

Parathënie

Aplikacionet web janë të ekspozuar në internet ndaj ndërtimi i tyre e përball programuesin me mundësi e problematika të ndryshme nga aplikacionet për desktop.

Duke qenë të ekspozuar në internet aplikacionet duhet të jenë të ndërtuara për tu bërë ballë sulmeve që synojnë të gjejnë vrima sigurie në to.

Aplikacionet nuk kanë nevojë për instalim ndaj edhe numri i përdoruesve të njëkohshëm i tyre mund të jetë i madh dhe në disa raste shumë i vështirë për tu parashikuar.

Natyra e jetesës ka rritur nevojën për të përdorur aplikacionin nëpërmjet ndërfaqeve jo tradicionale si si nga pajisjet celulare apo nëpërmjet zërit dhe telefonit.

Shpesh koha që kemi në dispozicion për ndërtimin dhe modifikimin e një aplikacioni web është e shkurtër. Një analizë e kujdesshme paraprake, një njohje e mirë e mjeteve në dispozicion dhe zbatimi i një metodologjie që ndan dizenjimin nga programimi, na mundëson që të aplikojmë ndryshime në aplikim, në mënytë të vecantë në ndërfaqen e tij, në një kohë të shkurtër dhe të ndajmë punën në ekip.

Ky libër ju njeh me mjetet dhe teknikat për ndërtimin e aplikimeve web në platformën php&mysql. Kjo është një platformë me kod burimi të hapur prandaj aplikacionet e ndërtuara në këtë platformë nuk kanë kosto për licenca të software, ndaj edhe janë me kosto të ulët për hostimin.

Libri fillon me mjetet e programimit në klient, html dhe javascript dhe vazhdon më tej me PHP dhe MySQL.

Sqarimet në libër janë bazuar shumë në shembuj konkretë dhe në fund të leksioneve ka ushtrime që do të shërbenin për përforcimin e njohurive të marra gjatë leksionit.

Një faqe interneti është ndërtuar për të dhënë zgjidhjet e disa ushtrimeve si edhe për informacione shtesë ndihmëse si udhëzime për instalimin e platformës apo teknika dhe truke në ndërtimin e aplikimeve.

Informacionet shtesë në lidhje me librin do ti gjeni në adresën <http://ictedu.info/books>

Autorët

Pasqyra e lëndës

Parathënie.....	i
Pjesa I - Programimi Client Side	1
1. HTML.....	2
1.1 Bazat e HTML	3
1.2 Struktura e dokumenteve HTML.....	4
1.3 Fontet.....	8
1.4 Linket	11
1.5 Listat	13
1.5 Imazhet	15
Ushtrime	17
2. Tabelat	18
2.1 Dizajnimi i faqes nëpërmjet tabelave.....	22
Ushtrime	24
3. Format.....	25
3.1 Elementët e Formës	26
Ushtrime	31
4. CSS	33
4.1 SPAN dhe DIV	34
4.2 Tipare të tjera të CSS	34
4.3 Trashegimia.....	35
Ushtrime	39
5. JavaScript.....	40
5.1 Programi i pare	40
5.2 Elementet baze te JavaScript.....	41
5.3. Tipet e te dhenave dhe variablat.....	41
Ushtrime	43
5.4 Strukturat e kontrollit	44
5.5 Eventet	49
5.6 Objektet.....	51
5.7 Përcaktimi i objekteve ne JavaScript.....	51
5.8 Vektorët.....	52
Ushtrime	55
5.9 Funksionet	56

5.10 jQuery.....	57
5.11. Aplikime JavaScript	60
6. DOM	70
Pjesa II - Programimi Server Side	75
7. Prezantim me PHP.....	77
7.1 Sintaksa, Variablat dhe Operatoret.....	79
7.2 Tipet e të dhënave ne PHP	79
7.3 Operatorët.....	80
Ushtrime	82
8. Strukturat e kontrollit ne Php	83
8.1 Degëzimi i kodit nepermjet If .. else	83
8.2 Cikli FOR	83
8.3 Cikli While dhe cikli do .. while	85
8.4 Degëzimi i kodit me instruksionin switch .. case	87
Ushtrime	92
9. Stringjet ne PHP	95
9.2 Funksionet mbi stringjet.....	97
9.2 Ruajtja e rreshtave të rinj nga teksti në html	100
Ushtrime	101
10. Funksionet dhe modulimi i kodit.....	103
10. 1 Kalimi i argumentave një funksioni	104
10. 2 Funksionet rekursive	106
10.3 Përfshirja e skedarëve në skedarë të tjerë PHP	107
Ushtrime	113
11. Komunikimi me browserin	114
Ushtrime	122
12. Array ne PHP.....	123
12.1 Deklarimi i nje array	123
12.2 Perdorimi i Array	125
Ushtrime	130
13. Puna me skedarët.....	132
13.1 Hapja e skedarëve nëpërmjet funksioni fopen()	132
13.2 Të lexojmë skedarin rresht për rresht nëpërmjet funksionit fgets().....	134

13.3 Të lexojme skedarin karakter për karakter nëpërmjet funksionit fgetc().....	134
13.4 Të shkruajmë në skedar nëpërmjet funksionit fwrite().....	135
13.5 Funksionet mbi datën dhe kohën.....	138
Ushtrime	140
14. Cookie-t dhe Sesionet	141
14. 1 Ruajtja e gjendjes së aplikimit në sesion	141
14.2 Startimi i sesionit	143
14.3 Cookie-t.....	144
14.4 Dërgimi i email	146
Ushtrime	150
15. Lidhja e PHP me MySQL	152
15.1 Hyrje në bazat e të dhënave.....	153
15.2 Relational Database Management Systems (RDBMS)	156
15. 3 Gjuha për bazat e të dhënave relacionale, SQL(Structured Query Language).....	161
15.4 Tipet e të dhënave	163
15.5 Tipet e të dhënave tekst.....	167
15. 6 Operatorët dhe funksionet.....	169
15. 7 Funksionet mbi stringjet.....	170
15.8 Funksionet mbi kohen	171
15.9 Komanda SELECT.....	173
15.10 Modifikimi i të dhënave (INSERT,UPDATE,DELETE)	182
15.11 Shembull ngarkimi i imazheve.....	190
Teste	196
Referencat.....	202

Pjesa I - Programimi Client Side

Dizenjimi Web është krijimi i faqeve Web dhe siteve duke përdorur HTML, CSS, JavaScript dhe gjuhë të tjera Web. Ai është si dizajni në përgjithësi: është kombinimi i vijave, formave, tekstit, dhe ngjyrave për të krijuar një pamje estetikisht të këndshme. Dizenjimi Web është puna e krijimit të dizajnit për faqet Web.

1. HTML

HTML (Hypertext Markup Language) është një gjuhë shenjash të cilat na lejojnë të shfaqim një përmbajtje të pasur me elementë të ndryshme, tu referohemi burimeve të tjera (sic mund të jenë imazhet, etj.), si dhe të krijojmë lidhje me dokumente të tjera për procesim të mëtejshëm.

HTML përdoret për të krijuar dokumente me strukturë *HyperText*. Një dokument *HyperText* përmban informacion që është i ndërlidhur me dokumente të tjera, duke na lejuar ne të kalojmë nga njëri dokument në tjetrin duke shfrytëzuar të njëjtin aplikacion që po përdornim për të parë dokumentin fillestar.

HTML mund të përdoret gjithashtu për të krijuar dokumente multimediale, p.sh ato që përmbajnë informacion i cili nuk është thjesht tekst si :

Imazhe

Tinguj

Video

Nën programe (plug-ins)

Documentat HTML quhen "Web pages". Browser merr faqet e Web nga Serveri Web që mund të ndodhet kudo në Internet. Për të shkruar HTML mund të përdoren editorët e tekstit si p.sh NotePad.

HTML është aktualisht në versionin 4.01 dhe vazhdoj të ofroj veçori të avancuara për të krijuar faqe me përmbajtje edhe më të pasur.

Është krijuar një specifikim i përputhshëm me HTML, XHTML (Extensible Hypertext Markup Language) e cila përdor sintaksën e XML dhe sjell një XML Schema që mund të përdoret për të vlerësuar një dokument, për të kontrolluar nëse ai është i krijuar saktë etj.

HTML nuk është gjuha e vetme në dispozicion për krijimin e dokumenteve HyperText, ka gjuhë që vinë para dhe pas HTML (SGML, XML, etj), por HTML është bërë gjuha e rekomandimit W3C për krijimin e përmbajtjes për Internetin.

1.1 Bazat e HTML

Dokumentet HTML janë krijuar si dokumente me tekst të thjeshtë (me formatim jo të veçantë) në të cilin i gjithë teksti i formatuar është përcaktuar duke përdorur shenja tekstuale të quajtura *tag-e*. *Tag-et* janë shenjat tekstuale që fillojnë me karakter <, e ndjekur nga emri i *tag*-ut pasuar me çdo atribut tjetër, dhe në fund mbyllet me karakterin > .

Tag-u hapet do të dukej si me poshtë:

```
<TAG>
```

Fundi i *tag*-ut fillon me karakter <, e ndjekur nga karakteri /, i ndjekur nga emri i *tag*-ut dhe karakteri >. *Tag-u* mbyllet do të dukej si më poshtë:

```
</TAG>
```

Tag-et janë *case-sensitive*, kuptimi dhe funksioni i tyre ngelet i njëjtë pavarësisht nëse janë shkruar me gërma të mëdha apo me gërma të vogla.

Shembull:

```
<title> emri i dokumentit </title>
<P> Shembull i përdorimit të tag-eve për të shënuar
tekst </ P>
<B>Bold<I>Italics</I>Bold</B>
```

Atributet e *tag-eve* të cilat tregojnë parametrat e tjera vendosen në *tag-et* hapëse si më poshtë:

```
<TAG ATRIBUT ATRIBUT...>
```

Forma e këtyre attributeve shfaqet ose vetëm me emrin e atributit emri i atributit i ndjekur nga =, ndjekur nga vlera që ne duam të caktojmë atë . Për shembull:

```
<A HREF="http://www.w3.org">Lidhëz</A>
<IMG SRC="image.jpg" BORDER=0 ALT="NAME">
```


Në disa raste, në HTML mund edhe të mos i përdorim *tag*-et mbyllës, kjo në rastet ku ato nuk rrethojnë tekst si p.sh rasti i mësipërm i një imazhi.

1.2 Struktura e dokumenteve HTML

```
<html>
<head>
  <title> vendoset titulli </title>
</head>
<body>

  Permbajtja e dokumentit

</body>
</html>
```

Të gjitha dokumentet HTML kanë pak a shumë të njëjtën strukturë. I gjithë dokumenti duhet të përfshihet në një *tag* HTML dhe është i ndarë në dy pjesë : pjesa e kokës, e përfshire në një *tag* HEAD dhe trupin e dokumentit (që përmban informacionin e dokumentit), i cili është i përfshirë brenda një *tag*-u të quajtur BODY. Koka përmban disa përkufizime të dokumentit: titullin e tij, shënime shtesë, fjalë kyçe etj. Një shembull mund të jetë:

Titulli i dokumentit

Cdo document HTML ka një titull. Për të caktuar titullin e dokumentit përdoret komanda <title>, psh:

```
<title>Dokumenti im I pare ne HTML</title>
```

Titulli vendoset midis tag-ut hapës <title> dhe tagut mbyllës </title>. Titulli vendoset në seksionin e kokës .

Komentet

Në HTML, ne mund të fusim komente në faqe duke e vendosur tekstin brenda *tag*-eve: <!-- dhe --!>. Përmbajtja brenda këtyre dy shenjave injorohet nga *browser*-i dhe nuk shfaqet te përdoruesit.

Paragrafët

Tag-u <P> përdoret për të ndarë paragrafët. Meqenëse HTML injoron ndërprerjen e rreshtave të marra nga skedari origjinal dhe i gjithë teksti është i vazhdueshëm për HTML, ne kemi nevojë për një mekanizëm për të treguar fillim dhe fundin e paragrafëve, ky mekanizëm është siguruar nga <P> dhe </P>.

Tag-u P mund të ketë gjithashtu një atribut ALIGN, që tregon drejtimin e tekstit në paragraf. Kjo mund të marre një nga vlerat e mëposhtme:

LEFT, e pozicionuar në të majtë.

RIGHT, e pozicionuar në të djathtë.

CENTRAL, teksti i qendëruar

Ndërprerje rreshti

Tag-u
 tregon një ndërprerje rreshti. Ai mund të përdoret si një shenjë fillestare dhe nuk kërkon një *tag* mbyllës. BR nuk do të ndryshojë parametrat e përcaktuara për paragrafin në të cilin ne jemi të vendosur në këtë kohë.

Ndarja e teksteve në blloqe

Elementi <DIV> përdoret për të ndarë tekstin në blloqe duke futur një rresht të vetëm midis blloqeve si BR, edhe pse ai mund të ketë cilësi të njëjta, si P, ne mund të përcaktojmë drejtimin e tekstit për secilin bllok

DIV. Parametri ALIGN te DIV merr vlerat e mëposhtme:
LEFT
RIGHT
CENTRAL

Tekst i para-formatuar

Teksti i futur në mes të *tag-eve* <PRE> dhe </ PRE> do të shfaqet nga *browser-i* në përputhje me formatin me të cilin teksti është shkruar duke lënë të pa ndryshuar çdo ndërprerje rresht apo hapësirë. Në shembullin e mëposhtëm mund të shohim përdorimin e disa prej këtyre *tag-eve*:

```
<HTML>
<HEAD>
<TITLE>Rreth Shqipërisë </TITLE>
</HEAD>
<BODY>
<P ALIGN=LEFT>
Shqipëria ka një sipërfaqe prej 28.748 km2. Ajo është e
pozicionuar në pjesën jug perëndimore të gadishullit
Ballkanit dhe kufizohet në veri me Malin e Zi, në veri-
lindje me Kosovën, në lindje me Maqedoninë dhe në jug e
jug -lindje me Greqinë. Në perëndim vendi laget nga
ujërat e deteve Adriatik dhe Jon. Shqipëria ndahet në
katër zona fiziko - gjeografike (Alpet e Shqipërisë,
Krahina Malore Qëndrore, Krahina Malore Jugore dhe
Ultësira Perëndimore). </P>
<DIV ALIGN=RIGHT>
Klima e Shqipërisë është tipike mesdhetare, me verë të
nxehhtë e të thatë dhe dimër relativisht të butë dhe të
lagët. Në pjesët malore, në brendësi të vendit
veçanërisht në veri, dimri është i ashpër dhe me
dëborë, ndërsa vera është e freskët me temperaturë të
këndshme. </DIV>
<DIV ALIGN=CENTER>
Shqipëria është në përgjithësi vend malor. Malet dhe
kodrat zënë 2/3 e sipërfaqes. Maja më e lartë e vendit
është Korabi 2.763 m, në rrethin e Dibrës në veri -
lindje të vendit. Ultësirat të cilat zënë 1/3 e
sipërfaqes shtrihen përgjatë bregdetit Adriatik si dhe
në zonën e Korçës në jug - lindje të vendit. </DIV>
<PRE>
```

Lumenjtë kryesorë janë: Drini, Vjosa, Shkumbini, Semani, Mati, Buna. Prej tyre vetëm Buna, në rrjedhën e poshtme është i lundrueshëm. Më shumë se një e treta e Shqipërisë është e mbuluar nga pyje. Një e treta është kullotë dhe vetëm rreth një e pesta është e kultivuar.

</PRE>

<CENTER>

<P>

Shqipëria ka një pozicion mjaft të përshtatshëm gjeografik për zhvillimin e industrisë turistike me një akses mjaft të volitshëm ndaj tregjeve kryesore evropiane, me një potencial klimaterik shumë të favorshëm, me një shtrirje të gjatë plazhesh ende të pastra, dhe me male të lartë dhe liqene të shumtë të mëdhenj e të vegjël..

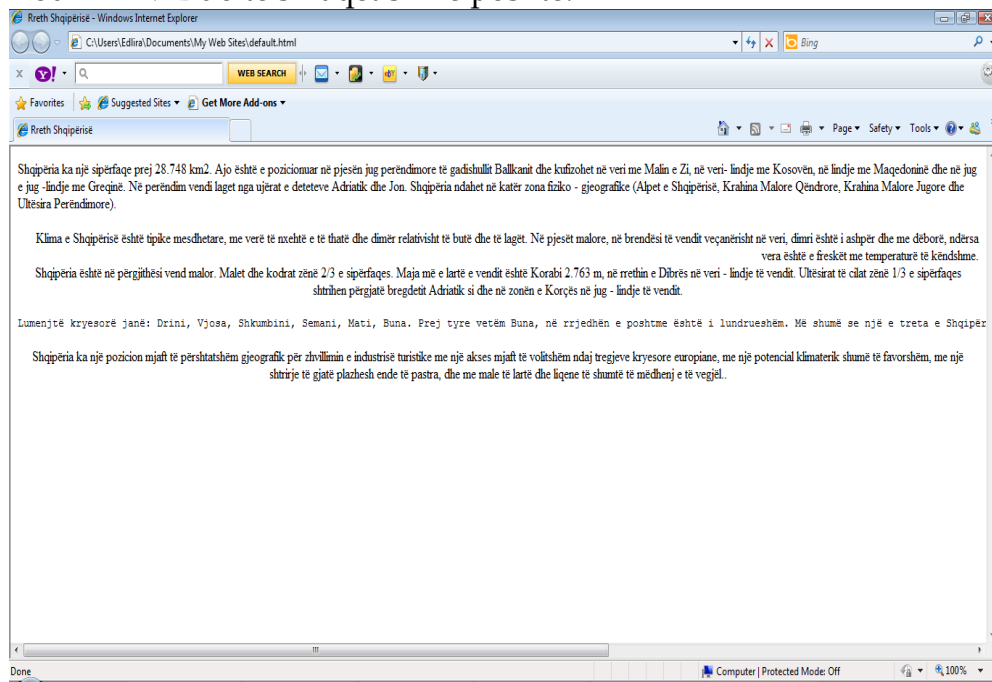
</P>

</CENTER>

</BODY>

</HTML>

Kodi HTML do të shfaqet si më poshtë:



1.3 Fontet

HTML përmban edhe *tag*-e për ndryshimin e attributeve të teksteve tona si fontin dhe ngjyrën.

Headers

Ekziston një element i quajtur `<Hx>` që në HTML mund ta përdorim për të përcaktuar pjesët e tekstit që duam ti konsiderojmë si *headers* (seksione, kapituj etj). Ky *tag* i cakton një madhësi më të madhe tekstit (në bazë të *x*, siç do ta shohim), e bën atë më të nxirë (*bold*) si dhe vendos fillimin e një paragrafi mbas këtij *header*-i.

Madhësia e *header* mund të ndryshojnë nga 1 në 6 dhe kemi 6 *tag*-e të mundshme: H1, H2, H3, H4, H5 dhe H6.

Font-et

Në HTML përdoret *tag*-u FONT për të dhënë tekstit stile , ngjyra dhe efekte të ndryshme të cilat bëjnë që ai të duket më i këndshëm kur të afishohet në faqen tonë të web-it. Në HTML 4.01 ky *tag* nuk përdoret dhe funksioni i tij zëvendësohet duke përdorur CSS të cilën do ta studiojmë në vazhdim.

Font styles

HTML ofron një sërë *tag*-esh të cilat mund të përdoren për ti vendosur stile të ndryshme tekstit që ndodhet brenda këtyre *tag*-eve.

Disa prej tyre janë:

B (bold).

I (italics).

U (underlined).

STRIKE (strikethrough).

SUP (superscript).

SUB (subscript).

BLINK

TT (teletype).

BIG (big).

SMALL (small).

Në HTML, ne mund ti vendosim tekstit edhe disa stile njëkohësisht si mund të jetë *bold* dhe *italic* si më poshtë:

<I>Tekst me dy stile të ndryshme.</I>

Më poshtë është paraqitur një shembull i cili ilustron përdorimin e stileve të sipërpërmendura.

```
<HEAD>
<TITLE> Titulli i dokumentit</TITLE>
</HEAD>
<BODY>
<h1>Header H1</h1>
<h2>Header H2</h2>
<h3>Header H3</h3>
<h4>Header H4</h4>
<h5>Header H5</h5>
<h6>Header H6</h6>
<b>Font size</b> <BR>
<font SIZE=1>1</font> <font SIZE=2>2</font>
<font SIZE=3>3</font> <font SIZE=4>4</font>
<font SIZE=5>5</font> <font SIZE=6>6</font>
<font SIZE=7>7</font> <font SIZE=6>6</font>
<font SIZE=5>5</font> <font SIZE=4>4</font>
<font SIZE=3>3</font> <font SIZE=2>2</font>
<font SIZE=1>1</font>
<P>
<B>Colours</b>
<font COLOR=#800000>C</font><font
COLOR=#000080>O</font>
<font COLOR=#000080>L</font><font
COLOR=#008000>O</font>
<font COLOR=#00FFFF>R</font><font
COLOR=#FF0000>E</font>
<font COLOR=#C0C0C0>S</font> . <font
COLOR=#800080>D</font>
<font COLOR=#008080>E</font> . <font
COLOR=#FF0000>L</font>
<font COLOR=#808080>E</font><font
COLOR=#FF00FF>T</font>
<font COLOR=#00FF00>R</font><font
COLOR=#808000>A</font>
<font COLOR=#FFFF00>S </font>
<P> <b>Bold</b> <br> <i>Italics</i> <br>
<u>Underlined</u><br>
```

```
<strike>Strikethrough</strike> <br>
A<sup>Superscript</sup> <br>
B<sub>Subscript</sub><br>
<blidhëz>Blidhëzing</blidhëz> <br>
<tt>Typewriter (Teletype)</tt> <BR> <big>Big
text</big> <br> <small>Small text</small>
</BODY>
</HTML>
```

Më poshtë jepet rezultati që shfaqet në *browser* pas ekzekutimit të kodit të mësipërm.

Header H1

Header H2

Header H3

Header H4

Header H5

Header H6

Font

size

1 2 3 4 5 6 7 6 5 4 3 2 1

Colours CO LO RE S . D E . L E T R A S

Bold

Italics

Underlined

~~Strikethrough~~

A^{Superscript}

B_{Subscript}

Blidhëzing

Typewriter(Teletype)

Big

text

Small text

Vendosja e karaktereve të veçanta

HTML përdor disa kode të veçanta për të mundësuar shtypjen e karaktereve të cilat nuk i shkruajmë dot nga tastiera. Tabela e mëposhtme paraqet kodet dhe simbolin që ato përfaqësojnë.

Kodi	Rezultati
á, Á, é, É,...	á,Á,é,É,...
¿	¿
¡	¡
º	º
ª	ª
™ or ™	Trademark
©	Copyright
®	Registered
 	(non-breaking space)
<	<
>	>
&	&
"	"

1.4 Linket

Një nga tiparet kyçe të Web-it, e cila e ka bërë dhe më të suksesshëm atë, është lidhja ndërmjet dokumenteve të cilat ndodhen në servera të ndryshëm. Lidhja mund të bëhet me imazhe, audio, video si dhe me faqe web-i të tjera. Lidhjet mund të krijohen duke përdorur *tag*-un A dhe atributet e tij NAME, HREF, TARGET.

Nëse lidhëza bën lidhjen e faqes me një faqe tjetër, ku kjo e fundit është pjesë e të njëjtit site atëherë lidhëza njihet si linke brendshme. Nëse lidhëza ju lidh me një faqe të re, por te site tjetër atëherë lidhëza njihet si linke jashtme. Gjithashtu mund të krijojmë një linke cila të na çoj në një pjesë tjetër të së njëjtës faqe.

Një link specifikohet me anë të *tag*-ut <a>. Çdo gjë që shkruhet midis *tag*-ut hapës <a> dhe *tag*-ut mbyllës do të jetë pjesë e lidhëzës e cila pasi klikohet nga përdoruesi do të kaloj në faqen e re.

Linket në një faqe te re WEB

Për të lidhur një faqe me një faqe tjetër nevojitet që *tag*-u hapës <a> të përmbaj atributin *href* dhe vlera e atributit është emri i skedarit për tek i cili nevojitet të kalojmë.

Një shembull mund të jetë:


```
<body>
  <p>Kthehu tek <a
href="shembull1.html">Shembulli i pare</a></p>
</body>
```

Nevojitet që të dy skedarët (fshembulli1.html dhe shembull2.html) të gjenden në të njëjtën dosje. Kur të klikojmë mbi link [Linku qe un krijova](#) do të hapet faqja shembull1.html në të njëjtën dritare.

Nëse duam të kalojmë në një faqe të re të një site tjetër përsëri do të përdorim elementin <a> dhe atributin *href* por kësaj here duhet që si vlerë të atributit href të vendosim URL-në e plotë faqes ku duam të shkojmë. Si p.sh :

```
<body>
  <p>A doni te vizitoni <a
href="http://www.uamd.edu.al/">faqen e UAMD-
se</a>?</p>
</body>
```

Kur të krijoni lidhëza kini parasysh që të përdorni fjalë të cilat përkufizojnë në mënyrë sa më koncize faqen ku ju do të drejtoheni. Nga ana tjetër për të ndihmuar përdoruesin nga ana vizuale këshillohet që linket të ngjyrosen në mënyrë që të bëhen më të dukshme midis fjalëve të tjera të përmbajtjes së faqes.

Shumë dizenjues të web-it përdorin imazhet brenda elementit <a>. Në rast se përdorni figurat duhet që ajo të jetë shpjeguese e mirë e faqes tek e cila të drejton.

Brenda elementit <a> mund të përdoret edhe atributi *title* dhe vlera e tij do të jetë shpjegimi që ju doni të shfaqet në rastin se afrojmë mousin mbi lidhëz. Ky trik është i vlefshëm sidomos në rastet kur lidhëza është figurë.

Kështu p.sh. shtojmë në shembullin tonë rreshtat e mëposhtëm:

```
<p>
<a href=http://www.google.com/ title="Kerkimi ne
Google"> Google </a> eshte nje motor kerkimi shume
i fuqishem.
</p>
```

Lidhja e adresave email

Në disa raste kemi parë që në faqe shfaqen adresat e personave të ndryshëm dhe pasi klikoni mbi to hapet një dritare e cila jep mundësinë e dërgimit të një emaili adresës së sapo klikuar.

Për të krijuar një link mbi një adresë email-i duhet të shtoni atributin *href* i cili si vlerë duhet të ketë:

```
<a href=mailto:emer@shembull.com>emer@shembull.com</a>
```

1.5 Listat

Në HTML kemi dy lloje listash:

Unordered list që janë në formën e pikave të plota të mbushura

Ordered list që janë një sekuencë me numra apo shkronja

Shembull i listave të parenditura

```
<html>
<body>
<h4>List e parenditur:</h4>
<ul>
  <li>Kafe</li>
  <li>Caj</li>
  <li>Qumesht</li>
</ul>
</body>
</html>
```

List e parenditur:

- Kafe
- Caj
- Qumesht

Shembull i listave te renditura

```
<html>
<body>
<h4>Liste e renditur:</h4>
<ol>
  <li>Kafe</li>
  <li>Caj</li>
  <li>Qumesht</li>
</ol>
</body>
</html>
```

Liste e renditur:

1. Kafe
2. Caj
3. Qumesht

Shembull i listave te nderthuruara

```
<html>
<body>
<h4>List e nderthurura:</h4>
<ul>
  <li>Kafe</li>
  <li>Caj
    <ul>
      <li>Caj i Zi</li>
      <li>Caj Jeshil</li>
    </ul>
  </li>
  <li>Qumesht</li>
</ul>
</body>
</html>
```

List e nderthurura:

- Kafe
- Caj
 - Caj i Zi
 - Caj Jeshil
- Qumesht

1.5 Imazhet

Për të vendosur imazhet ose grafikët në një faqe Web përdoret një *tag* i vetëm i quajtur ``. `` ka disa attribute të cilat përcaktojnë se cilin skedar imazhi do të përdorim, madhësinë e tij etj. Atributi i cili përcakton imazhin që do të shfaqet është SRC. Me anë të këtij *tag*-u ne mund të përcaktojmë një URL për skedarin e imazhit që do të kërkohet në server për tu paraqitur në browser. Imazhi mund të jetë i ruajtur në çfarëdo direktorie mjafton që vlera që ne do të japim *tag*-ut SRC të jetë një URL.

Nëse duam që të shtojmë një tekst imazhit tonë përdorim atributin ALT.

Atributet WIDTH dhe HEIGHT përdoren për të përcaktuar përmasat me të cilat duam që të shfaqet imazhi në faqen e Web-it. Nëse ne nuk i përcaktojmë këto përmasa, browseri do ta shfaqë imazhin me përmasat që ka vetë imazhi.



Ky imazh do të shfaqej në browser më anë të kodit të mëposhtëm.

```
<HTML>
<HEAD>
<TITLE>Document title</TITLE>
</HEAD>
<BODY>
<IMG SRC="html.jpg"> <P>
</BODY>
</HTML>
```

Imazhet mund të krijohen në mënyra dhe formate të ndryshme. Browseri njeh formatet GIF, JPEG dhe PNG. Për të shmangur mungesat është mirë të mos përdorim imazhe shumë të mëdha.

Ushtrime

1. Ndërtoni një faqe web e cila të përmbajë tekstet si më poshtë: emrin tuaj me ngjyrë jeshile dhe font Tahoma. Një paragraph me 4-5 fjali ku secila prej tyre ka një formatim ndryshe.
2. Ndërtoni një faqe web e cila afishon përshkrimin e një libri, sipër përshkrimit ndodhet titulli I librit I cili është I nënvijëzuar dhe me tekst të theksuar. Në fund të përshkrimit vendosni një imazh I cili përfaqëson kapaku i librit.
3. Afishoni në një faqe web një imazh I cili ka një border me madhësi 2, gjerësi dhe lartësi 200.
4. Ndërtoni një faqe web ku të ruani profilin Tuaj në formën e një CV të shkurtuar. Vendosni aty një foto Tuajën. Përdorni headings dhe paragrafët. Përdorni, tagun <a> për të krijuar lidhje me profilin Tuaj në facebook apo linked in. Përdorni listat për të ilustruar njohuritë Tuaja kompjuterike dhe të gjuhëve të huaja (gjuha, sqarimi, p.sh anglisht, shumë mirë , si në llojin e tretë të listave).

2. Tabelat

HTML ka një grup *tag*-esh të cilat përdoren për të paraqitur tekstin në formën e një table. Këto *tag*-e janë:

- TABLE: shënon fillimin dhe mbarimin e një table
- TR: shënon fillimin dhe mbarimin e një rreshti
- TH: shënon fillimin dhe mbarimin e një qelize në kokën e tabelës
- TD: shënon fillimin dhe mbarimin e një qelize
- CAPTION: përdoret për të vendosur tabelës një titull

Kodi për një tabelë të thjeshtë do të ishte:

```
<TABLE>
<TR><TH>Header 1</TH>...<TH>Header n</TH></TR>
<TR><TD>Cell 1.1</TD>...<TD>Cell n</TD></TR>
...
<TR><TD>Cell 1.1</TD>...<TD>Cell n</TD></TR>
<CAPTION>Title</CAPTION>
</TABLE>
```

Sic mund të shohim në pjesën e kodit tabela vendoset brenda *tag*-eve TABLE. Çdo rresht duhet të vendoset brenda *tag*-eve <TR> dhe </TR>. Për të paraqitur qeliza të veçanta kemi dy mundësi: duke përdorur *tag*-un <TH> ose <TD>. Dallimi është se mundësia e parë përdor tekst *bold* dhe e qendërzon kolonën.

Tag-u TABLE ka disa attribute të cilat shërbejnë për të dhënë tabelës formatin që na nevojitet.


- BORDER: përcakton madhësin e konturit të qelizës.
- CELLSPACING: përcakton madhësinë në pika të hapësirës ndërmjet qelizave.
- CELLPADDING: përcakton distancën në pika ndërmjet përmbajtjes së një qelize dhe borderit të sajë.
- WIDTH: specifikon gjerësinë e tabelës, mund të paraqitet me pika ose në përqindje duke u bazuar në raportin që ajo ka me gjerësinë totale. P.sh 100% përfaqson gjerësinë e të gjithë dritares së browserit.
- ALIGN: pozicionon tabelën në lidhje me faqen, në të majtë (LEFT), në të djathtë (RIGHT) ose në qendër (CENTER).
- BGCOLOR: përcakton ngjyrën e tabelës.

Shembull tabelle në të cilën përfshihen edhe atributet e mësipërme:

1.1 dhe 1.2		1.3
2.1 dhe 3.1	2.2	2.3
	3.2	3.3

Tabele e Thjeshte

Tabele me ngjyra dhe imazh

	Maj	Qershor	Korrik	Gusht
	22	23	3	29
	1234	1537	7	1930
	11000	13000	-500	60930

```
<HTML>
<HEAD>
<TITLE>Document title</TITLE>
</HEAD>
<BODY>
<TABLE BORDER=1>
<TR>
<TD COLSPAN=2>1.1 dhe 1.2</TD>
<TD>1.3</TD>
</TR>
<TR>
<TD ROWSPAN=2>2.1 dhe 3.1</TD>
<TD>2.2</TD>
<TD>2.3</TD>
</TR>
<TR>
<TD>3.2</TD>
<TD>3.3</TD>
</TR>
```



```
<CAPTION ALIGN=bottom><strong><br>Tabele e
Thjeshte</strong></CAPTION>
</TABLE>
<p><strong>Tabele me ngjyra dhe imazh</strong></p>
<TABLE BORDER=0 CELLSPACING=0 BGCOLOR=#0000FF>
<TR><TD>
<TABLE BORDER=0 CELLSPACING=1 CELLPADDING=2
WIDTH=400 BGCOLOR=#FFFFFF>
<TR>
<TH><IMG SRC="css_logo.png" height="53"></TH>
<TH>Maj</TH>
<TH>Qershor</TH>
<TH>Korrik</TH>
<TH>Gusht</TH>
</TR>
<TR>
<TD BGCOLOR=#A0A0A0>&nbsp;</TD>
<TD>22</TD>
<TD>23</TD>
<TD>3</TD>
<TD>29</TD>
</TR>
<TR>
<TD BGCOLOR=#A0A0A0>&nbsp;</TD>
<TD>1234</TD>
<TD>1537</TD>
<TD BGCOLOR=#FFa0a0>7</TD>
<TD>1930</TD>
</TR>
<TR>
<TD BGCOLOR=#A0A0A0>&nbsp;</TD>
<TD>11000</TD>
<TD>13000</TD>
<TD BGCOLOR=#FF4040>-500</TD>
<TD BGCOLOR=#a0a0FF>60930</TD>
</TR>
</TABLE>
</TD></TR>
</TABLE>
</BODY>
</HTML>
```

thead, tfoot, dhe tbody

Tag-u `<thead>` përdoret për të grupuar përmbajtjen e headerit të një tabelë HTML. Elementi `thead` duhet të përdoret bashkë me elementët `tbody` dhe `tfoot`.

Elementi `tbody` përdoret për të grupuar përmbajtjen e body në një tabelë HTML dhe tag-u `tfoot` përdoret për të grupuar përmbajtjen e footer-it.

Në një tabelë tag-u `<tfoot>` duhet të shfaqet para `<tbody>` në mënyrë që një browser të mund shfaqë footer-in përpara se të marrë të dhënat e të gjithë rreshtave.

Shembull i përdorimit të tag-eve **thead, tfoot, dhe tbody**:

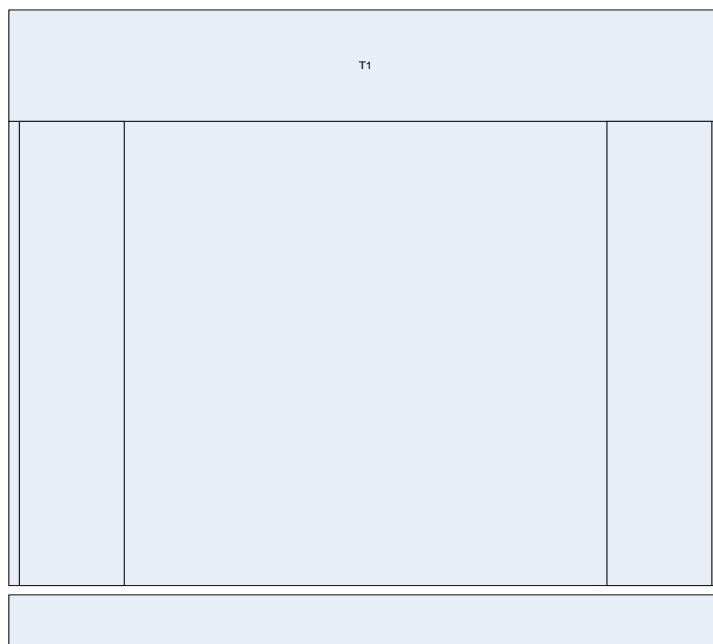
```
<table border="1">
  <thead>
    <tr>
      <th>Month</th>
      <th>Savings</th>
    </tr>
  </thead>
  <tfoot>
    <tr>
      <td>Sum</td>
      <td>$180</td>
    </tr>
  </tfoot>
  <tbody>
    <tr>
      <td>January</td>
      <td>$100</td>
    </tr>
    <tr>
      <td>February</td>
      <td>$80</td>
    </tr>
  </tbody>
</table>
```

2.1 Dizënjimi i faqes nëpërmjet tabelave

Tabelat në website janë përdorur për dy arsye :

- Për të organizuar dhe shfaqur informacionin në formën e një tabele, kur një gjë e tillë është e nevojshme.
- Për të krijuar layoutet e një faqe duke përdorur tabelat e fshehura.
- Përdorimi i tabelave për të ndarë një faqe në seksione të ndryshme është një mjet shumë i fuqishëm. Kryesisht në lidhje me layoutet, tabelat përdoren për të funksionet e mëposhtme:
- Për të ndarë faqen në seksione të ndryshme
Për të krijuar menu, zakonisht përdoret një ngjyrë për pjesën header dhe një tjetër për rreshtin tjetër ku gjenden linket.
- Për të shtuar fusha formash interactive. P.sh opsjonin e kërkimit.
- Krijohen header të cilët ngakohen më shpejt.
- Pozicionim më i lehtë i imazheve të cilat janë ndarë në copa të vogla.
- Një menyrë më e lehtë për ta shfaqur tekstin në kolona.
Analizojmë një faqe në web. P.sh. moh.gov.al e cila ka pamjen e mëposhtme:

Vëmë re që është një bashkësi tabelash brenda njëra tjetrës. Tabela e madhe ka gjerësi fikse për efektet design.



Ushtrime

1. Ndërtoni një faqe të thjeshtë HTML e cila përmban një tabelë që paraqet orarin tuaj javor. Çdo kolonë e cila përfaqëson edhe ditët e javës të shfaqet me një ngjyre të ndryshme.
2. Ndërtoni një faqe HTML layout-i i të cilës është ndërtuar me anë të një tabele me tre rreshta ku rreshti i parë përfaqëson header-in e faqes dhe brenda tij vendoset një imazh në formën e banerit. Rreshti i dytë, trupi i faqes, është i ndarë në tre kolona ku në kolonën e parë kemi të paraqitur me anë të listave katër herë fjalën "menu1". Kjo përfaqëson menunë vertikale. Në rreshtin e tretë vendosni "copyright" e cila përfaqëson footer-in e faqes.

3. Format

Format janë elementë HTML të cilat përdoren për të marrë informacion nga përdoruesi. Një form mund të përmbajë elementë inputi të tilla si fusha teksti, checkboxe, radio-buttona, butona submit etj.

Një mënyrë për të krijuar një formë është kjo:

```
<FORM ACTION="url process" METHOD="POST">
...
Elements
..
</FORM>
```

Atributet e Formave

- **ACTION:** ky atribut përcakton URL-në ku do të dërgohen të dhënat që do të shtypi përdoruesi. Atributi action tregon se çfarë i ndodh të dhënave kur shtypet butoni i dërgimit. Zakonisht vlera e atributit është një faqe ose një program në një web server që do të marrë dhe procesoj të dhënat e dërguara.

Për shembull, nqs keni një form logimi që kërkon një username dhe një password; këto të dhëna që fut përdoruesi do të kalojnë në një faqe login.php dhe në këtë rast vlera e atributit action i faqes sonë do të jetë `<form action="http://www.siteJuaj.org/login.php">`. Si URL mund të përdoret një adresë emaili p.sh: <mailto:address@e.mail> ose një HTTP URL <http://www.uamd.edu/form.html>.

- **METHOD:** metoda përcakton mënyrën se si të dhënat do të dërgohen. Ekzistojnë dy mundësi GET dhe POST. Metoda get i dërgon të dhënat si pjesë të URL. Metoda post i fsheh të dhënat në dicka të njohur si pjesë të headerit HTTP.

- **ENCTYPE:** përcakton tipin e kodimit të përdorur.

Atributi id

Atributi id ju lejon ju të identifikohen në mënyrë unike elementët brenda një elementi <form> ashtu sic ju identifikoni në mënyrë unike një element në një faqe.

Është praktikë e mirë që ju ti specifikoni një element id cdo elementi formë sepse shumë forma përdorin file për style dhe skripte të cilët kërkojnë përdorimin e atributit id në mënyrë që të bëhet dallimi i tyre.

Vlera e atributit id duhet të jetë unike brenda një dokumenti. Disa persona vendosin si vlerë të atributit id dhe name për formën karakteret frm dhe më pas përshkruajnë të dhënat si psh në rast të një formë logimi përdoret frmLogin apo në rast të një forme kërkimi frmKërkim etj.

Atributi name (deprecated)

Ashtu sic e kemi parë tashmë përdorimin e këtij atributi nëpër elementë të tjerë, atributi name është paraardhësi i atributit id dhe vlera e tij duhet të jetë unike në të gjithë dokumentin.

Në mënyrë përgjithësuese nuk do të shihni përdorim të atributit name porse nqs do tju duhet ta përdorni ath këshillohet që si vlerë të tij të vendosni vlerën që keni vendosur për atributin id. E ngjashme me atributin id këshillohet që si vlerë tek value të vendosni frm_qëllimiFormës si psh frmKërkimi apo frmLogimi.

3.1 Elementet e Formës

HTML ofron një gamë të gjerë elementësh që përdoren për input në forma. Ato mund të përdoren për funksione të ndryshme si për shkruajtur tekst apo për dërgim skedarësh.

<INPUT>

Elementi INPUT është më i përdoruri dhe përdoret si një fushe për të marrë të dhëna. Ekzistojnë disa lloje të ndryshme të elementit INPUT në varësi të vlerës që merr atributi TYPE:

- TYPE=RADIO: lejon që të zgjedhin nga një rang mundësish, por vetëm një në një kohe.
- TYPE=RESET: fshin të gjithë formën.
- TYPE=TEXT: i lejon përdoruesit që të fus një rresht tekst.

- TYPE=PASSWORD: i lejon përdoruesit që të fus një rresht tekst i cili paraqitet si

"*" në vend të tekstit. Zakonisht përdoret për pjesën ku do të shkruhet password-i.

- TYPE=CHECKBOX: na lejon të zgjedhim një ose më shumë opsione.

- TYPE=SUBMIT: merr të dhënat e futura në formë dhe kryen veprimin e caktuar.

- TYPE=HIDDEN: një fushë teksti e cila nuk i shfaqet përdoruesit. Përdoret për të ruajtur vlera.

Elementi INPUT ka edhe disa attribute opsionale:

- NAME: emërton fushën. Kjo është e rëndësishme për tu përdorur në kod për përpunime të tjera.

- VALUE: i vendos një vlerë fillestare fushës.

SELECT

SELECT përdoret për të zgjedhur një ose më shumë nga opsionet e mundshme.

Një shembull do të ishte:

```
<SELECT name="destination">
```

```
<option> Africa
```

```
<option> Antarctica
```

```
<option> America
```

```
<option> Asia
```

```
<option> Europe
```

```
<option> Oceania
```

```
</SELECT>
```

Atributet e elementit SELECT janë:

- SIZE: Nëse SIZE ka vlerën 1, vetëm një nga opsionet do të shfaqet, nëse vlera është më e madhe se 1 përdoruesit do ti shfaqet një listë me zgjedhje.

- MULTIPLE: përdoruesit mund të zgjedhin më shumë se një opsion, nëse kjo është e zgjedhur.

Elementi OPTION ka dy attribute:

- VALUE: vlera që do ti caktohet variabëliti kur të jete zgjedhur ky opsion.

- SELECTED: ky opsion zgjidhet vetvetiu.

TEXTAREA

TEXTAREA përdoret për të marrë nga përdoruesi disa rreshta tekst.

Formati i sajë është si më poshtë:

```
<TEXTAREA name="comments" cols=30 rows=6>
```

Jepni përshtypjet tuaja rreth faqes tonë!

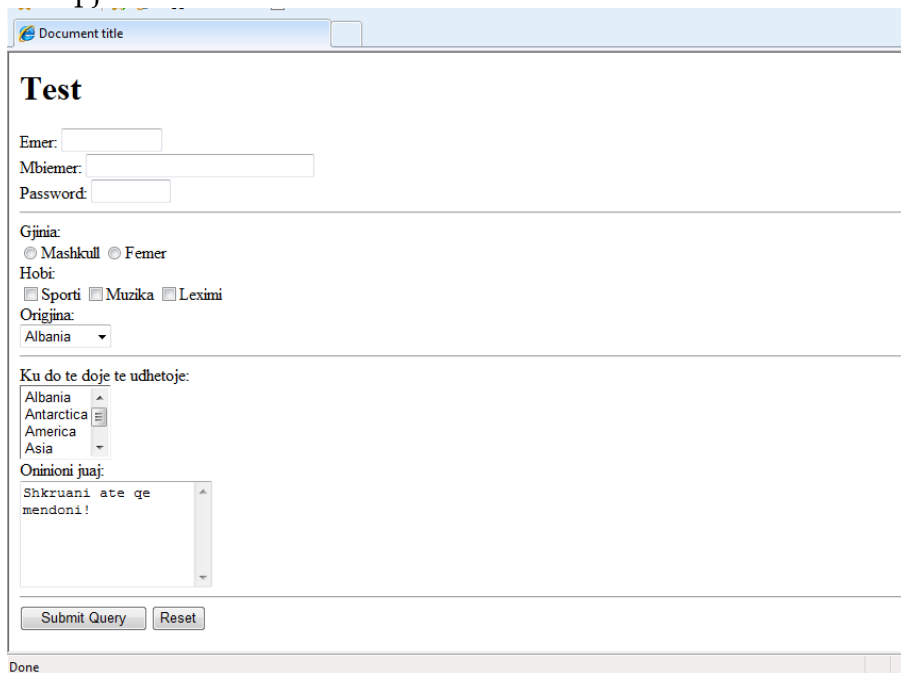
```
</TEXTAREA>
```

Përmbajtja që vendoset ndërmjet <TEXTAREA> dhe </TEXTAREA> përbën vlerën fillestare të kësaj fushe.

Atributet e TEXTAREA janë:

- ROWS: rreshtat që do të merren nga kutia e tekstit.
- COLS: kolonat.

Tani do të shohim një shembull i cili përdor elementet që pamë në këtë pjesë.



The screenshot shows a web browser window with a document titled "Document title". The main content area is titled "Test" and contains a form with the following elements:

- Form fields for "Emër:" (Name), "Mbiemër:" (Surname), and "Password:".
- A "Gjinia:" (Gender) section with radio buttons for "Mashkull" (Male) and "Femer" (Female).
- A "Hobi:" (Hobby) section with checkboxes for "Sporti" (Sports), "Muzika" (Music), and "Leximi" (Reading).
- An "Origjina:" (Origin) dropdown menu currently set to "Albania".
- A "Ku do te doje te udhetoje:" (Where would you like to travel) section with a list box containing "Albania", "Antarctica", "America", and "Asia".
- An "Opiniononi juaj:" (Your opinion) section with a text area containing the text "Shkruani ate qe mendoni!" (Write what you think!).
- Two buttons at the bottom: "Submit Query" and "Reset".

The browser's status bar at the bottom shows "Done".

Kodi HTML nga i cili kemi marrë rezultatin e mësipërm është:

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>Titulli i Dokumentit</TITLE>
```

```
</HEAD>
<BODY>
<H1>Test</H1>
<FORM METHOD=GET>
Emer: <INPUT TYPE=TEXT NAME=NAME SIZE=10><BR>
Mbiemer: <INPUT TYPE=TEXT NAME=SURNAME SIZE=30><BR>
Password:      <INPUT      TYPE=PASSWORD      NAME=PASS
SIZE=8><BR>
<HR>
Gjinia: <BR>
<INPUT TYPE="RADIO" NAME="Gender">Mashkull
<INPUT TYPE="RADIO" NAME="SEXO">Femer
<BR>
Hobi:<BR>
<INPUT TYPE="CHECKBOX" NAME="SPORT">Sporti
<INPUT TYPE="CHECKBOX" NAME="MUSICA">Muzika
<INPUT TYPE="CHECKBOX" NAME="LECTURA">Leximi <BR>
Origjina:<BR>
<SELECT name="ORIGIN">
<option> Albania
<option> Antarctica
<option> America
<option> Asia
<option> Europe
<option> Oceania
</SELECT>
<HR>
Ku do te doje te udhetoje:<BR>
<SELECT name="destination" MULTIPLE SIZE=4>
<option> Albania
<option> Antarctica
<option> America
<option> Asia
<option> Europe
<option> Oceania
</SELECT>
<BR>
Oninion jua j:
<BR>
<TEXTAREA COLS=20 ROWS=6 NAME="YOUR OPINION">
Shkruani atë që mendoni!
</TEXTAREA>
<HR>
```

```
<INPUT TYPE=SUBMIT> <INPUT TYPE=RESET>  
</FORM>  
</BODY>  
</HTML>
```

Butonat

Butonat përdoren në rastet më të shpeshta për të dërguar të dhënat e një forme, si dhe nganjëherë për të pastruar të dhënat nga një formë. Ju mund të krijoni butona në tre mënyra:

Duke përdorur elementin *<input>* me atributin *type* vlera e të cilit është *submit*, *reset* ose *button*.

Krijimi i butonit duke përdorur elementin <input>

Kur ju përdorni elementin *<input>* për të krijuar një buton, tipi I butonit që ju krijoni specifikohet në atributin *type*. Atributi *type* mund të marrë vlerat e mëposhtme për të krijuar butonin:

Submit, I cili krijon një buton që dërgon të dhënat e formës

Reset, I cili krijon një buton që në mënyrë automatike bën boshatisjen e kontrolleve të formës nga vlerat tyre inicializuese që plotësohen gjatë shkarkimit të faqes.

Button, i cili krijon një buton që përdoret për të startuar një script të anës klient kur përdoruesi klikon mbi një buton.

Ushtrime

Ndërtoni një faqe të thjeshtë në të cilën të vendosni dy textbox-e në të cilat ti kërkonit përdoruesit emrin dhe mbiemrin. Emërtojeni faqen revista.html.

Në faqen revista.html të cilën e krijuat në ushtrimin e parë, shtoni disa fusha të tjera të cilat duhet ti plotësojë përdoruesi si adresa, qyteti, shteti, kodi zip.

Duke supozuar që kjo faqe do të shërbejë për abonimin në një revistë, shtoni një dropdown list me pesë emra revistash të njohura si dhe dy radio butтона me anë të të cilave ne zgjedhim abonim një vjekar apo dy vjekar.

Shtoni një textbox në të cilin abonuesi të lërë komentet e tij.

Faqja do të duket si më poshtë:

Name:	<input type="text"/>
Address:	<input type="text"/>
City:	<input type="text"/>
State:	<input type="text"/>
Zip:	<input type="text"/>
Magazine:	<input type="text" value="Newsweek"/> ▼
Subscription:	<input checked="" type="radio"/> 1 year <input type="radio"/> 2 years
Additional Comments:	<div><div></div></div>
<input type="button" value="Send Your Order"/>	

4. CSS

CSS qëndron për Cascading Style Sheets. Stilet përcaktojnë si do të paraqiten elementet HTML. Kjo realizohet duke i bashkangjitur atribut prezantimi në çdo element HTML ose nënklasë të sajë.

Për shembull nëse ne duam që të gjithë paragrafët të kenë një background me ngjyrë të kuqe dhe tekst me ngjyrë të verdhë, do të përdornim kodin e mëposhtëm:

```
<STYLE TYPE="text/css"> P {color: red;
background:yellow;} </STYLE>
```

Nëse duam të përcaktojmë stilet që do të përdoren në një faqe përdorim *tag-un* STYLE , ndërsa *tag-u* LINK shërben për të treguar skedarin e jashtëm i cili mban stilet që do të përdorim ne në faqe.

Tag-u STYLE duhet të vendoset në kokën e faqes.

Parametri TYPE përdoret për të treguar sintaksën që do të përdorim për të përcaktuar stilet, në rastin tonë ne do të përdorim text/css.

Tag-u LINK përdoret si më poshtë:

```
<LINK REL="stylesheet" HREF="miweb.css"
TYPE="text/css">
```

Stilet ruhen zakonisht në një skedar të jashtëm me prapashtesën .css. Stilet e ruajtura jashtë bëjnë të mundur që ju të ndryshoni pamjen e të gjitha faqeve të një Web siti duke edituar një skedar të vetëm.

Formati i Style sheet

Sic pamë edhe më lartë formati i një style sheets është si më poshtë:

```
<element>{<format>}
```

Per shembull:

```
P {color: red; background:yellow;}
```

Sintaksa e CSS është case sensitive. Kjo sintaks na lejon që të përcaktojmë formatin që duam të kemi në paragrafët e website-it tonë. Në HTML 4.0 është shtuar një atribut CLASS i cili na mundëson krijimin e klasave elementësh tek të cilat do të zbatohet stili.

Për shembull për të përcaktuar një klas parafrafi të cilën e emërtojm *ifformatuar* do të shkruanim:

```
P. ifformatuar {color: red; background:yellow;}  
<P CLASS=" ifformatuar ">Paragraf i formatuar.</P>  
<P>Paragraf normal</P>
```

Ekziston edhe një metodë për caktimin e një stili të veçantë parafrafëve të veçanta. Për këtë duhet të përcaktojmë stilin e elementëve të HTML me anë të CSS duke përdorur sintaksën e mëposhtme:

```
#paragraph1 {color: green; background:yellow;}  
Këto identitete mund tja bashkëngjisim një elementi  
HTML duke përdorur atributin ID.  
<p CLASS="destacado">Nje parafrafi i theksuar</P>  
<P>Paragraf normal</P>  
<P CLASS="highlighted" ID="paragraph1">Paragraf i  
theksuar por duke përdorur atributin ID</P>
```

4.1 SPAN dhe DIV

Më parë, pamë se si të caktojmë stile të elementëve të HTML, por ne ndonjëherë duam të caktojmë stile në seksione të tekstit ose përmbajtje që nuk janë pjesë e një blloku HTML.

Kjo realizohet me anë të *tag*-eve DIV dhe SPAN.

Nëse duam të etiketojmë një pjesë të përmbajtjes në mënyrë që ti japim asaj stilin që dëshirojmë, duhet që këtë përmbajtje ta vendosim brenda një *tag*-u SPAN ose DIV. Në ndryshim nga SPAN, DIV vendos një ndërprerje rreshti në fillim dhe në fund të çdo seksioni duke krijuar në këtë mënyrë blloqe me tekst pa pasur nevoja që ti vendosim ato në *tag*-e të tjera si parafrafi.

4.2 Tipare të tjera te CSS

Disa nga veçorit që shërbejnë për të përcaktuar pamjen e tekstit janë:

- **font-family:** lloje shkrimi sic mund të jenë: serif, cursive, sans-serif, fantasy apo monospace. Ne mund ti caktojmë tekstit një lloj shkrimi ose disa duke i ndarë ato me presje nga njëra tjetra. Duhet pasur

kujdes që fontet që ne do të përcaktojm të jenë të instaluara në kompjuter.

- **font-size**: madhësia e fontit. xx-small, x-small, small, medium, large, x-large, xx-large si dhe vlera numerike.
- **font-style**: stili i fontit. Mund të përdoret normal, italic, italic small caps, oblique, oblique small caps dhe small caps.

Veçoritë e mëposhtme përdoren në blloqet e teksteve sic mund të jenë paragrafët.

- **margin-top, margin-right, margin-bottom, margin-left**: distanca minimale ndërmjet një blloku dhe elementit ngjitur.
- **padding-top, padding-right, padding-bottom, padding-left**: mbush në hapësirën në mes të kufirit dhe përmbajtjen e bllokut.
- **border-top-width, border-right-width, border-bottom-width, border-left-width**: gjerësia e konturit të bllokut në vlera numerike.
- **border-style**: stili i konturit të bllokut. none, solid ose 3D.
- **border-color**: ngjyra e konturit.

4.3 Trashëgimia

Është një ndër cilësitë më të fuqishme të CSS dhe ka domethënie se njëherë që kemi specifikuar një property për një element kjo property do të trashëgohet dhe tek elementët të cilët janë fëmijë të elementit të mëlartmë që ka të specifikuar cilësinë në fjalë. Për shembull nqs kemi specifikuar `font-family` për elementin `<body>` ath këtë property do ta trashëgojnë të gjithë elementët që përmbahen tek body I faqes. Kjo cilësi është mjaft lehtësuese për kodin tuaj meqenëse nuk ju duhet ta ripërcaktoni në cdo element të faqes porse vetëm në një element “prind”.

Por duhet të bëni kujdes! Nëse një property cilësohet në mënyrë të veçantë për një element atëherë atributi i trashëguar prej elementit prind do të bëhen overwrite me proprietinë e cilësuar në nivel të atij elementi. Në shembullin tonë të parë pamë se tipi I fontit I përcaktuar në nivel elementi `<body>` ishte arial. Porse kishte disa qeliza të vecanta të tabelës që përdornin tipin e fontit Courier. Këto qeliza ishin qelizat të cilat kishin të shoqëruar rregullat e class-es code. Pra edhe pse këto qeliza duke qenë “fëmijë” të elementit `<body>` trashëgojnë cilësinë e tipit të fontit Arial prej “prindit” të tyre; arrijnë

të mbishkruajmë këtë cilësi me cilësinë e Courier e cila përcaktohet në nivel elementi tek `code`. Kjo ndodh sepse selektuese duke qenë specifikisht I përcaktuar për `td.class` është në gjendje të paraprij atributet e vendosura në nivel më të lartë hierarkie.

Galeritë e Imazheve me CSS

CSS mund të përdoret edhe për të krijuar galeri imazhesh. Më poshtë është paraqitur kodi për një galeri imazhi. Galeria përmban tre imazhe të emërtuara `image1`, `image2`, `image3` dhe të ruajtuar në dosjen me emrin `image`.

```
<html>
<head>
<style>

#gallerywrapper{
    float:left;
    position:relative;
    z-index:5;
}

.gallerydisplay {
    padding:0;
    margin:0px 0 0 0;
    list-style-type:none;
    float:left;
}

.gallerydisplay img.mainimage,
#defaultimage img.mainimage{
    border:0;
    width:100px;
    height:100px;
}

.gallerydisplay li {
    float:left;
    margin:10px 0 10px 0;
}
```

```
.gallerydisplay li a img.mainimage {
    position:absolute;
    left:0;
    display:none;
    border:0;
    top:55px;
    z-index:5;
}

.gallerydisplay li a:active img.mainimage,
.gallerydisplay li a:hover img.mainimage,
.gallerydisplay li a:focus img.mainimage {
    display:block;
}

.gallerydisplay li img.thumbnail{
    height:27px;
    width:27px;
    margin-right:4px;
    border:1px solid #666;
}

.gallerydisplay li a:active img.thumbnail{
    border:#eee solid 1px;
}

#defaultimage img{
    position:absolute;
    top:55px;
    left:0;
    z-index:-1;
}

</style>
</head>

<body>

<div id="gallerywrapper">
    <div id="defaultimage">
        
```

```
</div>
<ul class="gallerydisplay">
  <li>
    <a href="#">
      
      
    </a>
  </li>

  <li>
    <a href="#">
      
      
    </a>
  </li>

  <li>
    <a href="#">
      
      
    </a>
  </li>
</ul>
</div>

</body>
</html>
```

Ushtrime

- Shkruani CV tuaj duke përdorur elementët header, list, paragraph si edhe elementë të tjerë. Aplikoni në dokument Inline CSS dhe Embedded CSS.
- Krijoni një web form me të paktën 4 textbox dhe labels, 1 bashkësi me 4 option button, një bashkësi me 4 check box, 1 combo box, 1 submit buton dhe një buton reset. Përdorni elementët paragraph dhe header. Aplikoni në dokument Inline CSS dhe External CSS.

5. JavaScript

HTML është një gjuhë shumë e pasur dhe për me tepër kur bashkohet me CSS, mund të ndërtohet çfardo dizajn faqe që mund të dëshirohet. Por HTML duke qenë një gjuhë markup, mund të përdoret vetëm për të përcaktuar pamjen e faqes dhe është e mjaftueshme në rastet kur nuk nevojiten animime ose ruajtje të dhënash online. Për ti shtuar funksionalitet të tilla duhet që të kalojmë në programimin në Web. JavaScript është një gjuhë programimi e interpretuar (script language).

JavaScript dhe Java janë dy gjuhë të ndryshme dhe me filozofi të ndryshme. E vetmja gjë e përbashkët është sintaksa duke qenë se Netscape ka bazuar ndërtimin e gjuhës JavaScript në sintaksën e Java.

5.1 Programi i parë

Sic ndodh shpesh kur shpjegohen gjuhë programimi do e fillojmë shpjegimin duke krijuar një program të thjeshtë JavaScript i cili paraqet mesazhin “Përshëndetje”.

Duke qenë se JavaScript është një gjuhë e lidhur me paget web, kodi do të jetë në një file HTML dhe do të paraqitet në një browser.

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="Javascript">
  function Greeting()
  {
    alert("Hello world");
  }
</SCRIPT>
</HEAD>
<BODY>
<FORM>
<INPUT TYPE="button" NAME="Button"
VALUE="Press" onClick="Greeting()">
</FORM>
</BODY>
</HTML>
```

Ky program JavaScript vizaton një buton në ekran, kur klikojmë në të shfaqet një dritare me mesazhin tonë.

Kodi JavaScript vendoset brenda *tag*-eve <SCRIPT>. Këto *tag*-e mund të vendosen në çdo pjesë të dokumentit. Browserat të cilët nuk suportojnë kod JavaScript thjeshtë do ta injorojnë përmbajtjen e këtyre *tag*-eve.

Zakonisht *taget* <SCRIPT> vendosen në kokën e faqes në mënyrë që kodi HTML të jetë sa më i lexueshëm.

5.2 Elementët bazë të JavaScript

Fjalit në JavaScript mbarojn me ; (njësoj si në C dhe Java) dhe mund të grupohen në blloqe duke përdorur kllapat gjarpërushe{ }. Simbolet sic mund të jenë variablat ose emrat e funksioneve janë case-sensitive.

Komentet

Komentet paraqiten në dy mënyra:

// Koment vetëm në një rresht.

/*

Koment i cili shtrihet në disa rreshta.

*/

Karakteret speciale

JavaScript përdor disa sekuenca karakteresh në mënyrë që të mund të vendosim karaktere speciale në konstantet që janë tekst.

Karakteret speciale më të përdoruara janë:

\n -rresht i ri

\t Tab

\' thonjës teke

\" thonjëza mbyllëse

\\ Backslash

\xxx numrat ASCII të karaktereve hexadecimalë

5.3. Tipet e të dhënave dhe variablat

Në JavaScript, tipet e të dhënave caktohen në mënyrë dinamike në momentin që ne i japim vlerë variablit. Tipi mund të jetë:

- character string
- integers
- real

- Boolean
- vector
- matrice
- reference
- objekt

Variablat

Në JavaScript, emrat e variablave fillojnë me një gërmë alfabetike ose me karakterin

'_', dhe mund të formohen nga gërma ose dhe nga kombinime me karakterin '_'. Variablat janë global dhe nuk ka nevojë për deklarim eksplisit të tyre. Në rastet kur nevojitet një variabël lokal ai deklarohet duke përdorur fjalën var dhe vendoset në trupin e një funksioni.

Referencat

JavaScript eliminimin shënjesit në kujtesë nga gjuha por përdor referencat. Referencat funksionojnë në mënyrë të ngjashme me shënjesit e memories, vetëm se ato nuk merren me menaxhimin e kujtesës nga ana e programuesit gjë e cila bënte që shënjesit të ishin të prirura ndaj gabimeve (error).

JavaScript lejon referenca të objektet dhe të funksionet. Kjo aftësi për të referuar funksionet do të shërbejë shumë kur të përdorim funksionet për të eliminuar dallimet ndërmjet browser-ave.

```
function onlyExplorer()
{
  ... }
function onlyMozilla()
{
  ...
}
function all()
{
  var function;
  if(browserMozilla)
  function=onlyMozilla;
  else
  function=onlyExplorer;
  function();
}
```

}

Ushtrime

1. Krijë një faqe HTML dhe JavaScript që mbledh dy numra.
2. Krijë një faqe HTML dhe JavaScript që gjen shumën e shifrave të një numri.
3. Krijë një faqe HTML dhe JavaScript që gjen të anasjelltin e një numri.
4. Krijë një faqe HTML dhe JavaScript që gjen nëse një numër është palindrome ose jo.
5. Krijë një faqe HTML dhe JavaScript që gjen nëse një numër është çift (even) ose tek (odd).

5.4 Strukturat e kontrollit

Si ë gjitha gjuhët eprogramimit, JavaScript ka disa struktura kontrolli.

If, Switch

JavaScript ofron dy strukturat më të njohura të kontrollit If dhe Switch

Struktura e kushtit If është si më poshtë. Kushti If mund të jetë i pasur me fjalën else mbas tij.

```
if (condition)
<code>
else
<code>
```

Shembull i kushtit IF :

Struktura e kushtit Switch:

```
switch(value)
{
case valuetest1:
<code>
break;
case valuetest2:
<code>
break;
...
default:
<code>
}
```

```
<script type="text/javascript">
<!--
var ngjyra = "Blu";

if (ngjyra == "Blu") {
    document.write("Si
ngjyra e detit!");
}
//-->
</script>
```

```
<script type="text/javascript">
<!--
var ngjyra = "E kuqe";

switch (ngjyra)
{
case "Blue":
    document.write("Si ngjyra e detit!");
    break
case "Red":
    document.write("E kuqe!");
    break
default:
    document.write("Ngjyra qe deshironi...");
}
//-->
</script>
```

Ciklet

Janë tre struktura ciklike, *while*, *do while*, dhe *for*.

```
while (condition)
```

```
<code>
```

```
do
```

```
{
```

```
<code>
```

```
}
```

```
while(condition);
```

```
for(start; condition; increase)
```

```
<code>
```

Shembull i ciklit **For**,

```
<script type="text/javascript">
<!--
var myBankBalance = 0;

for (myBankBalance = 0; myBankBalance <= 10;
myBankBalance++)
{
document.write("My bank balance is $" +
myBankBalance + "
");
}
//-->
</script>
```

Ushtrime

Kërkojini përdoruesit të shtypi nga tastiera një numër nga 20 në 100. Afishojini përdoruesit mesazhin “Shumë i vogël”, “Numri i duhur ” ose “Shumë i madh” në varesi të numrit që ai fut.

Cfare output-i do të nxjerrë kodi i mëposhtëm?

```
<SCRIPT LANGUAGE="JavaScript">
var first = "Hello", second = "Goodbye";
var number = 32;
if (first != "Hello" && second == "Goodbye")
    {   number += 12;
        number *= 2;
    }
else
    {   number -= 10;
    }
if (first == "Hello" || second != "Goodbye")
    {   number = 13;
    }
else
    {   number *= 5;
    }
alert("The value of number is " + number + "<P>");
</SCRIPT>
```

3. Krijoni një faqe web që konverton vlerën e futur në lekë në euro. Faqja duhet të përbëhet nga një textbox dhe një buton, i cili kur klikohet konverton vlerën e futur në textbox dhe e shfaq në faqe ose në një mesazh.

5.5 Eventet

Një nga aspektet më të rëndësishme të JavaScript është ndërveprimi me browser-in. Ngjarjet bëjnë të mundur që të shkruajm kod në JavaScript i cilin ndërvepron në një situat të caktuar. Shembuj ngjarjesh mund të jenë: Klikimi me maus, ngarkimi i faqes, ndryshimi i ndonjë fushe në form etj.

Për të ekzekutuar rreshta kodi në momentin e ndodhjes në ngjarjes, JavaScript ofron ato që quhen "*event handler*". Të gjithë mbajtësit e ngjarjet fillojnë me parashtesën *on*.

Me poshtë janë listuar disa prej tyre:

onLoad -Hapja e faqes përfundon

onUnload-Mbyllet faqja.

onMouseOver-Mausi ndodhet mbi element.

onMouseOut -Mausi sapo zhvendoset nga elementi.

onSubmit -Kur forma dërgohet.

onClick -Nje element është klikuar.

onBlur -Kursori nuk duket më.

onChange -Përmbajtja ka ndryshuar. Kjo ngjarje ndodh te elementët:

<INPUT>, <TEXTAREA>

onFocus-Kursori shfaqet. Kjo ngjarje ndodh te elementët: <INPUT>,

<TEXTAREA>

onSelect -Teksti është përzgjedhur. Kjo ngjarje ndodh te elementët:

<INPUT TYPE="text">, <TEXTAREA>

Ekzistojnë dy mënyra për ti caktuar funksionit një ngjarje:

```
<HTML>
```

```
<HEAD>
```

```
<SCRIPT LANGUAGE="Javascript">
```

```
function Alarm() {  
  alert("Pershendetje !");  
}
```

```
</SCRIPT>
```

```
</HEAD>
```

```
<BODY onLoad="Greeting()">
```

```
...
```

```
</BODY>
```

```
</HTML>
<HTML>
<HEAD>
<SCRIPT LANGUAGE="Javascript">
function Pershendetje() {
alert("Pershendetje!");
}
window.onload = Pershendetje;
</SCRIPT>
</HEAD>
<BODY>
...
</BODY>
</HTML>
```

Shembull

Ky është kodi për një faqe të thjeshtë web-i e cila paraqet vetëm një foto. Pjesa interesante është se me ane të kodit Javascript ne arrijmë të marrim gjerësin dhe lartësin e ekranit të kompjuterit të vizitorit dhe në varësi të tyre gjenerohet kod me anë të *document.write*. Krijohen dy tag-e të ndryshme ``. Nëse rezolucioni i ekranit është 1024 deri ne 768, ngarkohet imazhi me madhësi normale `normal.jpg`. Nëse rezolucioni është më i madh, ngarkohet imazhi më i madh `bigger.jpg`

```
<HTML>
<body>
<script language="javascript">
var w = screen.width;
var h = screen.height;
if ((w=1024) and (h=768))
{
document.write("<img src=normal-pic.jpg>");
}
if ((w > 1280) and (h > 1024))
{
document.write("<img src=bigger-pic.jpg>");
}
</script>
</body>
</HTML>
```

5.6 Objektet

Në JavaScript, një objekt është një struktur e dhënë e cila përmban edhe variablat edhe funksionet për të kapur objektet.

Programimi i orientuar në objekte që përdoret në Javascript është shumë më i thjeshtë se ai në Java ose C++. JavaScript nuk bën dallim ndërmjet objekteve dhe instancave. Vecorit dhe metodat e objekteve do të kapeshin si më poshtë:

object.property

value=object.method(parameter1, parameter2, ...)

Përcaktimi i objekteve ne JavaScript

Përpara se të përcaktojmë një objekt në JavaScript, duhet të krijojmë një funksion të veçantë qëllimi i të cilit është të krijoj një objekt.

Është e nevojshme që emri i funksionit, i quajtur ndryshe konstruktor, dhe emri i objektit të jetë i njëjtë.

```
function MyObject(attr1, attr2)
{
  this.attr1=attr1;
  this.attr2=attr2;
}
```

Tani ne mund të krijojmë objekte të tipit *MyObject*.

```
object=new MyObject(....)
object.attr1=a;
```

Nëse duam që ti shtojmë metoda objekteve duhet më parë ti përcaktojmë ato si funksione normale.

```
function Method1(attr1, attr2)
{
  Pjesa e kodit
}
```

Për ti caktuar këtë metodë një metode objekti duhet të shkruajmë:

```
object.method1=Method1;
```


Objektet e paracaktuara

Në implementimet e JavaScript janë të përfshira disa objekte të parapërcaktuara:

- **Arrays** Vektorët.
- **Date** Për të ruajtur dhe manipuluar me datat.
- **Matëherë** Metoda dhe konstantet matematike.
- **Number** Konstante.
- **String** Veprime me stringe.
- **RegExp** Veprime me shprehjet e rregullta.
- **Boolean** Vlera Boolean-e.
- **Function** Funksionet.

Vektorët

Një vektor është një variabël i vecantë i cili mban më shumë se një vlerë në një kohë.

Vektorët mund të krijohen në tre menyra. Në shembullin e mëposhtëm është deklaruar një objekt i tipit vektor i quajtur Ditet dhe krijimi i tij është paraqitur me të treja mënyrat.

1:

```
var                Ditet=new                Array();  
Ditet[0]="Hene";  
Ditet[1]="Marte";  
Ditet[2]="MERKURE";
```

2:

```
var Ditet=new Array("Hene","Marte","MERKURE");
```

3:

```
var Ditet=["Hene","Marte","MERKURE"];
```

Aksesimi i elementëve të një vektori

Për të kapur një element të një vektori duhet ti referohemi me anë të emrit të vektorit dhe numrit të indeksit. Indeksi fillon nga 0.

Nga kodi i mëposhtëm

```
document.write(Ditet[0]);
```

do të shfaqet output-i:

Hene

Modifikimi i vlerave të një vektori

Për të modifikuar vlerat e një vektori ekzistues thjeshtë i caktojm një vlerë të re elementit të vektorit me një indeks të caktuar:

```
Ditet[0]="Enjte";
```

```
document.write(Ditet[0]);
```

Tani rezultati është:

Enjte

Objekti Date

Objekti Date përdoret për të punuar me datën dhe orën. Objektet Date krijohen me anë të `new Date()`.

Një datë mund të inicializohet në menyra të ndryshme:

```
var d = new Date();
```

```
var d = new Date(milliseconds);
```

```
var d = new Date(dateString);
```

```
var d = new Date(year, month, day, hours, minutes, seconds,  
milliseconds);
```

Disa metoda të objektit Date

Metoda	Pershkrimi
<u>getDate()</u>	Kthen ditën e muajit (nga 1-31)
<u>getDay()</u>	Kthen ditën e javës (nga 0-6)
<u>getFullYear()</u>	Kthen vitin
<u>getHours()</u>	Kthen orën (nga 0-23)
<u>getMilliseconds()</u>	Kthen Milisekondat (nga 0-999)
<u>getMinutes()</u>	Kthen minutat (nga 0-59)
<u>getMonth()</u>	Kthen muajt (nga 0-11)
<u>getSeconds()</u>	Kthen sekondat (nga 0-59)

<u>getTime()</u>	Kthen numrin e milisekondave qe nga 1 Janar, 1970
------------------	---

Objekti Math

Më anë të objektit Math mund të kryhen veprime matematikore.

Math nuk është konstruktor dhe cdo atribut apo metod e tij mund të përdoren duke e konsideruar Math si një object.

Sintaksa:

```
var      x      =      Math.PI;      //      Kthen      PI
var y = Math.sqrt(16); // Kthen rrenjen katrore te 16
```

Metodat e objektit Math

Metoda	Pershkrimi
<u>abs(x)</u>	Kthen vleren absolute te x
<u>acos(x)</u>	Kthen arkosinuksin e x, ne radian
<u>asin(x)</u>	Kthen arksinuksin e x, ne radian
<u>atan2(y,x)</u>	Kthen arktagenting e argumenteve ne thnonjeza
<u>max(x,y,z,...,n)</u>	Kthen numrin me vlere me te madhe
<u>min(x,y,z,...,n)</u>	Kthen numrin me vlere me te vogel
<u>pow(x,y)</u>	Kthen vleren e x ne fuqi y
<u>random()</u>	Kthen nje numer random nga 0 ne 1
<u>round(x)</u>	Rrumbullakos x me numrin e plote me te afert
<u>sin(x)</u>	Kthen sinusin ne x
<u>sqrt(x)</u>	Kthen katrorin e rrenjes se x

Ushtrime

1. Shkruani një funksion `findLongestWord()` i cili merr një vektor me fjalë dhe kthen gjatësin e fjalës më të gjatë.
2. Ruani në një vektor ditët e javës dhe në varësi të numrit të shtypur nga përdoruesi ju i afishoni ditën në fjalë. P.Sh nëse përdoruesi shtyp 1 ju do të afishoni fjalën “E hënë”.
3. Ndërtoni një makinë llogaritëse e cila kryen veprimet e mbledhjes dhe zbritjes.

5.7 Funksionet

Një funksion përmban kod i cili do të ekzekutohet nga një ngjarje, ose nga një thirrje e këtij funksioni. Sintaksa e deklarimit dhe implementimit të një funksioni është si më poshtë:

```
function name(argument1, argument2, ..., argument n)
{
  kodi
}
```

Më poshtë do të shohin se si ta thërrasim një funksion, si ti kalojmë argumentet dhe në fund se si të kthejm të dhëna me anë të return.

Ky është një funksion:

```
function shembull(a,b) {
  number += a;
  alert('Ju keni zgjedhur: ' + b);
}
```

Funksioni thirret si më poshtë:

```
shembull(1, 'dhurate')
```

Argumentat

Një funksioni mund ti kalojm argument. Ato mund të jenë variabla, numra ose stringje me të cilat funksioni vepron. Outputi i një funksioni varet nga argumentet që i japim.

Në shembull ne i kaluam dy argumente, numrin 1 dhe stringun 'dhuratë':

```
shembull(1,'house');
```

Mund të perdorim aq argumenta sa ne duam.

```
function shembull(a,b,c,data,data2) {
  number += a;
  alert('Ju keni zgjedhur: ' + b);
  (veprime te tjera me argumentet c, data dhe data2)
}
```

Nëse harroni të vendosni ndonjë argument, funksioni nuk ekzekutohet. Supozojm se i kemi kaluar vlerat e mëposhtme:

```
shembull(1,'trendafila',16,'orkide')
```

Kjo thirrje do të shfaqë error pasi ne nuk kemi përcaktuar asnjë vlerë për argumentin data2.

Ne mund të shkruajmë edhe funksione të cilët nuk marrin fare argumente.

```
function shembull() {  
    number += 1;  
    alert('Ju keni zgjedhur: trendafila');  
}
```

Ky funksion kryen të njëjtin veprim dhe nxjerr të njëjtin output sa herë që ne e thërrasim. Nëse tentojmë ti japim vlera argumentash do ti injorojë.

Kthimi i vlerave

Një veprim tjetër që funksionet kryejnë është kthimi i vlerave. Supozojmë se kemi funksionin e mëposhtëm:

```
function calculate(a,b,c) {  
    d = (a+b) * c;  
    return d;  
}
```

Ky llogarit një numër në varësi të numrave që ju i kaloni si argument. Kur funksioni përfundon ekzekutimin kthen rezultatin e llogaritur.

Në praktik do të kishim:

```
var x = calculate(4,5,9);
```

```
var y = calculate((x/3),3,5);
```

dhe vlera e x do të bëhet 81 ndërsa e y 150.

Një funksion mund të kthejë edhe stringje apo vlera Boolean-e.

5.8 Vlersimi i Formave

JavaScript mund të përdoret për të vërtetuar të dhënat në format HTML para se të dërgohen për një server. Të dhënat që kontrollohen për arsye të ndryshme si: nëse fushat janë lënë bosh, nëse përdoruesi ka shkruar një adresë emaili-i të vlefshme etj.

Fushat e Kërkuara *

Funksioni i mëposhtëm kontrollon nëse ndonjëra nga fushat e forms është boshe, e paplotësuar nga përdoruesi. Nëse fusha është boshe atëherë do të shfaqet një mesazh dhe funksioni do të kthejë false duke mos lejuar kështu bërjen submit të forms.

```
function validateForm()
{
var x=document.forms["myForm"]["fname"].value;
if (x==null || x=="")
{
    alert("Duhet plotesuar fusha e emrit");
    return false;
}
}
```

Ky funksion më pasë thërret kur një form bëhet submit si në shembullin më poshtë:

```
<form name="myForm" action="demo_form.asp"
onsubmit="return validateForm()" method="post">
First name: <input type="text" name="fname">
<input type="submit" value="Submit">
</form>
```

Vlersimi i email-it

Funksioni i mëposhtëm nëse përmbajtja e fushës ka sintaksen e përgjithshme të një emaili ku të dhënat duhet të kenë një shenjë @ dhe të paktën një pikë (.). Shenja @ nuk duhet të jetë karakteri i parë i një adresë email-i dhe pika e fundit duhet të jetë pas shenjës @ dhe minimalisht 2 karaktere para fundit.

```
function validateForm()
{
var x=document.forms["myForm"]["email"].value;
```

```
var atpos=x.indexOf("@");
var dotpos=x.lastIndexOf(".");
if (atpos<1 || dotpos<atpos+2 ||
dotpos+2>=x.length)
{
    alert("Adresa nuk eshte e vlefshme");
    return false;
}
}
```

Ushtrime

1. Përcaktoni një funksion `max()` i cili merr vlerën e dy numrave dhe kthen më të madhin. Përdorni strukturën if-then-else.
2. Krijë një faqe HTML dhe JavaScript që gjen nëse një numër është i thjeshtë (prime number).
3. Krijë një faqe HTML dhe JavaScript që gjen factorial.
4. Përcaktoni një funksion `sum()` dhe një funksion `multiply()` të cilët mbledhin dhe shumëzojnë respektivisht të gjithë numrat në një vektor. Për funksionin `sum([1,2,3,4])` dhe duhet të kthejë 10, ndërsa `multiply([1,2,3,4])` duhet të kthejë 24.
5. Përcaktoni një funksion `reverse()` i cili llogarit të anasjelltin e një stringu. Për shembull fjala "Mire" duhet të kthehet në "eriM".

5.9 Aplikime JavaScript

Ndërtoni një faqe e cila të llogarisë shumën e dy numrave të shtypur nga përdoruesi.

Zgjidhje:

Përdoruesi shtyp dy numrat përkatesisht në textbox-in 1 dhe 2 dhe vlera shfaqet në textbox-in përbri fjalës Shuma. Klikimi në textbox kapet me anë të ngjarjes OnClick dhe në këtë ngjarje thirret funksioni *Shuma()* të cilin e kemi implementuar në kokën e faqes.

Numri1	Numri2	Veprimi	Vlera
<input type="text" value="0"/>	<input type="text" value="0"/>	Shuma	<input type="text"/>

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Shuma</title>
<script language="JavaScript">
function Shuma()
{
document.frm1.s1.value=parseInt(document.frm1.nr1.value)+parseIn
t(document.frm1.nr2.value);
}
</script>
<meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>
<table width="400" border="0">
<tr>
<td>Numri1</td>
<td>Numri2</td>
<td>Veprimi</td>
<td>Vlera</td>
```

```

</tr>
<form name="frm1">
<tr>
  <td><input      type="text"      name="nr1"      value="0"
onChange="javascript:Shuma();"></td>
  <td><input      type="text"      name="nr2"      value="0"
onChange="javascript:Shuma();"></td>
  <td>Shuma</td>
  <td><input type="text" name="s1"></td>
</tr>
</form>
</table>
</body>
</html>

```

Gjeni Shumën, diferencën, prodhimin dhe herësin e dy numrave çfarëdo të shtypur nga përdoruesi në faqen tuaj të *web-it*.

Zgjidhje:

Numri1	Numri2	Veprimi	Vlera
<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="button" value="shuma"/> <input type="button" value="diferenca"/> <input type="button" value="prodhimi"/> <input type="button" value="heresi"/>	<input type="text"/>

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Shuma</title>
<script language="JavaScript">
function Shuma()
{
document.frm1.s1.value=parseInt(document.frm1.nr1.value)+parseIn
t(document.frm1.nr2.value);
}
function Diferenca()
{

```

```
document.frm1.s1.value=parseInt(document.frm1.nr1.value)-
parseInt(document.frm1.nr2.value);
}
function Prodhimi()
{
document.frm1.s1.value=parseInt(document.frm1.nr1.value)*parseInt
(document.frm1.nr2.value);
}
function Heresi()
{
document.frm1.s1.value=parseInt(document.frm1.nr1.value)/parseIn
t(document.frm1.nr2.value);
}
</script>
<meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>
<body>
<table width="400" border="0">
  <tr>
    <td>Numri1</td>
    <td>Numri2</td>
    <td>Veprimi</td>
    <td>Vlera</td>
  </tr>
  <form name="frm1">
    <tr>
      <td><input type="text" name="nr1" value="0" ></td>
      <td><input type="text" name="nr2" value="0" ></td>
      <td><input type="button" value="shuma"
onClick="javascript:Shuma();"><br>
        <input type="button" value="diferenca"
onClick="javascript:Diferenca();"><br>
        <input type="button" value="prodhimi"
onClick="javascript:Prodhimi();"><br>
        <input type="button" value="heresi"
onClick="javascript:Heresi();"><br>
      </td>
      <td><input type="text" name="s1"></td>
```

```
</tr>
</form>
</table>
</body>
</html>
```

Ndërtoni një faqe në të cilën përdoruesi të mund të fusë të dhënat e tij personale dhe të vlerësojë saktësinë e tyre.

Username	<input type="text"/>
Password	<input type="password"/>
Confirm Password	<input type="password"/>
Name	<input type="text"/>
Lastname	<input type="text"/>

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
<script language="JavaScript">
function check()
{
check_user();
//check_name();
}
function check_user()
{
if (document.user.username.value == "")
{
alert('Kujdes funti username');
document.user.username.value='[enter username]';
document.user.username.focus();
```

```
}  
}  
</script>  
</head>  
<body>  
<form name="user">  
<table width="400" border="0">  
  <tr>  
    <td>Username</td>  
    <td><input type="text" name="username" value="[enter  
username]" ></td>  
  </tr>  
  <tr>  
    <td>Password</td>  
    <td><input type="password" name="pass1"></td>  
  </tr>  
  <tr>  
    <td>Confirm Password</td>  
    <td><input type="password" name="pass2"></td>  
  </tr>  
  <tr>  
    <td>Name</td>  
    <td><input type="text" name="textfield4"></td>  
  </tr>  
  <tr>  
    <td>Lastname</td>  
    <td><input type="text" name="textfield5"></td>  
  </tr>  
  <tr>  
    <td>&nbsp;</td>  
    <td>&nbsp;</td>  
  </tr>  
  <tr>  
    <td>&nbsp;</td>  
    <td><input type="submit" name="button" value="Kontrollo"  
onClick="check();"></td>  
  </tr>  
  <tr>  
    <td>&nbsp;</td>
```

```
<td>&nbsp;</td>  
</tr>  
</table>  
</form>  
</body>  
</html>
```

5.10 jQuery

jQuery është një librari JavaScript e shpejtë dhe e saktë e cila përdoret për të thjeshtuar punën me dokumentet HTML, punën me eventet, animimin, si dhe ndërveprimet me Ajax. Për të përdorur këtë librari mund ta shkarkoni atë nga faqja <http://jquery.com/> ose të shfrytëzoni kopje të saj të hostuara nga kompani të mëdha si p.sh. të ruajtur te Google nën linkun <http://ajax.googleapis.com/ajax/libs/jquery/1.7.1/jquery.min.js>.

Për të thirrur librarin e ruajtur në një host të jashtëm në programin tonë tagu <script> do të dukej si më poshtë:

```
<script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.
4.0/jquery.min.js?ver=1.4.0">
</script>
```

jQuery ekzekutohet pasi DOM është gati dhe është e ndare nga përmbajtja (HTML) dhe paraqitja (CSS). Kodi jQuery pozicionohet si më poshtë:

```
$(document).ready(function() {
    // i gjithë kodi jQuery vendoset ketu
});
```

Perzgjedhja e elementeve ne jQuery

Libraria jQuery bën të mundur selektimin e elementëve të XHTML duke i futur ato në kllapa dhe thonjëza dyshe ose teke të tilla \$(""). Më poshtë jepen disa shembuj të kapjes së elementëve:

```
$("div"); //përzgjedh të gjithë elementët div të HTML
```

`$("#myElement");` // për zgjedh një element HTML me ID "myElement"

`$(".myClass");` // për zgjedh elemente HTML me klas "myClass"

`$("p#myElement");` // për zgjedh element paragraf HTML me ID "myElement"

`$(":animated");` // për zgjedh elementët e animuar

`$(":button");` // për zgjedh elementët button

`$(":radio");` // për zgjedh radio button-at

`$(":checkbox");` // për zgjedh checkbox-et

`$(":checked");` // për zgjedh të gjithë checkbox-et ose radio button-at të cilët janë të selektuar

`$(":header");` // për zgjedh elementët header(h1, h2, h3, etj.)

Shtimi, Fshirja, dhe Bashkëngjitja e Elementëve dhe Përmbajtjes

Ka mënyra të ndryshme për manipulimin e grupeve të elementëve me ane të jQuery.

Për të marrë një element në HTML shkruajmë:

```
var myElementHTML = $("#myElement").html();
```

// variabëli përmban të gjithë HTML-në (përfshirë tekstin) brenda #myElement

Nëse nuk kemi nevojë për të aksesuar të gjithë HTML-në, por duam vetëm elementin tekst:

```
var myElementHTML = $("#myElement").text();
```


Duke shfrytëzuar këtë sintaksën e dy shembujve të mësipërm ne mund të ndryshojm edhe përmbajtjen e tekstit të një elementi:

```
$("#myElement").html("<p>Kjo eshte permbajtja e re.</p>");
```

ose

```
$("#myElement").text("Kjo eshte permbajtja e re.");
```

Për ti bashkangjitur përmbajtje një elementi dukë lënë të pandryshuar përmbajtjen ekzistuese:

```
$("#myElement").append("<p> Kjo eshte permbajtja e re.</p>");
```

jQuery gjithashtu ofron funksionet `appendTo()`, `prepend()`, `prependTo()`, `before()`, `insertBefore()`, `after()`, dhe `insertAfter()`, të cilat funksionojn afërsisht si `append()`.

Eventet në jQuery

Eventet mbahen duke përdorur kodin e mëposhtëm:

```
$("#a").click(function() {  
    // pjese kodi sipas problemit  
    //kur klikohet nje link  
});
```

Pjesa e kodit brenda funksionit `function()` do të ekzekutohet vetëm pasi të klikohet mbi link. Evente të tjera të ngjashme janë:

`blur`, `focus`, `hover`, `keydown`, `load`, `mousemove`, `resize`, `scroll`, `submit`, `select`.

Animimi dhe efektet

Me jQuery mund të rrëshqisni(slide) elementët, ti animoni, të ndaloni animimin. Për të rrëshqitur elementët lart ose poshtë shkruajmë kodin e mëposhtëm:

```
$("#myElement").slideDown("fast", function() {  
  // veprimi qe duam te ndodhi mbas rreshqitjes  
  poshte  
})  
$("#myElement").slideUp("slow", function() {  
  // veprimi qe duam te ndodhi mbas rreshqitjes lart  
})  
$("#myElement").slideToggle(1000, function() {  
  // veprimi qe duam te ndodhi mbas rreshqitjes  
  lart/poshte  
})
```

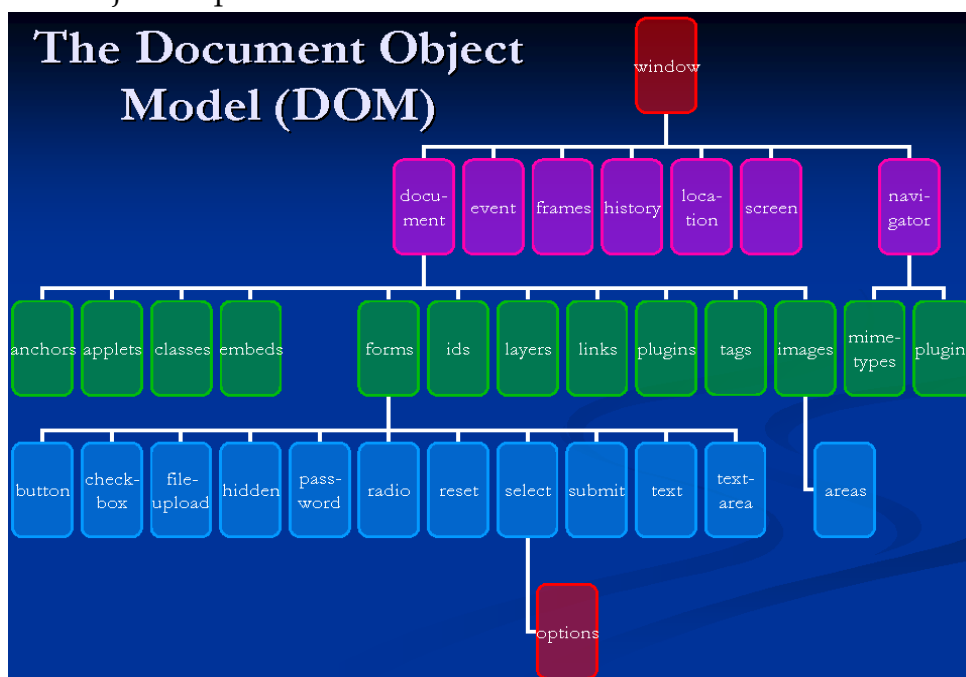
Animimi i një elementi bëhet duke komanduar CSS që të ndryshojë vlerat. jQuery do të vendosi stilet e reja në mënyrë graduale në varësi të shpejtësisë që është përcaktuar.

```
$("#myElement").animate(  
  {  
    opacity: .3,  
    width: "500px",  
    height: "700px"  
  }, 2000, function() {  
  }  
);
```

6. DOM

DOM është shkurtimi për Document Object Model. Kjo është një mënyrë për të parë HTML-në e cila bën mundur që zhvilluesit e web-it të krijojnë funksione dhe të manipulojnë me kodin në menyra të cilat do të ishin të pamundura pa këtë konceptim.

Ekzistojnë tre tipe bazë të DOM-it:



Pjesa kryesore e DOM, e cila përdoret për të përcaktuar dokumentin pavarësisht nga tipi XML DOM, i cili përdoret me dokumentet XML. HTML DOM, ndan një faqe HTML në nyje. Këto nyje janë të lidhura me njëra tjetrën në formën e një peme.

Ajo që e bën DOM-in kaq të rëndësishëm dhe të përdorshëm për zhvilluesit e HTML është se çdo pjesë e faqes bëhet e vecantë dhe e aksesueshme në raport me faqen. Duke pasur një mundësi copëzimi të tillë ne mund të veprojmë në pjesë të vecanta të faqes pa i prekur pjesët e tjera. Kjo ka një rëndësi të vecantë për programimin në Javascript duke qenë se kështu krijohet mundësia e kapjes së çdo elementi të HTML në mënyrë direkte. Duke ditur strukturën e saktë përmes DOM, ne mund të kalojmë në të gjithë elementët në të cilët duam të kryejmë veprime.

Tani më e kemi parë se një faqe HTML është e përbërë nga elementët e mëposhtëm:

```
<html>
<head>
<title>Faqe HTML </title>
</head>
<body>
<p id="sample">Kjo eshte nje faqe e thjeshte HTML.</p>
</body>
</html>
```

Me anë të DOM, çdo element, si p.sh tag-et <html></html>, është një nyje. Nyja <html> quhet nyja rrënjë dhe ka dy nyje bijë <head> dhe <body>. Në shembullin e mësipërm nyja <title> është fëmij i nyjës <head>.

HTML DOM është një gjuhë e pavarur nga platforma. DOM është e ndarë nga JavaScript teknikishtë dhe mund të kapet me anë të gjuhëve të skriptimit.

Lista e mëposhtëme paraqet listën e objekteve të HTML DOM të cilat përdoren për të marrë, nxjerrë informacione rreth ambientit të sistemit të përdoruesit. Informacionet mund të jenë si madhësia e ekranit, rezolucioni, historia e browserit, URL-ja, sistemi i shfrytëzimit, tipi i browserit etj.

Objektet janë:

Window

Document

Screen

Navigator

History

Location

Secili nga objektet ka atributet dhe metodat e tija. Ne do të shohim objektin Window i cili përfaqson dritaren e browserit. Lista e mëposhtëme jep disa nga metodat dhe atributet e objektit Window:

closed

defaultStatus

frames

history

length
location
name
opener
parent
personalbar
self
status
statusbar
scrollbars
top
toolbar
window
innerWidth
innerHeight
outerWidth
outerHeight
pageXOffset
pageYOffset
screenX
screenY
screenLeft
screenTop

window

Objekti window është në krye të hierarkisë së modelit DOM dhe përfaqëson të gjithë browserin.

Atributet:

location – Kthen URL-ne

Metodat:

alert() – Afishon një mesazh tekst

prompt() – Shfaq një dritare dialoguese e cila kërkon input nga përdoruesi.

Shembull:

Shembulli i shfaq përdoruesit dy butona të cilët kur shtypen shfaqin një mesazh dhe pyesin për input respektivisht s. Në fund të faqes gjithashtu shfaqet një tekst dhe url-ja e faqes së hapur.

```
<html> <head> <title>Objekti Window</title> </head> <body>
<input type="button" value="Click here for a message"
onclick="window.alert('Ky eshte mesazhi .');> <br /> <input
type="button" value="Kliko per te marre mesazhin"
onclick="window.prompt('Fusni te disa te dhena');> </script>
<script type="text/javascript"> document.write(" URL e kesaj faqe eshte " +
window.location); </script> </body> </html>
```

Objekti document

Objekti Document përfaqson një faqe të web dhe me anë të tij mund të kapim të gjithë elementët e kësaj faqe. Shohim metodat dhe propriet e tij.

Vetitë e objektit document

Këtu do të futen pothuajse gjithë atributet që ne mund të vendosim në elementin <body> të një faqeje XHTML. Disa properti kanë cilësinë vetëm të lexojnë ndërsa disa të tjera përpos leximit kanë dhe aftësinë e shkrimit. Për më shumë tabela që vijon:

Emri i proprietise	Qellimi	Read/write
alinkColor	Specifikon ngjyrat e linkut	Read/write
Bgcolor	Specifikon ngjyrën e sfondit	Read/write
Fgcolor	Specifikon ngjyrën e foreground të text	Read/write
lastModified	Data kur dokumenti ka qenë modifikuar për herë të fundit	Vetëm read
linkColor	Specifikon ngjyrën e linqeve	Read/write
Referrer	URL e faqes që përdoruesi do të kalojë nëqë klikon mbi një link.	Vetëm read
Title	Titullin e faqes që kapet tek elementi <title>	Vetëm read

vlinkColor	Ngjyrën e linqeve që janë klikuar një herë	Read/write
------------	--	------------

Për të aksesuar një prej propertive, ju duhet të përdorni shënimet e simbolit . (pikë); kështu psh nqs kërkon të kapni titullin e faqes ath ju duhet të shkruani:

`document.title`

nqs kërkon të kapni datën se kur është modifikuar për herë të fundit një dokument ath duhet të shkruani

document.lastModified

nqs serveri nuk suporton propertinë lastModified ath IE do të shfaq orën korrente ndërsa browsera të tjerë do të shfaqin 1 January 1970.

Metodat e objektit Document

Metodat kryejnë veprime dhe gjithmonë janë të ndjekur nga një çift kllapash. Brenda kllapave të disa metodave mund të kemi parametra ose argumenta të cilat do të ndikojnë në atë se çfarë do të kthej veprimi që do të kryhet.

Tabela më poshtë na tregon për dy metoda të cilat shkruajnë kontent të ri në faqen e web. Të dyja metodat do të marrin si argument një string (bashkësi me shkronja, numra, hapësira apo dhe shenjat e pikësimit etj) dhe stringu është ajo çfarë do të shkruhet në faqe.

Emri i metodës	Qëllimi
Write(string)	Ju lejon që të shtoni tekst ose element në një dokument
Writeln (string)	I njëjti efekt me write(), porse shton një rresht të ri pas shkrimit të stringut e njëjtë me atë që kemi shtypur tastin enter pasi kemi shkruar stringun.

Pjesa II - Programimi Server Side

7. Prezantim me PHP

PHP është një gjuhë mjaft popullore si gjuhë skriptimi në server. Gjuha PHP është ndërtuar bazuar në Perl, prandaj është e ngjashme me Perl, C, Java

Kodi PHP funksionon si me webserver ashtu edhe nëpërmjet me prompt (Command Line Interface)

Më poshtë listohen disa nga vetitë kryesore të PHP :

- Variablat kanë tipe dinamike dhe nuk ka nevojë të deklarohen
- Është e pajisur me array shumë dimensionale
- Mundëson programimin me objekte
- Ofron funksione për të punuar lehtësisht me bazat e të dhënave
- Ofron funksione për kontrollin e sesioneve dhe cookiet
- Në dallim nga JavaScript PHP ekzekutohet në server !

Ekzekutimi i një faqe PHP në web server do të prodhojë një *output*, i cili do ti dërgohet browser-it. Prandaj në browser nuk do të shikojmë asnjëherë kod PHP për vetëm rezultatin e ekzekutimit të një faqe PHP , që do të gjeneronte një tekst të përbërë nga HTML dhe Javascript.

Kur ne vendosim adresën e një faqe PHP në kutinë e adresës të lundruesit, ky i fundit bën një kërkesë në server për faqen PHP, adresën e të cilës ne e kemi në kutinë e adresës. Serveri kthen një përmbajtje HTML që gjenerohet duke u bazuar në hapat e mëposhtëm:

Serveri e lexon skedarin PHP rresht për rresht duke nisur nga fillimi

Nëse gjen kod html në faqe thjesht e kopjon këtë në output

Nëse gjen kod php (i cili vendoset midis `<?php` dhe `?>`) e ekzekuton atë dhe vendos në output rezultatin e ekzekutimit të kodit php

Hapat nga kërkesa e një faqe php deri tek afishimi në browser i saj janë:

- 1- Përdoruesi bën një kërkesë për skedarin PHP, duke vendosur adresën e tij në browser apo duke klikuar mbi një link që ka si destinacion skedarin php
- 2- Serveri gjen skedarin PHP dhe ia kalon atë interpretuesit të PHP i cili
- 3- Prodhon një skedar që ka vetëm kod html dhe javascript (nëse ka)

- 4- Web Serveri transmeton outputin e ekzekutimit të faqes PHP nëpërmjet një mesazhi HTTP Response
- 5- Kompjuteri merr përmbajtjen e dërguar nga web serveri dhe ia kalon atë browserit
- 6- Browseri interpreton kodin HTML duke prodhuar një ndërfaqe grafike për përdoruesin

Me poshte paraqitet ne te majte faqja php dhe ne te djathte rezultati i ekzekutimit te saj qe transmetohet ne browser . Vini re qe ne kodin qe i shkon browserit nuk ekziston kod PHP.

FAQJA PHP ne server

```
<html>
<head>
<title>shembull 2</title>
<?php
$name="Endri";
?>
</head>
<body>
<b>
<?php
echo "Pershendetje ".$name;
?>
</b>
</body>
</html>
```

Pas ekzekutimit, kodi që shkon në browser

```
<html>
<head>
<title>shembull 2</title>
</head>
<body>
<b>
Pershendetje Endri
</b>
</body>
</html>
```

7.1 Sintaksa, Variablat dhe Operatorët

Disa rregulla sintaksore të gjuhës PHP

Variablat në PHP fillojnë me shenjën e dollarit \$.

Emrat janë case sensitive.

Emrat ndjekin të njëjtat rregulla si javascript, emri duhet të fillojë me një shkrojnë ose me underscore (_).

Shembujt e mëposhtëm ilustrojnë rregullat që duhet të zbatojmë kur vendosim për emrat e variablave:

```
<?php
$var = "Bob";
$Var = "Joe";
echo "$var, $Var";    // afishon "Bob, Joe"
$4site = 'not yet';   // gabim; fillon me nje numer
$_4site = 'not yet';  // ne rregull; fillon me underscore
$täyte = 'mansikka';  // ne rregull; 'ä' është ASCII 228.
?>
```

7.2 Tipet e të dhënave në PHP

PHP ka disa tipe të dhënash, ndër të cilët janë tipet e mëposhtëm;

Boolean,

integer (zakonisht 32 bit)

numrat me presje (floating point numbers)

stringje ose vargjet e karaktereve

Tabelat ose arrays , të cilët shumë fleksibel dhe mund të përmbajnë të dhëna të tipeve të ndryshme

Klasat

Burimet (Resources)

NULL (variablat pa vlere)

Nëpërmjet funksionit `isset($v)` kontrollohet nëse një variabël ekziston apo jo.

Tipi i të dhënave Boolean ka dy vlera të mundshme : TRUE dhe FALSE

Me vlerën FALSE janë ekuivalente vlerat e mëposhtme:

Vlera integer 0

Vlera float 0.0
Stringu bosh ""
Vlera NULL
Një array me zero elementë
Të gjitha vlerat e tjera konsiderohen si TRUE.

7.3 Operatorët

Operatorët aritmetikë

-\$a Mohimi, e kundërta e \$a.
\$a + \$b Mbledhja, Shuma e \$a dhe \$b.
\$a - \$b Zbritja, Diferenca \$a - \$b.
\$a * \$b Prodhimi, Prodhimi i \$a dhe \$b.
\$a / \$b Pjesëtimi, \$a pjesëtim \$b.
\$a % \$b Mbetja, Mbetja e pjesëtimit të \$a me \$b.
\$a++ Operatori i pas inkrementimit
++\$a Operatori i pre inkrementimit
\$a-- Operatori i pas dekrementimit
--\$a Operatori i pre dekrementimit

Operatoret e krahasimit

\$a == \$b krahasimi për vler të njëjtë, TRUE nëse \$a është e barabartë me \$b, FALSE në të kundërt
\$a != \$b Jo të barabartë, TRUE nëse \$a është e ndryshme nga \$b, FALSE në të kundërt
\$a <> \$b Jo të barabartë, TRUE nëse \$a është e ndryshme nga \$b, FALSE në të kundërt
\$a < \$b Më e vogël se, TRUE nëse \$a është më e vogël se \$b.
\$a > \$b Më e madhe se, TRUE nëse \$a është më e madhe se \$b.
\$a <= \$b Më e vogël se ose e barabartë, TRUE nëse \$a është më e vogël ose e barabartë me \$b
\$a >= \$b Më e madhe se ose e barabartë, TRUE nëse \$a është më e madhe se \$b ose e barabartë me \$b

Operatoret llogjike

\$a and \$b, (AND) TRUE if both \$a and \$b are TRUE.
\$a or \$b, (OR) TRUE if either \$a or \$b is TRUE.
\$a xor \$b, (XOR) TRUE if either \$a or \$b is TRUE, but not both.

! \$a, (Not TRUE) if \$a is not TRUE.
\$a && \$b, (AND) TRUE if both \$a and \$b are TRUE.
\$a || \$b, (OR) TRUE if either \$a or \$b is TRUE.

Shembulli i mëposhtëm ilustron përdorimin e operatorëve llogjikë:

```
<?php
$a=0;
$b=1;
if($a&&$b) echo 'a &&b is true';
else echo 'a &&b is false';
// afishon a &&b is false

if($a and $b) echo 'a and b is true';
else echo 'a &&b is false';
// afishon 'a &&b is false

if($a || $b) echo 'a || b is true';
else echo 'a || b is false';
// afishon a || b is true

if($a or $b) echo 'a or b is true';
else echo 'a or b is false';
// afishon a or b is true

if (!$a) echo 'a is false';
else echo 'a is true';
// afishon a is false
?>
```

Ushtrime

1. Çfarë afishon kodi i mëposhtëm:

```
<?php
$a=40;
$b=21;
$d=$a/--$b;
echo $d;
echo '<br>'.$b;
?>
```

2. Çfarë afishon kodi i mëposhtëm:

```
<?php
$a=40;
$b=20;
$d=$a/$b--;
echo $d;
echo '<br>'.$b;
?>
```

3 -Krijoni një variabël me emrin \$number me një vlerë numerike cfarëdo. Afishoni numrin dhe katrorin e tij (\$number*\$number).

4- Duke ditur që formula sip = $(3.14) \cdot \text{rreze} \cdot \text{rreze}$, llogarit sipërfaqen e një rrethi , gjeni dhe afishoni sipërfaqen e një rrethi me rreze 7.

5- Kopjoni kodin më poshte. Provoni vlera të ndryshme për \$a dhe \$b.

```
<?php
$a = 5 ;
$b = 7 ;
if ($a < 7 && $b > 5)
{
    print "<p>true</p>" ;
}
else
{
    print "<p>false</p>" ;
}
```

?>

8. Strukturat e kontrollit ne Php

Strukturat e kontrollit në PHP janë të ngjashme me gjuhë të tjera të programimit si Java dhe C. Sintaksa e tyre është si më poshtë:

```
if (cond) {...}; optional else {...}    // degëzimi I kodit
for(init; cond; inc) { .... }          // cikli for
while (cond) { .... }                  // cikli while
do {...} while (cond)                  // cikli do while
switch case                            // degezimi me disa drejtime
```

Nëpërmjet shembujve të mëposhtëm do të ilustrojmë përdorimin e strukturave të mësipërme të kontrollit:

8.1 Degëzimi i kodit nëpërmjet If .. else

Shembull i përdorimit të if (cond) {...}; else {...}

```
<?php
$a=10;
$b=12;
    if ($a > $b)
    {
        echo "a është më e madhe se b";
    }
elseif ($a == $b)
{
    echo "a është e barabartë me b";
}
else
{
    echo "a është më e vogël se
b"
}
// në fund afishon a është më e vogël se
b
?>
```

8.2 Cikli FOR

Sintaksa e përgjithshme e ciklit for është :

```
for(shprehje inicializimi; shprehje llogjike; shprehje inkrementimi) {  
<bllok instruksionesh> }
```

Shembujt e mëposhtëm ilustrojnë përdorimin e funksionit for :

```
<?php
```

```
/*          shembull          1,          afishon          12345678910  
*/
```

```
for ($i = 1; $i <= 10; $i++)  
{
```

```
    echo $i;
```

```
}
```

```
/* shembull 2 afishon 12345678910 , dilet nga cikli me break, mungon  
shprehja llogjike  
*/
```

```
    for ($i = 1; ; $i++)
```

```
{
```

```
if ($i > 10)  
{
```

```
    break;
```

```
{
```

```
    echo $i;
```

```
}
```

```
/* shembull 3, inicializimi është para ciklit, shprehja llogjike mungon,  
dilet nga cikli me break, shprehja e inkrementimit vendoset brenda  
ciklit
```

```
*/  
  
$i =  
1;  
  
for (; ; )  
{  
  
if ($i > 10)  
{  
  
break;  
  
}  
  
echo $i;  
  
$i++;  
  
}  
  
?>
```

8.3 Cikli While dhe cikli do .. while

Shembujt e mëposhtëm shpjegojnë mënyrën e përdorimit të ciklit while.

Sintaksa e përgjithshme për ciklin while është:

```
while (kusht) {...}
```

Në dallim nga cikli for, shprehja inicializuese (nëse ka një të tillë), duhet vendosur përpara ciklit while.

Shprehja llogjike vendoset në ciklin while while (kusht), llogaritet vlera e saj dhe nëse ajo është e vërtetë (true) ekzekutohet blloku i instruksioneve brenda shenjës begin { dhe end }. Në fund të bllokut të instruksioneve brenda {...}, testohet sërish vlera e shprehjes llogjike. Nëse ajo është e vërtetë sërish ekzekutohet blloku i

instruksioneve e kështu me rradhë derisa shprehja llogjike të bëhet false.

Sic duket është e nevojshme që të vendoset brenda bllokut të instruksioneve një instruksion (shprehja e inkrementimit tek cikli for) i cili ndryshon gjendjen e shprehjes llogjike, në menyrë që të dilet nga cikli kur të plotësohet një kusht i caktuar, psh $\$i < 10$;

Shembull while (cond) { }

```
<?php
/* shembull 1 */
$i = 1;
while ($i <= 10)
{
    echo $i;
    $i++;      /* afishon vlerën e $i e më pas e rrit $i me 1, afishon
12...10 */
}
?>
```

Cikli do ..while

Në dallim nga cikli while , cikli do.. while së pari ekzekuton bllokun e instruksioneve midis begin dhe end {...} dhe më pas teston vlerën e shprehjes llogjike. Për këtë arsye instruksionet e ciklit do..while do të ekzekutohen të paktën një herë pavarësisht vlerës së shprehjes llogjike.

Sintaksa e përgjithshme e ciklit do..while është:
do {...} while (cond)

Për shembull, kodi i mëposhtëm do të afishonte: 0

```
<?php
$i = 0;
do
{
    echo $i;
} while ($i > 0);
?>
```

Për shembull, kodi i mëposhtëm do të afishonte: 0123456789

```
<?php
$i = 0;
do
{
    echo $i;
    $i=$i+1;
} while ($i < 10);
?>
```

8.4 Degëzimi i kodit me instruksionin switch .. case

Shembujt e mëposhtëm ilustrojnë përdorimin e instruksionit switch case për degëzimin në disa drejtime të kodit:

Shembull 1

```
<?php

if ($i == 0) {
    echo "i është e barabartë me 0";
} elseif ($i == 1) {
    echo "i është e barabartë me 1";
} elseif ($i == 2) {
    echo "i është e barabartë me 2";
}
/* i njëjti funksionalitet me Switch */

switch ($i) {
    case 0:
        echo "i është e barabartë me 0";
        break;
    case 1:
        echo "i është e barabartë me 1";
        break;
    case 2:
```

```
    echo "i është e barabartë me 2";  
    break;  
}  
?>
```

Shembull 2

```
<?php  
switch ($i) {  
    case "molle":  
        echo "i eshte molle";  
        break;  
    case "dardhe":  
        echo "i eshte dardhe";  
        break;  
    case "pjeshe":  
        echo "i eshte pjeshke";  
        break;  
}  
?>
```

Kombinimi i kodit PHP me kodin HTML dhe kodin Javascript

Përgjithësisht kodi HTML si edhe kodi Javascript mund të gjenerohen nga komandat echo ose print te një faqe php.

Në disa raste është më e lehtë që të shkruhet kodi sipas një teknike që kërkon një përzierje të kodit PHP me kodin HTML dhe javascript.

Për këto raste përdoret termi "kodi spaghetti", për shkak të përzierjes së kodit PHP dhe HTML.

Nëpërmjet shembullit të mëposhtëm ilustronim këtë teknike.

Shembull - Të afishohet një tabelë me 10 rreshta me kode dhe URL :
Kodi i faqes PHP:

```
<html>  
<head>  
<title>PHP , HTML Spagheti mode</title>  
<link href="css/style.css" rel="stylesheet" type="text/css">  
<script language="JavaScript">
```

```
function whereami(i)
{
alert('Keni klikuar ne rreshtin '+i);
}
</script>

</head>
<body>
<table class="cope1">
<thead>
<tr>
<td>Nr</td>
<td>Artikulli</td>
</tr>
</thead>
<tbody>
<?php
for($i=1;$i<10;$i++)
{
?>
<tr      onClick="whereami(<?php      echo      $i;      ?>);"
onMouseOver="this.bgColor='#99CCFF'"
onMouseOut="this.bgColor='#FFFFFF'">
<td><?php echo $i; ?></td>
<td><?php echo 'URL '.$i; ?></td>
</tr>
<?php
}
?>
</tbody>
</table>
</body>
</html>
```

Sic duket edhe nga shembulli më sipër kodi php mund të ndërprehet dhe të përfshijë kod HTML dhe Javascript. Le të shikojmë një shembull tjetër që ilustron përdorimin e cikleve në PHP.

Të ndërtojme një faqe php e cila afishon një tabelë shahu. Nëse përdoruesi klikon në njëren nga kutitë e fushës së shahut afishohet

një dritare që tregon koordinatat e qelizës ku klikuam. Më poshtë po paraqesim të dhënat e faqes shahu.php

```
<html>
<head>
<title>shembull 2</title>
<script language="JavaScript">
function whereami(i,j)
{
alert('Jeni ne rreshtin e '+i+'ne kollonen e '+j);
}
</script>

<?php
$name='Endri';
?>
</head>
<body>
<b>
<?php
echo 'Pershendetje '.$name;
?>
</b>
<?php

$n=8;
$m=8;

echo '<table width="400" border="1">';
for($i=1;$i<=8;$i=$i+1)
{
    echo ' <tr  >';
    for($j=1;$j<=8;$j++)
    {
        echo '          <td          width="50"          height="50"
onClick="javascript:whereami('.$i.','.$j.');"
if (((($i+$j)%2)==0)
{
        echo 'bgColor="#000000"; >';
```

```
}  
else  
{  
    echo 'bgColor="#FFFFFF";>'  
}  
  
echo '</td>  
';  
}  
echo '</tr>';  
}  
echo '</table>';  
  
?>  
</body>  
</html>
```


Ushtrime

1-Ndërtoni një faqe PHP që afishon një piramidë me *, me dimensionin e bazës së të cilës të barabartë n. Duke përdorur ciklin for.

```
*
***
*****
*****
*****
*****
*****
*****
*****
```

2- Ndërtoni një faqe PHP që afishon një piramidë me *, me dimensionin e bazës së të cilës të barabarte n. Duke përdorur ciklin while

3-Ndërtoni një faqe PHP që afishon një piramide me *, me dimensionin e bazës së të cilës të barabartë n. Duke përdorur ciklin do..while

4- Ndërtoni një faqe PHP e cila afishon kufizat e vargut $a(n)=2*n*2$ dhe shumën e tyre. Kufizat e vargut duhet të afishohen nga 15 kufiza për rresht dhe të ndara me dy hapësira (space) midis tyre. Pas afishimit të kufizave të vargut faqja PHP duhet të lërë tre rreshta bosh dhe të afishojë shumën e kufizave të vargut.

5- Ndërtoni një faqe PHP e cila afishon kufizat e vargut $a(n)=n$ në fuqi n dhe mesataren e tyre. Kufizat e vargut duhet të afishohen nga 5 kufiza për rresht dhe të ndara me dy hapësira (space) midis tyre. Pas afishimit të kufizave të vargut faqja PHP duhet të lërë tre rreshta bosh dhe të afishojë mesataren e kufizave të vargut. Mesatarja duhet të llogaritet pa gjetur më parë shumën e kufizave të vargut. Numri i kufizave të merret, $n=15$.

9. Stringjet në PHP

Stringjet përmbajnë karaktere 8-bitëshe dhe rrethohen me thonjëza dyshe ose me thonjëza teke. Nëse stringu rrethohet me thonjëza dyshe, thonjëza teke mund të jetë një katakter përbërës i stringut dhe anasjelltas nëse stringjet janë të rrethuar nga thonjëza teke, thonjëza dyshe mund të jetë element përbërës i stringut pa patur nevojë për karaktere speciale.

Për shembull :

```
<?php
$teke = " Shenja ', eshte nje thonjeze teke.";
$dyshe= 'Shenja " eshte thonjeza dyshe.';
?>
```

janë vlerëdhënie korrekte.

Ndërsa nëse një string duhet të përmbajë një thonjëzë të të njëjtit lloj me ato që e rrethojnë stringun, atëherë ky konsiderohet si karakter i vecantë dhe është e nevojshme të paraprihet nga vija e pjerrët \.

Për shembull kodi;

```
<?php
$emri= 'Et\'hem';
echo 'Mire se erdhet ne sistem z. '.$emri;
?>
```

afishon te njeitin rezultat me kodin:

```
<?php
$emri= "Et'hëm";
echo "Mire se erdhet ne sistem z. "$emri;
?>
```

qe do te ishte: Mire se erdhet ne sistem z. Et'hëm

Perkundrazi kodi:

```
<?php
$emri= 'Et\'hem';
echo 'Mire se erdhet ne sistem z. '.$emri;
```

?>

do të shkaktonte një mesazh gabimi të ngjashëm me mesazhin:

*Parse error: syntax error, unexpected T_STRING
in D:\xampp\htdocs\endri\libri\strings.php on line 2*

Gjithashtu, kodi:

```
<?php
$emri= "Et"hem";
echo "Mire se erdhet ne sistem z. "$emri;
?>
```

Do të gjeneronte të njëjtin gabim.

Përvec thonjzes ka edhe simbole të tjera të cilat, nëse duhet të përdoren si pjesë e një stringu duhet të paraprihen nga një vijë e pjerrët\.

Më poshtë po listohen disa prej këtyre simboleve:

\n fund rreshti(LF ose 0x0A (10) ne kodin ASCII)
\\ backslash, vije e pjerrret

Shembujt e mëposhtëm ilustrojnë përdorimin e këtyre simboleve:

```
<?php
```

```
// afishon: Imazhet ruhen ne C:\images
echo ' Imazhet ruhen ne dosjen C:\\images';
// Afishon: Ne browser \n nuk shkakton nje rresht te ri
echo 'Ne browser \n nuk shkakton nje rresht te ri';
?>
```

Në PHP stringjet mund të shtrihen në disa rreshta. Kjo veti është e dobishme në menyrë të vecantë kur stringu përmban kod html, pasi na mundëson që ta organizojmë kodin html në disa rreshta për një lexueshmëri më të mirë nga programuesi.

Për shembull, kodi i mëposhtëm është korrekt:

```
<?php
$t='
```

```
<table>
<tr>
<td>Nr</td>
<td>Emri</td>
</tr>
<tr>
<td>1</td>
<td>Albania</td>
</tr>
</table>';
echo $t;
?>
```

9.2 Funksionet mbi stringjet

PHP është e pajisur me një numër të madh funksionesh mbi stringjet, disa prej të cilave po i listojmë më poshtë:

Bashkimi (konkatenimi) i stringjeve

Për bashkimin e stringjeve përdoret operatori *.(pike)*, psh:

```
<?php
echo "nje " . " dhe " . "dy";
?>
```

Afishon: një dhe dy

Funksionet **ltrim**, **trim**, **rtrim** eliminojnë hapësirat boshe në fillim dhe në fund të stringut.

Gjatësia e stringut

Funksioni **strlen** kthen gjatësinë e stringut të matur sipas numrit të simboleve që ai përmban.

Për shembull, kodi:

```
<?php
$s='Sot eshte kohe e mire';
echo strlen($s);
?>
```

Do të afishonte 21

Konvertimi i stringut në shkronja të mëdha e të vogla

Funksionet **strtolower**, **strtoupper** e konvertojnë një string përkatësisht në shkronja të vogla e të mëdha.

Për shembull, kodi:

```
<?php
$s='Sot eshte kohe e mire';
echo strtolower($s);
echo "<br>";
echo strtoupper($s);
?>
```

Do të afishonte :

```
sot eshte kohe e mire
SOT ESHTE KOHE E MIRE
```

Krahasimi i dy stringjeve

Stringjet jane vargje me karaktere dhe si të tilla nuk mund të krahasohen duke përdorur operatorët e krahasimit që përdoren për numrat.

Për këtë qëllim php është e pajisur me një funksion që krahason dy stringje si vargje karakteresh, duke u bazuar në renditjen alfabetike të shkronjave të tyre; ('a'<'b') pasi ajo ndodhet më përpara në renditjen alfabetike.

Për këtë arsye 'Ana'<'Ani', pasi a<i.

Funksioni **int strcmp (string \$str1, string \$str2)** krahason dy stringjet dhe kthen:

- 1, Nese \$str1<\$str2
- 0, Nese \$str1 eshte identik \$str2 (\$str1=\$str2)
- 1, Nese \$str1>\$str2

Ekstraktimi i një pjese të stringut

Për të marrë një pjesë të stringut php është e pajisur me funksionin **string substr (string \$str, int \$start [, int \$length])** i cili kthen një pjesë të stringut.

Argumenti \$str është stringu nga ku do të ekstraktohet një pjesë, \$start është pozicioni në string ku nis pjesa që do të ekstraktohet dhe argumenti i tretë \$length përcakton gjatësinë në karaktere të pjesës që do të ekstraktohet.

Parametri `length` është opsional dhe nëse nuk vendoset funksioni `substr` ekstraktën një pjesë të stringut duke nisur nga pozicioni i përcaktuar nga argumenti `$start` e deri në fund të stringut.

Funksioni `substr`, është i dobishëm në disa raste si për shembull në rastin kur duam të afishojmë vetëm një preview të lajmeve në një website, kur duam të ekstraktojmë pjesë nga informacionet që morëm nga përdoruesi etj.

Më poshtë paraqiten disa shembuj të përdorimit të funksionit `substr`:

```
<?php
echo substr('abcdef', 1); // afishon bcdef
echo substr('abcdef', 1, 3); // afishon bcd
echo substr('abcdef', 0, 4); // afishon abcd
echo substr('abcdef', 0, 8); // afishon abcdef
?>
```

Funksioni `substr`, shpesh përdoret i kombinuar me funksionin **`strpos`**, i cili kthen pozicionin e vendodhjes së një sekuece karakteresh brenda një stringu.

Prototipi i funksionit `strpos` është si mëposhtë:

`int strpos(string $str, string $srchstr, int $startfrom)`

Argumenti i parë `$str` është stringu ku do të kërkohet, argumenti i dytë `$srchstr` është stringu për të cilin do të kërkohet brenda stringut `$str` dhe argumenti i tretë `$startfrom` përcakton se nga cili pozicion do ta fillojmë kërkimin.

Shembulli më poshtë ilustron përdorimin e funksionit **`strpos`**:

```
<?php
$newstring = 'abcdef abcdef';
$pos = strpos($newstring, 'a', 0); // afishon 0
echo $pos;
$pos = strpos($newstring, 'a', 1);
echo $pos; // afishon 7 jo 0, pasi kërkimi fillon nga pozicioni
1
```


?>

9.2 Ruajtja e rreshtave të rinj nga teksti në html

Sic dihet, browseri përdor një tag HTML për të përfunduar një rresht të tekstit që po afishon dhe për të vazhduar në një rresht të ri, tagun `
`.

Editorët e tekstit përdorin kombinim `\r\n` (carriage return, new line) për të sinjalizuar fundin e një rreshti dhe kalimin në rresht të ri.

Funksioni **nl2br** merr si argument një tekst dhe e kthen atë pasi ka zëvendësuar `\r\n` me tagun HTML `
`

Shembulli më poshtë ilustron përdorimin e funksionit `nl2br`;

```
<?php
echo nl2br("Pershendetje \r\nKy eshte dokument html");
?>
```

Do të afishonte në browser
Përsëndetje
Ky është dokument html

Ndërsa kodi:

```
<?php
echo "Pershendetje \r\nKy eshte dokument html";
?>
```

Do të afishonte
Përsëndetje Ky është dokument html

Ushtrime

Le të jetë faqja lajme.php si më poshtë:

```
<?php
```

```
$news='
```

Sipas një studimi të ri, një bisedë një orëshe me telefon celular shkakton rritje të aktivitetit biokimik tek përdoruesi.

Studimi nuk thotë nëse kjo është gjë e mirë apo e keqe për shëndetin, por ai ka ringjallur debatin mbi sigurinë e telefonave celularë dhe ka bërë që të ndërmerren studime të tjera mbi efektet e telefonit celular tek truri.';

```
// ketu do te vendosni kodin Tuaj
```

```
?>
```

Duke përdorur funksionet e php mbi stringjet;

Ushtrim 1 - Shkruani kodin php që ekstrakton fjalinë e parë dhe e afishon atë.

Ushtrim 2 - Shkruani kodin php që ekstrakton fjalinë e dytë dhe e afishon atë.

Ushtrim 3 - Shkruani kodin php që afishon numrin e fjalive në këtë tekst dhe gjatësinë e tyre në numër shkronjash.

Ushtrim 4 - Shkruani kodin php që afishon sa herë është përdorur shkronja ë në këtë tekst

Ushtrim 5 - Shkruani kodin php që afishon të gjithë tekstin e dhënë por duke konvertuar të gjithë shfaqet e fjalës celular në CELUAR .

Ushtrim 6- Shkruani kodin php që afishon fjalinë e parë të tekstit të dhënë duke ruajtur thyerjet e rreshtave.

10. Funksionet dhe modulimi i kodit

Funksionet në PHP

Funksionet janë një njësi llogjike pune që na lehtësojnë shkrimin e kodit, duke krijuar kod të ripërdorshëm.

Funksionet mund të vendosen në skedare librarie të cilët më pas mund të përfshihen në faqet php, që kanë nevojë të përdorin funksionet.

Sintaksa për të deklaruar një funksion në php është:

```
<?php
function f ($arg_1, $arg_2, ..., $arg_n) {
// trupi i funksionit
echo "Funksion shembull.\n";
$retvat=100;
return $retval; //kthe një vlerë
}
?>
```

Ndryshe nga emrat e variablave, emrat e funksioneve nuk janë case sensitive.

Për shembull kodi i mëposhtëm do të afishonte 2 herë 10

```
<?php
function shuma($a,$b)
{
return ($a+$b);
}
echo shuma(5,5); //afishon 10
echo "<br>";
echo SHUMA(5,5); //afishon 10 nuk jep mesazh gabimi ndonëse
emri i funksionit është me shkronja të mëdha
?>
```

Ndërsa kodi i mëposhtëm nuk do të ekzekutohej pasi do të shkaktonte një mesazh gabimi, për arsye se ndërsa argumentat e funksionit janë deklaruar me emrat \$a dhe \$b dhe në brendësi të

funksionit, argumentave i referohemi me \$A dhe \$B, pra me shkronja të mëdha.

```
<?php
function shuma($a,$b)
{
    return ($A+$B);
}
echo shuma(5,5);
?>
```

10. 1 Kalimi i argumentave të një funksioni

Kalimi i argumentave të një funksioni mund të bëhet me vlerë ose me referencë.

Nëse argumenti i kalohet si vlerë atëherë funksioni krijon një kopje lokale të variablit që i kaluam si argument.

Ndryshimet që bëhen brenda trupit të funksionit aplikohen në kopjen lokale të variablit që i kaluam si argument dhe nuk prekin vlerën e variablit që iu dha si argument funksionit. Kjo është mënyra bazë e kalimit të argumentave një funksioni.

Si shembull i kalimit të argumentave funksionit si vlerë mund të shikojmë funksionin e shumës që krijuam pak më lart.

Nëse duam që një variabël t'ia kalojmë funksionit si referencë do të duhet ta bëjmë këtë në mënyrë eksplicite duke i vendosur përpara emrit të argumentit shenjën &.

Mund të ktheni disa vlera nga një funksion duke kthyer një array.

Si default funksionit

i kalohen

argumentat si vlerë.

Nëse duam që ti kalojmë një argument si referencë (ndryshimi të ruhet edhe jashtë funksionit) atëherë emri i argumentit do të paraprihet me shenjën &.

Shembulli më poshtë ilustron kalimin e argumentave si vlerë dhe si referencë:

```
<?php
```

```
function katrori1($var)
{
    $var=$var*$var;
    return $var;
}

function katrori2(&$var)
{
    $var=$var*$var;
    return $var;
}

$a=5;
$k1=katrori1($a);
echo '<br>'.$a; // afishon 5
echo '<br>'.$k1; // afishon 25

$k2=katrori2($a);
echo '<br>'.$a; // afishon 25 vlera e a është 25
echo '<br>'.$k2; // afishon 25
?>
```

Në shembullin e mësipërm janë ndërtuar dy funksione identike `katrori1` dhe `katrori2`, të cilët gjejnë dhe kthejnë katrorin e vlerës së variablit që marrin si argument.

I vetmi dallim midis dy funksioneve është që njëri (`katrori1`) e merr argumentin si vlerë ndërsa tjetri (`katrori2`) e merr argumentin si referencë.

Të dy funksionet thërriten për të gjetur katrorin e variablit `$a`, i cili është inicializuar paraprakisht me vlerën 5.

Instrukcioni `$k1=katrori1($a);` thërret në ekzekutim funksionin `katrori1` dhe i kalon atij si argument **vlerën** e variablit `$a`, pra vlerën 5. Funksioni `katrori1` bën një kopje lokale të variablit `$a` nën emrin `$var` dhe vendos aty katrorin e 5, vlerën 25:

```
    $var=$var*$var;
së fundi kthen këtë vlerë:
    return $var;
```

vlera e kthyer nga funksioni i kalohet variablit \$k1 dhe më pas kur afishojme \$a dhe \$k1, ato do të kenë përkatësisht vlerat 5 dhe 25.

Vlera e variablit \$a nuk ka ndryshuar.

Ndërsa në rreshtat më poshtë thërret në ekzekutim funksioni katrori2 me argument variablin \$a

```
$k2=katrori2($a);
```

kur afishojmë \$a dhe \$k2, pasi kemi thërritur funksionin katrori2 ato kanë përkatësisht vlerat 25 dhe 25.

Si ka mundësi që \$a është bërë 25?

Përgjigja është tek mënyra se si funksioni katrori2 merr argumentin: function katrori2(&\$var)

Shenja & (and) përpara emrit të variablit \$var i tregon funksionit që argumentin duhet ta marrë si referencë dhe jo si vlerë.

Në këtë rast \$var bëhet një shënjes mbi vlerën e variablit \$a, është si një sinonim për variablin \$a. Si \$var ashtu edhe \$a referojnë të njëjtën zonë të memorjes ku ndodhet vlera e variablit \$a. Për këtë arsye kur modifikojmë variablin \$var modifikojmë edhe variablin \$a, pasi ata referojnë të dy të njëjtën zonë të memorjes.

Për këtë arsye kur brenda funksionit ekzekutohet instruksioni:

```
$var=$var*$var;
```

vlera që vendoset tek variabli \$var, $5*5=25$ është vlera që vendoset gjithashtu edhe vlera e variablit \$a. Prandaj kur afishojmë vlerën e \$a ajo është 25.

10. 2 Funksionet rekursive

Funksionet rekursive janë ato funksione që thërrasin vetveten. Funksionet rekursive janë shumë efektive në zgjidhjen e problemeve që mund të copëtohen në probleme më të vegjël të të njëjtës natyrë që varen nga njëri tjetri.

Le ta ilustrojme këtë me një shembull:

Shembull 1 Rekursiviteti - Ndërtoni një funksion që gjen shumën e n numrave të parë natyrorë (1,2,3,4,5,6,...)

```
<?php
```

```
// gjen shumën e n numrave të parë natyrorë
```

```
function shumarec($i,$n)
{
if ($i<$n) return $i+shumarec($i+1,$n);
else return $i;
}
echo shumarec(0,10); // afishon 55
?>
```

10.3 Përfshirja e skedareve në skedarë të tjerë PHP

Funksionet mund të mblidhen në skedarë librarie dhe këta të fundit mund të përfshihen sa herë që është e nevojshme për të përdorur këto funksione.

Për përfshirjen e skedarëve php ofron katër instruksione:

```
include()
require()
```

dhe

```
include_once()
require_once()
```

Le të ilustrojmë përdorimin e include dhe require nëpërmjet një shembulli:

Për shembull, faqja më poshtë nuk do të afishonte asgjë në browser :

```
<?php
require("kot.php");
?>
<html>
<head>
<title>Include vs Require</title>
</head>
<body>
<h1>
Sot eshte nje dite e bukur!
```



```
</h1>
</body>
</html>
```

do të gjeneronte një mesazh gabimi të ngjashëm me mesazhin e mëposhtëm:

Warning: require(kot.php) [function.require]: failed to open stream: No such file or directory in D:\xampp\htdocs\endri\libri\include.php on line 2

Fatal error: require() [function.require]: Failed opening required 'kot.php' (include_path='.;D:\xampp\php\PEAR') in D:\xampp\htdocs\endri\libri\include.php on line 2

dhe nuk do të afishonte në browser mesazhin Sot është një ditë e bukur!.

Sic duket ka dy mesazhe gabimi: i pari, është një paralajmërim (Warning) ndërsa i dyti është një gabim (Fatal error).

Në momentin që ekzekutohet instruksioni `require("kot.php");` gjenerohet një mesazh gabimi dhe ekzekutimi nuk vazhdon më tej.

Shembulli i mësipërm tregon se komanda `require` është një kërkesë e fortë për të përfshirë një skedar php në faqen tonë. Nëse skedari php që po tentojmë të përfshijme nuk gjendet, si në rastin e `kot.php`, gjenerohet një mesazh gabimi dhe ekzekutimi ndërpritet në këtë pikë. Ndërsa komanda `include`, është më tolerante në këtë aspekt. Nëse skedari që po përfshihet nuk gjendet, gjenerohet në një mesazh paralajmërimi (warning) dhe ekzekutimi vazhdon, në rreshtat kodit më poshtë.

Për shembull, faqja php :

```
<?php
include("kot.php");
?>
<html>
<head>
<title>Include vs Require</title>
```

```
</head>
<body>
<h1>
Sot eshte nje dite e bukur!
</h1>
</body>
</html>
```

Do te afishonte:

Warning: include(kot.php) [function.include]: failed to open stream: No such file or directory in D:\xampp\htdocs\endri\libri\include.php on line 2

Warning: include() [function.include]: Failed opening 'kot.php' for inclusion (include_path='.;D:\xampp\php\PEAR') in D:\xampp\htdocs\endri\libri\include.php on line 2

Sot është një ditë e bukur!

Siç duket edhe nga shembulli , ndryshme nga require, funksioni include nuk ndërpret ekzekutimin e faqes.

Të dy funksionet kanë versionin përkatës që e përfshin një skedar vetëm një herë.

Nganjëherë mund të jetë e nevojshme që një skedar ta përfshijme disa herë, për këtë arsye që të dy funksionet include dhe require, e përfshijnë një skedar brenda një faqe aq herë sa e thërrasim.

Për, shembull nëse kemi vendosur në një faqe kodin për një menu, mund të duam që menunë ta afishojmë në disa pjesë të faqes për një bredhje më të lehtë nga ana e përdoruesit, psh

faqja menu.php

```
<?php
echo '
<ul>
<li><a href="#">item 1</a></li>
<li><a href="#">item 2</a></li>
<li><a href="#">item 3</a></li>
<li><a href="#">item 4</a></li>
```

```
<li><a href="#">item 5</a></li>
</ul>';
?>
```

faqja index.php

```
<html>
<head>
<title>Mire se vini ne siten tone!</title>
</head>
<body>
<?php
include("menu.php");
?>
<h1>
Sot eshte nje dite e bukur!
</h1>
<?php
include("menu.php");
?>
</body>
</html>
```

Sic, duket edhe nga shembulli më lart, menunë e kemi përfshirë dy herë në faqe njëherë në fillimë faqes një herë në fund të faqes.

Faqja index.php do të afishontë:

[item 1](#)

[item 2](#)

[item 3](#)

[item 4](#)

[item 5](#)

Sot është një dite e bukur!

[item 1](#)

[item 2](#)

[item 3](#)

[item 4](#)

[item 5](#)

Në disa situata të tjera kjo mënyrë nuk është e përshtatshme. Për shembull, nëse kemi ndërtuar një faqe db.php, në të cilën kemi vendosur kodin php për të realizuar lidhjen me bazën e të dhënave. Përfshirja dy here e kesaj faqe do te tentonte te krijonte dy here lidhje me bazën e të dhënave.

Për të tilla situata, përdoren versionet e include the require që i përfshijnë vetëm një herë skedaret. Këto funksione janë përkatësisht **include_once** dhe **require_once**. Nëse në një faqe kemi përfshire dy herë një skedar , psh db.php nëpërmjet include_once ose require_once , skedari do të përfshihet vetëm herën e pare. Herën e dytë që haset include_once("db.php") instruksioni nuk do të merret parasysh.

Për shembull faqja index.php, nëse në të do të zëvendësonim include("menu.php"); me include_once("menu.php"); do ta përfshinte menunë vetëm herën e parë. Pra faqja index.php:

```
<html>
<head>
<title>Mire se vini ne siten tone!</title>
</head>
<body>
<?php
include_once("menu.php");
?>
<h1>
Sot eshte nje dite e bukur!
</h1>
<?php
include_once("menu.php");
?>
</body>
</html>
```

Do te afishonte:

[item 1](#)

[item 2](#)

[item 3](#)

[item 4](#)

[item 5](#)

Sot eshte nje dite e bukur!

Siç duket edhe nga rezultati i afishimit në browser, menu.php është përfshirë vetëm njëherë në fillim të faqes, pra herën e parë që thërritet `include_once("menu.php");` ndërsa hera e dytë që thërritet ky instruksion nuk merret parasysh.

Ushtrime

Ushtrim 1 - Ndërtoni një funksion rekursiv i cili afishon n numrat e parë çift duke nisur nga numrat më të mëdhenj deri tek 2.

Funksioni do të gjejë dhe të kthejë shumën e numrave që afishon prototipi i funksionit `function çift($n)`

Nëse e thërrasim në një instruksion `echo 'shuma është:'. çift(5);` në browser duhet të afishohet

10 8 6 4 2 shuma është: 30

Ushtrim 2 - Ndërtoni një funksion rekursiv i cili merr si argument një numër të plotë n dhe afishon një tabelë html me dimensione $n \times n$, ku elementet e diagonales kryesore të jenë të ngjyrosur me ngjyrë të zezë.

Ushtrim 3 - Ndërtoni një funksion rekursiv i cili merr si argument një string dhe e afishon atë në rendin e kundërt, për shembull merr si argument sot dhe kthen tos.

Ushtrim 4 - Ndërtoni një funksion rekursiv i cili merr si argument një fjalë dhe kthen tekstin "palindrome" nëse fjala është palindrome ose tekstin "nuk është palindrome" nëse fjala nuk është palindrome. Palindrome quhen fjalët që lexohen njësoj si nga e majta në të djathtë edhe nga e djathta në të majtë, për shembull sos - sos, soros-soros, stressed -desserts, *gateman* - *nametag*

11. Komunikimi me browserin

Deri më tani, kemi parë që faqet php janë programe (skripte) të cilat ekzekutohen në server dhe standart output-i, dhe standart error-i, i tyre janë protokollin http, i cili kujdeset që ta transferojë output-in, nga serveri tek browseri që e thërri këtë program, duke bërë një kërkesë për faqen.

Si ti kalojmë argumenta këtyre programeve?

Si ti kalojmë argumenta faqeve php?

Protokolli http, në formatin e mesazhit request që përgatit browseri për të bërë kërkesë për një faqe një server, ka parashikuar disa hapësira ku browseri vendos parametra që do ti kalojë faqes php, në server.

Kur kërkesa mbërrin në server, php i ofron programuesit disa objekte që i lejojnë atij të kape këto argumenta.

Për shembull kur përdoruesi klikon mbi një link të një lajmi në një faqe web, browseri formulon një kërkesë http për faqen lajm.php të cilës duhet ti dërgojë si argument kodin e lajmit që përdoruesi kërkon të lexojë? Në këtë rast linku për lajmin do të ishte i ngjashëm me lajm.php?item=10, ku item do të ishte emri i argumentit që i kalojmë web serverit, edhe në php me këtë emër do ta kërkojmë argumentin ndërsa 10 është vlera e këtij argumenti.

Kjo mënyrë e kalimit të argumentave një faqeje php, që i bashkëngjit emrit të faqes php, një liste me argumenta të ndarë me ? nga emri i faqes, quhet kalimi i argumentave nëpërmjet një stringu URL. Argumentat janë të tipit emër=vlerë dhe ndahen nga njëri-tjetri me shenjën &, për

shembull http://localhost/hello.php?une=alma&dita=e_hene

Kjo metode përdor metodën GET, të dërgimit të kërkesës nga browseri në server.

Përvec, metodës GET ka edhe një metodë POST, e cila në dallim nga metoda GET është më e sigurtë dhe mundëson ngarkimin e në server të një sasive më të madhe të dhënash. Nga ana tjetër, metoda POST, ç'aktivizon butonin BACK të browserit. Nuk është e mundur të shkojmë BACK nga një faqe e afishuar, kërkesa për të cilën është bërë me metodën POST.

Po e ilustruam me një shembull, mënyrën e kalimit të argumentave tek një faqe php, duke i bashkëngjitur ato në adresën e faqes në browser, si string URL.

Nëse përdoruesi vendos në dritarin e browserit `http://localhost/hello.php?une=alma&dita=e_hënë`, browseri bën një kërkesë për faqen `hello.php`, të ciles i bashkëngjitet dy argumenta:

argumentin me emrin **unë** i cili ka vlerën **alma** dhe argumentin me emrin **dita** që ka vlerën **e_hënë**.

Kur kërkesa shkon në serverin e web-it, faja thërritet në ekzekutim dhe kodi php i saj, mund të përdorë emrat e këtyre argumentave për të kapur vlerat e tyre. Bazuar në këto vlera mund të organizojë logjikën e kodit në varësi të argumentave.

Më poshtë po paraqesim kodin e faqes `hello.php`. Vini re në shembullin më poshtë se si kapen vlerat e argumentave që ka dërguar përdoruesi.

```
<html>
<head>
<title>Hello </title>
<?php
$emri=$_REQUEST['une'];
$sot=$_REQUEST['dita'];
?>
</head>
<body>
<b>
<?php
echo "Pershendetje ".$emri;
echo "<br> Mire se erdhet ne sistem. Sot eshte ".$sot;
?>
</b>
</body>
</html>
```

Rezultati i ekzekutimit të faqes do të transmetohet në browser nga protokollet e rrjetit dhe browseri do të marrte një HTML si më poshtë:

```
<html>
```



```
<head>
<title>Hello </title>
</head>
<body>
<b>
Pershendetje alma
Mire se erdhe ne sistem sot eshte e_hene
</b>
</body>
</html>
```

Kapja e të dhënave nga format HTML (textbox, radio button etj)

Ne kemi mesuar qe te shohim aplikacione web, ku ne plotesojme formulare, dritare logimi etj. Si i kapim vlerat qe vendos perdoruesi ne nje forme ne faqen php ne server?

Në mënyrë të ngashme mund të kapen edhe argumentat që vijnë nga format html.

Po e ilustruon këtë me një shembull. Do të ndërtojmë dy faqe :

1. Faqen login.php që afishon në formë login që i jep mund përdoruesit të futë username dhe fjalëkalimin
2. Faqen checklogin.php që kap të dhënat e username dhe fjalëkalimit dhe i afishon ato

Faqja login.php

```
<html>
<head>
<title>Hello </title>
</head>
<body>
<form method='POST' name='loginfrm' action='checklogin.php'>
Futni kodin dhe fjalekalimin<br>
Username :<input type='text' name='username'><br>
Fjalekalimi :<input type='password' name='fjalekalimi'><br>
<input type='submit' name='submit' value='submit'>
</form>
</body>
</html>
```

Faqja login.php do të shfaqte në browser:

Faqja checklogin.php

```
<?php
echo 'username që futet është:';
echo $_REQUEST['username'];
echo '<br>';
echo 'fjalëkalimi që futet është:';
echo $_REQUEST['fjalëkalimi'];
?>
```

Siç duket edhe nga shembulli, faqja login.php përmban formën ku janë krijuar kutitë username (`<input type='text' name='username'>`) dhe fjalëkalimi (`<input type='text' name='password'>`). Forma ka si destinacion checklogin.php dhe përdor metodën POST për të dërguar të dhënat tek faqja checklogin.php (`<form method='post' name='loginfrm' action='checklogin.php'>`)

Kur përdoruesi klikon në butonin submit, browseri formulon një mesazh të tipit HTTP REQUEST, për faqen që është në vetinë **action**, të formës, pra për faqen checklogin.php

Në trupin e kërkesës browseri vendos argumentat që i lakohen faqes PHP. Kur kërkesa mbërrin në server, kodi PHP i faqes checklogin.php, mund të përdorë emrat e kutive për të lexuar vlerat e tyre, përkatesisht `$_REQUEST['username']` dhe `$_REQUEST['fjalëkalimi']`.

Më tej për efekt ilustrimi faqja checklogin.php i afishon ato në browser, duke afishuar:

Vini re që komandat `echo $_REQUEST['username'];` dhe `echo $_REQUEST['fjalëkalimi'];` afishojnë tekstin që përdoruesi vendosi në kutitë e username dhe fjalëkalimit.

Gjithashtu vini re adresën që shfaqet në kutinë e adresës së browserit. Forma është dërguar nga browseri nëpërmjet metodës POST. Nëse forma do të ishte dërguar nëpërmjet metodës GET, atëherë username dhe fjalëkalimi do të shfaqeshin në kutinë e adresës së browserit.

Më poshtë shfaqet sërish faja login.php , e modifikuar në mënyrë të tillë që forma të përdorë metodën GET (<form method='GET' name='loginfrm' action='checklogin.php'>), në vend të metodës POST:

```
<html>
<head>
<title>Hello </title>
</head>
<body>
<form method='GET' name='loginfrm' action='checklogin.php'>
Futni kodin dhe fjalekalimin<br>
Username :<input type='text' name='username'><br>
Fjalëkalimi :<input type='password' name='fjalëkalimi'><br>
<input type='submit' name='submit' value='submit'>
</form>
</body>
</html>
```

Nëse përdoruesi do të vendoste sërish të njëjtat vlera dhe do të klikonte butonin submit, browseri në mënyrë të ngjashme do të formulonte një kërkesë HTTP REQUEST, për faqen checklogin.php, ku do të bashkëngjiste vlerat e kutive, por kësaj rradhe do të përdorte metoden GET. Faja checklogin.php edhe në këto raste do të përdorte echo `$_REQUEST['username'];` dhe echo `$_REQUEST['fjalëkalimi'];` për të kapur vlerat e kutive, por duke qenë se përdoret metoda GET, emrat dhe vlerat e kutive do të shfaqeshin në kutinë e adresës së browserit:

Shembull - Të ndërtojmë një faqe web që të shërbejë si një mjet buxheti

Nëpërmjet këtij shembulli do të ilustrojmë disa koncepte që janë trajtuar deri tani, si edhe do të japim një model për mënyrën e lidhjes së php, javascript dhe html.

Do të ndërtojmë një faqe web e cila do të marrë si argument një numër të plotë që tregon numrin e zerave që do të ketë një buxhet dhe do të afishojë një formë me kutia për numrin, emrin e produktit/sherbimit, cmimin, sasinë dhe vlerën.

Afishimi i faqes do të ishte i ngjashëm me figurën më poshtë:

Vini re, që faqja duhet të thërritet me një argument `items`, që përcakton sasinë e zërave që do të buxheti, në këtë shembull `items=4`. Faqja duhet të afishojë kutite për të vendosur emrin e zërit në buxhet, psh `computer`, `monitor` etj. cmimin, sasinë dhe automatikisht sapo përdoruesi të modifikojë cmimin ose sasinë e një produkti, faqja duhet të llogarisë vlerën e rreshtit dhe totalin në fund. Po paraqesim më poshtë kodin e plotë të faqes `buxheti.php`, të cilin do ta komentojmë më pas.

```
<?php
$items=$_REQUEST['items']; // së pari lexojmë sa elementë do të ketë
buxheti në variablin $items
?>
<html>
<head>
<title>Buxheti</title>
<script language="JavaScript">
function sumline(i)
{
var price,qty,itemvalue;
price=document.getElementById("Item_Price_"+i);
qty=document.getElementById("Item_Qty_"+i);
itemvalue=document.getElementById("Item_Value_"+i);
itemvalue.value=parseFloat(price.value)*parseFloat(qty.value);
//recalculate totals
totals();
}

function totals()
{
var n,i,s;
s=0; // inicializojme shumen me 0
n=<?php echo $items; ?>;
for(i=1;i<=n;i++)
{
itemvalue=document.getElementById("Item_Value_"+i);
s=s+parseFloat(itemvalue.value); // shtojmë vlerën e rreshtit tek totali
}
buxheti.total.value=s;
```

```
}
</script>
</head>
<body>
<form name="buxheti" action="#" method="post">
<table bgcolor="#99CCFF" >
<tr>
<th>No</th>
<th>Item</th>
<th>Price</th>
<th>Quantity</th>
<th>Value</th>
</tr>
<?php
for($i=1;$i<=$items;$i++)
{
?>
<tr>
<td><input
value="<?php echo $i ?>" size="2" type="text" name="Item_<?php ec
ho $i; ?>"></td>
<td><input
size="50" type="text" name="Item_Name_<?php echo $i; ?>" id="Item
_Name_<?php echo $i; ?>" ></td>
<td><input
size="5" value="0.0" type="text" name="Item_Price_<?php echo $i; ?>
" id="Item_Price_<?php echo $i; ?>" onChange="sumline(<?php echo
$i; ?>);" ></td>
<td><input
size="5" value="0.0" type="text" name="Item_Qty_<?php echo $i; ?>"
id="Item_Qty_<?php echo $i; ?>" onChange="sumline(<?php echo $i;
?>);" ></td>
<td>
<input
size="5" value="0.0" type="text" name="Item_Value_<?php echo $i; ?
>" id="Item_Value_<?php echo $i; ?>"></td>
</tr>
<?php
}
?>
```

```
<tr>
<td>---</td>
<td>TOTAL</td>
<td>---</th>
<td>---</td>
<td><input size="5" type="text" name="total"></td>
</tr>
</table>
</form>
</body>
</html>
```

Ushtrime

Ushtrim 1- Ndërtoni një faqe e cila afishon dy kuti dhe një buton llogarit. Kur përdoruesi të fusë dy numra dhe të shtypë butonin llogarit duhet të afishohet shuma e të dy numrave dhe poshtë shumës sërish dy kutitë dhe butoni llogarit.

12. Array ne PHP

Trajtim i array, në php është paksa i ndryshëm nga gjuhët e programimit tradicionale.

Array nuk ruajnë vetëm vlera të të njëjtit tip të vendosura në mënyrë të njëpasnjëshme dhe të indeksueshme numerikisht.

Në PHP array ciftojmë emrat me vlera, pra indekset mund të jenë edhe tekst jo vetëm numra, array kanë një numër variabël nivelesh.

Kjo mënyrë ruajtje është e përshtatshme për shumë përdorime dhe mund të trajtohet si:

- një matricë normale

- një vektor (matricë njëdimensionale)

- një stive

- një rradhë

- një pemë

- një hash table

- një fjalor

- një bashkësi (collection)

Duke qenë se vlera të një array mund të jenë array të tjerë , është gjithashtu e mundur që të ndërtojmë array multidimensionale.

Vlerat mund të jenë të ndryshme brenda të njëjtit array ndaj themi që array janë heterogjen.

Celësat e array mund të jenë vetëm strings ose integers.

Më poshtë nëpërmjet shembujve do të ilustrojmë përdorimet e array.

12.1 Deklarimi i nje array

Sintaksa e përgjithshme e deklarimit të një array është:

```
array( celes => vlere
```

```
    , ...  
 )
```

```
// celësi mund të jetë vetëm integer ose string
```

```
// value mund të jetë një vlerë e cfarëdo tipi
```


Komanda e php array() merr si parameter nje numer cfaredo me cifte celes(emer) => vlere te ndara me presje dhe krijonj nje array me keto vlere.

Per shembull,

```
<?php
$arr = array("food" => "apple", 12 => true);

echo $arr["food"];    // afishon apple
echo $arr[12];        // afishon 1
?>
```

Siç duket edhe nga shembulli më sipër indekset ose elementet e array mund të jenë të tipve të ndryshëm si tekst (\$arr["food"]); dhe numër i plotë, brenda të njëjtit array (\$arr[12];)

Për më tepër vlerat e një array (jo celësat) mund të jenë array të tjerë. Le ta ilustrojmë këtë me një shembull:

```
<?php
$arr = array("njearray" => array(6 => 5, 13 => 9, "a" => 42));

echo $arr["njearray"][6];    // afishon 5
echo $arr["njearray"][13];   // afishon 9
echo $arr["njearray"]["a"];  // afishon 42
?>
```

Në qoftë se nuk specifikohet celësi në momentin e krijimit të një array atëherë php e llogarit automatikisht duke marrë vlerën INT të rradhës duke i shtuar celësit INT me vlerën më të madhe 1.

Për shembull

```
<?php
// Ky array është i njëjtë ...
array(5 => 43, 32, 56, "b" => 12);

// ...me këtë array
```

```
array(5 => 43, 6 => 32, 7 => 56, "b" => 12);  
?>
```

12.2 Përdorimi i Array

Pas krijimit të array mund të shtojmë e të modifikojmë vlerat e tij sipas nevojës.

Shembulli i mëposhtëm ilustron përdorimin e array

```
<?php  
$arr = array(5 => 1, 12 => 2);  
  
$arr[] = 56; // Kjo komandë është njëlloj si $arr[13] = 56;  
  
$arr["x"] = 42; // Shtojmë një element të ri në array  
// me celësin "x"  
  
unset($arr[5]); // Fshin një element të array  
  
unset($arr); // Fshin të gjithë array  
?>
```

Array si vektor dhe matricë

Nëse deri në këtë pikë array, Ju duken paksa të komplikuar kjo vjen nga fakti që array mund të përdoren për disa struktura të ndryshme të dhënash.

Le të marrim disa shembuj të përdorimit të array si vektorë dhe matrica, sic jemi mësuar ti përdorim në gjuhë të tjera programimi.

Në këtë drejtim php është shumë tolerante, madje nuk kemi nevojë as ti deklarojmë array.

Shembulli më poshtë ilustron përdorimin e array. Është marrë një array me emrin varg në të cilin vendosen kufizat e vargut $a(n)=2*n+1$.

Më pas nëpërmjet një cikli afishohen këto vlera. Më poshtë paraqitet kodi i faqes array.php

```
<?php
```

```
$n=5;
for($i=1;$i<$n;$i++)
{
    $varg[$i]=(2*$i+1);
}

for($i=1;$i<$n;$i++)
{
    echo $i.') 2*'. $i.'+1 = '.$varg[$i];
    echo '<hr>';
}
?>
```

Siç duket edhe nga shembulli cikli i parë for mbush me vlera vektorin me emrin \$varg, ndërsa cikli i dytë for afishon vlerat e vektorit.

Le të marrim një tjetër shembull i cili ilustron përdorimin e array. Po ndërtojmë një array që të ruajë listën e lëndëve që ofron një shkollë, lëndët.php

```
<?php
$lende=5;
$lendet=array(1=>'Matematike',2=>'Fizike',3=>'Letersi',4=>'Informati
ke',5=>'Fiskulture');
for($i=1;$i<=$lende;$i++)
{
    echo $i.') '.$lendet[$i];
    echo '<hr>';
}
?>
```

Kodi i mësipërm ka të njëjtin funksion me versionin e mëposhtëm të faqes lëndët.php

```
<?php
$lende=5;
$lendet[1]='Matematike';
$lendet[2]='Fizike';
$lendet[3]='Letersi';
$lendet[4]='Informatike';
```

```
$lendet[5]='Fiskulture';  
for($i=1;$i<=$lende;$i++)  
{  
    echo $i.') '.$lendet[$i];  
    echo '<hr>';  
}  
?>
```

Le ta çojmë ca më tej shembullin tonë, duke ndërtuar një faqe php e cila përdor dy vektorë dhe një matrice.

Vektori lëndët i cili ruan emrat e lëndëve, vektori studentët që ruan emrat e studentëve dhe matrica notat që ruan notat e çdo studenti në lëndën përkatëse.

Ky shembull ilustron përdorimin e array si vektorë dhe matrica.

Më poshtë po paraqesim kodin për faqen notat.php

```
<?php  
$lende=5;  
$studente=5;
```

```
$lendet[1]='Matematike';  
$lendet[2]='Fizike';  
$lendet[3]='Letersi';  
$lendet[4]='Informatike';  
$lendet[5]='Fiskulture';
```

```
$studentet[1]='Amanda Jones';  
$studentet[2]='Andrew Fuller';  
$studentet[3]='John Smith';  
$studentet[4]='Anne Godsworth';  
$studentet[5]='James Grennespun';
```

```
$notat[1][1]=5;  
$notat[1][2]=5;  
$notat[1][3]=8;  
$notat[1][4]=5;  
$notat[1][5]=9;
```

```
$notat[2][1]=6;
```

```
$notat[2][2]=7;  
$notat[2][3]=8;  
$notat[2][4]=5;  
$notat[2][5]=9;
```

```
$notat[3][1]=5;  
$notat[3][2]=4;  
$notat[3][3]=8;  
$notat[3][4]=5;  
$notat[3][5]=9;
```

```
$notat[4][1]=7;  
$notat[4][2]=10;  
$notat[4][3]=8;  
$notat[4][4]=5;  
$notat[4][5]=9;
```

```
$notat[5][1]=4;  
$notat[5][2]=10;  
$notat[5][3]=8;  
$notat[5][4]=8;  
$notat[5][5]=9;
```

```
echo '<table border=1>  
  <tr>  
    <td>---</td>';  
for($j=1;$j<=$studente;$j++)  
{  
  echo '<th>'.$studentet[$j]. '</th>';  
}  
echo '</tr>';
```

```
for($i=1;$i<=$lende;$i++)  
{  
  echo '<tr>';  
  echo '<th>'.$lendet[$i]. '</th>';  
  for($j=1;$j<=$studente;$j++)  
  {
```

```
echo '<td align="center">'.$notat[$i][$j].</td>';  
}  
}  
echo '</table>';  
?>
```

Siç duket nga shembulli më lart për të kapur vlerat e një matrice përdorim sintaksen `notat[i][j]`.

Ushtrime

Ushtrim 1 - Ndërtoni një funksion në php i cili merr afishon një matricë 8×8 , që përfaqëson një tabelë shahu. Në tabelën e shahut do të vendosen tetë mbretëresha, në menyre të tillë që ato të mos "e hanë" njëra-tjetrën

Ushtrim 2- Ndërtoni një funksion në php me emrin `isordered` i cili merr si argument një vektor me numra dhe kthen 0 ose 1 në varesi të faktit që vektori është ose jo i renditur në rendin rritës

Ushtrim 3-Ndërtoni një funksion `indexof` i cili merr si argumente një vektor me numra të plotë dhe një vlerë. Funksioni kthen vendndodhjen e parë të kësaj vlere në vektori. Nëse kjo vlerë nuk gjendet në vektor atëherë funksioni do të kthejë -1.

Ushtrim 4-Ndërtoni një funksion `indexof` i cili merr si argumente një vektor me numra të plotë dhe një vlerë. Funksioni kthen vendndodhjen e fundit të kësaj vlere në vektori. Nëse kjo vlerë nuk gjendet në vektor atëherë funksioni do të kthejë -1.

Ushtrim 5- Ndërtoni një funksion rekursiv në php me emrin `isordered` i cili merr si argument një vektor me numra dhe kthen 0 ose 1 në varësi të faktit që vektori është ose jo i renditur në rendin rritës

Ushtrim 6-Ndërtoni një funksion rekursiv `indexof` i cili merr si argumente një vektor me numra të plotë dhe një vlerë. Funksioni kthen vendndodhjen e parë të kësaj vlere në vektori. Nëse kjo vlerë nuk gjendet në vektor atëherë funksioni do të kthejë -1.

Ushtrim 7-Ndërtoni një funksion rekursiv `indexof` i cili merr si argumente një vektor me numra të plotë dhe një vlerë. Funksioni kthen vendndodhjen e fundit të kësaj vlere në vektori. Nëse kjo vlerë nuk gjendet në vektor atëherë funksioni do të kthejë -1.

Ushtrim 8- Ndërtoni një funksion që kthen shumën e elementëve që formojnë konturin e një matrice me dimension $m \times n$, të cilën e merr si argument.

Ushtrim 9-Ndërtoni një funksion që kthen numrin e rreshtit që ka shumën më të madhe të një matrice $m \times n$

13. Puna me skedaret

13.1 Hapja e skedarëve nëpërmjet funksioni fopen()

Që të mund të lexojmë ose shkruajmë në një skedar duhet ta hapim atë për lexim apo shkrim. Për të hapur një skedar php ofron funksionin fopen(). Funksioni fopen() merr dy argumenta: emrin e skedarit që do të hape dhe modën në të cilën do ta hapë këtë skedar.

Më poshtë po listojmë mënyrat si mund të hapet një skedar:

Mënyra

Përshkrimi

r

Vetëm për lexim, nis që nga fillimi i skedarit

r+

Për lexim dhe shkrim. Nis nga fillimi i skedarit

w

vetëm për shkrim, e hap skedarin dhe fshin gjithë përmbajtjen e tij ekzistuese. Nëse skedari nuk ekziston krijon një skedar të ri

w+

Për lexim dhe shkrim

a

Append, shkruan në fund të skedarit. Nëse skedari nuk ekziston krijon një të ri

a+

Lexim dhe shkrim. Shkruan në fund të skedarit

x

Vetëm për shkrim. Krijon një skedar të ri. Kthen FALSE nëse skedari ekziston tashmë.

x+

Për Lexim dhe shkrim. Krijon një skedar të ri. Kthen false nëse skedari ekziston tashmë

Shenim: Nëse funksion fopen nuk mund ta hapë skedarin e kërkuar atëherë ai lthen 0 (false)

Le ta ilustrojmë përdorimin e fopen me një shembull.

```
<html>
<body>
<?php
$file=fopen("welcome.txt","r");
?>
</body>
</html>
```

Shembulli më poshtë merr parasysh edhe rastin kur fopen nuk mund ta hapë skedarin:

```
<html>
<body>
<?php
$file=fopen("welcome.txt","r") or exit("Unable to open file!");
?>
</body>
</html>
```

Pasi kemi përfunduar punë është e nevojshme që ta mbylлим skedarin. Për këtë përdorim funksionin fclose() .

Shembulli më poshtë ilustron përdorimin e këtij funksioni:

```
<?php
$file = fopen("test.txt","r");

//kod që lexon ose shkruan në skedar

fclose($file);
?>
```

Kur lexojmë në një skedar na duhet të kuptojmë nëse kemi mbërritur apo jo në fund të skedarit.

Për këtë përdorim funksionin feof() të php

Për shembull, kodi më poshtë kontrollon nëse kemi mbërritur në fund të skedarit.

```
<?php
if (feof($file)) echo "End of file";
?>
```

Deri në këtë pike kemi parë si të hapim e të mbyllim një skedar si edhe si të kuptojmë nëse kemi mbërritur në fund të skedarit. Më poshtë do të shohim si të lexojmë dhe shkruajmë në një skedar.

13.2 Të lexojmë skedarin rresht për rresht nëpërmjet funksionit fgets()

Funksioni fgets() na lejon të lexojmë një skedar rresht pas rreshti. Kur thërrasim funksioni fgets në një skedar të hapur, funksioni do të kthente rreshtin e parë dhe pas kësaj shënjesi në skedar do të pozicionohej në rreshtin pasardhës, duke qenë gati për të lexuar rreshtin tjetër.

Le të ilustrojmë përdorimin e funksionit gets() nëpërmjet një shembulli, i cili e lexon një skedar rresht pas rreshti dhe e afishon atë derisa të mbërrijë në fund.

Skedari php i paraqitur më poshtë quhet lexo.php

```
<?php
$file = fopen("lajm.txt", "r") or exit("nuk e hap dot skedarin!");
while(!feof($file))
{
    $line=fgets($file);
    echo $line. "<br />";
}
fclose($file);
?>
```

Skedari lexo.php, lexon skedarin lajm.txt rresht për rresht dhe e afishon atë në browser.

18. 3 Të lexojmë skedarin karakter për karakter nëpërmjet funksionit fgetc()

Funksion `fgetc()` lexon një karakter nga skedari i hapur dhe e kthen atë. Pasi thërret funksioni `fgetc`, ky i fundit lexon dhe kthen karakterin që ndodhet në pozicionin ku ndodhet shënjeshti në skedar dhe e sposton shënjeshtin tek karakteri pasardhës.

Le ta ilustrojmë përdorimin e `fgetc`, nëpërmjet një shembulli. Po ndërtojmë një skedar lexo për karakter.php i cili lexon dhe afishon një skedar karakter për karakter.

```
<?php
$file = fopen("lajm.txt", "r") or exit("nuk e hap dot skedarin!");
while(!feof($file))
{
    $char=fgetc($file);
    echo $char;
}
fclose($file);
?>
```

13.4 Të shkruajmë në skedar nëpërmjet funksionit `fwrite()`

Për të shkruan në një skedat të cilin e kemi hapur në një mënyrë që lejon shkrimin(R+,W+,X,X+) ose shtimin në fund të të dhënave në skedar (A,A+).

prototipi i funksionit `fwrite` është : `int fwrite(file,string,length);`

funksioni merr si argument një shënjesht mbi një skedar të hapur, stringun që duam të shkruajmë në skedar, dhe një argument opsional `length` që përcakton numrin maksimal të karakterëve të stringut që duhen shkruar në skedar.

Le ta ilustrojmë nëpërmjet një shembulli përdorimin e funksionit `fwrite`. Do të ndërtojmë një faqe php e cila bën një kopje të skedarit `lajm.txt` në një skedar `lajm1.txt` dhe do të afishojë përmbajtjen e këtij skedari të ri.

më poshtë është kodi i faqes `shkruaj.php`

```
<?php
```

```
$file = fopen("lajm.txt", "r") or exit("nuk e hap dot skedarin!");
$file2 = fopen("lajm2.txt", "w") or exit("nuk e hap dot skedarin!");
while(!feof($file))
{
    $line=fgets($file);
    fwrite($file2,$line);
}
fclose($file);
fclose($file2);
$file2 = fopen("lajm2.txt", "r") or exit("nuk e hap dot skedarin!");
while(!feof($file2))
{
    $line=fgets($file2);
    echo $line.'<br>';
}
?>
```

Ngarkimi i skedarëve dhe funksionet mbi datën dhe kohën e një aplikacion web mund të jetë e nevojshme ti lejojmë përdoruesit të ngarkojë skedare në serverin web.

Do të bazohemi në një shembull që realizon ngarkimin e imazheve por shembulli është analog.

Së pari do të ndërtojmë një formë për upload-in.

```
<html>
<body>
<form action="upload_file.php" method="post"
enctype="multipart/form-data">
Filename:
<input type="file" name="file" id="file" />
<br />
<input type="submit" name="submit" value="Submit" />
</form>
</body>
</html>
```

Vini re formën HTML më sipër:

Atributi enctype i tagut <form> përcakton çfarë tip të dhënash do të ketë forma: "multipart/form-data". enctype duhet të ketë këtë vlerë nëse forma përmban të dhëna binare si skedare etj.

Për të përzgjedhur skedarin që duam të ngarkojmë është ndërtuar një input i tipit file:

```
<input type="file" name="file" id="file" />
```

Ky input do të shfaqë një buton browse që i mundëson përdoruesit të përzgjedhë skedarin që do të ngarkojë.

Kjo formë i dërgon të dhënat tek skedari upload_file.php, i cili kap të dhënat dhe ruan skedarin e ngarkuar në dosjen e dëshiruar në web server.

Le të shohim se si kapen të dhënat në server në faqen upload_file.php

Kodi i faqes upload_file.php

```
<?php
if ($_FILES["file"]["error"] > 0)
{
    echo "Error: " . $_FILES["file"]["error"] . "<br />";
}
else
{
    echo "Upload: " . $_FILES["file"]["name"] . "<br />";
    echo "Type: " . $_FILES["file"]["type"] . "<br />";
    echo "Size: " . ($_FILES["file"]["size"] / 1024) . " Kb<br />";
    echo "Stored in: " . $_FILES["file"]["tmp_name"];
    $file_dir = "images/";
    $newfile = $file_dir.$_FILES['file']['name'];
    move_uploaded_file($_FILES['file']['tmp_name'], $newfile);
}
?>
```

Skedari upload_file.php afishon emrin e skedarit të sapo ngarkuar, tipin e skedarit, madhësinë në KB, dhe vëndndodhjen e përkoheshme ku e ka ngarkuar web serveri imazhin.

Së fundi nëpërmjet funksionit move_uploaded_file(\$_FILES['file']['tmp_name'], \$newfile); , zhvëndoset skedari i ngarkuar nga dosja e përkoheshme

ku e ka ruajtur web serveri në adresën e re që në këtë shembull është në images, brenda rrënjës të sitet ku ndodhet faqja upload_file.php. Në vazhdim të këtij libri pasi të shikojmë mënyrën e ruajtjes së të dhënave në bazën e të dhënave do ti rikthehemi këtij shembulli duke ruajtur të dhënat e imazhit në një tabelë të bazës së të dhënave.

13.5 Funksionet mbi datën dhe kohën

PHP ka disa funksionet për datën dhe kohën, që i mundësojnë programuesit të punojë me datat dhe kohën.

Funksioni date()

Funksioni datë formaton një kohë sipas një formati të caktuar. Funksioni datë merr si argument formatin dhe kohën që duhet të formatojë. Nëse nuk e vëndosim argumentin e dytë atëherë funksioni datë formaton kohën e momentit në web server

Prototipi i përgjithshëm i funksionit është: date(format,timestamp)

Për formatimin funksioni datë përdor karakterët e formatimit, disa prej të cilave janë listuar më poshtë:

d - Përfaqëson një ditë të muajit (01 deri 31)

m -Përfaqëson një muaj (01 deri 12)

Y - Përfaqëson një vit i shprehur në katër shifra

Midit karakterëve të formatimit mund të vendosen edhe karakterë ndarëse si"/", ".", "-". Shembujt më poshtë ilustrojnë përdorimin e funksionit datë.

```
<?php  
echo date("Y/m/d") . "<br />";  
echo date("Y.m.d") . "<br />";  
echo date("Y-m-d")  
?>
```

Të gjenerojmë një kohë me funksionin mktime

Funksioni mktime krijon një kohë e cila mund të formatohet me funksionin datë. Shembujt më poshtë ilustrojnë përdorimin e mktime.

```
<?php
$tomorrow = mktime(0,0,0,date("m"),date("d")+1,date("Y"));
echo "Tomorrow is ".date("Y/m/d", $tomorrow);
?>
```


Ushtrime

Ushtrim 1 - Ndërtoni një faqe php e cila krijon një skedar me emrin vargu.txt. Në këtë skedar vendos kufizat e vargut $a[n]=2*n+1$, duke vendosur një vlerë në rresht të ri.

Programi më pas lexon elementet nga skedari varg.txt kufizat e vargut dhe gjen shumën e tyre.

Ushtrim 2 - Ndërtoni një faqe php e cila ruan në një skedar të dhënat mbi lëndët , studentët dhe notat e tyre. Më pas programi i lexon këto të dhëna dhe i afishon në një tabelë në html. Për këtë ushtrim bazohuni në shembullin e dhënë në leksionin mbi array në php

Ushtrim 3 - Ndërtoni një version të faqes upload_file.php e cila pasi e ruan skedarin e ngarkuar në dosjen images si në shembullin në leksion. Afishon një logo të aplikacionit me të cilin është krijuar skedari. Logoja duhet të jetë e klikueshme në mënyrë të tillë që nëse përdoruesi klikon mbi të të hapet skesari i ngarkuar.

Ushtrim 4. Ndërtoni një faqe php e cila nëpërmjet një cikli afishon datat një e nga një për periudhën 1 mars 2011 deri 31 mars 2011. Çdo datë do të afishohet në një rresht të ri .

14. Cookie-t dhe Sesionet

Protokolli HTTP është një protokoll pa gjendje. Aplikacionet e web bazohen në kërkesa dhe përgjigje; browseri bën një kërkesë për një faqe php, web serveri e ekzekuton atë dhe rezultatin e ekzekutimit ia kthen browserit. Çdo faqe php është një program (skript) më vehte që fillon ekzekutimin në momentin që thërritet në rreshtin e parë të kodit dhe përfundon ekzekutimin në rreshtin e fundit të kodit. Me përfundimin e ekzekutimit variablat e krijuar në këtë faqe shkatërrohen.

Aplikacioni web është një bashkësi faqesh php të cilat kryejnë funksione të ndryshme dhe janë të lidhura me njëra-tjetra për të mundësuar llogjikën e të gjithë aplikimit. Ku i ruajnë faqet informacionet që duhet ti gjejnë apo ti modifikojnë faqet e tjera? Thënë ndryshe si ruhet gjendja e aplikimit?

14.1 Ruajtja e gjendjes së aplikimit në sesion

Le të marrim si shembull një aplikacion i cili ka disa faqe të mbrojtura me fjalëkalim.

Aplikacioni ynë do të kishte një faqe login.php e cila ia sherben për të afishuar një formë ku përdoruesi të vendose kodin dhe fjalëkalimin. Më pas faqja login.php do t'ia dërgonte të dhënat për verifikim faqes checklogin.php në momentin që përdoruesi klikon butonin login.

Faqja checklogin.php verifikon kredencialet e përdoruesit duke lexuar në bazën e të dhënave dhe nëse janë të sakta e lejon përdoruesin të hyje në faqet e mbrojtura të aplikimit.

Aplikacioni ynë mund të ketë një faqe addnews.php e cila shton një lajm të ri në bazën e të dhënave e cila duhet paraprakisht të verifikojë që përdoruesi që po tenton ta hape këtë faqe është një përdoruesi i loguar.

Ku do ta gjejë këtë informacion?

Duke qenë se faqja checklogin.php ka përfunduar ekzekutimin e saj, do të thotë që të gjithë variablat e kësaj faqe ku mund të jetë ruajtur një informacion mbi rezultatin e identifikimit të përdoruesit janë shkatërruar.

Me sa duket faqja checklogin.php nuk e ka ruajtur informacionin mbi rezultatin e identifikimit në variablat në faqen e saj por në një

vënd tjetër, i cili nuk shkatërrohet me përfundimin e ekzekutimit të faqes.

Ku ruhet ky informacion?

Për skenare të tilla ku na duhet të ruajmë variabla që kanë të bëjnë me gjendjen e aplikimit përdoret sesioni dhe cookiet.

veçanërisht për skenarin e mësipërm si edhe për skenare ku informacionet që ruhen duhet të jene te rezervuara, përdoret sesionet pasi sesionet ruhen në server ndërsa cookiet dërgohen dhe ruhen në browser në formën e skedarëve të vegjël të tekstit. Duke qenë se cookit ruhen në browser ato janë me pak të sigurt si edhe mund të fshihen nga përdoruesi.

Së fundi browseri mund të jetë i konfiguruar që të mos lejojë cookiet, dhe në këtë rast përdoruesit nuk do të ishin në gjendje të logoheshin në aplikim.

Faqja login.php dërgon kredencialet e përdoruesit tek faqja checklogin.php. Faqja checklogin.php verifikon këto kredenciale në bazën e të dhënave dhe nëse janë të sakta ruan një variabël në sesion (skematikisht është paraqitur më një rreth) me emrin loguar dhe si vlerë të këtij variable vendos username e personit , kredencialet e të cilit sapo u verifikuan me sukses.

Pas kësaj faqja checklogin.php ridrejton përdoruesin tek faqja hyrëse në aplikim info.php, nga e cila përdoruesi mund të shkojë tek shtuartikull.php për të hedhur të dhëna në bazën e të dhënave. Pas kësaj faqja checklogin.php përfundon ekzekutimin e saj, ndërsa variabli i krijuar në sesion **loguar**, mbetet dhe faqet info.php shtuartikull.php si edhe faqet e tjera mund të kontrollojnë nëse ky variabël ekziston apo jo. Nëse variabli ekziston në sesion do të thotë që faqja checklogin e ka krijuar atë pasi ka verifikuar kredenciale e përdoruesit. Nëse variabli nuk ekziston, atëherë do të thotë që përdoruesi nuk është loguar.

Kodi:

```
<?php
if (!isset($_SESSION['loguar']))
{
header("location=login.php");
exit();
}
```

?>

Kontrollon për ekzistencën e variablit loguar në sesion. Nëse ky variabël nuk ekziston (!isset) atëherë browseri ridrejtohet tek faqja login.php (header("location=login.php");) dhe dilet nga ekzekutimi i skriptit/faques php (exit();). Nëse këtë kod e vendosim në fillim të faqeve që nuk duhet të hapen pa u loguar përdoruesi, atëherë përfundimi i ekzekutimit tek instruksion exit do të garantonte që përdoruesi nuk e hap dot faqen e kërkuar pa u loguar. Në skicën më sipër, kodi php që kontrollon për ekzistencën e variablit të loguar në sesion është vendosur në një faqe të vecantë php, me emrin kontrollo.php. Kjo faqe mund të përfshihet në fillim të të gjitha faqeve php, që duhet të jenë të mbrojtura me fjalëkalim nëpërmjet require("kontrollo.php");

Siç duket edhe nga shembulli më lart për të ruajtur një variabël në sesion mund të përdorim sintaksen:

```
$_SESSION['loguar']='admin';
```

Shënim: Mund të përdorim gjithashtu funksionin session_register(var).

Ndërsa për të kontrolluar nëse një variable ekziston në sesion dhe për të kapur vlerën e tij, mund të përdorim sintaksën e ilutruar në shembullin më poshtë.

```
session_start();
if (isset($_SESSION['loguar']))
{
    $u=$_SESSION['loguar'];
    echo $u;
}
```

14.2 Startimi i sesionit

Që të ruajmë të dhëna në sesion është e nevojshme që të startojmë paraprakisht sesionin. Sesioni mund të jetë startuar automatikisht nga web serveri ose duhet që të startohet nga programuesi. PHP ka një

funkcion `session_start()`; i cili starton sesionin. Nëse web serveri nuk është konfiguruar që ti startojë automatikisht sesionet atëherë programuesi duhet të thërrasë së pari funksionin `session_start()`, përpara se të përdorë sesionin, në sejcilën nga faqet.

Do të ishte e rekomandueshme që të ndërtohej një faqe me emrin `start.php` e cila të kishte vetëm një instruksion `session_start()`;

Faqja `start.php` do të përfshihej më pas (`include("start.php");`) nga faqet e aplikacionit tonë.

Kur aplikimi ndodhet në një server që i starton sesionet në mënyrë automatike atëherë ky rresht bëhet koment

faqja `start.php` në një server që i starton automatikisht sesionet:

```
<?php
//session_start();
?>
```

Kështu kontrollojmë lehtësisht nga një vënd i vetëm startimin e sesionit për aplikimin tonë.

14.3 Cookie-t

Çfarë është një cookie?

Një cookie është një skedar i vogël i cili dërgohet nga serveri në kompjuterin e përdoruesit . Sa herë që browseri bën një kërkesë në aplikacionin tonë ai do të dërgojë bashkë me kërkesën edhe cookie

Si ti krijojmë dhe si ti lexojmë cookiet në php? .

Cookiet krijohen nëpërmjet funksionit php `setcookie()`

Kujdes! Funksioni `setcookie()` modifikon headerat e përgjigjes që dërgohet në browser, prandaj ai duhet të thërritet përpara tagut `<html>`, dhe nuk duhet të ketë asnjë komandë `echo` që gjeneron output për browserin përpara se të thërritet funksioni `setcookie`. Përndryshe veprimi do të dështonte me një mesazh gabimi të ngjashëm me : *Headers cannot be modified. Output already started!*

Sintaksa e funksionit `set cookie` është:

```
setcookie(name, value, expire, path, domain);
```

name= emri i cookiet

value= vlera e cookiet

exipre = per sa kohe skadon

path=adresa ne server

domain= domain i aplikimit tone

Le të shpjegojmë përdorimin e funksionit setcookie nëpërmjet disa shembujve:

Shembull 1

Në këtë shembull do të krijojmë një cookie me emërin "user" dhe do ti caktojmë një vlerë "John Smith". Do të përcaktojmë që ky cookie do të skadojë për një orë:

```
<?php  
setcookie("user", "John Smith", time()+3600);  
?>
```

```
<html>
```

.....

Shembull 2

Në shembullin më poshtë do të ilustrojmë një mënyrën alternative për përcaktimin e periudhës së skadimit të cookiet:

```
<?php  
$expire=time()+60*60*24*30; // skadon pas një muaji  
setcookie("user", "John Smith", $expire);  
?>
```

```
<html>
```

.....

Në shembullin më sipër përcaktuam si kohë skadimi kohën aktuale $\text{time() + (60 sec * 60 min * 24 hours * 30 days)}$.

Pra koha aktuale plus një muaj i shprehur në sekonda.

Si te kapim vleren e nje Cookie?

Vlerat e cookie që dërgohen nga browseri me një kërkesë HTTP , në server vendosen në një koleksion në mënyrë të ngjashme me \$_FILES apo \$_REQUEST, në koleksionin me emrin \$_COOKIE.

Në shembullin më poshtë do të ilustrojmë kapjen e vlerës së një cookie nëpërmjet koleksionit \$_COOKIE. Do të lexojmë vlerën e cookiet user që krijuam më lart:

```
<?php
$u=$_COOKIE["user"];
echo $u;
?>
```

Në shembullin më poshtë do të përdorim funksionin isset() për të verifikuar nëse ekziston cookie përpara se të lexojmë vlerën e tij:Pas

```
<html>
<body>
<?php
if (isset($_COOKIE["user"]))
    echo "Welcome " . $_COOKIE["user"] . "!"<br />;
else
    echo "Welcome guest!"<br />;
?>
</body>
</html>
```

Si të fshijmë një Cookie?

Për të fshirë një cookie, mjafton të caktojmë kohën e skadimit më të vogël se momenti aktual.
për shembull

```
<?php
// vendosim kohën e skadimit një orë më parë se tani
setcookie("user", "", time()-3600);
?>
```

14.4 Dërgimi i email

PHP na mundëson ti dërgojmë një email direkt nga aplikacioni. Kjo mund të jetë e nevojshme në shumë raste si për shembull kur prodhuesi harron fjalëkalimin, kur përdoruesi bën një porosi dhe duam ti dërgojmë një informacion mbi porosinë me email etj.

Për këtë mund të përdorim funksionin mail.

Sintaksa e funksionit mail është:

mail(to,subject,message,headers,parameters)

to Është një parametër i detyrueshëm dhe përcakton marrësit ose marrësit e email.

subject Është një parametër i detyrueshëm. Përcakton subjektin e email.

message Është një parametër i detyrueshëm. Është trupi i mesazhit që do të dërgohet. Në fund të çdo rreshti do të vendoset një LF (\n). Rreshtat nuk mund të jenë më të gjatë se 70 karaktere.

headers Parameter opsional. Në këtë parametër vendosen header-at shtesë, si From, Cc, dhe Bcc. Header-at shtesë duhet të ndahen me një CRLF (\r\n)

parameters Parametër opsional.

Shenim: Që të funksionojë funksioni mail, është e nevojshme që serveri të këtë një shërbim email të konfiguruar paraprakisht. Në ambjentet linux php përdor programin sendmail për të dërguar email.

Shembull 1 - Të dërgojmë një email nëpërmjet php

Në shembullin më poshtë së pari i japim vlerë variablave (\$to, \$subject, \$message, \$from, \$headers), dhe më pas thërrasim funksionin mail duke i kaluar këto variabla si argumenta:

```
<?php
$to = "someone@example.com";
$subject = "Test mail";
$message = "Hello! This is a simple email message.";
$from = "someoneelse@example.com";
```



```
$headers = "From: $from";  
mail($to,$subject,$message,$headers);  
echo "Mail Sent."  
?>
```

Shembull 2- Të ndërtojmë një formë email

Më poshtë do të ndërtojmë një faqe që di ti mundësojë përdoruesve të na japin përshtypjet e tyre:

```
<html>  
<body>  
<?php  
if (isset($_REQUEST['email']))  
// if "email" is filled out, send email  
{  
// send email  
$email = $_REQUEST['email'];  
$subject = $_REQUEST['subject'];  
$message = $_REQUEST['message'];  
mail("info@company.com", "Subject: $subject", $message, "From:  
$email");  
echo "Thank you for using our mail form";  
}  
else  
// if "email" is not filled out, display the form  
{  
echo "<form method='post' action='mailform.php'>  
Email: <input name='email' type='text' /><br />  
Subject: <input name='subject' type='text' /><br />  
Message:<br />  
<textarea name='message' rows='15' cols='40'>  
</textarea><br />  
<input type='submit' />  
</form>";  
}  
?>  
</body>  
</html>
```

Siç duket edhe nga shembulli më lart faqja kontrollon së pari nëse është shtypur është dërguar forma, nëse kjo është e vertetë atëherë përgatit dhe dërgon email, duke lexuar nga kutitë që plotësoi përdoruesi adresën e tij të email dhe mesazhin.

Përndryshe nëse forma nuk është dërguar atëherë faqja afishon një formë me disa kutia tek të cilat mund të vendosen të dhënat e përdoruesit dhe të dërgohen ato në server.

Ushtrime

Ushtrim 1

Ndërtoni një faqe preference.php që afishon një kuti me emrin preferencë:

Pasi përdoruesi përcakton një preference dhe shtyp butonin RUAJ faqja preference i dërgon të dhënat tek faqja ruaj preference.php. Faqja ruaj preference ruan preferencën e përdoruesit në një cookie me emrin preferencë dhe e ridrejton browserin tek faqja preference.php. Pas kësaj faqja preference.php nuk afishon më formën për përcaktimin e preferencës por një tekst që i thotë përdoruesit se cila ishte preferenca që ai përcaktoi.

Ushtrim 2

Në seksionin **komunikimi me browserin**, është dhënë një shembull për ndërtimin e një faqe php që do të shërbente si një buxhet online.

Të modifikohet faqja buxheti.php duke i shtuar një buton të tipit submit dhe në mënyrë të tillë që ti dërgojë të dhënat në një faqe dërgo buxhet.php

Të ndërtohet faqja dërgobuxhet.php e cila do të kapë vlerat e elementëve të buxhetit dhe do ti dërgojë ato me email tek adresa e email Tuaj.

15. Lidhja e PHP me MySQL

PHP ka “built in database operations” me disa baza të dhënash si për MySQL dhe DBMS të tjerë janë të ngjashëm.

Lidhja me DB

```
mysql_connect("host", "login", "password"))
or die("Could not connect.");
mysql_select_db("carbase") or
die (mysql_error());
```

Disa veprime baze ne bazen e te dhenave

Shembull 1 - check login

```
<?php
$u=$_REQUEST['username']; // une
$p=$_REQUEST['password']; // sot
$q="select * from përdoruesit where username=' ".$u."' and
password=' ".$p."' ";
/* Vlera e stringut - sql dinamike
select * from përdoruesit where username = 'une' and password='sot'
*/
```

```
$res=mysql_query($q)
//n mysql_query ekzekuton komanden ne DB
or die("nuk ecen");
if ($rec=mysql_fetch_array($res))
{
echo "welcome ".$u;
}
else
{
echo "username password nuk jane te sakte";
echo "<a href='login.php'>kliko per tu loguar perseri</a>";
}

?>
```

Shembull 2 - Afisho listën e studentëve

```
<?php
```

```
$q=" select * from studente where  
qyteti=' ".$_REQUEST("qyteti")." ' order by emri,mbiemri ";  
$res=mysql_query($q)  
or die("nuk ecen");  
while $rec=mysql_fetch_array($res)  
{  
echo $rec["emri"]." ".$rec["mbiemri"];  
echo "<br>";  
}  
?>
```

15.1 Hyrje në bazat e të dhënave

Çfarë është një sistem i menaxhimit të të dhënave (DBMS)

Një sistem i menaxhimit të të dhënave përbëhet nga një bashkësi komponentes:

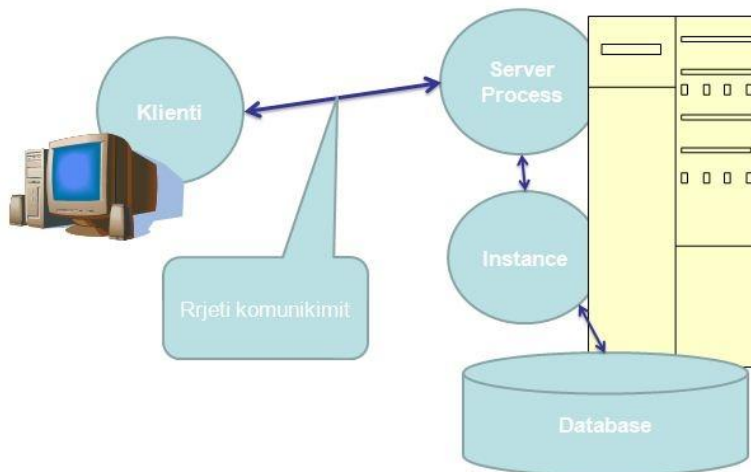
- c programet aplikative mbi bazat e të dhënave
- c Komponentet klient
- c Serverat e Database
- c Database-t

Një program aplikativ mbi bazat e të dhënave është për një qëllim të caktuar.

Ndërsa një komponente klient është një software që përdoret për veprime të përditëshme mbi bazën e të dhënave, për administrimin , për raportime etj.

Nëpërmjet një komponenti klient , përdoruesit mund të përdorin një bazë të dhënash në të njëjtin kompjuter apo në një kompjuter(server) me të cilin komunikojnë nëpërmjet një rrjeti.

Funksioni i një serveri të bazës së të dhënave është të administrojë të dhënat. Çdo klient mund ti përdore të dhënat vetëm duke i dërguar serverit një kërkesë (query) dhe serveri pas disa kontrollëve nëse klienti është i autorizuar të kryejë veprimin e kërkuar mbi të dhënat e kryen veprimin dhe i kthen një përgjigje me rezultatin klientit.



Nje DBMS ofron funksionet e mëposhtme:

- Ndërfaqe për përdoruesit, që të kenë mundësi të bëjnë kërkesa për shërbime mbi të dhënat
- Pavarësi të të dhënave fizike (Physical data independence)
- Pavarësi llogjike të të dhënave (Logical data independence)
- Optimizim të kërkesave (Query optimization)
- Integritin e të dhënave (Data integrity)
- Kontroll të konkurencës (Concurrency control)
- Backup dhe recovery
- Siguri të bazës së të dhënave (Database security)

Le ti shohim me konkretisht këto funksione

Ndërfaqe për përdoruesit

Shumë baza të dhënash janë ndërtuar për tu përdorur nga përdorues të kategorive të ndryshme me nivel të ndryshëm njohurish.

Ndaj DBMS duhet të ofrojë disa ndërfaqe për përdoruesit me menu, forma, raporte si edhe një gjuhë programimi(SQL) për përdoruesit me të përparuar.

Pavarësia fizike e të dhënave

Është e nevojshme që programet që përdorin të dhënat në një bazë të dhënash të mos jenë të varur nga struktura fizike në të cilën janë ruajtur të dhënat. Kjo gjë na mundëson të modifikojmë të dhënat pa ndikuar në programin aplikativ. Për shembull duhet të kemi

mundësi të zhvendosim të dhënat në një server të ri më të shpejtë pa modifikuar aplikacionin.

Ose, nëse një tabelë është shumë e madhe dhe po ndikon në shpejtësinë e sistemit, mund të vendosim ta ndajme atë në dy skedare ku secili prej tyre të ruhet në një disk me vehte për të rritur shpejtësinë e sistemit.

Sigurisht, nuk duam të modifikojmë aplikacionin për këtë gjë.

Pavarësia llogjike e të dhënave

Po e ilustron këtë me një shembull.

Nëse duam të shtojmë një kollonë me adresë në një tabelë Employees, mund ta bëjmë këtë pa modifikuar aplikacionin,

Nëse të dhënat mbi punonjësit do të ruheshin në një skedar atëherë shtimi i një fushe në skedar do të kërkonte modifikimin e aplikacionit mbi të dhënat

Optimizimi i kërkesave

Të gjithë DBMS kanë një komponent që quhet optimizues, i cili merr në konsideratë disa mënyra të ndryshme për të ekzekutuar komandat në bazën e të dhënave. Mënyra e zgjedhur quhet plan ekzekutimi. Optimizuesi merr vendimet duke u bazuar në informacione si madhësia e tabelave ku po ekzekutohet komanda, çfarë indeksesh ka, dhe çfarë operatorësh llogjikë (AND, OR, NOT) përdoren në klauzolen WHERE

Integriteti i të dhënave

Një nga funksionet e DBMS është të identifikojë të dhëna që janë llogjikisht jo të sakta dhe të mos i pranojë të ruaje ato në bazën e të dhënave. Për shembull nëse ruajmë moshën e një punonjësi DBMS nuk duhet të pranojë një moshë jashtë intervalit (18-70).

Kontrolli i konkurrencës

Pjesa më e madhe e DBMS të sotëm, janë sisteme shumë-përdorues dhe i mundësojnë disa përdoruesve në të njëjtën kohë (në mënyrë konkurente) të punojnë me të dhënat.

Puna në mënyrë të njëkoheshme me të dhënat nga disa përdorues, paraqet disa probleme të cilat duhet ti shmangë (administroje) DBMS. Për ilustrim po marrim një shembull tipik:

1. Një llogarie bankare numër 4711 në bankën X ka një balancë prej \$2,000.

2. Dy pronarët e kësaj llogarie, Znj. A dhe Z. B, shkojnë në dy sportele të ndryshme dhe secili prej tyre tërheq 2000 dollarë

DBMS nuk duhet ta lejojë këtë situatë. Llogaria duhet të "kycet" në momentin që telleri i parë fillon të kryejë një veprim tërheqje nga kjo llogari dhe telleri i dytë duhet të vendoset në pritje derisa të mbarojë punë telleri i parë.

Telleri i dytë fillon të punojë me llogarinë pasi ka mbaruar telleri i parë, duke gjetur një balancë të llogarisë 0 dollarë, gjë që nuk i mundëson Z. B të tërheqë 2000 dollarët e tij pasi ato më parë i ka tërhequr Znj. A.

Backup dhe Recovery

Një DBMS duhet të ketë mjete për të ruajtur të dhënat në mënyrë të tillë që ti rigjenerojë ato në rast të një dëmtimi të pajisjeve ose të të dhënave,

Siguria e bazës së të dhënave

Siguria e bazës së të dhënave lidhet me dy procese: autentikimin dhe autorizimin.

Autentikimi është procesi verifikimit të kredencialëve të përdoruesit për të mos lejuar përdorues të paautorizuar të përdorin DBMS. Autentikimi zakonisht realizohet nëpërmjet një emri përdoruesi dhe fjalëkalimi. Ky informacion kontrollohet nga DBMS dhe nëse është i saktë përdoruesi lejohet të përdorë sistemin. Për të shmangur vjedhjen e fjalëkalimit gjatë transmetimit në rrjet apo gjatë ruajtjes së tij përdoret enkriptimi, ndërsa

Autorizimi është proces që ndodh pas autentikimit. Tani me DBMS e di identitetin e përdoruesit (nëpërmjet autentikimit) dhe sa herë që një përdorues tenton të përdorë një objekt të të dhënave DBMS verifikon nëse përdoruesi është lejuar të kryejë këtë veprim. Nëse po e autorizon përdoruesin të kryejë veprimin e kërkuar.

15.2 Relational Database Management Systems (RDBMS)

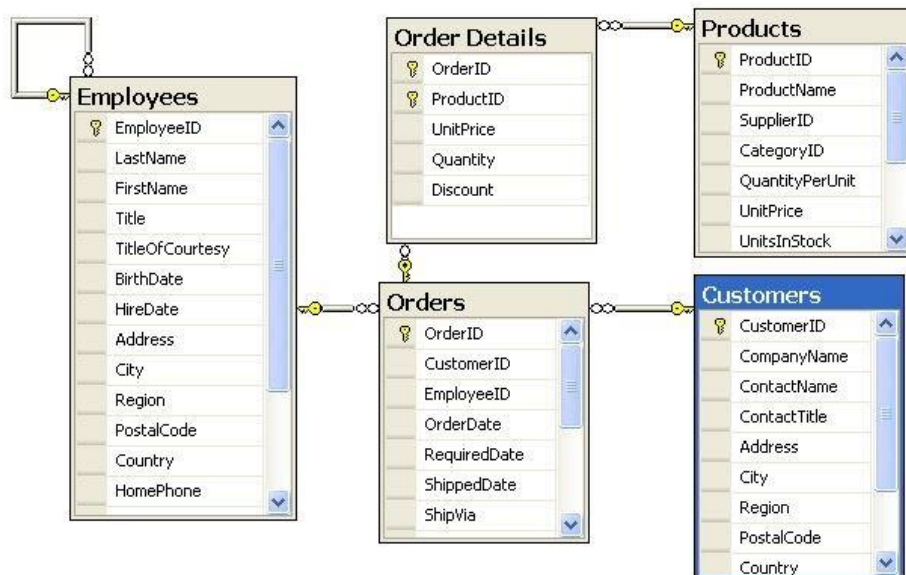
Sistemet relacionale të menaxhimit të të dhënave

Koncepti i bazave të dhënave relacionale është prezantuar për herë të parë nga E. F. Codd në artikullin e tij “A Relational Model of Data for Large Shared Data Banks” në vitin 1970. Ne dallim nga sistemet e mëparshme (rrjetet dhe hierarkik), sistemet relacionale bazohen në modelin e të dhënave relacional që bazohet në algjebrën relacionale.

Koncepti bazë i modelit relacional është relacioni (= tabela), prandaj në një sistem relacional tabela është e vetmja strukture për ruajtjen e të dhënave. Tek një kollinë e një rreshti ruhet me saktësi një vlerë skalare.

Për të ilustruar bazat e të dhënave relacionale do të bazohemi tek baza e të dhënave Northwind.

Po paraqesim më poshtë disa nga tabelat e kësaj baze të dhënash.



Siç duket baza e të dhënave Northwind ruan informacion mbi shitjet e një kompanie

Tabela Orders

OrderID	CustomerID	Employ...	OrderDate	RequiredDate	ShippedDate	ShipVia	Freight	ShipName
10248	VINET	5	1996-07-04 0...	1996-08-01 ...	1996-07-16 0...	3	32.38	Vins et alcools Chevalier
10250	HANAR	4	1996-07-08 0...	1996-08-05 ...	1996-07-12 0...	2	65.83	Hanari Carnes
10251	VICTE	3	1996-07-08 0...	1996-08-05 ...	1996-07-15 0...	1	41.34	Victualles en stock
10252	SUPRD	4	1996-07-09 0...	1996-08-06 ...	1996-07-11 0...	2	51.30	Suprêmes délices
10253	HANAR	3	1996-07-10 0...	1996-07-24 ...	1996-07-16 0...	2	58.17	Hanari Carnes
10254	CHOPS	5	1996-07-11 0...	1996-08-08 ...	1996-07-23 0...	2	22.98	Chop-suey Chinese
10255	RICSU	9	1996-07-12 0...	1996-08-09 ...	1996-07-15 0...	3	148.33	Richter Supermarkt
10256	WELLI	3	1996-07-15 0...	1996-08-12 ...	1996-07-17 0...	2	13.97	Wellington Importadora
10257	HILAA	4	1996-07-16 0...	1996-08-13 ...	1996-07-22 0...	3	81.91	HILARION-Abastos
10258	ERNSH	1	1996-07-17 0...	1996-08-14 ...	1996-07-23 0...	1	140.51	Ernst Handel
10259	CENTC	4	1996-07-18 0...	1996-08-15 ...	1996-07-25 0...	3	3.25	Centro comercial Moctezuma

Tabela Orders ruan informacion mbi porositë e kryera në kompani. Për çdo porosi ruhen disa informacione ndër të cilat disa janë informacione të detyrueshme e disa opsionale.

Kështu për një porosi ruhet :

- Kodi në kollonën me emrin OrderId,
- Kodi i klientit në kollonën me emrin CustomerID,

Kodi i punonjësit që realizoi këtë porosi në kollonën me emrin EmployeeId,

Data e porosisë në kollonën me emrin OrderDate

si edhe të tjera informacione si

- datën kur kërkohet që porosia të lëvrohet tek klienti,
- në cilën datë duhet të nisë transporti(Shippeddate),
- kodi i kompanisë së transportit(ShipVia) etj.

Tabela Employees

EmployeeID	LastName	FirstName	Title	TitleOfC...	BirthDate	HireDate	Address
1	Davolio	Nancy	Sales Representative	Mrs.	1948-12-08 ...	1992-05-01 0...	507 - 20th Ave. E, Apt. 2A
2	Fuller	Andrew	Vice President, Sales	Mrs.	1952-02-19 ...	1992-08-14 0...	908 W. Capital Way
3	Leverling	Janet	Sales Representative	Mrs.	1963-08-30 ...	1992-04-01 0...	722 Moss Bay Blvd.
4	Peacock	Margaret	Sales Representative	Mrs.	1937-09-19 ...	1993-05-03 0...	4110 Old Redmond Rd.
5	Buchanan	Steven	Sales Manager	Mrs.	1955-03-04 ...	1993-10-17 0...	14 Garrett Hill
6	Suyama	Michael	Sales Representative	Mrs.	1963-07-02 ...	1993-10-17 0...	Coventry House Miner Rd.
7	King	Robert	Sales Representative	Mrs.	1960-05-29 ...	1994-01-02 0...	Edgeham Hollow Winchester Way
8	Callahan	Laura	Inside Sales Coordi...	Mrs.	1958-01-09 ...	1994-03-05 0...	4726 - 11th Ave. N.E.
9	Dodsw...	Anne	Sales Representative	Mrs.	1966-01-27 ...	1994-11-15 0...	7 Houndstooth Rd.

Tabela Employees ruan informacion mbi punonjësit e kompanisë. Për cdo punonjës ruhen disa informacione ndër të cilat disa janë informacione të detyrueshme e disa opsionale.

Kështu për një punonjës ruhet :

- Kodi në kollonën me emrin EmployeeId,
- Mbiemri i punonjësit në kollonën me emrin Lastname,

- Emri i punonjësit në kollonën me emrin Firstname,
 - Pozicionin e punës së punonjësit në kollonën me emrin Title
 - Datëlindjen në kollonën me emrin Birthdate
 - Datën e marrjes në punë në kollonën me emrin Hiredate
- si edhe informacione të tjera mbi punonjësit.

Tabela Customers

CustomerId	CompanyName	ContactName	ContactTitle	Address	City
WOLZA	Wolski Zajazd	Zbyszek Piestrzeniewicz	Owner	ul. Filtrowa 68	Warszawa
OCEAN	Océano Atlántico Ltda.	Yvonne Moncada	Sales Agent	Ing. Gustavo Moncada 858...	Buenos Aires
LAUGB	Laughing Bacchus Wine Cel...	Yoshi Tannamuri	Marketing Assistant	1900 Oak St.	Vancouver
HUNGC	Hungry Coyote Import Store	Yoshi Latimer	Sales Representative	City Center Plaza 516 Main St.	Elgin
CHOPS	Chop-suey Chinese	Yang Wang	Owner	Hauptstr. 29	Bern
BSBEV	B's Beverages	Victoria Ashworth	Sales Representative	Fauntleroy Circus	London
AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	London
DRACD	Drachenblut Delikatessen	Sven Ottlieb	Order Administrator	Walsenweg 21	Aachen
NORTS	North/South	Simon Crowther	Sales Associate	South House 300 Queensbri...	London
RANCH	Rancho grande	Sergio Gutierrez	Sales Representative	Av. del Libertador 900	Buenos Aires
ERNSH	Ernst Handel	Roland Mendel	Sales Manager	Kirchgasse 6	Graz
WANDK	Die Wandernde Kuh	Rita Müller	Sales Representative	Adenauerallee 900	Stuttgart

Tabela Customers ruan informacion mbi klientët. Për çdo klient ruhen disa informacione ndër të cilat disa janë informacione të detyrueshme e disa opsionale.

Kështu për një klient ruhet :

- Kodi në kollonën me emrin CustomerId,
- Emri i kompanisë në kollonën me emrin CompanyName,
- Personin e kontaktit në kollonën me emrin ContactName,
- Pozicionin e punës së personit të kontaktirne kollonën me emrin ContactTitle
- Adresën
- Qytetin

si edhe informacione të tjera.

Tabela OrderDetails

OrderID	ProductID	UnitPrice	Quantity	Discount
10250	41	7.70	10	0
10250	51	42.40	35	0.15
10250	65	16.80	15	0.15
10251	22	16.80	6	0.05
10251	57	15.60	15	0.05
10251	65	16.80	20	0
10252	20	64.80	40	0.05
10252	33	2.00	25	0.05
10252	60	27.20	40	0

Tabela OrderDetails ruan informacion mbi detajet e porosive. Për çdo detaj (rresht të porosisë, kujtoni një faturë që keni parë së fundmi) ruhen informacionet e mëposhtme:

- Kodi i porosisë në kollonën me emrin OrderId,
- Kodi i produktit në kollonën me emrin ProductId,
- Cmimi i produktit në kollonën me emrin UnitPrice,
- Sasine e porositur nga ky produkt në kollonën me emrin Quantity
- Ndonjë ulje të mundshme në kollonën me emrin Discount

Tabela Products

ProductID	ProductName	SupplierID	CategoryID	QuantityPerUnit	UnitPrice	UnitsInSto...	UnitsOnOrder
1	Chai	1	1	10 boxes x 20 bags	18.00	39	0
2	Chang	1	1	24 - 12 oz bottles	19.00	17	40
3	Aniseed Syrup	1	2	12 - 550 ml bottles	10.00	13	70
4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	22.00	53	0
5	Chef Anton's Gumbo Mix	2	2	36 boxes	21.35	0	0
6	Grandma's Boysenberry Spread	3	2	12 - 8 oz jars	25.00	120	0
7	Uncle Bob's Organic Dried Pears	3	7	12 - 1 lb pkgs.	30.00	15	0
8	Northwoods Cranberry Sauce	3	2	12 - 12 oz jars	40.00	6	0
9	Mishi Kobe Niku	4	6	18 - 500 g pkgs.	97.00	29	0
10	Ikura	4	8	12 - 200 ml jars	31.00	31	0
11	Queso Cabrales	5	4	1 kg pkg.	21.00	22	30

Tabela Products ruan informacion mbi produktet që shet kompania. Për çdo produkt ruhen informacionet e mëposhtme:

- Kodi i Produktit në kollonën me emrin ProductId,
- Emri i produktit në kollonën me emrin ProductName,
- Kodi i furnitorit (të cilët ruhen në një tabelë më vhtë me emrin Suppliers) në kollonën me emrin SupplierId ,

- Kodi i kategorise së produktit (të cilat ruhen në një tabelë më vehte Categories)
- Cmimin e produktit në kollonën me emrin UnitPrice
- sasinë në stok në kollonën me emrin UnitsInStock

etj

Siç vihet re edhe nga baza e të dhënave e marrë si shembull:

- Rreshtat e një table nuk kanë ndonjë rend të paracaktuar
- Kollonat në një tabelë nuk kanë ndonjë rend të përcaktuar
- Nuk kemi dy kollona me të njëjtin emër brenda të njëjtës tabelë
- Mund të kemi kollona me të njëjtin emër nëse ato ndodhen në dy tabela të ndryshme (Për shembull kollona OrderId në tabelën Orders dhe kollona OrderId në tabelën OrderDetails si edhe EmployeeId në tabelën Employees dhe kollona EmployeeId në tabelën Orders)
- Të gjitha të dhënat që ruhen në një kollonë të një rreshti të tabelës janë skalare
- Në çdo tabelë, ka të paktën një kollonë (ose kombinimin kollonash si në tabelën Order Details kollonat OrderId dhe ProductID) që përmban vlera unike. Në modelin relacional të të dhënave këto kollona që shërbejnë si identifikues të rreshtave referohen si çelës kandidat. Nëse një tabelë ka disa çelësa kandidatë në një tabelë , dizenjuesi i bazës së të dhënave përcakton njërin prej çelësave kandidatë si çelësi primar i tabelës, p.sh OrderId është çelës primar i tabelës Orders, EmployeeId i tabelës Employees dhe CustomerId i tabelës Customers, ndërsa në tabelën OrderDetails çelësi primar përbehet nga kombinimi i dy kollonave OrderId dhe ProductId
- Në asnjë nga tabelat nuk ekzistojnë dy rreshta me përmbajtje identike

15. 3 Gjuha për bazat e të dhënave relacionale, SQL(Structured Query Language)

Versioni i gjuhës SQL që përdor SQL Server quhet Transact-SQL. Transact SQL konsiderohet si nj dialekt i Structured Query Language.

Lindja e gjuhës SQL është e lidhur ngushtë me një projekt të quajtur System R, të IBM në fillim të viteve 1980. Ky projekt provoi se është e mundur të përdoret model teorik i E. F. Codd për të ndërtuar një sistem relational të të dhënave.

SQL, gjuhë e bazuar tek bashkësitë

Ndryshe nga gjuhët e programimit si C, C++, apo Java, SQL është një gjuhë e bazuar tek bashkësitë. Kjo nënkupton që SQL mund të kërkojë të dhëna në shumë rreshta nga një ose disa tabela me një komandë të vetme. Kjo karakteristike është një nga avantazhet më të rëndësishme të SQL, pasi ofron një ndërfaqe programimi të një niveli të lartë abstraksioni, më larg gjuhës së makinës por më afër gjuhës natyrale njërëzore.

SQL gjuhë jo procedurale

Gjithashtu SQL është një gjuhë jo procedurale. Të gjithë programet e shkruajtura në një gjuhë procedurale (C, C++, Java) përshkruan si të kryhet një punë, hap pas hapi. Përkundrazi, SQL, përshkruan çfarë kërkon përdoruesi dhe është sistemi DBMS që gjen mënyrën më të mirë për t'iu përgjigjur kërkesës së përdoruesit.

DDL dhe DML

SQL përmban dy bashkësi komandash:

data definition language (DDL) dhe

data manipulation language (DML).

Komandat DDL përdoren për të përshkruar skemën e objekteve të bazës së të dhënave.

DDL përmban tre komanda të përgjithshme:

CREATE object

ALTER object

DROP object

Këto komanda krijojnë, modifikojnë dhe fshijnë objekte të bazës së të dhënave si database, tabelë, kollona, dhe indekse

Ndërsa DML përmbledh komandat që manipulojnë përmbajtjen e këtyre objekteve: afishimin, shtimin, fshirjen, dhe modifikimin.

- Komanda për gjetjen dhe afishimin është SELECT
- Komanda për gjetjen dhe afishimin është INSERT
- Komanda për gjetjen dhe afishimin është DELETE

- Komanda për gjetjen dhe afishimin është UPDATE

Kontroll Njohurish

1.1. Çfarë do të thotë pavarësi e të dhënave?

1.2. Cila është struktura e ruajtjes së të dhënave në bazat e të dhënave relacionale?

15.4 Tipet e të dhënave

MySQL ka disa tipe të dhënash për numrat, datat dhe kohen dhe stringjet. MySQL është gjithashtu e pajisur me struktura e mjete për ruajtjen e të dhënave gjeografike.

Tipet e të dhënave numerikë

MySQL është e pajisur me tipe të dhënash për numrat e plotë (INTEGER, SMALLINT), numrat e saktë me presje (DECIMAL, NUMERIC), si edhe numrat me presje me përafrim (FLOAT, REAL, dhe DOUBLE PRECISION).

Tipi i të dhënave Integer

MySQL është e pajisur me tipet standarte të numrave të plotë: INTEGER (ose INT) dhe SMALLINT. Gjithashtu ajo është e pajisur me tipet TINYINT, MEDIUMINT, dhe BIGINT. Këto tipe ndryshojnë nga njëri-tjetri nga madhësia e numrave që ato mbajnë. Tabela e mëposhtme tregon segmentet e numrave të plotë që mund të ruajë secili nga tipet e mësipërme të numrave të plotë:

Tipi	Madhësia	Vlera më e vogël	Vlera më e madhe
	(Ne Byte)	(me/pa shenjë)	(me/pa shenjë)
<u>TINYINT</u>	1	-128	127
		0	255
<u>SMALLINT</u>	2	-32768	32767

		0	65535
<u>MEDIUMINT</u>	3	-8388608	8388607
		0	16777215
<u>INT</u>	4	-2147483648	2147483647
		0	4294967295
<u>BIGINT</u>	8	-922337203685477580 8	922337203685477580 7

Numrat me presje me përafrim

Tipet FLOAT dhe DOUBLE përfaqësojnë numrat me presje me përafrim. MySQL përdor 4 byte për vlerat me precizion të njëfishtë dhe 4 byte për numrat me precizion të dyfishtë.

Numrat me presje me precizion

Tipi i të dhënave DECIMAL ruan numrat ekzakte me presje. Ky tip të dhënash përdoret kur përafrimi i numrave nuk është i lejueshëm për shembull vlerat monetare, numrat shkencore etj.

Atributet e tipeve numerike

Tipet integer mund të kenë një atribut: UNSIGNED. Atributi unsigned mund të përdoret për të lejuar vetëm numra jo negative në një kollonë. Për shembull nëse një kollonë e tipit INT është UNSIGNED, segmenti i vlerave që mund të mbajnë numrat e kësaj kollonë është nga 0 në 4294967295, në vënd të intervalit -2147483648 deri 2147483647 ku ndodhen normalisht numrat e tipit INT.

Edhe numrat me presje mund të përcaktohen si UNSIGNED.

Për tipin e të dhënave Integer ekziston edhe një tjetër atributë: AUTO_INCREMENT. Një kollonë që është përcaktuar si AUTO_INCREMENT i vendos vlerat e elementeve në mënyrë automatike, zakonisht duke nisur nga 1 dhe duke vazhduar me një rritje me+1.

Tipet për datat dhe kohën

Tipet e të dhënave për datat dhe kohën janë: DATETIME, DATE, TIMESTAMP, TIME, and YEAR.

Tipet DATETIME, DATE, dhe TIMESTAMP

Tipi DATETIME përdoret për të ruajtur vlera të kohës që përmbajnë datën dhe orën. MySQL i ruan vlerat e tipit datetime në formatin 'YYYY-MM-DD HH:MM:SS'. Intervali i vlerave të mundshme është nga '1000-01-01 00:00:00' deri në '9999-12-31 23:59:59'.

Tipi DATE përdoret për të ruajtur vlera të datës pa orën. MySQL i ruan vlerat e tipit DATE në formatin 'YYYY-MM-DD'. Interevali i vlerave të mundshme është nga '1000-01-01' deri në '9999-12-31'.

Tipi TIMESTAMP ruan vlera në intervalin nga '1970-01-01 00:00:01' UTC deri në '2038-01-19 03:14:07' UTC.

MySQL është dhënë fleksibël për mënyrën e paraqitjes së vlerave të kohës. Për shembull të gjithë formatet e mëposhtëm janë të saktë për MySQL:

'YYYY-MM-DD HH:MM:SS' ose 'YY-MM-DD HH:MM:SS'

Gjithashtu si ndarës midis komponentëve mund të përdoret çdo shenjë tjetër pikësimi.

Për shembull, '98-12-31 11:30:45', '98.12.31 11+30+45', '98/12/31 11*30*45', apo '98@12@31 11^30^45' janë që të gjitha të sakta dhe ekuivalente me njëra-tjetrën.

Një mënyrë për të shprehur të dhënat e tipit date është 'YYYY-MM-DD' ose 'YY-MM-DD'.

Për shembull, '98-12-31', '98.12.31', '98/12/31', dhe '98@12@31' janë të sakta dhe të njëjta me njëri-tjetrin.

Të dhënat datë e kohë mund të ruhen gjithashtu edhe në një format që nuk ka ndarje midis komponentëve të datës dhe kohës: 'YYYYMMDDHHMMSS' ose 'YYMMDDHHMMSS'. Për shembull, '20070523091528' dhe '070523091528' interpretohen si '2007-05-23 09:15:28', ndërsa '071122129015' është e pasaktë pasi

tek pjesa e minutave ka një vlerë jo llogjike ndaj do të interpretohej si '0000-00-00 00:00:00'.

Për shembull, '20070523' dhe '070523' interpretohen që të dyja si '2007-05-23', ndërsa '071332' nuk është e saktë pasi pasi tek pjesa e muajit dhe e ditës ka një vlerë të pasaktë (13 dhe 32 përkatësisht, ndaj do të interpretohet si '0000-00-00'.

Funksionet CURTIME dhe NOW

Funksioni CURTIME() kthen kohën e momentit, për shembull

```
SELECT CURTIME()
```

do të afishonte: 21:56:14

Funksioni NOW() kthen datën dhe kohën e momentit, për shembull

```
SELECT NOW()
```

do të afishonte: 2011-03-06 21:58:13

Tipi i të dhënave TIME

Ruan kohën në formatin 'HH:MM:SS' (ose 'HHH:MM:SS'). Intervali i vlerave TIME është nga '-838:59:59' deri në '838:59:59'. Arsyeja pse komponenti orë është i madh se 24 qëndron në faktin që tipi i të dhënave time mund të përdoret në aplikacione që ruajnë kohëzgjatjen e një eventi në orë, për shembull nëse duam të ruajmë kohën që ka punuar i gjithë ekipi në një projekt .

Formatet e të dhënave në tipin TIME janë shumë fleksibël. Të dhënat mund të paraqiten në disa formate si ' HH:MM:SS' , 'HH:MM', 'HHMMSS'.

Tipi YEAR

Ruan të dhënat për vitin dhe përdor një byte për ruajtjen e të dhënave.

Tipi YEAR mund të deklarohet si YEAR(2) ose YEAR(4) duke përcaktuar nëse viti do të ruhet me katër shifra ose me dy. Nëse nuk e përcaktojmë madhësinë YEAR nënkupton YEAR(4)

15.5 Tipet e të dhënave tekst

Tipet e të dhënave tekst janë me gjatësi fikse dhe me gjatësi variabël.

Tipet e të dhënave me gjatësi fikse (char) mundësojnë që të ruhen të tekste me gjatësi nga 0 deri tek gjatësia maksimale si edhe tipet e të dhënave me gjatësi variabël (varchar).

Tipet e të dhënave me gjatësi fikse ndryshe nga tipet e të dhënave me gjatësi variabël, rezervojnë vend në rreshtin e tabelës pavarësisht nëse ato kanë të dhëna ose jo, ndërsa tipet e të dhënave me gjatësi variabël zënë aq hapësirë sa tekstet përmbajnë.

Kjo mund të konsiderohet si një përparësi e tipeve tekst me gjatësi variabël, por kjo përparësi vjen me një kosto në performancë. Tipet e të dhënave me gjatësi fikse janë më të shpejtë.

Tipet e të dhënave tekst

Char(n) (n<=255)

Përdoret për tekste me gjatësi fikse me n shkronja/numra si për shembull nr_amzës , Nr_targes të cilëve ia dime paraprakisht numrin e shkronjave

Varchar(n) (n<=255)

Fjalë me gjatësi variabël 0 deri në n shkronja

TINYTEXT

Gjatësia maksimale është 255 karaktere

TEXT

Gjatësia maksimale është 65,535 karaktere

MEDIUMTEXT

Gjatësia maksimale 16,777,215 karaktere

LONGTEXT

Gjatësia maksimale 4,294,967,295 karaktere

Kjo kollonë nuk përdoret shumë pasi gজেেসia maksimale e stringut që lejon MySQL েসhtë vetëm 16 milion byte

MySQL ka gjithashtu tipet e të dhënave binary për ruajtjen e të dhënave binare:

BINARY(n)

Përdoret për të dhëna binare me gjatësi fikse.

VARBINARY(n)

Përdoret për të dhëna binare me gjatësi variabël.

Për të dhënat multimediale si imazhe etj përdoret tipi i të dhënave **BLOB**

TIPI ENUMERATE

Tipi enum përdoret për të përcaktuar një liste me vlera të mundshme.

Një kollonë mund të përcaktohet enum si më poshtë:
enum ('value1', 'value2', 'value3' ...) [default 'value']

Për shembull:

```
create table my_table (  
id int auto_increment primary key,  
Pyetje varchar(200),  
Përgjigje enum ('po', 'jo') default 'po'  
);
```

15. 6 Operatorët dhe funksionet

MySQL është e pajisur me operatorë dhe funksione të cilët janë përmbledhur në tabelën e mëposhtme:

Operatori	Përshkrimi
AND, &&	AND llogjik
=	Operatori i vlerëdhënies (përdoret në një komandë SET ose UPDATE)
:=	Operatori i vlerëdhënies
BETWEEN ... AND ...	kontrollon nëse një shprehje ndodhet midis dy vlerash
BINARY	konverton një string në vlerë të tipit binar
&	Operatori mbi bite AND
~	e kundërta e një biti (NOT)
 	OR mbi bite
^	XOR mbi bite
CASE	operatori Case
DIV	Pjesëtim i plotë
/	Pjesëtimi
<=>	NULL-safe equal to operator
=	operatori i krahasimit
>=	operator krahasimi, më i madh ose baraz
>	operatori i krahasimit më i madh se
IS NOT NULL	kontrollon që vlera nuk është null
IS NOT	Kontrollon një vlerë me kundërt një vlerë boolean
IS NULL	kontrollon nëse është null
IS	teston një vlerë kundërt një boolean
<<	Operator mbi bite, zhvendos majtas
<=	Operator krahasimi, më i vogël ose baraz
<	Operator krahasimi, më i vogël

<u>LIKE</u>	Operatori i krahasimit të stringjeve kundrejt një maske
<u>=</u>	Zbritja
<u>%</u>	Operatori i mbetjës
<u>NOT BETWEEN ... AND ...</u>	Operatori që krahason që vlera nuk është midis intervalit
<u>!=, <></u>	Operatori i ndryshëm
<u>NOT LIKE</u>	krahasim stringjesh
<u>NOT REGEXP</u>	Negation of REGEXP
<u>NOT, !</u>	e kundërta e një vlerë boolean
<u> , OR</u>	OR llogjik
<u>+</u>	Operatori i mbledhjes
<u>REGEXP</u>	kontrollon një string kundrejt një maske
<u>>></u>	Operatori mbi bite, zhvendos djathtas
<u>RLIKE</u>	Sinonim i REGEXP
<u>SOUNDS LIKE</u>	Funksion krahasimi
<u>*</u>	Operatori i shumëzimit
<u>-</u>	ndryshon shenjën e një numri
<u>XOR</u>	XOR llogjik

15.7 Funksionet mbi stringjet

My SQL është e pajisur me një numër të madh funksionesh dhe operatorësh mbi stringjet. Tabela më poshtë paraqet një listë me funksionet e MySQL mbi stringjet

Emri i funksionit	Pershkrim i funksionit
ASCII()	Kthen kodin e karakterit të parë të stringut
BIN()	Kthen një paraqitje binare e stringut
BIT_LENGTH()	Kthen gjatësinë në bite të argumentit
CHAR_LENGTH()	Kthen gjatësinë në numër shkronjash të

	stringut që merr si argument
CHAR()	Kthen shkronjën e një kodi, psh SELECT char(65) kthen A
CHARACTER_LENGTH()	Sinonim për CHAR_LENGTH()
CONCAT_WS()	Bashko stringjet me një simbol ndarës midis tyre
CONCAT()	Bashkon stringjet
CONV()	Konverton numrat në sisteme të ndryshme numërimi
INSTR()	Kthen indeksin e vendodhjes së parë të një cope stringu në një string.
LEFT()	Kthen një pjesë të stringut në të majtë të tij.
LENGTH()	Kthen gjatësinë e një stringu
LIKE	Krahason stringjet
LOWER()	Konverton një string në shkronja të vogla
LTRIM()	Heq hapësirat në të majtë të një stringu
MID()	Kthen një substring duke nisur nga një pozicionin i dhënë
REPEAT()	Përsërit një string një numër të dhënë heresh
REPLACE()	Zëvendëson karakterë një string
REVERSE()	Kthen stringun mbrapsht
RIGHT()	Kthen një pjesë stringu nga e djathta
RTRIM()	Heq hapësirat në të djathtë të stringut
STRCMP()	Krahason dy stringje
UPPER()	Konverton një string në shkronja të mëdha

15.8 FunkSIONET mbi kohën

Tabela më poshtë paraqet një listë me funksionet më të përdorura për datat dhe kohën:

Emri i funksionit	Përshkrimi
-------------------	------------

ADDDATE()	I shton një datë një vlerë
ADDTIME()	I shton një kohë një vlerë
CONVERT_TZ()	Konverton nga një timezone tek një tjetër
CURDATE()	Kthen datën korrente
CURRENT_DATE(), CURRENT_DATE	Sinonime të CURDATE()
CURRENT_TIME(), CURRENT_TIME	Sinonime të CURTIME()
CURRENT_TIMESTAMP(), CURRENT_TIMESTAMP	Sinonime për NOW()
CURTIME()	Kthen kohën korrent
DATE_ADD()	Mbledh data
DATE_FORMAT()	Formaton një datë sipas një formati të dhënë
DATE()	Nxjerr pjesën e datës nga një shprehje e tipit datetime
DATEDIFF()	Zbritje/diference midis dy datave
DAY()	Sinonim për DAYOFMONTH()
DAYNAME()	Kthen emrin e ditës së javës
DAYOFMONTH()	Kthen ditën e muajit (1-31)
DAYOFWEEK()	Kthen ditën e javës
DAYOFYEAR()	Kthen ditën e vitit (1-366)
EXTRACT	Nxjerr një pjesë të një date
FROM_DAYS()	Konverton një numër në një date
HOUR()	Ekstrakton orën nga një datetime
LAST_DAY	Kthen ditën e fundit të muajit për datën e marrë si argument
LOCALTIME(), LOCALTIME	Sinonim i NOW()
LOCALTIMESTAMP, LOCALTIMESTAMP()	Sinonim i NOW()
MAKEDATE()	Krijon një date
MAKETIME	Krijon një kohe
MINUTE()	Kthen minutat nga vlera që ka marrë si argument

MONTH()	Kthen muajin nga vlera që ka marrë si argument
MONTHNAME()	Kthen emrin e muajit nga vlera që ka marrë si argument
NOW()	Kthen datën dhe kohën e momentit
SECOND()	Kthen sekondat (0-59)
STR_TO_DATE()	Konverton një string në një date
TIME_FORMAT()	Formaton si kohë
TIME_TO_SEC()	Kthen vlerën e kohës marrë si argument në sekonda
TIME()	Kthen kohën e shkrehjës që ka marrë si argument
TIMEDIFF()	Zbritje/ difference midis dy kohëve

15.9 Komanda SELECT

Të marrim të dhëna nga SQL Server.

Sintaksa e komandës SELECT

SELECT (listë me kollona ose *)
FROM emertabele
WHERE (shprehje llogjike)
.GROUP BY (listë me kollona)
HAVING (shprehje llogjike)

Për shembull për të afishuar të dhënat e punonjësit me kodin 1 do të shkruanim komandën e mëposhtme:

```
SELECT *  
FROM employees  
WHERE EmployeeId=1
```

Shprehja llogjike mund të jetë e përbërë nga disa shprehje llogjike të lidhura nëpërmjet operatorëve llogjike AND,OR. Në këtë rast është e rekomandueshme të përdoren kllapat për të përcaktuar prioritetin e veprimeve.

Për shembull komanda më poshtë afishon listën e punonjësve që janë zonja ose zonjusha

```
SELECT *  
FROM employees  
WHERE TitleOfCourtesy = 'Ms.' OR TitleOfCourtesy = 'Mrs.'  
Operatori BETWEEN
```

Komanda e mëposhtme afishon një listë me punonjësit që janë marrë në punë vitin e fundit:

```
SELECT firstname, lastname, hiredate  
FROM employees  
WHERE hiredate >= '1-Jan-2010' AND hiredate <='31-Dec-2010'
```

Kjo komandë mund të shkruhet në një mënyrë më të përmblodhur duke përdorur operatorin BETWEEN

```
SELECT firstname, lastname, hiredate  
FROM employees  
WHERE hiredate  
between '1-Jan-2010' AND '31-Dec-2010'
```

Operatori NOT

Operatori NOT na mundëson llogaritjen e të kundërtës së një shprehje llogjike.

Shembulli i mëposhtëm, i cili afishon një listë të punonjësve që nuk janë marrë në punë vitin e fundit.

```
SELECT firstname, lastname, hiredate  
FROM employees  
WHERE hiredate  
NOT between '1-Jan-2010' AND '31-Dec-2010'
```

Operatori IN

Operatori IN krahason një shprehje kundrejt një liste vlerash. Lista e vlerave mund të jetë statike ose një listë me vlera e kthyer nga një komandë SELECT. Shembujt e mëposhtëm ilustrojnë të dy këto raste:

Komanda e mëposhtme afishon të gjithë punonjësit, mbiemri i të cilëve nuk ndodhet në listën ('Jones', 'Smith', 'Keenan')

```
SELECT firstname, lastname  
FROM employees  
WHERE lastname NOT IN ('Jones', 'Smith', 'Keenan')
```

Ndërsa komanda e mëposhtme afishon një liste me punonjësit që nuk kanë realizuar asnjë shitje

```
SELECT firstname, lastname  
FROM employees  
WHERE EmployeeId  
NOT IN  
(SELECT EmployeeId from Ordes)
```

Operatori EXISTS

Komanda e mëposhtme afishon një listë me punonjësit që kanë realizuar të paktën një shitje

```
SELECT * from Employees  
where  
exists  
(  
select * from Orders where  
Orders.EmployeeID=Employees.EmployeeID  
)
```

Operatoti LIKE

Operatori LIKE është një operator mbi stringjet, dhe mundëson krahasimin e një stringu(teksti) kundrejt një maske , që përmban shkronja, numra , shenja pikësimi si edhe disa simbole të vecantë si %

dhe _ që quhen wildcards.

Tabela më poshtë sqaron funksionin e wildcards

Simboli % nënkupton një numër variabël simbolesh nga 0 deri në n.

Simboli _ nënkupton saktësisht një simbol

psh [a-f] nënkupton një shkronjë nga a deri f

psh [^a-f] nënkupton një shkronjë çfarëdo mëpërjashtim të intervalit a deri f

[abc] nënkupton një simbol a ose b ose c

[^abc] nënkupton një simbol çfarëdo me përjashtim të a, b dhe c

Shembujt e mëposhtëm sqarojnë funksionin e operatorit LIKE:

Komanda

```
SELECT * from employees where Firstname like 'A%'
```

afishon një listë të punonjësve emrat e të cilëve fillojnë me shkronjën A , ndërsa komanda e mëposhtme:

```
SELECT * from employees where homephone like '(206) _ _ _ - _ _ _ _'
```

afishon një listë të punonjësve të cilët e kanë numrin e telefonit me prefix (206) , e me pas vazhdon me 3 shifra, vazhdon me simbolin - dhe me pas ka katër shifra.

Operatori IS NULL

Për të testuar nëse një kollonë e caktuar ka vlerën NULL duhet të përdorim operatorin IS NULL (ose IS NOT NULL kur duam të gjejmë vlera jo boshe).

Nuk është e mundur të krahasojmë për vlera NULL me shenjën e barazimit.

Homephone=NULL konsiderohet gabim , shkrimi korrekt do të ishte Homephone IS NULL

Për shembull, komanda e mëposhtme:

```
SELECT * from Employees where Birthdate IS NULL
```

afishon një listë të punonjësve për të cilët nuk e kemi të regjistruar datëlindjen, ndërsa komanda e mëposhtme:

```
SELECT * from Employees where Birthdate  
IS NOT NULL
```

afishon një listë të punonjësve për të cilët e kemi të regjistruar datëlindjen.

Te rendisim rezultatin nepermjet klauzoles ORDER BY

Klauzola Order By, mundëson të rëndisim rezultatin e një query sipas një ose disa kollonash.

ASC - ASCENDING, përcakton rendin rritës ndërsa

DESC - DESCENDING, përcakton rendin zbritës

Nëse nuk e përcaktojmë SQL nënkupton që po kërkojmë ti rendisim rreshtat në rendin rrites ASC.

Për shembull, komanda e mëposhtme:

```
SELECT firstname, lastname  
FROM Employees  
ORDER BY firstname
```

afishon një listë të punonjësve të renditur sipas emrit të tyre në rendin rritës (A->Z), ndërsa komanda e mëposhtme:

```
SELECT firstname, lastname  
FROM Employees  
ORDER BY firstname DESC ,lastname ASC
```

afishon një listë të punonjësve të renditur sipas emrit të tyre në rendin zbritës(Z->A) dhe më pas sipas mbiemrit në rendin rritës (A->Z)

Funksionet e Grupit

Ka disa funksione të SQL që aplikohen mbi grupet e rreshtave. Këto funksione si psh SUM, COUNT etj, shoqërohen me klauzolën GROUP BY.

Tabela më poshtë paraqet një përshkrim të funksioneve të grupit:

AVG - Mesatarja
COUNT - Numëron vlerat
MAX - maksimumi
MIN - minimumi
SUM - shuma
STDEV - Deviacioni standart (statitike)
VAR - Varianca (statistike)

Në rastin e SELECT të shoqëruar me Group By, vetëm kolumnat që ndodhen në listën e GROUP BY mund të jenë në listën e kolumnave të SELECT.

Rezultatet pas grupimit mund të filtrohen nëpërmjet HAVING BY, e cila është e ngjashme me WHERE, por në dallim nga kjo e fundit aplikohet pas llogaritjes së rezultateve mbi rreshtat e grupuara.

Për shembull, komanda e mëposhtme afishon numrin e shitjeve për çdo kod punonjësi:

```
SELECT EmployeeId,  
       Count(*) 'NoOfOrdersHeDealed'  
from Orders  
Group by (EmployeeId)
```

ndërsa komanda e mëposhtme afishon kodin e një porosie dhe vlerën financiare të saj si total

```
SELECT OrderId,SUM(Quantity*UnitPrice) 'totalamount' from  
[Order Details]  
Group by OrderID
```

Ndërsa komanda e mëposhtme afishon vetëm kodet e punonjësve që kanë bërë më shumë se 10 shitje:

```
SELECT EmployeeId,  
       Count(*) 'NoOfOrdersHeDealed'  
from Orders  
Group by (EmployeeId)  
Having Count(*) >10
```

Ushtrime të zgjidhura

Ushtrim 1

Afishoni për çdo kod funksitori numrin e produkteve që blejmë nga ai furnitor

-- renditini sipas numrit të produkteve bëj rendin zbritës

```
SELECT SupplierID,COUNT(*) 'prods'
```

```
From Products
```

```
GROUP BY SupplierID
```

```
ORDER BY COUNT(*) DESC
```

Ushtrim 2

Afishoni për çdo kod funksitori numrin e produkteve që blejmë nga ai furnitor

-- renditini sipas numrit të produkteve bëj rendin zbritës

-- afishoni vetëm furnitoret nga ku blejmë më shumë se 3 produkte

```
SELECT SupplierID,COUNT(*) 'prods'
```

```
From Products
```

```
GROUP BY SupplierID
```

```
HAVING COUNT(*)>3
```

```
ORDER BY COUNT(*) DESC
```

Ushtrim 3

Afishoni sa produkte kemi për çdo kategori

```
SELECT CategoryID,COUNT(*) 'Prods'
```

```
FROM Products
```

```
GROUP BY CategoryID
```

Ushtrim 4

Afishoni sa produkte kemi për çdo kategori

-- renditni rezultatin sipas numrit të produkteve në rendin zbritës

```
SELECT CategoryID,COUNT(*) 'Prods'
```

```
FROM Products
```

```
GROUP BY CategoryID
```

```
ORDER BY COUNT(*) DESC
```

Ushtrim 5

Afishoni sa produkte kemi për çdo kategori

-- renditni rezultatin sipas numrit të produkteve në rendin zbritës

-- afishoni vetëm kategoritë me më shumë se 4 produkte

```
SELECT CategoryID,COUNT(*) 'Prods'
```

```
FROM Products
```

```
GROUP BY CategoryID
```

```
HAVING COUNT(*)>4
```

```
ORDER BY COUNT(*) DESC
```

Ushtrim 6

Afishoni sa porosi kemi kryer me sejcilën nga postat (Shippers)

```
SELECT ShipVia 'Posta',COUNT(*) 'Porosi'
```

```
FROM Orders
```

```
GROUP BY ShipVia
```

Ushtrim 7

Afishoni sa porosi kemi kryer me sejcilën nga postat (Shippers)

-- renditni rezultatin sipas numrit të porosive në rendin zbritës

```
SELECT ShipVia 'Posta',COUNT(*) 'Porosi'
```

```
FROM Orders
```

```
GROUP BY ShipVia
```

```
ORDER BY COUNT(*) DESC
```

Ushtrim 8

Afishoni sa porosi kemi kryer me sejcilën nga postat (Shippers)

-- renditni rezultatin sipas numrit të porosive në rendin zbritës

-- afishoni vetëm postat me më shumë se 150 porosi

```
SELECT ShipVia 'Posta',COUNT(*) 'Porosi'
```

```
FROM Orders
```

```
GROUP BY ShipVia
```

```
HAVING COUNT(*)>150
```

ORDER BY COUNT(*) DESC

Ushtrim 9

Afishoni sa porosi kemi kryer me sejcilën nga postat (Shippers)

--gjate vitit 1998

-- renditni rezultatin sipas numrit të porosive në rendin zbritës

-- afishoni vetëm postat me më shumë se 50 porosi

```
SELECT ShipVia 'Posta',COUNT(*) 'Porosi'
```

```
FROM Orders
```

```
WHERE OrderDate between '1998-01-01' AND '1998-12-31'
```

```
GROUP BY ShipVia
```

```
HAVING COUNT(*)>50
```

```
ORDER BY COUNT(*) DESC
```

Ushtrim 10

Afishoni klientët me të mirë (më shumë se 5 porosi) për vitin 1998

```
SELECT CustomerId 'Klienti',COUNT(*) 'Porosi'
```

```
FROM Orders
```

```
WHERE OrderDate between '1998-01-01' AND '1998-12-31'
```

```
GROUP BY CustomerID
```

```
HAVING COUNT(*)>5
```

```
ORDER BY COUNT(*) DESC
```

Ushtrim 11

Afishoni sa lloje produktesh kemi gjendje (UnitInStock>0) për sejcilën kategori

```
SELECT CategoryID,COUNT(*) 'PRODUKTE_STOK'
```

```
FROM Products
```

```
grouP BY CategoryID
```

Ushtrim 12

Afishoni sa është vlere financiare e çdo porosie.

--Interesohemi vetëm për porosite e vitit 1998.

-- Porositë të renditen sipas vlerës financiare nga më e shtrenjta tek më e lira

```
SELECT OrderID,SUM(UnitPrice*Quantity) 'amount'
```

```
FROM [Order Details]
```

```
GROUP BY OrderId
```

15.10 Modifikimi i të dhënave (INSERT,UPDATE,DELETE)

Komanda INSERT

Komanda INSERT ruan rreshta në tabelë. Kjo komandë ka dy forma:

```
1) INSERT [INTO] tab_name [(col_list)]  
   DEFAULT VALUES | VALUES ({ DEFAULT | NULL | expression } [  
   ,...n] )
```

```
2) INSERT INTO tab_name | view_name [(col_list)]  
   {select_statement | execute_statement}
```

Përdorimi i mënyrës së parë bëmë të mundur që vetëm një rresht të ruhet në tabelën `tab_name`. Mënyra e dytë e komandës INSERT ruan një 'result set' nga një komandë SELECT ose nga një "STORED PROCEDURE", e cila ekzekutohet me anë të komandës "EXECUTE".

Me të dy mënyrat, çdo vlerë që ruhet duhet të ketë një tip të dhënës i cili të jetë kompatibël me tipin e të dhënës së kolonës korresponduese të tabelës. Për të garantuar kompatibilitetin, të gjitha vlerat me bazë karakteresh duhet patjetër që të futën në apostrofa, kurse për vlerat numerike një gjë e tillë nuk është e nevojshme.

Ruajtja e një rreshti të vetëm

Në të dy mënyrat e komandës INSERT specifikimi eksplicit i kolonave është opsional. Kjo do të thotë që

Në të dy mënyrat e komandës INSERT specifikimi eksplicit i kolonave është opsional. Kjo do të thotë që mungesa e listës së kolonave është ekuivalente me specifikimin e listës së të gjithë kolonave të tabelës.

Opsioni I "DEFAULT VALUES" (vlerat default) ruan vlera default për ato kolona të cilat nuk janë specifikuar në mënyrë eksplicite në komandën INSERT. Nqs një kolonë është e tipit të të dhënave "TIMESTAMP" ose në property është vendosur "IDENTITY", vlera që krijohet automatikisht nga sistemi do të ruhet në kolonën përkatëse. Për tipet e tjera të të dhënave që lejojnë vlera NULL do të ruhet vlera e specifikuar në mënyrë eksplicite ,ose vlera default në mënyrë implicite, ose po qe se nuk është specifikuar default do të

ruhet vlere null. Nqs kolona nuk lejon vlere null dhe nuk ka vlere default të përcaktuar në property, atëherë komanda INSERT nuk do të ekzekutohet me sukses dhe do të afishoj errorin përkatës.

Ushtrimi 1

Ruajtja e të dhënave në tabelën employee

USE sample;

```
INSERT INTO employee VALUES (25348, 'Matthew', 'Smith','d3');
```

```
INSERT INTO employee VALUES (10102, 'Ann', 'Jones','d3');
```

```
INSERT INTO employee VALUES (18316, 'John', 'Barrimore', 'd1');
```

Ushtrimi 2

Ruajtja e të dhënave në tabelën department

USE sample;

```
INSERT INTO department VALUES ('d1', 'Research', 'Dallas');
```

```
INSERT INTO department VALUES ('d2', 'Accounting', 'Seattle');
```

```
INSERT INTO department VALUES ('d3', 'Marketing', 'Dallas');
```

Ushtrimi 3

Ruajtja e të dhënave në tabelën project

USE sample;

```
INSERT INTO project VALUES ('p1', 'Apollo', 120000.00);
```

```
INSERT INTO project VALUES ('p2', 'Gemini', 95000.00);
```

```
INSERT INTO project VALUES ('p3', 'Mercury', 186500.00);
```

Ushtrimi 4

Ruajtja e të dhënave në tabelën works_on

USE sample;

```
INSERT INTO works_on VALUES (10102,'p1', 'Analyst', '2006.10.1');
```

```
INSERT INTO works_on VALUES (10102, 'p3', 'Manager', '2008.1.1');
```

```
INSERT INTO works_on VALUES (25348, 'p2', 'Clerk', '2007.2.15');
```

```
INSERT INTO works_on VALUES (18316, 'p2', NULL, '2007.6.1');
```

```
INSERT INTO works_on VALUES (29346, 'p2', NULL, '2006.12.15');
```

```
INSERT INTO works_on VALUES (2581, 'p3', 'Analyst', '2007.10.15');
```

```
INSERT INTO works_on VALUES (9031, 'p1', 'Manager', '2007.4.15');
```

```
INSERT INTO works_on VALUES (28559, 'p1', 'NULL', '2007.8.1');
```

Ka dhe mënyra të tjera për ruajtjen e të dhënave në një rresht. Shembujt nga 5 deri tek 7 tregojnë për gjithë mënyrat e tjera të mundëshme.

Ushtrimi 5

USE sample;

```
INSERT INTO employee VALUES (15201, 'Dave', 'Davis', NULL);
```

Mënyra e ruajtjes së të dhënave në Ushtrimin 5 I korrespondon komandës INSERT të paraqitur në ushtrimet nga 1 deri në 4. Paraqitja në mënyrë eksplicite të NULL ruan vlera null në kolonën respective.

Ushtrimi 6

USE sample;

```
INSERT INTO employee (emp_no, emp_fname, emp_lname)  
VALUES (15201, 'Dave', 'Davis');
```

Ushtrimi 5 dhe 6 janë ekuivalent.

Ushtrimi 7

USE sample;

```
INSERT INTO employee (emp_lname, emp_fname, dept_no,  
emp_no)  
VALUES ('Davis', 'Dave', 'd1', 15201);
```

Është e domosdoshme që renditja e vlerave në "VALUES" të jetë e njëjtë me renditjen e kolonave në "INSERT INTO Table".

Ruajtja e një ose ëe shumë se një rreshti

Mënyra e dytë e komandës INSERT ruan një ose më shumë rreshta ma anë të një "subquery".

Ushtrimi 8

Të merren të gjithë numrat dhe emrat e departamentëve te një lokacioni dhe të ruhen në një tabelë të re.

USE sample;

```
CREATE TABLE dallas_dept  
(dept_no CHAR(4) NOT NULL,
```

```
dept_name CHAR(20) NOT NULL);
```

```
INSERT INTO dallas_dept (dept_no, dept_name)
SELECT dept_no, dept_name
FROM department
WHERE location = 'Dallas';
```

Subquery në komandën INSERT selekton të gjithë rreshtat që kanë vlerën “Dallas” në kolonën location. Rreshtat e selektuar me anë të subquery do të ruhen në tabelën e re.

Përmbajtja e tabelës dallas_dept mund të shihet me anë të komandës SELECT.

```
SELECT * FROM dallas_dept;
```

Rezultati është si më poshtë:

<i>dept_no</i>	<i>dept_name</i>
d1	Research
d3	Marketing

“Table Value Constructors” dhe INSERT

SQL Server 2008 ka shtuar një funksionalitet të ri të quajtur “table value constructor”, e cila ju lejon të vendosni disa rreshta në një DML. Ushtrimi 9 ju tregon se si mund të vendosni rreshta duke përdorur konstruktoret e komandës INSERT.

Ushtrimi 9

```
USE sample;
INSERT INTO department VALUES ('d4', 'Human Resources',
                                'Chicago'),
                                ('d5', 'Distribution', 'New Orleans'),
                                ('d6', 'Sales', 'Chicago');
```

Komanda insert në ushtrimin 9 ruan 3 rreshta në të njëjtën kohë në tabelën department duke përdorur “table value constructor”.

Komanda UPDATE

Komanda UPDATE modifikon vlerat e një tabele. Kjo shprehje ka formën e përgjithshme si më poshtë:

```
UPDATE tab_name  
{ SET column_1 = {expression | DEFAULT | NULL} [,...n]  
[FROM tab_name1 [,...n]]  
[WHERE condition]
```

Rreshtat në tabelën tab_name modifikohen duke u bazuar në ato rreshta që plotësojnë kushtin WHERE. Për çdo rresht që do të modifikohet shprehja UPDATE ndryshon vlerën e kolonave që janë specifikuar në "SET", duke ju vendosur një vlerë konstante ose një shprehje variablash. Në kushtin WHERE mundon atëherë komanda UPDATE do të modifikojë të gjithë rreshtat e tabelës.

Ushtrimi 10

Vendos që punonjësi me numër 18316, i cili punon në projektin p2 të jetë 'Manager':

```
USE sample;  
UPDATE works_on  
SET job = 'Manager'  
WHERE emp_no = 18316  
AND project_no = 'p2';
```

Komanda UPDATE në ushtrimin 10 modifikon ekzaktësisht një rresht në tabelën works_on, sepse kombinimi i kolonave emp_no dhe project_no përbën një çelës primar të tabelës dhe është për pasojë unike.

Ushtrimi 11 modifikon rreshtat e një table me një shprehje variablash.

Ushtrimi 11

Ndrysho buxhetin e të gjithë projekteve duke i paraqitur vlerat e tyre në pound. Kursi i kembimit është 0.51

```
USE sample;  
UPDATE project
```

```
SET budget = budget*0.51;
```

Në këtë shembull, të gjithë rreshtat e tabelës project do të modifikohen sepse mungon klauzola WHERE. Rreshtat e modifikuar në tabelën project mund të shihen me komandën më poshtë:

```
SELECT * FROM project;
```

Rezultati është:

<i>project_no</i>	<i>project_name</i>	<i>budget</i>
p1	Apollo	61200
p2	Gemini	48450
p3	Mercury	95115

Ushtrimi 12

Për shkak të sëmundjes së saj të gjithë projektet e Mrs. Jones duhet të bëhen NULL:

```
USE sample;  
UPDATE works_on  
SET job = NULL  
WHERE emp_no IN  
(SELECT emp_no  
FROM employee  
WHERE emp_lname = 'Jones');
```

Ushtrimi 12 përdor një query të brendëshme në klauzolen WHERE të komandës UPDATE.

Për shkak të përdorimit të operatorit IN, më shumë se një rresht kthehen nga kjo query.

Ushtrimi 13 ilustron përdorimin e klauzolës FROM. Ky ushtrim është identik me të mëparshmin.

Ushrtimi 13

```
USE sample;  
UPDATE works_on  
SET job = NULL  
FROM works_on, employee
```



```
WHERE emp_lname = 'Jones'  
AND works_on.emp_no = employee.emp_no;
```

Ushtrimi 14 ilustron përdorimin e shprehjes CASE në komanden UPDATE.

Ushtrimi 14

Buxheti për çdo project duhet të rritet me një përqindje të caktuar (20, 10 ose 5) në varësi të sasisë së mëparshme të parave. Projektet me një buxhet më të vogël do të rriten në përqindje më të madhe.

```
USE sample;  
UPDATE project  
SET budget = CASE  
                WHEN budget > 0 and budget < 100000 THEN  
budget*1.2  
                WHEN budget >= 100000 and budget < 200000 THEN  
budget*1.1  
                ELSE budget*1.05  
                END
```

Komanda DELETE

Komanda DELETE fshin rreshta nga tabelat. Kjo komandë ka dy forma të ndryshme:

```
DELETE FROM table_name  
[WHERE predicate];
```

```
DELETE table_name  
FROM table_name [...]  
[WHERE condition];
```

Të gjithë rreshtat që plotësojnë kushtin WHERE do të fshihen. Paraqitja në mënyrë eksplicite e kolonave të një table nuk është e nevojshme (ose nuk lejohet), sepse komanda DELETE operon mbi rreshtat dhe jo mbi kolonat.

Ushtrimi 15

Fshi të gjithë manageret në tabelën works_on:

```
USE sample;  
DELETE FROM works_on  
WHERE job = 'Manager';
```

Kushti WHERE në komandën DELETE mund të përmbaje një query të brendëshme:

Ushtrimi 16

Punonjësi Moser është duke lënë punën. Të fshihen të gjithë rreshtat në database që i përkasin atij.

```
USE sample;  
DELETE FROM works_on  
WHERE emp_no IN  
(SELECT emp_no  
FROM employee  
WHERE emp_lname = 'Moser');
```

```
DELETE FROM employee  
WHERE emp_lname = 'Moser';
```

Ushtrimi 16 mundet gjithashtu të realizohet duke përdorur FROM, si është shpjeguar në ushtrimin 17. Kjo ka të njëjtën structure si në komandën UPDATE.

Ushtrimi 17

```
USE sample;  
DELETE works_on  
FROM works_on, employee  
WHERE works_on.emp_no = employee.emp_no  
AND emp_lname = 'Moser';
```

```
DELETE FROM employee  
WHERE emp_lname = 'Moser';
```

Përdorimi I klauzolës WHERE në komandën DELETE është opsional. Nqs klauzola WHERE mungon, të gjithë rreshtat e tabelës do të fshihen, si në ushtrimin 18.

Example 18

```
USE sample;
```

```
DELETE FROM works_on;
```

20.11 Shembull ngarkimi i imazheve

Më poshtë do të ilustrojmë ngarkimin e imazheve nëpërmjet kodit php. Të dhënat mbi imzhet do të ruhen në një tabelë me emrin images.

Së pari do të krijojmë një tabelë për ruajtjen e imazheve.

```
CREATE TABLE `images` (  
  `code` int(11) NOT NULL auto_increment,  
  `title` varchar(150) NOT NULL,  
  `path` varchar(200) NOT NULL,  
  `description` text,  
  `section` int(11) NOT NULL default '0',  
  `rendi` int(11) default NULL,  
  `active` int(11) NOT NULL default '1',  
  `attachedto` int(11) NOT NULL default '1',  
  PRIMARY KEY (`code`)  
) ENGINE=MyISAM DEFAULT CHARSET=latin1  
AUTO_INCREMENT=169
```

Dosja ku do të ngarkohen imazhet duhet të jetë me të drejta shkrimi

në linux chmod 777

në windows eveyone read,write access

Do të përdorim dy skedare.

SKEDARI 1 - newimg.php

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
</head>

<body>
<form name="form1" method="post" action="newimg1.php"
enctype="multipart/form-data">
  <p>Prona :
    <input type="text" name="section" value="1">

<div id="preview"></div>
</p>
<select name="attachedto">
  <option value="1"> article </option>
  <option value="2"> section </option>
  <option value="3"> property </option>
  <option value="4"> customers </option>
  <option value="5"> employees </option>
</select>
<p>
Titulli <input type="text" name="title">
</p>
<p>Pershkrim
<textarea rows="5" cols="50" name="description" > </textarea>
</p>
<p>Rendi
<input type="text" name="rendi">
</p>
<p>
Active
<select name="active">
<option value="0">Fotoja nuk do të afishohet</option>
<option value="1">Fotoja do të afishohet</option>
</select>
```

```
</p>
<p>fotoja:<input type="file" name="myfiles[]" >
</p>
<input type="submit" value="--NGARKO--">

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

dhe skedari newimg1.php

```
<?php
require_once("start.php");
require_once("myconfig.php");
require_once("db.php");
require_once("kontrollo.php");
require_once("mylib.php");

// elementi ku është lidhur imazhi
if (!isset($_REQUEST["section"]))
{
    exit("cilën element??");
}
$section=$_REQUEST["section"];

// elementi ku është lidhur imazhi
if (!isset($_REQUEST["title"]))
{
    exit("cilën është titulli??");
}
$title=$_REQUEST["title"];

// elementi ku është lidhur imazhi
if (!isset($_REQUEST["description"]))
{
    exit("cilën description??");
```

```
}
$description=$_REQUEST["description"];

// elementi ku është lidhur imazhi
if (!isset($_REQUEST["rendi"]))
{
    exit("cilën rendi??");
}
$rendi=$_REQUEST["rendi"];

// elementi ku është lidhur imazhi
if (!isset($_REQUEST["active"]))
{
    exit("cilën element??");
}
$active=$_REQUEST["active"];

// elementi ku është lidhur imazhi
if (!isset($_REQUEST["attachedto"]))
{
    exit("cilën attavh??");
}
$attachedto=$_REQUEST["attachedto"];

// ngarkojmë dokumentat

$numoffile = 1;
// Fix path of your file to be uploaded, don't forget to CHMOD 777
to this folder
$file_dir = "../images/";

if ($_POST) {
    for ($i=0;$i<$numoffile;$i++) {
        if (trim($_FILES['myfiles']['name'][$i])!="") {
            $newfile = $file_dir.$_FILES['myfiles']['name'][$i];
            move_uploaded_file($_FILES['myfiles']['tmp_name'][$i],
            $newfile);
        }
    }
}
```

```
    }  
  }  
}  
  
//fund ngarkimi i dokumentave  
  
$f1=$_FILES['myfiles']['name'][0];  
  
/*  
$f2=$_FILES['myfiles']['name'][1];  
$f3=$_FILES['myfiles']['name'][2];  
*/  
$sql="insert                                     into  
images(title,path,description,section,rendi,active,attachedto)  
values  
('".$title."','".$f1."','".$description."','".$section."','".$rendi."','".$active."','".$a  
ttachedto.")";  
echo $sql;  
mysql_query($sql)  
or die(mysql_error());  
  
echo 'imazhet u ngarkuan';  
  
?>
```


Teste

1. Kur bëjmë submit një form, të dhënat e saj transmetohen me anë të:
 - a.ciftit name/value
 - b.name
 - c. value
 - d. asnjë nga të lartmet
2. Për rreshtin e kodit
`<input type="textarea" name="gabimi" rows="20" cols="20"> Gjej gabimin </textarea>` mund të themi se:
 - a. kemi maksimum 20 karaktere që mund të mbaj një rresht i tekstit
 - b. kemi një kontroll formë që mund të shtypim vetëm tekst dhe jo numra
 - c. vlera fillestare që i paraqitet përdoruesit është gabimi
 - d. asnjë nga të mëlartmet
3. Që një qelizë table të këtë sfond gri, shkrim të nënvizuar dhe largësi prej kontureve 5 pixel duhet të zgjedhim cilin option:
 - a.`td { background-color:gray; text-decoration:underlined; padding:5px;}`
 - b.`th { background-color:gray; text-decoration:underlined; padding:5px;}`
 - c.`th{ bgcolor:gray; padding:5 px; text-decoration:underlined}`
 - d.`table { background-color:gray; text-decoration:underlined; padding:5px;}`
4. Drejtimi i tekstit specifikohet me anë të:
 - a. text-align
 - b.vertical-align
 - c. direction
 - d. asnjë nga të mëlartmet
5. Nqs kemi elementin në HTML `<caption></caption>` ath në stile CSS do të përkthehet si:
 - a.`caption{ font-weight:500;}`
 - b.`caption{ font-weight:700;}`

c.caption{ font-weight:bold;}
d.të gjitha të mëlartmet

6. Në rastin e formës file upload është e vërtetë:
 - a. përdoret elementi <input>
 - b. përdoret elementi <file>
 - c.vlera e metodës së formës për të bërë upload vendoset në get
 - d. asnjë nga të mëlartmet
7. Kontrolli i formës radio button:
 - a.selekton të paktën një opsion në një grup opsionesh
 - b.selekton të shumtën një opsion në një grup opsionesh
 - c. nuk prano atributin checked
 - d. asnjë nga të mëlartmet
8. Atributi href tek CSS specifikon:
 - a. URL absolute te file css
 - b. URL relative te file css
 - c. URL e dokumentit që duam të bëjmë link
 - d. asnjë nga të mëlartmet
9. Nqs duam të specifikojmë në nivel elementi t që konturet të jenë horizontalisht dhe vertikalisht të qëndëruara duhet të përcaktoj properties:
 - a. text-align dhe vertical-align
 - b. vertical-align dhe text-indent
 - c. text-indent dhe text-align
 - d. asnjë nga të mëlartmet
10. Nqs duam të paraqesim vetëm njërin prej kontureve të dy qelizave fqinje ath duhet të përdorim opsionin:
 - a. border-collapse:collapse
 - b. border-collapse:separate
 - c. border-separate:collapse
 - d. border-separate:separate
11. Vlera e atributit action në një form nuk mund të jetë:
 - a. logimi.php
 - b. www.SiteIm.com/logimi.php

- c. MyPages/logimi.php
 - d. asnjë nga të mëlartmet
12. Në rastin e kontrollit të formës textarea atributi rows tregon :
- a. numrin e rreshtave të kontrollit textarea
 - b. specifikon madhësinë e kontrollit të texarea
 - c. tregon lartësinë e textarea
 - d. të gjitha të mëlartmet
13. Nqs propertia font-wight ka vlerën 100 ath shkrimi paraqitet si:
- a. normal
 - b. bold
 - c. italic
 - d. asnjë nga të mëlartmet
14. Vlera Capitalize specifikohet për propertine:
- a. text-decoration
 - b. text-transform
 - c. text-shadow
 - d.asnjë nga të mëlartmet
15. Nqs duam që distanca midis kontureve horizontale të jetë dyfishi i distancës vertikale ath duhet të zgjedhim:
- a.border-spacing:8px 4px
 - b.border-spacing:4px 8px
 - c.vborder-spacing:4px
hborder-spacing:8px
 - d.asnjë nga të mëlartmet
16. Cila prej opsioneve të mëposhtme mund ti vendoset atributit method në një form:
- a. send
 - b.submit
 - c. post
 - d. asnjë nga të mëlartmet
17. Kontrolli i formave checkboxe:
- a. nuk pranon atributin checked
 - b. selekton vetëm një opsion ndërmjet një grupi me opsione
 - c. selekton të paktën një opsion në një grup opionesh

- d. asnjë nga të mëlartmet
18. Për cilin nga rangjet e mëposhtme shkrimi paraqitet me i theksuar duke përdorur stilet CSS:
- a. 0-100
 - b. 100-400
 - c. 500-900
 - d. asnjë nga të mëlartmet
19. Nqs në HTML duam të kemi elementin `<th width="36px">` ath në CSS do kishim:
- a. `td{ width:36px;}`
 - b. `th{ width:36px;}`
 - c. `td{ width:36pt;}`
 - d. `th{ width:36pt;}`
20. Nqs duam që elementi `caption` i një table në CSS të vendoset majtas tabelës do të zgjedhim:
- a. `align:left`
 - b. `caption-side:left`
 - c. `caption-direction:left`
 - d. asnjë nga të mëlartmet
21. Cila prej opsioneve të mëposhtme mund ti vendoset atributit `method` në një form:
- a. `connect`
 - b. `receive`
 - c. `get`
 - d. asnjë nga të mëlartmet
22. Në rastin e kontrollit të formës `textarea` themi se :
- a. `textarea` përmban aq kolona sa është vlera e përcaktuar për atributin `cols`
 - b. një kolonë është sa gjërësia mesatare të një shkronjë
 - c. numrin e karakterëve që mund të mbaj një rresht në `textarea`
 - d. të gjitha të mëlartmet
23. Propertia `text-indent` ju lejon të specifikoni :

- a. largësine nga konturi i majte i elementit për rreshtin e parë të shkrimit
 - b. largësine nga konturi i djathtë i elementit për rreshtin e parë të shkrimit
 - c. largësine nga konturi i majtë i elementit për rreshtin e fundit të shkrimit
 - d. largësine nga konturi i djathtë i elementit për rreshtin e fundit të shkrimit
24. Nqs në HTML do kishim elementin `<table bgcolor="gray">` ath si stil CSS do të përzgjidhnim:
- a. `table{background-color:gray;}`
 - b. `table{color:gray;}`
 - c. `table{bgcolor:gray;}`
 - d. asnjë nga të mëlartmet
25. Në rast se `background -position` ka vlerën `7% 7%` ath imazhi që përdoret si sfond do jetë:
- a. i fiksuar në pozicionin 7 px me 7 px të elementit
 - b. i fiksuar në pozicionin 7 px me 7 px të faqes
 - c. ripërmasimi i dritares së browserit do të ruaj pozicionim e figures në përqindjet e dhëna
 - d. asnjë nga të mëlartmet
26. Atributi `target` në nivel formë tregon:
- a. rezultati i formës shfaqet në dritare të re
 - b. rezultati I formës shfaqet në të njëjtën dritare më formën
 - c. se në cilën dritare do shfaqet rezultati i formës
 - d. asnjë nga të mëlartmet
27. Në rastin e kontrollit të formës `select-boxes` atributi `size` tregon:
- a. numrin e opsioneve të dukshëm për përdoruesin
 - b. `size` ka gjithmonë vlerën 1
 - c. numrin e opsioneve që janë të selektueshëm prej përdoruesit
 - d. asnjë nga të mëlartmet
28. Nqs një properti specifikohet në mënyrë eksplicite në nivel të një elementi, `propertia` e trashëguar prej elementit prind a bëhet `overwrite`:

- a. Po
 - b. Jo
29. Propertia font-wight me vlerë 500 tregon se:
- a. shkrimi është normal
 - b. shkrimi është i pjerret
 - c. shkrimi është i theksuar
 - d. asnjë nga të mëlartmet
30. Propertia background-position me vlerë x% dhe y% tregon se figura që përdoret si sfond:
- a. vendoset në pozicionin x dhe y pixel duke nisur nga cepi lart majtas faqe
 - b. vendoset në pozicionin x dhe y pixel duke nisur nga cepi lart majtas elementit
 - c. pozicionohet në raportin x dhe y duke u orientuar sipas aksëve të faqes
 - d. pozicionohet në raportin x dhe y duke u orientuar sipas aksëve të elementit
31. Funksionaliteti i në butoni mund të arrihet me anë të:
- a. elementit <button>
 - b. elementit <input> të ndërthurur me tipin button
 - c. elementit figurë të mbështjellë me elementin submit
 - d. asnjë nga të mëlartmet

Referencat

1. Introduction to Web Applications Development, Carles Mateu, 2010
2. Introduction to Dynamic HTML, *Aaron Weiss*, 1998
3. Web Life 2.0: A Guide through the World Wide Web, Linda Goin, 2010
4. Building Android Apps with HTML, CSS, and JavaScript, Jonathan Stark, ISBN-10: 1449383262, 2010
5. Building A JavaScript Framework, Alex Young, 2010
6. The Web Book: The ultimate beginner's guide to HTML, CSS, JavaScript, PHP and MySQL, Robert Schifreen, 2010
7. <http://www.w3schools.com/>
8. <http://php.net>
9. <http://jquery.com/>