-----------

font	<i>font-style</i> <i>font-variant</i> <i>font-weight</i> <i>font-size/</i> <i>line-height</i> <i>font-family</i> <i>caption icon</i> <i>menu</i> <i>message-box</i> <i>small-caption</i> <i>status-bar</i>	O proprietate cu care se pot defini într-o singură declarație toate proprietățile fontului
font-family	<i>family-name generic-family</i>	O listă de nume de fonturi care se vor utiliza dacă există pe mașina pe care rulează browserul în ordinea stabilită
font-size	<i>xx-small</i> <i>x-small</i> <i>small</i> <i>medium</i> <i>large</i> <i>x-large</i> <i>xx-large</i> <i>smaller</i> <i>larger</i> <i>mărime %</i>	Setează dimensiunea fontului

font-stretch	<i>normal wider narrower ultra-condensed extra-condensed condensed semi-condensed semi-expanded expanded extra-expanded ultra-expanded</i>	Specifică modul de cât de condensat este scris textul
font-style	<i>normal italic oblique</i>	Setează stilul de scriere
font-variant	<i>normal small-caps</i>	Afișează textul cu “CAPITALE MICI” sau normal.
font-weight	<i>normal bold bolder lighter 100 200 300 400 500 600 700 800 900</i>	Setează grosimea fontului

Exemplu:

1. Setarea tipului de font, fel, mărime

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Font</title>
  <style type="text/css">
    p{font-family:Verdana, Arial, Helvetica,
      sans-serif;
      font-variant:small-caps;
      font-size:small
    }
  </style>
</head>
<body>
  <p> Textul Meu</p>
</body>
</html>
```

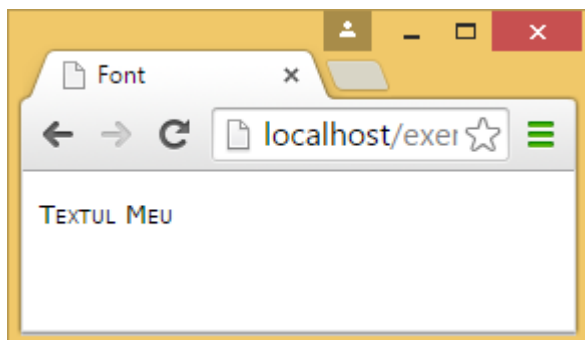


Fig. 70. Modificarea fontului

3.7.5 Proprietățile CSS pentru tabele

Proprietate	Valori	Descriere
width, height	px pt	dimensiunea lățime și înălțime
margin margin-top margin-right margin-bottom margin-left	px pt	distanța de la elementul superior
border		Aici nimic de scris – stergem atunci
border-width	px pt	grosimea bordurii
border-style	solid dotted dashed double groove ridge inset outset none hidden	tipul bordurii
border-color	hexa rgb()	culoarea bordurii
border-top border-right border-bottom border-left border-top-width border-top-style border-top-color border-right-width		specificarea anumitor borduri

border-right-style border-right-color border-bottom-width border-bottom-style border-bottom-color border-left-width border-left-style border-left-color		
border-radius border-top-left-radius border-top-right-radius border-bottom-left-radius border-bottom-right-radius	px	colțuri rotunjite
text-align	start end left right center justify	alinierea textului
vertical-align	top middle bottom	Alinierea verticală a textului într-o celulă
border-spacing	px	Stabilește distanța dintre celulele unui tabel
border-collapse	separate collapse	Stabilește distanța dintre celulele unui tabel

Proprietățile CSS sunt mereu în schimbare și dezvoltatorii browserelor adaugă tot mai mult suport pentru aceste proprietăți. Prezenta lucrare nu își propune să fie una exhaustivă în ceea ce privește sintaxa CSS.

4 PHP

De la apariția sa WEB-ul a parcurs 2 mari etape: WEB-ul static (WEB 1.0) – în care utilizatorii citeau doar informația conținută în paginile web și care eventual era animată cu ajutorul limbajului JavaScript și WEB-ul dinamic (WEB 2.0) care a apărut la începutul anilor 2000 când au apărut primele bloguri și rețelele de socializare care au permis interacțiunea useri-lor cu acest mediu. Astfel au apărut limbajele de script care se execută pe servere și codul executat nefiind accesibil utilizatorilor. Astfel de limbaje de script sunt: php, jsp, asp, ruby, python ș.a.

În această lucrare ne vom concentra asupra modelului php – MySQL care împreună cu serverul de web Apache este un trio gratuit și extrem de des utilizat pentru realizarea paginilor de web dinamice și găzduirea acestora. Nu vom prezenta exhaustiv limbajul php și ne vom concentra doar asupra unor instrucțiuni de bază necesare la crearea paginilor de web dinamice.

Scripturile php incluse în codul html sunt executate pe server. Asta are ca efect faptul că dacă încercați să vizualizați codul sursă al unei pagini web dinamice o să observați că el conține doar tag-uri html, acestea fiind rezultatul obținut în urma interpretării codului php.

PHP (se pronunță în română pe-haș-pe) este un limbaj

de programare ce rulează pe server, proiectat special pentru WEB și a fost conceput în anul 1994. Acesta permite ca într-o pagina HTML să puteți îngloba cod PHP care va fi executat la fiecare cerere a paginii.

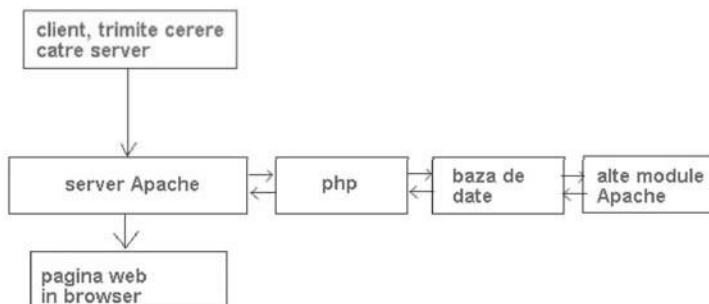


Fig. 71. Schema de interpretare a codului php

Codul PHP este interpretat pe serverul WEB și generează un cod HTML care va fi văzut de utilizator (clientul), browserului fiindu-i transmis numai cod rezultat ca și HTML.

4.1 Sintaxa php

Un fișier php conține tag-uri normale de html și cod php propriu. Marcarea începutului codului php se face cu tagul „<?php” iar sfârșitul blocului de cod php cu tagul „>”.

Codul pe care îl scriem între cele două taguri seamănă foarte mult cu cod scris în limbajul C++.

Exemplu:

Clasicul exemplu cu „Hellou lume” transformat în script php:

```
<html>
<body>
<?php echo "Hellou lume"; ?>
</body>
</html>
```

„echo” este o instrucțiune din php care permite afișarea unui text în browser la poziția curentă

Observații:

Fișierul de mai sus trebuie salvat cu extensia php pentru ca severul de web să transmită fișierul acesta interpretorului php.

Rezultatul afișat în browser este:

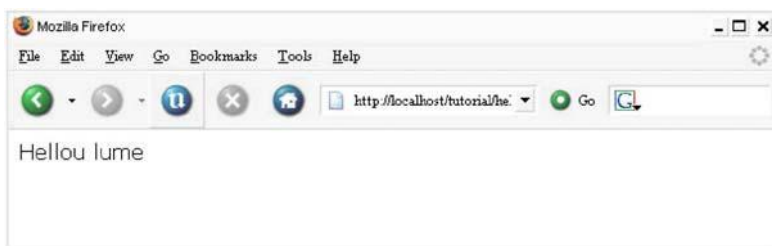


Fig. 72. Hellou lume - php

iar codul sursă returnat (vizibil de altfel cu ajutorul comenzii View Source din browser) este următorul:

```
<html>
<body>Hellou lume</body>
</html>
```

Un bloc de script php poate fi inserat în orice loc a unui document html. Fiecare instrucțiune php trebuie separată de o alta instrucțiune cu ajutorul caracterului „;”.

4.2 Afișarea de text în php

4.2.1 Afișarea cu ajutorul lui “echo”

În realizarea paginilor dinamice avem față de programarea clasică o problemă în plus și anume trebuie avut în vedere că afișarea (scrierea) trebuie făcută într-un fișier care apoi este trimis utilizatorului prin intermediul serverului de web. Practic pe server se găsesc doar niște scripturi care la apelare generează de fiecare dată conținut html.

Pentru a afișa ceva în PHP (adică codul html pentru o pagină) se pot utiliza mai multe funcții: **echo**, **print** și **printf**. Funcția **printf()** se utilizează asemănător ca în C în sensul că permite formatarea afișărilor. Noi vom folosi funcția **echo** care este puțin mai rapidă ca **print**. Diferența dintre *echo* și *print* este ca *print* returnează întotdeauna valoarea 1 după ce a fost executată și deci poate fi utilizată în construcții mai complexe.

Exemplu:

```
<?php
echo "Acesta este textul care se va afiș a";
?>
```

După cum se observă lipsesc parantezele specifice funcțiilor. Asta se întâmplă deoarece echo nu este o funcție ci doar un macrou al limbajului PHP. În schimb dacă am folosi printf(), aceasta este o funcție, deci atunci trebuie utilizate parantezele.

Exemplu:

```
<?php  
echo "Acesta este textul care se va afiș a";  
echo 'Acesta este al doilea text care se va  
afiș a';  
?>
```

Dacă salvăm acest fișier și îl uploadăm pe server, obținem următorul rezultat:

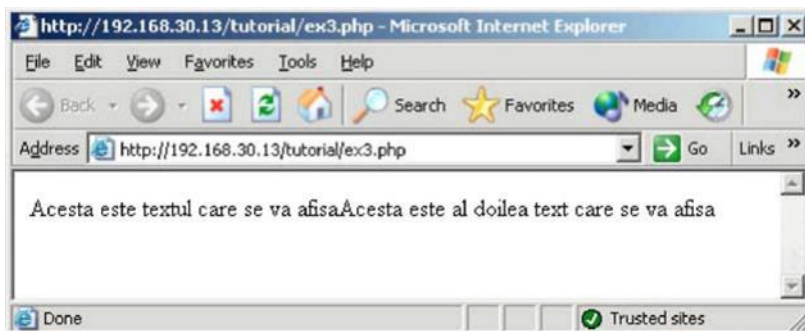


Fig. 73. Output neformatat

Pagina generată de php este:

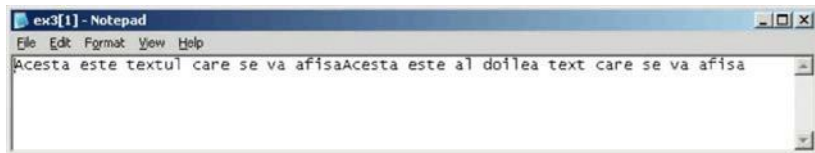


Fig. 74. Codul sursă generat

Observăm că pagina noastră conține doar text simplu fără niciun tag html. Deocamdată pentru simplificare nu am introdus și structura standard HTML5.

Echo poate fi folosit cu ghilimele simple sau duble. Diferența dintre cele două posibilități este că în cazul utilizării ghilimelei duble dacă în șirul destinat afișării se găsește o variabilă atunci aceasta este înlocuită cu valoarea ei.

Observăm că al doilea string a fost adăugat primului fără salt la rând nou cu toate că am avut două instrucțiuni `echo` pe rânduri diferite în codul php. Problema este că după afișarea primului string cursorul imaginar al PHP-ului a rămas după primul string, al doilea echo fiind afișat începând cu aceea poziție.

Pentru a avea un salt la linie nouă trebuie să utilizăm elemente de formatare.

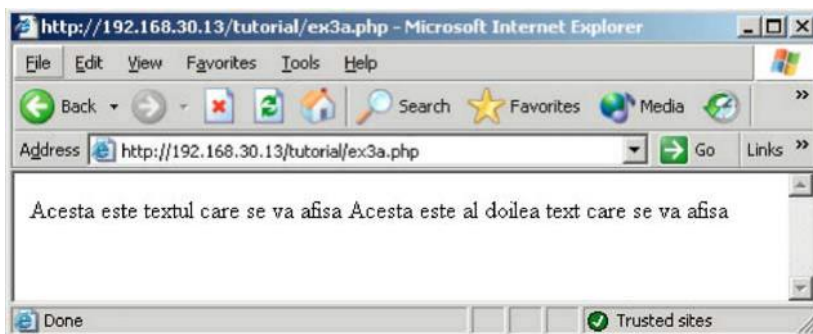
4.2.2 Elemente de formatare

Pentru a forța salt la rând nou în php utilizăm caracterul `\n`. Saltul la rând nou este vizibil doar în sursa generată de php. În browser nu va avea nici un effect.

Exemplul:

```
<?php  
echo "Acesta este textul care se va afi$ a\n";  
echo "Acesta este al doilea text care se va  
afi$ a";  
?>
```

Rezultatul în browser este:



Adică cam tot ca la exemplul anterior. Totuși dacă vedem codul generat de php observăm modificarea.

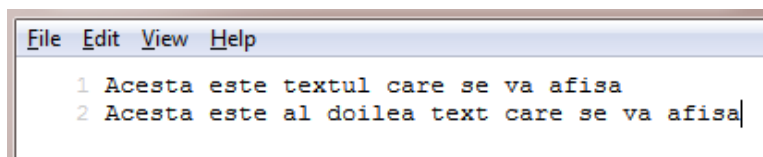


Fig. 75. Sursă formatată cu \n

În principiu utilizăm elementele de formatare pentru

lizibilitatea codului.

Totuși pentru a putea afișa și în fereastra browser-ului cele două string-uri pe două rânduri trebuie să adăugăm și taguri html pe care le vom trata în php ca fiind simple șiruri de caractere și elemente html.

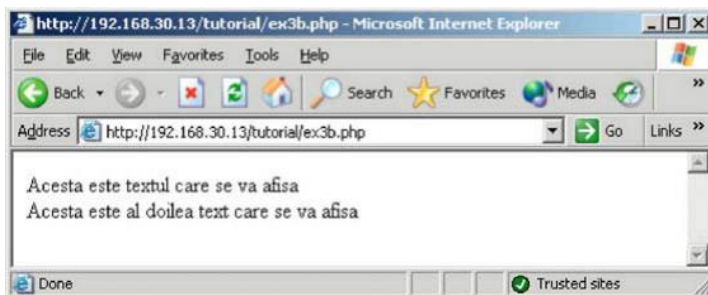
4.2.3 Afișarea elementelor html cu ajutorul lui php

Pentru a forța un salt la rând nou utilizăm elementul html

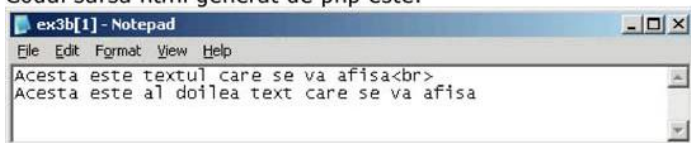
Exemplul 4

```
<?php
echo  "Acesta este textul care se va
afiș a<br>\n";
echo  "Acesta este al doilea text care se va
afiș a"; ?>
```

Rezultatul este:



Codul sursa html generat de php este:



4.2.4 Afișarea caracterelor speciale

Ce vrea să spună este faptul că anumite caractere speciale cum sunt ghilimelele, slash-ul etc. sunt folosite ca și coduri de către interpretorul php. Adică dacă avem un text care ar conține ghilimele am avea probleme la afișarea sa.

Exemplu:

```
<?php
echo "Acesta este "textul" care se va
afiș a<br>\n";
echo "Acesta este al doilea text\scris care se va
afiș a"; ?>
```

Dacă vom lăsa php-ul să interpreteze acest script vom

obține un rezultat nesatisfăcător:

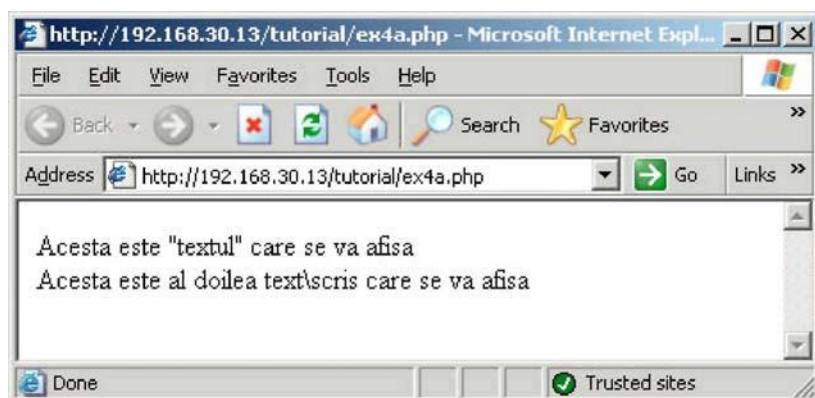


Fig. 76. Mesaj de eroare în php

Linia a doua conține mai mult de 2 ghilimele și acest lucru duce la eroarea afișată. Pentru a evita astfel de probleme trebuie să „marcam” ghilimelele care sunt necesare în text și care nu fac parte din sintaxă. Acest lucru se face cu ajutorul caracterului slash– se pune un \ înaintea ghilimelelor:

Codul ar trebui să arate astfel:

```
<?php
echo "Acesta este \"textul\" care se va
afiș a<br>\n";
echo "Acesta este al doilea text\\scris care se
va afiș a"; ?>
```



**Fig. 77. Caracterul escape **

Observăm faptul că pentru a afișa un \ a trebuit să adăugăm în codul sursa un al doilea \ ca și în limbajul C.

De asemenea sunt cazuri în care am dori să afișăm semnul „\$” în textul nostru. În PHP acesta este un semn rezervat pentru variabile în sensul că orice șir de caractere care începe cu caracterul \$ este considerat nume de variabilă.

Pentru a rezolva această problemă utilizăm tot semnul de slash \

Exemplu:

```
<?php
echo "Semnul de dolar \$ este o unitate
monetara"; ?>
```

4.3 Variabile în PHP

În PHP variabilele nu trebuie declarate. Variabilele sunt

salvate în fișiere text într-un folder TMP (dacă la instalarea PHP-ului observați acest folder nu-l stergeți!!!!!!). În momentul în care scriptul care a folosit anumite variabile și pagina generată au fost trimisă serverului de web și mai departe utilizatorului aceste variabile sunt distruse automat.

4.3.1 Declararea variabilelor

Toate variabilele încep cu semnul de dolar „\$” urmat de `_` sau o literă. Variabilele sunt case-sensitive astfel încât `$ID` nu este aceeași cu variabila `$Id`. Nu se acceptă nume de variabile care după caracterul \$ urmează o cifră.

4.3.2 Atribuirea de valori unei variabile

Atribuirea unei valori (indiferent ca este vorba de string, integer, float, sau bool) se face folosind operatorul de atribuire „=”.

4.3.3 Șiruri de caractere

Exemplu:

```
<?php
$xy = "Salutare!"; $_10 = "Studenti";
?>
```

În ambele cazuri variabilelor `$xy` și `$_10` li se atribuie cate un string. Acesta poate fi ulterior afișat cu ajutorul comenzii `echo`.

Pentru a uni două șiruri de caractere în aceeași variabilă se utilizează operatorul “.” (punct).

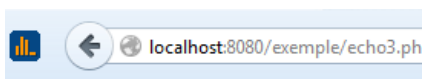
Exemplu:

```
<?php
$variable="Buna"." ziua!"; // va afișa Buna ziua!"
echo $variable;
echo "Salutare"."studenti"; // va afișa Salutarestudenti!"
?>
```

Se pot uni inclusiv o variabilă de tip string cu un string sau două variabile de tip string.

Exemplu:

```
<?php
$bla = "Mama";
$name = $bla." mea";
echo $name;
echo $name." si a tasi a ta"
?>
```



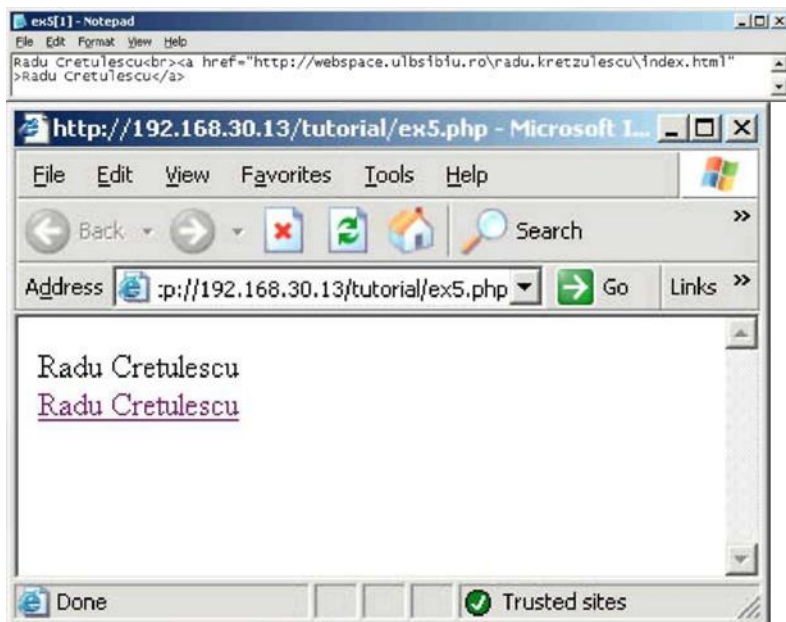
Mama meaMama mea si a tasi a ta

Într-o expresie se pot utiliza mai mulți operatori de concatenare. Oricum nu se va ține cont dacă se unesc string-uri sau variabile.

Exemplu:

```
<?php
```

```
$prenume = "Radu";  
$nume = "Cretulescu";  
$nume = $prenume." ".$nume;  
echo $nume;  
echo "<a href=\"http://webpace.ulbsibiu.ro\\radu.kret  
zulescu\\index.html\">\" ".$prenume." ".$nume."</a>\\n";  
?>
```



4.3.4 Tipuri de variabile

4.3.4.1 Variabile predefinite

`$_SERVER[]` – este o variabilă globală de tip array (vector) care conține o serie de variabile ale căror valori sunt setate de server-ul web; majoritatea valorilor variabilelor din acest

vector depind de mediul de execuție al script-ului curent.

Exemplu:

```
<?php
echo $_SERVER['SERVER_NAME'];
?>
```

`$_GET[]` și `$_POST []` – sunt tot vectori globali și în care sunt automat salvate valorile venite din link sau din formulare.

`$_COOKIE[]` conține valorile variabilelor care cuprind informații referitoare la cookie-urile păstrate pe calculatorul utilizatorului ce accesează pagina web.

Exemplu:

```
<?php
echo "Hello ". htmlspecialchars($_COOKIE['name'])
. "!" ;? >
```

`$_FILES[]` - conține variabile primite de script referitoare la datele despre fișierele uploadate de către utilizatori trimise prin metoda *post*.

Exemplu:

1. Formularul de preluare a fișierului:

```
<html>
<body>

<form action="upload_file.php" method="post">
```

```
enctype="multipart/form-data">
<input type="file" name="file">
<br>
<input type="submit" name="submit" value="Submit" >
</form>

</body>
</html>
```

2. Scriptul de preluare a datelor venite din formular

```
<?php
if ($_FILES["file"]["error"] > 0)
{
    echo "Error: " . $_FILES["file"]["error"] . "<br
/>";
}
else
{
    echo "Numele fisierului: " . $_FILES["file"]["name"]
. "<br />";
    echo "Tip: " . $_FILES["file"]["type"] . "<br >";
    echo "Marime: " . ($_FILES["file"]["size"] / 1024) .
" Kb<br />";
    echo "Salvat in: " . $_FILES["file"]["tmp_name"];
    move_uploaded_file($_FILES["file"]["tmp_name"],
$noul_director;
}
?>
```

3.

4.3.4.2 Variabile de tip întreg

La atribuirea valorii unei variabile se stabilește automat

tipul variabilei. Dacă dorim să definim o variabilă `int` atunci îi atribuim direct o valoare întreagă.

Exemplu 1:

```
<?php
$valoare = 3;
echo "Numarul ales este ".$valoare."\n";
?>
```

La început pare foarte interesant pentru că nu mai trebuie să declarăm aceste variabile înainte să le folosim. În PHP pentru că nu avem compilator variabilele se memorează într-un fișier text la nivel de server.

4.3.4.3 Variabile de tip float

Numerele care conțin parte zecimală sunt de tip float. Dacă o variabilă o cu un număr real aceasta devine de tip float.

Exemplu:

```
<?php
$g = 9.81;
echo "Constanta gravitațională g= ".$g."
unitatea de masura\n";
?>
```

La acest tip de numere este important să reținem că separatorul zecimal este semnul punct, semnul virgulă se utilizează pentru separarea parametrilor unei funcții.

Exemplu:

```
<?php
xyz(5.6);
abc(5,6);
abc(5, 6); // ca si mai sus
?>
```

În exemplul de mai sus 5.6 este un parametru de tip float al funcției `xyz()` iar în linia 3 și 4 valorile 5 și 6 sunt doi parametri ai funcției `abc()`.

Mai apare încă o problemă: Punctul este și operatorul de concatenare. Dacă am dori să unim două numere avem următoarele posibilități:

1. Atribuim mai întâi numerele unor variabile și apoi unim acestea cu operatorul punct.

Exemplu:

```
<?php
$unitati = 4;
$zeci = 5;
echo $zeci.$unitati;
?>
```

2. Dacă dorim să unim două valori întregi cu ajutorul operatorului punct atunci trebuie să inserăm câte un spațiu înainte și după operator.

Exemplu:

```
<?php
echo 4 . 5; // Afiseaza 45
echo 4.5; // Afiseaza 4.5
```

?>

4.3.5 Array-uri și Array asociativ

Aceste tipuri de vectori sunt foarte des utilizați în realizarea scripturilor PHP care generează pagini web. Față de vectorii „normali”, care au la index valori întregi de exemplu `a[3]` vectorii asociativi au ca și indecși chei (cuvinte) de exemplu `a["denumire"]`. Atribuirea de valori pentru astfel de vectori se poate face ca în exemplul de mai jos:

Exemplu:

```
<?php
$a = array ('denumire' => 'portocala',
'familie' => 'citrice',
'culoare' => 'portocaliu',
'forma' => 'rotunda',
'gust' => 'dulce'
);
echo $a['denumire'].' <br> '.$a['familie'];
?>
```

Practic scriptul de mai sus va afișa pe prima linie cuvântul `portocala` iar pe a doua linie `citrice`.

În PHP nu există noțiunea de matrice dar există vectori de vectori.

Exemplu:

<?php

```
$fructe = array (
```

```

'denumire' => array('p' => 'portocala', 'a' => 'ananas',
'm' => 'mar'),
'numere' => array(1, 2, 3, 4, 5, 6),
'ordine' => array('prima', 5 => 'a II-a', 'a III-a')
);
echo      'Denumire:      '.$fructe['denumire']['p'].'      -
          '.$fructe['denumire']['a'].' <br>';
echo      'Numere:        '.$fructe['numere'][0].'        -
          '.$fructe['numere'][1].' <br>';
echo      'Ordine:        '.$fructe['ordine'][0].'        -
          '.$fructe['ordine'][5].' <br>';
?>

```

4.4 Operatori PHP

Operatorii în php sunt aproape identici cu cei din C++

4.4.1 Operatori aritmetici

| Operator | Descriere | Exemplu | Rezultat |
|----------|-----------------|---------------------|-------------|
| + | Adunare | x=2
x+2 | 4 |
| - | Scădere | x=2
5-x | 3 |
| * | Înmulțire | x=4
x*5 | 20 |
| / | Împărțire | 15/5
5/2 | 3,
2.5 |
| % | Modul
(rest) | 5%2
10%8
10%2 | 1
2
0 |

| | | | |
|----|------------------|------------|-----|
| ++ | Increment
are | x=5
X++ | x=6 |
| -- | Decrement
are | x=5
x-- | x=4 |

Ordinea operatorilor este cea din C.

4.4.2 Operatori de atribuire

| Operator | Exemplu | Corespunde |
|----------|---------|------------|
| = | x=y | x=y |
| += | x+=y | x=x+y |
| - = | x-=y | x=x-y |
| *= | x*=y | x=x*y |
| /= | x/=y | x=x/y |
| %= | x%=y | x=x%y |

4.4.3 Operatori de comparare

| Operaor | Descriere | Exemplu |
|---------|------------------------------|-----------------------|
| == | este egal cu | 5==8 returnează false |
| != | nu este egal | 5!=8 returnează true |
| > | este mai mare | 5>8 returnează false |
| < | este mai mic | 5<8 returnează true |
| >= | este mai mare
sau egal ca | 5>=8 returnează false |
| <= | este mai mic
sau egal ca | 5<=8 returnează true |

4.4.4 Operatori logici

| Operaor | Descriere | Exemplu |
|---------|-----------|--|
| && | si | x=6 y=3 (x < 10 && y > 1)
returnează true |
| | sau | x=6 y=3 (x==5 y==5)
returnează false |
| ! | not | x=6 y=3 !(x==y)
returnează true |

4.5 Instrucțiuni php

4.5.1 Instrucțiuni condiționale

4.5.1.1 IF

Pentru a lua o decizie, în scriptul nostru PHP, putem folosi instrucțiunea „if”. Această instrucțiune primește ca și parametru o condiție pe care să o folosească, iar dacă acea condiție este adevărată, va fi executat blocul de cod de după ea. Condițiile din instrucțiunea „if” trebuie să fie trecute între paranteze rotunde ()

Exemplu:

```
<?php
$a = 12;
$b = 8;
$rezultat = $a + $b;
if($rezultat == '20') {
    echo "Rezultatul este perfect";
}
?>
```

Puneți codul într-un fișier cu numele `if.php`, salvați fișierul pe server (altfel codul php nu va fi interpretat) și apoi

vizualizați în browser accesând `http://url/...`

Observați, condiția noastră, și anume aceea că valoarea rezultată în urma adunării dintre variabila `a` (12) și variabila `b` (8) să fie egală cu numărul 20, este adevărată și în acest caz, codul de după `if` dintre acolade a fost executat.

Dacă valoarea adunării dintre variabila `a` și variabila `b` nu era 20, atunci în browser nu se afișa nimic.

Exemplu:

```
<?php
$a = 155;
$b = 8;
$rezultat = $a + $b;
if($rezultat == '20') {
    echo "Rezultatul este perfect";
}
?>
```

4.5.1.2 Instrucțiunea ELSE

De multe ori, pe lângă decizia de a executa o acțiune, atunci când condiția este adevărată, dorim să executăm o alta în caz contrar (în cazul în care condiția nu este adevărată) să returneze o altă bucată de cod.

Exemplu:

```
<?php
$a = 20;
$b = 8;
$rezultat = $a + $b;
if($rezultat == '20') {
```

```
echo "Rezultatul este perfect";
} else {
echo "Rezultatul nu este egal cu cel din
conditie";
}
?>
```

4.5.1.3 Instrucțiunea *ELSEIF*

Aceasta instrucțiune este (dupa cum vedeți) o combinație dintre instrucțiunea `if` și cea `else`.

Aceasta poate verifica fiecare condiție până în momentul în care una dintre condițiile găsite returnează o valoare adevărată.

Exemplu:

```
<?php
$a = 20;
$b = 1;
$rezultat = $a + $b;
if($rezultat == '20') {
    echo "Rezultatul este egal cu 20";
}
elseif ($rezultat == '21') {
    echo 'Rezultatul este egal cu 21';
}
else {
echo "Rezultatul nu este egal cu cel din
conditie";
}
?>
```

4.5.1.4 SWITCH

Această instrucțiune funcționează asemănător cu `if`, însă permite ca și condiția să aibă mai mult de 2 valori (evitare utilizare intructiunea `if-elseif` cascadata). Într-o instrucțiune `if`, condiția poate fi adevărată sau falsă, însă într-o instrucțiune `switch` condiția poate lua orice valoare dintre cele specificate. Această instrucțiune trebuie să conțină o instrucțiune `case` care să manevreze fiecare valoare pe care o permite.

Exemplu:

```
<?php
if(!isset($_GET['modul']))
$_GET['modul'] ="";

switch($_GET['modul']) {
case "":
    echo "Pagina switch.php";
    break;
case "pagina1":
    echo "Pagina switch.php?modul=pagina1";
    break;
case "pagina2":
    echo '"Pagina switch.php?modul=pagina2";
    break; }

?>
```

4.5.2 Instrucțiuni repetitive

4.5.2.1 Instrucțiunea WHILE

Cel mai simplu tip de buclă PHP este `while(condiție)` {

`instrucțiuni;}`. Asemenea instrucțiunii `if`, ea se bazează pe o condiție. Diferența dintre `if` și `while` este aceea că instrucțiunea `if`, dacă găsește adevărată condiția, execută o singură dată bucata de cod din ea, iar în instrucțiunea `while`, instrucțiunile din interiorul blocului `while` vor fi atâta timp executate până când condiția din `while` devine falsă.

Exemplu:

```
<?php
$numar = 1;
while($numar <= 5) {
    echo $numar. "<br>";
    $numar++; }
?>
```

4.5.2.2 Instrucțiunea FOR

În cazul în care se cunoaște de câte ori trebuie executată bucla atunci se poate utiliza structura repetitiva `for`. Sintaxa este foarte asemănătoare cu cea din limbajele C/C++ și anume:

```
for(expresie1; condiție; expresie2) {
//instrucțiuni
}
```

Prima expresie este evaluată o singură dată, înainte de începerea execuției buclei.

Expresia `condiție` este testată înaintea fiecărei repetări a buclei. Dacă expresia returnează `false`, repetarea se oprește.

Expresia 2 este executată la sfârșitul fiecărei repetări

înainte de a se reevalua expresia din condiție.

Instrucțiunile dintre acolade se vor executa la fiecare repetare a buclei.

Oricare dintre cele trei expresii poate lipsi; în cazul în care o expresie lipsește, se consideră că ea are valoarea `true` tot timpul.

Exemplu:

```
<?php
for ($var = 1; $var <= 10;$var++) {
    echo $var. "<br>"; }
?>
```

Un alt exemplu pentru folosirea instrucțiunii `for`:

```
<?php
echo "<table>\n<tr><td>Celula</td></tr>\n";
$culoare = "yellow";
for ($variabila = 1; $variabila <= 10;$variabila++)
{
    if($culoare == "yellow")
        $culoare = "red";
    else
        $culoare = "yellow";
    echo "<tr><td bgcolor=\".$culoare.\">".
        .$variabila."</td></tr>\n";
}
echo "</table>";
?>
```

4.5.2.3 Instrucțiunea *FOREACH*

Această instrucțiune este foarte utilă în parcurgerea elementelor unui vector asociativ.

Exista doua sintaxe acceptate pentru această instrucțiune și anume:

```
foreach(expresie_vectoriala as $valoare) {  
    //instructiuni  
}
```

La fiecare iterație valoarea elementului curent este atribuită variabilei `$valoare`, și apoi se trece la elementul următor

Exemplu:

```
<?php  
$sir = array("Ion", "Vasile", "Gheorghe",  
"Patru", "Mihai");  
foreach($sir as $valoare) {  
    echo "Valoare: ".$valoare." <br />\n";  
}  
?>
```

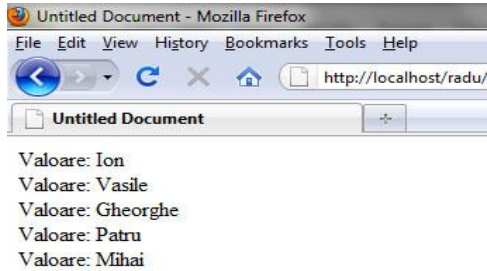


Fig. 78. FOREACH

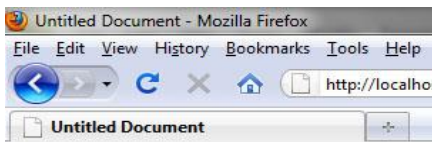
Varianta 2:

```
foreach(expresie_vectoriala as $cheie => $valoare) {  
    //instructiuni  
}
```

La fiecare iterație valoarea indexului elementului curent este atribuită variabilei *\$cheie*.

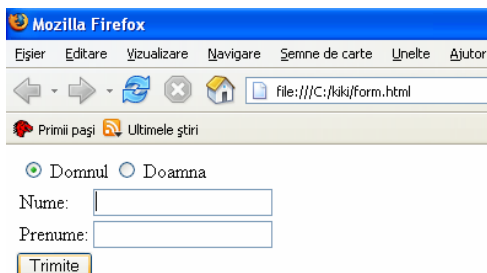
Exemplu:

```
<?php  
$sir  =  array("Ion",    "Vasile",    "Gheorghe",  
"Patru", "Mihai");  
foreach($sir as $index => $valoare) {  
    echo "Index: ".$index." Valoare: ".$valoare." <br  
    />\n";  
}  
?>
```



Index: 0 Valoare: Ion
Index: 1 Valoare: Vasile
Index: 2 Valoare: Gheorghe
Index: 3 Valoare: Patru
Index: 4 Valoare: Mihai

4.6 Preluarea datelor dintr-un formular



```
<form action="formular.php" method="get">
<table>
<tr>
<td colspan=2>
<input type="radio" name="a" value="b">
domnul
<input type="radio" name="a" value="f">
doamna
<tr>
<td>name:
<td><input type="text" name="nu" size="20">
<tr>
<td>vorname:
<td><input type="text" name="pre" size="20">
```

```
</table>
<input type="submit" value="trimite">
</form>
```

Formularul prezentat în exemplul de mai sus este pur cod html, deci este static. Importante sunt datele pe care le trimite. În cazul nostru din formularul de mai sus datele vor fi trimise, conform cu atributul `action` la scriptul `"formular.php"`. Acesta le va prelua în funcție de metoda cu care au fost trimise datele în variabila globală `$_POST[]` sau `$_GET[]`. În cazul nostru datele au fost trimise cu ajutorul metodei `get`. Scriptul `formular.php` arată astfel.

```
<?php
$nume=$_GET['nu'];
$prenume=$_GET['pre'];
$titlu=$_GET['a'];
if ($titlu=="b")
    echo "Domnul";
else
    echo "Doamna";
echo "Numele este:" $nume;
echo "Prenumele este:" $prenume;

?>
```

5 MySQL

În acest capitol vom prezenta modalitățile de conectare la un server de baze de date și executarea unor instrucțiuni simple SQL.

5.1 Conectarea la MySQL folosind funcții PHP

Cel mai ușor mod de a utiliza un script de conectare la o bază de date este să realizăm acest lucru într-un fișier separat și care apoi poate fi inclus cu ajutorul directivei `include` sau `include_once` în scripturi PHP care au nevoie de conexiuni la un server de baze de date.

PHP-ul pune la dispoziția programatorilor pentru fiecare tip de sever de baze de date un set de funcții PHP specifice pentru acel server. În continuare vom folosi setul de funcții pentru serverul MySQL.

Conectarea la serverul de baze de date și selectarea bazei de date cu care lucrăm:

Exemplu depreciat dar des întâlni:

```
<?php
// Informatii baza de date
$host = "localhost"; //aici numele serverului
$user = "root"; // aici userul
$password = "parola_mysql"; //parola
```

```
$db_name = "teste"; // numele bazei de date

$conexiune = mysql_connect($host,$user,$passwd) or
die("Nu ma pot conecta la MySQL!");
mysql_select_db($db_name,$conexiune) or die("Nu
gasesc baza de date!");
?>
```

Exemplu modalitate nouă (config.php):

```
<?php
// Informatii baza de date
$host = "localhost"; //aici numele serverului
$user = "root"; // aici userul
$passwd = "parola_mysql"; //parola
$db_name = "teste"; // numele bazei de date

$conexiune = mysql_i_connect($host,$user,$passwd,$db);
if(!$conexiune)
    echo mysqli_connect_error();
?>
```

Acesta este fișierul de configurare cu care vom realiza conexiunea la baza noastră de date.

Observăm **i**-ul care s-a adăugat după `mysql` care este prescurtarea de la cuvântul `improved`

Variabilei `$host` îi este atribuită valoarea `localhost` deoarece aceasta este adresa serverului. (Adică, serverul Apache+PHP este instalat pe același calculator ca și pachetul MySQL). În cazul în care serverul de baze de date rulează pe un al calculator atunci acest parametru poate fi un URL sau o

adresă de IP.

Variabilelor `$user` și `$passwr` le sunt atribuite `username`-ul și parola pentru accesul la serverul de baze de date. Aceste valori sunt transparente pentru utilizatorul aplicației de web. **NU TREBUIE SALVATE ÎN SCRIPTURI CARE SE EXECUTĂ LA UTILIZATOR (gen Javascript).**

Execuția codului de mai sus produce o fereastră de browser goală. Dacă se afișează ceva înseamnă că este o problemă cu serverul de baze de date.

`mysqli_connect_error` – este un macrou (funcție) care afișează automat un mesaj cu eroarea....

5.2 Comenzi SQL

5.2.1 Comanda SELECT

Exemplificăm în continuare utilizarea comenzilor SQL prin scripturi php. Ideea de bază de la care plecăm este aceea că înainte de a executa o comandă pe serverul de baze de date prima dată o salvăm într-o variabilă locală a scriptului apoi o trimitem cu ajutorul funcției `mysqli_query()` spre execuție la serverul de baze de date. În momentul finalizării execuției comenzii SQL funcția `mysqli_query()` va returna un obiect sau valorile `true` sau `false`:

Exemplu un fișier cu numele `extragere_date.php`:

```
<?php
```

```
include 'config.php';
// Selectare date din baza de date
$cerereSQL = 'SELECT * FROM `formular`';
$resultat = mysqli_query($conexiune, $cerereSQL);
while($rand = mysqli_fetch_array($resultat)) {
    echo $rand['nume'];
}
?>
```

Și acum să înțelegem codul PHP de mai sus:

Prima linie din script trebuie să fie **funcția include**, funcție care include, în scriptul nostru `extragere_date.php` fișierul `config.php`; adică datele din `config.php` vor fi transmise în scriptul nostru. Vom avea acces inclusiv la variabila `$conexiune`.

Instrucțiunea

```
$cerereSQL = "SELECT * FROM formular";
```

Atribuie variabilei `$cerereSQL` un șir de caractere care de fapt este comanda SQL pentru a extrage datele din tabela `formular`. Ea se interpretează cam așa: **SELECTEAZA** tot **DIN** `formular`. Unde *formular* este o tabelă din baza noastră de date care poate arata după cum am scris mai jos în exemplu.

Instrucțiunea

```
$resultat = mysqli_query($conexiune, $cerereSQL);
```

În această linie avem variabila `$resultat` cu valoarea returnată de funcția **mysqli_query**, funcție care trimite la

server comanda SQL .

Instrucțiunea

```
while($rand = mysqli_fetch_array($rezultat)) {  
    echo $rand['nume'];  
}
```

În interiorul acestei bucle putem afișa informațiile conținute în tabela `formular`.

Să observăm următoarea secvență de cod:

```
while($rand = mysqli_fetch_array($rezultat)
```

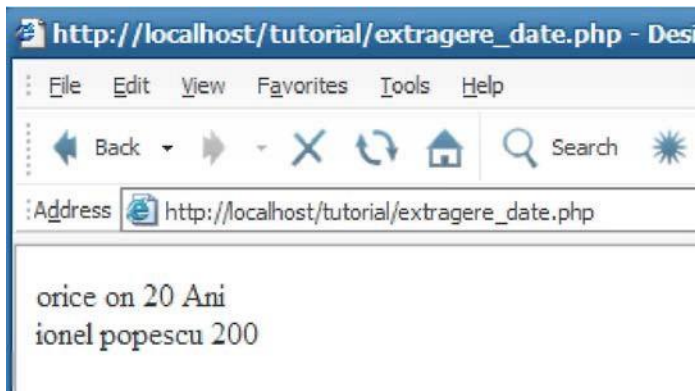
În variabila `$rand` funcția `mysqli_fetch_array($rezultat)` va returna la fiecare apelare (de aceea am pus-o într-o buclă) câte o un „rând” din obiectul `$rezultat` (acest obiect fiind returnat de funcția `mysqli_query()`). În momentul în care nu mai sunt rânduri de returnat la următorul apel funcția `mysqli_fetch_array` va returna `fals` și se iese din buclă. Funcția `mysqli_fetch_array` returnează la fiecare apel un vector asociat care se va salva în `$rand` (care devine un vector asociativ). Fiind vector asociativ, cheile sunt chiar numele câmpurilor din tabela `formular` și astfel putem afișa un nume salvat în tabela `formular` în câmpul `numele` prin instrucțiunea:

```
echo $rand['nume'];
```

Identic și prenumele, și vârsta, modificând instrucțiunea `echo`, și anume:

```
echo      $rand['nume']. "      " . $rand['prenume']. "      " .
```

```
.$rand['varsta']. " <br>";.
```



În exemplul de sus, avem 2 intrări în tabela formular si anume:

← T →		id	nume	prenume	varsta
☐	 	1	orice	on	20 Ani
☐	 	2	ionel	popescu	200

În cererea noastră SQL de SELECT am folosit atributul * adică toate câmpurile din tabela formular, însă se poate selecta numai un câmp sau mai multe și anume:

```
$cerereSQL = "SELECT `nume,prenume` FROM  
formular";
```

Sintaxa comenzii SELECT cu toate clauzele SQL este:

```
SELECT
[ALL | DISTINCT | DISTINCTROW ] [HIGH_PRIORITY]
[STRAIGHT_JOIN]
[SQL_SMALL_RESULT]                        [SQL_BIG_RESULT]
[SQL_BUFFER_RESULT]
[SQL_CACHE | SQL_NO_CACHE] [SQL_CALC_FOUND_ROWS]
select_expr, ... [INTO OUTFILE 'file_name'
export_options
| INTO DUMPFILE 'file_name'] [FROM
table_references
[WHERE where_definition]
[GROUP BY {col_name | expr | position} [ASC |
DESC], ... [WITH ROLLUP]]
[HAVING where_definition]
[ORDER BY {col_name | expr | position} [ASC |
DESC] , ...]
[LIMIT {[offset,] row_count | row_count
OFFSET offset}]
[PROCEDURE procedure_name(argument_list)] [FOR
UPDATE | LOCK IN SHARE MODE]]
```

În următorul exemplu, pentru a selecta anumite înregistrări din tabela `formular`, vom adăuga clauza `WHERE`.

```
<?php
include 'config.php';
// Selectare date din baza de date
$cerereSQL = "SELECT * FROM formular WHERE
nume='orice'";
$rezultat = mysqli_query($conexiune, $cerereSQL);
```

```
while($rand = mysqli_fetch_array($rezultat)) {  
    echo $rand['nume']. " " . $rand['prenume'].  
    ." " . $rand['varsta']. " <br>";  
}  
?>
```

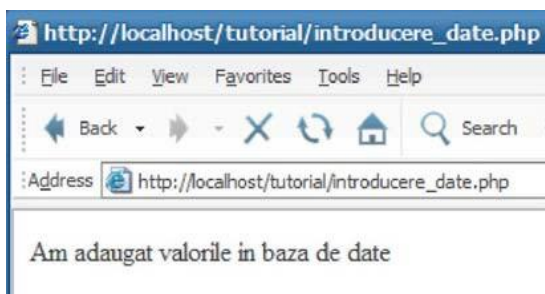
Se va observa că pentru condiția `WHERE nume="orice"`, rezultatul o să fie numai prima linie din tabela `formular`, deoarece primul `nume` de acolo este egal cu numele dat de noi în condiție (orice).

5.2.2 Comanda INSERT

Comanda SQL `INSERT` se folosește pentru a adăuga o înregistrare (un rând) în tabela unei baze de date. Să luăm următorul exemplu:

```
<?php  
include 'config.php';  
$cerereSQL = "INSERT INTO formular (nume  
    , prenume, varsta)  
VALUES ('Ivascu', 'Valentin', '20')";  
$rez = mysqli_query($conexiune, $cerereSQL;  
if($rez){  
    echo "Am adaugat valorile in baza de date";  
}else{  
    echo "Nu am reusit adaugarea in baza de  
date";  
}  
?>
```

În fereastra browser-ului se va afișa în caz de succes mesajul:



Practic execuția comenzii INSERT nu are un output iar în cazul în care eșuează insertul funcția `mysqli_query` va returna false.

Sintaxa comenzii SQL INSERT:

```
INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY]
[IGNORE] [INTO] tbl_name [(col_name,...)] VALUES
({expr | DEFAULT},...), (...),... [ ON DUPLICATE
KEY UPDATE col_name=expr, ... ]
```

sau:

```
INSERT [LOW_PRIORITY | DELAYED |
HIGH_PRIORITY] [IGNORE] [INTO] tbl_name SET
col_name={expr | DEFAULT}, ... [ ON DUPLICATE KEY
UPDATE col_name=expr, ... ]
```

sau:

```
INSERT [LOW_PRIORITY | HIGH_PRIORITY] [IGNORE]
[INTO] tbl_name [(col_name,...)] SELECT ... [ ON
DUPLICATE KEY UPDATE col_name=expr, ...
]
```

5.2.3 Comanda UPDATE

Comanda `UPDATE` se folosește pentru a modifica înregistrări sau anumite câmpuri ale unor înregistrări dintr-o tabelă a unei baze de date.

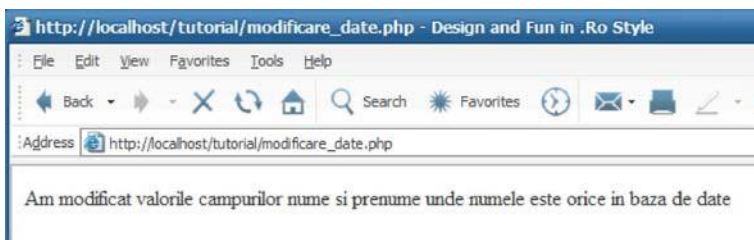
Exemplu:

```
<?php
include "config.php";

$cerereSQL = "UPDATE formular SET nume='nume',
prenume='prenume' WHERE nume='orice' ";
$rez=mysqli_query($conexiune, $cerereSQL);
if ($rez)
    echo "Am modificat valorile campurilor nume
si prenume unde numele este orice in tabela";
else
    echo"Problema la UPDATE";

?>
```

Codul de mai sus va afișa în caz de succes doar mesajul :



Observăm clauza `WHERE` scrisă cu roșu pentru a atenționa cititorul asupra acestui aspect. În cazul în care se omite această clauză evident comanda `UPDATE` se aplică

fiecărei înregistrări din tabelă.

Sintaxa comenzii SQL UPDATE

Instrucțiune simplă

```
UPDATE          [LOW_PRIORITY]          [IGNORE]          SET
col_name1=expr1 [, col [WHERE where_definition]
[ORDER BY ...] [LIMIT row_count]
```

Instrucțiune multiplă

```
UPDATE [LOW_PRIORITY] [IGNORE] table_references
SET   col_name1=expr1 [, col_name2=expr2 ...]
[WHERE where_definition] tbl_name name2=expr2
...]
```

5.2.4 Comanda DELETE

Comanda DELETE se folosește pentru a șterge înregistrări dintr-o tabelă. Se pot șterge doar înregistrări întregi (adică tot rândul din tabelă nu doar o parte din el).

Exemplu:

```
<?php
include "config.php";
$cerereSQL = "DELETE FROM formular WHERE
```

```
nume='Ion';
$rez=mysqli_query($conexiune,$cerereSQL);
if(!$rez)
    echo "Probleme la stergere";
else
    header("location:afis.php");
```

Acest script va șterge din tabela *formular* înregistrarea la care câmpul nume=Ion. Dacă instrucțiunea nu se poate executa atunci browser-ul va afișa Probleme la stergere. La fel ca și la instrucțiunea UPDATE este foarte importantă condiția din clauza WHERE, altfel vor fi șterse toate înregistrările.

Sintaxa comenzii DELETE:

Sintaxa simpla

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE] FROM
tbl_name [WHERE where_definition] [ORDER BY
sau:
...] [LIMIT row_count]
```

Sintaxa multiplă

```
DELETE          [LOW_PRIORITY]          [QUICK]          [IGNORE]
tbl_name[.*]    [,    tbl_name[.*]    ...]    FROM
table_references [WHERE where_definition]

DELETE [LOW_PRIORITY] [QUICK] [IGNORE]
FROM tbl_name[.*] [, tbl_name[.*] ...] USING
table_references [WHERE where_definition]
```

6 Anexa 1 – Exemplu <div>

Structurarea paginii cu ajutorul tagurilor <div>

```
<!DOCTYPE html>
<meta charset="UTF-8">
<title>Exemplu</title>
<link href="stil.css" rel="stylesheet" type="text/css">
</head>

<body>
<div id="wrapper">
<div id="header">
<h1>Acesta este header-ul paginii</h1>
</div>

<div id="left-column">
<h2>Meniul de navigare</h2>
<?php
include "meniu.php";
?>
</div>

<div id="right-column">
<h2>Studentii din scoala noastra</h2>
<?php
include "conect.php";
$sql="SELECT * FROM studenti";
$res=mysqli_query ($link, $sql);
?>
<table>

    <tr>

        <th>Nr Crt</th>
        <th>Nume</th>
        <th>Prenume</th>
        <th>Semi-grupa</th>
        <th>Email</th>
        <th>Data nasterii</th>

    </tr>

    <?php
        $i=1;
        while($row=mysqli_fetch_array($rez)
        {
            echo"<tr>";
                if($i%2==0)
                    echo" class=\"still\"";

            echo"><td>". $i."</td><td>". $row['nume']."</td><td>". $row['prenume'
]. "</td><td>".
```

```

        $row['sgrupa']."</td><td>".$row['email']."</td><td>".$row['data_n']
    ]. "</td></tr>";
    $i++;
}
?>
</table>
</div>

<div id="footer">
<p>Footer-ul</p>
</div>
</div><!--aici se termina wrapperul-->

</body>
</html>

```

Acest cod va afișa lista cu înregistrările din tabela studenți

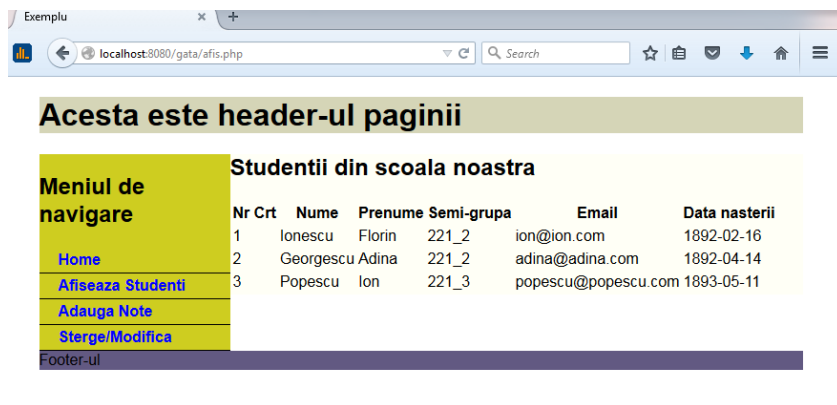


Fig. 79. Structură pagina – afișare studenți

Fișierul de stil inclus stil.css:

```

@charset "utf-8";
/* CSS Document */
*{font-family:Arial;
}

```

```
#wrapper {
    background: #fff;
    width: 800px;
    margin: 0 auto;
}
#header {
    background: #D5D5B7;
}

#left-column {
    background: #CECD20;
    float:left;
    width:200px;
}
#right-column {
    background: #FFFFFF6;
    margin-left:200px;
    width:600px;
}

#footer {
    background: #625982;
    clear:both;
}
#navbar li{list-style-type:none;}

#navbar {
margin: 0;
padding: 0;
}
#navbar li a:link, #navbar li a:visited {
display: block;
text-decoration: none;
font-weight: bold;
border-bottom: solid #000000 1px;
padding-top: 3px;
padding-bottom: 3px;
padding-left: 20px;
}
#navbar li a:hover{
display: block;
text-decoration: none;
font-weight: bold;
border-bottom: solid #000000 1px;
padding-top: 3px;
padding-bottom: 3px;
padding-left: 20px;
background-color:#CFC;
color:#903;
}

```

Fișierul de conectare la baza de date `test` este:

```
<?php
    $user="root";
    $pass="";
    $host="localhost";
    $db="test";

$link=mysqli_connect($host, $user, $pass, $db);

if(mysqli_connect_errno())
    echo mysqli_connect_error($link);

?>
```

Pentru a putea modifica ușor structura de navigare am optat pentru a scrie meniul de navigare în alt fișier `menu.php` și care va fi inclus în fiecare fișier unde avem nevoie de meniu de navigare cu instrucțiunea `php include "menu.php":`

```
<ul id="navbar">
<li><a href="index.php">Home</a></li>
<li><a href="afis.php">Afiseaza Studenti</a></li>
<li><a href="adauga.php">Adauga Note</a></li>
<li><a href="sterge_modi.php">Sterge/Modifica</a></li>
</ul>
```

Pentru a adăuga înregistrări noi în tabela studenți am utilizat un formular dinamic. Acest formular este dinamic pentru că cele două input-uri `<select> ... </select>` se generează dinamic utilizând informații conținute în tabela studenți și materii. Utilizatorul are astfel posibilitatea de a alege studentul și materia la care vrea să adauge o notă.

Codul:

```
<!DOCTYPE html>
```

```

<meta charset="UTF-8">
<title>Formular dinamic</title>
<link href="stil.css" rel="stylesheet" type="text/css">
</head>
<body>
<div id="wrapper">
<div id="header">
<h1>This is the Page Header</h1>
</div>

<div id="left-column">
<h2>The Navigation</h2>
<?php
include "menu.php";
?>
</div>

<div id="right-column">
<h2>Adauga note</h2>
<form action="insert_note.php" method="post">
Student:
    <?php include "conect.php";
    $sql1="SELECT * FROM studenti";
    $rez1=mysqli_query($link,$sql1);
    ?>
    <select name="idul_stud">
    <option value=0">Selectati</option>
    <?php
        while($row1=mysqli_fetch_array($rez1))
        {
            echo "<option
value=\"".$row1['id']."\">".$row1['nume']."</option>";
        }
    ?>
    </select></br>
Materie:
    <?php include_once "conect.php";
    $sql2="SELECT * FROM materii";
    $rez2=mysqli_query($link,$sql2);
    ?>
    <select name="idul_mat">
    <?php
        while($row2=mysqli_fetch_array($rez2))
        {
            echo "<option
value=\"".$row2['id']."\">".$row2['denumire']."</option>";
        }
    ?>
    </select></br>
Nota:
    <input type="text" name="nota"/><br/>

```

Data:

```
<input type="text" name="data"/><br/>

<input type="submit" value="Trimite"/>
<input type="reset" value="Sterge"/>
</form>
</div>

<div id="footer">
<p>&copy; 2015 RCDM</p>
</div>
</div><!--aici se termina wrapperul-->

</body></html>
```

Observație:

După ce utilizatorul a completat formularul datele sunt trimise către scriptul specificat în atributul `action` al formularului `<form action= "insert_note.php"...>`

Codul pentru `insert_note.php`:

```
<?php
$a = $_POST['idul_stud'];
$b = $_POST['idul_mat'];
$c = $_POST['nota'];
$d = $_POST['data'];
include "conect.php";

$sql = "INSERT INTO note(id_student, id_materie,
nota, data) VALUES ('$a', '$b', '$c', '$d')";
$res=mysqli_query($link,$sql);
if($rez)
{
    header("Location:adauga.php");
}
else
{
    echo "probleme la inserare". mysqli_error();
}

?>
```

Observăm că după inserare browserul este redirectat către fișierul `adauga.php` cu ajutorul instrucțiunii `header("Location:adauga.php");`

Pentru a șterge sau pentru a modifica o înregistrare am optat pentru alegerea dintr-un tabel prin link a respectivei înregistrări (`sterge_modi.php`).

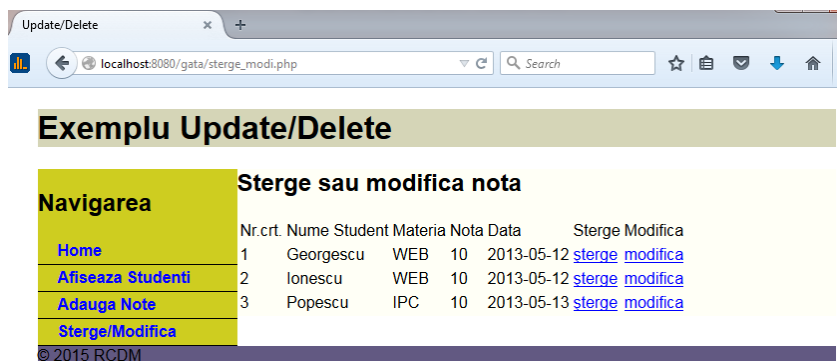


Fig. 80. Exemplu UPDATE/DELETE

Codul din fișierul pentru formularul afișat mai sus:

```
<!DOCTYPE html>
<head>
<meta charset="UTF-8">
<title>Update/Delete</title>

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

<body>
<div id="wrapper">
<div id="header">
<h1>Exemplu Update/Delete</h1>
</div>

<div id="left-column">
<h2>Navigarea</h2>
```

```

<?php
include "menu.php";
?>
</div>

<div id="right-column">
<h2>Sterge sau modifica nota</h2>
<?php
    include "conect.php";

    $sql="SELECT studenti.numa, materii.denumire, nota.nota,
    nota.data, nota.id FROM studenti,materii,nota WHERE
    nota.id materie=materii.id AND nota.id student=studenti.id";
    $rez= mysqli_query($link,$sql);
    ?>
<table>
    <tr> <td>Nr.crt.</td> <td>Nume Student</td>
<td>Materia</td> <td>Nota</td> <td>Data</td>
<td>Sterge</td><td>Modifica</td>
    </tr>
    <?php
    $i=1;
        while($row=mysqli_fetch_array($rez))
        {
            echo"<tr><td>".$i."</td>
<td>".$row['numa']."</td> <td>".$row['denumire']."</td>
<td>".$row['nota']."</td> <td>".$row['data']."</td> <td><a
href=\"sterge.php?id=".$row['id']."\">sterge</a></td><td><a
href=\"form_modi.php?id=".$row['id']."\">modifica</a></td></tr>";
            $i++;
        }
    ?>
</table>
</div>

<div id="footer">
<p>&copy; 2015 RCDM</p>
</div>
</div><!--aici se termina wrapperul-->

</body></html>

```

Am optat pentru trimiterea prin link a id-ului înregistrării care dorim să o modificăm sau să o ștergem, valoarea o regăsim după semnul ? -

```
sterge.php?id=".$row['id'].
```

Fișierul `sterge.php` primește valoarea trimisă în vectorul asociativ `$_GET["id"]`.

Codul din fișierul `sterge.php`:

```
<?php
include "conect.php";
$id=$_GET['id'];
$sql="DELETE FROM note WHERE id='$id'";
$res=mysqli_query($link,$sql);
if($res)
    header("Location: sterge_modi.php");
else
    echo"Problema la ...";

?>
```

Observăm că acest fișier nu respect structura standard a unui document html și aceasta din motivul că el este doar un script de execuție la nivel de server. Utilizatorul nu va primi niciun output dacă totul e ok.

Identic și fișierul `form_modi.php` va primi prin link id-ul înregistrării și pe baza acestui id se completează automat formularul de modificare (numele și materia).

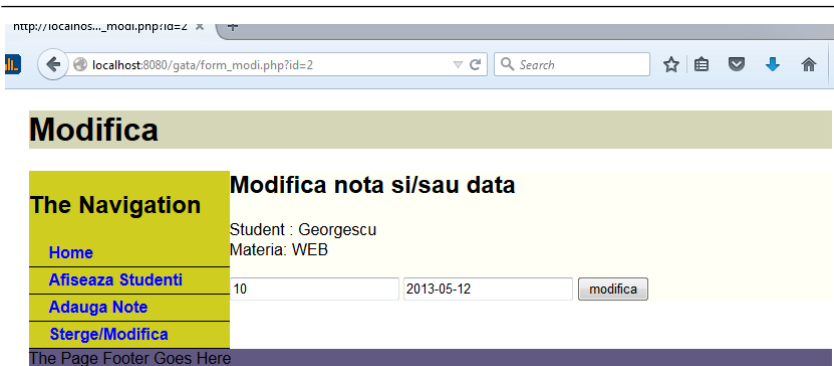


Fig. 81. Formular modificare

Codul acestui formular:

```
<!DOCTYPE html>
<head>
<meta charset="UTF-8">
<titleFormula modifica</title>

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

<body>
<div id="wrapper">
<div id="header">
<h1>Modifica</h1>
</div>

<div id="left-column">
<h2>The Navigation</h2>
<?php
include "menu.php";
?>
</div>

<div id="right-column">
<h2>Modifica nota si/sau data</h2>
<?php
    include "conect.php";
    $id=$_GET['id'];
    $sql="SELECT * FROM note,materii,student
        WHERE note.id='$id'
        AND note.id_student=studenti.id
```

```

        AND note.id_materie=materii.id";
        $rez=mysqli query($link,$sql);
        $row=mysqli fetch array($rez);
        $idstudent=$row['id student'];
        $idmaterie=$row['id_materie'];

    ?>

    <p>Student : <?php echo $row['nume']; ?><br />
    Materia: <?php echo $row['denumire']; ?><br />

    <form action="update_s.php" method="post">
    <input type="text" name="nota" value="<?php echo
$row['nota'] ?>" />
    <input type="text" name="data" value="<?php echo
$row['data'] ?>" />
    <input type="hidden" name="idu" value="<?php echo $id
?>" />
    <input type="submit" value="modifica" />

    </form>
</div>

<div id="footer">
<p>The Page Footer Goes Here</p>
</div>
</div><!--aici se termina wrapperul-->

</body></html>

```

Interesantă este instrucțiunea

```

SELECT * FROM note,materii,student WHERE note.id='$id'
AND note.id_student=studenti.id
AND note.id_materie=materii.id";

```

unde selectăm date din cele 3 tabele: note, materii, student. Datele modificate în acest formular sunt apoi trimise către scriptul `update_s.php`:

```

<?php
    include "conect.php";

```

```
$c=$_POST['nota'];
$d=$_POST['data'];
$e=$_POST['idu'];

$sql="UPDATE note SET nota='$c',data='$d' WHERE id='$e'";

$rez=mysqli_query($link,$sql);
if($rez)
    header("Location: sterge_modi.php");
else
    echo"Problema la ...";

?>
```

7 Bibliografie

- [1] Berners Lee, Tim, *WorldWideWeb: Proposal for a HyperText Project*, 1990, <http://www.w3.org/Proposal.html>
- [2] Berners Lee, Tim, *Information Management: A Proposal*, , 1998-1990, <http://www.w3.org/History/1989/proposal.html>
- [3] Bush Vannevar, *As We May Think*, iulie 1945
<http://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881/>
- [4] Nelson, Theodor, *Computer Lib: You Can and Must Understand Computers Now; Dream Machines: New Freedoms Through Computer Screens— A Minority Report*. Self-published, 1974, . ISBN 0-89347-002-3.
- [5] Vinton G. Cerf And Robert E. Kahn, *A Protocol for Packet Network Intercommunication*, IEEE Trans on Comms, Vol Com-22, No 5 May 1974
(<http://www.cs.princeton.edu/courses/archive/fallo6/cos561/papers/cerf74.pdf>)
- [6] <http://www.zakon.org/robert/internet/timeline/> (accesat în: 02.02.2015)
- [7] RFC 3305, <http://tools.ietf.org/html/rfc3305> (accesat în: 02.02.2015)
- [8] <http://www.w3.org/TR/html5/syntax.html> (accesat în: 02.02.2015)
- [9] HTML5 recommendation-
<http://www.w3.org/TR/html5/> (accesat în: 02.02.2015)
- [10] <http://www.w3.org/DOM> (accesat în iunie 2014)