

Univerzitet u Zenici



Pedagoški fakultet

Katedra za matematiku i informatiku

Programiranje za Internet

doc.dr. Samra Mujačić

v.as.mr. Samir Lemeš

Zenica, 2006

**Besplatan
primjerak!**

Sadržaj

1.	Uvod	1
1.1.	Internet i njegova organizacija	1
1.1.1.	Nastanak Interneta	1
1.1.2.	Organizacija Interneta	2
1.1.3.	Ko upravlja Internetom?	3
2.	Klijentsko-serverski model	5
2.1.	Mrežne adrese	5
2.1.2.	Rezervirane IP adrese	7
2.1.3.	Kako dobiti IP adresu	7
2.2.	Imena računara	8
2.3.	Maskiranje	9
2.4.	DNS serveri: pretvaranje FQDN u IP adresu	10
2.4.1.	Obrnuto DNS mapiranje: pretvaranje IP u FQDN adresu	11
3.	Osnovne mrežne usluge	13
3.1.	Portovi	13
3.2.	Rad na udaljenom računaru (Telnet)	15
3.3.	HTML	15
3.3.1.	Historija HTML-a	16
3.4.	Pristup dokumentima i URL-ovi	19
4.	HTML	20
4.1.	Struktura stranice	21
4.2.	Vrste naredbi	22
4.3.	Sintaksa naredbi	23
4.4.	SSI	26
4.5.	Osnovni elementi jezika	28
4.5.1.	Naslovi	28
4.5.2.	Uobičajeni argumenti	31
4.5.3.	Odlomci	31
4.5.4.	Novi redak	32
4.5.5.	Vodoravne linije	32
4.5.6.	Citati	33
4.5.7.	Adrese	34
4.5.8.	Napomene i fusnote (bilješke)	34
4.5.9.	Neproporcionalno pismo	35
4.5.10.	Liste	36
4.6.	Tablice	42
4.7.	Stilovi	47
4.8.	Hipertekstualne veze	49
4.9.	Slike	51
4.9.1.	Slikovne mape	56
4.10.	Multimedija	57
4.11.	Obrasci	58
4.12.	Client-Pull – klijentsko povlačenje?	61
4.13.	Skripting jezici	65

4.13.1.	Skripting jezici koji se izvršavaju na serveru (Server-side scripting)	65
4.13.2.	Skripting jezici koji se izvršavaju kod klijenta (Client-side scripting)	67
5.	Macromedia Dreamweaver	68
5.1.	Uvod u Macromedia Dreamweaver	68
5.2.	Razvijanje strukture prezentacije	71
5.3.	Definisanje lokalne web prezentacije	72
5.4.	Zadavanje browsera	75
5.5.	Pravljenje i snimanje nove strane	76
5.5.1.	Dajte naslov web strani	78
5.5.2.	Zadavanje boje pozadine	78
5.5.3.	Slika na pozadini strane	80
5.5.4.	Zadavanje podrazumijevane boje fonta	80
5.6.	Smještanje teksta na stranu	80
5.6.1.	Uvoženje teksta	81
5.6.2.	Centriranje i uvlačenje teksta	82
5.6.3.	Pravljenje spiskova	83
5.6.4.	Ugnježdivanje lista	84
5.6.5.	Formatiranje znakova	84
5.6.6.	Promjena fonta	85
5.6.7.	Promjena veličine fonta	86
5.6.8.	Mijenjanje boje slova	87
5.7.	Definisanje html stilova	87
5.7.1.	Primjena html stilova	88
5.8.	Dodavanje specijalnih znakova	89
5.9.	Dodavanje horizontalnih linija	89
5.10.	Automatsko dodavanje datuma	90
5.11.	Umetanje grafičkih elemenata na stranu	91
5.11.1.	Mijenjanje veličine grafičkih elemenata i vraćanje na prvobitnu veličinu	93
5.11.2.	Pozicioniranje slika na strani	94
5.11.3.	Dodavanje okvira oko slike	95
5.11.4.	Omotavanje teksta oko slike	95
5.11.5.	Prilagođavanje prostora oko slike	96
5.11.6.	Poravnavanje slike u odnosu na jedan red teksta	96
5.11.7.	Izrada mapiranih slika	97
5.12.	Crtanje tabela u prikazu layout	98
5.12.1.	Mijenjanje nacrtu tabele	100
5.12.2.	Formatiranje ćelija	100
5.12.3.	Širina tabele i slike koje održavaju razmak	101
5.12.4.	Izrada tabele u prikazu standard	102
5.13.	Pravljenje šablona (Template-a)	104
5.13.1.	Ubacivanje oblasti koje se mogu mijenjati u šablon	104
5.13.2.	Uklanjanje promjenljivih oblasti	106
5.13.3.	Izrada opcionih oblasti šablona	106
5.13.4.	Pravljenje oblasti koje se ponavljaju	107
5.13.5.	Pravljenje strana pomoću šablona	108
5.13.6.	Kontrolisanje pojavljivanja opcionih oblasti	109
5.13.7.	Izmjena stranica pomoću šablona	109

6.	Izrada obrasca	111
6.1.	Dodavanje polja za jedan red teksta	112
6.2.	Dodavanje polja za više redova teksta	115
6.3.	Dodavanje polja za potvrdu	116
6.4.	Dodavanje radio-dugmadi	117
6.5.	Dodavanje spiskova i menija	118
6.6.	Dodavanje dugmadi	120
6.7.	Provjera valjanosti podataka u obrascima	121
6.8.	Testiranje obrasca	123
7.	Meta oznake i komentari	125
8.	Pravljenje spoljnih kaskadnih opisa stilova	127
8.1.	Mijenjanje postojećeg stila	130
8.2.	Izrada kaskadnih opisa CSS stilova za hiperveze	132
8.3.	Povezivanje s postojećim opisom stila	133
9.	Rad s bojama	134
9.1.	Pozadina stranice	134
9.2.	Kodiranje boja	135
10.	Slike u HTML dokumentima	137
10.1.	Veličina slika	137
10.2.	Korištenje manjih datoteka	138
10.3.	Prozirne i isprepletene GIF slike	139
11.	CGI programiranje	140
11.1.	Upoznavanje sa skriptama	140
12.	JavaScript	147
12.1.	Java i JavaScript	148
12.2.	Kada JavaScript? Kada Java? Kada CGI?	149
12.3.	JavaScript i HTML	150
12.4.	Svojstva JavaScript jezika	152
12.4.1.	Varijable i automatsko pretvaranje tipova	152
12.4.2.	Pisanje vrijednosti i komentara	154
12.4.3.	Svojstva objekata i nizovi	154
12.4.4.	Kontrola tijeka programa	155
12.4.5.	Operatori i njihovi prioriteti	155
12.4.6.	Funkcije	156
12.5.	Stvaranje objekata	157
12.6.	Ugrađeni objekti	157
12.6.1.	Objekt Window	158
12.6.2.	Objekt Document	159
12.6.3.	Objekti Date i Math	160
12.7.	Obrada događaja	160
12.8.	Praktične primjene JavaScripta	163
12.8.1.	Pozdravne poruke	163
12.8.2.	Ispisivanje pomoći	164

12.8.3.	Automatsko preusmjeravanje korisnika	165
12.8.4.	Automatsko preusmjeravanje uz potvrdu	165
12.8.5.	Stvaranje upita	166
12.8.6.	Provjera starosti korisnika	167
12.9.	Kolačići (cookies)	169
13.	PHP	172
13.1.	Šta je PHP?	172
13.2.	Ubacivanje PHP naredbi u HTML	172
13.3.	Prva PHP naredba	172
13.4.	Varijable (promjenljive)	173
13.5.	Pred-definisane (već određene) promjenljive	173
13.6.	Jednostavne računske operacije	174
13.7.	Povezane riječi ('strings')	174
13.8.	Petlje	175
13.9.	Daljnje naredbe za kontrolu programa	175
13.10.	Uslovno grananje programa	177
13.12.	Proširena poređenja	180
13.13.	Predefinisane promjenljive	180
13.14.	Priključivanje datoteka	181
13.15.	Rad sa datotekama	181
13.16.	Obrada slika	184
13.17.	Povezivanje sa bazama podataka	186
13.17.	Osnove SQL-a	187
13.18.	SQL-naredbe sa PHP-om	188
13.19.	SQL naredba za brisanje (DELETE)	190
13.20.	Funkcije	190
13.21.	String-funkcije	192
13.22.	Cookies (kolačići)	192
13.23.	Promjenljive	193
13.24.	Polja (arrays) iz formulara	194
14.	ASP	195
14.1.	Kompromis između CGI i ISAPI	195
14.2.	Kako izgleda ASP stranica?	195
14.3.	Kako testirati ASP stranicu?	197
14.4.	ASP objekti	198
14.4.1.	Application objekt	198
14.4.2.	Request objekt	199
14.4.3.	Response objekt	200
14.4.4.	Server objekt	200
14.4.5.	Session objekt	200
15.	Pristup bazama podataka	202
15.1.	API funkcije za pristup bazama podataka	202
15.1.1.	ODBC	203
15.1.2.	OLE DB	203
15.1.3.	JDBC	204
15.1.4.	DAO	204

15.1.5.	ADO.....	205
15.2.	Baze podataka na web sajtovima	205
15.2.1.	Windows vs. LINUX vs. ASP vs. PHP.....	206
15.3.	Spajanje na bazu podataka	207
15.4.	Manipulacija podacima.....	209
15.5.	Korisni linkovi	210
15.6.	Razvoj PHP & MySQL aplikacije korištenjem Dreamweavera	210
15.6.1.	Priprema.....	211
15.6.2.	Tabele	211
15.6.3.	Definisanje sajta.....	212
15.6.4.	Konekcija na bazu.....	212
15.6.5.	Unos novosti	213
15.6.6.	Zaštita administrativnog dijela baze	213
15.6.7.	Kreiranje novih korisnika	214
15.6.8.	Login forma	216
15.6.9.	Ograničenje pristupa stranicama.....	217
15.6.10.	Log out.....	217
16.	.NET	218
16.1.	Šta je .NET Framework?	218
16.2.	Osnovni koncepti	218
16.2.1.	Common Language Runtime	218
16.2.2.	Skupovi-assemblies	219
16.2.3.	Opsezi imena	220
16.3.	Konfigurisanje okruženja.....	221
16.3.1.	Instalacija IIS-a.....	221
16.3.2.	Instalacija Visual Studio .NET-a	222
16.4.	Pregled procesa dinamičkih strana.....	222
16.5.	Pregled obrade Web Service-a.....	223
16.6.	Kreiranje Login.aspx strane.....	223
16.7.	Korištenje Web kontrola.....	226
16.7.1.	Dostupne Web kontrole	228
16.7.2.	Definisanje Web formi.....	228
16.7.3.	Kodiranje iza formi u ASP.NET-u.....	228
16.8.	ASP.NET	230
16.8.1.	Način pisanja koda.....	231
16.8.2.	Programska poboljšanja.....	233
17.	XML	235
17.1.	DTD – Document Type Definition	237
17.2.	Prikazivanje XML dokumenata u browseru	239
18.	Budućnost: Ajax, Gadget,.....	245
18.1.	Ajax	245
18.2.	Ajax Agent 0.1	246
18.3.	Gadget.....	251
18.3.1.	Kako rade gadgeti	252
18.3.1.	Gadget datoteke	253
18.3.2.	Kreiranje gadgeta.....	253

18.3.3.	Sistemske API-je gadgeta	256
18.3.4.	Postavke gadgeta	257
19.	Planiranje i procedura dizajna web prezentacije	261
19.1.	Profiliranje	261
19.2.	Planiranje dizajna	261
19.3.	Struktura sajta	262
19.3.1.	Slike	262
19.3.2.	Fontovi	263
19.3.3.	Navigacija	263
19.4.	Izgradnja: sklapanje sadržaja u cjelinu	264
19.4.1.	Flash ili ne?	265
19.4.2.	Veličina dokumenata	265
19.4.3.	Podloge	266
19.4.4.	GIF ili JPEG?	266
19.4.5.	Web tehnologije	267
19.5.	Postavljanje na server – web hosting	268
19.6.	Izbor domene	269
19.7.	Prijavljivanje na pretraživače	270
19.7.1.	Meta tagovi	270
19.7.2.	Prijava na pretraživače	272
19.7.3.	Trikovi za prijavljivanje na Yahoo	274
19.7.4.	Domaći pretraživači	275
19.8.	Održavanje sajta	275
20.	Literatura	278

1. Uvod

1.1. Internet i njegova organizacija

Nije nimalo lako definirati Internet. I najveći stručnjaci nisu saglasni oko toga kako precizno odrediti ovaj komunikacijski fenomen, no do danas nisu smislili ništa revolucionarno novoga. Često je pitanje "Šta je Internet?" Razlog zašto se to pitanje tako često postavlja jest upravo u tome što se niko ne može složiti oko odgovora koji bi ukratko opisao Internet. Internet se može promatrati kao skup osnovnih komunikacijskih protokola, kao fizička zbirka raznog hardvera, poput routera i drugog mrežnog hardvera, ili čak i kao zasebna ideja o povezivanju ljudi i komuniciranju. Neke češće definicije kažu da je Internet:

- Mreža koja se sastoji od manjih mreža računara koje koriste protokole iz skupine TCP/IP
- Ljudska zajednica koja razvija i koristi te mreže
- Zbirka resursa i podataka do kojih možete doći na tim mrežama

(E.Krol, User Services Working Group, Internet Engineering Task Force (IETF), FYI 20)

Sve tri definicije su sasvim dobre i ispravne. U tehničkom pogledu, Internet zaista jest mreža svih nacionalnih, akademskih, naučnih, komercijalnih i drugih svjetskih mreža. Tako povezan, Internet omogućava pojedincima iz svih tih mreža da međusobno komuniciraju i razmjenjuju informacije. Druga je definicija u sociološkom smislu također prihvatljiva: Internet osim hardvera, čine i pojedinci koji takvu globalnu mrežu primjenjuju za pristup do udaljenih podataka ili za korištenje usluga nekog drugog računara u svijetu.

1.1.1. Nastanak Interneta

Američko Ministarstvo odbrane (Department of Defense, DoD) odlučilo je uložiti novac u razvoj svoje nove mreže. On je povjeren organizaciji Advanced Research Projects Agency (ARPA). Amerikanci su osnovali tu agenciju za "napredna istraživanja" nakon što je 1957. Sovjetski Savez lansirao Sputnika, nadajući se da će im ta agencija pomoći da dostignu Ruse u svemirskoj utrci. Tako je 1968. nastala mreža ARPAnet, koja se smatra prethodnicom današnjeg Interneta. ARPAnet je bio eksperimentalni projekt. Sama mreža bila je zasnovana na tada novoj, a danas opće prihvaćenoj tehnologiji paketnog prespajanja. Smatra se da je Internet nastao onoga dana kada je Ministarstvo odbrane odlučilo svoj ARPAnet povezati s nekim drugim javnim mrežama, univerzitetima i naučnim institucijama. Stvoren je protokol kojim se paketi prenose mrežom i nazivan je IP – Internet Protocol. "Internet" u tom trenutku još nije bio ime mreže, već je taj pojam predstavljao skraćenu riječ internetworking, povezivanje mreža. Razvojem takvog protokola omogućeno je da međusobno komuniciraju različiti tipovi računara: bilo

je samo važno da se međusobno dobro "razumiju". U početku, ARPAnet je spajao samo četiri institucije (uglavnom univerzitetske), dok ih je 1972. bilo oko dvadeset.

Do velikog buma dolazi početkom osamdesetih, tačnije 1983., kada nastaju prve lokalne mreže jer je Ethernet mrežna tehnologija postala svima dostupna. To je omogućilo priključivanje mnogih novih radnih stanica na Internet, a sve su one većinom koristile tada vrlo popularni operativni sistem Berkeley BSD Unix i protokol IP za komunikaciju s drugim računarima. Skupina protokola TCP/IP razvijena je početkom osamdesetih, a 1983. prihvaćena kao standardni komunikacijski protokol na ARPAnetu. Na mrežu se nisu priključivale samo pojedine radne stanice, već i cijele lokalne mreže: svako je htio u ARPAnet. ARPAnet je tako postajao medij komuniciranja i za mnoge komercijalne organizacije, a ne samo za vojne i univerzitetske ustanove, kako je prvobitno bilo zamišljeno. Prednosti su bile sasvim očite: korisnik s jedne mreže mogao bi bez problema komunicirati s korisnikom za nekim drugim računarom, bilo gdje na ARPAnetu.

Tako su nastajale neke namjenske lokalne mreže, poput NSFnet-a. NSF (Nacional Science Foundation) je agencija američke vlade koja se bavi znanošću i ona je stvorila NSFnet s namjerom da nudi računarske usluge drugim korisnicima. Konkretno, napravljeno je nekoliko centara sa super-računarima (iako danas ono što se onda zvalo "super-računarom" stane u jedan sasvim prosječni Pentium) koji su trebali pružati usluge složenih matematičkih izračunavanja i slične usluge drugim obrazovnim ustanovama. Napravljeno je samo pet takvih centara.

1.1.2. Organizacija Interneta

Značaj NSFnet-a je u tome što je započeo s organiziranjem Internetove komunikacijske infrastrukture. Isprva, NSFnet je bio povezan s drugim preko ARPAneta. To rješenje nije bilo najbolje: najveća prepreka bila je birokracija i administracija, jer je trebalo urediti odnose između takve dvije mreže čiji su vlasnici dvije sasvim različite vladine ustanove. Stoga su u NSF-u odlučili napraviti svoju vlastitu mrežu, ali su pritom koristili ARPAnetovu IP tehnologiju za komuniciranje. Pojedine čvorove u NSFnetu spojili su zakupljenim telefonskim linijama koje su prenosile oko 7 KB podataka u sekundi. U usporedbi s današnjim propusnostima to je bilo dosta malo, ali sasvim zadovoljavajuće za zahtjeve koji su se tada postavljali pred mrežu.

Međutim, trebalo je riješiti još jedan problem: kako povezati udaljene računare na novu mrežu? Iznajmljene telefonske linije vrlo su skupe u cijelom svijetu, te si je rijetko tko mogao dopustiti plaćanje stotina ili hiljada kilometara posebne telefonske linije do nekog zamišljenog centra mreže. Zbog toga je Internet od prvih trenutaka zamišljan kao potpuno decentralizirana mreža. Problem je riješen tako da su stvorene regionalne mreže u državama i većim gradovima. Zatim su se one međusobno povezivale, čime su troškovi umrežavanja znatno pali, jer bi svaka od

mreža bila povezana samo s njom najbližom. Na taj način, svaki računar može doći do bilo kojeg drugog računara. Doduše, ne neposredno, ali ARPAnetov razvijeni protokol IP već je predviđao da drugi računari moraju znati kako isporučiti neku poruku na ispravno odredište. Do današnjih dana, Internet upravo tako radi. Mnoge druge mreže koje su tada postojale (npr. DECnet, Bitnet i slične) povezale su se s Internetom iako tada nisu koristile protokol IP. Pomoću posebnih hardversko-softverskih rješenja, nazvanih pristupnici (gateway), one su spojene na tu, sad već veliku, globalnu mrežu računara.

Godine 1986. započeo je rad na intenzivnoj komercijalizaciji mreže. Odlučeno je da se svi vojni računari isključe u zasebnu podmrežu, a zastarjeli i spori ARPAnetov sistem zamijenjen je boljim, NSFnetovim. Tako nam je ostao vrlo dobro organizirani Internet, s NSFnetom kao kičmom. (Kičmom mreže – backbone – smatra se struktura mreže na koju su spojeni svi drugi dijelovi mreže). U aprilu 1995. NSFnet je zaslužen "umirovljen", a kičmom Interneta postale su sve do tada povezane mreže. Zanimljivo je da tako Internet danas odlično radi bez dva osnovna početna dijela: ARPAneta i NSFneta. U međuvremenu je osnovana i organizacija CIX (Commercial Internet Exchange) putem koje su povezane sve tvrtke koje pružaju pristup Internetu na komercijalnoj osnovi.

1.1.3. Ko upravlja Internetom?

Internet je decentraliziran u svojoj tehničkoj organizaciji, pa tako ne postoji niti jedno vrhovno nadzorno tijelo koje bi nadziralo cijelu mrežu. Umjesto toga, poslovi oko održavanja mreže, definiranja novih pravila, standarda i komunikacijskih protokola povjereni su većem broju organizacija u cijelom svijetu.

Kada bi službeno postojao vrh, na njemu bi se našlo neprofitabilno društvo koje se naziva The Internet Society (ISOC; <http://www.isoc.org>). Njegove su zadaće promocija Interneta i komunikacijske tehnologije. U njemu su članovi koji se uglavnom bave tehničkim razvojem Interneta, planiranjem njegove budućnosti, razvojem protokola i arhitektura, te donošenjem osnovnih pravila za rad mreže.

RFC je skraćenica od Request for Comment (zahtjev za komentarom). Kada se neko dosjeti nečega što bi moglo unaprijediti Internet, neku uslugu ili neki protokol, napiše svoj prijedlog i javno ga objavi na Internetu, tražeći od drugih da mu pošalju svoje komentare. Nakon toga dopunjuje osnovni prijedlog prema pristiglim komentarima i tako se stvaraju prijedlozi novih standarda ili usluga. Svaki RFC nosi svoj broj. Pisci RFC-ova su dobrovoljci, to rade besplatno, a RFC-ovi se mogu slobodno i neograničeno koristiti. Dakle, ne radi se o službenoj dokumentaciji ili standardima, ali ako ste zaista zainteresirani za proučavanje mreže i Interneta, RFC-ovi su golem izvor korisnih informacija. Mnogi od njih namijenjeni su i početnicima, pa naprimjer, sadrže rječnike, objašnjenja pojmova ili upute za rad s pojedinim uslugama.

FYI je skraćenica of For Your Information (za vašu informaciju). Radi se o sličnim dokumentima kao RFC, samo što su općenitije prirode. Prvenstveno su namijenjeni početnicima jer sadrže općenite informacije o Internetu. Svi FYI-ovi su također objavljeni i kao RFC-ovi, ali ako tražite sasvim elementarne stvari o Internetu, prvo pogledajte među FYI-ovima. Poput RFC-ova i FYI-ovi se mogu slobodno distribuirati i koristiti.

Ako znate koristiti FTP ili pristupati FTP serverima preko Weba, možete na taj način doći do RFC-ova i FYI-ova. Naprimjer, oni se nalaze na FTP serverima `ftp.carnet.hr`, `ds.internic.net`, `src.doc.ic.ac.uk`, `wuarchive.wustl.edu`, i mnogim drugim. Većina RFC-ova pisana je u čistom tekstualnom formatu (ime datoteke u obliku `rfc1444`), dok su rijetki raspoloživi isključivo u formatu PostScript (datoteke poput `rfc1444.ps`).

Godine 1983. osnovan je Internet Activities Board (IAB). Njegova je dužnost, kako kažu u njegovom statutu, "da bude nezavisni odbor istraživača i profesionalaca čiji je tehnički interres rad i razvoj Internet sistema", a bavi se osmišljavanjem tehničkih pojedinosti Interneta. IAB ima dva odjela: Internet Research Task Force (IRTF) i Internet Engineering Task Force (IETF). IETF se dalje bavi mnogim drugim aspektima Interneta, poput određivanja načina komunikacije među mrežama, prenošenja podataka mrežom, pisanja dokumentacije i tako dalje. Radne grupe sastavljaju izvještaje o pojedinim temama i javno ih objavljuju na Internetu u obliku tekstualnih datoteka. Važno je napomenuti da bilo ko može prisustvovati sastancima IETF-a, jer je to sasvim otvorena organizacija.

U razvoju Interneta sudjeluju i neke druge organizacije, poput InterNICa (Internet Network Information Center). Njegova je zadaća, naprimjer, da vašoj tvrtki dodijele mrežnu adresu (ime domene) koju ćete koristiti na Internetu za svoje računare.

kako se Internet finansira? IAB, ISOC, IRTF, IETF i InterNIC su sasvim neprofitabilne organizacije. Organizacija (tvrtka) A sama finansira svoju vlastitu mrežu, a organizacija (tvrtka) B svoju. Zatim se te dvije organizacije same dogovore kako će povezati svoje dvije mreže, i troškove te veze također same plate. Upravo tako je stvoren cijeli Internet.

Korištenje usluga drugih računara i mreža se ne naplaćuje: one se smatraju javnim dobrom. Da pojasnimo: ako su organizacije A i B povezale svoju mrežu, a u međuvremenu se organizacija C povezala s mrežom organizacije B, to ne znači da organizacija B ima pravo naplaćivati organizacijama A i C, koje koriste njenu mrežnu infrastrukturu za međusobnu komunikaciju, troškove te komunikacije. Kada pošaljete poruku preko Interneta, vi zapravo nikako ne možete znati kojim će putem i preko čije mreže ona proći na putu do svog odredišta.

Pojavili su se komercijalni davatelji usluga, koji se jednostavno bave poslom umrežavanja i naplaćuju korištenje svoje mrežne infrastrukture za pristup Internetu drugim tvrtkama i pojedincima koji žele doći do Interneta.

2. Klijentsko-serverski model

Kovanica client/server jedna je od onih koje se najčešće čuju posljednjih godina. Različiti pojedinci i različite tvrtke tumače ovaj pojam na sasvim različite načine kada pričaju o svojoj mreži. Zamislite bilo kakvu mrežu računara. Mrežu valjda ne trebamo definirati: čuli ste za takve stvari? Valjda jeste, nešto su jako popularne u zadnje vrijeme... Svaki računar koji pruža neku vrstu usluge ili koji nudi neke svoje resurse na korištenje (npr. ispis na priključenom pisaču ili korištenje podataka s njegovog tvrdog diska) naziva se serverom. S druge strane, svaki računar koji koristi usluge servera naziva se klijentom.

U općenitom slučaju, serverom se naziva bilo koji računar koji daje neke svoje usluge na korištenje drugim računarima. Jednako tako, klijent je bilo koji računar koji "posuđuje" i koristi uslugu ili neki resurs drugog računara. Zbog toga svaki računar može biti ili klijent ili server, i jedno i drugo, ili ništa od toga.

Druga važna stvar je način komunikacije između klijenta i servera. To je posebno važno, jer se svi komunikacijski protokoli, uključujući i TCP/IP i HTTP, sporazumijevaju upravo na taj način. Kada klijent želi zatražiti pristup nekoj usluzi ili resursu servera, šalje mu zahtjev za pristup usluzi. Nakon toga server šalje odgovor klijentu i omogućava mu korištenje zatražene usluge. Na taj se način zahtjevi i odgovori razmjenjuju sve do završetka komunikacije.

Klijentsko-serverski model nije danas tek tako prihvaćen kao jedno od najboljih rješenja za realizaciju računarske mreže. Međutim, postoje mnoge njegove prednosti. Za početak, taj je model mreže vrlo ekonomičan, jer se veliki računarski kapaciteti i kapaciteti za pohranjivanje podataka traže samo na strani servera. Osim toga, pokazao se dosta praktičnim u svakodnevnom radu, jer je moguće rasporediti opterećenje jednog računara (npr. glavnog servera u tvrtki) na više njih.

Klijentski program je jednostavno program kojim pristupate nekoj određenoj usluzi na serverski. Naprimjer, pretraživač za World Wide Web je primjer klijentskog programa. Serverski program izvršava se na serverskom računaru i opslužuje zahtjeve klijenata. Bitno je jedino da klijent i server razmjenjuju podatke po istim pravilima; informatičari bi rekli: koristeći se istim protokolom.

2.1. Mrežne adrese

Na velikoj mreži kakva je Internet, najveći je problem bio osmisлити sistem imenovanja računara, koji će koristiti svi protokoli, usluge i – na kraju krajeva – sami korisnici da bi komunicirali s nekim računarom. Sistem IP adresa koji je razvijen imao je mnogo prednosti nad nekim drugim tada predloženim sistemima, pa je prihvaćen za korištenje na ARPAnetu i, kasnije, NSFnetu i Internetu.

Svaki računar na mreži dobiva adresu u obliku četiri broja odvojena tačkama – to je ono što ćemo zvati IP adresom. Kako računari najbolje rade s binarnim brojem

sistemom, adrese su zapravo predstavljene s 32 bita. Naprimjer, mrežna adresa nekog računara može biti

10100001 00110101 00000011 00101000

Ljudima takav format, iz sasvim očitih razloga, nije baš prihvatljiv, pa se takav 32-bitni niz binarnih cifara pretvara u četiri dekadski broja koja su odvojena tačkom. Dekadski adresa dobiva se tako da se svaka grupa od 8 bitova zasebno pretvori u svoj dekadski brojni ekvivalent. Adresa gornjeg računara je tako

161.53.3.40

što je prihvatljivije za pamćenje. Budući da dekadski vrijednost svakih 8 bitova može biti između 0 i 255, može se izračunati da bi se na taj način moglo adresirati nešto preko četiri milijarde računara.

Numerička adresa ima dva važna dijela, koja nisu baš očita ako je promatrate samo kao niz od četiri broja. Međutim, važno je znati da dio adrese predstavlja broj mreže na koju je priključen računar, a da drugi dio adrese predstavlja broj računara u toj mreži. Naravno, postavlja se pitanje "šta je šta" ako imate adresu od četiri broja: postoje barem tri mogućnosti.

Upravo se sve te tri mogućnosti koriste. Već smo rekli da je Internet velika mreža manjih, lokalnih mreža koje se nalaze u tvrtkama ili organizacijama. Kada se razmišljalo o načinu izvedbe IP adresiranja, pretpostavilo se da će postojati:

- mali broj samostalnih mreža s jako velikim brojem računara
- određen broj mreža koje imaju srednji broj računara
- vrlo veliki broj mreža koje povezuju manji broj računara.

Zbog toga su osmišljene klase adresa. Ako opći format adrese napišemo kao

aaa.bbb.ccc.ddd

Klasa A su one adrese kojima je dio aaa adresa mreže, a ostali brojevi adresa računara. Tako može postojati samo nešto više od 120 takvih mreža, ali svaka može imati preko 16 miliona računara (jer posljednja tri broja označavaju samo računar). U klasi B, dio aaa.bbb predstavlja adresu mreže. Budući da su brojevi do 126 već zauzeti za klasu A, u klasi B prvi broj je veći od 128. Tako može postojati 16 hiljada mreža u toj klasi, a svaka od njih ima do 65 hiljada računara (jer računar predstavljaju posljednja dva broja). U klasi C, prva tri broja, aaa.bbb.ccc, označavaju mrežu i zato je moguće imati oko 1 miliona mreža u toj klasi. Broj aaa mora biti veći od 192. Samo zadnji broj, ddd, koristi se za oznaku računara, pa mreža klase C može u sebi imati samo malo više od 250 računara.

U prvom stupcu tabele 1 je oznaka klase, u drugom i trećem prvi brojevi iz IP adrese između kojih se nalazi ta klasa, a posljednja dva stupca pokazuju koliko mreža može biti u toj klasi i koliko svaka od tih mreža može imati računara.

Tabela 1 Raspoložive IP adrese

Klasa	Prvi broj: od	Prvi broj: do	Broj mreža u klasi	Broj računara u mreži
A	1	126	126	16.777.214
B	128	191	16.384	64.534
C	191	223	2.097.152	254

2.1.2. Rezervirane IP adrese

Neke IP adrese ne smiju se koristiti za označavanje broja mreže ili računara. Broj računara ili mreže ne smije biti 0, zato jer nula označava "ovu mrežu". Naprimjer, ako napišete adresu 161.53.0.0, to označava mrežu čija je adresa 161.53.

Broj 255 koristi se za slanje nekih podataka na sva računara. Naprimjer, adresa 161.53.255.255 odnosi se na sve računare u mreži 161.53. Ako pošaljete neke podatke na tu adresu, oni će doći do svih računara u mreži 161.53.

Oznaka mreže 127 također je rezervirana (nema je u tablici 1). Adrese koje počinju sa 127 imaju posebnu primjenu; radi se o tzv. loopback adresama (povratnim adresama) koje se koriste samo za testiranje rada mreže. Mrežne poruke i paketi koji su adresirani na mrežu 127 nikada neće otići dalje u mrežu, već će se vratiti nazad. Posebno je zanimljiva adresa 127. 0. 0. 1, jer ona uvijek označava taj isti računar.

Zadnji broj u IP adresi ne smije biti 0 ili 255 (osim u gornja dva posebna slučaja). Prvi broj također ne smije biti veći od 223; brojevi 224 i 225 rezervirani su za neke posebne slučajeve i najčešće se uopće ne susreću.

Ova su ograničenja ujedno razlog zašto su brojevi mogućih računara i mreža u tablici 1 takvi kakvi jesu.

2.1.3. Kako dobiti IP adresu

Ako ste "običan" korisnik koji samo pristupa Internetu preko modema, nije potrebno imati posebnu IP adresu. Organizacija koja je zadužena za dodjeljivanje imena domena jest već spomenuti InterNIC. Adresu klase A ne možete dobiti jer su već odavno razgrabljene: takve su adrese dobile tvrtke i organizacije koje imaju jako mnogo umreženih računara, poput IBM-a MIT-a, i sličnih. Postoji tek malo broj klasa B na raspolaganju, ali ih je teško dobiti jer su rezervirane samo za preostale velike "manje" tvrtke i organizacije i eventualno koju novu državicu. Dakle, možete bez problema jedino dobiti adresu u C klasi.

Najveći međunarodni lanac trgovina u kojima se prodaju igračke poznat je po maštovitom imenu Toys R Us. Početkom 1996., ta je tvrtka imala problema s pravima na svoje vlastito ime. Nakon što jednom registrirate ime svoje tvrtke kod

odgovarajućih organizacija, imate sva prava na njeno korištenje. Slično je i na Internetu: nakon što InterNIC prihvati ime vaše domene, sva su prava na njeno korištenje samo vaša.

Time se pokušao okoristiti jedan desetogodišnji dječak iz Amerike, koji je registrirao domenu toysrus.com kod InterNIC-a prije no što su to učinili iz same tvrtke. Zatim je tražio od ljudi iz tvrtke Toys R Us da mu u zamjenu za ime domene daju jedan bicikl i (jedan) računar. Predsjednik tvrtke smatrao je prijedlog simpatičnim i povoljnim, pa je odlučio "otkupiti" ime domene.

Međutim, cijela ta ideja s registriranjem imena domena poznatih tvrtki nije bila dječakova, nego njegovog pohlepnog ujaka, koji je registrirao još nekoliko imena domena poznatih tvrtki prije no što su one to stigle učiniti. Ujak je zabrljao cijelu stvar jer se nije mogao odlučiti koje računar želi dobiti od Toys R US, pa su za inat neki direktori iz tvrtke odlučili da nema šanse niti da dobiju bicikl. InterNEC je riješio problem tako da je domenu dao onome kome i pripada – tvrtki Toys R Us – a dječacić i ujak ostali su i bez računara i bez bicikla.

Da biste od InterNIC-a dobili svoju grupu IP adresa i registrirali je kao domenu, morate popuniti poseban formular. Do njega možete doći pomoću FTP-a na adresu <ftp.internic.net>, pomoću World Wide Web-a na adresi <http://www.rs.internic.net> ili elektroničkom poštom s adrese hostmaster@internic.net. Odgovor na elektronički zahtjev dobit ćete u roku od dva dana.

2.2. Imena računara

IP adrese u dekadskom obliku (četiri broja s tačkama) u svakom slučaju ljudi lakše pamte od binarnih adresa, ali to još uvijek nije to. Zato su mrežama i računarima nadjevena sasvim normalna imena. Ime se također sastoji od nekoliko riječi, koje su opet odvojene tačkama. Naprimjer, pogledajte adresu:

`pi.unze.ba`

Posljednja oznaka, ba, predstavlja oznaku države u kojoj se računar nalazi. Oznaka prije nje, u našem slučaju unze, ime je organizacije u kojoj se računar nalazi. Ime organizacije i oznaka države zajedno se nazivaju domena. Konačno, pi je ime računara koje se nalazi u domeni unze.ba. Adresa računara napisana u tom obliku naziva se Fully Qualified Domain Name (FQDN). Gotovo uvijek kada se radi o Webu, susretat ćete se upravo s ovakvim, a ne s IP adresama.

Broj oznaka u ovakvoj adresi nije ograničen, ali ih je obično između tri i pet. Naprimjer, moguća je i adresa poput:

`www.pf.unze.ba`

Ovdje je www ime računara, a sve ostalo predstavlja domenu. Kao što vidite, domena ima tri dijela: osim zemlje i imena institucije (unze), u njoj se nalazi i oznaka odjela (pf) te institucije gdje se računar nalazi.

Zadnja oznaka u ovakvom imenu najčešće je oznaka države (ba za BiH, hr za Hrvatsku, si za Sloveniju, de za Njemačku, uk za Veliku Britaniju, fr za Francusku, au za Australiju itd). No, u slučaju računara koji se nalaze u Sjedinjenim Američkim Državama najčešće nema oznake us, već na tom mjestu stoji oznaka tipa mreže u kojoj se nalazi računar. Naprimjer, edu označava obrazovanu ustanovu, com komercijalnu organizaciju, gov vladinu ustanovu, mil vojno računar, org neku međunarodnu organizaciju, net neku mrežu koja se prostire na većem području i tako dalje.

Ovakav način imenovanja računara ostavlja mogućnost dodavanja aliasa, odnosno alternativnih zamjenskih imena za svaki računar. Naprimjer, ako računar pi.unze.ba služi kao Web server, on obično dobiva dodatno ime www.unze.ba, ili kada bi na njemu bila FTP arhiva s programima, ftp.unze.ba. Na taj je način olakšan pristup korisnicima pojedinih usluga. Numerička IP adresa i dalje ostaje ista, ali računar ima nekoliko FQDN imena.

2.3. Maskiranje

Podmreže dijele jednu veću mrežu na više manjih. Ako ste mrežni administrator, to vam može ili ne treba koristiti. Naprimjer, ako vaša tvrtka dobije IP adresu u klasi C koja dopušta samo 254 računara, podmreže van ne trebaju: Ali, ako se radi o nekoj većoj mreži, poželjno je njeno dijeljenje u više podmreža, jer se manje mreže lakše održavaju, mrežni promet smanjuje, a sigurnost mreže poboljšava. Recimo, mreža 161.53 predstavlja Hrvatsku. Podmreža 2 je zagrebačko SRCE (adresa računara 181.53.2.x), a podmreža 128 je Institut Ruđer Bošković (adresa računara 161.53.128.x). Loše izvedeno maskiranje najčešće ima za posljedicu potpuni raspad i nefunkcionalnost mreže, jer se poruke gube, putuju krivim putevima, prođu cijeli svijet da bi se zatim vratile na susjedni računar i tako dalje.

Kada administrator mreže upotrijebi neke adrese za označavanje podmreže mora na neki način reći protokolu TCP/IP kako da razlikuje jednu podmrežu od druge. Za to se koristi maskiranje. Podmrežna maska je, poput IP adrese, također 32-bitni binarni broj, u kojemu se bitovi za adresu mreže (zajedno s podmrežom) postavljaju na 1, a bitovi za adresu računara na 0. To zahtijeva malo opreznog binarnog računanja i planiranja.

Naprimjer, ako vaše računar pretražuje adresu klase B, 161.53.3.20 uz podmrežnu masku 255.255.0, tada dio "161.53.3" smatra mrežom (podmrežom), a dio "20" adresom računara.

Podmrežna maska mora biti ista za sva računara u podmreži. Inače, mreža ima određenu podmrežnu masku čak i ako nema podmreža. Naprimjer, maska za adrese iz klase A je 255.0.0.0, za klasu B 255.255.0.0, a za klasu C 255.255.0. Da ste se poslužili onime što znate o tome što označava mreža a što računar u pojedinim klasama adresa i činjenicom da 8 bitova koji su postavljeni na jedinicu tvore dekadski broj 255 i sami biste došli do ovoga.

2.4. DNS serveri: pretvaranje FQDN u IP adresu

Vidite da jedan računar ima IP adresu i FQDN adresu. Jedini način adresiranja koji je pogodan hardveru i softveru su IP adrese; FQDN adresiranje izmišljeno je samo zbog ljudi. Kako Internet "zna" da je računar kojemu želite pristupiti, recimo jagor.srce.hr, označeno IP adresom 161.53.2.120?

U ranim danima ARPAneta, problem se rješavao trivijalno, jednom običnom tekstualnom datotekom (koja se naziva hosts) koja se održavala na mreži. U njoj je pisalo, recimo, u dva stupca:

161.53.2.120	jagor.srce.hr
115.214.4.44	neko.računar.edu

Razlog zašto se to više ne koristi je sasvim očit: da bi računari mogli ispravno pretvarati jedan tip adrese u drugi, moraju uvijek imati posljednju, najnoviju verziju takve datoteke. Broj računara na Internetu davno je prešao broj nakon kojega je održavanje takvog popisa ne samo nepraktično, već je i organizacijski i fizički nemoguće prenositi takve popise mrežom. Ipak, hosts datoteke se i dalje vrlo često koriste u lokalnim mrežama za definiranje IP adresa i pripadajućih im FQDN imena računara.

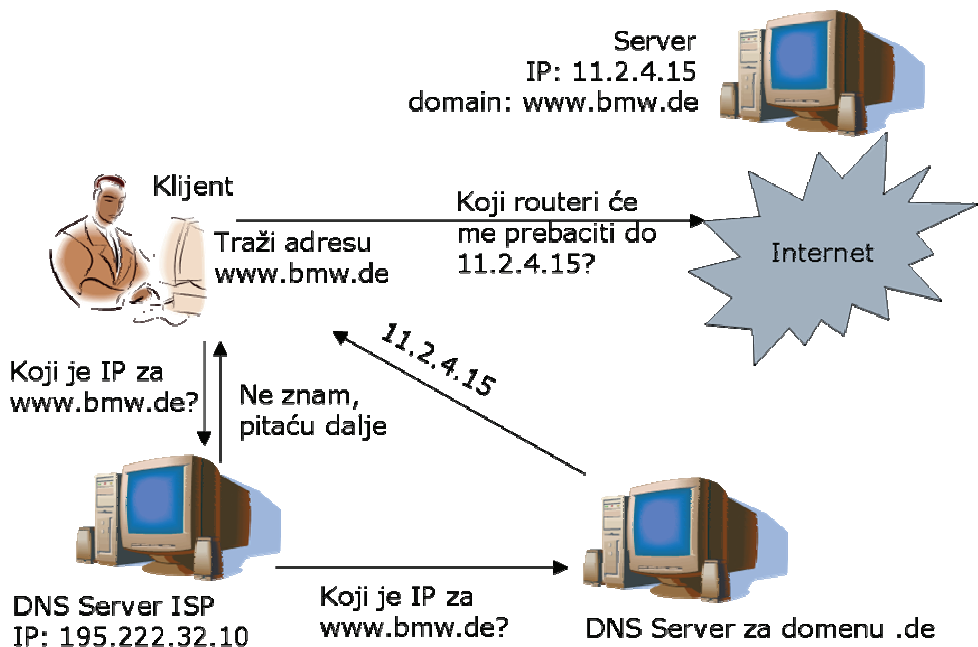
Na Internetu danas taj posao uglavnom obavljaju DNS-ovi. DNS je skraćenica od Domain Name Server, server za imenovanje domene. (Neki kažu da je S iz skraćenice i Service ili System). To je računar koji zna pretvoriti IP adresu u FQDN ime računara za računare iz svoje domene. Svaki od DNS servera brine se o održavanju tablice s IP i FQDN adresama za svoju domenu, a za ostale domene zna kojem drugom DNS serveru treba proslijediti zahtjev. Na primjer, ako neki računar želi znati IP adresu nekog drugog računara i pošalje zahtjev DNS serveru, moguća su tri rezultata:

- DNS server odmah zna odgovor, jer ga pronalazi u svojoj bazi podataka o računarima te domene.
- DNS server ne zna koja je IP adresa i mora pitati drugog servera.
- DNS server zna odgovor zato što je neko već ranije tražio adresu tog računara, pa ju je on sačuvao nakon što ju je saznao od drugog servera.

Tako razlikujemo primarne i sekundarne DNS servere. Primarni su odgovorni za cijelu svoju domenu. Često se primarni serveri rasterećuju tako da dio posla prepustaju sekundarnim DNS serverima. Sekundarni DNS server, naprimjer, može obavljati posao oko pretvaranja adresa za mrežu neke tvrtke, a primarni je na višoj razini i zna da informacije o računarima u toj tvrtki može dobiti od tog sekundarnog servera. Primarni DNS poslužitelj predstavlja "krajnju istinu": sve informacije i podaci koji se nalaze na njemu odgovaraju stvarnom stanju mreže.

Najveća razlika između primarnih i sekundarnih DNS servera je u tome gdje pronalaze svoje informacije o adresama. Primarni ih imaju zapisane u bazama podataka, a sekundarni ih dobivaju od primarnih, ili od drugih sekundarnih servera.

I obični računari mogu se smatrati malim DNS serverima, iako ne u pravom smislu te riječi. Naime, nakon što računar pita DNS servera (primarnog ili sekundarnog) za IP adresu nekog drugog računara, on će je upamtiti u nekoj svojoj datoteci kako slijedeći put ne bi moralo pitati DNS za adresu. Šta ako se promijeni IP adresa nekog računara i podatak zapisan na lokalnom računaru postane pogrešan? Računar se želi povezati s računarom kojemu je promijenjena IP adresa, a on zna samo staru. Zbog toga najčešće te keširane adrese imaju "rok trajanja", nakon čega se brišu s običnog računara, pa ono ponovo mora kontaktirati DNS server da bi dobilo adresu.



Slika 1. Funkcionisanje DNS-a

2.4.1. Obrnuto DNS mapiranje: pretvaranje IP u FQDN adresu

Evo i suprotnog slučaja. Zna IP adresu, ali željeli biste znati i ime računara. Naprimjer, Web server može poželjeti saznati ime računara s kojeg se javljate pomoću vaše IP adrese koju je dobilo prilikom uspostavljanja veze s vašim klijentom. DNS zato može ustanoviti i ime nekog računara, ako zna njegovu IP adresu. Taj se proces naziva reverse lookup ili obrnuto mapiranje adresa, a ima i drugih naziva.

Da bi se podržalo obrnuto mapiranje, na Internetu postoji posebna domena, `in-addr.arpa`, koja se u prikazanom stablu adresa nalazi na vrhu. Čvorovi u imenu te domene su zapravo IP adrese. Tako domena `in-addr.arpa` ima 256 poddomena, koje odgovaraju prvom broju IP adrese. Svaka od tih poddomena dalje opet može imati 256 poddomena koje odgovaraju drugom broju IP adrese, a svaka od njih opet ima sljedećih 256 poddomena za treći broj IP adrese. Konačno, one se dijele na još 256 poddomena za četvrti broj iz IP adrese. U toj krajnjoj grani nalazi se FQDN ime računara.

Pogledajmo kako to radi: pretpostavimo da želite saznati koje se računar koje iza adrese `161.63.2.120`. DNS kreće od vrha stabla `in-addr.arpa`. On počinje pretraživati stablo od domene `161.in-addr.arpa`, zatim ide dalje u dubinu do `53.161.in-addr.arpa`, i tako sve dok ne dođe do konačnih `120.2.53.161.in-addr.arpa`. U toj grani stabla zapisana je adresa `jagor.srce.hr`, što znači da je navedenoj IP adresi odgovarao upravo taj računar. Ovakva stabla adresa održavaju se na DNS serveru paralelno s bazama podataka koje pretvaraju FQDN adresu u IP adresu, pa su tako moguće pretvorbe adresa u oba smjera.

3. Osnovne mrežne usluge

Povrh TCP/IP protokola, izvršavaju se mnoge mrežne usluge i njihovi protokoli više razine koji su najzanimljiviji samim korisnicima: prenosi se elektronička pošta, prenose se datoteke, pristupa se bazama podataka ili World Wide Web stranicama. Za svaku od tih usluga ponovno se brine poseban protokol komunikacije. No, taj protokol na višoj razini može razgovarati s drugim računarom koje koristi isti protokol samo ako se komunikacija odvija preko osnovnog protokola, TCP/IP-a. Kao što vidite TCP/IP je i dalje osnova svega, jer je on odgovoran za prenošenje svih podataka mrežom: ako smo TCP/IP nazvali "zajedničkim jezikom", tada su mrežne usluge jednostavno razgovori o različitim temama. Ponovno primjer iz stvarnog života: nuklearni fizičar koji govori engleski i nuklearni fizičar koji govori kineski neće se razumjeti, mada razgovaraju o istoj temi (nuklearna fizika), ali ima osnovni "protokol" - jezik - nije zajednički i razumljiv obojici.

Značaj World Wide Weba je upravo u tome što ujedinjuje sve mrežne usluge: za pristup pojedinim uslugama i dalje se koriste različiti, već ustanovljeni protokoli, no program kojim pristupate World Wide Webu "zna" da treba razgovarati s WWW serverom na jedan, a s nekim drugim serverom na drugi "način".

3.1. Portovi

U komunikaciji sa korisnicima Interneta može se čuti da je nečiji "Web server na nestandardnom portu 8080", da je on "igrao interaktivne igre na portu 7755" ili "razgovarao s drugim korisnicima Interneta na portu 6667".

Portovi su, međutim, sasvim jednostavna stvar. Kad smo govorili o klijentsko-serverskom modelu mreže, rekli smo da jedan računar može istovremeno biti server za više usluga. Naprimjer, jedan računar na Internetu može istovremeno služiti i kao arhiva korisnog softvera i kao server za World Wide Web. Iako vam se prvi na pogled čini da ovdje nema nikakvih problema i prepreka da se takvo nešto ostvari, zapravo nije tako.

Klijentski programi - programi pomoću kojih se pristupa serveru za određenu uslugu - koriste različite protokole (komunikacijske "jezike") za obraćanje serveru. Klijentski program za prenošenje datoteke razgovara s serverom na jedan, a klijentski program za prikaz Web stranice razgovara s serverom na drugi način. Problem je, dakle, u tome što je sasvim opravdana potreba da jedan računar bude server za više stvari odjednom. Kada dva programa pokušaju komunicirati preko Interneta, očito nije sasvim dovoljno da znaju samo IP adresu računara na kojemu se izvršava onaj drugi program. Kako klijent za World Wide Web može znati da će se, ako pristupi računar s IP adresom 161.53.2.40, povezati upravo s serverom za Web, a ne s nečim sasvim desetim?

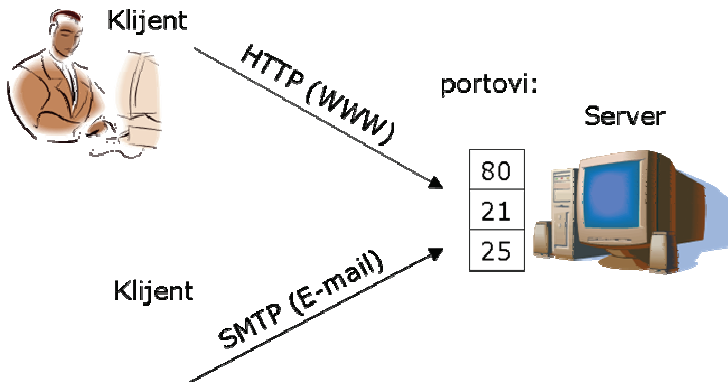
Tako je nastao i termin port. Radi se jednostavno o nekoj identifikacijskoj oznaci pojedine usluge. Ili, još slikovitije, radi se o "vratima" neke usluge na pojedinom serveru. Port se označava brojem između 0 i 65535. Neke osnovne i uobičajene usluge na Internetu imaju brojeve do 1000, dok se druge koriste na pojedinim operativnim sistemima ili za neke ne toliko standardizirane usluge.

Evo i kako stvar radi: kada se pokrene serverski program za neku uslugu, on za sebe rezervira neki broj porta (pogledajte tablicu 2) i čeka da u pristignu zahtjevi upravo na taj port. Kada tome serveru pristignu neki paketi s podacima od nekog klijenta, a zatim protokol za komunikaciju – TCP/IP – prvo pogleda koji je broj porta u tim paketima, a zatim proslijedi zahtjev odgovarajućem serveru, pod uvjetom da svi oni rade na različitim portovima. Klijentski programi znaju na kojem se od portova izvršava serverski program i tako mu znaju pristupiti. Stvar je riješena, lako i jednostavno.

Tabela 2 Osnovni portovi i usluge koje se na njima izvršavaju

Broj porta	Osnovni protokol	Aplikacijski protokol ili usluga	Opis usluga
13	TCP/UDP	daytime	Daje trenutno lokalno vrijeme
17	TCP	qotd	Quote of the Day
21	TCP	ftp	Prenošenje datoteka
23	TCP	telnet	Terminalske povezivanje
25	TCP	SMTP	Prenošenje elektroničke pošte
37	TCP/UDP	timeserver	Time server
42	TCP	nameserver	Name server
43	TCP	whois	Dobivanje podataka o korisniku/računaru
53	TCP/UDP	DNS	Domain Name Server
69	UDP	tftp	Trivial File Transfer Protocol
70	TCP	gopher	Gopherski serveri
79	TCP	finger	Dobivanje podataka o korisniku/računaru
80	TCP	HTTP	World Wide Web serveri
110	TCP	POP3	Post-Office-Protocol (elektronička pošta)
119	TCP	NNTP	Pristup novinskim grupama
210	TCP	WAIS	Pristup bazama podataka
517	UDP	talk	Razgovor s drugim korisnikom
518	UDP	ntalk	Nova verzija talka
540	TCP	UUCP	Zastarjeli protokol Unix-to-Unix Copy

Pogledajte sliku 2. Na toj slici, jedan je računar istovremeno server za tri usluge: World Wide Web (HTTP), prenošenje datoteka (FTP) i E.mail (SMTP). Ti se serverski programi izvršavaju na portovima 80, 21 i 25. Ako klijent želi pristupiti Web serveru, upućuje svoj zahtjev na port 80. Istovremeno, neki sasvim drugi klijent na sasvim drugom kraju svijeta može komunicirati s portom 25 na istom serverskom računar i tako pristupiti serverskom programu za E-mail. Više serverskih programa na jednom računar tako ne smetaju jedan drugome i poslužuju svaki svoje klijente.



Slika 2. Primjer komunikacije klijenta i servera, gdje klijenti pristupaju različitim portovima na istom serveru kako bi koristili različite usluge

Pogledajmo još na trenutak tablicu 2. U prvom se stupcu nalaze brojevi portova nekih važnijih usluga na Internetu. U drugom je stupcu tip protokola pomoću kojega se pristupa pojedinom portu. Kao što vidite, ovdje stoji ili oznaka TCP, ili oznaka UDP, ili u nekim slučajevima obje. Ta se oznaka zapravo odnosi na osnovni komunikacijski protokol niske razine pomoću kojega se pristupa tom portu. Skupina komunikacijskih protokola TCP/IP zapravo sadrži dva takva protokola niske razine, TCP i UDP. Ta dva protokola samo prenose podatke mrežom i oni mogu bez problema dijeliti jedan isti broj porta, zato što je za uspješno uspostavljanje veze sa serverskim programom potrebno osim IP adrese i broja porta znati i tip osnovnog protokola pomoću kojeg mu se pristupa.

U trećem stupcu tabele 2 nalazi se ime aplikacijskog protokola (protokola više razine) koji se koristi za komunikaciju s pojedinom uslugom. Konačno, u četvrtom je stupcu samo kratak opis usluge koja se izvršava na navedenom protu.

3.2. Rad na udaljenom računar (Telnet)

Rad na udaljenom računar je jedna od osnovnih Internet usluga. Program kojim se to omogućuje standardni je dio Unixa i naziva se Telnet. Telnet omogućava da se, sa računara na kojemu radite, povežete s računarom u susjednoj sobi ili na drugom kraju svijeta i koristite njegove usluge. Vaše računar tada postaje terminal, koji na svom ekranu samo prikazuje rezultate izvršavanja programa na udaljenom računar, a sve što otipkate također odmah prosljeđuje udaljenom računar. Da biste se povezali s udaljenim računarom, trebate samo znati njegovu adresu i pokrenuti program za telnet sa imenom računara na koji se spajate.

3.3. HTML

HTML (Hypertext Markup Language) je alat kojim se stvaraju hipertekstualni dokumenti za upotrebu na Webu. Nastao je iz općenitijeg jezika SGML (Standard

Generalized Markup Language), koji se koristi za stvaranje dokumenata prenosivih među različitim platformama. Pomoću HTML-a opisuje se grafički izgled Web dokumenta, uključujući način ispisa teksta, položaje slika, tabela i drugih elemenata.

Hipertekstualni dokument stvara se ubacivanjem HTML oznaka u običnu tekstualnu ASCII datoteku. Naprimjer, masna slova označavaju se oznakom , a kosa oznakom <I>. Na sličan se način među tekst uključuju slike, tablice, pobrojane liste ili označavaju veze. Takva ASCII datoteka nalazi se na disku servera; nakon što je prime, pretraživači za World Wide Web tumače ubačene oznake (kodove, tagove), formatiraju dokument na odgovarajući način i prikazuju ga na ekranu. Budući da se radi o uobičajenim ASCII datotekama, one su čitljive na svim računarima, čime je postignuta univerzalnost i prenosivost Weba.

Jezik HTML samo opisuje izgled stranice. U njemu piše, naprimjer, "tu i tu ide slika iz te i te datoteke", ali sama HTML datoteka ne sadrži sliku, već se ona posebno prenosi od servera do pretraživačkog programa. Dakle, kad kažemo "WWW dokument", ne radi se o jednoj datoteci, već o konačnom prikazu dobivenom na ekranu, koji je u pravilu sačinjen od jedne osnovne tekstualne HTML datoteke koja opisuje izgled stranice i od niza grafičkih datoteka.

HTML se stalno razvija: službeno su prihvaćene verzije 0.9, 1.0 i 2.0, dok HTML 3 nikada nije službeno prihvaćen, iako ga vodeći pretraživači djelimično ili u potpunosti podržavaju. Službeni standardi jezika propisuju sve naredbe koje se smiju koristiti u pojedinoj verziji HTML-a, no kako su reakcije tržišta obično daleko brže od reakcije onih koji propisuju standarde, mnogi proizvođači pretraživačkih programa za WWW, poput Netscapea, Microsofta i NCSA, samostalno su dodavali nove naredbe jeziku. Danas se uglavnom vodi bitka oko toga tko će što podržavati od takvih proširenja. Kako su Netscape i Microsoft vodeće tvrtke po broju nedavnih dodataka jezika (dodana je podrška okvirima, slikovnim mapama, tabelama, izradi stilskih predložaka za stranicu, radu s tipovima pisama i mnogim drugim stvarima), sve se na kraju svodi na to da drugi proizvođači samo prihvaćaju njihova proširenja jezika. Pretraživački programi koji ne podržavaju njihova proširenja praktički su odavno ispali iz igre, uključivši i Mosaic.

3.3.1. Historija HTML-a

Upotrebljavajući SGML kao predložak, u ljeto 1991. godine napravljene su prve specifikacije HTML-a (HyperText Markup Language). Za razliku od SGML-a HTML je bio fiksiran programski jezik sa tačno definisanim setom naredbi, u obliku HTML tagova, kojih se svi HTML dokumenti moraju pridržavati.

Iako je sazrijevao i razvijao se kroz period od svoga nastanka do danas, omogućujući manipulaciju i korištenje slika u HTML dokumentima i integraciju sa ostalim tehnologijama, HTML je prvobitno bio namijenjen samo formatiranju

teksta, što možete vidjeti i iz njegovog punog imena. Napravljen je kako bi omogućio developerima da definišu kako će se tekst u njihovim dokumentima prikazati na korisničkom računaru, te da omogući korisniku pristup ostalim dokumentima povezanih hiperlinkovima. Mnogo tehnologija je razvijeno kako bi popunile rupe u područjima gdje HTML ne zalazi, jednostavno zbog toga što nije bio namijenjen i za šta više osim formatiranja teksta nezavisno od operativnog sistema. HTML je prošao nekoliko revizija i standardizacija.

HTML 1.0

Ovo je prva verzija HTML jezika. Definisana je krajem 1990. godine i tada je samo NeXT operativni sistem mogao čitati HTML dokumente. "Otac" HTML-a, Tim Berners-Lee, primijetio je nedostatak tekstualnog programskog jezika neovisnog o operativnom sistemu za upotrebu na Internetu, tako da je specifikacije HTML-a 1.0 postavio na Internet. Prije objavljivanja specifikacija, Berners-Lee je napravio reviziju HTML-a kako bi postigao neovisnost o operativnom sistemu kroz softver za kompajliranje, danas poznat kao Browser. U samo nekoliko mjeseci browseri prve generacije počeli su podržavati ovaj standard, međutim stvari nisu bile tako jednostavne. Gotovo svaki od browsera imao je svoju specifičnu verziju HTML-a, a samo nekolicina njih je bila međusobno kompatibilna. Godinama su razne grupe za standarde pokušavale unificirati ovaj jezik i napraviti ga industrijskim standardom, ali ti pokušaji nisu urodili plodom sve do pojave HTML-a 2.0.

HTML+

U 1993. godini, dok je borba za standarizaciju HTML-a uveliko trajala, pojavila se inicijativa za nadogradnju jezika za verziju HTML+. Na žalost, ova verzija nikada nije priznata, iako je nudila veću fleksibilnost od HTML-a 2.0 koji je kasnije postao standard. Međutim, važno je pomenuti HTML+ iz razloga što je on imao uticaja na razvoj sljedećih verzija HTML-a.

HTML 2.0

Ovo je prva verzija ovog jezika koja je postala standardom. Ugledala je svjetlo dana 1995. godine, dakle trebalo je četiri godine i nekoliko neuspjelih pokušaja da bi se došlo do standarda za HTML jezik. Ubrzo nakon objavljivanja specifikacija HTML-a 2.0 pojavila su se dva browsera koja danas zauzimaju najveći udio tržišta – Netscape Navigator 2.0 i Microsoft Internet Explorer 2.0. Oba su potpuno podržali specifikacije HTML-a 2.0, a treba napomenuti kako je to prva verzija ovog jezika koja je imala podršku za upotrebu slika i formi u web dokumentima.

HTML 3.0

Ubrzo nakon eksplozije weba i Interneta uopšte, što je i donijelo HTML-u 2.0 veliku popularnost, web developeri, odnosno dizajneri web stranica, počeli su primjećivati nedostatak fleksibilnosti tadašnje verzije HTML-a. Kao rezultat toga, započet je razvoj nasljednika HTML-a 2.0. Dosta toga uzeto je iz nikada priznatog HTML+ jezika, a najveće novosti su podrška za tabele, više mogućnosti formatiranja teksta, veća fleksibilnost u formatiranju izgleda stranica... Međutim,

poput svoga prethodnika – HTML+ jezika, i HTML 3.0 je prošao poprilično loše na tržištu. Promjene u odnosu na HTML 2.0 su bile ogromne i zahtijevale su velike prilagodbe browsera. Ali, kompanije koje su pravile browsere odlučile su kako im je to prevelik i neisplativ poduhvat, tako da HTML 3.0 nikada nije u potpunosti bio podržan u njihovim proizvodima.

HTML 3.2

U želji da zaobiđu nedostatke HTML-a 2.0 neki browseri su počeli da koriste svoje specifične tagove kako bi postigli veću fleksibilnost jezika. Na žalost, ovo je dovelo u pitanje originalnu ideju iza HTML jezika - univerzalnog jezika za publiciranje dokumenata na webu. Zato je W3 Consortium pokrenuo inicijativu za razvijanje nove verzije HTML-a. Rezultat je bio verzija 3.2 koja je sadržavala HTML 2.0, HTML 3.0, kao i neke specifične tagove koje su browseri počeli koristiti. U maju 1996. godine objavljene su specifikacije nove verzije, a svi veliki browseri su ih u potpunosti podržavali. Istina, u finalnoj verziji HTML 3.2 nedostajale su neke mogućnosti poput style sheetsa, napredne matematičke funkcije, okviri, kao i internacionalizacija jezika.

HTML 4.0

HTML 4.0 je konačno postao standardom u decembru 1997. godine i samo je djelimično bio podržan u browserima te generacije - Netscape Navigatoru 4.0 i Microsoft Internet Exploreru 4.0. Do tada se razvoj HTML odvijao uvijek tako da proširenja jezika odgovaraju browserima na tržištu, ali to se promijenilo u HTML-u 4.0 koji predstavlja prvi pokušaj da se ovaj jezik zaista proširi i postane fleksibilan i prilagodljiv. Uvođenje taga OBJECT omogućilo je ubacivanje koda skripti direktno u HTML dokumente, uvedene su Cascading Style Sheets (CSS) i mnoge druge stvari koje su nedostajale u HTML-u 3.2. Sada je granica u razvoju web stranica bila samo mašta web dizajnera i ništa više.

Dynamic HTML (DHTML)?

DHTML nije nova verzija, odnosno revizija HTML jezika. Nije niti neka specifična tehnologija, kao što su na primjer ActiveX ili CSS. DHTML kombinira mnoge tehnologije kao što su CSS, JavaScript, VBScript, Document Object Model, ActiveX zajedno sa HTML-om kako bi stvorio dinamične web stranice. Jednostavno rečeno, DHTML je HTML koji se može mijenjati čak i nakon što je stranica downloadovana i prikazana u browseru. Kada predete mišem preko neke slike, ona se promijeni u drugu sliku, ili kada predete preko nekog paragrafa, njegova se boja promijeni... Ili, otvorite stranicu, a naslov doleti s desne strane i smjesti se u vrh stranice. Sve što se može uraditi sa HTML-om može pomoću DHTML-a biti promijenjeno i nakon što je stranica downloadirana u browser. HTML je ono što mijenjate, DOM (Document Object Model) je ono što ga čini promjenljivim, a client-side scripting (VBScript i JavaScript) je ono što ga mijenja. I upravo to je Dynamic HTML.

3.4. Pristup dokumentima i URL-ovi

Da biste pristupili nekom dokumentu na Internetu, potrebno je znati više stvari. Za početak, ime računara na kojem se nalazi, direktorij u kojem je pohranjen i samo ime dokumenta. No, ni to nije dovoljno: trebate znati koju uslugu ćete koristiti za pristup do dokumenta. Iako je poznato da World Wide Web omogućuje i pristup do FTP i Gopher servera ili servera novinskih grupa, pretraživaču treba znati "objasniti" gdje da nešto potraži i kako da pristupi udaljenom računar na kojem se nalazi željeni dokument. Osim toga, za korištenje nekih usluga potrebno je navesti i dodatne informacije, poput broja porta (ukoliko se usluga izvršava na nekom nestandardnom portu), a u slučaju pristupanja nekim uslugama, naprimjer Telnetu, potrebno je navoditi i korisnička imena i lozinke.

Kako bi sve to bilo što jednostavnije, i kako bi se usluge, gledano sa strane korisnika, zaista mogle ujediniti pod jednim zajedničkim sučeljem (što Web i radi), osmišljen je sistem URL adresa. URL je skraćenica od Uniform Resource Locator; to je adresa kojom je jedinstveno ukazano na neki resurs (uslugu, sadržaj) na Internetu. Općeniti način pisanja URL adrese jest:

Protokol://ime.računara:port/direktorij/direktorij/.../datoteka

Ali i tu postoji nekoliko izuzetaka i dodataka; pisanje URL-a započinje navođenjem protokola koji će se koristiti za pristup dokumentu navedenom na kraju URL-a. U slučaju World Wide Weba, protokol je, naravno, najčešće http, no vidjet ćemo da se kao protokol mogu navesti i neki drugi, koji omogućavaju pristup do drugih usluga Interneta neposredno iz pretraživačkog programa za World Wide Web. Pretraživač podrazumijeva da se radi o http protokolu, ako se protokol eksplicitno ne navede u URL-u.

Nakon naziva protokola dolazi dvotačka, a zatim dvije obične kose crte. Postoji jedna iznimka od toga pravila, a to su lokalni URL-ovi koji se koriste za pristup datotekama što se nalaze na tvrdom disku računara na kojem radite, a ne negdje na nekom udaljenom računar na mreži.

Iza druge kose crte slijedi ime računara kojemu se pristupa. Ono može biti pisano ili u obliku IP adrese ili kao uobičajeno slovo ime (FWDN adresa). Ako se server za uslugu kojoj pristupamo, a čije smo ime protokola naveli na početku URL-a, izvršava na nekom portu različitom od standardnog, možemo navesti broj porta odmah iza broja računara, razdvajajući ga dvotačkom. Broj porta vrlo rijetko se susreće u World Wide Web adresama. Nakon još jedne kose crte, slijedi put do datoteke ili dokumenta kojemu želimo pristupiti. Prije njegovog imena treba, naravno, navesti i potpuno ime direktorija gdje se nalazi.

Uniform Resource Identifier (URI) navodi samo imena direktorija i datoteke na računar, te ne sadrži, kao URL, i cijelo ime računara zajedno s protokolom koji se koristi. Tako se URI-i koriste kada je veza servera i klijenta već otvorena, pa nije potrebno ponovno navoditi protokol ili ime servera s kojim se povezuje.

4. HTML

Web server je mehanizam namijenjen distribuiranju dokumenata kroz Internet. Iako će drage volje prebacivati po Mreži bilo koje tipove dokumenata, najvažniju klasu predstavljaju oni koji su sastavljeni u jeziku HTML. Radi se o "maternjem jeziku" Weba kojim se određuje formacija, sadržaj i funkcija nekog hipetekstualnog dokumenta. Njime se kombinira nekoliko važnih elemenata: namijenjen je kontroliranju načina na koji će pretraživački programi prikazati dokument, kreiranju hipertekstualnih veza između različitih mjesta u dokumentu, kreiranju hiperveza između različitih dokumenata, te kreiranju hiperveza s drugim servisima na Internetu. HTML vam također omogućava da u dokumente ubacujete grafiku, zvuk i druge multimedijalne zapise. Kako su HTML dokumenti mjesta na kojima je većina pravih podataka doista pohranjena, dobar dio kreiranja jednog cjelovitog Web servera sastoji se od njihovog pisanja.

Stranica na World Wide Webu sastoji se od skupa datoteka pohranjenih negdje na disku nekog računara na kojem je instaliran Web server, odnosno rjeđe, FTP server. Tačna lokacija tih datoteka određena je URL adresom. Web stranica se u većini slučajeva sastoji od teksta i slike, s time da se sav tekst i svaka pojedina slika nalaze u zasebnoj datoteci. Slikama se uvijek pristupa pomoću imena datoteke u kojima su pohranjene, pa se tako jedna te ista slika može koristiti više puta unutar jedne ili više različitih stranica. Svim drugim multimedijalnim elementima pristupa se kroz pomoćne aplikacije koje pokreće pretraživački program kada korisnik klikne na odgovarajuću hipertekstualnu vezu.

Oblikovanje Web stranica u mnogim se pogledima razlikuje od tradicionalnog stolnog izdavaštva. Najveća je razlika u tome što sada kreirate stranicu koja nije ograničena svojim tačnim formatom. Korisnikov pretraživački program će automatski prelomiti rečenice teksta u zasebne retke, ovisno o širini prozora, te će mu omogućiti da se šeće gore-dolje po stranici. U stolnom izdavaštvu potrebno je tačno odrediti tipove pisma i stilove, podesiti razne veličine, te tačno zadati poziciju svakog pojedinog elementa na stranici. Autori Web stranice ne moraju se zamarati s takvim preciznim podešavanjima, jer se za konačan izgled stranice ne brinu oni, već korisnikov pretraživački program.

Kada kreirate Web stranicu, vi zapravo u tekst koji će se prikazati ubacujete HTML naredbe kojima opisujete kako biste htjeli da izgleda svaki pojedini element u dokumentu. Tim postupkom se koristite nekom apstraktnom vrstom programiranja, budući da na neki način kreirate pravu malu multimedijalnu aplikaciju. Razlika između izrade HTML dokumenta i programiranja je u tome što će se krajnji rezultat moći koristiti neovisno o hardverskoj platformi i operativnom sistemu, a korisnik ga neće moći upotrebljavati za obradu nekih svojih podataka. Ipak, koliko god da se oblikovanje Web stranica razlikuje i od stolnog izdavaštva i od programiranja, ono obuhvaća elemente obiju ovih radnji. Ako se uzmu u obzir i činjenice da je za upotrebu naprednih mogućnosti HTML-a kakve su obrada obrazaca i ugradnja Java aplikacija potrebno ući u programerske vode, te da morate

kreirati grafički vrlo privlačnu stranicu ako se želite pred korisnicima istaknuti među milionima drugih Web stranica koje već postoje na Internetu, dolazi se do zaključka da pravi autor HTML dokumenata mora biti i dobar dizajner i dobar programer.

4.1. Struktura stranice

Datoteka koja se sastoji od teksta obogaćenog HTML naredbama općenito se naziva HTML dokumentom. Svaki HTML dokument započinje naredbom `<HTML>`, a završava s `</HTML>`. On se sastoji od dva dijela; zaglavlja koje je obuhvaćeno naredbama `<HEAD>` i `</HEAD>`, te tijela koje se nalazi unutar oznaka `<BODY>` i `</BODY>`. Zaglavlje sadrži neke osnovne informacije o dokumentu, dok se tijelo sastoji od teksta i naredbi kojima se taj tekst oblikuje. Najjednostavnija informacija koju možemo smjestiti unutar zaglavlja je naslov naše Web stranice, a to se čini upotrebom naredbe `<TITLE>`. Da bi vam sve ovo bilo jasnije, pogledajte sljedeći primjer minimalnog HTML dokumenta:

```
<HTML>
<HEAD>
    <TITLE>mala Web straničica</TITLE>
</HEAD>
<BODY>
    Ovdje se nalazi tekst koji želite prikazati na Webu,
    zajedno s HTML oznakama koje ga formatiraju, te s
    kojima se pozivaju slike i neki drugi elementi.
</BODY>
</HTML>
```

Važno je razumjeti da upravo naredbe `<BODY>` razdvajaju dokument u dvije cjeline, a ne reci ili razmaci koje smo ostavili praznima između redaka. Da uopće nismo koristili tipku Enter u prethodnom primjeru, tj. da smo cijeli ovaj dokument napisali kao jednu dugačku liniju teksta, on bi još uvijek generirao Web stranicu koja izgleda potpuno jednako. Naš HTML dokument, prema tome, bit će potpuno drugačije formatiran kada ga gledamo s editorom teksta, a drugačije kad se gleda s pretraživačkim programom. Slično tome, svi dodatni razmaci između riječi, odlomaka ili znakova, svi dodatni Enteri koje ćete nabaciti tek tako da HTML kod bude što duži i svi tabulatori, u pretraživačkom programu će se jednostavno zanemariti, što znači da će biti zamijenjeni jednim jedinim razmakom. Stoga slobodno upotrebljavajte te elemente prilikom uređivanja HTML dokumenta kako biste ga učinili čitljivijim za daljnju upotrebu ili za neke kasnije ispravke.

U primjeru smo kao jedini element zaglavlja naveli naslov dokumenta. Kod većine Web stranica to je jedna stvar koju ljudi redovno stavljaju u zaglavlje, premda se

tamo može smjestiti još i čitav niz drugih više ili manje korisnih informacija. Mogućnost uvrštavanja tih drugih informacija u zaglavlje ovisi o tipu Web servera, pa je najbolje da se o tome raspitate kod svojeg Web administratora. Jedno je sigurno, u zaglavlje uvijek možete staviti naslov dokumenta koji će biti ispisan na okviru prozora pretraživačko programa i u bookmark (Favorites) listi ukoliko korisnik odluči tamo pohraniti vašu stranicu.

Sve HTML naredbe koje smo koristili u prethodnom primjeru mogu se slobodno izostaviti. HTML dokumenti u biti i ne moraju sadržavati zaglavlje i tijelo da bi dobro izgledali u pretraživačkom programu, no tada će biti u koječemu zakinuti. Tako će, na primjer, stranica koja ne sadrži oznake zaglavlja, a sadrži naslov, u nekim pretraživačkim programima imati ispisan naslov u okviru prozora, dok u drugim neće. Isto tako, stranica bez oznaka za tijelo dokumenta ne može nikako sadržavati pozadinu ispunjenu nekom bojom ili slikovnim uzorkom, budući da se ti elementi označavaju unutar HTML koda u obliku argumenata koji se dodjeljuju naredbi `<BODY>` Nemojte grafičku prezentaciju svoje Web stranice ograničavati na račun uštede nekoliko sekundi potrebnih za upisivanje ovih jednostavnih naredbi, jer će vam kasnije biti žao. Te su oznake izmišljene zato da bi se koristile, a ne preskakale, pa nema razloga da vi činite suprotno od ostalih. Pametni znaju čemu služe oznake. Programi za kreiranje Web stranica prilikom kreiranja novog dokumenta automatski u taj prazan HTML dokument ubacuju te oznake.

4.2. Vrste naredbi

Postoji samo nekoliko sintaktičkih pravila kojih se valja pridržavati da bi se ovladalo jezikom HTML. Prvo, postoje dvije vrste elemenata od kojih se jezik sastoji: naredbe namijenjene oblikovanju stranice, te oznake za posebne znakove. Prethodni primjer kojim je ilustrirana struktura dokumenta sadrži četiri naredbe. Nije teško zaključiti o kojima se radi. Oznake za posebne znakove koriste se kod definiranja onih slovnih oznaka koje se ne mogu upisati u okolni tekst na uobičajen način, putem tipkovnice. Takve oznake započinju simbolom `&` (ampersand) iza čega se navodi naziv posebnog znaka, te naposljetku tačka-zarez. Umjesto naziva posebnog znaka može se navesti i simbol `#` iza kojega se navodi decimalna vrijednost željenog znaka u ISO Latin-1 skupu simbola. Tako se, naprimjer, tilda (`~`) generira nizom `~M`.

Budući da je vrlo teško zapamtiti decimalne vrijednosti više od nekoliko ovakvih posebnih znakova, puno će se lakše snaći ako upotrijebite standardne oznake u kojima se između simbola `&` i tačka-zarez navodi naziv željenog znaka. U nastavku donosimo tablicu koja popisuje nekoliko češće korištenih posebnih znakova.

Tabela 3. Upotreba posebnih znakova unutar teksta

Simbol	HTML oznaka	Značenje
&	&	ampersand
<	<M	manje od
>	>	veće od
®	®	registrirana oznaka
©	©	copyright-autorsko pravo
á	á	malo a s desnom kosom crticom
à	à	malo a s lijevom kosom crticom
â	â	malo a s obrnutom kvačicom
ä	ä	malo a s preglasom (dvije točkice)
å	&arubg;	malo a s kružicom
ç	ç	malo c s kukicom, francusko slovo
ñ	ñ	malo n s tildom
ø	ø	malo o prekríženo kosom crtom
ß	&szling	njemačko slovo oštro s
æ	&aeling;	posebna oznaka za izgovor
	 	neizbrisivi razmak

4.3. Sintaksa naredbi

HTML naredbe mogu biti pune ili prazne. PUNE naredbe odnose se na tekst koji je njima obuhvaćen, a sastoje se od početne i završne oznake. Početna oznaka započinje simbolom < (manje od), iza čega se navodi sama naredba te eventualno neki dodatni argumenti. Kraj početne oznake prepoznaje se po simbolu > (veće od). Završna oznaka razlikuje se od početne po tome što ne smije posjedovati nikakve dodatne argumente, ali zato između simbola < i naredbe uvijek mora sadržavati kosu crtu (/). Primjer pune naredbe bio bi:

```
<A HREF="piramida.html">Saznajte sve o piramidi</A>
```

U ovom primjeru početna oznaka je , a završna . Primjećujete da početna oznaka sadrži i jedan argument. Za razliku od punih naredbi kojima se obuhvaćao određeni dio teksta, prazne nam služe kako bismo nešto mogli ubaciti unutar teksta, poput slike, vodoravne linije ili oznake za novi redak. Prazne naredbe ne dolaze u parovima, pa tako kod njih postoji samo početna oznaka, a ne i završna koja bi sadržavala kosu crtu. Slijedi nekoliko primjera praznih naredbi.

```
<BR> Kraj retka;  
nadolazeći tekst bit će ispisan u novom retku.
```

```
<HR> Vodoravna linija koja se iscrtava duž stranice.
```

Kao i kod punih, tako se i kod praznih naredbi mogu koristiti argumenti kojima se određuju neke konkretne vrijednosti. Sljedeći primjer prikazuje naredbu IMG

kojom se unutar teksta ubacuje slika. Ta naredba uvijek dolazi s argumentom SRC kojim se dodjeljuje ime datoteke koja sadrži željenu sliku.

```
<IMG SRC="piramida.gif!"> Slika piramide.
```

Argumenti se zapisuju u obliku varijabla=vrijednost, gdje vrijednost mora odgovarati tipu varijable. Ako se, kao u našem primjeru, varijabli dodjeljuje neka tekstualna oznaka poput imena neke datoteke, onda te vrijednost mora biti obuhvaćena navodnim znacima, dok se oni kod numeričkih veličina izostavljaju. Katkada se argumentima ne dodjeljuju nikakve vrijednosti ukoliko oni predstavljaju neku logičku vrijednost. Tako je, naprimjer svejedno hoćete li kao argument navesti BORDER="yes" ili samo BORDER. Ako vaša naredba sadrži više od jednog argumenta, onda se oni odvajaju razmacima, a ne zarezom kao što je to običaj kod većine programskih jezika.

Kod primjera za punu oznaku upotrijebili smo naredbu A koja se koristi za definiranje hipertekstualnih veza. Izraz "Saznajte sve o piramidi" u pretraživačkom programu bi se kod tog primjera prikazao drugačije od ostalog teksta, najčešće označen plavom bojom i podcrtan, kako bi se korisniku dalo do znanja da će se klikom na njega učitati nova stranica. U ovom slučaju radi se o stranici koja je sadržana u dokumentu piramida.html, koji smo dodijelili kao vrijednost argumentu HREF (Hypertext REference). U nastavku je dan primjer koji ilustrira upotrebu svih dosad spomenutih sintaktičkih pravila, te upotrebu obiju vrsta HTML naredbi i oznaka za posebne znakove.

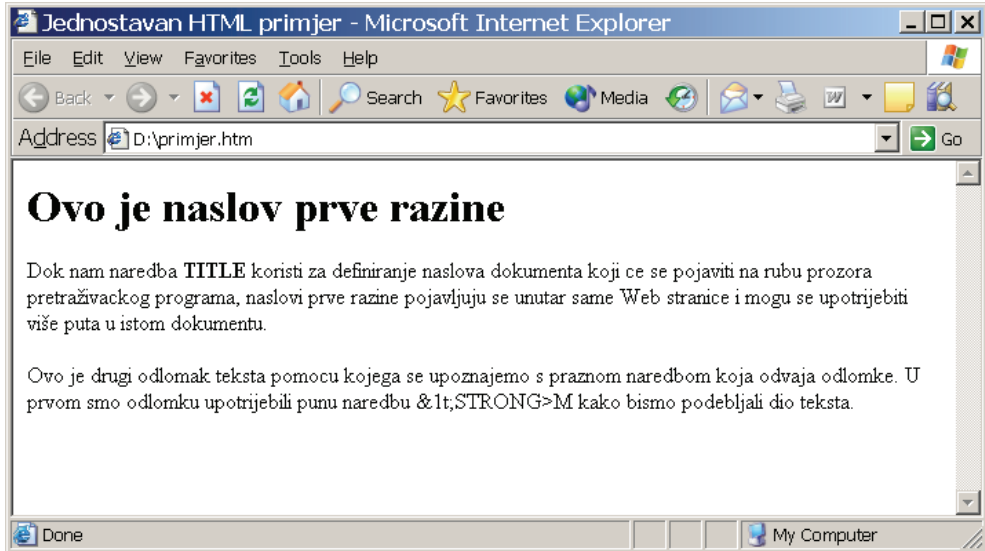
```
<HTML>
<HEAD>
  <TITLE>Jednostavan HTML primjer </TITLE>
</HEAD>
<BODY>
  <H1>Ovo je naslov prve razine </H1>

  Dok nam naredba <STRONG>TITLE</STRONG> koristi za
  definiranje naslova dokumenta koji će se pojaviti na
  rubu prozora pretraživačkog programa, naslovi prve
  razine pojavljuju se unutar same Web stranice i mogu se
  upotrijebiti više puta u istom dokumentu. <P> Ovo je
  drugi odlomak teksta pomoću kojega se upoznajemo s
  praznom naredbom koja odvajava odlomke. U prvom smo
  odlomku upotrijebili punu naredbu &lt;STRONG> kako
  bismo podebljali dio teksta.

</BODY>
</HTML>
```

Slika 3 prikazuje kako ovaj primjer izgleda kada se učita u Internet Explorer. Uočite odmah da se naslov "Jednostavan HTML primjer" pojavljuje u gornjem

rubu prozora. Tijelo ovog dokumenta sastoji se od naslova prve razine koji smo obuhvatili oznakama `<H1>` i `</H1>`, te dva tekstualna odlomka odvojena naredbom `<P>`. Uvjerite se da prelasci teksta u novi redak kod HTML dokumenta ne utječu na raspored redaka kod Web stranice, ali da su odlomci razdvojeni praznim retkom iako ih mi u HTML-u nismo razdvojili ručno.



Slika 3. Jednostavan primjer HTML dokumenta

U drugom smo odlomku upotrijebili oznake za posebne znakove `<` i `>` kako bismo na ekranu dobili string "`<STRONG>`". Da smo umjesto ovih oznaka izravno upisali znakove `< i >`, ovaj se string ne bi prikazao u pretraživačkom programu, već bi sav preostali dio teksta bio podebljan poput riječi "TITLE" u prvom odlomku.

Iako je HTML dokument u ovom primjeru uredno formiran, nije bitno gdje ćete tačno smjestiti naredbe jezika u odnosu na okolni tekst. Sljedeća tri primjerka prikazuju kako se na različite načine može u pretraživaču dobiti isti naslov kao i onaj sa slike 3.

```
<H1>Naslov prve razine </H1>

<h1>

Naslov prve razine

</h1>

<H1> Naslov

prve razine </h1 >
```

Tačno! Svejedno je pišete li HTML naredbe malim ili velikim slovima. Slobodno možete miješati mala i velika slova unutar iste naredbe, no budite oprezni sa suvišnim razmacima. Upotreba razmaka kod HTML naredbi dozvoljena je samo

ispred znaka `>`, premda ne postoji neki razuman razlog zašto biste ih uopće koristili bilo gdje. String:

```
< H1 >Tu nešto ne valja!< /H1 >
```

neće biti prikazan kao naslov. Simboli `<` i `>` bit će ignorirani zajedno sa sadržajem koji se nalazi između njih, a prikazat će se samo rečenica "Tu nešto ne valja!" i to kao običan tekst. Pretraživački programi su vrlo tolerantni prema pogreškama i većinu će ih jednostavno ignorirati, ne javljajući korisniku ništa. Zbog toga je važno da poštuju sintaksu jezika HTML jer za njega ne postoji kompajler koji bi vas kao autora obavijestio o učinjenim sintaktičkim pogreškama.

HTML naredbe mogu se ugnježdavati, tj. koristiti jedna unutar druge, pa ćete tako moći koristiti hipertekstualne veze u listama ili kosi tekst unutar podebljanog. Neke se naredbe, međutim, ne smiju kombinirati kao, na primjer, hipertekstualne veze unutar drugih hipertekstualnih veza. Postoje i situacije kada je dozvoljeno kombiniranje, ali ga pretraživački programi ignoriraju. To se, naprimjer, događa kod upotrebe naredbi za tekstualne stilove unutar naslova.

Unutar HTML dokumenta možete koristiti i komentare kako biste svoj rad učinili što preglednijim, kao i kod drugih programskih jezika. Svaki pretraživački program će u potpunosti ignorirati komentare, što znači da unutar komentara smijete upisivati čak i HTML naredbe te posebne znakove. To, međutim, nije preporučljivo budući da postoje starije verzije nekih pretraživača koje ne prepoznaju oznake za komentare. Komentari se moraju smjestiti unutar oznaka `<!--` i `-->`, a ne smiju se ugnježdavati. Postoji nepisano pravilo da se svaki pojedini redak komentara obuhvaća ovim oznakama, kako bi HTML dokument bio što pregledniji.

4.4. SSI

Ako se vaše stranice nalaze na računarima na kojem je instaliran Web server koji podržava SSI, moći ćete iskoristiti SSI (Server-Side Includes) tehnologiju kako biste dotjerali svoje HTML dokumente bez potrebe za izradom tzv. CGI skripte. SSI vam omogućava da na jednostavan način unutar stranice ugradite korisne stvari poput trenutnog vremena, datuma ili veličine dokumenta. Također ćete moći ubacivati druge dokumente u sadržaj trenutnih HTML dokumenata, što je korisno kod nekih uobičajenih tekstova poput poruke o zaštiti autorskih prava ili adrese vaše tvrtke.

Kada je Web server konfiguriran tako da radi s SSI tehnologijom, on prepoznaje poseban tip HTML dokumenata čiji je MIME tip nazvan

```
Text/x-server-parsed-html.
```

Ti dokumenti posjeduju ekstenziju `.shtml`, osim ako server nije konfiguriran drugačije. Kada se serveru pošalje zahtjev za jednim takvim dokumentom, on će ga otvoriti, te će potražiti sadržaj li ugrađene posebno označene komentare. Ti komentari sastavljeni su sljedećom sintaksom:

```
<!-- #naredba param1="vrijednost1"  
param2="vrijednost"...-->
```

Naredbe govore serveru što želite napraviti (ispred njih se obavezno piše znak #), dok jedan ili više parametara daju dodatne podatke. Postoji šest različitih naredbi: echo, include, fsize, flastmod, exec i config, a u nastavku je opisano kako se svaka pojedina koristi.

Naredba echo zahtijeva od servera da unutar HTML dokumenta ispiše vrijednost neke globalne varijable koja se koristi na serverskom računaru. Postoje četiri varijable koje možete koristiti. Prva je DOCUMENT_NAME, a namijenjena je ispisivanju datotečnog imena trenutnog dokumenta, druga je DATE_LOCAL, a koristi se za ispisivanje lokalnog vremena i datuma; vrijeme po Greenwichu ispisuje se varijablom DATE_GMT, a varijablom LAST_MODIFIED može se ispisati datum zadnje izmjene trenutnog dokumenta. Evo i primjera:

```
<!-- #echo var="DATE_LOCAL"-->
```

Naredbom include može se u sadržaj trenutnog dokumenta ugraditi sadržaj nekog drugog HTML dokumenta. S ovom se naredbom mogu koristiti parametri virtual i file, ovisno o tome navodi li se URL adresa drugog dokumenta relativno u odnosu na trenutni direktorij, odnosno relativno u odnosu na korijenski direktorij računara s Web serverom. Na mjestu gdje u trenutnom dokumentu želite uvrstiti sadržaj drugog dokumenta, upišite ovakvu naredbu:

```
<!-- #include file="/eurosong/lordi.html"-->
```

Naredbama fsize i flastmod može se unutar Web stranice ispisati veličina, odnosno datum zadnje promjene bilo koje datoteke koja se nalazi na vašem Web serveru. Kao i kod naredbe include, moći ćete upotrijebiti parametre virtual i file, kao u sljedećem primjeru:

```
<!-- #fsize file="/eurosong/lordi.html"-->
```

Pomoću naredbe exec omogućeno je izvršavanje bilo kojih programa na Web serveru čiji se ispis preusmjerava u trenutni dokument. Radi se o potencijalno opasnoj opciji, pa je mnogi administratori onemogućavaju. Ovisno o tome želite li izvršiti neki program koji predstavlja sastavni dio UNIX ljuske sistema na kom radite ili neka CGI skripta, upotrijebit ćete parametre cmd ili cgi. U nastavku je dan primjer pomoću kojeg možete unutar HTML dokumenta upotrijebiti naredbu finger, podržanu na svim UNIX sistemima:

```
<!-- #exec cmd="/usr/bin/finger slemes"-->
```

Posljednja naredba, config, koristi se kako bi se odredilo ponašanje servera u slučaju da dođe do greške prilikom korištenja drugih naredbi SSI tehnologije, te kako bi se odredio način obavještanja korisnika Web stranice o pogreškama. Parametrom errmsg navodi se tekst koji će se ispisati u dokumentu u slučaju pogreške, kao što je navedeno u primjeru:

```
<!-- #config errmsg="došlo je do SSI pogreške..."-->
```

Više o SSI tehnologiji pročitajte na URL adresi:

<http://hoohoo.ncsa.uiuc.edu/docs/tutorials/includes-html>.

4.5. Osnovni elementi jezika

HTML naredbe možemo podijeliti u dvije zasebne kategorije – postoje naredbe koje utječu na strukturu stranice, te naredbe koje su namijenjene promjeni tekstualnih stilova. Među one prve možemo ubrojiti naredbe za definiranje naslova, listi, tabela, okvira i drugih sličnih elemenata, a zajedničko svojstvo im je da prilikom svake upotrebe automatski stvaraju kraj odlomka u tekstu koji ih okružuje. Zato ih i nazivamo naredbama koje formiraju strukturu stranice. Stilske HTML naredbe definiraju tipografiju teksta, u što ubrajamo promjenu veličine pisma, podebljavanje ili nagibanje slova, te druge radnje koje ne utječu na opći raspored elemenata na stranici. Naredba za kreiranje hipertekstualne veze, naprimjer, spada u ovu potonju grupu, budući da vizuelno mijenja samo dio neke rečenice unutar tekstualnog odlomka. HTML oznake pomoću kojih se unutar stranice ugrađuju slike pripadaju zasebnoj skupini naredbi o kojima će biti više riječi kasnije u ovom poglavlju.

4.5.1. Naslovi

Različite tematske cjeline svakog dokumenta predstavljaju se i razdvajaju naslovima. HTML podržava šest razina naslova koji se definiraju parovima naredbi `<H1></H1>`, `<H2></H2>`, `<H3></H3>`, `<H4></H4>`, `<H5></H5>` i `<H6></H6>`. Šest razina naslova sasvim je dovoljno za kreiranje čak i najsloženijih tekstualnih dokumenata, posebno ako se uzme u obzir da se hijerarhijska struktura dokumenta puno jednostavnije rješava kreiranjem hipertekstualnih veza koje veliki dokument razdvajaju na više manjih, zasebnih dijelova.

H1 je najviša razina naslova. Običaj je da se ova naredba uvijek uvrsti kao prvi element unutar tijela dokumenta kako bi Web stranica sadržavala i unutrašnji naslov, za razliku od onog vanjskog koji se vidi na rubu prozora, dobivenog naredbom `<TITLE>` u zaglavlju dokumenta. Upotreba naredbi za definiranje naslova rezultirat će automatskom pojavom kraja prethodnog, odnosno početka sljedećeg odlomka teksta, što znači da prije oznake `<H1>`, te poslije oznake `</H1>` ne morate koristiti naredbu `<P>` kako biste kreirali razmak između odlomaka. Isto, dakle, vrijedi i za sve ostale razine naslova.

Naslovi bi se trebali koristiti prirodnim slijedom razina, počev od najviše redom prema nižima, no dozvoljeno je preskakanje pojedinih razina. Tako se, na primjer, nakon naslova prve razine slobodno može upotrijebiti naredba za definiranje naslova treće razine. Slijedi primjer koji ilustrira upotrebu svih šest razina naslova:

```
<HTML>

<HEAD>

  <TITLE>Razine naslova</TITLE>

</HEAD>

<BODY>

  <H1>Naslov prve razine - najveća slova </H1>

  Ovo je prvi odlomak teksta koji se mora odvajati
  posebnom naredbom.

  <H2>Naslov druge razine - velika slova</H2>

  Ovo je drugi odlomak teksta.

  <H3>Naslov treće razine - mala slova </H3>

  Ovo je drugi odlomak teksta.

  <H4>Naslov četvrte razine - mala slova </H4>

  Četvrti odlomak teksta ispisan je slovima gotovo
  iste veličine kao i naslov četvrte razine.

  <H5>Naslov pete razine - mala slova </H5>

  Peti odlomak teksta vidi se bolje od naslova pete
  razine.

  Kakve su ovo buhice gore?

</BODY>

</HTML>
```

Na slici 4 možete vidjeti kako ovaj primjer izgleda nakon učitavanja u pretraživački program.

Naslovi se uvijek pojavljuju ispisani uz lijevi rub Web stranice. Upotrebom argumenta ALIGN uz naredbe za kreiranje naslova, oni se mogu centrirati uz sredinu stranice čime se postižu bolji vizualni efekti. Da bismo to postigli, argumentu ALIGN moramo dodijeliti vrijednost CENTER. Slika 5 prikazuje kako naš sljedeći primjer u kojem je prikazana upotreba ovog argumenta izgleda nakon prikazivanja u pretraživaču.

```
<H3 ALIGN=CENTER>Ova knjiga </H3>

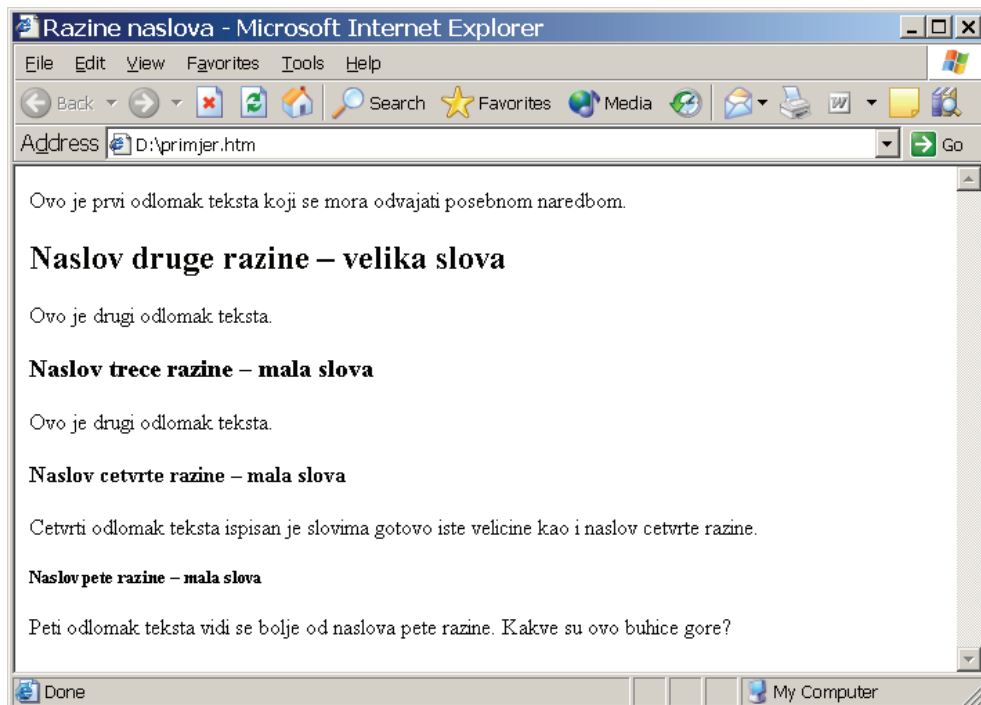
<H5 ALIGN=CENTER>predstavlja</H5>

<H1 ALIGN=CENTER>Centrirani naslov prve razine!</H1>

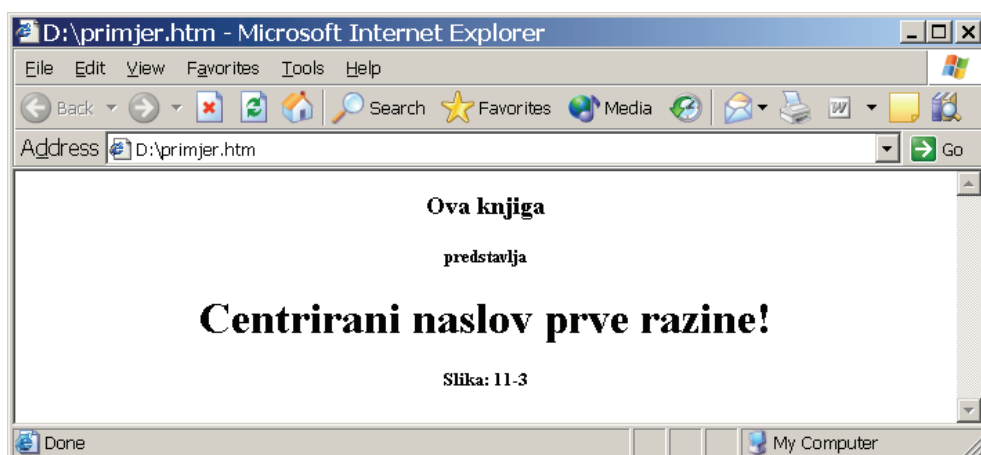
<H5 ALIGN=CENTER>Slika: 11-3</H5>
```

Kao ni imena HTML naredbi, tako ni njihovi argumenti ne prepoznaju razliku između velikih i malih slova, pa je sasvim svejedno kako ćete ih dopisivati uz naredbe. Međutim, radi što bolje preglednosti dokumenta dobro je sve naredbe i

argumente s dodijeljenim im vrijednostima pisati u potpunosti velikim slovima. Neki autori preferiraju pisanje argumenata velikim slovima, a njihovih vrijednosti malim, što je također dobro i uredno. Na vama je da odlučite koji ćete oblik zapisivanja koristiti kod svojih dokumenata. Važno je samo da se svoje odluke u svakom trenutku pridržavate.



Slika 4. Šest razina naslova



Slika 5. Centrirani naslovi

4.5.2. Uobičajeni argumenti

ALIGN je tek jedan od uobičajenih argumenata koji se mogu dopisivati uz naslove i većinu drugih strukturalnih HTML naredbi opisanih u ovom poglavlju: Argumentu ALIGN mogu se dodijeliti vrijednosti LEFT, RIGHT, JUSTIFY i CENTER, tj. određeni elementi Web stranice mogu biti poravnati uz lijevi rub, desni rub, oba ruba ili centrirani. Većina pretraživačkih programa prepoznaju i naredbu <CENTER> kojom se također mogu centrirati različiti elementi. Ta naredba uvijek dolazi u paru s </CENTER>.

Argument NOWRAP može se dodijeliti većini naredbi da bi se isključilo normalno prelamanje rečenica u retke. Ovom argumentu ne mora se pridružiti nikakva vrijednost, što je isto kao da ste mu pridružili vrijednost "YES". Ako unutar teksta na koji se odnosi naredba koja sadrži argument NOWRAP želite ostvariti prelazak teksta u novi redak, morat ćete upotrijebiti naredbu
.

Argumentom LANG može se odrediti alternativni jezik na kojem je napisan sadržaj teksta obuhvaćenog naredbom koja ga sadrži. Posljednji argument je CLASS, a namijenjen je pridruživanju stilske klase obuhvaćenom tekstu. Stilska klasa može se definirati u zaglavlju dokumenta, a radi se o skupini stilskih naredbi obuhvaćenih jednim imenom koje se dodjeljuje ovom argumentu kao vrijednost. Argument CLEAR koji može poprimiti vrijednosti LEFT, RIGHT i ALL, namijenjen je poravnavanju teksta koji pluta oko lika unutar Web stranice. Ukoliko ste neki tekst ispisivali s lijeve strane slike, a želite da se novi odlomak nalazi ispod nje, ispisan odmah uz lijevi rub stranice, upotrijebit ćete, naprimjer, naredbu <P XCLEAR="LEFT">. Slično ćete postupiti ako te tekst ispisivali s desne strane slike, dodjelom vrijednosti RIGHT argumentu CLEAR, odnosno između dviju slika, pomoću vrijednosti ALL.

4.5.3. Odlomci

Odlomci se u HTML dokumentima razdvajaju naredbom <P>, no postoji još nekoliko oznaka čija će upotreba uzrokovati prebacivanje teksta u novi redak i ubacivanje praznog retka između okolnih elemenata na stranici.

Zanimljivo je da je standardom HTML 3 predloženo dodavanje završne oznake </P> za naredbu <P> kako bi se omogućilo korištenje argumenata koji bi se odnosili samo na jedan odlomak teksta. Želite li, naprimjer, izbjeći upotrebu naredbe <CENTER> kako biste centrirali neki određeni odlomak, obuhvatit ćete ga oznakama <P ALIGN=CENTER> i </P>. Ovdje se ne radi o nekom značajnom poboljšanju same strukture jezika, posebno ako odlomak ne želite centrirati, jer ćete morati u kod dokumenta upisivati jednu oznaku više za svaki odlomak.

4.5.4. Novi redak

Kada samo želite da se tekst prebaci u novi redak, bez dodatnog razmaka koji bi korisniku nagovještavao novi odlomak, upotrijebite naredbu
. Ta naredba je prazna, tj. ne dolazi sa završnom oznakom, pa je jednostavno ubacujte svugdje gdje biste kod pisanja teksta u nekom editoru inače pritisnuli Enter. Ova se naredba najčešće upotrebljava kod dugačkih naslova koji se moraju prelomiti u više redaka, ali i u mnogim drugim sličnim situacijama. U programima za izradu Web stranica ta se oznaka dobija i kombinacijom Shift+Enter.

4.5.5. Vodoravne linije

Između većih tematskih cjelina na jednoj stranici koje se sastoje od nekoliko odlomaka teksta, moći ćete ubaciti vodoravnu liniju pomoću naredbe <HR>. Time ćete automatski stvoriti i razmak između dva odlomka, pa ispred i iza ove naredbe ne morate upotrebljavati oznaku <P>. Pretraživački program će dužinu linije prilagoditi širini prozora unutar kojeg je prikazana stranica, a moći ćete sami određivati dužinu linije zahvaljujući proširenjima koja je uvela tvrtka Netscape.

Najkorisnije proširenje ove naredbe koristi se upravo za reguliranje dužine vodoravne linije upotrebom argumenta WIDTH kojemu valja dodijeliti željenu dužinu u postocima. Ako, naprimjer, želite da vaša linija bude široka tačno za polovicu prozora, argumentu WIDTH dodijelit ćete vrijednost 50%. Slično, pomoću argumenta SIZE može se odrediti i visina linije, što je posebno korisno kod centriranja linija koje se kreću od cijele širine prozora. Naredbom:

```
<HR SIZE=5 WIDTH=75% ALIGN=CENTER>
```

dobiva se centrirana vodoravna linija visine 5 piksela i širine tri četvrtine prozora pretraživačkog programa. Ukoliko pogledate sliku 6, primijetit ćete da linije posjeduju i sjenu, čime se postiže lagani efekt trodimenzionalnosti. Ukoliko želite izbjeći taj efekt i liniju učiniti potpuno crnom, naredbi <HR> dodajte još i argument NOSHADE kojem se ne dodjeljuje nikakva vrijednost. Sljedeći primjer prikazuje upotrebu vodoravnih linija, te naredbi za razdvajanje odlomaka i prelaska teksta u novi redak.

```
<HTML>
<HEAD>
  <TITLE>Pjesmica</TITLE>
</HEAD>
<BODY>
  <H1>Jedna super knjiga</H1>
  <HR SIZE=4>
  Ima tome dosta, al' pamtim još uvijek, <BR>
```

```

Mislio sam, mom neznanju ne postoji lijek. <BR>
Sjećanje na stare dane k'o struja me strese, <BR>
Razlikovao tad nisam CD-ROM od Web adrese.

<P>
Nekad nisam znao ni modem uključiti' <BR>
Mislio sam da to neću nikada naučiti'. <BR>

<HR SIZE=4>

</BODY>

</HTML>

```

Slika 6 prikazuje ovaj primjer nakon učitavanja u pretraživački program.



Slika 6. Odlomci i vodoravne linije

4.5.6. Citati

Često ćete dio stranice htjeti posebno istaknuti kako biste korisniku dali do znanja da se radi o materijalu citiranom iz nekog drugog dokumenta. Kada želite jedan ili više odlomaka označiti kao citate, obuhvatit ćete ih oznakama `<BLOCKQUOTE>` i `</BLOCKQUOTE>`. Tekst na koji se odnosi ova naredba većina će pretraživačkih programa prikazati uvećano od lijeve margine, a katkada i kosim slovima.

U slučaju da citirate više odlomaka teksta, upotrijebite naredbu `<BLOCKQUOTE>` samo jednom, a odlomak razdvojite naredbom `<P>`. To je puno bolje rješenje od višestruke upotrebe naredbe `<BLOCKQUOTE>`, jer se time izbjegava moguće

pojavljivanje duplih razmaka između odlomaka kod nekih pretraživačkih programa.

4.5.7. Adrese

Par oznaka `<ADDRESS>` i `</ADDRESS>` koristi se kod isticanja potpisa, adresa i drugih informacija koje se odnose na autora dokumenta, a najčešće se pojavljuju pri dnu stranice. Tekst koji je obuhvaćen ovom naredbom bit će prikazan kosim slovima, a ponekad i uvučen prema desno s obzirom na lijevu marginu – ovisno o raspoloženju pretraživačkog programa koji korisnik koristi.

Adrese se najčešće pišu u više redaka, pa i ovdje vrijedi slična napomena kao i kod citata: ne razdvajajte retke učestalom upotrebom naredbe `<ADDRESS>`, već se poslužite oznakom `<BR>`. Ako tekst koji ste označili ovom naredbom sadrži vaše ime, napravite hipertekstualnu vezu pomoću koje korisnik klikom na ime može učitati vašu sliku.

4.5.8. Napomene i fusnote (bilješke)

Napomene, savjeti, upozorenja i drugi slični dodaci tekstu označavaju se naredbama `<NOTE>` i `</NOTE>`. Tekst koji se nalazi unutar ovih oznaka neki će pretraživački programi prikazati uokviren, neki ispisan manjim slovima, a neki će dodati i odgovarajuću ikonu koja vizualno ukazuje na ulogu tog teksta. Uloge se određuju pomoću argumenta `ROLE` kojemu ćete u slučaju savjeta pridružiti vrijednost `TIP`, a u slučaju upozorenja vrijednost `WARNING`.

Fusnote (sitan tekst što se ponekad pojavljuje pri dnu stranica nekih knjiga kao opis za nepoznate izraze u tekstu označene zvjezdicom ili sitnim brojem), označavaju se naredbama `<FN>` i `</FN>`. Početna oznaka mora sadržavati argument `ID` kojemu ćete dodijeliti naziv fusnote, kako biste je razlikovali od drugih fusnota koje su eventualno sadržane u tekstu. Riječ koja je opisana tom fusnotom obuhvatit ćete oznakom `<A>` i `</A>`, gdje ćete uz početnu oznaku nadopisati i argument `HREF` kojem se dodjeljuje ime odgovarajuće fusnote, prethodno oznakom `#`. Zvuči zamršeno, ali sve će vam biti jasnije kada pogledate slijedeći primjer koji ilustrira upotrebu fusnota i napomena.

```
<HTML>
<HEAD>
    <TITLE>Napomene i fusnote</TITLE>
</HEAD>
<BODY>
    <H1>Evo jedan dobar savjet:</H1>
    <HR>
```

<NOTE ROLE=TIP>

Uvijek je korisno u džepu imati desetak KM viška.

</NOTE>

<HR>

Ako pri ruci nemate višak novca, uvijek se možete poslužiti i sitnim

poklonima, koja čovjeka vesele.

<P>

<FN ID="pokloni">

Ručni satovi, walkmani, CD-ROMovi, foto-aparati, SIMM moduli, te ostala sitna tehnika.

</FN>

</BODY>

</HTML>

4.5.9. Neproporcionalno pismo

Neproporcionalna pisma razlikuju se od običnih po tome što je svako slovo jednako široko. Obična pisma još se nazivaju i proporcionalnim, a najjednostavniji primjer bio bi tekst koji upravo čitate. Neproporcionalnim pismom je, naprimjer, ispisan prethodni primjer. Kako je osnovno pismo kojim se ispisuje tekst na Web stranicama proporcionalno, postoji naredba kojom možete cijeli odlomak označiti kao tekst koji se treba ispisivati neproporcionalnim pismom. Radi se o paru oznaka <PRE> i </PRE>, a valja napomenuti da će svi razmaci i prelasci teksta u novi redak koji se nalaze između ovih oznaka biti zapamćeni i identično prikazani u svakom pretraživačkom programu. Također je važno napomenuti da se unutar naredbe za neproporcionalno pismo smiju koristiti samo one HTML naredbe koje su namijenjene promjeni stila unutar rečenice, te naredbe za stvaranje hipertekstualne veze i ugradnju slika. Sljedećim primjerom prikazana je upotreba naredbe <PRE>.

<HTML>

<HEAD>

<TITLE>Neproporcionalno pismo</TITLE>

</HEAD>

<BODY>

<H1>Glavolomka</H1>

<PRE>

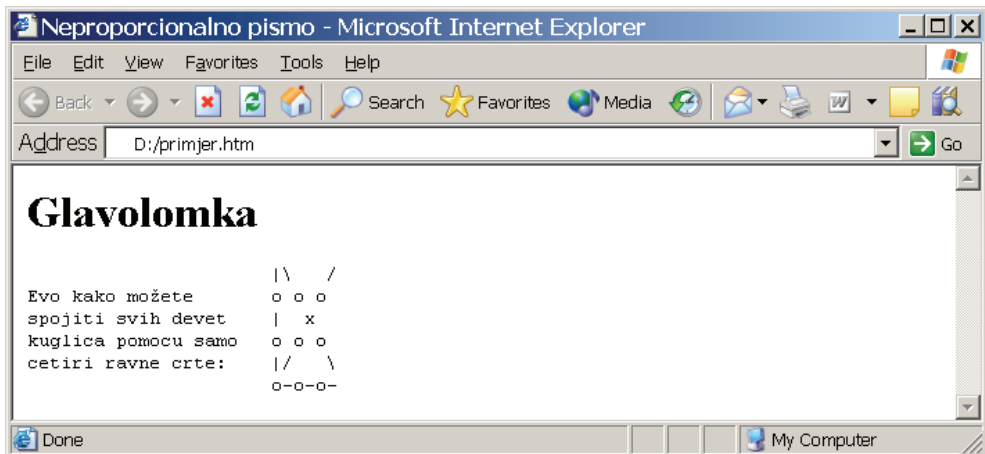
```

                                | \   /
Evo kako možete              o o o
spojiti svih devet           |  x
kuglica pomoću samo          o o o
četiri ravne crte:           | /   \
                                o-o-o-

</PRE>
</BODY>
</HTML>

```

Na slici 7 možete pogledati kako ovaj primjer izgleda nakon učitavanja u pretraživački program.



Slika 7. Neproporcionalno pismo

Naredbi za neproporcionalno pismo može se dodati i argument **WIDTH** kojim se određuje najveća moguća dužina jednog retka teksta na koji se naredba odnosi. Dužina retka se navodi kroz broj slova, a ovisno o navedenoj vrijednosti pretraživački program će prilagoditi veličinu neproporcionalnog pisma. Vrijednosti 40, 80 i 132 u pravilu daju najbolje rezultate. Evo i kratkog primjera kojim se prikazuje upotreba ovog argumenta:

```

<PRE WIDTH=80>Neproporcionalnost je izmislila Vlatka
Pokos</PRE>

```

4.5.10. Liste

Lista je strukturirani odlomak koji sadrži niz određenih elemenata. U jeziku HTML postoje tri vrste lista: pobrojane, nepobrojane i definitne liste. Kod pobrojanih lista

elementi su označeni rednim brojevima, a kod nepobrojanih malih kružićima, kvadratićima ili nečim sličnim (tzv. bulletima). Svaki element obiju ovih lista obuhvaća se oznakom ili . Cijela lista umeće se unutar oznaka i kada se radi o pobrojanoj listi, odnosno unutar oznaka i u slučaju nepobrojane liste. Kod prikazivanja unutar pretraživačkih programa, elementi lista su obično malo uvučeni u odnosu na lijevi rub prozora.

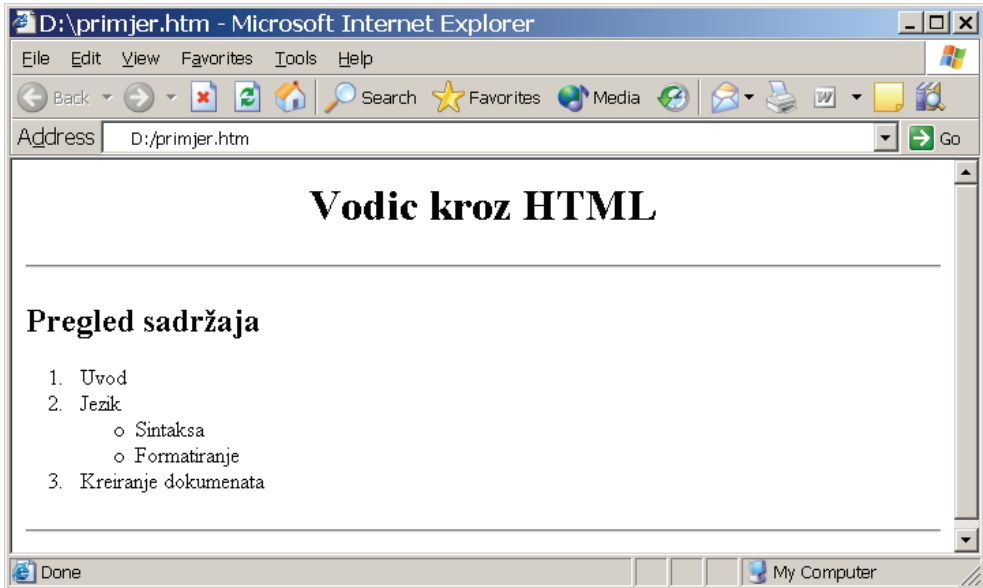
Kod definitivnih lista svaki se element sastoji od dva manja objekta: definirajućeg izraza i same definicije. Izraz koji definiramo mora biti smješten unutar oznaka <DT> i </DT> a pretraživački program će ga ispisati odmah uz lijevu marginu. Definicija se obuhvaća oznakama <DD> i </DD>, a bit će malo odmaknuta od lijevog ruba. Elementi definitivnih lista neće biti označeni niti rednim brojevima, niti kružićima ili kvadratićima.

Liste mogu biti ugniježdene, što ih čini idealnima za prikazivanje različitih sadržaja ili pregleda nekih većih materijala. Osim ugnježđivanja, te hipertekstualnih veza i slika, unutar lista se ne bi smjeli koristiti nikakvi drugi elementi jezika HTML koji će rezultirati stvaranjem praznih redaka koji impliciraju kraj odlomka, poput naslova, vodoravnih linija, tabela i obrazaca, dok je upotreba naredbi namijenjenih stilskom formatiranju prihvatljiva. Evo jednog primjera ugniježdene liste u kojoj se nepobrojana lista nalazi unutar pobrojane.

```
<HTML>
<HEAD>
    <TITLE>Liste</TITLE>
</HEAD>
<BODY>
<H1 ALIGN=CENTER>Vodič kroz HTML</H1>
<HR>
<H2>Pregled sadržaja</H2>
<OL>
    <LI>Uvod</LI>
    <LI>Jezik</LI>
    <UL>
        <LI>Sintaksa</LI>
        <LI>Formatiranje</LI>
    </UL>
    <LI>Kreiranje dokumenata</LI>
</OL>
<HR>
```

```
</BODY>
</HTML>
```

Slika 8 prikazuje ovaj primjer u pretraživačkom programu.



Slika 8. Pobrojana i nepobrojana lista

Uvlačenja pojedinih redaka u prethodnom primjeru korištena su samo zato da bi HTML kod bio što čitljiviji i uredniji; to niti u najmanjoj mjeri ne utječe na način na koji će pretraživački program formatirati tekst. Uočite da se nepobrojana lista nalazi između dva elementa pobrojane liste, a ne unutar nekog od elemenata. Drugim riječima, bilo bi neispravno napisati nešto poput:

```
<LI>
<UL>
<LI>Sintaksa</LI>
<LI>Formatiranje</LI>
</UL>
</LI>
```

Po standardu HTML 2, oznaka `<LI>` ne dolazi u paru sa završnom oznakom `</LI>`, pa ćete na Webu pronaći još mnoštvo dokumenata kojima krajevi elemenata unutar lista uopće nisu označeni. To je sasvim u redu i ne predstavlja nikakvu pogrešku.

I pobrojanim i nepobrojanim listama može se dopisati argument `COMPACT` kojim se smanjuje prazan prostor između pojedinih elemenata u listi, te argument `PLAIN` koji pretraživačkom programu govori da se elementi ne smiju označavati niti rednim brojevima, niti kružićima niti bilo kako drugačije.

Za standard HTML 3 predloženo je uvrštavanje i naredbe `<LH></LH>` koja bi se koristila za navođenje naslova liste, no rijetko koji pretraživački program podržava ovu mogućnost, budući da se isti efekt može postići i uobičajenim naredbama za izradu naslova.

Tvrtka Netscape već je odavno za svoj Netscape pripremila argument `START` pomoću kojega kod pobrojane liste možete navesti broj s kojim će biti označen prvi element liste, drugačiji od jedinice. Prema prijedlogu standarda HTML 3, isti se efekt može postići i argumentom `SEQNUM`. Ukoliko vam je potrebna ova mogućnost jezika HTML, svakako se poslužite Netscapeovim argumentom, jer ga već sada podržava gotovo svaki pretraživački program. Slično je i s argumentom `CONTINUE` pomoću kojega se pretraživačkom programu navodi da listu počne odbrojavati tamo gdje je stao s prethodnom pobrojanom listom.

Netscape je sličan argument uveo i za određivanje načina kojim će se odbrojavati elementi list. Radi se o argumentu `TYPE` koji se može koristiti i kod nepobrojanih lista. Kod pobrojanih lista taj argument može poprimiti vrijednosti navedene u tablici 3, dok se kod nepobrojanih mogu navesti vrijednosti `CIRCLE`, `SQUARE` i `DISC`, kako bi se odredilo hoće li se elementi označavati kružićima, kvadratićima ili tačkama.

Tabela 3: Načini odbrojavanja pobrojane liste

HTML oznaka	Značenje
<code>TYPE="1"</code>	Odbrojavanje običnim brojevima
<code>TYPE="I"</code>	Odbrojavanje velikim rimskim brojevima
<code>TYPE="i"</code>	Odbrojavanje malim rimskim brojevima
<code>TYPE="A"</code>	Abecedno odbrojavanje velikim slovima
<code>TYPE="a"</code>	Abecedno odbrojavanje malim slovima

Kod definitivnih lista ne postoje ograničenja pri upotrebi pojedinih HTML naredbi, kao što je bio slučaj s pobrojanim i nepobrojanim listama, kako kod navođenja definirajućeg izraza, tako i kod navođenja definicije. To definitne liste čini vrlo snažnim elementima jezika HTML, pomoću kojih se mogu postizati najrazličitiji efekti. Pogledajmo najprije jedan jednostavan primjer.

```
<HTML>
<HEAD>
  <TITLE>Popis uloga</TITLE>
</HEAD>
<BODY>
  <H2>Uloge</H2>
  <HR>
  <DL>
```

```

<DT>C.C.</DT><DD>Glava obitelji</DD>
<DT>C.C.</DT><DD>C.C.-eva kćerka</DD>
<DT>Cruise</DT><DD>Muž od Eden</DD>
<DT>Sophia</DT><DD>Cruiseova punica</DD>

</DL>

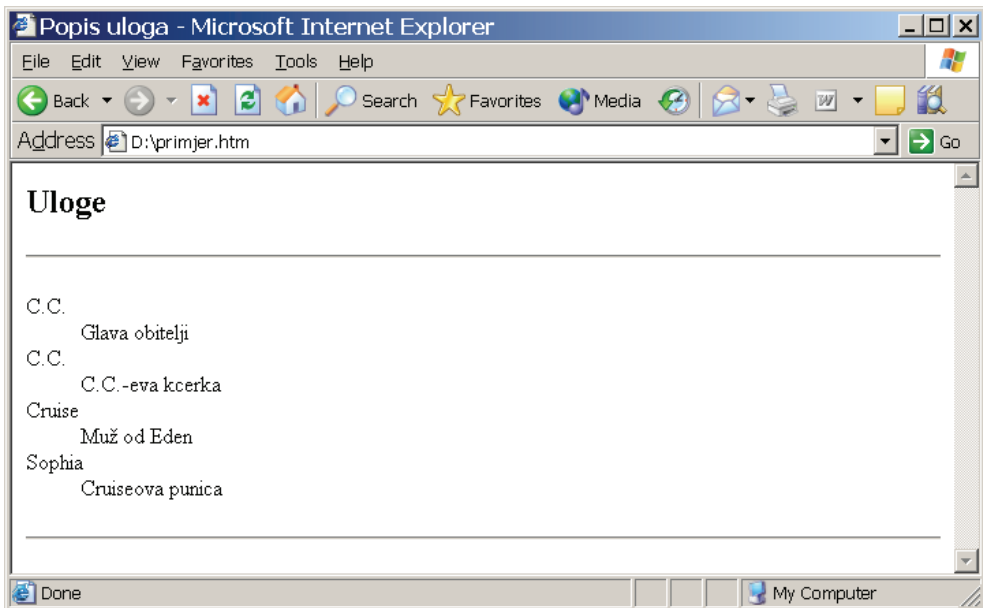
<HR>

</BODY>

</HTML>

```

Slika 9 prikazuje kako ovaj primjer izgleda na ekranu.



Slika 9. Jednostavna definitna lista

Vjerojatno ste odmah primijetili da između definirajućeg izraza i definicije ne postoji prazan prostor koji bi pozitivno utjecao na grafički izgled stranice. Uobičajena tehnika kojom se definirajući izrazi pretvaraju u male naslove prikazana je novim primjerom.

```

<HTML>

<HEAD>

    <TITLE>Operativni sistemi</TITLE>

</HEAD>

<BODY>

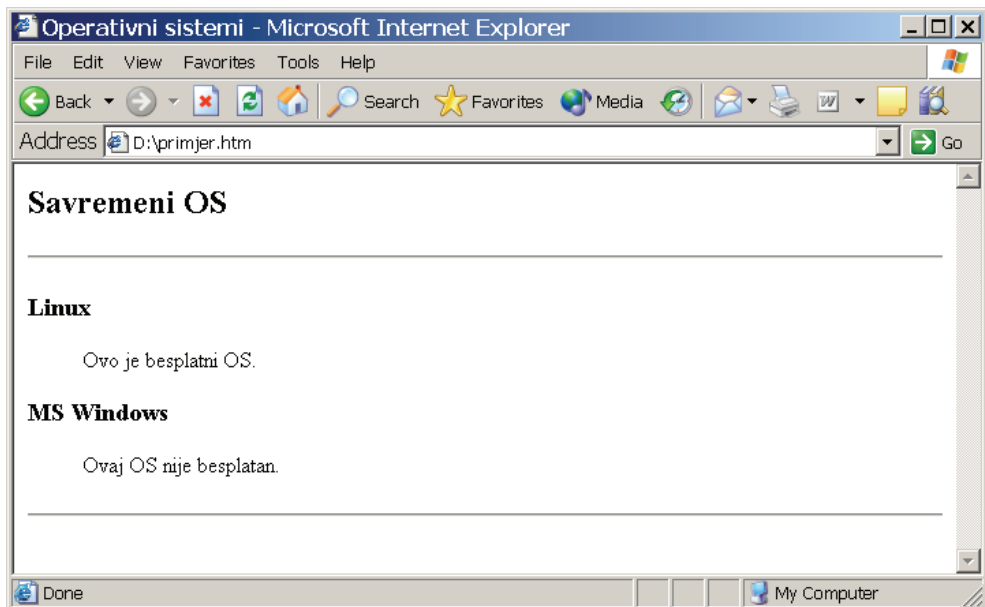
    <H2>Savremeni OS</H2>

```

```

<HR>
<DL>
    <DT>
    <H3>Linux</H3>
    </DT>
    <DD>
    Ovo je besplatni OS.
    </DD>
    <DT>
    <H3>MS Windows</H3>
    </DT>
    <DD>
    Ovaj OS nije besplatan.
    </DD>
</DL>
<HR>
</BODY>
</HTML>

```



Slika 10. Definitna lista s naslovima

Na slici 10 možete pogledati kako izgleda ovaj primjer nakon učitavanja u pretraživački program.

4.6. Tablice

Često ćete se naći u situaciji da neke informacije poželite formatirati u neku složeniju strukturu od one koja se dobiva pomoću raznih vrsta lista. Tablice će vam omogućiti da podatke organizirate u retke i stupce, a uvedene su tek standardom HTML 3.

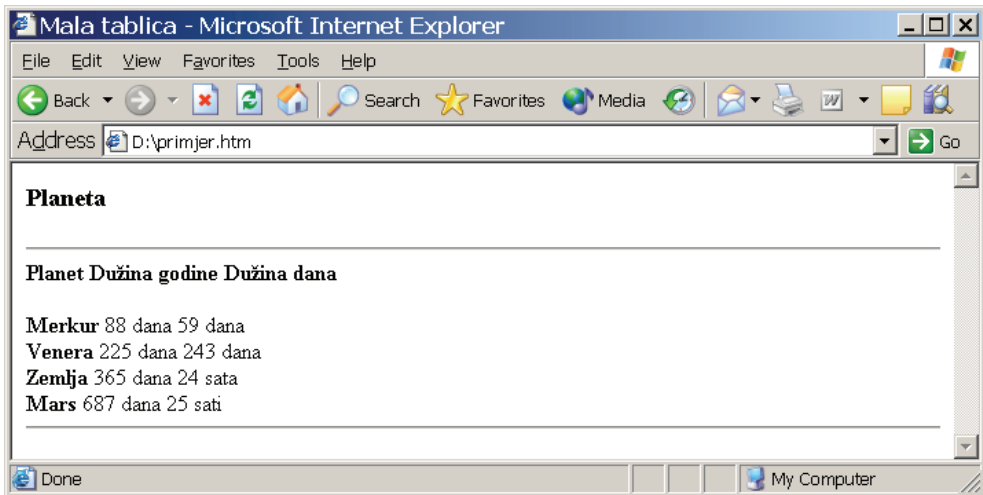
Kod izrade jednostavnih tabela koje većinom sadrže brojčane podatke, najbolje je upotrijebiti naredbu <TAB> kojom se kreiraju tabularni razmaci. Uvrštavanje ove naredbe, u jezik HTML također je predloženo tek standardom HTML 3. Kako biste odredili poziciju na kojoj se mora poravnavati tekst, upotrijebit ćete argument ID kojemu ćete dodijeliti neko simboličko ime. Potom ćete u svakom novom retku elemente tablice poravnavati u određeni stupac pozivajući se na odgovarajuće simboličko ime pomoću argumenta TO. Za ovu naredbu postoje još i argumenti ALIGN i INDENT s kojima se određuje način poravnavanja pojedinog stupca, te razmak između stupaca. Razmak se zadaje u jedinicama koje odgovaraju polovici veličine pisma kojom se ispisuje trenutni tekst. Sve će vam biti puno jasnije kada pogledate sljedeći primjer.

```
<HTML>
<HEAD>
    <TITLE>Mala tabela</TITLE>
</HEAD>
<BODY>
<H3>Planeta</H3>
<HR>
<B>
Planet
<TAB INDENT=30 ID="st1" ALIGN=RIGHT>Dužina godine
<TAB INDENT=30 D="st2" ALIGN=RIGHT>Dužina dana
</B><P>
<B>Merkur</B>
<TAB TO="st1"> 88 dana<TAB TO="st2"> 59 dana<BR>
<B>Venera</B>
<TAB TO="st1"> 225 dana<TAB TO="st2"> 243 dana<BR>
<B>Zemlja</B>
```

```

<TAB TO="st1"> 365 dana<TAB TO="st2"> 24 sata<BR>
<B>Mars</B>
<TAB TO="st1"> 687 dana<TAB TO="st2"> 25 sati<BR>
<HR>
</BODY>
</HTML>

```



Slika 11. Tabela izrađena upotrebom tabulatora.

Slika 11 prikazuje kako bi ovo izgledalo u pretraživačkom programu. Argument ALIGN može osim vrijednosti RIGHT kojom se elementi poravnavaju po desnom rubu stupca, poprimiti i vrijednosti LEFT za lijevo poravnavanje, te CHAR za poravnavanje po zadanom simbolu. Taj simbol se mora dodijeliti kao posebna vrijednost dodatnom argumentu CHAR. Ako, na primjer, želite neki stupac poravnavati po decimalnoj točki, upotrijebiti ćete sljedeću naredbu:

```
<TAB ID="cijena" ALIGN=CHAR CHAR=".">
```

Za izradu složenijih tabela potrebno je upotrijebiti pravu HTML naredbu za tablice, a ne tabulatore. Njome se tablice definiraju kao skupovi redaka od kojih svaki sadrži određen broj kućica. Postoje dvije vrste kućica: one u koje upisujemo obične podatke unutar tablice i one u koje upisujemo naslove, odnosno oznake pojedinih redaka i stupaca. Sadržaj običnih kućica poravnava se po lijevom rubu kućice, dok se naslovi redaka i stupaca centriraju i ispisuju podebljanim slovima. Naravno, dodavanjem odgovarajućih argumenata koji idu uz naredbe namijenjene izradi tabela, to se može promijeniti. Svaka kućica u tablici može sadržavati bilo koje druge HTML elemente – liste, slike, naslove – čak i nove tablice.

Tablicu započinjemo i završavamo oznakama <TABLE> i </TABLE>. Redovi se obuhvaćaju oznakama <TR> i </TR>, dok se za obične kućice i kućice s

naslovima koriste redom parovi naredbi `<TR></TR>` i `<TH></TH>`. Svaka tabela mora sadržavati najmanje jedan redak i jednu kućicu. Pomoću opcionalnog elementa `<CAPTION></CAPTION>` tabelama se mogu dodavati naslovi. Oni će biti prikazani podebljanim slovima, iznad tablice, a bit će i centrirani u odnosu na tablicu. Sljedećim primjerom generira se jednostavna tabela s tri retka i tri stupca:

```
<HTML>
<HEAD>
  <TITLE>Jednostavna tabela</TITLE>
</HEAD>
<BODY>
  <TABLE>
    <CAPTION>Broj dnevnih obroka</CAPTION>
    <TR><TH></TH> <TH>Ručak</TH> <TH>Večera</TH><TR>
    <TR><TH>U kuhinji</TH> <TD>2</TD> <TD>1<TD></TR>
    <TR><TH>U sobi</TH> <TD>1</TD> <TD>3</td></Tr>
  </TABLE>
</BODY>
</HTML>
```

Argument `ALIGN` može se upotrijebiti kako kod naredbe `<TABLE>` tako i kod oznaka `<TR>`, `<TH>` i `<TD>`, a dodjeljuju mu se već standardne vrijednosti `LEFT`, `CENTER` i `RIGHT`, namijenjene lijevom poravnavanju, centriranju i desnom poravnavanju. Taj bi se argument unutar naredbe `<TABLE>` trebao koristiti u kombinaciji s argumentom `WIDTH` kako bi se tačno mogao kontrolirati smještaj tablice unutar stranice. Naredbom:

```
<TABLE ALIGN=CENTER WIDTH=50%>
```

kreirat ćete tablicu koja će biti široka kao polovica prozora s pretraživačkim programom, a nalazit će se tačno na sredini istog prozora. Kod upotrebe s naredbom `<CAPTION>`, argumentu `ALIGN` mogu se dodijeliti vrijednosti `TOP` za smještanje naslova iznad tablice (što pretraživački rade automatski, odnosno `BOTTOM` kada naslov želite smjestiti ispod tablice).

Prilikom upotrebe argumenta `ALIGN` s drugim elementima tablice, poput redaka i obiju vrsta kućica, upravlja se smještajem sadržaja unutar kućica. Dozvoljene vrijednosti su `LEFT`, `CENTER`, `RIGHT`, `JUSTIFY` i `CHAR`. `JUSTIFY` služi poravnavanju po oba ruba kućice (koristi se rijetko) a `CHAR` poravnavanju po određenom znaku. Poravnavanje navedeno kod naredbe za kreiranje kućice ima prednost pred onim koje je navedeno kod naredbe za kreiranje retka, koje opet ima prednost pred poravnavanjem navedenim za cijelu tablicu, kod naredbe `<TABLE>`.

Slično je i s upotrebom argumenta VALIGN. On može poprimiti vrijednosti TOP, MIDDLE i BOTTOM, a namijenjen je okomitom poravnavanju sadržaja unutar kućica u tablici. Ukoliko se argument ne navodi, pretraživački programi će sadržaj poravnavati po sredini okomice. Očito je da se kod složenijih sadržaja i formata tabela argumenti ALIGN i VALIGN moraju upotrebljavati vrlo oprezno.

Naredbi <TABLE> mogu se dodati još tri argumenta, od kojih se svakome dodjeljuje cjelobrojna vrijednost koja predstavlja broj piksela na ekranu. Prvi takav argument je BORDER, kojim se tabela smješta unutar rešetaka čiji vanjski rub ima debljinu dodijeljene vrijednosti. Ukoliko e ovom argumentu ne dodjeli nikakva vrijednost, rub tablice bit će najtanji mogući. Drugi je CELLPADDING, kojim se navodi broj piksela za koji će sadržaj pojedine kućice biti udaljen od ruba tablice, dok je treći CELLSPACING, kojim se određuje razmak između svake pojedine kućice.

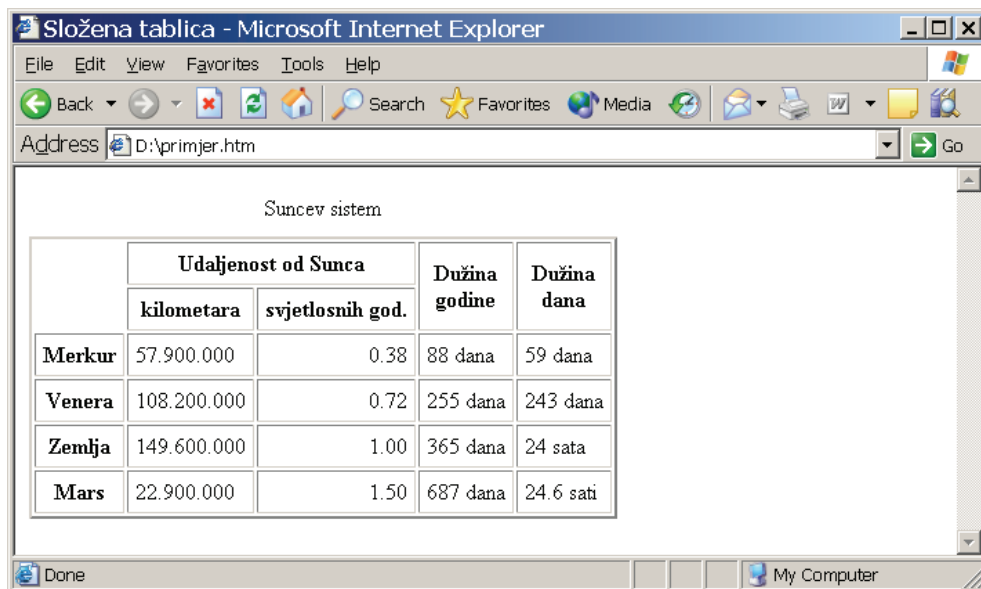
Naredbama <TD> i <TH> opcionalno se dodjeljuju i argumenti ROWSPAN i COLSPAN, koji također poprimaju cjelobrojne vrijednosti. Oni se koriste kod složenih tabela, u slučaju da želite kreirati kućicu koja se visinom (ROWSPAN) i širinom (COLSPAN) proteže kroz područje drugih kućica. One kućice koje su "prekrivene" ovakvom povećanom kućicom, ne smiju se ponovo navoditi upotrebom naredbi <TD> i <TH> unutar odgovarajućih redaka. Najbolje je da to sve pogledate na primjeru.

```
<HTML>
<HEAD>
    <TITLE>Složena tabela</TITLE>
</HEAD>
<BODY>
<TABLE CELLPADDING=5 BORDER=2>
    <CAPTION ALIGN=BETTOM>Sunčev sistem</CAPTION>
    <TR><TH ROWSPAN=2></TH>
    <TH COLSPAN=2>Udaljenost od Sunca</TH>
    <TH ROWSPAN=2>Dužina<BR>godine</TH>
    <TH ROWSPAN=2>Dužina<BR>dana</TH>
</TR><TR>
    <TH>kilometara</TH>
    <TH>svjetlosnih god.</TH>
</TR><TR>
    <TH>Merkur</TH> <TD>57.900.000</TD>
    <TD ALIGN=RIGHT>0.38</TD>
```

```

        <TD>88 dana</TD> <TD>59 dana</TD>
    </TR><TR>
        <TH>Venera</TH> <TD>108.200.000</TD>
        <TD ALIGN=RIGHT>0.72</TD>
        <TD>255 dana</TD> <TD>243 dana</TD>
    </TR><TR>
        <TH>Zemlja</TH> <TD>149.600.000</TD>
        <TD ALIGN=RIGHT>1.00</TD>
        <TD>365 dana</TD> <TD>24 sata</TD>
    </TR><TR>
        <TH>Mars</TH> <TD>22.900.000</TD>
        <TD ALIGN=RIGHT>1.50</TD>
        <TD>687 dana</TD> <TD>24.6 sati</TD>
    </TR>
</TABLE>
</BODY>
</HTML>

```



Slika 12. Složena tabela

Standard HTML 3 dozvoljava upotrebu još nekih dodatnih argumenata koji se mogu koristiti s naredbom <TABLE>:

- **FLOAT** Ovom argumentu mogu se dodijeliti vrijednosti **LEFT** i **RIGHT** kako bi se odredilo hoće li tekst koji slijedi iza kraja tablice označenog s naredbom `</TABLE>` biti ispisan s lijeve ili s desne strane u odnosu na tablicu. Kada se ovaj argument ne upotrebljava, pretraživački programi smještaju tekst ispod tablice.
- **COLS?** Određivanje broja stupaca u tablici. Znajući unaprijed ovaj broj, pretraživački programi mogu brže odrediti izgled tablice.
- **FRAME** Vrijednosti za ovaj argument su **TOP**, **BTTOM**, **TOPBOT**, **SIDES** i **ALL**, a njime se određuje hoće li tabela imati okvir samo na gornjem rubu, donjem rubu, oba ruba (i gornji i donji, obje stranice (lijevoj i desnoj), ili posvuda.

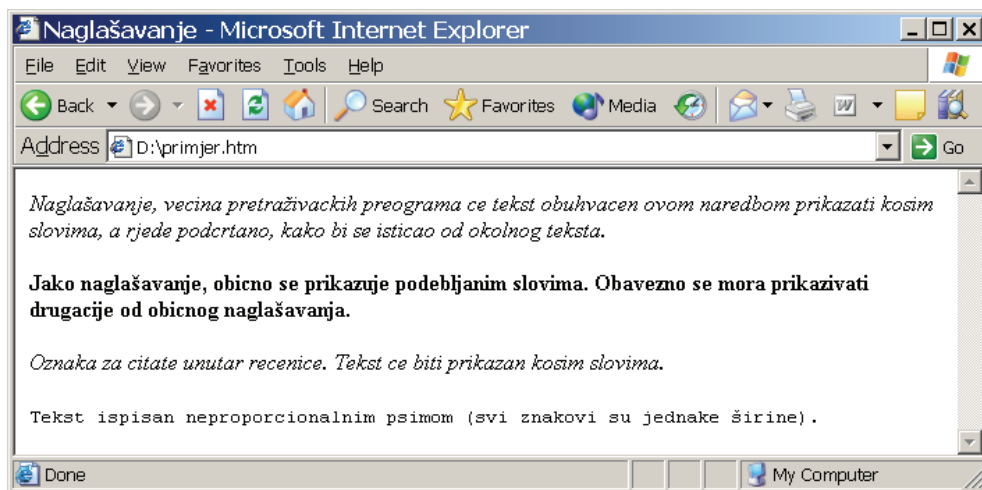
Tablice mogu imati zaglavlje i tijelo, koji se odvajaju parovima oznaka `<THEAD></THEAD>` i `<TBODY></TBOD>>`. Naredbe su korisne samo kod jako velikih tabela. Zaglavlje će biti prikazano na ekranu uvijek, bez obzira na to što cijela tabela ne stane u prozor pretraživačkog programa, dok će se tijelo pomicati u odnosu na pomicanje cijele stranice. Kod tabela koje cijele stranu na ekran, razlika između zaglavlja i tijela je neprimjetna. Obje naredbe mogu se koristiti s argumentima **ALIGN** i **VALIGN**.

4.7. Stilovi

Stilskim naredbama mijenja se grafički izgled sadržanog teksta. Takve naredbe se smiju ugnježdjivati jedne unutar drugih slično kao što u nekom tekstnom procesoru možete na dijelove rečenica primijeniti više stilskih promjena. HTML stilovi su, međutim, apstraktni i logično konstruirani, dok su stilovi koji se upotrebljavaju kod tekstnih procesora konkretni. Većina drugih naredbi kojima se utječe na strukturu dokumenta slobodno se može pojaviti unutar stilskih naredbi, premda bi takve situacije trebalo izbjegavati. Najčešće korištene stilske naredbe koje prepoznaju svi pretraživački programi su sljedeće:

- `<EM></EM>` Naglašavanje, većina pretraživačkih programa će tekst obuhvaćen ovom naredbom prikazati kosim slovima, a rjeđe podcrtano, kako bi se isticao od okolnog teksta.
- `<STRONG></STRONG>` Jako naglašavanje, obično se prikazuje podebljanim slovima. Obavezno se mora prikazivati drugačije od običnog naglašavanja.
- `<CITE></CITE>` Oznaka za citate unutar rečenice. Tekst će biti prikazan kosim slovima.
- `<TT></TT>` Tekst ispisan neproporcionalnim pismom (svi znakovi su jednake širine).

Na slici 13 vide se rezultati upotrebe ovih naredbi na gornjem tekstu. Pokušajte sami sastaviti takvu Web stranicu.



Slika 13. Tekstualni stilovi

Tabela 4. Različiti logički stilovi

HTML oznaka	Značenje
<ABBREV></ABBREV>	Oznaka za skraćenicu
<ACRONYM></ACRONYM>	Oznaka za akronime poput SIPA, FTP, FBI...
<AU></AU>	Autor teksta, odnosno Web stranice
<BIG></BIG>	Tekst ispisan nešto većim pismom od uobičajenog
	Oznaka za izbrisan tekst koji je "preživio" u dokumentu
<INS></INS>	Novi tekst ubačen u stari sadržaj
<PERSON></PERSON>	Naznačavanje imena neke osobe
<Q></Q>	Tekst koji se nalazi između znakova navoda
<S></S>	Precrtan tekst, naredba skraćuje oznaku <STRIKE></STRIKE>
<SMALL></SMALL>	Tekst ispisan nešto manjim pismom od uobičajenog
	Oznaka za indekse i drugi niže ispisan tekst
	Oznaka za eksponente i drugi više ispisan tekst
<CODE></CODE>	Oznaka za kod ispisan nekim programskim jezikom
<VAR></VAR>	Oznaka za varijablu, parametar, argument i slično
<KBD></KBD>	Tekst koji korisnik mora utipkati na tipkovnici
<SAMP></SAMP>	Niz znakova kojima se nešto nabraja

HTML ipak sadrži i neke konkretne stilove koji se dobivaju naredbama `<B></B>` za masno otisnut tekst, `<I></I>` za tekst ispisan kosim slovima i `<U></U>` za podcrtan tekst. Kada god je to moguće, trebali biste izbjegavati upotrebu ovih naredbi i koristiti logičke stilove za naglašavanje teksta, posebno ako radite s dokumentima na kojima povremeno radi više autora, kako ne bi došlo do miješanja stilova. S druge strane, ako ste vi jedini autor nekog dokumenta i ako želite biti sigurni da će se neki tekst prikazati kosim ili podebljanim slovima, slobodno upotrijebite naredbe za konkretne stilove.

Netscape Navigator prepoznaje i naredbu `<FONT>` kojom se može promijeniti veličina pisma. Ova naredba uvijek dolazi s argumentom `SIZE`, kojemu se može dodijeliti pozitivna ili negativna vrijednost od 1 do 7 kako bi se odredilo za koliko jedinica pismo treba biti veće ili manje od osnovnog kojim se ispisuje običan tekst. Ovo je sjajan način za falsificiranje teksta ispisanog umanjenim velikim slovima. Evo primjera:

```
<FONT SIZE=+3>Z</FONT>GODAN TRI<FONT SIZE=+3>K</FONT>
```

4.8. Hipertekstualne veze

Hipertekstualne veze predstavljaju dio stranice – to su oni plavi podcrtani dijelovi teksta na koje možete kliknuti mišem kako biste se odveli negdje drugdje. Te veze međusobno povezuju Web stranice. Naredba `<A>` kojom se kreiraju hipertekstualne veze uvijek se piše s argumentom `HREF`, kojemu se dodjeljuje URL adresa ciljanog dokumenta. Osnovna sintaksa je, prema tome, sljedeća:

```
<A HREF="neka-URL-adresa">Otiđi na neki URL</A>
```

U slučaju da se i početak i kraj hipertekstualne veze nalaze unutar istog dokumenta, što je ujedno i najjednostavniji oblik povezivanja na Webu, vrijednost koja se dodjeljuje argumentu `HREF` je ime odredišta kojem prethodi znak `#`. Na primjer:

```
Adresa <A HREF="#izvori">dodatni izvor</A> možete  
pronaći na kraju ovog poglavlja.
```

*

*

*

```
<P ID="izvori">
```

Evo i nekih dodatnih izvora informacija...

Stariji pretraživački programi neće prepoznati argument `ID` koji možete nadopisati uz gotovo svaku naredbu, pa je zato puno pametnije upotrijebiti argument `NAME` koji se uvijek dopisuje uz naredbu `<A>`.

```
<P>
```

```
<A NAME?"izvori">Evo i nekih dodatnih izvora  
informacija... </A>
```

Klikom miša na tekst "dodatnih izvora" prebacit ćete se u pretraživačkom programu na onaj dio dokumenta koji je imenovan odgovarajućim imenom pomoću argumenta ID ili NAME. Tekst koji je obuhvaćen naredbom <A> uz koju je umjesto argumenta HREF napisan argument NAME neće biti prikazan kao hipertekstualna veza u pretraživačkom programu, već kao običan tekst.

Kada želite kreirati hipertekstualnu vezu na dokument koji se nalazi u istom direktoriju kao i trenutni dokument, dovoljno je argumentu HREF dodijeliti ime željene datoteke. To se naziva relativnim URL adresiranjem, a primjenjuje se svaki put kada je početak i kraj hipertekstualne veze nalaze na istom serveru. Evo kako možete kreirati jednu takvu vezu.

```
Moj pas se zove <A HREF="lisa.html">Lisa</A>.
```

Upotrebom relativnog adresiranja vaš cjelokupni hipertekstualni rad ostaje kompaktniji, budući da svaki put kada stranice selite na novu lokaciju nećete morati mijenjati HTML kod kako biste popravili hipertekstualne veze. Naravno, to vrijedi samo u slučaju kada ne mijenjate strukturu direktorija, odnosno kada je sadržaj svakog direktorija na staroj lokaciji istovjetan onome na novoj. Da biste kreirali hipertekstualnu vezu na neku određenu točku unutar drugog dokumenta u istom direktoriju, ponovno ćete upotrijebiti ime te tačke koje je u ciljanom dokumentu označeno argumentom ID ili NAME:

```
<A HREF="lisa.html#hrana">Šta Lisa jede?</A>
```

Pretpostavimo da se datoteka koju ciljate nalazi u poddirektoriju direktorija koji sadrži trenutni dokument. Tada ćete u HTML dokumentu ovako kreirati hipertekstualnu vezu:

```
<A HREF="psi/lisa.html">Lisa</A>
```

Nije važno na koji su način konstruirane putanje, direktoriji i imena datoteka na operativnom sistemu pod kojim je pokrenut vaš Web server – svedjedno je koriste li se obrnute kose crte (\) kao u Windowsima, ili uglate zagrade ([,]) kao na VMS sistemima- URL sintaksa uvijek koristi obične kose crte za ovu namjenu (/). Server će se sam pobrinuti o prevođenju ove sintekse u onu koja se upotrebljava na tom operativnom sistemu.

Ako želite kreirati hipertekstualnu vezu na datoteku koja se nalazi na drugom Web serveru, morat ćete argumentu HREF dodijeliti njezinu apsolutnu URL adresu. Format URL adresiranja dopušta vam da pristupate gotovo svim objektima na Internetu kroz vašu Web stranicu, pa tako možete kreirati hipertekstualne veze na Gopher servere, Usenet novinske grupe, FTP arhive, e-mail adrese i drugo. Evo i jednog primjera:

```
<A HREF="http://unze.ba/pf/ispiti.html">IS</A>
```

Ovdje smo upotrijebili cijelu, apsolutnu URL adresu, a ne relativnu. Jedina pretpostavka koju moramo imati na umu je da Web server unze.ba radi na portu 80, što je osnovni port za World Wide Web servere. Ako to nije slučaj – recimo, ako unze.ba posluhuje na portu 8080 – potrebno je unutar URL adrese navesti i taj broj:

```
<A HREF="http://unze.ba:8080/pf/ispiti.html">IS</A>
```

Kao što vidite, URL adrese mogu biti prilično dugačke, no kao što znate, pomoću njih možete kreirati i hipertekstualne veze na mnoge druge tačke na Internetu koje nisu Web stranice. U nastavku je dano nekoliko primjera pomoću kojih možete kreirati veze s FTP arhivom, Telnet serverom i novinskom grupom na Usenetu.

```
<A HREF="ftp://avalon.irb.hr/pub">Shareware  
programi</A>
```

```
<A HREF="telnet://compuserve.com">Servis CompuServe</A>
```

```
<A HREF="news:hr.comp.literature">Domaća literatura</A>
```

Ako URL adresa za neki FTP server završava s imenom konkretne datoteke, tada će se prebaciti na korisnikov računar i ovisno o njezinom tipu prikazati u prozoru pretraživačkog programa ili pohraniti na tvrdi disk. Ako adresa pokazuje na neki direktorij, tada će se u pretraživačkom programu prikazati sadržaj tog direktorija, uključivši sve datoteke i poddirektorije koji su sadržani. Kod kreiranja hipertekstualne veze koja vodi do FTP servera, naredbi `<A>` možete nadopisati i argument `TITLE` kojemu ćete dodijeliti naslov koji želite da korisnik vidi na gornjem rubu prozora s pretraživačkim programom.

4.9. Slike

Slika ili dvije, ugrađene u HTML dokument, poprilično će pridonijeti atraktivnosti vaše Web stranice. Slike koje se nalaze na vašoj početnoj stranici dat će korisniku informacije koje se ne mogu pročitati iz teksta; jednostavni linearni grafikon u svakom je slučaju puno informativniji od tablice prepune brojeva. Slike će na vašoj Web stranici odigrati vrlo veliku funkcionalnu, vizualnu i informativnu ulogu, pa ih ni u kom slučaju ne biste smjeli izostaviti.

Da biste unutar svoje stranice ugradili sliku, upotrijebite naredbu `<IMG>`. Oko slike se neće pojaviti nikakvo razdvajanje odlomaka ili dodatni prazan prostor. Ako tok teksta oko slike nije potrebno određen nekim argumentima, ona će biti prikazana unutar rečenice jednako kao da se radi o nekom slovu koje ima malo čudnu veličinu. Za razliku od rada u programima za stono izdavaštvo gdje se za svaku sliku zna tačna pozicija, unutar Web stranice slika predstavlja najobičniji dio teksta unutar kojega je ugrađena. Na bilo koje mjesto gdje možete smjestiti neko slovo, moći ćete smjestiti i sliku.

Uz naredbu `<IMG>` mogu se dopisati argumenti. Prvi je `SRC`, a valja mu dodijeliti URL: adresu datoteke u kojoj je pohranjena slika koju želite prikazati na Web stranici. URL adresa navodi se na isti način kao i kod hipertekstualnih veza:

relativno ako se slika nalazi na istom WEB serveru kao i dokument, odnosno apsolutno ako je smještena negdje drugdje na Internetu. Argument SRC je obavezan, pa ga morate navesti kod svake upotrebe naredbe .

Argumentom ALT navodi se tekst koji će biti prikazan umjesto slike u pretraživačkim programima s tekstualnim korisničkim sučeljem poput Lynxa, te u slučaju da je u grafičkom pretraživačkom programu isključeno automatsko prikazivanje slika. Sljedećim primjerom prikazano je kako se slike ugrađuju u HTML dokument.

```
<HTML>

<HEAD>

    <TITLE>Slika</TITLE>

</HEAD>

<BODY>

<H1>Slika na Web stranici</H1>

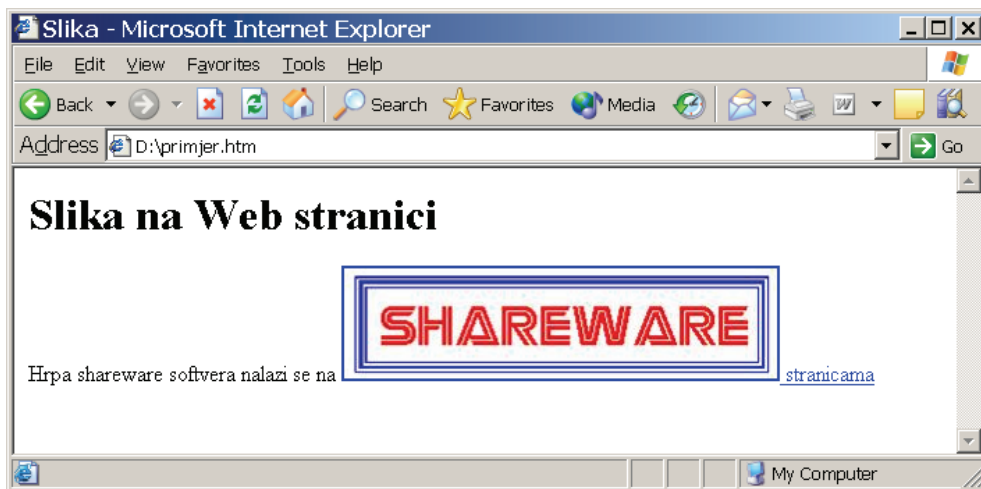
Hrpa shareware softvera nalazi se na

<A HREF="http://pubwww.srce.hr/shareware/">

<IMG SRC="croshare.gif" ALT="Croatia Shareware">
stranicama</A>

</BODY>

</HTML>
```



Slika 14. Slika na Web stranici

Na slici 14 možete pogledati kako izgleda ovaj primjer učitao u pretraživački program. Vjerojatno ste odmah primijetili da je slika iskorištena i kao polazište za hipertekstualnu vezu; jednostavno smo je zajedno s tekстом "stranicama"

obuhvatili naredbom <A>. Sjetite se: sliku možete postaviti svugdje gdje možete postaviti i tekst, pa prema tome i unutar oznake za kreiranje hipertekstualnih veza. Kada korisnik klikne mišem na ovu sliku, učitat će se stranica s URL adrese navedene unutar naredbe <A>.

Važno je napomenuti da se slike na Webu mogu prikazivati samo ako su pohranjene u GIF ili JPEG formatu datoteka, jer su to jedini formati koje podržavaju svi pretraživački programi s grafičkim korisničkim sučeljem: To, međutim, ne znači da, ako vi postavite hipertekstualnu vezi na neki drugi grafički format poput TIFF, PCX ili BMP zapisa, korisnik neće znati što da napravi s tom datotekom. On će je moći prebaciti na svoje računare i kasnije pogledati u nekom programu koji podržava taj zapis, ili će u svojem pretraživačkom programu konfigurirati pomoćnu aplikaciju koja će taj program pokrenuti svaki put kada se na nekoj stranici pojavi nepoznati format.

Prikazivanje slika kao dio teksta i nije baš nešto posebno, pa zato postoji argument, ALIGN kojim se može odrediti način prikazivanja teksta koji se nalazi oko slike. Osnovne tri vrijednosti koje se mogu dodijeliti ovom argumentu su TOP, MIDDLE i BOTTOM, što znači da se tekst koji se nalazi ispred i iza slike može poravnati uz gornji rub slike, uz njezinu sredinu i uz njezino dno.

U slučaju kada je slika viša nego šira, zasigurno ćete htjeti da tekst normalno teče u njezine lijeve ili desne strane, a ne da ona predstavlja dio tog teksta. U tu svrhu će vam poslužiti Netscapeova proširenja za naredbu kojima se argumentu ALIGN mogu dodijeliti još i vrijednosti LEFT i RIGHT. U prvom slučaju slika će biti prikazana uz lijevi rub prozora s pretraživačkim programom, dok će tekst koji je u HTML dokumentu smješten iza naredbe za sliku biti smješten s njezine desne strane. U drugom slučaju će situacija, dakako, biti obrnuta. Za naredbu ne postoji završna naredba , pa ćete, u trenutku kada zaželite tekst ponovo ispisivati po cijeloj širini prozora, morati upotrijebiti naredbu
 uz koju ćete dopisati argument CLEAR, koji je objašnjen ranije u ovom poglavlju. Ako ste argumentu ALIGN kod naredbe dodijelili vrijednost LEFT, istu ćete vrijednost dodijeliti i argumentu CLEAR kod naredbe
. Slično ćete postupiti i u slučaju da ste koristili vrijednost RIGHT. Cijelu ovu zbirku oko poravnavanja slika i teksta najlakše je razjasniti jednim primjerom.

```
<HTML>
<HEAD>
  <TITLE>Slike</TITLE>
</HEAD>
<BODY>
  <B>ALIGN=TOP</B><BR>
  <IMG SRC="image002.jpg" ALIGN=TOP>Shareware<P>
  <B>ALIGN=MIDDLE</B><BR>
```

```
<IMG SRC="image002.jpg" ALIGN=MIDDLE>Shareware<P>
<B>ALIGN=BOTTOM</B><BR>
<IMG SRC="image002.jpg" ALIGN=BOTTOM>Shareware
<HR>
<IMG SRC="image002.jpg" ALIGN=LEFT>
<B>ALIGN=LEFT<B/><BR>
Lorem ipsum dolor sit amet, consetetur sadipscing
elit, sed diam nonumy eirmod tempor invidunt ut labore
et dolore magna aliquyam erat, sed diam voluptua. At
vero eos et accusam et justo duo dolores et ea rebum.
Stet clita kasd gubergren, no sea takimata sanctus est
Lorem ipsum dolor sit amet.

<BR CLEAR=LEFT>
<HR>
<IMG SRC="image002.jpg" ALIGN=RIGHT>
<B>ALIGN=RIGHT<B/><BR>
Lorem ipsum dolor sit amet, consetetur sadipscing
elit, sed diam nonumy eirmod tempor invidunt ut labore
et dolore magna aliquyam erat, sed diam voluptua. At
vero eos et accusam et justo duo dolores et ea rebum.
Stet clita kasd gubergren, no sea takimata sanctus est
Lorem ipsum dolor sit amet.

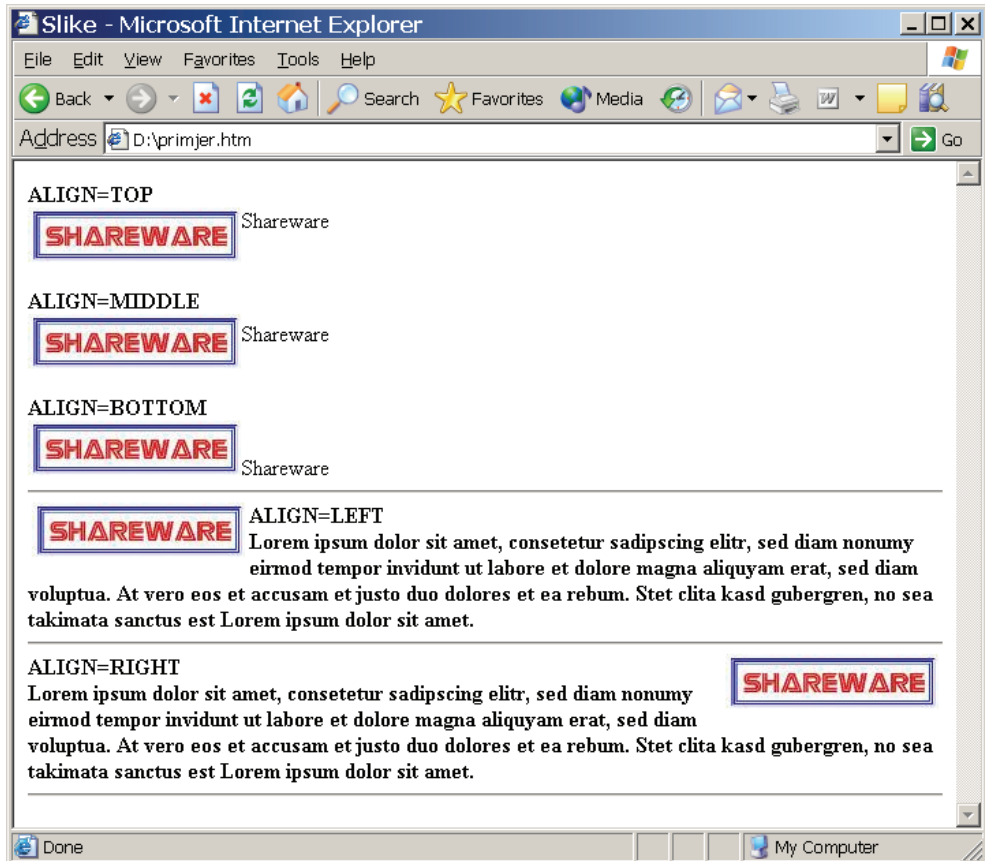
<BR CLEAR=RIGHT>
<HR>
</BODY>
</HTML>
```

Slika 15 prikazuje ovaj primjer u pretraživačkom programu.

Za dodatnu kontrolu nad smještajem slika, tvrtka Netscape je smislila i argumente HSPACE i VSPACE. Tim se argumentima dodjeljuje numerička vrijednost kojom se određuje broj piksela za koji će slika biti udaljena od teksta što je okružuje. Argumentom HSPACE određuje se razmak s lijeve i s desne strane slike, a argumentom VSPACE s gornje i s donje strane. Općenito govoreći, što je veća slika, ovim ćete argumentima htjeti dodijeliti veću vrijednost kako bi stranica izgledala grafički ukusno. Uobičajene vrijednosti kreću se od pet do petnaest piksela.

Argumentu BORDER koji se također može dodijeliti naredbi , pridružuje se brojčana vrijednost kojom se u pikselima određuje debljina okvira za neku sliku. Dakako, upotreba ovog argumenta ima smisla samo kada se slika nalazi u ulozi hipertekstualne veze, budući da u suprotnom uopće neće biti uokvirena. Ukoliko ne

želite da vaša slika ima okvir čak niti onda kada predstavlja početak hipertekstualne veze, što je korisno u slučajevima kada svojim sadržajem navodi korisnika da na nju klikne, argumentu BORDER ćete dodijeliti vrijednost 0.



Slika 15. Upotreba argumenta ALIGN za razmještanje slika

Posljednja dva argumenta koja ćemo spomenuti također su Netscapeova proširenja jezika HTML. Radi se o argumentima HEIGHT i WIDTH kojima se također dodjeljuje vrijednost u pikselima, no njihova uloga ne utječe na izgled stranice i smještaj slika u odnosu na tekst, već na brzinu i način učitavanja cijele stranice u pretraživački program. Ako uz naredbu dopišete i ova dva argumenta, te im kao vrijednost dodijelite redom visinu i širinu slike u pikselima, pretraživački program će nastaviti s formatiranjem stranice za vrijeme prenošenja slike s serverskog na korisnikov računar. Korisnik će zahvaljujući tome moći brže pročitati sav tekst koji je sadržan u nekom HTML dokumentu i potom se vratiti i pogledati one slike koje ga zanimaju. Ukoliko se ova dva argumenta ne navedu, pretraživački program će nastaviti s formatiranjem stranice tek kada veličinu slike dozna čitajući prve dijelove GIF ili JPEG datoteke.

Argumentima HEIGHT i WIDTH valja dodijeliti tačnu visinu i širinu slike u pikselima, jer će Netscape u protivnom sliku automatski povećati ili smanjiti, već prema tome jeste li unijeli veće ili manje vrijednosti od stvarnih. Kako za to mijenjanje veličine Netscape ne koristi baš neke savršene algoritme, poslužite se nekom grafičkom aplikacijom poput Corel PhotoPainta ili Adobe PhotoShopa ako želite promijeniti format slike.

4.9.1 Slikovne mape

Slikovne mape predstavljaju važno proširenje koncepta hipertekstualnog povezivanja dokumenata. Takve mape su zapravo obične slike na kojima su definirana određena područja na koja korisnik može kliknuti i učitati stranicu s nove URL adrese. Ovdje ćemo objasniti dva načina na koji se mogu kreirati slikovne mape: prvi je pomoću standardnih metoda koje su uvedene standardom HTML 2, a drugi je pomoću Netscapeovih proširenja jezika HTML koja su uvedena Navigatorom 2.0. U prvom slučaju govorimo o serverskim slikovnim mapama, budući da se zahtjeva prisutnost posebnog programa na serverskom računar, dok u drugom govorimo o klijentskim mapama, jer one ne zahtijevaju nikakvo procesiranje na serveru već se u potpunosti interpretiraju unutar pretraživačkog programa. S korisničkog stanovišta, obje vrste slikovnih mapa izgledaju i djeluju podjednako, no sa stanovišta autora HTML dokumenta, radi se o dva vrlo različita procesa.

Upotrebljava se argument USEMAP kojemu se dodjeljuje ime tačke unutar istog dokumenta, koja je označena s argumentom NAME. Imenu kod argumenta USEMAP mora prethoditi znak #, slično kao i kod hipertekstualnih veza kojima se skače do određene tačke u nekom ili istom dokumentu. Argument NAME dopisuje se uz naredbu <MAP> kojom se definiraju područja slikovne mape, a koju valja smjestiti na kraju tijela HTML dokumenta. Pojedino područje na slici označava se naredbom <AREA>. U nastavku predočujemo primjer koji će vam razjasniti o čemu se radi:

```
<IMG SRC="slike/putokaz.gif"
ALT="ovo je slikovna mapa" USEMAP="#putokaz">
*
*
*
<MAP NAME="putokaz">
<AREA SHAPE="POLY" HREF="diskete.html"
COORDS="21,68,5,82,18,97,132,93,,131,62,21,68>
<AREA SHAPE="POLY" HREF="kritike.html"
COORDS="10,100,8,133,126,136,142,121,124,103,10,100>
```

```
<AREA SHAPE="POLX" HREF="sadržaj.html"  
COORDS="8,28,9,62,129,51,137,34,129,25,102,22,8,28">  
<AREA SHAPE="POLY HREF="ocjena.html"  
COORDS="4,154,18,171,130,175,131,145,22,140,4,154">  
<AREA SHAPE="POLY" HREF="znak.html"  
COORDS="12,181,12,213,124,211,144,194,125,179,12,181">  
</MAP>
```

Nakon malo detaljnijeg proučavanja ovog primjera nije teško zaključiti da se tip geometrijskog lika određuje pomoću argumenta SHAPE, URL adresa ciljanog dokumenta argumentom HREF, a koordinate područja odvojene zarezom argumentom COORDS. Svi argumenti se, naravno, dopisuju uz naredbe AREA kojih ima onoliko koliko ima i ključnih područja u slikovnoj mapi. Argument SHAPE može osim vrijednosti POLY poprimiti i vrijednost RECT CIRCLE i DEFAULT kako biste kreirali područja u obliku pravokutnika i kružnice, odnosno odredili URL adresu dokumenta koji će se uočiti ako korisnik klikne na dio slike koji nije obuhvaćen niti jednim područjem.

4.10. Multimedija

Kako biste ugradili multimedijalni objekt unutar svoje Web stranice, bilo da se radi o nekom zvuku, filmu, kombinaciji filma i zvuka ili nečemu drugom, jednostavno kreirajte običnu hipertekstualnu vezu koja će vaš dokument povezati s binarnom datotekom koja sadrži taj objekt. Na korisniku je da ispravno konfigurira svoj pretraživački program i pomoćne aplikacije ili plug-in module kojima ga nadograđuje kako bi mogao upotrijebiti taj objekt na svojem računar.

Običaj je, ali i dio uljudnog ponašanja, upozoriti korisnika da se radi o hipertekstualnoj vezi koja nije uobičajena. Svakako navedite o kojem tipu datoteke se radi, te kolika je njezina veličina u bajtima, kako bi korisnik mogao procijeniti koliko je vremena potrebno da se ta datoteka prebaci do njegovog računara. Najuoobičajeniji tipovi zvučnih zapisa koji se koriste na Webu su AU i SND formati datoteka, no nerijetko ćete naići i na WAV ili MP3 zvučne zapise. Što se tiče videozapisa, najuoobičajeniji formati su QuickTime i MPEG, a u novije vrijeme WMV i AVI (Microsoftovi video formati).

Evo i jednog primjera kreiranja hipertekstualne veze kojom se korisnik upućuje na zvučnu datoteku:

```
<A HREF=" ../audio/lordi.mp3">  
<IMG SRC="zvučnik.gif" ALT="Hard rock, zvučni zapis">  
Hard rock</A> (150 Kb, mp3 format).
```

Uvijek je zgodno kod multimedijalnih objekata unutar hipertekstualne veze postaviti i sličicu koja simbolizira tip zapisa o kojem se radi, kako bi se korisniku omogućilo da odmah stekne dojam o tome što mu se nudi, te da mu se omogući da klikom na sličicu započne s prebacivanjem datoteke na svoj računar: Ukoliko ste unutar svoje stranice ugradili zvučni zapis, postavite u dokumentu, naprimjer, sličicu malog zvučnika, a u slučaju videozapisa sličicu male kamere. Možete se poslužiti i nekim drugim simbolom, sve dok ste korisniku dovoljno informativni i zanimljivi.

4.11. Obrasci

Interaktivni obrasci omogućavaju vam da od korisnika dobavite neke informacije. Obrazac na Web stranici je vrlo sličan onim pretplatničkim kartonima koji se dobivaju s raznim časopisima. Korisnik upisuje informacije u polja koja su sadržana unutar obrasca. Kada završi, sadržaj obrasca se šalje na procesiranje; bilo na neku e-mail adresu, bilo skriptu na serveru. Slanje sadržaja serveru ima neke prednosti – informacije se mogu provjeriti i procesirati u realnom vremenu, te se može poslati prilagođeni automatizirani odgovor koji će kreirati skripta na serveru. Ako sadržaj obrasca šaljete na e-mail adresu, velika prednost je da ne morate kreirati serverske programe koje će obrađivati taj sadržaj. Ti programi, poznati kao CGI skripte, mogu se pisati u mnogim programskim jezicima – Perlu, TCLu, C-u, AppleScriptu, UNIX Shell – već prema tome koji operativni sistem koristite za posluživanje svojih Web stranica i koje programske jezike poznajete. Da bi se kreirale CGI skripte, potrebno je dakle poznavati neke osnove programiranja. Stoga će prvi primjer koji ćemo obraditi u ovom poglavlju ipak koristiti slanje sadržaja obrasca na e-mail adresu.

Obrazac je zasebni dio HTML stranice, često prikazan kao jedna cjelina ili odvojena stranica. On može sadržavati različite tipove polja u koje ubrajamo okvire i polja za unos teksta, padajuće izbornike, kružne izbornike i kontrolne kvadratiće. Unutar jedne stranice može se nalaziti proizvoljan broj obrazaca od kojih je svaki obuhvaćen s oznakama <FORM> i </FORM>. Početna oznaka za obrazac mora sadržavati argument ACTION kojim se određuje što će se dogoditi s informacijama koje je korisnik upisao. Tom se argumentu dodjeljuje URL adresa CGI skripta na serveru ili "mailto" URL adresa, kao u ovom primjeru:

```
<FORM METHOD=POST ACTION="mailto:v.predstavnik@gov.ba">
```

Da bi ovo funkcionisalo, korisnik mora imati ispravno konfigurisan E-mail klijentski softver. Sljedeći primjer prikazuje jednostavan obrazac kojim se od korisnika traži da unese komentar.

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>Jednostavni obrazac</TITLE>
```

```
</HEAD>

<BODY>

<H1>Hoćemo komentare!!!</H1>

Zanima nas vaše mišljenje o ovim stranicama.
Upotrijebite sljedeći obrazac kako biste nam poslali
bilo kakav komentar.

<P>

<FORM METHOD=POST ACTION="mailto:v.predstavnik@gov.ba">
Upišite svoje ime:
<INPUT TYPE="TEXT" NAME="ime" SIZE=35><BR>
Upišite svoju e-mail adresu:
<INPUT TYPE="TEXT" NAME="e-mail" SIZE=35><BR>
<P><HR><P>

U sljedeći okvir za unos teksta upišite bilo kakav
komentar. Kliknite na tipku <B>Slanje</B> kada
završite.

<P>

<TEXTAREA NAME="komentar" ROWS=6 COLS=45></TEXTAREA>

<P>

Zahvaljujem na komentaru.

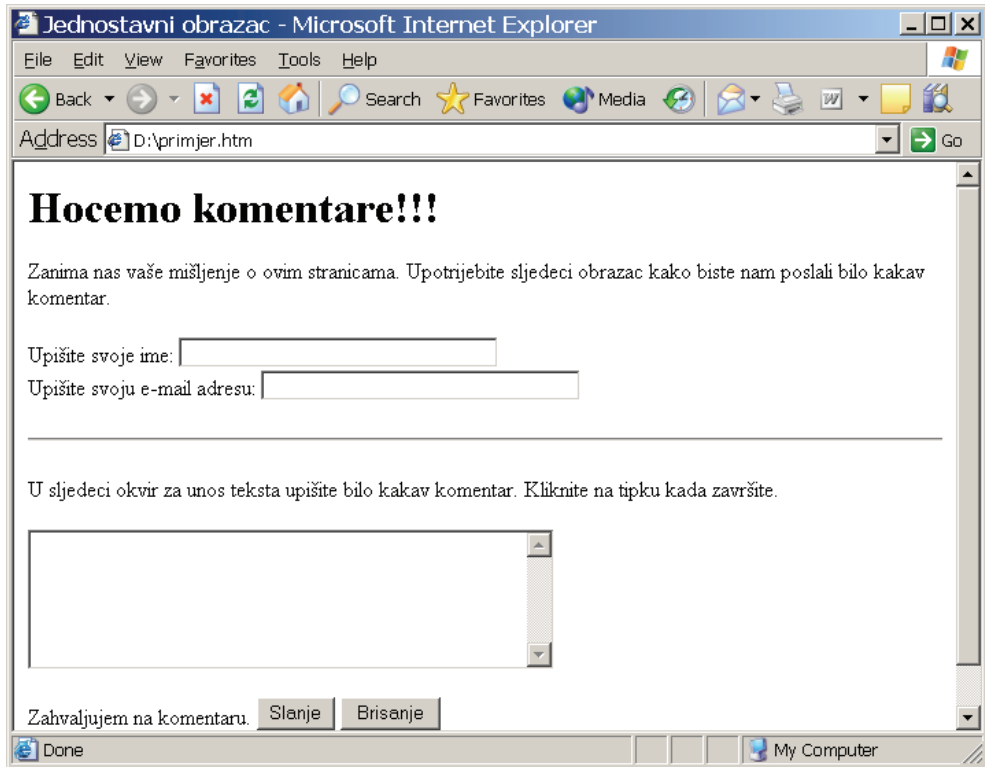
<INPUT TYPE="SUBMIT" VALUE="Slanje">
<INPUT TYPE="RESET" VALUE="Brisanje">

<P>
</FORM>
</BODY>
</HTML>
```

Slika 16 prikazuje ovaj primjer nakon učitavanja u pretraživački program. Uočite odmah da se drugi HTML elementi mogu slobodno koristiti kako izvan, tako i unutar obrasca.

Naredba <FROM>, kao što vidite, može sadržavati i argument METHOD koji je namijenjen određivanju načina na koji će se sadržaj obrasca poslati na e-mail adresu ili CGI skriptu. Najbolje je uvijek dodjeljivati vrijednost POST ovom argumentu, kako bi se naznačilo da se sadržaj šalje preko standardnog ulaza. Druga vrijednost koju možete navesti je GET, a ona se koristi kada se sadržaj obrasca šalje serveru kao dio URL adrese. Neki stariji serveri imaju ograničenje na količinu podataka koja se ovako može poslati, pa je najbolje da koristite POST metodu.

Pomoću argumenta ACTION, u ovom smo primjeru naveli da želimo podatke iz obrasca poslati na e-mail adresu v.predstavnik@gov.ba.



Slika 16. Jednostavan obrazac za komentar

Prva dva polja u ovom primjeru zatražit će od korisnika da unese svoje ime i e-mail adresu, a kreirali smo ih pomoću naredbe `<INPUT>`. Argumentom TYPE koji se dopisuje uz ovu naredbu određuje se tip polja koje želimo ugraditi u obrazac, pa smo mi upotrijebili vrijednost TEXT kako bi dobili polje za unos teksta. Tu je još i argument SIZE kojim smo odredili širinu tekstualnog polja koja se mjeri u broju znakova, ali i argument NAME koji je obavezan za svako polje obrasca, a kojemu se dodjeljuje neko jedinstveno ime. To ime će nam poslužiti kao oznaka varijable u kojoj će biti pohranjeno ono što je korisnik upisao unutar odgovarajućeg polja. Kasnije ćemo tu varijablu upotrijebiti u skripte koja obrađuje obrazac.

Naredbom `<TEXTAREA>` kreirali smo okvir za unos teksta kojemu smo veličinu u recima i stupcima odredili redom pomoću argumenata ROWS i COLS. Za razliku od naredbe `<INPUT>`, ova naredba sadrži i završnu oznaku `</TEXTAREA>`, a sav tekst koji upišemo između početne i završne oznake bit će već upisan unutar okvira za unos teksta, što je korisno ako korisniku želimo unaprijed kazati što želimo čuti. U našem primjeru smo okvir za tekst ostavili praznim.

Na kraju obrasca nalaze se dvije tipke koja smo kreirali pomoću naredbi `<INPUT TYPE="SUBMIT">` i `<INPUT TYPE="RESET">`. Kada korisnik klikne na prvu tipku, pretraživački program će započeti slati sadržaj obrasca na određeno naznačeno argumentom `ACTION`, i to metodom navedenom pomoću argumenta `METHOD` u naredbi `<FROM>`. Ukoliko korisnik klikne na drugu tipku, sadržaj svih polja u obrascu će se izbrisati i neće uopće doći do slanja podataka. Pomoću argumenta `VALUE`, u oba slučaja navodimo tekst koji želimo da se ispiše na tipki. Ovo su ujedno i jedina dva tipa polja uz koje se nikada ne navodi argument `NAME`.

Što će biti poslano na e-mail adresu ili CGI skriptu? Odgovor je jednostavan. Bit će poslan niz znakova unesenih u različita polja u obrascu i to u obliku `ime-varijable=vrijednost`. Svaki ovakav član bit će razdvojen znakom `&`. Ime varijable dobiva se od vrijednosti koja je dodijeljena argumentu `TYPE` kod svakog pojedinog polja, a njoj se dodjeljuje ono što je korisnik upisao. Da je, naprimjer, neko ko se zove Edo upisao svoje ime i svoju e-mail adresu u obrazac sa slike 16, te da je u okvir za unos teksta upisao "provjera...", sadržaj e-mail poruke poslane na adresu `v.predstavnik@gov.ba` bi bio ovakav:

`Ime=Edo+Alić&e-mail=edoalic@student.unze.ba&komentar=provjera...`

Vjerojatno ste odmah primijetili da se uneseni razmaci zamjenjuju znakom plus (+). Ukoliko korisnik unese neke druge posebne znakove u obrazac, oni će biti interpretirani na poseban način, sa znakom `%` iza kojega se navodi heksadecimalni ASCII kod znaka.

4.12. Client-Pull – klijentsko povlačenje?

Pretraživački programi su napravljeni tako da njima u potpunosti upravljaju čitatelji upisujući URL adrese i klikajući po hipertekstualnim vezama. Tvrtka Netscape je pojavom Navigatora 1.1 malo promijenila tu situaciju, uzevši u HTML koncept dinamičkih dokumenata koji se izrađuju pomoću tehnologija nazvanih Client-Pull i Server-Push. Client-Pull se svodi na to da u slučaju kada korisnik ne klikne na niti jednu hipertekstualnu vezu unutar neke stranice ili na neki drugi način iz nje ne izađe, Web server automatski šalje klijentu novi dokument.

Kod tog oblika dinamičkog dokumenta koristi se naredba `<META>`, smještena unutar zaglavlja dokumenta, koja pretraživačkom programu govori kada da "povuče" novi dokument, ili kada da jednostavno ponovno učita isti dokument, kao u sljedećem primjeru:

`<HTML>`

`<HEAD>`

`<META HTTP-WQUIV="Refresh" CONTENT=5>`

`<TITLE>Prva stranica</TITLE>`

```
</HEAD>

<BODY>

    <H1>Ovo je PRVA STRANICA!</H1>

    Ovdje nabacate malo običnih HTML naredbi...

</BODY>

</HTML>
```

Ako ovaj primjer učitate u neki pretraživački program koji podržava Client-Pull tehnologiju (neki stariji pretraživački i Lynx je ne podržavaju), primijetit ćete da se on svakih pet sekundi ponovno učitava. Naredba `<META>` svakih će pet minuta simulirati pritiskanje tipke Reload (Refresh) u pretraživaču, te će ponovno učitati isti dokument nakon onoliko sekundi koliko ste naveli argumentom `CONTENT`. Ovo je korisno u slučaju da se sadržaj HTML dokumenta automatski kreira i mijenja u određenim vremenskim razmacima. Istom naredbom moći ćete "povući" i neki drugi HTML dokument, kao u novom primjeru:

```
<META HTTP-EQUIV="Refresh"

CONTENT="60; URL=http://unze.ba/stats/">
```

Uočite da je obavezno navođenje apsolutne URL adrese i to odmah nakon tačke-zareza koju valja navesti iza broja za oznaku sekundi. Sljedeća stranica u nizu također može biti Client-Pull stranica, što znači da možete kreirati čitave nizove dinamičkih dokumenata koji će se automatski učitavati. Ova tehnologija korisna je i u slučaju da ste svoje Web stranice preselili na drugi server, a želite da se čitatelji koji još uvijek koriste staru adresu automatski upute na novu. Tada ćete, naravno, navesti nulu kao broj sekundi nakon kojih će pretraživač skočiti do novog odredišta.

Kako postoje pretraživački programi koji ne podržavaju naredbu `<META>`, dobra ideja je da negdje pri vrhu stranice objasnite da će dokument nakon toliko i toliko sekundi nestati (odnosno da neće nestati), i na taj način navedete korisnika da klikne na neku hipertekstualnu vezu.

Kada želite da korisnik odabere neku vrijednost iz nekog konačnog popisa koji mu je ponuđen, upotrijebit ćete kružne izbornike, kontrolne kvadratiće ili padajuće izbornike. Kružni izbornici rade na taj način, da, kada se jedna opcija odabere, nijedna druga više ne može biti istovremeno odabrana. Zbog toga kružne izbornike treba koristiti onda i samo onda kada želite da korisnik odabere tačno jednu vrijednost iz izbora koji mu se nudi. Svaki kružić unutar izbornika predstavljen je jednom `<INPUT TYPE="RADIO">` naredbom no, međutim, argumentu `NAME` se unutar jednog izbornika uvijek mora dodijeliti ista vrijednost. Pomoću argumenta `VALUE` određuje se vrijednost koja će biti dodijeljena varijabli navedenoj argumentom `NAME` kada korisnik izabere jednu od ponuđenih mogućnosti. Kod onog kružića koji želite odmah prikazati kao odabranu opciju, nadopisat ćete i argument `CHECKED`. Evo primjera:

```
<HTML>

<HEAD>

    <TITLE>Kružni izbornik</TITLE>

</HEAD>

<BODY>

<FORM METHOD=POST ACTION="cgi-bin/sendmail">

<H1>Najviše volim:</H1>

<INPUT TYPE="RADIO" NAME="voli"
VALUE="pjevače">Pjevače<BR>

<INPUT TYPE="RADIO" NAME="voli" VALUE="pjevačice"
CHECKED>Pjevačice<BR>

<INPUT TYPE="RADIO" NAME="voli" VALUE="grupe">Grupe
izvođača<BR>

</FORM>

</BODY>

</HTML>
```

Kontrolni kvadratići se koriste kada se korisniku želi omogućiti izbor od nijedne, jedne ili više ponuđenih opcija. Svaki kvadratić prikazuje se pomoću jedne `<INPUT TYPE="CHECKBOX">` naredbe, a kod svake naredbe argument `NAME` poprima različitu vrijednost. Argumentom `VALUE` iskazuje se vrijednost koja će biti dodijeljena varijabli navedenoj pomoću argumenta `NAME` u slučaju da je korisnik odabrao tu opciju. Unaprijed odabrani kvadratići označavaju se argumentom `CHECKED`. Na primjer:

```
<HTML>

<HEAD>

    <TITLE>Kontrolni kvadratići</TITLE>

</HEAD>

<BODY>

<FORM METHOD=POST ACTION="cgi-bin/sendmail">

<H1>Muzika koju slušam:</H1>

<INPUT TYPE="CHECKBOX" NAME="pop" VALUE="da">Pop<BR>

<INPUT TYPE="CHECKBOX" NAME="jazz" VALUE="da"
CHECKED>Jazz<BR>

<INPUT TYPE="CHECKBOX" NAME="funk" VALUE="da"
CHECKED>Funk<BR>

<INPUT TYPE="CHECKBOX" NAME="rap" VALUE="da">Rap<BR>
```

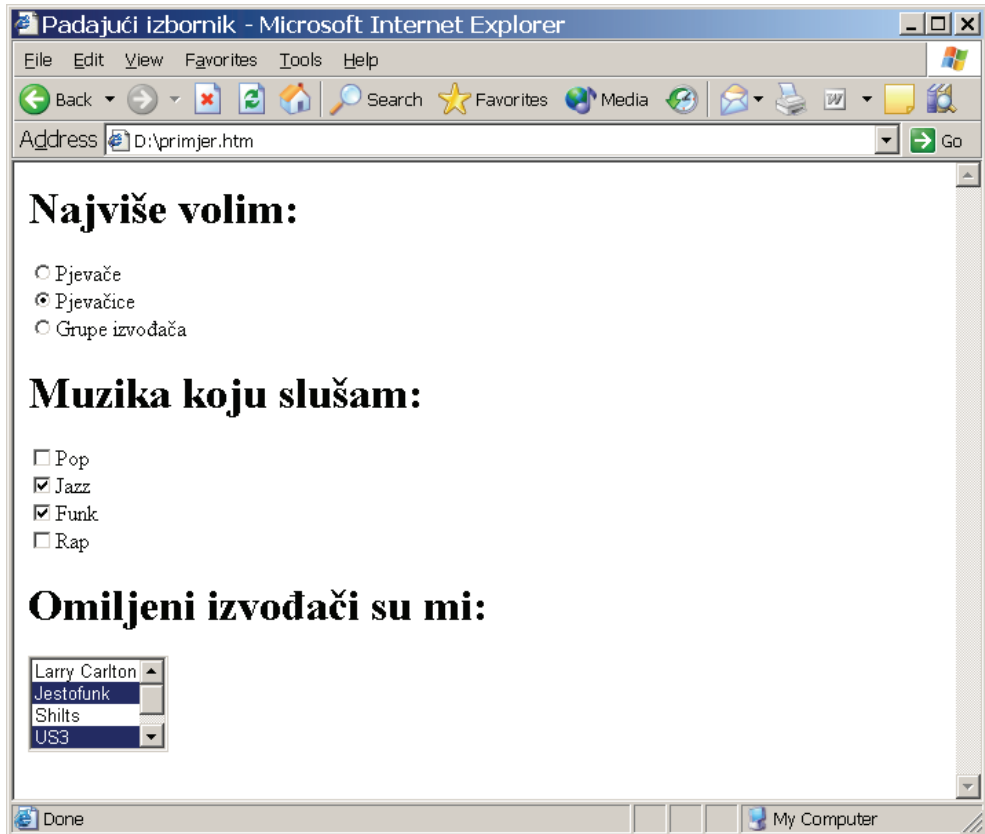
```
</FORM>
</BODY>
</HTML>
```

Padajući izbornici kreiraju se pomoću naredbe `<SELECT>`, a također su korisni kada se korisniku želi omogućiti odabir jedne opcije od više ponuđenih, kao što je slučaj i s kružnim izbornicima. Naredba `<SELECT>` sadrži i završnu oznaku `</SELECT>`, tako da se između početne i završne oznake mogu nalaziti samo naredbe `<OPTION>` kojima se nabrajaju mogućnosti izbora. Argument `NAME` ovaj put se dopisuje samo kod naredbe `<SELECT>`, dok se `VALUE` navodi kod svake pojedine naredbe `<OPTION>`. Pomoću padajućih izbornika korisniku se može omogućiti i izbor više opcija, kao kod kontrolnih kvadratića, no tada se uz naredbu `<SELECT>` mora navesti i argument `MULTIPLE`. One opcije za koje želimo da unaprijed budu izabrane, označit ćemo argumentom `SELECTED`. Naprimjer:

```
<HTML>
<HEAD>
    <TITLE>Padajući izbornik</TITLE>
</HEAD>
<BODY>
    <FORM METHOD=POST ACTION="/cgi-bin/sendmail">
        <H1>Omiljeni izvođači su mi:</H1>
        <SELECT NAME="sluša" MULTIPLE>
            <OPTION VALUE="Larry">Larry Carlton
            <OPTION VALUE="Jesto" SELECTED>Jestofunk
            <OPTION VALUE="Shilts">Shilts
            <OPTION VALUE="US3" SELECTED>US3
            <OPTION VALUE="Crusaders" SELECTED>Crusaders
        </SELECT>
    </FORM>
</BODY>
</HTML>
```

Na slici 17 možete pogledati kako zadnja tri primjerka izgledaju kada se učitaju u pretraživač. U tim smo primjerima kod argumenta `ACTION` u naredbi `<FORM>` koristili relativnu URL adresu CGI skripta koji je pretpostaviti ćemo, instaliran u `cgi-bin` direktoriju na našem Web serveru. Ovdje se slobodno mogu navoditi i apsolutne URL adrese raznih CGI skripte, što znači da nije obavezno koristiti

serverske programe za obradu obrazaca koji su instalirani na onom Web serveru na kojem se nalaze vaše stranice.



Slika 17. Kružni izbornik, kontrolni kvadratići i padajući izbornik

4.13. Skripting jezici

4.13.1. Skripting jezici koji se izvršavaju na serveru (Server-side scripting)

ASP - Active Server Pages

Active Server Pages, odnosno kraće ASP, je Microsoftovo rješenje za serverside scripting. Kod običnih HTML stranica, klijent (odnosno web surfer) zatraži web stranicu sa nekog servera (npr. www.unze.ba), a server jednostavno pošalje dokument klijentu (obično `index.html`, ako nije navedeno ime) i surfer vidi tu stranicu prikazanu u svom web browseru. Sa ASP-om, server ima priliku da izmijeni dokument prije nego ga pošalje korisniku, odnosno klijentu. Dakle, kada surfer zatraži neku stranicu sa nastavkom `.asp`, server je prvo obradi, izvrši komande koje su zadane u ASP dokumentu, i zatim šalje dokument klijentu. Na ovaj način postiže se interaktivnost, odnosno dinamičnost web stranica. Pomoću

ASP-a možete samo začiniti svoje stranice dodajući sitne "cake" poput datuma, ili brojača posjeta, ali možete napraviti i ogromne web-aplikacije za pristup bazama podataka, web-trgovine i slično. Da biste uspješno koristili ASP, morate koristiti Microsoftov web server, ili tačnije - Microsoft Internet Information Server (IIS) 3.0 ili veći. To je ujedno i glavna mana ove tehnologije - ovisna je o Windows platformi, a na polju servera Unix i Linux zauzimaju značajan procenat. Postoje i programi koji omogućavaju pokretanje ASP-a na Unix serverima.

Svakako dobra strana ASP-a je što on nije scripting jezik, dakle ne morate učiti njegovu sintaksu i komande da biste ga koristili. Microsoft ga naziva server-side scripting environment, tj. okruženjem za server-side scripting. On u stvari predstavlja kombinaciju HTML-a i nekog od skripting jezika - JavaScripta i VBScripta. Pošto se stranice obrađuju na serveru, ne morate voditi računa o kompatibilnosti koda koji ste napisali sa oba browsera, što je svakako još jedan plus za ovu tehnologiju.

CGI - Perl

Obični HTML je sasvim dobar kada je riječ o prikazivanju web stranica, grafike, postavljanja linkova, teksta i ostalih, "uobičajenih" radnji. Međutim, šta ako želite uraditi neke stvari koje su izvan njegove mogućnosti, kao npr. prikazati brojač posjeta, napraviti sobu za chatanje, prepoznati koji web browser koristi surfer i slično? Ove stvari zahtijevaju slanje podataka od surferovog web browsera ka serveru, koji zatim te podatke šalje programima koji ih obrađuju i ponovno prosljeđuju serveru, nakon čega on šalje HTML dokument korisniku.

CGI, što je skraćenica od Common Gateway Interface, nije programski jezik nego protokol - skupina pravila po kojima server postupa pri "razgovoru" sa programom. Dakle, kada korisnik popuni formular na stranici i pritisne Enter, HTML kontroliše slanje podataka web serveru. On zatim prosljeđuje podatke Perl skripti (odnosno programu) koja procesira podatke. CGI određuje kako se ti procesirani podaci vraćaju nazad do web servera i zatim ponovo do korisnika.

Perl je programski jezik koji se može izvršavati na i na Windows i na Unix serverima. Jedini uslov za funkcionisanje Perl skripti na serveru jeste postojanje Perl kompajlera. Dosta je raširen na Internetu i možete naći ogromnu količinu gotovih Perl skripti koje možete koristiti bez ikakvog plaćanja autoru. Izvršava se brže od ASP-a i PHP-a, tako da je dobar izbor za procesiranje veće količine podataka i pristup velikim bazama podataka. Velika mana ovog programskog jezika jeste njegova kompleksnost i težina učenja.

Ova "jednostavna" Perl skripta će prikazati vrijeme na web stranici:

```
#usr/local/bin/perl
print "Content-type:text/plain","\n\n",
$time="Trenutno je;",$Time;
exit(0);
```

A da sada upotrijebimo ASP za isti zadatak:

```
<%=time%>
```

PHP - Personal Home Pages

PHP, što dolazi od pomalo čudne skraćenice Personal Home Pages, je jezik za server-side scripting, nešto poput Active Server Pages-a. Za razliku od ASP-a, PHP se može pokretati na obje platforme koje danas susrećemo na web serverima - i na Windows i na Unixu. To mu je ujedno i velika prednost, jer nije ovisan o platformi na kojoj se pokreće, što predstavlja preduslov za popularnost nekog jezika. Poput ASP-a, PHP komande su inkorporirane u HTML dokumente, samo što oni tada imaju nastavke .php, .php3 ili .phtml. Kada korisnik zatraži dokument sa tim nastavkom od web servera, server prvo izvršava komande koje se nalaze u njemu i nakon toga šalje dokument korisniku tj. web surferu. Ovaj jezik se u kombinaciji sa MySQL-om (koji također postoji u verzijama za WinNT i Unix) smatra najjačom i najboljom programskom platformom za pristup bazama podataka na webu. PHP je besplatan i distribuira se pod Open-source licencom, što mu je još jedna prednost.

4.13.2. Skripting jezici koji se izvršavaju kod klijenta (Client-side scripting)

JavaScript

Osnovna namjena JavaScripta jeste da omogućí autorima web stranica jednostavan način za postizanje raznih smicalica i zanimljivih efekata na stranicama. Kao što mu samo ime govori, JavaScript je baziran na Javi, multiplatformskom jeziku oko kojeg se svojevremeno diglo (neopravdano) mnogo prašine. Ima mnogo manje mogućnosti i smanjenu fleksibilnost nego Java, ali za ono što je namijenjen. JavaScript služi sasvim dobro.

VBScript

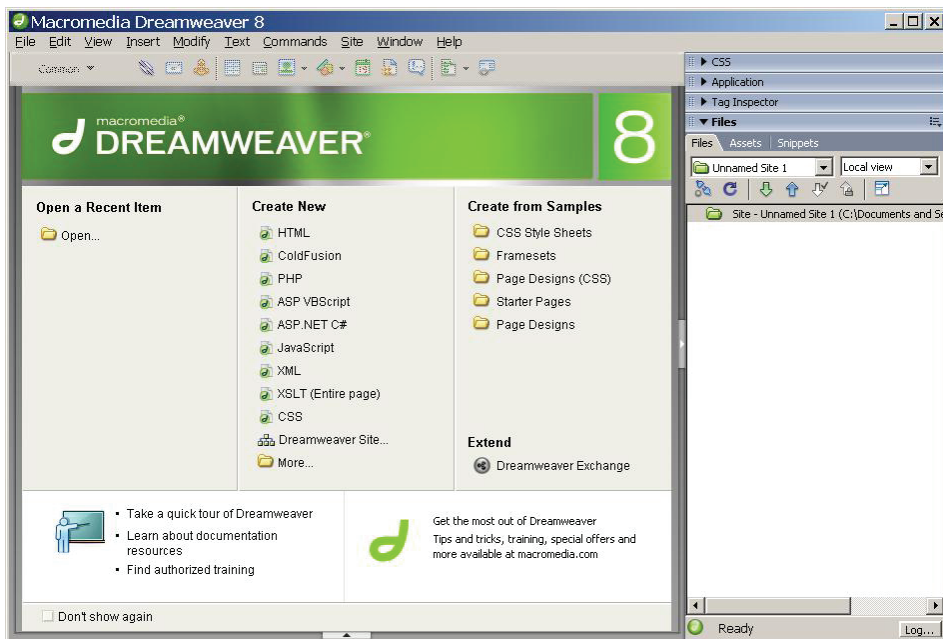
VBScript predstavlja Microsoftovu alternativu, odnosno rješenje za client-side skriptiranje. Vjerovatno pogađate kako je VBScript potpuno nečitljiv u Netscape Navigatoru tj. kako je ova tehnologija rezervisana samo za Internet Explorer. To je bila i osnovna prepreka njenom širenju, tako da ćete danas rijetko naći neku stranicu koja sadrži VBScript u sebi. Međutim, ova tehnologija nije izumrla. Našla je odličnu primjenu kao default jezik u ASP-u. Upravo zato prelazi na ASP developerima koji imaju iskustva u VB-u uopšte nije težak. Visual Basic Scripting je podskup Visual Basic for Applications, koji je proizvod kompanije Microsoft .

5. Macromedia Dreamweaver

5.1. Uvod u Macromedia Dreamweaver

Iako se web stranice mogu kreirati 100% pomoću običnog tekst-editora, kao što je Notepad, postoje programski paketi koji taj posao uveliko olakšavaju. Najpoznatiji takav program je Macromedia Dreamweaver.

Prije nego što počnete da pravite Web strane pomoću Macromedia Dreamweaver-a, treba da upoznate razne Dreamweaverove alate i panoe, koji će vam omogućiti da efikasno pravite Web prezentacije. Ako prvi put pokrećete Dreamweaver, prikazat će se prozor dobrodošlice s informacijama za nove korisnike, kao i sažetak novih mogućnosti za one koji već poznaju program. Dugmad na paleti u dnu prozora služe za kretanje kroz informacije o programu. Po otvaranju Dreamweavera vidi se i prozor za nov dokument, u kome se obavlja projektovanje i programiranje. Možda se vide i panoi za dodavanje i mijenjanje teksta i objekata, za rad s datotekama prezentacije i drugo. Među panoima koji se vide mogu biti traka Insert, pano za prikazivanje svojstava (Properties), prozor prezentacije (Site) i pano za odgovore (Answers).



Slika 18. Pozdravni prozor Dreamweaver-a

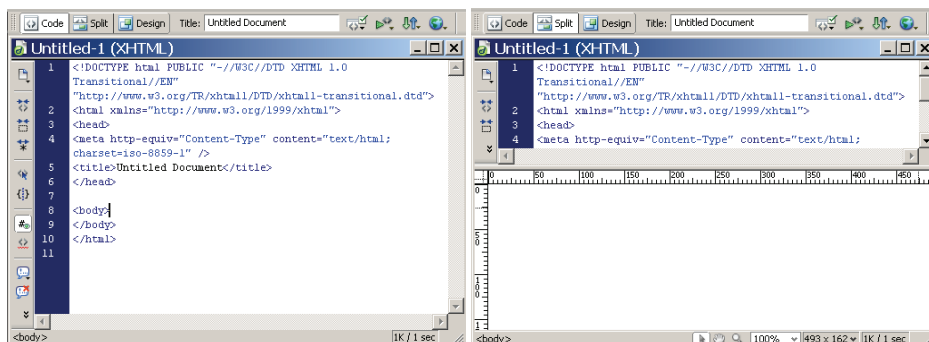
Većina Dreamweaverovih panoa svrstana je prema svojim funkcijama u slijedeće grupe: Design, Code, Application, Files i Answers. Grupe panoa omogućavaju da najčešće korištenim panoima brzo pristupate i da ih jednako brzo sakrivate. Svim panoima (u pomenutim grupama i izvan njih) možete da pristupite iz menija Window.

Ako ste već otvarali Dreamweaver, panoi će biti smješteni tamo gdje ste ih ostavili kada ste posljednji put zatvorili program. Znak za potvrdu u meniju Window označava da su pano ili paleta alatki otvoreni i vidljivi. Može se desiti da pano bude sakriven ispod drugog panoa ili prozora dokumenta. Da biste prikazali sakriveni pano, ponovo ga izaberite u meniju Window. Ako se potrebni pano ne nalazi u trenutno vidljivoj grupi panoa, pojavit će se i pano i njegova grupa.

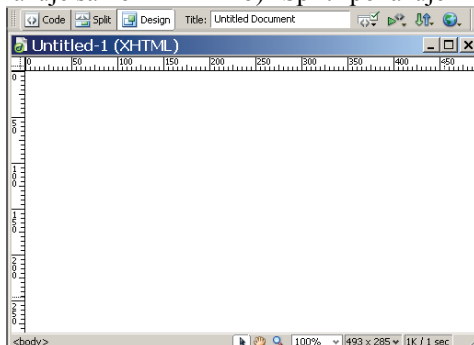
Ako je pano izabran, a ipak se ne vidi, izaberite Window, Arrange Panels da bi se svi paneli vratili na svoja podrazumijevana mjesta. Traka Insert će otići na svoje mjesto u gornjem lijevom uglu, pano svojstava (Properties) preći će na dno prozora, a svi ostali otvoreni panoi na desnu stranu prozora.

Najveći dio posla obavljate u prozoru dokumenta, gdje možete da umetnete, promijenite i izbrisete mnoštvo elemenata od kojih se sastoji Web strana. Dok radite, u prozoru dokumenta prikazuje se približan izgled strane u browseru: Nova strana se podrazumijevano zove "Untitled Document", što se vidi na naslovnoj traci dokumenta. Poslije naslova dokumenta, koji je prikazan i u polju Title na paleti alatki dokumenta, u zagradama je navedeno ime datoteke.

U Dreamweaveru postoje tri načina prikazivanja: Design View, Code View i Split Design and Code. Dugmad za uključivanje ovih prikaza nalaze se na lijevoj strani palete alatki dokumenta. Dugme aktivnog režima prikazivanja je uvijek istaknuto.



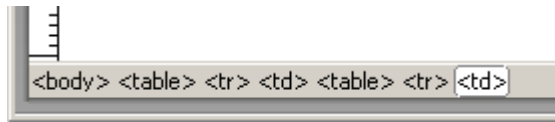
a) "Code" prikazuje samo HTML b) "Split" pokazuje i HTML i izgled



c) "Design" prikazuje stranicu kako će izgledati u pretraživaču

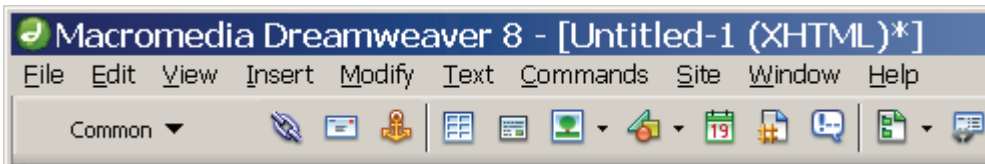
Slika 19. Načini prikazivanja stranice

U donjem lijevom uglu prozora dokumenta nalazi se birač HTML oznaka (Tag Selector). U njemu se hijerarhijski prikazuju HTML oznake koje pripadaju trenutno izabranom elementu. (Početna HTML oznaka je uvijek <body>).



Slika20. Birač HTML oznaka

Birač HTML oznaka može da koriste za kretanje kroz elemente strane pomoću HTML oznaka koje odgovaraju tim elementima. Ova alatka omogućava brzo kretanje kroz hijerarhiju koda strane, pregled elemenata s kojima radite i lako biranje ostalih elemenata. To je posebno zgodno kod rada sa višestrukim umetnutim tabelama; tada se u biraču HTML oznaka pojavljuju oznake tabela <Table>, redova <TR> i ćelija <TD>.

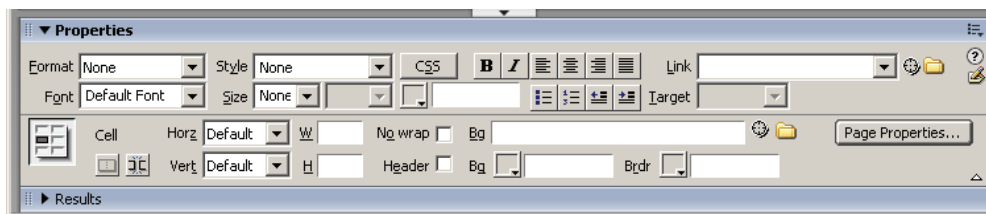


Slika 21. Osnovni meni i traka Insert

Traka Insert sadrži više objekata i elemenata koje možete da postavite na stranu: slike, tabele, specijalne znakove, obrasce i okvire. Elementi koje možete da umetnete razvrstani su po grupama u zavisnosti od vrste. Grupe su Common (uobičajeni elementi), Frames (okviri), Forms (obrasci), Layout (raspored elemenata), Characters (znakovi) i druge. Grupe birate pritiskanjem odgovarajućeg jezička kartice. Primjera radi, u grupi Characters nalaze se brojni specijalni znakovi, dok se u grupi Common nalaze uobičajeni elementi, kao što su slike i tabele. Svim ovim elementima i objektima možete da pristupate i kroz meni Insert.

Da biste umetnuli element, prevucite njegovu ikonicu sa trake Insert na odgovarajuće mjesto u prozoru dokumenta. Drugi način je da najprije u dokumentu zadate (postavite) tačku umetanja, a potom da pritisnete ikonicu objekta na panou. Kada pritisnete ikonicu, element će se pojaviti u dokumentu na mjestu tačke umetanja.

U zavisnosti od objekta koji izaberete, može da se pojavi i okvir za dijalog s opcijama za svojstva i smještaj elementa. Za neke objekte, kao što su tabele i slike, možete da zaobiđete okvir za dijalog i da odmah u dokument umetnete oznaku za rezervisano mjesto (sa tipkom Alt). Oznake za rezervisano mjesto koriste se kada pravite strane za koje fotografije i slike još nisu spremne.



Slika 22. Pano Properties služi za podešavanje osobina objekata

Pano Properties se koristi za pregled i mijenjanje atributa (svojstava) izabranog teksta, slika, tabela i ostalih elemenata strane. Njegov sadržaj se mijenja u zavisnosti od konteksta: atributi koje prikazuje zavise od toga šta je izabrano u prozoru dokumenta. Da biste smanjili ili proširili pano Properties, pritisnite dugme za proširenje u donjem desnom uglu panoa. Ako je pano smanjen, može biti dodatnih svojstava za čije prikazivanje nema mjesta i koja će ostati nevidljiva dok ne proširite pano.

Većina panoa koje ćete koristiti tokom pravljenja Web strana jesu usidreni – složeni u obliku kartica s jezičkom – unutar grupa panoa. Sidrenje povećava slobodnu površinu prozora, a istovremeno omogućava brz pristup potrebnim panoima. Svaka grupa panoa može biti proširena tako da se prikažu svi njeni panoi, odnosno sažeta tako da se vidi samo ime grupe.

Ponekad nećete moći da pronađete određeni pano (jedan od razloga za to može biti i promjena rezolucije). Ako vam neki pano treba, a ne možete da ga prikazete ni kada ga izaberete u meniju Window, izaberite opciju menija Window, Arrange Panels, pa će se panoi priljepiti uz ivice ekrana.

5.2. Razvijanje strukture prezentacije

Posvetite se pravljenju detaljne skice ili dijagrama toka prezentacije, jer ćete tako lakše razviti svoje zamisli, bolje shvatiti opseg projekta i uštedjeti vrijeme i resurse. Dobra Web prezentacija je intuitivna i ostavlja pozitivan i jedinstven utisak na posjetioca. Izrada dobre Web prezentacije počinje definisanjem i sažetim prikazivanjem razloga i potrebe za takvom Web lokacijom. Zapitajte se ili pitajte naručioca prezentacije: zašto je ona potrebna i čemu će služiti? Šta hoćete da izrazite pomoću ove prezentacije i na koji način ćete to postići? Kakav utisak hoćete da ostavite i šta posjetioci ove Web lokacije treba da zapamte?

Poznavanje posjetilaca je suštinski značajno. Ko će posjećivati Web lokaciju? Definisanjem opšteg profila posjetilaca lakše ćete dostići ciljnu grupu posjetilaca. Kada shvatite ko sačinjava vašu buduću publiku, zapitajte se kakvom tehnologijom oni raspolažu. Kakve softverske dodatke, browsere i operativne sisteme koristi većina budućih posjetilaca? Sve to morate uzeti u obzir da bi im Web lokacija bila dostupna. Primjera radi, bilo bi pogrešno napraviti prezentaciju s elementima koje

podržavaju samo najnoviji i najsavremeniji browseri, ako vaša publika uglavnom koristi starije računare na kojima se takvi browseri uopšte ne mogu pokrenuti.

Web lokacije u velikoj mjeri zavise od strukture i korišćenja datoteka; loše struktuirana Web lokacija zbunjuje posjetioca dok se kreće po njoj, i teško se koristi i održava. Da biste napravili jasnu lokaciju, koja je komunikativna i lako se koristi, morate do kraja da isplanirate strukturu i hijerarhiju datoteka i direktorija unutar Web lokacije prije nego što počnete da pravite HTML dokumente.

Važan dio priprema za projektovanje i izradu Web prezentacije jeste definisanje i prikupljanje njenih sastavnih dijelova: teksta, slika i multimedijских elemenata.

5.3. Definisanje lokalne web prezentacije

Prvi korak u pravljenju Web prezentacije – koji treba obaviti prije nego što počnete izradu Web stranica – jeste biranje postojećeg ili pravljenje novog direktorija koji će sadržavati cjelokupnu buduću Web prezentaciju. Ovaj postupak se naziva "definisanje lokalne "Web prezentacije". Izabrani direktorij, lokalni korijenski direktorij, definiše granice lokalne Web prezentacije na vašem lokalnom disku i odražava stanje udaljene Web lokacije, tj. stvarne lokacije na Web serveru kojoj će posjetioci pristupati. Pri definisanju lokalne Web prezentacije možete da zadate istu hijerarhiju direktorija u lokalnoj i udaljenoj verziji, što je nezaobilazno prilikom izrade i održavanja aktivne lokacije.

Pravljenjem lokalne Web prezentacije, koja se sastoji od lokalnog korijenskog direktorija unutar kojega ćete izgraditi strukturu datoteka i direktorija buduće lokacije, spriječit ćete tu lokaciju da pristupa drugim datotekama izvan lokalnog korijenskog direktorija. Datoteke na lokalnom disku koje se nalaze izvan lokalnog korijenskog direktorija neće biti dostupne udaljenom serveru.

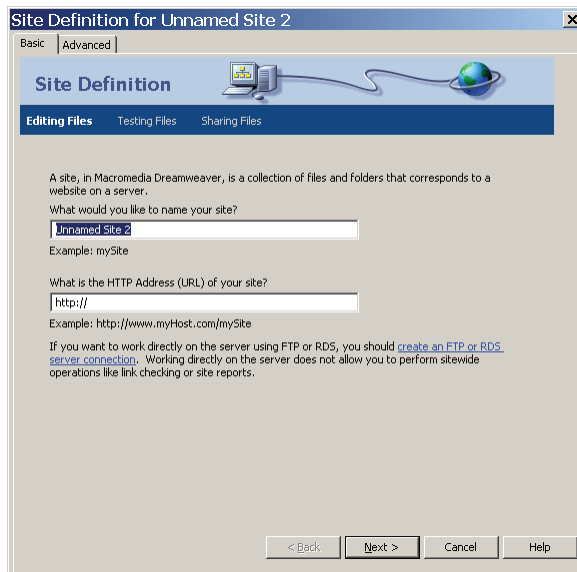
Da bi mnoge Dreamweaverove funkcije, kao što je mogućnost ažuriranja svih referenci datoteke koja je premještena u drugi direktorij lokacije, mogle da rade u potpunosti, lokalna lokacija mora biti definisana. Mjesto čuvanja i hiperveze do svih datoteka koje lokacija sadrži bit će relativne, zadate u odnosu na korijenski direktorij lokacije. Uvijek bi trebalo da pravite lokalne Web prezentacije i da radite unutar njih. Ako to ne budete činili, imat ćete problema s hipervezama, putanjama i korištenjem datoteka.

Prezentaciju ćete razvijati unutar lokalne prezentacije na lokalnom disku, gdje ćete napraviti i testirati Web strane. Kada otpočnete izradu novih Web prezentacija, morat ćete za njih da napravite ili izaberete osnovne direktorije.

Ime osnovnog direktorija može da bude jednako imenu odgovarajuće prezentacije, a i ne mora. Ako pravite mnogo prezentacija, koristite imena koja se lako razlikuju. Ime osnovnog direktorija služi samo vama, za rad s datotekama, pošto ga posjetioci buduće Web lokacije neće vidjeti.

U Dreamweaveru izaberite Site, New Site. Pređite na kraticu Basic u okviru za dijalog Site Definition.

Otvorit će se okvir za dijalog Site Definition s dvije kartice, Basic i Advanced, koje služe za biranje postupka definisanja prezentacije. Kartica Basic, koja se podrazumijevano prikazuje kada otvorite okvir za dijalog, vodi vas kroz postupak korak po korak. Kartica Advanced daje još nekoliko dodatnih opcija i parametara za podešavanje, ali ne sadrži tekstualna objašnjenja koja se prikazuju u režimu Basic.



Slika 23. Definisanje prezentacije (Basic)

Ako dosad niste napravili prezentaciju u Dreamweaveru, prozor Site (ako je otvoren) prikazuje direktorij u kome nema nikakve prezentacije.

Dreamweaver postavlja pitanje: "What would you like to name your site?" (Kako ćete nazvati prezentaciju?) Izaberite i upišite neko ime u polje za ime i pritisnite Next.

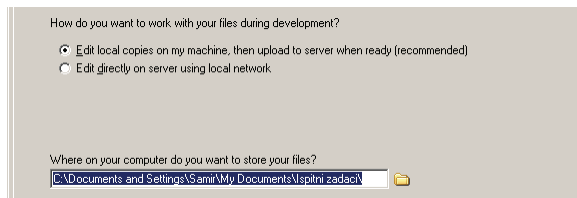
"Ispitni zadaci" je ime projekta prezentacije koju pravite u ovoj lekciji. Kada sami budete pravili prezentacije, imenovat ćete ih proizvoljno. Jasna i specifična imena omogućavaju da razlikujete prezentacije već po imenu, što olakšava rad s više prezentacija. Ovo ime prezentacije služi samo vama, jer ga posjetioci lokacije neće vidjeti.

Ime prezentacije se na kartici Advanced upisuje u polje Site Name u kategoriji Local Info.

U drugom koraku Dreamweaver pita: "Do you want to work with a server technology such as ColdFusion, ASP.NET, ASP, JSP or PHP?" (Želite li da radite sa serverskim tehnologijama Cold Fusion, ASP.NET, ASP, JSP ili PHP?) Pritisnite

radio-dugme "No, I do not want to use a server technology". (Neću.) Pritisnite Next kako biste prešli u sljedeći odjeljak.

Za izradi statičkih prezentacija izaberite opciju No. Definiciju prezentacije uvijek možete da izmijenite kad izaberete opciju menija Site, Edit Sites... Ovaj dio kartice Basic odgovara kategoriji Testing Server na kartici Advanced, gdje imate i dodatne mogućnosti za pravljenje dinamičkih prezentacija, kao što je biranje modela servera koji će se koristiti na udaljenom serveru.



Slika 24. Izbor načina rada sa datotekama

Na vrhu ovog područja Dreamweaver pita: "How do you want to work with your files during development?" (Izaberite način rada s datotekama tokom razvoja ove prezentacije.) Pritisnite radio dugme, "Edit local copies on my machine, then upload to server when ready (recommended)".

U istom području, Dreamweaver pita: "Where on your computer do you want to store your files?" (Gdje ćete smjestiti datoteke ove prezentacije?). Treba izabrati neki lokalni direktorij.

Ovdje izabrani direktorij odgovara korijenskom direktoriju udaljene Web lokacije. Dreamweaver će pomoću lokalnog korijenskog direktorija odrediti putanje dokumenata, slika i hiperveza vaše prezentacije.

Kada budete pravili druge prezentacije, a niste sami prethodno napravili potreban direktorij, možete pustiti Dreamweaver da automatski napravi korijenski direktorij na osnovu imena prezentacije koje ste zadali.

Ovo područje kartice Basic odgovara polju Local Root Folder u kategoriji Local Info na kartici Advanced. Napredni (eng. advanced) način definisanja prezentacije ima i opciju Refresh Local File List Automatically, koja je podrazumijevano uključena, a čini da Dreamweaver automatski ažurira spisak korištenih datoteka kad god dodate novu datoteku u direktorij prezentacije. Ako uklonite potvrdu ove opcije, morat ćete ručno da ažurirate lokalne datoteke kad god dodate ili uklonite neku datoteku. Još jedna podrazumijevano uključena opcija na kartici Advanced je Enable Cache, koja dodjeljuje memoriju za često korištene elemente prezentacije. Time se povećava brzina povezivanja i izvršavanja zadataka održavanja Web lokacije. Premda ovu opciju obično treba ostaviti potvrđenu, imajte u vidu da ponovno upisivanje sadržaja u memoriju usporava rad na veoma velikim Web lokacijama.

Pritisnite Next da biste prešli na sljedeći korak. Izaberite "None" u padajućoj listi ispod pitanja "How do you connect to your remote server?" (Kako se povezujete s udaljenim serverom?), ako radite isključivo na lokalnoj prezentaciji i ako nećete morati da pristupate udaljenom Web serveru.

Ovom području kartice Basic odgovara kategorija Remote Info na kartici Advanced, gdje ima dodatnih opcija za definisanje prenosa datoteka na udaljeni server.

Pritisnite Next da biste prešli na sljedeći korak. Pregledajte podatke o prezentaciji koje ste dosad zadali i pritisnite dugme Done na dnu okvira za dijalog.

Kako ste u odjeljcima Application Server i Remote Info izabrali No, Dreamweaver će kao vrijednost obje opcije prikazati tekst "Access; I'll set this up later". Kada pritisnete dugme Done, Dreamweaver će možda prikazati obavještenje da priprema memoriju za ovu prezentaciju. Pritisnite OK i Dreamweaver će pregledati datoteke u direktoriju DWMX-project, kako bi mogao da pripremi memoriju i da prikaže prozor Site kada završi s tim.

Aktivirat će se pano Site (Windows) i prikazati kreirani direktorij, u desnom oknu prozora, u koloni Local Files.

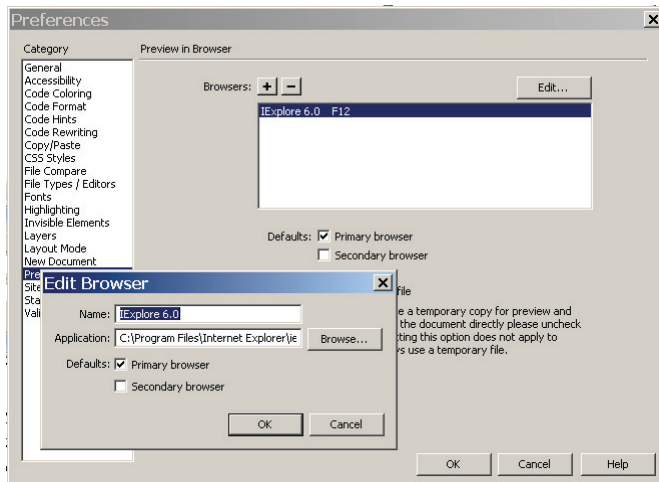
5.4. Zadavanje browsera

Tokom razvoja Web strana morat ćete stalno da provjeravate kako vaš rad izgleda u različitim browserima, kao što su Netscape, Firefox ili Internet Explorer. Ono što vidite u prozoru Dreamweavera samo je približno jednako stvarnom izgledu prezentacije. Svaki browser prikazuje Web strane na različit način, i mada su neke od tih razlika neznatne, druge su veoma značajne. Primijetit ćete razlike čak i između prikaza koje daju različite verzije istog browsera. Što više budete provjeravali izgled prezentacije u raznim browserima i operativnim sistemima i pravili odgovarajuće izmjene, bit ćete sigurniji da će posjetioci Web lokacije njene strane vidjeti kao što ste zamislili. Kartica Preferences u Dreamweaveru služi za zadavanje browsera u kojima ćete provjeravati izgled strana prezentacije. Da biste ubrzali provjeru, zadajte prvi i drugi podrazumijevani browser i za svaki definišite prečicu s tastature.

Izaberite File, Preview in Browser, Edit Browser List.

Otvorit će se okvir za dijalog Preferences i prikazati browsere izabrane za provjeru. Dreamweaver će automatski prikazati spisak browsera instaliranih u računar.

Kada pritisnete ime browsera u spisku, polja za potvrdu ispod spiska (Defaults: Primary Browser, odnosno Secondary Browser) pokazat će da li se radi o primarnom ili sekundarnom browseru. Ako imate više od dva instalirana browsera, ni u jednom od polja za potvrdu neće biti znaka za potvrdu, što znači da tek treba da izaberete koji browseri će biti podrazumijevani.



Slika 25. Izbor web browsera

Ako hoćete da dodate nov browser u spisak, pritisnite dugme plus (+).

Kada se okvir za dijalog otvori, prokrstariće lokalnim diskom, pronađite i izaberite odgovarajuću aplikaciju (browser). Ako hoćete da je, radi provjere izgleda prezentacije, pokrećete pomoću tastera F12, potvrdite polje Primary Browser. Ako hoćete da je pokrećete pomoću kombinacije tastera Ctrl+F12 (Windows), potvrdite polje Secondary Browser. Ako hoćete da uklonite određeni browser iz spiska, izaberite ga i pritisnite dugme minus (-). Ako hoćete da izmijenite primarni ili sekundarni browser za provjeru Web strana, izaberite ga u spisku, pritisnite dugme Edit i pronađite drugi browser na lokalnom disku.

Opcija Preview Using Temporary File podrazumijevano je potvrđena. Ostavite je potvrđenu, kako bi Dreamweaver za provjeru izgleda prezentacije u browseru pravio privremenu datoteku.

5.5. Pravljenje i snimanje nove strane

Iako Dreamweaver otvara nov dokument bez naslova kada pokrenete program, često ćete sami morati da pravite nove strane. Pri tome uvijek treba prvo da snimate dokument.

Ako dokument nema naslov jer ga je Dreamweaver otvorio automatski, možete ga zatvoriti, a da ga ne snimate: izaberite taj dokument i opciju menija File, Close.

Izaberite File, New. Pritisnite karticu General i izaberite opciju Basic Page u spisku. U padajućoj listi Basic Page izaberite opciju HTML i pritisnite dugme Create.

U okviru za dijalog New Document postoje i druge opcije, za pravljenje strana različitih vrsta: HTML, XML i dinamičkih strana koje koriste skripte kao što su

ASP i ColdFusion. U spisku kategorija birate vrstu strane - ponuđeni su šabloni (eng. Templates), kaskadni opisi stilova (eng. CSS Style Sheets), skupovi okvira (eng. Framesets) i gotovi modeli HTML strana (eng. Page designs).

Na dnu okvira za dijalog New Document nalazi se opcija Make Document XHTML Compliant (neka dokument bude kompatibilan sa jezikom XHTML). Jezik za opisivanje Web strana HTML ograničen je, naročito po tome što nove mogućnosti nisu kompatibilne s njegovim starijim verzijama, a postoji i mnogo problema s izvršavanjem na različitim platformama. Jezik XHTML (eXtensible HTML) proširuje mogućnosti HTML-a, pretvarajući ga u jezik tipa XML (eXtensible Markup Language) u koji su ugrađene mogućnosti proširenja. Prednosti korištenja XHTML-a obuhvataju kompatibilnost sa starijim i novijim verzijama, izvršavanje na različitim uređajima za pristupanje Webu - kao što su mobilni telefoni i ručni računari, i mogućnost proširenja.

Ne odlazite snimanje Web strane dok na nju ne stavite tekst ili slike, već ih snimajte čim otvorite nov dokument. Ukoliko najprije snimate datoteku, sve putanje, koje referenciraju slike i ostale elemente koje budete uvozili, biće propisno formirane. Ukoliko ne snimate datoteku, za opis mjesta elementa koji se uvozi koristit će se referenca "file://putanja", relativno u odnosu na hard disk. Ako pokušate da umetnete objekat na stranu, a da prethodno niste snimili dokument, Dreamweaver će vas upozoriti da za taj element mora da upotrijebi referencu file://putanja.

Jedna od najčešćih početničkih grešaka u HTML dokumentu su apsolutne putanje koje pokazuju na lokalni disk onoga ko je kreirao web prezentaciju umjesto na datoteku na serveru gdje je mogu vidjeti svi korisnici Interneta.

Možete da koristite nastavak imena (oznaku tipa) datoteka .html ili .htm. Kada snimate dokument, Dreamweaver mu automatski dodaje nastavak imena .htm ili .html. Podrazumijevani nastavak imena možete da vidite kada izaberete Edit, Preferences i kategoriju New Document – on je prikazan u tekstualnom polju, mada je nedostupan (sive boje). Oba nastavka imena predstavljaju istu vrstu datoteke, ali na istom serveru (i u istom direktoriju) možete da čuvate i uvod.html i uvod.htm, jer te datoteke ipak imaju različita imena. Obično je lakše držati se jednog ili drugog, kako biste izbjegli greške prilikom pravljenja hiperveza. (Hiperveza s datotekom uvod.html ne povezuje posjetioca s datotekom uvod.htm.)

Davanje imena datotekama koje će se koristiti na Web serveru razlikuje se od imenovanja datoteka za korišćenje sa lokalnog diska. Prvo, morate da saznate operativni sistem koji se izvršava na serveru – najčešće su to Unix, Linux, Windows NT (Windows 2000 server) i Macintosh. Struktura imena se razlikuje na svakoj od ovih platformi. Primjera radi, Unix i Linux razlikuju mala i velika slova u imenima i ekstenzijama datoteka, pa u njima datoteka_1.htm nije isto što i DATOTEKA_1.HTM. Korištenje malih slova pojednostavljuje imenovanje datoteka i olakšava održavanje ujednačenosti. Za imena datoteka upotrebljavajte samo slova (A-Z) i brojeke (0-9). Evo još nekih važnih konvencija:

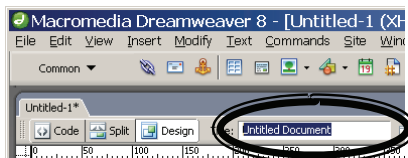
- U imenima datoteka ne smije biti razmaka (praznih mjesta). Ako treba razdvojiti riječi, upotrijebite podcrtu (_) ili crticu (-). Razmaci izazivaju probleme zato što ih browseri zamjenjuju oznakom %20.
- Ne smijete koristiti specijalne znakove kao što su %, *, > ili /.
- Ne počinjite imena datoteka brojkom.

Koristite što kraća imena datoteka i direktorija. Vodite računa o tome da ime direktorija postaje sastavni dio URL adrese koju korisnik mora da otkuca da bi pristupio strani. Prilikom davanja imena direktorijima, pridržavajte se gore navedenih smjernica za imenovanje datoteka.

5.5.1. Dajte naslov web strani

Svaki HTML dokument mora imati naslov, koji služi prvenstveno za identifikaciju. On se prikazuje na naslovnoj traci browsera, ukazuje na sadržaj strane i automatski postaje ime obilježivača (eng. Bookmark, Favorite) koji korisnik može da priloži svojoj kopiji Web strane. Birajte kratke, informativne izraze koji opisuju namjenu dokumenta. Neka vam pređe u naviku da svakoj strani date naslov prije nego što počnete da joj dodajete tekst i slike. Ako zaboravite na to, Dreamweaver će strani dati ime Untitled Document. Na Internetu možete prepoznati stranice koje su uradili loši dizajneri upravo po ovom detalju.

Upišite Ispitni zadaci u polje Title na paleti alatki dokumenta. Pritisnite Enter.



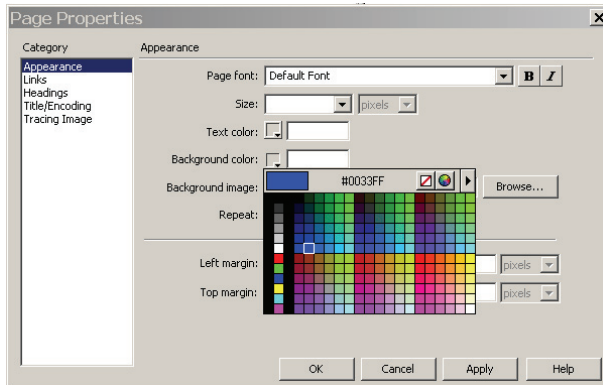
Slika 26. Naslov web stranice

Ako ne vidite paletu alatki dokumenta na kojoj je polje Title, izaberite opciju menija View, Toolbars, Document. Polje Title na početku prikazuje naslov "Untitled Document", pa ćete sada zamijeniti taj naslov pravim naslovom uvodne strane prezentacije.

5.5.2. Zadavanje boje pozadine

U Dreamweaveru se boja pozadine lako mijenja pomoću specijalne palete boja za Web (eng. Web-safe color palette). Na toj paleti ima 216 boja koje jednako izgledaju u Windowsu i na Macintoshu. Pristupit ćete toj paleti iz okvira za dijalog Page Properties i promijeniti boju pozadine dokumenta. Izaberite Modify, Page Properties. Otvorit će se okvir za dijalog Page Properties. U njemu nema podrazumijevanih opcija, premda je u Dreamweaveru podrazumijevana boja

pozadine u prozoru dokumenta bijela. Ako ne zadate boju pozadine, strana će se prikazati s podrazumijevanom bojom pozadine koja je definisana u browseru korisnika. Pritisnite polje s bojom do opcije Background. Prikazat će se paleta boja, a pokazivač miša će se pretvoriti u pipetu. Dovedite pipetu iznad uzorka boje na paleti.



Slika 27. Izbor boje pozadine stranice

Uočite da se heksadecimalne šifre boja prikazuju na vrhu palete dok mišem prelazite preko uzoraka. Boja se u HTML-u definiše heksadecimalnim brojem koji opisuje količinu njene crvene, zelene i plave komponente. Heksadecimalni brojevi koriste bazu 16, a za cifre koriste brojeke 0-9 i slova A-F (koja odgovaraju ciframa 10, 11, 12, 13, 14 i 15). Od šest cifara koje se u HTML- u koriste za opis boje, prve dvije označavaju iznos crvene komponente, druge dvije iznos zelene komponente, a posljednje dvije – plave komponente. Na primjer, broj #00FF00 označava boju koja nema crvenu komponentu (na prva dva mjesta su nule), ima jaku zelenu komponentu (FF označava maksimalnu količinu zelene komponente, jer je F najveća cifra u heksadecimalnom brojnom sistemu), i uopšte nema plavu komponentu; boja #000000 nema ni crvene, ni zelene, ni plave komponente – to je crna. Pipetom možete da birate boju pozadine i tako što ćete u prozoru dokumenta uzimati "uzorak" boje iz tekstova i slika. Prilikom izbora boja treba voditi računa i o vizuelnom kvalitetu strane; nemojte kombinovati slične boje za tekst i za podlogu, a izbjegavajte i drečave boje, radi kojih se stranica teško čita. Nećete pogriješiti ako izaberete kombinaciju boja sa neke postojeće web stranice koju su radili profesionalci.

Pritiskom na dugme Apply primijenit ćete sve izabrane parametre, a okvir za dijalog Page Properties ostat će otvoren. Ako hoćete da zatvorite okvir za dijalog, pritisnite OK i svi izabrani parametri će se automatski primijeniti prije zatvaranja.

Ako hoćete veći izbor boja, pritisnite strelicu u gornjem desnom uglu prozorčića u uzorcima boja i u meniju koji će se otvoriti izaberite paletu boja. Imajte u vidu da ostale palete ne sadrže boje koje jednako izgledaju na svim platformama.

5.5.3. Slika na pozadini strane

Na pozadinu se obično stavlja mali grafički objekat, čijim kopijama se popločava cijela strana. Strana može imati definisanu i boju i sliku za pozadinu. Ako je veza korisnika s Internetom spora ili mu je browser spor, prvo će se prikazati boja pozadine, što je dobar razlog da je zadate čak i u slučaju da namjeravate da stavite sliku na pozadinu. Učitana slika za pozadinu prikazuje se i prekriva definisanu boju pozadine.

Izaberite Modify, Page Properties. Otvorit će se okvir za dijalog Page Properties. Pritisnite dugme Browse do polja Background Image. Izaberite sliku; zatim pritisnite OK da bi se zatvorio okvir za dijalog Page Properties.

Sve što ste izabrali primjenjuje se na dokument; vidjet ćete kako slika popločava prozor dokumenta.

Ako treba da uklonite sliku sa pozadine, otvorite okvir za dijalog Page Properties i obrišite ime datoteke u polju Background. Ako niste snimili datoteku odmah po otvaranju dokumenta, u polju Background se prikazuje cjelokupna putanja izabrane slike na disku. Kada snimite datoteku, putanja će se izmijeniti i pokazivat će mjesto čuvanja slike u odnosu na lokalni korijenski direktorij. Uvijek treba snimiti datoteku prije uvoženja bilo kakvog sadržaja strane, čak i slike za pozadinu.

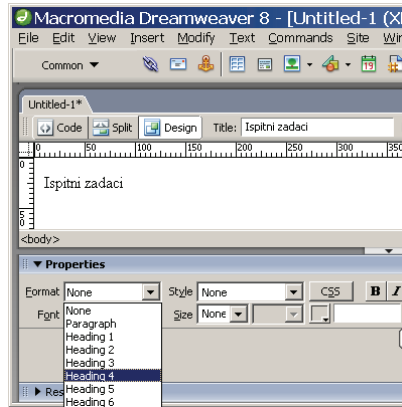
5.5.4. Zadavanje podrazumijevane boje fonta

Kada izmijenite boju pozadine ili stavite sliku na nju, može postati uputno da se izmijeni i boja slova teksta koji se prikazuje u prvom planu: primjera radi, crni tekst se ne vidi na crnoj pozadini: prilikom biranja šeme boja za dokument, izaberite kombinacije boja koje se dobro slažu i koje su dovoljno kontrastne. Slične boje se teško razlikuju, a isto važi i za komplementarne boje, naročito na ekranu monitora.

Izaberite Modify, Page Properties. Otvorit će se okvir za dijalog Page Properties. Pritisnite polje za boju teksta, desno od odrednice Text. Prikazat će se paleta boja kao ona za boju pozadine. Pipetom izaberite boju s palete, ili upišite njen heksadecimalni kod u tekstualno polje. Kada pritisnete OK, zatvorit će se okvir za dijalog Page Properties i vratit ćete se u dokument.

5.6. Smještanje teksta na stranu

Tekst možete da upišete neposredno u dokument. Obično se na početku stranice nalazi neki naslov, koji se označi tagom "Heading". U padajućoj listi Format na panou Properties izaberite opciju Heading 4. Upisanom tekstu ste dali HTML oznaku naslova četvrtog nivoa.



Slika 28. Izbor oznake naslova 4. nivoa (Heading 4)

HTML ima šest nivoa naslova, s oznakama od Heading 1 do Heading 6. Naslov prvog nivoa, Heading 1, ima najveća slova. Naslovi se prikazuju većim ili debljim slovima od ostatka teksta. Kada pasusu dodate HTML oznaku naslova (eng. Heading), automatski se ispod njega dodaje određen prazan prostor, čiju veličinu ne možete da mijenjate.

U mnogim dokumentima, prvi naslov na strani jednak je naslovu prezentacije. Ako dokument ima više dijelova, tekst prvog naslova treba da se odnosi na prvi dio dokumenta, npr. naslov prvog poglavlja. Prethodno zadati naslov prezentacije treba da odredi dokument u širem kontekstu, npr. da sadrži i naslov knjige i naslov poglavlja.

Kada pritisnete Enter, time ste ispod naslova "Ispitni zadaci" dodali još jedan red, koji podrazumijevano dobija HTML oznaku pasusa. Za običan tekst koristi se, po pravilu, format pasusa.

Kad god izmijenite dokument, Dreamweaver dodaje zvjezdicu (*) do imena datoteke na vrhu prozora dokumenta. Ona označava da je datoteka promijenjena, a da potom nije snimljena. Zvjezdica će nestati kada snimate dokument. Često snimajte dokument kako biste umanjili rizik od gubitka uloženog rada kada se Dreamweaver ili operativni sistem sruše (radi brzine koristite: Ctrl+S).

5.6.1. Uvoženje teksta

Tekst se strani može dodati kopiranjem i prenošenjem iz drugog dokumenta. Ako tekst potiče iz aplikacije koja podržava kopiranje prevlačenjem i puštanjem (poput Microsoftovog Word-a, na primjer), možete da otvorite i Dreamweaver i tu aplikaciju i zatim da kopirate i umetnete tekst u Dreamweaver, ili da ga izaberete i prevučete u Dreamweaver.

Dreamweaver može da otvori datoteke napravljene u programima za obradu teksta ili programima za prelom strana, ukoliko su snimljene kao ASCII datoteke (čist

tekst). Na primjer, Dreamweaver može da otvori dokument Microsoftovog Worda, ako je bio snimljen kao tekstualna datoteka (s nastavkom .txt) ili kao HTML dokument, što se postiže biranjem opcije menija File, Save as Web Page. Tekstualne (.txt) datoteke se u Dreamweaveru uvijek otvaraju u novom prozoru, u prikazu Code View. Kada otvorite tekstualnu datoteku u Dreamweaveru, tekst možete kopirati i prenijeti u druge datoteke.

Jednostavne formate dokumenta, kao što su znakovi za kraj redova i za kraj pasusa, možete da zadržite, ali treba da razumijete razliku između ASCII formata na Windows i Macintosh platformama. Datoteke napravljene u Windowsu imaju nevidljivi kontrolni znak za prelazak u novi red (eng. Line feed), što označava mjesto u tekstu gdje treba preći u novi red. Macintosh računari ne koriste znak za kraj reda. Ako Windowsovu tekstualnu datoteku otvorite u aplikaciji SimpleText na Macintoshu, vidjet ćete mali pravougaonik na početku svakog pasusa, koji stoji umjesto znaka za kraj reda. Ako Macintosh tekstualnu datoteku otvorite u Windowsu, svi njegovi pasusi će biti spojeni, jer u toj datoteci nema znaka za prelazak u novi red.

Radi bržeg rada, koristi se tipka Enter za prelazak u novi red sa oznakom <p>, odnosno kraj pasusa, a Shift+Enter za prelazak u novi red bez završetka pasusa, tj. za oznaku
.

5.6.2. Centriranje i uvlačenje teksta

Mogućnosti za poravnanje teksta su podrazumijevani izgled (bez poravnanja), poravnanje ulijevo (eng. align left), centralno poravnanje (eng. justify). Podrazumijevano je poravnanje ulijevo.

Ako su dva reda dio istog pasusa, sva formatiranja pasusa koja primijenite, npr. poravnanje ili zadavanje nivoa naslova, utjecat će na cjelokupan sadržaj pasusa.

S dva uzastopna znaka za prelazak u novi red simulirate izgled novog pasusa. Međutim, to je samo privid – i dalje se radi o jednom pasusu, pa možete imati poteškoća kada pokušate da na taj tekst primijenite stilove formatiranja.

Osim poravnanja, pasuse teksta možete i da uvučete od margina. Za to se koriste dugmad Text Indent i Text Outdent na panou Properties (slika 29).

Različitim uvlačenjem teksta izdvajate dijelove teksta. Alatkom Text Indent tekst se uvlači u odnosu na obje margine, i lijevu i desnu. Veličinu tog uvlačenja ne možete da zadate, jer se ona mijenja u zavisnosti od browsera koji prikazuje tekst.

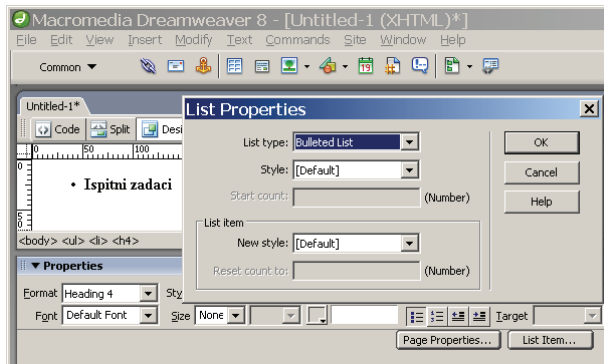
Ako pri uvlačenju jednog pasusa, uvučete više susjednih pasusa, provjerite da li su umjesto znaka za prelazak u novi pasus upotrijebljena dva uzastopna znaka za prelazak u novi red. Dovedite pokazivač miša na početak pasusa koji hoćete da uvučete. Zatim pritisnite Backspace sve dok ne dođete do kraja prethodnog pasusa i potom pritisnite Enter, kako biste zaista završili prethodni pasus i prešli u novi.



Slika 29. Alati za poravnanje teksta, označavanje spiskova i za uvlačenje teksta

5.6.3. Pravljenje spiskova

Dreamweaver omogućava pravljenje dvije vrste spiskova (lista): neuređenih i uređenih. Uređena lista (eng. ordered list) sadrži stavke odvojene rednim brojevima ili slovima abecede, i poredane u numerisani, odnosno abecedni spisak. Možete da koristite arapske ili rimske brojeve, odnosno velika ili mala slova abecede. Neuređena lista (eng. unordered list) često se naziva lista nabiranja, jer svaka stavka u njoj počinje znakom za nabiranje (bulletom). Podrazumijevani znak za nabiranje koji Dreamweaver prikazuje može biti zamijenjen punim krugom, kružnicom ili kvadratom.



Slika 30. Osobine neuređene liste i tipka "List Item..."

Šemu numerisanja uređene liste promijenit ćete izmjenom njenih svojstava. Pritisnite bilo koji red spiska. Zatim pritisnite dugme List Item na panou Properties.

Izaberite samo jedan red liste. Ako izaberete cijelu listu, dugme List Item će postati nedostupno i sivo. Ako se dugme ne vidi, pritisnite strelicu za proširenje u donjem desnom uglu panoa Properties. Otvorit će se okvir za dijalog List Properties. Uređena lista se u padajućoj listi List Type naziva Numbered List.

U padajućoj listi Style izaberite opciju Alphabet Small (a, b, c). Zatim pritisnite OK. Alphabet Small se prikazuje kao opcija u padajućoj listi Style samo ako ste pritisnuli dugme Ordered List. Ako ste pritisnuli dugme Unordered List, prije biranja stila morate da izaberete opciju Numbered List u padajućoj listi List Type. Redni brojevi liste pretvoreni su tako u mala slova abecede.

Područje List Item na dnu okvira za dijalog sadrži padajuću listu New Style, pomoću koje mijenjate izgled pojedine stavke ili grupe stavki spiska, umjesto cijelog spiska. U istom području je i polje Reset Count To, koje služi za proizvoljno zadavanje rednog broja stavke u kojoj se nalazi tačka umetanja; sve naredne stavke automatski dobijaju slijedeće redne brojeve.

Slijedi opis postupka izrade i modifikovanja neuređene liste. Izaberite dio teksta i pritisnite dugme Unordered List na panou Properties. Izabrani tekst će biti uvučen i dobit će znakove za nabranjanje.

Podrazumijevani znak za nabranjanje neuređenih lista možete da promijenite imenom odgovarajućih svojstava, isto kao kod uređenih lista. U padajućoj listi List Type, neuređena lista se naziva Bulleted List. Pritisnite bilo koji red u listi i izaberite opciju menija Text, List, Properties. Otvorit će se okvir za dijalog List Properties. U padajućoj listi Style izaberite npr. opciju Square i pritisnite OK. Sve stavke u listi dobit će kvadrat (eng. square) kao znak za nabranjanje.

Boja rednih brojeva i znakova za nabranjanje, koji se koriste u uređenim i neuređenim listama, jednaka je podrazumijevanoj boji teksta dokumenta, koju zadajete u okviru za dijalog Page Properties.

5.6.4. Ugnježdivanje lista

Liste se mogu ugnježdivati, odnosno stavljati unutar drugih lista. Vrstu liste je moguće promijeniti i kada je ugnježđena. Primjera radi, unutar neuređene liste možete imati uređenu listu. Kada ugnježđujete liste, podrazumijevano se koriste puni krugovi, kružnice i kvadrati (tim redom) kao znakovi za nabranjanje. U Dreamweaveru se ti znakovi za nabranjanje nazivaju bullet, circle i square. Odgovarajući termini u HTML-u su disc, circle i square.

Neki browseri prikazuju šuplje kvadrate kao znakove za nabranjanje. Takav je, na primjer, Netscape 4.7 za Macintosh. Internet Explorer 5.0 za Macintosh prikazuje ispunjene kvadrate. U Windowsu su kvadrati ispunjeni.

Da biste ugnijezdili stavku koju ćete napraviti, pritisnite dugme Text Indent na panou Properties. Stavka će biti uvučena na slijedeći nivo.

Ako hoćete da uklonite ugnijezđenu listu, stavite kursor u nju, ali nemojte je označiti. Pritisnite dugme Text Outdent na panou Properties i zatim uklonite ugnijezđeni tekst.

5.6.5. Formatiranje znakova

Da biste istakli određene stavove, riječi ili izraze, u Dreamweaveru možete na tekst da primijenite više opcija formatiranja. To su polucrni (podebljani) ispis (eng. bold), kurzivni ispis (eng. italic) i podvlačenje (eng. underline) teksta.

Za ispisivanje izabranog teksta polucrnim slovima možete da upotrijebite i opciju menija Text, Style, Bold. Prečica s tastature za to je Ctrl+B. Na isti način ćete tekst ispisati kurzivom (eng. italic), kombinacijom tipki Ctrl+I.

Izbjegavajte podvlačenje teksta na Web stranama, jer se tako najčešće označavaju hiperveze, pa ćete podvlačenjem teksta koji nije hiperveza možda zbuniti posjetioce.

5.6.6. Promjena fonta

Web strane će biti zanimljivije i lakše će se čitati ako promijenite font kojim se prikazuje tekst. Premda se o korištenju različitih slova u štamparstvu mnogo zna, sve to ne važi na Webu. Morate voditi računa o činjenici da korisnici mogu da mijenjaju veličinu prozora i fonta, kao i boju slova. Izgled teksta na Web strani mijenja se od korisnika do korisnika. Najveća razlika je u veličini teksta u Windowsu i na Macintoshu. Na Macintosh računarima tekst se prikazuje otprilike 25 odsto manjim slovima nego na Windows računarima.

Ako ste navikli da raspoređujete elemente na stranicama štampanih medija (kao što su časopisi i brošure), razočarat ćete se, jer u HTML-u ne možete da utičete npr. na prored teksta i međuslovna rastojanja. U Web tekstu ne možete da utječete na prored teksta u pasusima.

Sanserifni fontovi (npr. Arial) se na ekranima monitora po pravilu čitaju lakše od serifnih fontova (npr. Times New Roman).

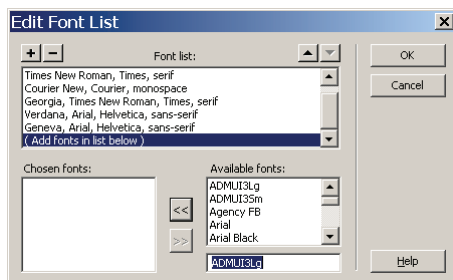
Da bi posjetilac vidio stranu onako kako ste je napravili, mora na svom računaru imati instaliran font koji ste zadali. Nemojte misliti da svi korisnici imaju sve fontove. Kombinacije instaliranih fontova čine da browser prikazuje tekst u drugom fontu. Ako prvi font u spisku fontova nije instaliran, browser će pokušati da upotrijebi drugi, a na kraju i riječi. Ako nijednog od tih fontova nema na korisnikovom računaru, tekst se prikazuje u podrazumijevanom fontu browsera.

Kombinacije fontova (kao što su Arial, Helvetica, Sans-serif) jesu korisne, ali ne obuhvaćaju uvijek baš one fontove koje ste htjeli da upotrijebite. Kombinaciju fontova možete da promijenite kada izaberete opciju Edit Font List u padajućoj listi Font na panou Properites, odnosno kada pomoću opcije menija Text, Font, Edit Font List otvorite okvir za dijalog Edit Font List.

Izaberite kombinaciju fontova i zatim jednu od slijedećih opcija:

- Da biste dodali ili uklonili font: izaberite font i pritisakajte dugmad sa strelicama, koja premješta fontove između spiskova Chosen Fonts (izabrani fontovi) i Available Fonts (dostupni fontovi).
- Da biste dodali ili uklonili kombinaciju fontova: pritisakajte dugmad sa znakovima plus i minus, u gornjem lijevom uglu okvira za dijalog.

- Da biste dodali font koji nije instaliran na sistemu: upišite ime fonta u polje ispod liste Available Fonts i pritisnite dugme sa dvostrukom strelicom; time ćete izabrani font dodati kombinaciji. Dodavanje fonta koji nije instaliran na sistemu korisno je, na primjer, za zadavanje određenog Windowsovog fonta kada radite na Macintoshu.
- Da biste premjestili kombinaciju fontova naviše ili naniže u listi: pritisnite dugmad sa trouglicima okrenutim nagore i nadole u gornjem desnom uglu okvira za dijalog.



Slika 31. Izbor kombinacije fontova

Možete da uklonite podešene parametre fontova i vratite tekst u podrazumijevani font: najprije izaberite tekst ispisan fontom koji hoćete da uklonite, a potom izaberite opciju Default Font u padajućoj listi Font na panou Properties. Možete i da izaberete opciju menija Text, Font, Default Font.

5.6.7. Promjena veličine fonta

U HTML-u su mogućnosti mijenjanja veličine fonta prikazanog na strani ograničene. Izgled strane ćete bolje kontrolisati ako koristite kaskadne opise stilova (eng. cascading style sheets, CSS).

Veličine fontova u listi poslagane su od 1 do 77, od +1 do +7 i od -1 do -7. Ako izaberete broj bez predznaka (1 je najmanji font, a 7 najveći), zadali ste apsolutnu veličinu fonta. Biranjem broja s predznakom plus ili minus, zadajete relativnu veličinu fonta u odnosu na njegovu osnovnu veličinu. Broj +1, na primjer, zadaje font za jedan broj veći od njegove osnovne veličine.

Podrazumijevana osnovna veličina fonta u browseru je 3. Ako izaberete +3 za veličinu fonta, zapravo ste zadali 6 (3+3). Najveća veličina fonta je 7, a najmanja 1. Svaki font veći od 7 prikazuje se veličinom 7; primjera radi, ako izaberete veličinu fonta +6, 3+6 je veće od 7, ali će se ipak prikazati font veličine 7.

Korištenje relativnih veličina fontova omogućava veću kontrolu nad izgledom strane kao cjeline, odnosno nad izgledom dijela teksta u odnosu na ostatak strane; po pravilu, korisnici lakše rade sa relativnim veličinama nego sa apsolutnim.

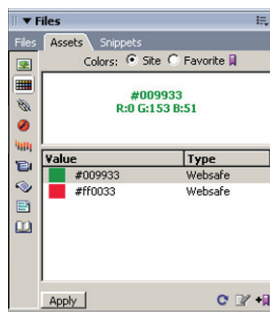
5.6.8. Mijenjanje boje slova

U Dreamweaveru se boja teksta lako mijenja.

Prozor za biranje boja se automatski zatvara kada pritisnete neki uzorak boje, koju Dreamweaver odmah primjenjuje na tekst. Prozor za biranje boja možete otvoriti i pomoću opcije menija Text, Color, čime nekoliko takvih prozora postaje dostupno.

Sve boje koje upotrijebite u prezentaciji čuvaju se na panou Assets. Da biste isti skup boja mogli dosljedno da koristite u cijeloj prezentaciji, snimite najčešće korištene boje u direktorij Favorites na panou Assets: Pritisnite strelicu lijevo od grupe panoa Files. Prikazat će se grupa panoa Files, u kojoj se nalazi i pano Assets. Ako grupa panoa Files nije otvorena, izaberite Window, Assets. Na panou Assets pritisnite ikonicu Colors u lijevoj koloni, a zatim radio-dugme Site pri vrhu panoa. Prikazat će se sve boje koje ste već koristili u prezentaciji "Ispitni zadaci".

Izaberite jednu od boja iz spiska boja upotrijebljenih u prezentaciji, prikazanog na panou Assets. Pritisnite dugme Add to Favorites u donjem desnom uglu panoa Assets. Dreamweaver će prikazati okvir za dijalog s porukom da je izabrana boja dodata u listu Favorites. Da biste vidjeli spisak najčešće korištenih boja, Favorites, pritisnite istoimeno radio-dugme pri vrhu panoa Assets.

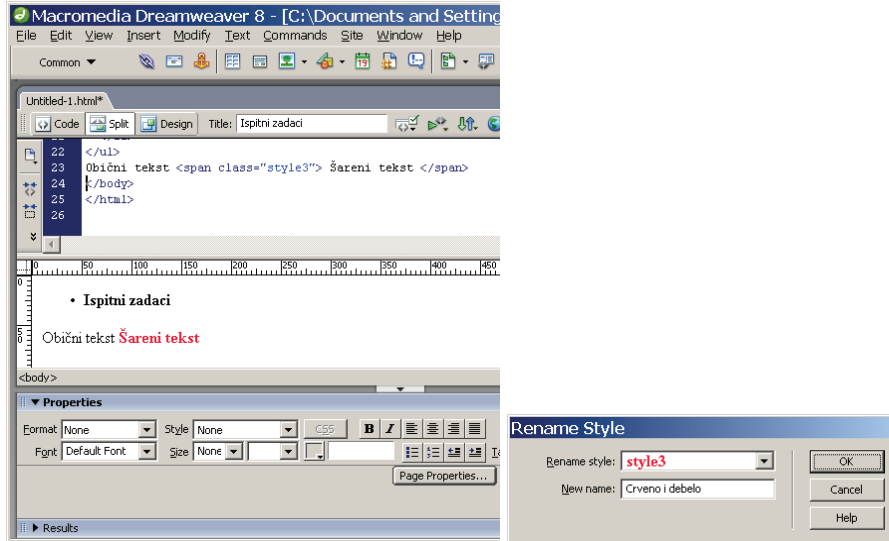


Slika 32. Lista korištenih boja u dokumentu

5.7. Definisanje html stilova

Da bi Web strane bile zanimljivije, ispisuju se raznim fontovima. Oblik, veličinu i boju slova možete lako da mijenjate, ali ako hoćete da upotrijebite nekoliko kombinacija boja, oblika i veličina slova za cijelu prezentaciju, te kombinacije parametara morali biste da pamtite od strane do strane i od jedne selekcije do druge. HTML stilovi čuvaju formatiranje slova i pasusa izabranog dijela teksta, koje možete ponovo da upotrijebite u drugim pasusima istog dokumenta i u drugim dokumentima. HTML stilovi čine da dokumenti izgledaju ujednačenije, postupak formatiranja teksta je znatno ubrzan, a iste stilove možete da koristi cijeli tim dizajnera. Ako izmijenite ili obrišete određeni HTML stil, to nimalo ne utiče na izgled teksta već formatiranog tim stilom.

Od teksta koji je formatiran na način koji će se koristiti i na drugim dijelovima teksta u dokumentu ili u drugim dokumentima u web prezentaciji, može se kreirati html stil. Dreamweaver automatski kreira stilove sa svakom promjenom teksta i dodaje im imena Style 1, Style 2, Style 3,...



Slika 33. Tekst prikazan kao HTML kod, u "Design" načinu prikazivanja i način promjene imena predefinisano stila

Na slici 33. je prikazano kako se dio teksta "Šareni tekst" nalazi unutar oznaka `<span class="Style 3">` i `</span>`. U padajućem meniju Style se naredbom "Rename..." može promijeniti podrazumijevano ime stila.

5.7.1. Primjena html stilova

Nakon što ste definisali nekoliko HTML stilova, vrijeme je da ih primijenite. Primjena stila na izabrani tekst znači da tekst prethodno morate da označite. Primjena stila na cijeli pasus zahtjeva samo da kursor bude unutar tog pasusa – nema potreba da birate tekst cijelog pasusa.

Da biste izmijenili određeni HTML stil, ne smije biti izabranog teksta u prozoru dokumenta. Jedan način da otvorite okvir za dijalog Define HTML Style jeste da dvaput pritisnete ime određenog stila. Ako nešto izmijenite, npr. atribut fonta, i pritisnete OK, izmijenjeni stil će se primjenjivati ubuduće, na nov tekst koji budete unijeli dok je izabran taj stil, odnosno na tekst na koji ga primijenite, a ne na tekst prethodno formatiran stilom istog imena. Ako hoćete da tekući tekst bude formatiran novim stilom, stil morate ponovo da primijenite na njega.

Brisanje stila umije da bude zamršeno. Najprije morate da provjerite da nema izabranog teksta u dokumentu i da polje Auto Apply u donjem lijevom uglu panoa

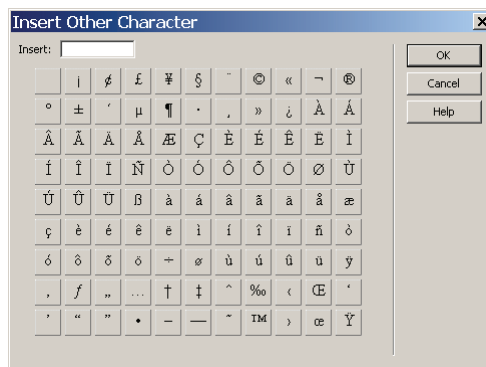
HTML Styles nije potvrđeno. Zatim treba da izaberete stil koji hoćete da obrišete i da pritisnete sličicu korpe za otpatke u donjem desnom uglu panoa. Ako zaboravite da isključite opciju Auto Apply, a ima izabranog teksta, taj tekst će možda biti formatiran stilom koji ste htjeli da uklonite. Formatiranje teksta možete da uklonite kada izaberete tekst i zatim opciju Clear Paragraph Style ili Clear Selection Style na panou HTML Styles. Tekst će izgubiti sve formatiranje, bez obzira na to da li mu je ono preneseno pomoću HTML stila ili pomoću panoa Properties. Ako brišete stil pasusa, a kursor možete da smjestite bilo gdje u pasus – ne morate da birate cijeli pasus.

HTML stilovi koje napravite mogu se primijeniti u drugim projektima i dijeliti s ostalim korisnicima. Kada definišete HTML stil, u direktoriju Library se snima datoteka Styles.html. Tu datoteku možete da kopirate i premjestite u druge direktorije prezentacije, ali da je podijelite s ostalim saradnicima.

5.8. Dodavanje specijalnih znakova

Prilikom rada u Dreamweaveru katkada zatrebaju znakovi i ostale informacije kojima ne možete pristupiti direktno sa tastature. Ovi specijalni znakovi imaju specifične HTML kodove ili alternativne prečice s tastature koje je teško zapamtiti.

Zadajte naredbu Insert, HTML, Special Characters, a zatim izaberite znak koji treba ubaciti u tekst. Ako tog znaka nema na spisku ponuđenih, izaberite naredbu Other...



Slika 34. Dodavanje specijalnih znakova

5.9. Dodavanje horizontalnih linija

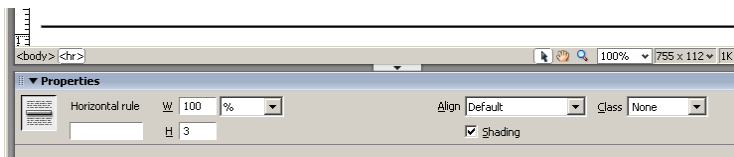
Horizontalna linija se pruža preko cijele strane i vizuelno odvaja dijelove dokumenta. Zadajte naredbu Insert, HTML, Horizontal Rule. Dok je horizontalna linija izabrana, upišite 70 u polje W na panou Properties. Izaberite % u padajućoj listi desno od broja koji ste upravo upisali.

Horizontalna linija će tako zauzeti 70 odsto širine (eng. width, W) prozora browsera, bez obzira na širinu samog prozora. Linija se prikazuje kao tanka pruga.

Ako hoćete da zadate apsolutnu širinu linije, izaberite opciju pixels u padajućoj listi. U tom slučaju, linija ne mijenja širinu kada korisnik promijeni veličinu prozora browsera.

Pritisnite polje Shading kako biste uklonili znak za potvrdu iz njega. U polje H upišite 1. Kad se ukloni potvrda iz polja Shading, prikazuje se puna linija. Horizontalna linija je visoka (eng. height, H) 3 piksela.

Horizontalnu liniju možete i da poravnate s lijevom marginom, sa desnom marginom ili centralno. Podrazumijevano poravnanje horizontalne linije je centralno.



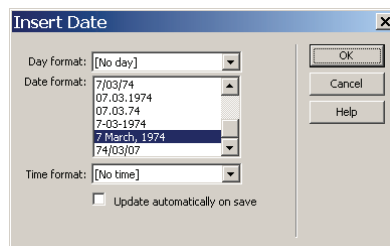
Slika 35. Osobine horizontalne linije

Izgled linije možete da promijenite kada je izaberete i promijenite joj parametre na panou Properties.

5.10. Automatsko dodavanje datuma

Katkada treba voditi računa o tome kada ste posljednji put izmijenili stranu u prezentaciji. Dreamweaver može da stavi datum i vrijeme posljednje izmjene na sve strane prezentacije i da ih automatski ažurira svaki put kada snimate datoteku, da to ne biste radili ručno.

To nije dinamički datum koji se mijenja svaki put kada korisnik pristupi strani, nego datum koji govori posjetiocima kada ste posljednji put ažurirali sadržaj Web lokacije. Postavite kursor tamo gdje hoćete da umetnete datum. Datum se obično prikazuje na dnu strane. Okvir za dijalog Insert Date može da otvorite pomoću opcije menija Insert, Date.



Slika 36. Dodavanje datuma

Potvrdite polje Update Automatically on Save, pa će se datum ažurirati svaki put kada snimate dokument. Prikazuju se tekući dan, datum i vrijeme. Te informacije će se ažurirati svaki put kada snimate dokument. Thursday je u okviru za dijalog Insert Date korišten kao primjer koji pokazuje kako će se u dokumentu prikazivati ime dana u sedmici. Primjeri su i izabrane opcije za datum i vrijeme.

Format datuma možete da promijenite ako se ažurira automatski. Pritisnite datum u dokumentu. Zatim na panou Properties pritisnite ikonicu Edit Date Format. Otvorit će se istoimeni okvir za dijalog, jednak okviru za dijalog Insert Date, ali u njemu možete da mijenjate parametre. Promijenite šta hoćete i pritisnite OK. Novi parametri će se primijeniti na dokument.

5.11. Umetanje grafičkih elemenata na stranu

Najrašireniji i najbolje podržani formati slika na Webu jesu format za razmjenu grafičkih elemenata (eng. Graphic Interchange Format, GIF) i format združene grupe fotografskih stručnjaka (eng. Joint Photographic Experts Group, JPEG).

Kada odlučujete da li da snimate grafiku u formatu GIF ili JPEG, neka vam cilj bude najbolji kvalitet slike i što manja datoteka.

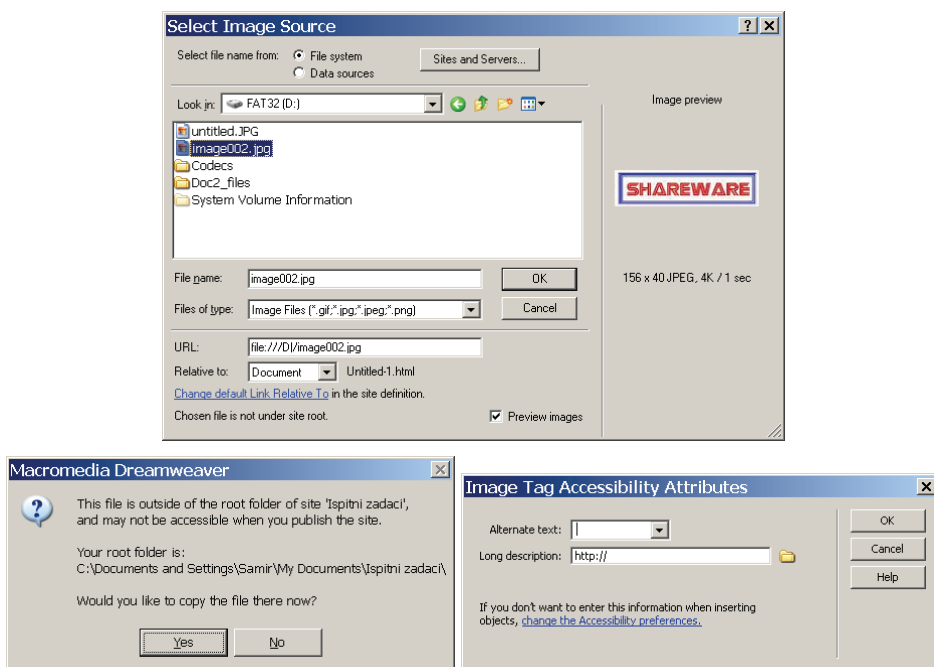
Po pravilu, GIF se koristi kada slika ima velike jednobojne površine bez tekture i malo ili nimalo prelijevanja boja. GIF je dobar pri radu s tekстом, vektorskom grafikom, slikama s ograničenim brojem boja i slikama malih dimenzija. GIF slike se mogu snimiti s maksimalno 256 boja, koliko može da se zapamti u 8 bita odvojenih za opis boje svake tačke GIF slike. GIF datoteke se obično brže učitavaju, imaju više mogućnosti za optimizaciju i podržavaju providnost i animaciju.

JPEG se obično koristi za fotografije i druge slike s velikim opsegom boja. Format JPEG odlično reprodukuje prelive boja i daje fotografije mnogo boljeg kvaliteta od fotografija u formatu GIF, premda za to koristi mnogo manje datoteke. Format JPEG opisuje svaku tačku slike s 24 bita, čuva sve njene boje, ali je komprimuje uz gubitak redundantnih podataka. Što je JPEG nižeg kvaliteta, to se odbacivanjem podataka gubi više informacija.

Biranjem opcije menija Insert, Image otvorit će se okvir za dijalog Select Image Source, pomoću koga možete da umetnete grafički objekat na stranu.

Pritisnite dugme Preview Images ako hoćete da pogledate umanjeni prikaz svih popisanih grafičkih elemenata. Grafički objekti se prikazuju u okviru za dijalog Select Image Source, uz dimenzije slike, veličinu datoteke i približno vrijeme preuzimanja.

Po pravilu, treba koristiti reference u odnosu na dokument. Samo ako imate izuzetno veliku prezentaciju ili namjeravate često da premještate strane, opravdano je da koristite reference u odnosu na osnovni direktorij lokacije.



Slika 37. Umetanje slike u html stranicu

Dok ne snimate datoteku, Dreamweaver ne može da napravi ovakve reference, pa mora umjesto njih da pravi putanje na osnovu položaja slika na lokalnom disku, što nisu valjane hiperveze i ne rade kada se prezentacija postavi na Web. Obavezno snimate dokument prije nego što počnete da umećete grafičke elemente. Ako to ne uradite, Dreamweaver će prikazati upozorenje i sačekati dok ne snimate datoteku, da bi popravio putanju slika. Ako se slika koju ste izabrali u okviru za dijalog Select Image Source nalazi izvan lokalne prezentacije, Dreamweaver će prikazati i sljedeće upozorenje ispod menija Relative To: The file is outside the root folder (Izabrana datoteka se nalazi izvan osnovnog direktorija lokacije).

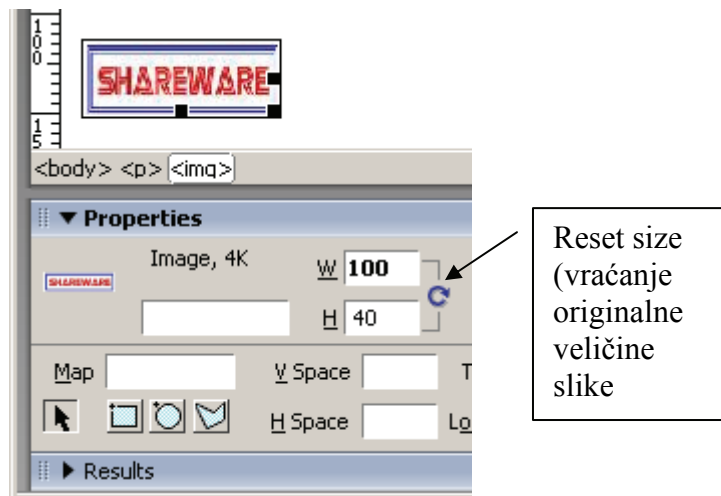
Od verzije 8 Dreamweaver prilikom svakog umetanja slike zahtijeva da se unese i alternativni tekst za svaku sliku. Tekst iz polja Alt prikazuje se umjesto slike ako je posjetilac onemogućio prikazivanje slika u svom browseru, ako njegov browser uopšte ne može da prikazuje slike ili ako učitavanje određene slike nije uspjelo. Umjesto slike, tada se prikazuje tekst koji ste upisali u polje Alt. Također služi i za indeksiranje stranica u pretraživačkim servisima. Svim grafičkim objektima koji su neophodni za navigaciju po lokaciji morate dodati HTML oznake <alt> (tj. upisati tekst u polje Alt). Dodavanje oznaka <alt> je korisno i kod ostalih grafičkih elemenata, jer ako korisnik onemogućio prikazivanje slika u svom browseru ili ima browser koji može da prikazuje samo tekst, ipak će vidjeti barem dio informacija koje bi mu inače promakle. Slabovide osobe koriste specijalne browsere, koji izgovaraju tekst <alt> oznaka, kao i tekst ispisana na Web strani. Konačno, <alt>

tekst se nakratko prikazuje u Internet Exploreru kada korisnik zaustavi pokazivač iznad grafike.

5.11.1. Mijenjanje veličine grafičkih elemenata i vraćanje na prvobitnu veličinu

Kada uvezete grafički objekat, njegova širina i visina prikazuju se na panou svojstava i automatski upisuju u HTML kod, što browseru daje podatke potrebne za definisanje izgleda strane. Veličina slike bitno utiče na brzinu njenog učitavanja. U Dreamweaveru možete da promijenite te dimenzije, korištenjem miša ili pomoću polja u panelu svojstava.

Koristite HTML oznake za širinu i visinu kako bi grafika izgledala manja, a da samu grafičku datoteku ne mijenjate. Veličina datoteke, prikazana na panou svojstava, nije se promijenila.



Slika 38. Promjena veličine slike: širina je smanjena neproporcionalno, te je u panou za promjenu osobina ispisana podebljano

Parametri širine i visine automatski se prilagođavaju novim dimenzijama slike i ispisuju podebljanim brojkama. Ovo formatiranje je pokazatelj da je grafičkom objektu mijenjana veličina. Dešava se da slučajno promijenite veličinu slike, pa podebljani brojevi treba da vas podsjetu da je veličina mijenjana. Veličinu slike možete promijeniti i upisivanjem novih vrijednosti u polja za širinu (eng. Width, W) i visinu (eng. Height, H).

Ako hoćete da odnos širine i visine slike ostane nepromijenjen prilikom mijenjanja njene veličine, držite taster Shift dok mišem povlačite ručice u uglovima slike.

Uočite kako je kvalitet slike opao kada ste je povećali. Slike se u browserima prikazuju u rezoluciji ekrana, koja iznosi 72 tačke po inču (eng. dots per inch, dpi). Ova rezolucija nije dovoljna za lijepo prikazivanje uvećanih slika. Sliku možete da smanjite i tada bi izgledala dobro, ali time ne mijenjate veličinu grafičke datoteke,

pa učitavanje slike traje jednako. Kada u Dreamweaveru mijenjate veličinu slike, utičete samo na veličinu slike prikazane u browserima, ne i na veličinu same grafičke datoteke. Da biste smanjili veličinu grafičkih datoteka na najmanju moguću mjeru, obavezno ih najprije prilagodite u programima za obradu slika kao što su Macromedia Fireworks ili Adobe Photoshop.

Pano svojstava za slike ima i druge Edit u gornjoj polovini i na desnoj strani panoa. Pomoću njega možete brzo da otvorite i dotjerate slike u spoljnim programima za obradu slika. Te programe birate pomoću opcije menija Edit, Preferences, kada treba da u okviru za dijalog Preferences izaberete File Types/Editors u spisku Categories. U ovom okviru za dijalog, grafičkim datotekama koje imaju različite nastavke imena možete da pridružite razne programe za obradu slika. Rad u programima Dreamweaver, Flash i Fireworks možete da objedinite pomoću tzv. kružnog uređivanja (eng. roundtrip editing), kojim se sve izmjene grafičkih datoteka učinjene u jednom programu, prenose i u ostala dva.

Dugmetom Reset Size na desnoj strani panoa svojstava slika će se vratiti na originalnu veličinu. Uočite da su širina i visina ispisane običnim (nepodebljanim) brojkama, što znači da je slika prikazana u originalnoj veličini.

Kada koristite veoma veliku sliku, ili sliku koja se čuva na drugom serveru, za početak prikazivanja možete umjesto nje da upotrijebite sličnu sliku niskog kvaliteta. Definisanjem izvorne slike niskog kvaliteta, dakle, birate sliku koja će se prva prikazati na datom mjestu. Kada se učitavanje slike visokog kvaliteta završi, ona će biti prikazana na istom mjestu. Kad god izaberete neku sliku, na panou svojstava se prikazuje polje Low Src za definisanje slike niskog kvaliteta. Da biste izabrali tu sliku, pritisnite ikonicu Browse for File (sa sličicom omotnice) do polja Low Src.

5.11.2. Pozicioniranje slika na strani

Kada smjestite sliku neposredno u tijelo dokumenta, nemate mnogo mogućnosti da je premještate. Sljedeća vježba prikazuje metodu za poravnavanje slika pomoću HTML oznaka <div>. Te oznake su kontejneri koji definišu poravnanje svega što se nalazi između njih.

Nakon što označite sliku, na panou svojstava pritisnite dugme Align Center. Slika će biti centrirana na strani.

Kada radite s više grafičkih elemenata, stavite svaki od njih u zaseban pasus, jer samo tako možete zasebno da ih poravnate. Ne možete centrirati dio pasusa, a ostatak poravnati s lijevom marginom. Ako hoćete da umetnete tekst neposredno ispod slike, treba da stavite pravi znak za kraj pasusa (pritisnite Enter). Ako stavite samo znak za prelazak u novi red (pomoću kombinacije tastera Shift+Enter), svaki pokušaj poravnanja utjecat će na sliku i na tekst, pošto su oni dio istog pasusa.

5.11.3. Dodavanje okvira oko slike

Katkada treba istaknuti sliku u odnosu na pozadinu. Jedan od načina da se to postigne jeste i dodavanje okvira oko slike. Okvir može da privuče pažnju na sliku i da doprinese ujednačenosti stila cijele Web prezentacije. Ponekad okvir može da posluži i kao hiperveza. Na panou svojstava upišite 1 u polje Border. Dreamweaver će oko grafike nacrtati okvir, debljine 1 piksel.

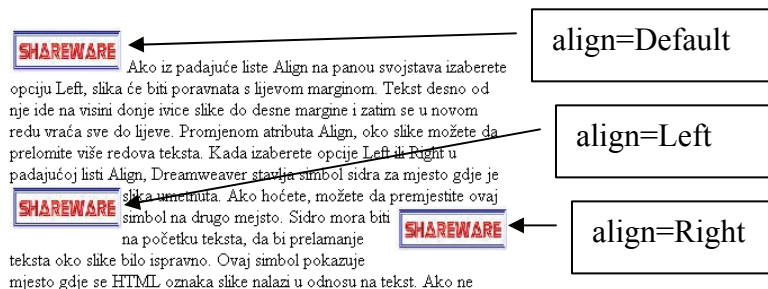
Debljinu okvira možete proizvoljno da zadajete. Boja okvira jednaka je podrazumijevanoj boji teksta, zadatoj u okviru u dijalog Page Properties.

Ako slici pridružite hiperlink, boja okvira te slike postat će jednaka podrazumijevanoj boji hiperlinkova, zadatoj u okviru za dijalog Page Properties. Na boju okvira možete da utičete i dodavanjem HTML oznaka za boju fonta prije i poslije HTML oznake same grafike, u HTML kodu.

5.11.4. Omotavanje teksta oko slika

Mogućnosti definisanja rasporeda elemenata strane u HTML-u obuhvataju i omotavanje teksta oko slika. Slika se prikazuje u podrazumijevanom položaju, u kome prvi red teksta počinje na visini donje ivice slike.

Ako iz padajuće liste Align na panou svojstava izaberete opciju Left, slika će biti poravnata s lijevom marginom. Tekst desno od nje ide na visini donje ivice slike do desne margine i zatim se u novom redu vraća sve do lijeve. Promjenom atributa Align, oko slike možete da prelomite više redova teksta. Kada izaberete opcije Left ili Right u padajućoj listi Align, Dreamweaver stavlja simbol sidra za mjesto gdje je slika umetnuta. Ako hoćete, možete da premjestite ovaj simbol na drugo mjesto. Sidro mora biti na početku teksta, da bi prelamanje teksta oko slike bilo ispravno. Ovaj simbol pokazuje mjesto gdje se HTML oznaka slike nalazi u odnosu na tekst. Ako ne vidite taj simbol, izaberite opciju menija View, Visual Aids, Invisible Elements i potvrdite istoimenu stavku. Simbol sidra se ne vidi u browseru.



Slika 39. Tok teksta oko slika

Lista Align sadrži nekoliko opcija za slike, među kojima su Top i Text Top. Ako va to zbunjuje, zapamtite da tekst može da omota grafiku samo s lijeve ili s desne

strane, pa su jedine opcije koje možete da izaberete Left i Right. Ostale opcije služe za smještanje samo jednog reda teksta do grafike.

Imajte u vidu da poravnanje slike mijenjate kad god izaberete opciju poravnanja različitu od Browser Default (kako je podrazumijevano u browseru). Pri tome se slika neznatno pomjera u odnosu na originalni položaj. Taj pomjeraj ne možete da kontrolirate, niti da ga izbjegnute. Veličina pomjeranja se mijenja u zavisnosti od browsera, ali obično iznosi samo nekoliko piksel, pa nije upadljiva. Međutim, ako pokušate da poravnate sliku u ćeliji tabele, i to malo pomjeranje može da smeta. U tom slučaju, upotrijebite druge metode za kontrolu položaja grafike.

Količina teksta koji se omotava oko grafike zavisi od veličine slova kojima je tekst ispisan u prozoru browsera, količine teksta i veličine prozora browsera. Kada promijenite veličinu prozora (bez obzira na to da li se radi o Dreamweaveru ili o browseru), promijenit će se i način na koji je tekst prelomljen oko grafike. Vodite računa i o tome da je slika koju vidite u Dreamweaveru samo otprilike jednaka onome što će se vidjeti u browseru.

Kad umetnete grafiku, sadržaj dokumenta koji dolazi poslije slike biva gurnut nadole. Kada sliku poravnate s lijevom marginom pomoću dugmeta Align Left, taj sadržaj se ne vraća uvijek na svoje mjesto, ukoliko koristite tabele. Zbog toga može izgledati da u dokumentu ima viška praznog prostora. Ako dođe do toga, samo pritisnite izvan posljednje tabele, tj. unutar tijela dokumenta.

5.11.5. Prilagođavanje prostora oko slike

Kada omotate tekst oko slike, vjerovatno ćete htjeti i da prilagodite prostor oko nje. Možete da dodate vertikalni prostor (eng. V space) i horizontalni prostor (eng. H space).

Ako na panou svojstava, upišete 10 u polje H Space, ovaj parametar znači da će s lijeve i s desne strane slike biti 10 piksela slobodnog prostora. Ne možete da dodate prostor na samo jednu stranu slike.

Ako upišete 6 u polje V Space, ovaj parametar znači da će s gornje i s donje strane slike biti 6 piksela slobodnog prostora. Ne možete da dodate prostor na samo jednu stranu slike.

5.11.6. Poravnavanje slike u odnosu na jedan red teksta

Često treba poravnati sliku u odnosu na jedan red teksta kraj nje. Relativni položaj slike prema tekstu možete da promijenite pomoću opcija poravnanja.

Absolute Middle poravnava sredinu reda teksta sa sredinom slike. Ostale opcije su:

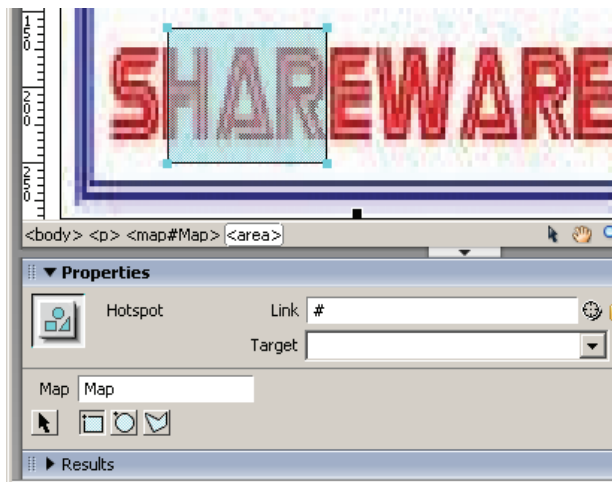
- Baseline: Poravnava dno slike sa osnovnom linijom teksta. Ova opcija je obično podrazumijevana u browserima.

- Top: Poravnava sliku s vrhom najvišeg objekta u redu. Taj objekat može biti tekst ili veća slika.
- Middle: Poravnava osnovnu liniju teksta sa sredinom slike.
- Bottom: Poravnava isto kao opcija Baseline.
- Text Top: Radi onako kako biste očekivali da radi opcija Top: poravnava sliku s vrhom najvišeg teksta u redu. (Najčešće daje iste rezultate kao opcija Top, ali ne uvijek).
- Absolute Middle: Poravnava sredinu slike sa sredinom reda teksta ili sa najvećim objektom u redu.
- Absolute Bottom: Poravnava dno slike s najnižom tačkom reda teksta.

Ove opcije možete da upotrijebite i za poravnanje slika s drugim slikama; niste ograničeni na korištenje teksta.

5.11.7. Izrada mapiranih slika

Vidjeli ste kako se lako pravi grafička hiperveza s Web stranom. Korisnik pritisne sliku kako bi skočio na povezanu stranu. Sliku možete da podijelite i na nekoliko hiperveza pomoću mapirane slike (eng. image map), što je naziv za sliku s više aktivnih područja. Aktivna područja ne moraju biti pravougaona, već mogu imati i druge oblike.



Slika 40. Kreiranje mapirane slike

Alatom Rectangular Hotspot ispod imena mape na panou svojstava može se nacrtati aktivno područje na slici. Oko imena na tako označenom dijelu slike pojaviti će se providno plavozeleno područje s ručicama, a pano Properties će prikazati svojstva tih aktivnih područja. Dreamweaver automatski upisuje znak za

broj (#) u polje Link panoa svojstava. Ne brišite ovaj znak jer on rezerviše mjesto za hipervezu i ukazuje na to da se područje može pritisnuti.

Kada napravite jednu ili više mapiranih slika, pojavit će se ikonica mape na dnu dokumenta (najčešće). Ova ikonica liči na ikonicu sidra, a vidi se samo u Dreamweaveru, ne i u browseru. Vidljivost ovakvih objekata možete da uključujete i isključujete pomoću opcije menija View, Visual Aids.

Aktivna područja koja napravite lako je preurediti: možete im mijenjati veličinu, premještati ih i brisati kad god hoćete. Da biste premjestili aktivno područje, dovedite pokazivač miša u područje i prevucite ga. Ako hoćete da uklonite aktivno područje, izaberite ga i pritisnite Backspace.

Na isti način prave se i kružna aktivna područja na slikama. Ako se dva i više aktivnih područja preklapaju, prvo koje ste napravili ima prvenstvo u odnosu na sva kasnija.

Ako znak # u polju Link zamijenite URL adresom i izaberete opciju "_blank" u padajućoj listi Target, ta se hiperveza otvara u novom prozoru browsera.

Ako aktivno područje treba da ima složeniji oblik, možete ga dobiti pomoću alatke za poligonalna aktivna područja (Polygon Hotspot), kojom treba da pritisnete tjemena područja. Ako je alatka Polygon izabrana, svaki pritisak miša pravi jedno tjeme aktivnog područja. Linija povezuje svako slijedeće tjeme s prethodnim. Kako budete pritiskivali sliku i pravili tjemena, vidjet ćete kako se stvara providno aktivno područje definisano tim tjemena. Nastavite da pritiskate i pravite tjemena sve dok ne dobijete oblik koji želite. Ne morate da "zatvorite" oblik ponovnim pritiskanjem početnog tjemena. Što više tjemena aktivnog područja zadate, to će u dokumentu biti potreban duži kod da bi ih opisao.

Kada završite rad na mapiranoj slici, pritisnite izvan aktivnog područja, dakle neki drugi dio slike. Pritiskom izvan mapirane slike, na panou svojstava se prikazuju svojstva slike.

Ako hoćete da dobijete kvadratno aktivno područje pomoću alatke za pravougaona aktivna područja (Rectangular Hotspot), držite taster Shift dok crtate područje mišem.

Ako mapiranu sliku kopirate i prenesete u drugi dokument, Dreamweaver će u kopiji zadržati sve hiperveze i aktivna područja.

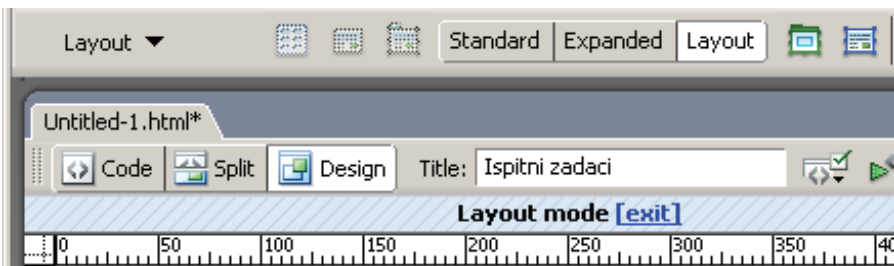
5.12. Crtanje tabela u prikazu layout

Dreamweaver MX pruža dvije mogućnosti za pravljenje tabela: prikaze Layout i Standard. Prikaz Layout je sličan programu za pravljenje rasporeda elemenata na strani, gdje se crtaju pravougaonici u koje zatim ubacujete tekst ili slike. Možete da mijenjate veličinu tih pravougaonika i da ih razmještate po strani kako hoćete.

Na traci Insert izaberite karticu Layout i pritisnite opciju Layout View. Prešli ste u prikaz Layout, u kome se elementi lako smještaju na stranu. Prikazat će se obavještenje pod naslovom "Getting Started in Layout View", koje ukratko opisuje glavne alatke: za pravljenje nacrt ćelije i za pravljenje nacrt tabele. Kada pročitate ovo objašnjenje, pritisnite OK.

U pomenutom objašnjenju postoji polje "Don't show me this message again" (Više ne prikazuj ovu poruku). Ako ne potvrdite ovo polje, isto objašnjenje će se ponovo prikazati kada restartujete Dreamweaver i pređete u prikaz Layout.

Neposredno ispod naslovne trake dokumenta, pojavit će se svijetlosiva traka s natpisom "Layout View". Mada izgleda kao da je sastavni dio dokumenta, ova traka se ne vidi u browserima i koristi se u Dreamweaveru samo da bi vas obavijestila u kom prikazu radite.

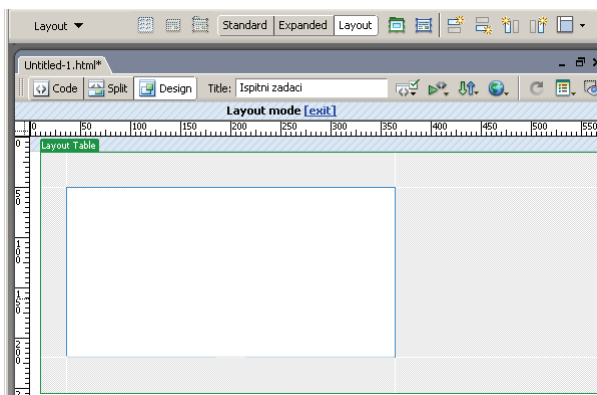


Slika 41. Layout način rada

SAVJET: Drugi način da pređete u prikaz Layout daje opcija menija View, Table Mode, Layout Mode. Možete da upotrijebite i prečicu s tastature Ctrl+F6.

Pritisnite ikonicu Draw Layout Cell u području Layout Insert. Kada dovedete pokazivač u prozor dokumenta, on će se promijeniti u znak plus (+). Alatka Draw Layout Cell služi za crtanje ćelija, odnosno pravougaonika koje zovemo ćelije, bilo gdje na strani. Sadržaj tabele je podijeljen na njene ćelije, jer sve tabele imaju jednu ili više ćelija u presjeku kolona i redova. U prikazu Layout ne morate da brinete o broju i rasporedu redova i kolona kada crtate tabelu, jer se o njima automatski brine Dreamweaver kada naznačite mjesto ćelija na strani.

Dovedite pokazivač na sredinu strane, zatim pritisnite miša i vucite ga dok kontura ćelije ne bude odgovarajuće veličine. Automatski će se iscrtati struktura tabele, koja sadrži ćeliju koju ste sami nacrtali. Nacrt tabele je širok kao prozor dokumenta, što vas ne sprječava da veličinu te tabele mijenjate kako god hoćete. Nacrt ćelije je iscrtan plavom bojom da bi se razlikovao od nacrt tabele, koji je zelen. Puna plava linija naznačava da je tačka umetanja u ćeliji, dok isprekidana plava linija naznačava da tačka umetanja nije u ćeliji. Svi ostali dijelovi nove tabele, osim ćelije, iscrtani su sivom bojom. Tanke bijele linije naznačuju redove i kolone tabele koju je Dreamweaver nacrtao oko nacrt vaših ćelija. Kada dovedete pokazivač miša iznad ivice nacrt ćelije, on postaje crven, i ukazuje na ćeliju iznad koje se nalazite.



Slika 42. Crtanje tabele pomoću alata "Draw Layout Cell"

Na vrhu nacrt tabele podrazumijevano se prikazuje traka koja olakšava prepoznavanje tabele; traka sadrži mjeru širine kolona tabele i njihove podmenije. Zbog ove trake, tabela je pomjerena od vrha strane, ali to se u browserima ne vidi.

5.12.1. Mijenjanje nacrt tabele

Dok budete radili u prikazu Layout, dodavat ćete nove ćelije, mijenjati im veličinu i premještati. Nacrt ćelije ne smije da se preklopi s drugim ćelijama, niti može da se premjesti izvan nacrt tabele.

Dovedite pokazivač iznad plave ivice tabele i pritisnite tipku miša kada se pokazivač oboji u crveno. Ivica ćelije će poplaviti; pojaviće se i ručice koje možete da povučete i tako promijenite veličinu ćelije.

Da biste prikazali ručice za promjenu veličine ćelije, pritisnite je držeći taster Ctrl.

Tasteri sa strelicama pomjeraju ćeliju za po jedan piksel kada ih pritisnete. Ako pri tom budete držali taster Shift, pomjerat će se po 10 piksela.

Kada crtate ćeliju na strani, prikazuju se bijele vođice koje olakšavaju crtanje drugih ćelija koje treba da budu poravnate s prvom ćelijom. Upotrijebite horizontalne vođice za poravnanje vrhova ćelija.

Ako hoćete da nacrtate više ćelija, a da pri tom ne morate više puta da pritiskate ikonicu Draw Layout Cell, držite taster Ctrl dok budete crtali prvu ćeliju. Možete da nastavite crtanje novih ćelija sve dok ne otpustite taj taster.

5.12.2. Formatiranje ćelija

Nekoliko svojstava svake ćelije možete da mijenjate. Možete da obojite pozadinu svake ćelije zasebno ili cijele tabele. Isto tako, možete da kontrolišete horizontalno i vertikalno poravnanje objekata unutar svake ćelije. Sadržaj ćelije se u HTML-u podrazumijevano horizontalno poravnava s lijevom ivicom ćelije, a vertikalno sa

sredinom ćelije. Kad sami napravite nacrt ćelije, Dreamweaver poravnava njen sadržaj s gornjom ivicom ćelije, ali to se može lako promijeniti.

U polje Bg na panou svojstava možete upisati oznaku boje pozadine ili izabrati boju iz ponuđene palete (ili pipete). Ćelija se bira pritiskanjem njene ivice (tako da vidite njene ručice za biranje), a ne postavljanjem kursora u nju.

Ako hoćete da promijenite boju pozadine cijele tabele, pritisnite njenu zelenu ivicu kako biste je izabrali; možete da pritisnete i bilo koju sivu površinu unutar tabele. Zatim pritisnite polje Bg i izaberite boju iz palete.

Ako u padajućoj listi Horz na panou svojstava izaberete opciju Center, time ste promijenili poravnanje teksta u ćelijama, koji je sada centriran u odnosu na lijevu i desnu ivicu ćelije. Opcija Middle tekst centrira po vertikali ćelija.

5.12.3. Širina tabele i slike koje održavaju razmak

U prikazu Layout možete da kontrolišete širinu tabele na dva načina: korištenjem podrazumijevane širine ili opcije Autostretch, koja prouzrokuje da ćelije automatski mijenjaju širinu u zavisnosti od širine prozora browsera. U ovoj vježbi primijenit ćete opciju Autostretch i naučiti da koristite slike koje održavaju razmak (eng. spacer images).

Ukoliko određena slika za održavanje razmaka nije pridružena prezentaciji, pojavit će se okvir za dijalog Choose Spacer Image.

Ako se gornji okvir za dijalog prikaže, izaberite opciju Use an Existing Spacer Image File (Upotrijebi postojeću sliku za održavanje razmaka), pritisnite OK i pronađite sliku. Prikazani okvir za dijalog sadrži i opciju za pravljenje slike za održavanje razmaka. Ako radite na prezentaciji kojoj nije pridružena slika za održavanje razmaka, izaberite opciju Create a Spacer Image File i pritisnite OK, kako biste došli do direktorija u koji Dreamweaver treba da snimi sliku za održavanje razmaka. Najbolje mjesto za to je direktorij Images.

Kolona koja automatski prilagođava svoju širinu, u tabeli se prikazuje cik-cak linijom. Kada izaberete opciju automatskog prilagođavanja širine kolone Autostretch, Dreamweaver umeće slike za održavanje razmaka kako bi održao zadatu, fiksnu širinu kolona. Slika za održavanje razmaka određuje razmak ćelija u tabeli, ali se ne vidi u prozoru browsera.

Ako hoćete da uključite automatsko održavanje širine kolone, možete da otvorite i podmeni u zaglavlju kolone i da tamo izaberete opciju Make Column Autostretch.

Dok mijenjate širinu prozora browsera, mijenja se i širina ove kolone, odnosno svih njenih ćelija. Pomoću bijelih vođica na strani, utvrdite da li pojedina ćelija pripada koloni koju ste izabrali.

Ako odlučite da ne koristite slike za održavanje razmaka, kolone mijenjaju veličinu, pa čak i nestaju, ukoliko nemaju sadržaja. Slike za održavanje razmaka

možete i sami da umećete i uklanjate, a možete pustiti i Dreamweaver da ih automatski dodaje prilikom pravljenja Autostretch kolone. Ako hoćete sami da umećete i uklanjate slike za održavanje razmaka, izaberite jednu od sljedećih opcija u podmeniju na vrhu kolone:

- Add Spacer Image: Umeće sliku za održavanje razmaka u kolonu. Tu sliku nećete vidjeti, ali kolona može malo da se pomjeri.
- Remove Spacer Image: Uklanja sliku za održavanje razmaka i kolona može malo da se pomjeri.
- Remove All Spacer Images: Sve ćelije mogu malo (ili mnogo) da se pomjere, u zavisnosti od sadržaja. Kolone u kojima nema sadržaja mogu potpuno da nestanu.

Podmeni na vrhu kolone je kontekstni i mijenja se u zavisnosti od kolone koju ste izabrali. Nemaju sve kolone sve tri gornje opcije.

5.12.4. Izrada tabele u prikazu standard

Kada ste pravili tabelu u prethodnoj vježbi, Dreamweaver je uradio glavninu posla. Baš zato, u prikazu Layout donekle gubite kontrolu nad izgledom tabele. U standardnom prikazu morate sami sve da uradite. Prednost standardnog prikaza je preciznost i mogućnost detaljnije kontrole izgleda tabele. Osim toga neke važne Dreamweaverove mogućnosti, kao što je podjela na slojeve, uopšte ne postoje u prikazu Layout, nego morate da koristite prikaz Standard.

Premda je u prikazu Layout lako crtati ćelije i tabele, često treba pregledati stranu u standardnom prikazu, jer se tek tu vidi HTML struktura tabela. U ovom prikazu možete sami da pravite tabele i da pregledate tabele koje je Dreamweaver napravio dok ste crtali u prikazu Layout.

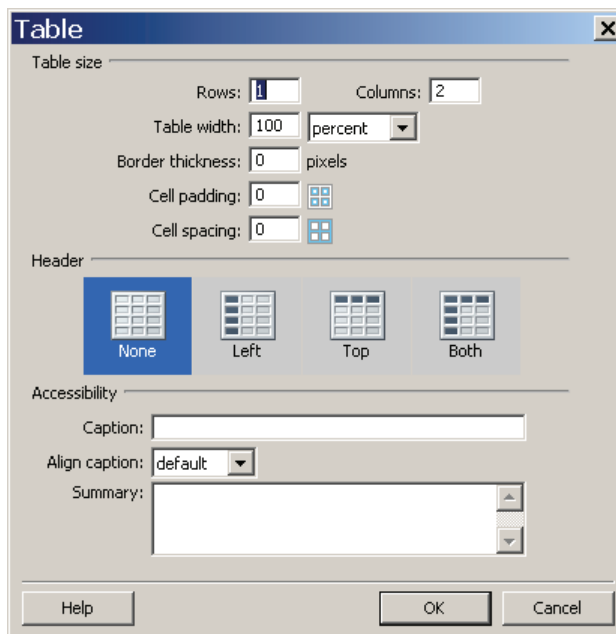
Ako su informacije koje hoćete da prikazete podijeljene u redove i kolone, lakše je upotrijebiti standardnu tabelu nego nacrtati novu. Često ćete u standardnom prikazu imati veće mogućnosti da kontrolišete izgled tabele.

Pritisnite opciju Standard View na kartici Layout trake Insert. Kada umetnete tabelu u standardnom prikazu, njene ćelije i ivice se vide odmah, a ikonice za crtanje ćelija i tabela su nedostupne. Za prelazak u standardni prikaz možete da upotrijebite i opciju menija View, Table Mode, Standard Mode. Treći način za prečice s tastature; Shift+Ctrl+F6.

Za umetanje tabele može da posluži dugme Table na kartici Tables trake Insert. Otvorit će se okvir za dijalog Insert Table. Okvir za dijalog Insert Table sadrži polja za davanje slijedećih parametara:

- Rows: Broj redova tabele. Podrazumijevana vrijednost u Dreamweaveru je 3.
- Columns: Broj kolona tabele. Podrazumijevana vrijednost je 3.

- **Width:** Širina tabele u pikselima ili u procentima širine prozora browsera. Za precizno raspoređivanje teksta i slika bolje su dimenzije zadate u pikselima. Dimenzije u procentima su dobre kada je odnos širine i visine kolona važniji od njihove apsolutne širine. Podrazumijevana vrijednost širine u Dreamweaveru je 75%.
- **Border:** Debljina ivice tabele. Podrazumijevana vrijednost u Dreamweaveru je veličina fonta 1 piksel
- **Cell Padding:** Razmak između sadržaja ćelija tabele i njenih ivica. Ako ne zadate ovaj parametar, njegova podrazumijevana vrijednost je 1 piksel. Ako uopšte nećete razmak između sadržaja ćelija tabele i njenih ivica, upišite 0 u ovo polje. Prazno polje znači da je izabrana Dreamweaverova podrazumijevana vrijednost.
- **Cell Spacing:** Razmak između ćelija tabele, ne računajući ivice. Ako ne zadate ovaj parametar, njegova podrazumijevana vrijednost je 1 piksel. Ako uopšte nećete razmak između ćelija tabele, upišite 0 u ovo polje. Prazno polje znači da je izabrana Dreamweaverova podrazumijevana vrijednost.



Slika 43. Insert Table

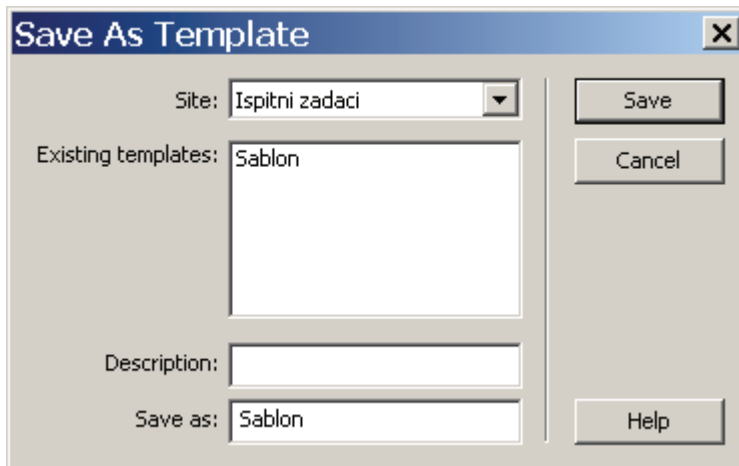
Za kretanje između ćelija možete koristiti tastere sa strelicama ili taster Tab, odnosno Shift+Tab. Tab je najbrža metoda za prelazak u slijedeću ćeliju udesno, odnosno u prvu ćeliju slijeva u slijedećem redu, ukoliko se nalazite u posljednjoj ćeliji na kraju reda. Ako pređete u ćeliju koja već ima sadržaj, on će biti izabran kada koristite Tab.

5.13. Pravljenje šablona (Template-a)

Šablon je posebna vrsta datoteke u Dreamweaveru koja se koristi kada postoji dosta stranica u kojima se nalazi dio sitog koda (teksta, slika, grafike, tabela,...). Unutar šablona se definišu područja koja su zajednička i područja u koje se unosi sadržaj koji se nalazi samo u toj stranici. Svaka promjena na šablonu može se automatski reflektovati na izmjene svih stranica za koje je taj šablon vezan, odnosno od kojeg su napravljene.

Šablon definiše izgled i raspored strana koje ćete napraviti pomoću njega. Šablon možete da pravite od nule, a možete da uzmete i postojeću HTML stranu i da je snimate kao šablon, naredbom File, Save As Template. U dijalogu "Save As Template" birate prezentaciju u koju ćete snimiti šablon.

Dreamweaver šablonu automatski daje ime datoteke iz koje je nastao. Ako hoćete da promijenite ime šablona, upišite ga u polje Save As. Ono ionako služi samo vama: Šablonima dajte opisna imena. Strana je tako snimljena kao šablon, pa možete da je koristite za pravljenje drugih strana.



Slika 44. Snimanje šablona

Šablon se snima u direktorij Templates sa nastavkom imena, .dwt. Ako takav direktorij ne postoji, Dreamweaver će ga automatski napraviti.

5.13.1. Ubacivanje oblasti koje se mogu mijenjati u šablon

Drugi korak u pravljenju šablona jeste definisanje oblasti na strani koje se mogu mijenjati u dokumentima napravljenim na osnovu tog šablona.

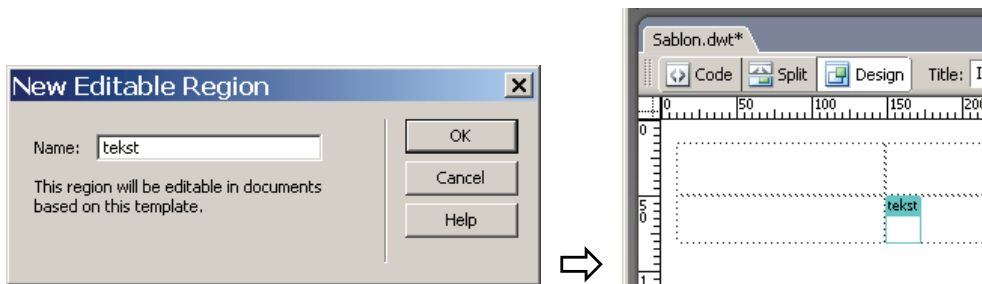
Po pravilu, sve oblasti šablona su nepromjenljive. Ako hoćete da mijenjate informacije na stranama napravljenim pomoću šablona, morate da definišete oblasti na njima koje se mogu mijenjati. Ove oblasti na mnogim Web lokacijama sadrže tekst. Svi dijelovi šablona koji nisu izričito definisani kao promjenljivi, daju

nepromjenljive oblasti strana zasnovanih na tom šablonu. Prilikom rada na šablonu možete da mijenjate i oblasti koje će biti promjenljive i one koje to neće biti, dok na strani napravljenoj pomoću šablona možete da mijenjate samo oblasti definisane kao promjenljive.

Za vježbu kreirajte prazan HTML dokument, a zatim u njemu nacrtajte tabelu sa 2 reda i 2 kolone. Snimite datoteku kao šablon, pod imenom Sablon.dwt.

U šablonu Sablon.dwt izaberite drugu ćeliju u drugom redu tabele pomoću birača oznaka na dnu prozora dokumenta: pritisnite oznaku <td>. Ovaj odjeljak strane mora biti promjenljiv, kako biste mogli da mijenjate njegovu sadržaj na stranama koje ćete napraviti pomoću ovog šablona.

Pritisnite ikonicu Editable Region na kartici Templates trake Insert. Otvorit će se okvir za dijalog New Editable Region (nova promjenljiva oblast). Upišite riječ "tekst" kao ime promjenljive oblasti šablona i pritisnite OK.



Slika 45. Pravljenje promjenljive oblasti u šablonu

Imena oblasti ne smiju da sadrže specijalne znakove (znakove navoda, zagrade itd.). Svaka oblast jednog šablona mora imati drugačije ime.

Pošto ste izabrali ćeliju tabele kada ste definisali promjenljivu oblast, cijela ćelija je istaknuta plavim okvirom na čijem vrhu jezičak prikazuje ime promjenljive oblasti. Ako se ne vide imena i konture izabranih oblasti, izaberite opciju menija View, Visual Aids, Invisible Elements.

Ne možete definisati više ćelija tabele kao jednu promjenljivu oblast. Ako hoćete da više ćelija tabele bude promjenljivo, podesite da cijela tabela bude promjenljiva ili je podijelite na nekoliko promjenljivih područja. Ako izaberete nekoliko ćelija tabele i definišete ih kao promjenljive, cijela tabela će postati promjenljiva oblast.

Sve što je u ćeliji izvan izabranog područja oivičenog plavom konturom, biće nepromjenljivo u dokumentima napravljenim pomoću ovog šablona.

Imena svih promjenljivih oblasti koje ste napravili prikazuju se na dnu menija Modify, Templates. Ako izaberete promjenljivu oblast, do njenog imena u spisku pojaviti će se znak za potvrdu; isto važi i kada je kursor u toj oblasti ili je izabran neki element unutar nje. Sve što treba da se mijenja mora biti u promjenljivoj oblasti, uključujući i hiperveze.

Ako budete pravili hiperveze u originalnoj datoteci šablona, za pronalaženje hiperveze upotrijebite ikonicu direktorija ili alatku za pravljenje hiperveza Point-to-File. Obje se nalaze na panou svojstava. Hipervezu ne smijete upisivati neposredno u odgovarajuće polje panoa svojstava, jer hiperveze u šablonu neće raditi. Kako se šabloni čuvaju u direktoriju Templates, njihove putanje su drugačije od onoga što bi se očekivalo. Dreamweaver automatski formira ispravne putanje kada koristite jednu od dvije predložene metode.

Više alati i mogućnosti predviđenih za pravljenje i modifikovanje šablona dostupne su samo u prikazu Design u kome ste radili: Neki kontrolni objekti za šablone nisu dostupni u prikazu Code. Dakle, ako radite u prikazu Code i ne možete da obavite određene operacije sa šablonom, pređite u prikaz Design.

5.13.2. Uklanjanje promjenljivih oblasti

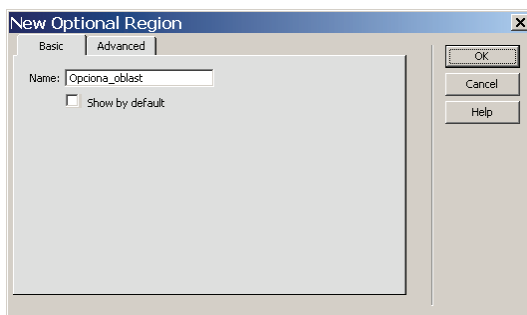
Mijenjanje oblasti šablona definisanih kao "promjenljive" možete naknadno da onemogućite. U takvim "zaključanim" oblastima elementi se ne mogu mijenjati neposredno na strani napravljenog od šablona, nego samo u originalnom šablonu.

Naredbom Modify, Templates, Remove Template Markup nastat će kontura koja označava ćeliju kao promjenljivu; sada je taj dio šablona zaključan i ne može se mijenjati u datotekama napravljenim pomoću ovog šablona. Da bi se ta naredba mogla koristiti, treba prvo označiti oblast koju treba ukloniti.

Ako uklonite promjenljivu oblast iz šablona pomoću koga su već pravljene strane, one će se izmijeniti kada ih ažurirate poslije snimanja šablona. Time će biti obrisane sve izmjene nekadašnje promjenljive oblasti na tim stranama, zato što se prethodno promjenljiva oblast mijenja u skladu s novim stanjem tog dijela šablona.

5.13.3. Izrada opcionih oblasti šablona

Možete da birate da li će se sadržaj neke oblasti prikazivati na stranama napravljenim pomoću šablona ili neće, u zavisnosti od vrijednosti izraza koji se izračunava ili od parametra koji zadajete.



Slika 46. Opciona oblast šablona

Naredbom Insert, Template Objects, Optional Region kreirate opcionu oblast.

Ako namjeravate koristiti opcionu oblast šablona na većini strana prezentacije koje ćete napraviti pomoću njega, ostavite ovo polje potvrđeno. U ovom primjeru, međutim, oblast će se prikazati na samo jednoj strani, pa će prilikom izrade drugih strana biti lakše ako ova oblast bude podrazumijevano skrivena.

Ako ste na stani već napravili neku opcionu oblast, pomoću kartice Advanced možete da je povežete s novom opcionom oblašću koju upravo pravite. Na istoj kartici definišete i izraze (Template Expressions) koji određuju kada se oblast prikazuje.

5.13.4. Pravljenje oblasti koje se ponavljaju

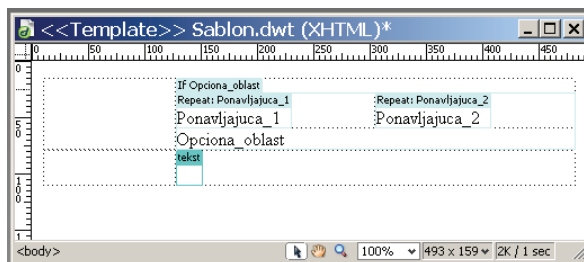
Oblast koja se ponavlja (eng. repeating region) jeste područje strane koje se ponavlja više puta. Oblasti koje se ponavljaju naročito su pogodne za objekte koji se ponavljaju promjenljiv broj puta na stranama napravljenim pomoću šablona. Ove oblasti omogućavaju precizno kontrolisanje izgleda tabela s više stavki.

Naredbom Insert, Template Objects, Repeating Region kreirate opcionu oblast.



Slika 47. Ponavljajuća oblast šablona

Kreirajte dokument koji liči na primjer prikazan na sljedećoj slici; Unutar tabele 2 x 2 kreirajte jednu opcionu oblast (u prvom redu, sa imenom "Opciona_oblast") i jednu običnu oblast (u drugom redu, sa imenom "tekst"). Unutar opcione oblasti ubacite još jednu tabelu sa 2 kolone i u svaku njenu ćeliju ubacite po jednu ponavljajuću oblast (sa imenima "Ponavljajuca_1" i "Ponavljajuca_2"). Sadržaj tako kreiranih oblasti se neće moći mijenjati u stranici kreiranoj pomoću ovog šablona, jer se ne nalaze unutar promjenljive oblasti.



Slika 48. Primjer šablona sa oblastima

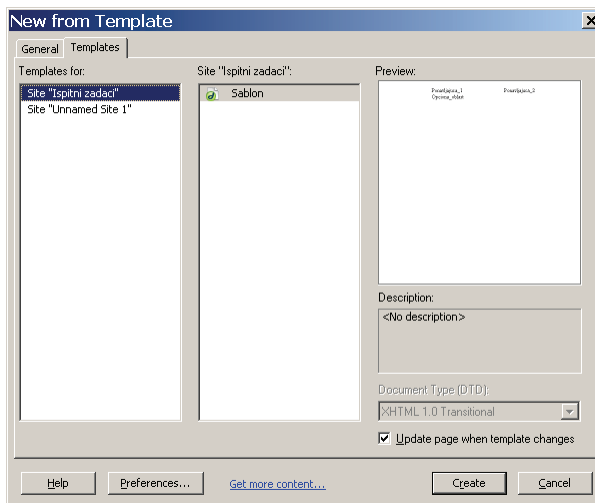
Snimite i zatvorite datoteku Sablon.dwt.

5.13.5. Pravljenje strana pomoću šablona

Sljedeći korak u izradi prezentacije pomoću šablona jeste pravljenje strana na osnovu originalnog šablona.

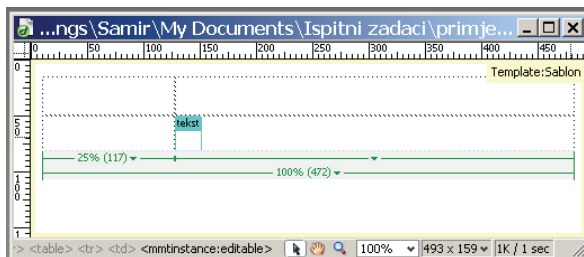
U prikazanom primjeru napraviti ćete nove strane na osnovu šablona profile koji ste napravili u prethodnom primjeru. Te strane će naslijediti sadržaj originalnog šablona; moći ćete da mijenjate samo one dijelove koje ste definisali kao promjenljive oblasti.

Izaberite File, New i pritisnite karticu Templates u okviru za dijalog New Document. Na kartici Templates prikazuje se spisak definisanih prezentacija i spisak svih šablona pridruženih izabranoj prezentaciji.



Slika 49. Kreiranje nove stranice pomoću šablona

Izaberite profile u spisku šablona prezentacije "Ispitni zadaci", ostavite polje "Update Page when Template Changes" (ažuriraj strane kada se izmijeni šablon) potvrđeno i pritisnite dugme Create. Napravili ste novu stranu na osnovu šablona "Šablon". Premda novi dokument prikazuje svu naslijeđenu sadržaj šablona, ipak ga treba snimiti. Snimite datoteku pod imenom primjer.html



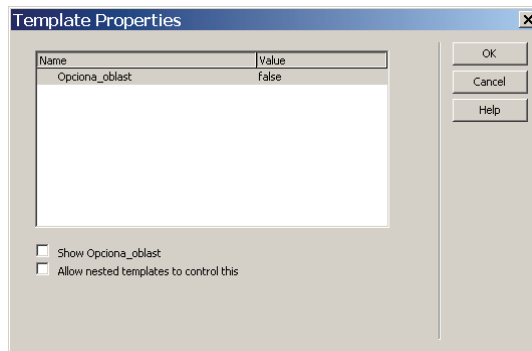
Slika 50. Nova html stranica kreirana pomoću šablona

Na novoj strani vidite boju za isticanje zakljućanih oblasti (podrazumijevana boja je svijetložuta). Ime šablona se prikazuje na jeziku iste boje, u gornjem desnom uglu prozora dokumenta. Kada dovedete pokazivać miša iznad zaključane oblasti (ili pokušate da je pritisnete), on će se pretvoriti u precrtan krug. To pokazuje da se te oblasti ne mogu mijenjati.

Na isti način kreirajte još jednu html stranicu.

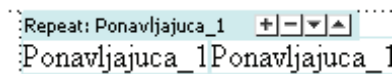
5.13.6. Kontrolisanje pojavljivanja opcionih oblasti

Naredbom Modify, Template Properties može se definisati da li će se opcione oblasti iz šablona u toj stranici prikazati a koje ne.



Slika 51. Upravljanje opcionim oblastima

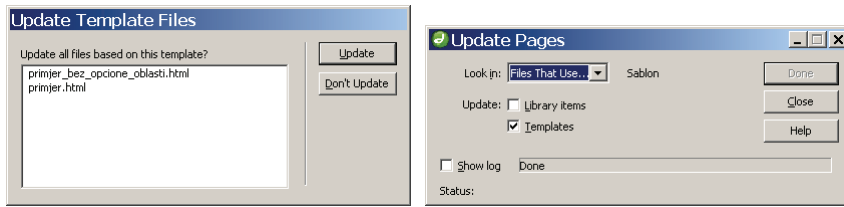
Kada uključite opcionu oblast, ista će se pojaviti u stranici. Ponavljajuće oblasti koje se nalaze unutar nje ćete moći dodavati pomoću tipke "+" koja se pojavi pored imena oblasti. Nećete ih moći brisati, jer se ni opciona ni ponavljajuće oblasti ne nalaze unutar promjenljive oblasti. Iz toga vidite da prilikom kreiranja oblasti u šablonu treba poštovati hijerarhiju; sve oblasti koje se nalaze u šablonu moraju biti unutar promjenljive oblasti.



Slika 52. Tipke za upravljanje ponavljajućim oblastima

5.13.7. Izmjena stranica pomoću šablona

Ako je potrebno izvršiti izmjenu svih stranica koje su vezane za neki šablon, potrebno je otvoriti datoteku sa šablonom (sa ekstenzijom .dwt), izvršiti izmjene u njoj, a zatim je snimiti na disk. Prilikom snimanja, pojavi se dijalog sa pitanjem da li želite da se te promjene reflektuju na sve stranice koje su vezane za taj šablon (Update) ili da se snimi samo datoteka šablona, bez promjena na stranicama koje su za njega vezane ((Don't Update).



Slika 53. Ažuriranje stranica pomoću šablona

Naredbom Modify, Templates, Update Current Page se stranica može naknadno ažurirati tako da prihvati izmjene koje su urađene na šablonu.

Naredbom Modify, Templates, Detach from Template se stranica odvaja od šablona, tako da se njen sadržaj ne mijenja, a samo se raskida njena veza sa šablonom (prilikom ažuriranja se promjene na šablonu neće reflektovati na sadržaj stranice).

6. Izrada obrasca

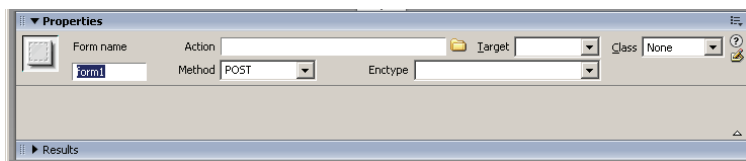
Web stranice često prikupljaju podatke od korisnika. Obrasci (eng. Forms) omogućuju da korisnici pomoću njihovog browsera unose neke podatke i šalju ih na e-mail, pohranjuju na serveru ili prosljeđuju nekoj aplikaciji. Obrasci u sebi sadrže različite vrste polja za unos ili izbor ponuđenih podataka.

Prije dodavanja pojedinih polja, morate da stavite obrazac na stranicu. On će sadržati polja u koja će posjetioci upisivati podatke i zadati šta da se radi s podacima.

Obrazac možete da stavite na stranicu naredbom Insert, Form, Form.

Crvene isprekidane linije označavaju površinu obrasca u prozoru dokumenta; ista površina je definisana unutar oznaka `<form>` i `</form>` u kodu. Crvene linije su nevidljivi elementi i prikazuju se samo u Dreamweaveru; površinu obrasca u browseru ništa ne označava. Te linije ne možete prevlačiti. Veličina obrasca zavisi od onoga što umetnete u njega, a on će se produžiti koliko treba da bi primio sav sadržaj koji mu dodate.

Izaberite obrazac pritiskom na crvenu isprekidanu liniju. Na panou svojstava prikazuju se svojstva obrasca. Na njemu je nekoliko opcija.



Slika 54. Pano svojstava obrasca

U polje Form Name upišite ime obrasca, ukoliko namjeravate da ga kontrolišete pomoću aplikacije za izvršavanje skriptova, poput JavaScripta. Dreamweaver obrascima automatski dodjeljuje opšta imena: form1, form2 itd.

Ako nemate na raspolaganju CGI skripte dok radite ovu lekciju, informacije koje slijede tumačite samo kao objašnjenje. Polje Action ostavite prazno, a Method ne mijenjate. Action saopštava browseru šta da uradi s podacima iz obrasca. Zadaje putanju ili URL adresu lokacije i ime aplikacije (obično CGI skripta ili dinamičke strane) koja će obrađivati podatke kada posjetilac pritisne dugme Submit. CGI skripte se nalaze na Web serveru koji obrađuje podatke koje obrazac pošalje.

Method definiše metodu kojom se podaci iz obrasca šalju: GET, POST ili Default. Podaci koje šalje obrazac čine kontinuiran tekstualni niz, koji sačinjavaju informacije koje je upisao posjetilac. Metoda GET dodaje sadržaj obrasca URL adresi zadatoj u polju Action; te informacije su stoga vidljive na adresnoj traci browsera. GET nije bezbjedna metoda prenosa podataka, pa se ne smije koristiti za slanje povjerljivih podataka kao što su brojevi kreditnih kartica i slično. Kako browseri i serveri često nameću ograničenja dužine URL adrese, Get može da šalje malu količinu podataka. Ograničenja su različita, pa metoda Get nije dobra ni za

slanje obrazaca u koje posjetilac može da unese mnogo podataka, jer će obrasci izgubiti sve informacije koje pređu ograničene veličine ili dužine. Metoda POST, s druge strane, može da šalje mnogo više podataka, pouzdanija je i bezbjednija. POST koristi HTTP zahtjev za slanje podataka iz obrasca u tijelu poruke. Opcija Default koristi podrazumijevanu metodu browsera, što je obično GET.

Unutar obrasca se obično kreira tabela, da bi se lakše kontrolisali položaji pojedinih polja obrasca; svako polje se onda postavlja u posebnu ćeliju tabele.

Tabela poboljšava izgled obrasca i olakšava poravnanje teksta i slika sa poljima obrasca, kako bi im se dodao natpis. Tabelu možete da umetnete u obrazac i obrnuto, ali tabela mora biti potpuno unutar obrasca ili obrazac unutar tabele (ne smiju se preklapati).

Vrijednost "0" u poljima Cell Padding i Cell Spacing nije isto što i nepostojanje vrijednosti u tim poljima. Ako nikakve vrijednosti nema, podrazumijeva se 1 piksel, iako se broj 1 ne vidi u tim poljima.

Definisanjem širine kolona prije umetanja teksta ili drugih objekata, lakše ćete održati veličinu ćelija dok u njih umećete potrebne objekte. Također je zgodno podesiti i poravnanje u pojedinim ćelijama tabele, kako bi se polja poravnala po lijevom ili desnom rubu.

Obično ćelije s lijeve strane sadrže natpise za polja obrasca, koje ćete upisati kada ih napravite. Ćelije s desne strane sadržat će sama polja.

Na istoj strani možete imati više obrazaca. Međutim, u HTML-u nije moguće ugnijezditi jedan obrazac unutar drugog. Zbog ovog ograničenja, Dreamweaver sprečava slučajno ugnježđivanje jednog obrasca u drugom. Ta opcija neće biti siva i nedostupna, ali nećete moći da je primijenite. Ako ručno umetnete HTML oznake jednog obrasca unutar oznaka drugog, Dreamweaver će istaknuti neispravne oznake da bi vam skrenuo pažnju na grešku.

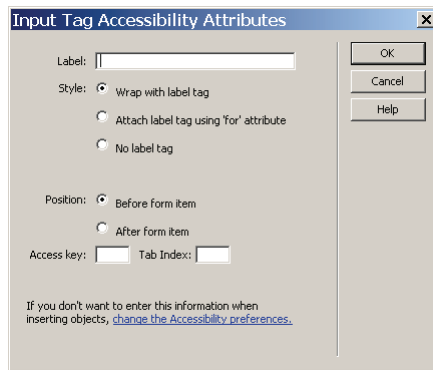
6.1. Dodavanje polja za jedan red teksta

Polja služe za prikupljanje informacija koje posjetilac upisuje u njih. Obično se u polja za jedan red teksta upisuje ime i prezime, dio poštanske adrese ili adresa elektronske pošte.

Sva polja i dugmad obrasca moraju biti smješteni unutar crvenih isprekidanih linija; u protivnom, neće biti dio obrasca. Ako pokušate da umetnete polja obrasca izvan crvenih linija, Dreamweaver će prikazati upozorenje, pitati da li želite da dodate HTML oznaku obrasca i ponuditi odgovore Yes i No. Ako izaberete No, polje ili dugmad neće raditi kao dio obrasca.

Veoma je važno navesti natpise uz sve objekte na obrascu (tekstualna polja, polja za potvrdu itd.) da bi posjetilac znao šta da upiše u njih. Bez natpisa, obrazac samo zbunjuje.

Izaberite naredbu Insert, Form Objects, Text Field.

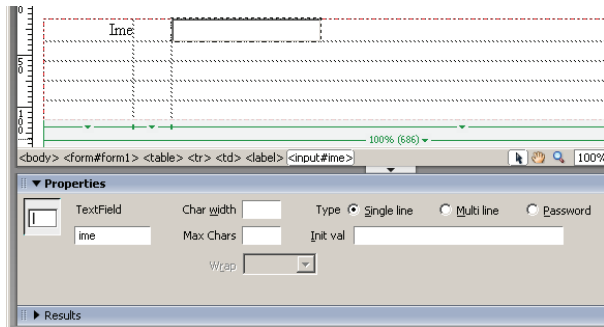


Slika 55. Dodavanje polja za jedan red teksta

Polje za jedan red teksta umetnuto je na obrazac. Na panou svojstava prikazuju se svojstva polja kada ga umetnete ili pritisnete u prozoru dokumenta. Podrazumijevana opcija vrste polja je Single line (za jedan red teksta).

U ovom primjeru, ostavit ćemo polje Init Val prazno. Možete da zadate početnu vrijednost (eng. initial value) teksta koja se prikazuje u polju kada se strana učita. Premda posjetilac može da promijeni ovaj tekst, koristite ovu mogućnost oprezno. Posjetioci koji hoće da na brzinu prođu kroz obrazac, mogu slučajno da preskoče polje koje već ima neki tekst, jer misle da su ga sami upisali. Početna vrijednost teksta daje posjetiocima primjer kakve informacije se od njih traže.

Izaberite polje koje ste upravo umetnuli i zamijenite textfield riječju "ime" u polju Name na panou svojstava.



Slika 56. Osobine polja u obrascu

Kada se obrazac pošalje, ime tekstualnog polja opisuje informacije koje su unijete u njega. U ovom slučaju, "ime" označava da je u polje upisano ime posjetioca. Sva polja obrasca moraju imati imena. U imenima ne smije biti razmaka niti specijalnih znakova, a valja zapamtiti da se u njima razlikuju mala i velika slova. Dreamweaver poljima za tekst automatski dodjeljuje imena textfield1, textfield2 itd.

Važno je da svim poljima date kratka, opisna imena. Pretpostavimo da na strani imate dva polja za tekst, svako sa odgovarajućim natpisom, u koje posjetilac treba da upiše broj telefona kod kuće i na poslu. Ako im ostavite imena `textfield1` i `textfield2`, po njima nećete moći da zaključite koji je broj kućni, a koji na poslu. Zabunu ćete izbjeći, ako im date imena kao što su npr. `brojkuca` i `brojnaposlu`.

Upišite 40 u polje `Char Width` na panou svojstava. Širina tekstualnog polja će se povećati tako da u njega stane približno 40 znakova. Početna širina polja je otprilike 24, iako je polje prazno. Stvarna veličina polja u browseru mijenja se u zavisnosti od veličine podrazumijevanog fonta u browseru posjetioca. I visina polja u browseru mijenja se u zavisnosti od veličine podrazumijevanog fonta u browseru posjetioca.

Upišite 50 u polje `Max Chars` na panou svojstava. Parametar `Max Chars` ograničava ukupan broj znakova koje posjetilac može da upiše. Na početku je ovo polje prazno, što znači da broj znakova koje posjetilac može da upiše nije ograničen. Ako informacije šaljete bazi podataka u kojoj je broj znakova svakog polja ograničen po definiciji, i sami morate da stavite slično ograničenje.

Ako je vrijednost parametra `Max Chars` veća od vrijednosti parametra `Char Width`, posjetilac može da nastavi sa upisivanjem teksta koji će se automatski pomjerati u polju. Pomjeranje će prestati kada se dostigne vrijednost parametra `Max Char`.

Pažljivo zadajte vrijednost parametra `Max Chars` za polja koja prihvataju podatke kao što su adrese e-pošte i URL adrese. Posjetioci neće moći da upišu cijelu URL adresu ili drugu vrstu informacija, ako je ona duža od vrijednosti parametra `Max Chars`, jer više neće moći ništa da upisuju kada dostignu zadato ograničenje.

Obično tekstualno polje prikazuje informacije u browseru dok ih posjetilac upisuje. Tekstualno polje za lozinku liči na ostala polja za tekst, ali se tekst upisan u njega prikrija na ekranu kružićima ili zvjezdicama. Tako se tekst samo sakriva od onih kojih vam vire iza leđa dok upisujete lozinku, ali se upisani znakovi ne šifruju niti obezbjeđuju. Za šifrovanje je neophodno da se na Web serveru izvršava softver tzv. bezbjednog servera (eng. secure server). Ako hoćete da umetnete polje za lozinku, ubacite obično polje za jedan red teksta, a za vrijednost opcije `Type` na panou svojstava izaberite `Password`. Ova opcija obezbjeđuje prikazivanje kružića ili zvjezdica dok posjetilac upisuje podatke u polje. Polja za lozinke mogu primati samo jedan red teksta. Vrijednost parametra `Max Chars` za lozinke treba zadati prema maksimalnoj dužini lozinke na serveru.

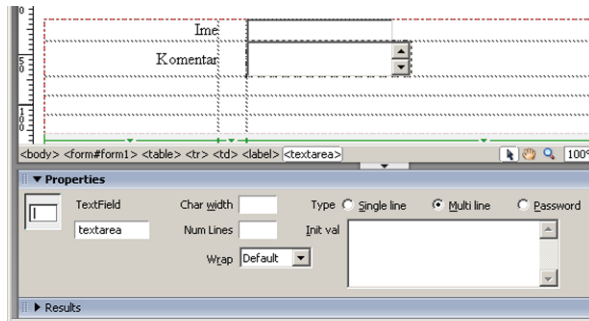
6.2. Dodavanje polja za više redova teksta

Polja za više redova teksta olakšavaju prikupljanje velikih količina informacija od posjetilaca, jer daju oblast od više redova gdje posjetilac može da upisuje svoj

tekst. U tipična polja za više redova teksta posjetioци upisuju svoje komentare, mišljenja i zahtjeve. U ovoj vježbi stavit ćete polje za više redova teksta u tabelu unutar obrasca.

Polje za više redova teksta biće umetnuto u istu kolonu kao i polja za jedan red teksta u prethodnom primjeru.

Izaberite Insert, Form Objects, Textarea.



Slika 57. Polje za više redova teksta

Polje za više redova teksta umetnuo je na obrazac, pa se na panou Properties vide njegova svojstva. Polje za jedan red teksta možete da pretvorite u polje za više redova teksta: izaberite ga a zatim pritisnite radio-dugme Multi line u opciju Type na panou svojstava.

Prilikom umetanja polja za više redova teksta, na panou svojstava imate i jednu dodatnu opciju. Padajuća lista Wrap dostupna je samo za višeredna polja. (Nedostupna je i siva za jednoredna polja i za polja za lozinke). U njoj se bira način prikazivanja teksta upisanog u polje, ukoliko ga ima više nego što staje na njegovu vidljivu površinu. Opcije su Default, Off, Virtual i Physical.

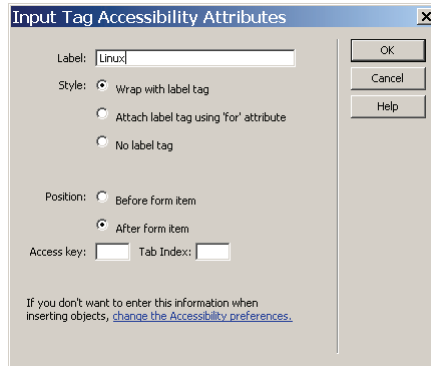
- Default koristi podrazumijevani parametar browsera. Ova opcija je automatski izabrana kada odaberete Multi line za opciju Type.
- Off sprečava prelamanje teksta u novi red. Tekst se nastavlja u jednom redu sve dok se ne pritisne Enter (Return).
- Virtual prelama tekst u novi red kada se stigne do ivice vidljive površine polja, ali se u tekst koji se šalje serveru ne ubacuju znakovi za prelazak u novi red.
- Physical prelama tekst u novi red kada se stigne do ivice vidljive površine polja i u tekst koji se šalje serveru ubacuje znakove za prelazak u novi red.

Polje "Num Lines" na panou svojstava određuje broj redova koji se prikazuju na površini za pomjeranje teksta polja. (Broj redova koje posjetilac može da upiše nije ograničen).

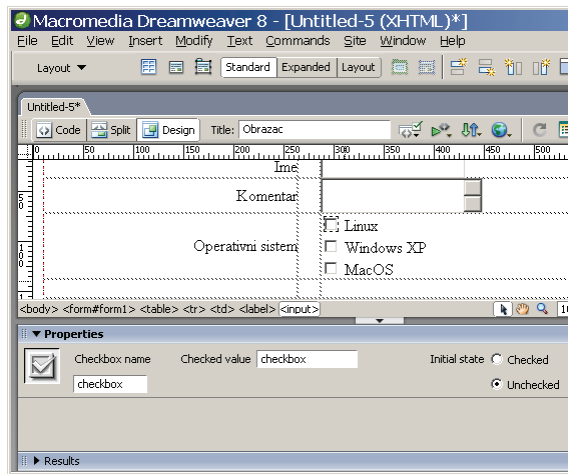
6.3. Dodavanje polja za potvrdu

Polja za potvrdu omogućavaju korisnicima da izaberu jednu ili više opcija u grupi povezanih stavki. Polja za potvrdu se obično koriste kada posjetilac treba da izabere koliko god hoće prikazanih opcija. Ako hoćete da posjetilac izabere samo jednu opciju, onda upotrijebite radio-dugme.

Izaberite Insert, Form, Check Box.



Slika 58. Polje "Label" će ispisati opis pored polja za potvrdu (checkbox-a)



Slika 59. Osobine polja za potvrdu

Polje za potvrdu je umetnuto u obrazac, a pano Properties prikazuje njegova svojstva. Obično se u obrascu dodaje više polja za potvrdu (odvojite ih sa Shift+Enter).

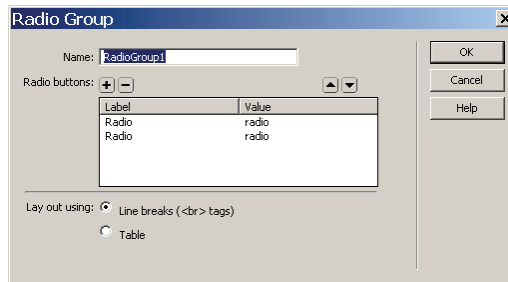
Dok dodajete objekte u obrazac, tabela se produžava naniže kako bi sve to primila. Pri tom se može desiti da se crvena isprekidana linija ne pomjera zajedno sa dnom tabele. U tom slučaju, pritisnite izvan tabele u prozoru dokumenta i natjerat ćete Dreamweaver da osvježi prikaz tabele.

Kada u dokument upisujete natpise za polja obrasca, pazite na to da tekst bude dovoljno blizu odgovarajućeg dugmeta ili polja, i ne previše blizu ostalih objekata na obrascu. Tako nećete zbuniti posjetioce Web lokacije.

Opcija "Initial State" može imati dvije vrijednosti, od kojih zavisi da li će polje biti uključeno (checked) ili isključeno (unchecked) kada korisnik prvi put otvori stranicu sa obrascem. Naravno, korisnik će moći mijenjati stanje tih polja, to su samo početna stanja.

6.4. Dodavanje radio-dugmadi

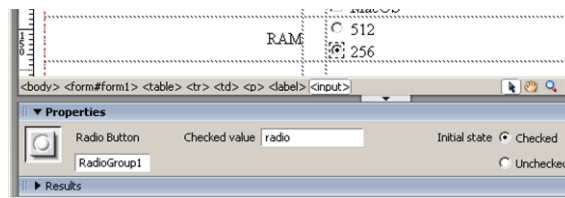
Radio-dugmad su grupa opcija. Biranje jedne opcije automatski poništava izbor svih ostalih opcija grupe. Radio-dugmad se obično koristi za izbor vrste kreditne kartice i odgovore tipa da/ne. Izaberite Insert, Form, Radio Group.



Slika 60. Dijalog za kreiranje grupe radio-dugmadi

Pomoću dugmadi plus (+) i minus (-) dodajete i brišete stavke u listama. Redoslijed dugmadi mijenjate pomoću dugmadi s trouglovima okrenutim nagore i nadole (slika 60). Pritisnite dugme plus (+) i zamijenite novi primjerak riječi "Radio" drugom riječju (npr. brojem 512). U koloni Value, zamijenite sve primjerke riječju "radio" odgovarajućim tekstom.

Kada koristite radio-dugmad, istim imenom morate da nazovete cijelu grupu stavki. Radio-dugmad treba da omogući izbor samo jedne stavke u grupi.



Slika 61. Grupa od dva radio-dugmeta

Ako umetnete jedno po jedno radio-dugme (pomoću naredbe Insert, Form, Radio Button), svu tu dugmad možete da učinite dijelovima iste grupe kada im date isto ime. Korištenje istog imena za više radio-dugmadi označava da ta dugmad pripadaju istoj grupi. Ako imena nisu ista, ta radio-dugmad se smatra pripadnicima

različitih grupa, što je protivno svrsi radio-dugmadi. Možete da umetnete i samo jedno radio-dugme, ali imajte u vidu da, kada ga posjetilac pritisne, jedini način da poništi izbor tog dugmeta jeste da obriše cijeli sadržaj obrasca dugmetom Reset.

Vodite računa i o tome da se u imenima objekata na obrascu razlikuju velika i mala slova.

Svaka grupa radio-dugmadi ima podrazumijevano najmanje dva dugmeta. Dodajte ih više, prema potrebi. Kada pritisnete primjerak dugmeta u listama, istaknut ćete taj tekst i moći da ga mijenjate.

Kada posjetilac pošalje obrazac, njegovi podaci se šalju skripte koja obrađuje obrasce na serveru. Važno je da svakom radio-dugmetu date drugu vrijednost (u listi Value), da biste znali šta je posjetilac izabrao, tj. koje je dugme pritisnuo.

Opcija Line Breaks stavlja radio-dugmad u dokument tako da svaka stavka stoji u zasebnom redu teksta. Opcija Table umeće tabelu, tako da svaka stavka stoji u zasebnom redu tabele.

Ako hoćete da testirate radio-dugmad, pritisnite svako dugme u grupi. Pri tom mora da bude poništen izbor svih ostalih dugmadi u grupi. Testiranje i biranje se mora obaviti u browseru, jer Dreamweaver ne prikazuje ni izbor ni potvrdu objekata na obrascima u prozoru dokumenta.

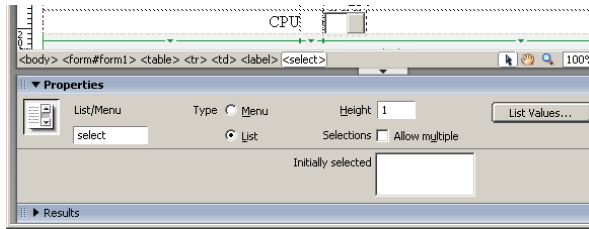
6.5. Dodavanje spiskova i menija

Spisak/meni omogućava da posjetilac bira opcije iz liste čija se sadržaj pomjera, ili iz menija. Lista čija se sadržaj pomjera omogućava da posjetilac izabere više susjednih ili nesusjednih stavki, dok meni ograničava posjetioca na izbor samo jedne stavke. U oba slučaja, izabrane stavke bivaju istaknute.

Izaberite Insert, Forms, List/Menu. Na panou svojstva izaberite opciju List u polju Type i upišite 4 u polje Height. Potvrdite vrijednost Allow multiple za polje Selections.

Na obrazac je dodat mali meni, a i pano Properties prikazuje svojstva menija, jer Dreamweaver podrazumijevano umeće meni, a ne listu. Promijenili ste format menija u format liste čija se sadržaj pomjera. Da bi objekat na obrascu bio promijenjen od menija u listu, morate mu zadati visinu (eng. height) veću od 1.

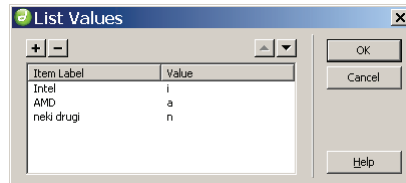
Format liste ima i dodatnu opciju koje nema u slučaju menija. Možete izabrati da li će biti zadovoljeno biranje više opcija (eng. allow multiple selections) u listi. Podrazumijevano, ova opcija nije potvrđena. Ako je potvrdite, posjetilac može da izabere nesusjedne stavke držeći taster Ctrl. Susjedne stavke posjetilac bira držeći taster Shift. Ako odlučite da dozvolite biranje više stavki u listi, obavijestite posjetioce o tome i o načinu na koji se to radi. Mnoštvo posjetilaca ne zna navedene naredbe. Uvijek treba pružiti posjetiocima sve informacije i alatke potrebne za interakciju s lokacijom.



Slika 62. Osobine liste

U polje Height treba upisati broj redova koji će se vidjeti u listi čiji se sadržaj pomjera. Ukoliko polje Selections ostavite nepotvrđeno, obavezno upišite broj vidljivih redova veći od 1. U protivnom, umjesto liste čija se sadržaj pomjera bit će prikazan meni. Uvijek zadajte visinu liste, jer ako to ne uradite, u njoj se prikazuje broj redova podrazumijevan u browseru posjetioca, a on je različit.

Pritisnite dugme List Values na panou svojstava. Otvorit će se okvir za dijalog List Values, isti za spiskove i za menije.



Slika 63. Dodavanje stavki u listu ili meni

Pritisnite dugme sa znakom plus (+) za dodavanje stavke. U polje "Item Label" upišite "Intel" i zatim pritisnite taster Tab. Širinu spiska/menija određuje dužina najduže stavke u spisku. U polje Value upišite "i". Ovaj tekst se šalje CGI skriptu ili serveru kao znak da je opcija izabrana.

Pritisnite Tab ili dugme sa znakom plus (+) u gornjem lijevom dijelu okvira za dijalog, da biste dodali još jednu opciju u spisak. Za brisanje stavki iz dijaloga List Values upotrijebite dugme sa znakom minus (-).

Ponovite te korake da biste dodali u spisak stavke "AMD" i "neki drugi". U polje Value za ove stavke upišite "a" i "n". Za mijenjanje redoslijeda stavki u listi upotrijebite dugmad iznad polja Value (s trouglovima okrenutim nagore i nadole).

Ako hoćete da dodate više stavki u listu, poslije unošenja svake stavke pritisnite Tab i automatski ćete prijeći na dodavanje nove. Zatvorite okvir za dijalog pritiskom na OK. Lista prikazuje dodate opcije.

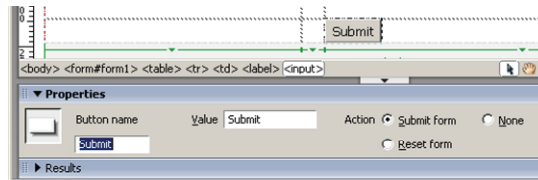
Za sve stavke liste/menija, na panou svojstava postoji opcija Initially Selected (unaprijed izabrano). Bilo koja od svaki može biti izabrana kada se strana učita. Možda to nije poželjno u listama čija se sadržaj pomjera, ali u menijima pomaže ako u prvom redu postoji primjer ili uputstvo kao "Izaberite jednu..."

Ako neku stavku unaprijed potvrdite, nema načina da to poništite, tj. da bi stavka izgubila svoju potvrdu, morate je obrisati u okviru za dijalog List Values i ponovo dodati.

6.6. Dodavanje dugmadi

Obrasci najčešće imaju dva dugmeta: jedno za slanje (Submit) i drugo za brisanje podataka iz obrasca (Reset). Dugmetom Submit saopštava se browseru da pošalje podatke u skladu sa zadatom akcijom (Action) i metodom (Method). Dugme Reset briše informacije iz svih polja na strani.

Izaberite Insert, Form, Button.



Slika 64. Dugme za slanje obrasca

Dugme Submit je umetnuto na obrazac, a pano Properties prikazuje njegova svojstva. Kako je dugme Submit podrazumijevano, ne mijenjate nijednu opciju za ovo dugme.

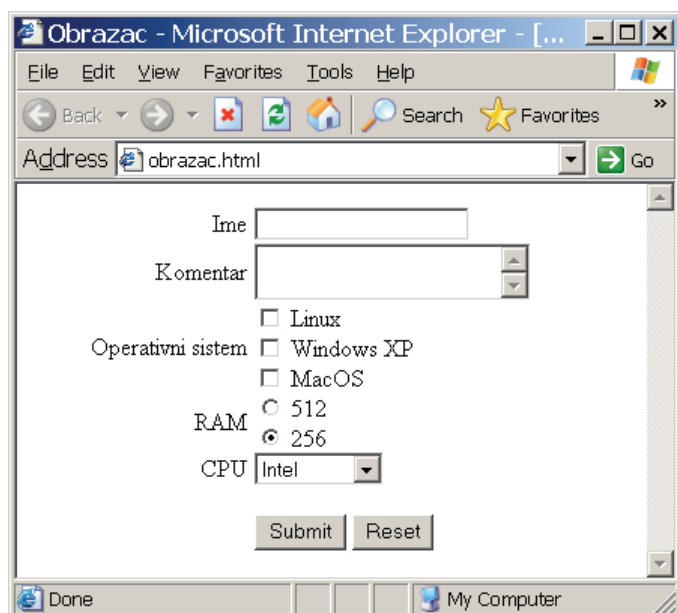
Dodajte još jedno dugme Submit. Jedina razlika je u imenu - ovo se zove submit2, jer dva dugmeta ne smiju imati isto ime. Jedini objekti na obrascu koji mogu (i treba) da imaju isto ime jesu radio-dugmad koja pripadaju istoj grupi. Dugme s kojim radite u ovoj vježbi razlikuje se, jer se ne može grupisati s drugom dugmadi i dodijeliti će mu se samostalna akcija u sljedećem koraku.

Izaberite Reset form u području Action na panou svojstava. Tekst u polju Label (oznaka dugmeta na obrascu) automatski se mijenja u "Reset". Interno ime dugmeta (eng. button name) ostaje nepromijenjeno.

Akcija resetovanja koju ste izabrali znači da se sva tekstualna polja, polja za potvrdu i radio-dugmad brišu i vraćaju u prvobitno stanje, koje su imala kada je strana bila učitana.

Treća opcija u području Action je None. Za razliku od opcija Submit i Reset, opciji None nije pridružena nikakva akcija. Da bi takvo dugme nešto uradilo, treba ga povezati sa skriptom (napisanim u JavaScriptu, na primjer), koji će recimo, izračunati ukupan zbir ili kamatu i posjetiocu dati konačan rezultat.

Dugmadi dajte jasna imena, a u skladu s očekivanjima posjetilaca. Submit i Reset su uobičajene oznake dugmadi na obrascima koje posjetioци razumijevaju, jer se veoma često koriste.



Slika 65. Gotov obrazac u Internet Exploreru

6.7. Provjera valjanosti podataka u obrascima

Postupak Validate Form provjerava da li je posjetilac unio odgovarajuću sadržaj u polja obrasca. Slučajno ili namjerno, posjetioci katkada unose pogrešne podatke u polja ili ih preskaču. Umjesto da čekate na slanje obrasca, možete da provjerite podatke dok ih posjetilac unosi ili neposredno prije slanja serveru.

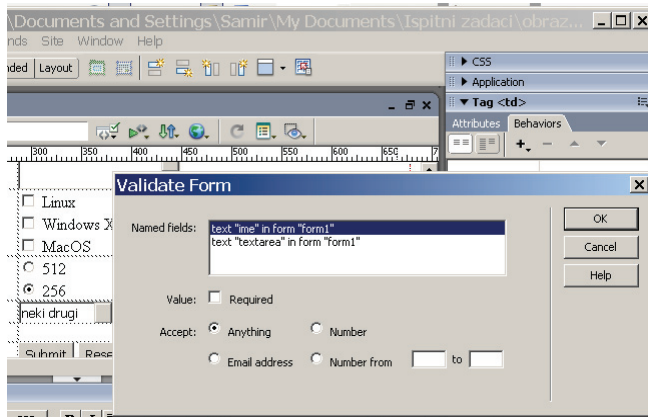
U ovom primjeru, dodat ćete postupak koji provjerava informacije upisane u obrazac, kako bi sva polja bila popunjena odgovarajućim podacima.

Otvorite pano Behaviors u grupi panoa Design. Za otvaranje panoa Behaviors mogli ste da izaberete i Window, Behaviors.

Pritisnite dugme plus (+) i u padajućoj listi na panou Behaviors izaberite Validate Form. Otvorit će se okvir za dijalog Validate Form sa spiskom polja na obrascu; prvo polje u spisku je podrazumijevano izabrano.

Izaberite polje ime u spisku Named Fields okvira za dijalog. Potvrdite vrijednost Required za opciju Value. Time ste zadali da je neophodno da polje ime nešto sadrži. Akcija Validate Form se dodaje na pano Behaviors, kao posljedica događaja onBlur. Ako posjetilac pređe u sljedeće polje, a da ovo ne popuni, pojavit će se poruka o grešci. Događaj onBlur se aktivira kada posjetilac napusti polje. U ovom primjeru, ako posjetilac pređe tabulatorom u sljedeće polje obrasca, a polje "ime" ostavi prazno, prikazat će se poruka o grešci. Ukoliko posjetilac pritisne određena

polja i popuni ih, umjesto da tabulatorom prolazi kroz sva polja obrasca, ova početna provjera valjanosti se zanemaruje.



Slika 66. Postupak Validate Form

Premda vam možda izgleda kao da je dobro provjeravati svako polje dok se posjetilac kreće kroz obrazac, budite oprezni, jer tako možete da naljutite posjetioce. Mnogi ne popunjavaju polja redom, nego preskaču i naknadno se vraćaju da popune propušteno.

Ako izaberete Email Address iz grupe opcija Accept, ova opcija provjerava da li je posjetilac upisao znak @ u polje. Ne provjerava se da li upisana adresa elektronske pošte zaista postoji.

Opcija Number zadaje da polje smije sadržati samo cifre. Opcija Number from zadaje da polje smije sadržati samo broj unutar zadatog opsega brojeva, što je naročito pogodno za godine.

Izaberite dugme Submit u prozoru dokumenta. Pritisnite dugme plus (+) i u padajućoj listi na panou Behaviors izaberite Validate Form. Akcija Validate Form se dodaje na pano Behaviors, kao posljedica događaja onSubmit.

Izaberite polje Ime iz spiska Named Fields u okviru za dijalog. Potvrdite opciju Required za parametar Value. Pritisnite Anything u grupi opcija Accept. Opcija Anything zadaje da se u polje nešto mora upisati, ali ne određuje vrstu tih podataka. Kada posjetilac pritisne dugme Submit, provjerit će se valjanost.

Ako zadate početnu vrijednost tekstualnog polja, nema smisla provjeravati valjanost podataka u njemu (pogotovo ako je izabrana opcija Anything), jer u njemu već stoji tekst.

Pritisnite polje ime i potom taster Tab, ali u polje nemojte upisati ništa. Prikazat će se poruka o grešci koju generiše postupak provjere valjanosti podataka na obrascu Validate Form. Provjera popunjenosti pojedinih polja odgovarajućim podacima radi jedino u slučaju da posjetilac (tasterom Tab) premješta tačku umetanja iz polja u polje. U protivnom, ne prikazuje se nikakva poruka o grešci. Dakle, ako

posjetilac ne koristi taster Tab i ne postavlja kursor u određeno polje, poruka o grešci se ne prikazuje.

Ako hoćete da se i polje ime provjeri nakon što posjetilac pritisne dugme Submit, ponovo otvorite okvir za dijalog Validate Form: dvaput pritisnite akciju Validate Form na panou Behaviors. Izaberite polje ime i željenu opciju. Kada završite, pritisnite OK.

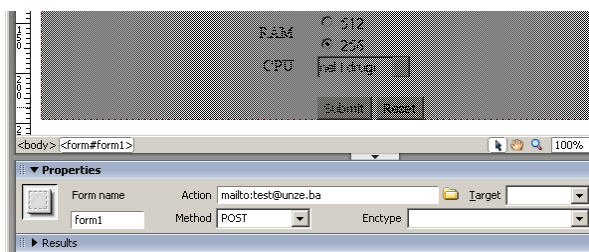
6.8. Testiranje obrasca

Obrazac se može poslati na adresu e-pošte čak i ako se ne serveru na izvršava CGI skripta. Ovu metodu koristite samo za testiranje svojih obrazaca.

U dokumentu izaberite obrazac biranjem oznake <form> na biraču HTML oznaka. U polje Action na panou svojstava upišite mailto: i svoju adresu e-pošte.

Ne zaboravite da upišete dvotačku i ne ostavljajte razmake. Adresa treba da liči na mailto:test@unze.ba; svoju adresu e-pošte upišite umjesto "test@unze.ba".

Izaberite POST u padajućoj listi Method i upišite text/plain u polje Enctype.



Slika 67. Dodjeljivanje akcije slanja obrasca na e-mail

Zadali ste vrstu kodiranja (eng. encode type) za običan (eng. plain) tekst. U protivnom, poslati tekst bi kodiranjem bio doveden u gotovo nerazumljiv oblik.

Parametar Enctype zadaje način kodiranja podataka sa obrasca. Opcija text/plain formatira podatke tako da svaki element obrasca dobije vlastiti red teksta. Tako se lakše čitaju rezultati u e-mail poruci. Ako ne zadate vrstu kodiranja, browser će upotrijebiti podrazumijevanu vrstu koja formatira podatke. Kako je podrazumijevana ona vrsta koju treba koristiti u najvećem broju slučajeva, vrsta kodiranja se obično ne zadaje. Ovaj primjer je izuzetak, jer podatke šaljete u poruci e-pošte radi testiranja.

Ako hoćete da poruke e-pošte s podacima obrazaca imaju i temu u polju Subject, promijenite sadržaj polja Action na:

mailto:test@unze.ba? Subject=Tema se upisuje ovdje.

Dio "? Subject =" definiše tekst koji slijedi kao temu poruke. U tekstu teme može biti razmaka, ali ne i specijalnih znakova kao što su navodnici, polunavodnici,

tačke i kose crte (sem onih u dijelu "? Subject =" koji razdvajaju adresu poruke od njene teme), jer bi se pogrešno protumačili u HTML kodu.

Ova akcija obrasca ne radi u svim browserima. Upotrijebite je samo za testiranje. Za slanje obrazaca uvijek koristite CGI skripte ili PHP programe. Ako vaš browser nije konfigurisan za slanje e-pošte, nećete moći da testirate obrazac na ovaj način.

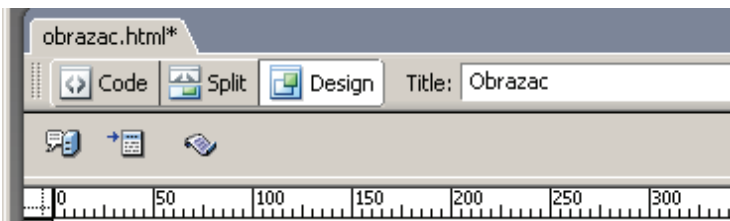
7. Meta oznake i komentari

U kod se mogu umetnuti neki elementi koji se ne vide u browseru, ali su ipak važni za dokument. Meta oznake i HTML komentari primjeri su takvih elemenata.

Meta oznake se koriste u više svrha: one identifikuju i opisuju dokument, daju podatke o vlasniku autorskih prava, identifikuju njegove autore ili dizajnere, preusmjeravaju posjetioce na druge strane, određuju njegove autore ili dizajnere, preusmjeravaju posjetioce na druge strane, određuju izgled sažetka dokumenta koji se prikazuje u nekim pretraživačima i utiču na redoslijed pojavljivanja dokumenta u njima. HTML komentari se koriste za pravljenje bilježaka u kodu, za objašnjavanje svrhe pojedinih dijelova koda, ili kada treba onemogućavati izvršavanje dijela koda, da se on ne obriše.

Izaberite View, Head Content.

Odmah iznad okna za prikaz elemenata pojavit će se područje Head Content. U njemu će se pojavljivati ikonice za elemente smještene u zaglavlju dokumenta, između oznaka `<head>` i `</head>`. Trenutno su u ovom području ikonice za naslov dokumenta, za meta oznaku `http-equiv` i dijelovi Java Script koda koji treba da budu u zaglavlju dokumenta.



Slika 68. Ikonice za zaglavlje dokumenta

Kada pritisnete neku od ikonica na panou zaglavlja (Head Content), biće istaknut kod odgovarajućih elemenata u oknu koda.

Da biste vidjeli pano zaglavlja, morate biti u prikazu elemenata (Design View) ili u kombinovanom prikazu elemenata i koda (Code and Design View). Ako radite u prikazu koda, opcija View, Head Content nije dostupna. U ovom primjeru koristite kombinovani prikaz elemenata i koda.

Dovedite pokazivač miša iznad ivice koja razdvaja prikaz elemenata od prikaza koda. Kada se pokazivač pretvori u dvostranu strelicu, pritisnite i povucite ivicu nagore, kako bi se okno koda povećalo, a okno elemenata smanjilo.

Sadržaj zaglavlja vidite pri vrhu prozora dokumenta, neposredno iznad okna elemenata. I od okna elemenata vidi se minimalan dio, dok se lako vidi odgovarajući kod u donjem oknu kada se pritisnu ikonice na panou zaglavlja.

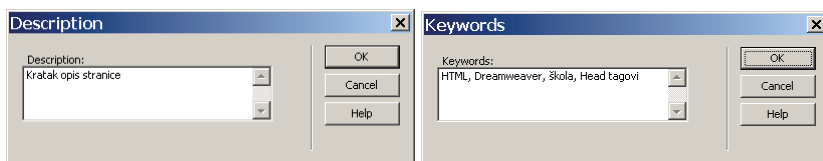
U oknu koda, postavite tačku umetanja na kraj reda koji sadrži naslov dokumenta, neposredno poslije oznake `</title>`, i zatim pritisnite Enter. Izaberite Insert, HTML, Head Tags, Description.

Otvorit će se okvir za dijalog Description (opis), s poljem u koje možete da upišete opis strane. Neki pretraživači u svoje rezultate uključuju meta oznaku description, u koju se stavlja kratak sažetak strane. Dodavanje ove meta oznake važno je za promociju prezentacije koju pravite.

Opis treba da je kratak - najviše 200 znakova. U većini pretraživača ograničen je broj znakova i odbacuje se sve što prekorači taj broj. Dobar opis je veoma kratak i sažeto upućuje na sadržaj dokumenta.

Kursor treba da je na početku sljedećeg reda.

Kada pišete spisak ključnih riječi (eng. keywords), razdvojite pojedine riječi i izraze zareзима. Ne ponavljajte istu ključnu riječ ili izraz - ako u spisak ključnih riječi stavite "novost novost novost novost", to će se smatrati slanjem neželjene sadržine na Web, zbog ponavljanja riječi "novost". Ključne riječi treba da predstavljaju sadržaj strane i da se na njoj zaista upotrebljavaju.



```
<title>Obrazac</title>
<meta name="description" content="Kratak opis stranice"/>
<meta name="keywords" content="HTML, Dreamweaver, škola, Head tagovi"/>
```

Slika 69. Insert, HTML, Head Tags, Description

Pomenuta dva meta-taga (meta-oznake): Description i Keywords se najčešće koriste kod opisa stranica za koje je bitno da se pojave u pretraživačkim mašinama kao što su Google, Yahoo i sl.

8. Pravljenje spoljnih kaskadnih opisa stilova

Opisi stilova mogu biti snimljeni sami za sebe i povezani s jednim ili više dokumenata. Kada napravite spoljni opis stila za dokument, on se automatski vezuje s dokumentom za koji je bio napravljen. Spoljni opis stila je tekstualna datoteka koja sadrži samo specifikaciju stila. Jedna od prednosti spoljnih opisa stilova jeste da ih možete povezati i s drugim dokumentima u prezentaciji i tako postići ujednačen izgled svih njenih strana.

Druga prednost korištenja spoljnih opisa stilova jeste mogućnost uređivanja stilova, jer se sve izmjene automatski prenose na sve dokumente povezane s tim opisom stila. Upotrijebite spoljni opis stila da biste ujednačili izgled teksta na više strana i da biste definisali izmjene koje će se automatski primijeniti na sve povezane dokumente.

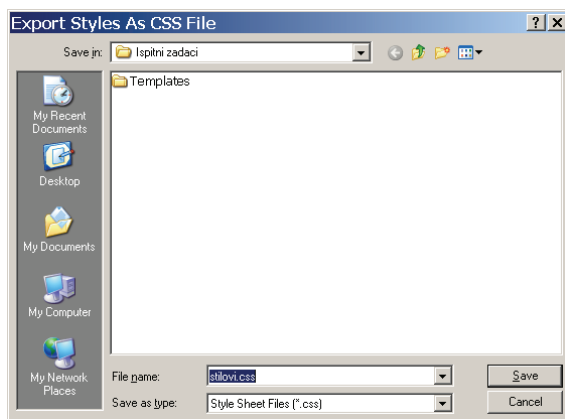
Kreirajte novu datoteku i snimite je pod imenom `primjer2.html`. U tu datoteku unesite neki tekst, označite taj tekst i promijenite mu sljedeće osobine:

- Font: Times New Roman
- Veličina slova: 14
- Boja slova: crvena (#FF0000)

U kodu dokumenta označeni tekst će se umetnuti između oznaka `<span class="Style 1">` i `</span>`, a u zaglavlju dokumenta će se kreirati definicija stila "Style 1".

```
<style type="text/css">
<!--
.style1 {
    font-family: "Times New Roman", Times, serif;
    font-size: 16px;
    font-weight: bold;
    color: #FF0000;
}
-->
</style>
```

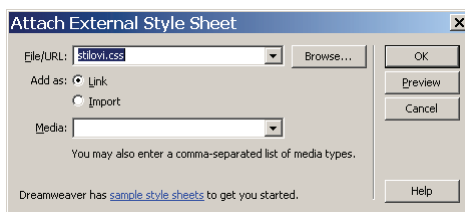
Naredbom `Text`, `CSS Styles`, `Export` možete taj stil izvesti u spoljnu datoteku, kojoj možete dati proizvoljno ime (u ovom primjeru `stilovi.css`), a ta datoteka automatski dobija ekstenziju `.css`.



Slika 70. Izvoz stila u spoljnu datoteku kaskadnih opisa stilova

Ovako kreirana datoteka stilovi.css je obična tekst datoteka, koja sadrži samo definicije stilova.

Naredbom Text, CSS Styles, Attach style sheet pridružujete datoteku sa spoljnim opisima stilova html datoteci.



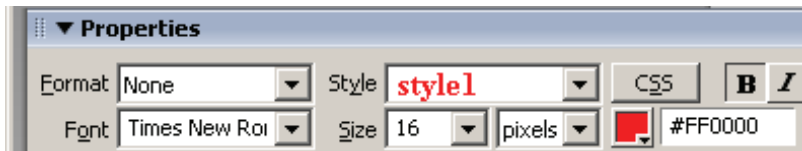
Slika 71. Pridruživanje datoteke sa spoljnim kaskadnim opisima stilova

Unutar oznaka <head> i </head> pojaviće se tekst:

```
<link href="stilovi.css" rel="stylesheet"
type="text/css" />
```

Nakon toga se može obrisati dio koda između <style type="text/css"> i </style>, jer se ta definicija već nalazi u spoljnoj datoteci.

Tako kreirani opis stila se na neki drugi tekst primjenjuje tako što se tekst označi, a zatim u panelu svojstava izabere taj opis stila iz polja Style.

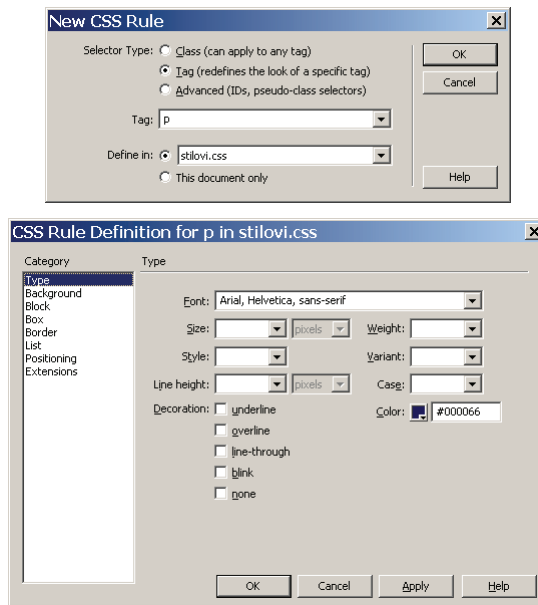


Slika 72. Izbor već definisanog opisa stila

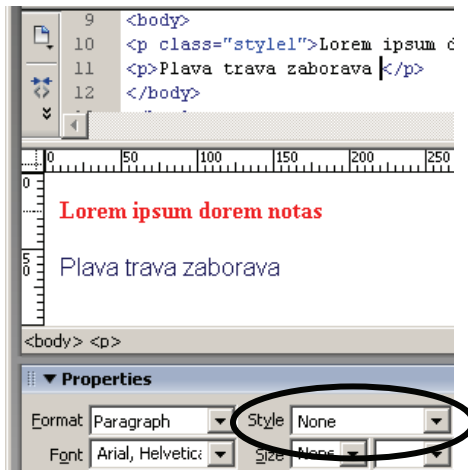
Dreamweaver podržava neke attribute koji jesu u specifikaciji CSS, ali ih ne podržavaju svi tekući browseri. Obavezno provjerite stilove i u Netscape

Navigatoru i u Internet Exploreru, da biste vidjeli da li ovi browseri podržavaju attribute koje ste izabrali.

Mogu se redefinisati postojeće oznake (tagovi) tako da imaju posebne osobine, kao što su boja slova, veličina, font,... Izaberite Text, CSS Styles, New. U dijalogu (slika 72) izaberite "Tag" za redefinisanje postojećih oznaka, a zatim u padajućem meniju "Tag" izaberite npr. oznaku "p" (tako će svi pasusi u dokumentu koji se nalaze unutar oznaka <p> i </p> imati te osobine).



Slika 72. Redefinisanje standardnih oznaka (oznaka <p>)



Slika 73. Iako je drugi pasus u tekstu označen bez posebnog opisa stila, on će imati osobine definisane u spoljnoj datoteci

U datoteci "stilovi.css" se nakon redefinisavanja stila oznake "p" pojavljuje sljedeći tekst:

```
p {
    font-family: Arial, Helvetica, sans-serif;
    color: #000066;
}
```

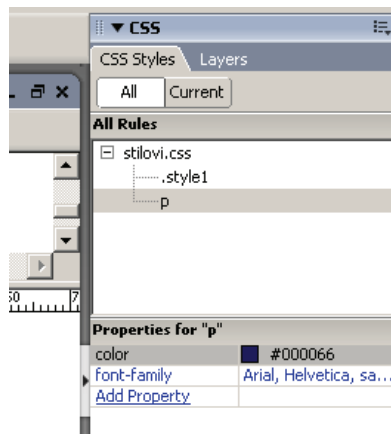
Obratite pažnju da se ispred oznake "p" ne nalazi tačka (za razliku od prethodnog stila, čije je ime napisano kao ".style1". Tačka ispred imena označava da se taj opis stila može dodati uz bilo koju drugu html oznaku (naprimjer, uz oznaku reda tabele <tr> ili oznaku hiperlinka <a>), tako što se izabere taj stil iz panela svojstava ili se uz oznaku upiše class="naziv opisa stila", kao u sljedećem primjeru:

```
<p class="style1">Lorem ipsum dorem notas</p>
```

Sve osobine nabrojane u definiciji opisa stila nalaze se unutar vitičastih zagrada "{" i "}", a svaka se završava znakom tačka-zarez (;).

8.1. Mijenjanje postojećeg stila

Morate znati kako da promijenite stil definisan u postojećem opisu stila. Izmjenom spoljnog opisa stila mijenjate sve dokumente koji ga koriste. To je korisno, jer mijenjanjem spoljnog opisa stila možete da promijenite izgled teksta na više strana ili u cijeloj prezentaciji.



Slika 74. Izmjena postojećih opisa stilova

Potrebno je otvoriti pano CSS (naredba Window, CSS Styles). U spisku stilova definisanih u opisu stila stilovi.css, izaberite stil "p" (ako se ne vide svi stilovi, treba kliknuti na tipku "All"). Desnom tipkom miša kliknite na ime stila kojeg treba promijeniti i zadajte naredbu Edit.

Otvorit će se okvir za dijalog CSS Style Definition, isti kao onaj u prethodnom primjeru u kome ste birali attribute formatiranja.

Sve što izmijenite primijenjeno je na dokument. Kada bi drugi dokumenti koristili ovaj opis stila, sav tekst unutar oznaka `<p>` u njima, bio bi formatiran prema redefiniciji stila koju ste upravo napravili.

Kaskadni opis stila se primjenjuje po uređenoj sekvenci. Kada se na isti tekst primjeni dva i više stilova, browser primjenjuje kombinaciju atributa svih pozvanih stilova. Ako je na isti tekst primijenjeno više stilova čiji se atributi sukobljavaju, browser koristi specifikacije stila koji je najviše ugniježđen (najbliži samom tekstu). Stilovi koje ste posljednje primijenili ugniježđeni su unutar onih koji su već primijenjeni. Pošto su stilovi koji su posljednji primijenjeni, u stvari HTML oznake koje su fizički najbliže tekstu, one imaju prioritet nad prethodnim stilovima i zadaju konačan izgled teksta. Redoslijed primjene stilova je:

- spoljni stilovi (stilovi koji su najdalje od teksta; najniži prioritet)
- unutrašnji stilovi
- namjenski stilovi primijenjeni na tekst strane
- lokalno formatiranje, recimo polucrni ili kurzivni ispis teksta, primijenjeno na tekst strane pomoću panoa svojstava (stil najbliži tekstu; najviši prioritet, jači od svih opcija zadatih u gore nabrojanim stilovima).

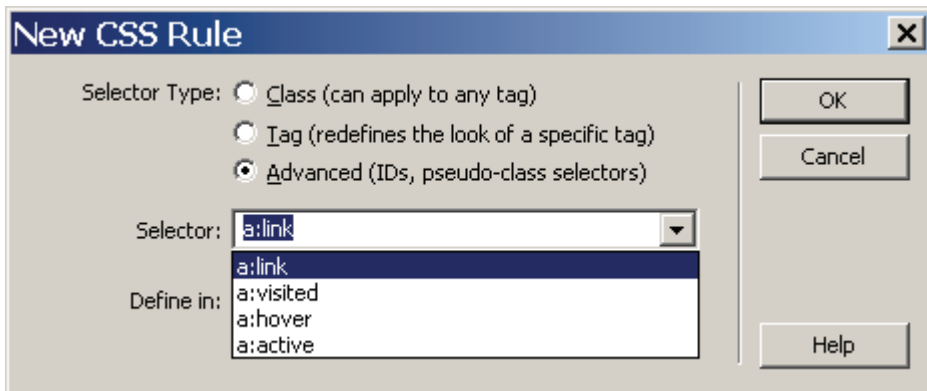
Ako dokument koristi spoljni opis stila, stilovi u tom opisu primjenjuje se na cio dokument. Pretpostavimo, međutim, da spoljni opis stila sadrži definicije za Heading 3 i Heading 4, a da ste napravili i jedan unutrašnji stil koji redefiniše oznaku Heading 3. Unutrašnji stil ima prednost pred spoljnim ukoliko im se atributi sukobe.

I ručno formatiranje dijelova teksta ima prednost nad stilovima. Pretpostavimo da ste pomoću panoa svojstava (Properties) primijenili drugu boju na jedan od redova teksta unutar oznaka Heading 3. To je lokalno formatiranje i ima prednost pred ostalim stilovima (ako oni zadaju boju). Međutim, atributi namjenskih stilova imaju prednost pred atributima stilova HTML oznaka. Ako hoćete da pomoću stilova definišete izgled pasusa, uklonite sve ostale opcije formatiranja (ukoliko se sukobljavaju sa stilovima).

Kad primijenite namjenski stil, kako ćete potom da ga uklonite? Ako je namjenski stil primijenjen na cio pasus, postavite tačku umetanja u pasus i pritisnite dugme No CSS Style na panou CSS Styles. Time ste obrisali i definiciju stila i njegovo formatiranje teksta. Ako je namjenski stil bio primijenjen sam na dio teksta u pasusu, postavite tačku umetanja u riječ formatiranu pomoću tog stila i zatim pritisnite dugme No CSS Style na panou CSS Styles.

8.2. Izrada kaskadnih opisa CSS stilova za hiperveze

Primjenom stilova možete da mijenjate izgled hiperveza na strani. Kaskadni opisi stilova mogu se upotrijebiti za mijenjanje izgleda dinamičke hiperveze s različitim atributima za njena različita stanja. Različita stanja oznake <a> aktiviraju se kada posjetilac npr. pokaže hipervezu, pritisne je itd. U padajućoj listi Selector u okviru za dijalog New Style, Dreamweaver nudi četiri standardna stanja oznake <a> koja olakšavaju izbor formatiranja hiperveza na stranama. Ova vrsta CSS stilova zove se pseudoklasa.



Slika 75. Izrada kaskadnih opisa CSS stilova za hiperveze

Četiri moguća stanja hiperveza pojavljuju se u padajućoj listi Selector zato što se ona mijenjaju u zavisnosti od aktivnosti posjetioca. Opcije u padajućoj listi su a:link; a:visited, a:hover i a:active.

- Opcija Active formatira hipervezu kada je izabrana ili neposredno pošto je pritisne posjetilac; podrazumijevana boja u ovom stanju je crvena. U padajućoj listi prikazuje se kao a:active.
- Opcija Link formatira hipervezu kada je u normalnom stanju; podrazumijevana boja u ovom stanju je plava. U padajućoj listi prikazuje se kao a:link.
- Opcija Visited formatira hipervezu nakon što ju je pritisnuo posjetilac; podrazumijevana boja u ovom stanju je ljubičasta. U padajućoj listi prikazuje se kao a:visited.
- Opcija Hover formatira hipervezu kada je posjetilac pokaže; podrazumijevana boja u ovom stanju je crvena. U padajućoj listi prikazuje se kao a:hover. (Ovu opciju ne podržavaju svi browseri).

Kaskadni opisi stilova hiperveza ne prikazuju se na panou CSS Styles.

Pomoću opisanih kaskadnih opisa stilova, hiperveze se mogu prikazivati i bez uobičajenog podvlačenja: u okviru za dijalog stilovi.css koji ste koristili u ovom primjeru, izaberite a:link i pritisnite dugme Edit. U okviru za dijalog Style

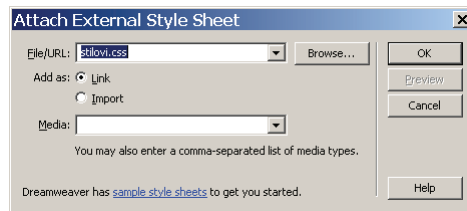
Definition koji će se otvoriti, izaberite None u grupi Decoration, i - više nema podvlačenja.

8.3. Povezivanje s postojećim opisom stila

Sada imate spoljni opis stila sa nekoliko definisanih stilova. Kako se radi o spoljnoj datoteci, možete da je povezujete sa raznim dokumentima. Definicije stilova (osim namjenskih) automatski se primjenjuju na sve dokumente povezane sa opisom sitla. Namjenski stilovi se ručno primjenjuju na pasuse ili izabrane dijelove teksta.

Dreamweaver nudi više gotovih kaskadnih opisa stilova koje možete koristiti za svoje Web prezentacije. Da biste upotrijebili neki od ovih stilova, izaberite File, New i na kartici General izaberite CSS Style Sheets u spisku Category. U spisku CSS Style Sheets izaberite opis stila koji hoćete da koristite i pritisnite dugme Create. Opis stilova možete da upotrebljavate kakvi jesu ili da ih mijenjate i prilagođavate. Novi kaskadni opis stila otvorit će se u prikazu koda. Snimite novi CSS dokument sa oznakom tipa .css. Povežite dokumente sa opisom stila prateći ovaj postupak.

Zadajte naredbu Text, CSS Styles, Attach Style Sheet. Otvorit će se okvir za dijalog Link External Style Sheet.



Slika 76. Povezivanje s postojećim opisom stila

Povezali ste spoljni opis stila s dokumentom. Sve oznake u tekućem dokumentu mijenjaju se u skladu sa stilovima spoljnog opisa stila.

9. Rad s bojama

Rad s bojama omogućila je tvrtka Netscape kada je napravila verziju 1.1 svojeg Navigatora, no danas gotovo svi pretraživački programi podržavaju to proširenje jezika HTML.

Da biste pozadinu svojih dokumenata mogli ispuniti slikovnim uzorkom ili nekom bojom, te da biste tekst i hipertekstualne veze obojili u neke nove nijanse, poslužite se sa šest argumenata koji se mogu dopisati uz naredbu `<BODY>`. Argumentima `BACKGROUND` i `BGCOLOR` kontrolira se izgled, odnosno boja "papira" na kojem je "otisnut" sadržaj vaše stranice. Pomoću preostala četiri argumenta, `TEXT`, `LINK`, `VLINK` i `ALINK` određuje se boja različitih dijelova teksta. Argumentu `TEXT` dodjeljuje se boja kojom je "otisnut" sadržaj vaše stranice. Pomoću preostala četiri argumenta, `TEXT`, `LINK`, `VLINK` i `ALINK` određuje se boja različitih dijelova teksta. Argumentu `TEXT` dodjeljuje se boja kojom će biti ispisan običan tekst, koji se u "neobojenim" stranicama uvijek prikazuje crnom bojom. Vrijednost argumenta `LINK` predstavlja boju hipertekstualnih veza, koja je inače plava, dok vrijednost argumenta `VLINK` predstavlja boju već posjećenih hipertekstualnih veza, koja je inače ljubičasta. Boja kojom će biti ispisan hipertekstualna veza u trenutku kada korisnik na nju klikne mišem dodjeljuje se argumentu `ALINK`.

9.1. Pozadina stranice

Da biste odredili koja slička će se koristiti za ispunjavanje pozadine Web dokumenta uzorkom, dodijelit ćete njezinu URL adresu (po mogućnosti relativnu) argumentu `BACKGROUND` koji valja dopisati uz naredbu `<BODY>`. Na primjer:

```
<BODY BACKGROUND="slike/uzorak.gif">
```

Kada se u pretraživački program učita stranica čije je tijelo označeno ovakvom naredbom, cijela njezina pozadina bit će ispunjena slikom `uzorak.gif` koja će se višestruko ponavljati, zalijepljena na samu sebe.

Nekih stvari, međutim, morate biti svjesni kada radite s pozadinskim slikama. Prije svega one moraju biti što manje. Što su manje, brže će se učitati na korisnikov računar. Drugo, koristite što je moguće manje boja. Neki operativni sistemi mogu prikazati samo ograničen broj boja, a također postoje i mnogi korisnici sa zastarjelim grafičkim karticama koje mogu prikazivati samo 256 boja. Ako ukupan broj boja koji se koristi na vašoj stranici prelazi kapacitet čitateljevih hardverskih mogućnosti, viškovi boja će se zamijeniti s bojama koje su dostupne, pa će vaša pozadinska slika izgledati, ružno. Stoga uvijek provjerite kako vaša pozadina izgleda na ekranskim razlučivostima s različitim brojem boja.

9.2. Kodiranje boja

Argumentu BGCOLOR, te preostalim argumentima koji se koriste za bojenje teksta, mora se dodijeliti tzv. RGB vrijednost. Radi se o uređenoj trojki vrijednosti kojima se određuju redom nijanse crvene, zelene i plave boje, kako bi se standardnim miješanjem tih triju nijansi dobila neka nova boja iz uobičajenog spektra podržanog na većini grafičkih kartica. U jeziku HTML, RGB vrijednost se određuje na slijedeći način:

```
#RRGGBB
```

Za svaku od tri spomenute boje koriste se po dva znaka ispred kojih je smješten simbol #. Svaki od ova tri para znakova predstavlja heksadecimalnu numeričku vrijednost između 0 i 255. Svaki od dva znaka "R", naprimjer, predstavlja neku vrijednost od 0 do 15. Prvi "R" označava broj "šesnaestica", a drugi "R" broj "jedinica". I onda se to sabere. Isto tako je i s oznakama "G" i "B". Prema tome, heksadecimalna vrijednost 46 jednaka je zbroju četiri šesnaestice i šest jedinica, što decimalno iznosi 70. Kao što vidite, simboli koje koristimo u obične brojke od 0 do 9, dok se za vrijednosti od 10 do 15 koriste slova A, B, C, D, E i F.

Crna boja se kodira kao RGB vrijednost #000000, dok je bijela sasvim suprotna - #FFFFFF. Svaka boja kojoj su sve vrijednosti RR, GG i BB jednake, predstavlja nijansu sive. Miješanje boja na računarima je vrlo jednostavan proces zbrajanja: veće vrijednosti predstavljaju više svjetla koje pripada konačnoj boji. Zbog toga je, naprimjer, #0000FF oznaka za svijetlo plavu, a ne tamnosmeđu, u što se možete uvjeriti ako pogledate tablicu 5. Evo i jednog jednostavnog primjera bojenja stranice i teksta:

```
<BODY BGCOLOR="#000000" TEXT="#FFFFFF" LINK="#FF0000"
VLINK="#400000" ALINK="#404000">
```

Pozadinu smo obojili u crno, tekst u bijelo, hipertekstualne veze u svijetlocrvenu, posjećene veze u tamnocrvenu, a one veze na koje korisnik klika mišem u žutu boju.

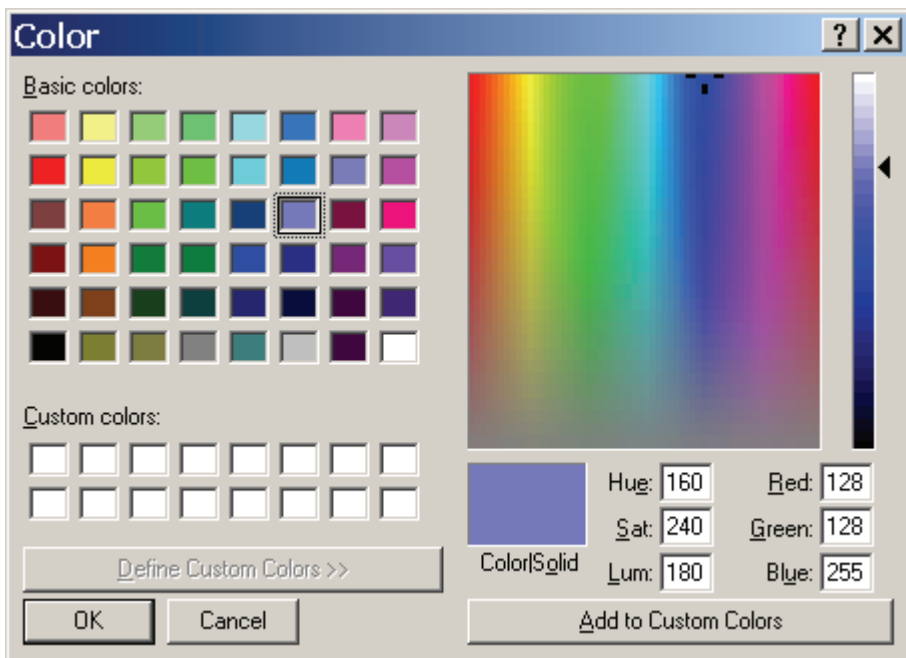
U tabeli 5 koja je dana u nastavku nalaze se RGB kodovi nekih uobičajenih boja, pa nećete morati sami računati.

Tabela 5: RGB kodovi nekih uobičajenih boja

Boja	Tamna	Srednja	Svijetla
Crvena	#400000	#7F0000	#FF0000
Žuta	#404000	#F77F00	#FFFF00
Zelena	#004000	#007F00	#00FF00
Cijan	#004040	#007F7F	#00FFFF
Plava	#000040	#00007F	#0000FF
Ljubičasta	#400040	#7F007F	#FF00FF

Šta će se dogoditi ako istu vrijednost dodijelite argumentima BGCOLOR i, naprimjer, VLINK? Kao i kod ispunjavanja pozadine uzorkom, obavezno provjerite kako vaša stranica izgleda na računarima koja mogu prikazati manje boja od vašeg. Svakako provjerite i stanje s već posjećenim hipertekstualnim vezama, kako ne biste došli u situaciju da morate iz iskustva odgovoriti na pitanje s početka ovog odlomka.

I za kraj, spomenut ćemo još i argument COLOR koji možete dopisati uz većinu naredbi kojima se obuhvaća određeni dio teksta, kakve su , i druge. Ovom argumentu može se dodijeliti RGB vrijednost neke boje na isti način kao što smo to radili i do sada, s time da će se neovisno o argumentima navedenima kod naredbe <BODY> dio teksta na koji se ovaj argument odnosi obojiti željenom bojom. To vam daje mogućnost kreiranja vrlo raznobojnih dokumenata, no budite pažljivi - što više boja koristite, to ćete više vremena morati posvetiti provjeravanju vaših stranica.



Slika 77. Izbor boja u Dreamweaveru: R, G i B se mogu birati i decimalno, a u html kodu će se pojaviti njihov heksadecimalni ekvivalent. U ovom slučaju, oznaka boje je #8080FF, što predstavlja po 50% crvene i zelene i 100% plave.

10. Slike u HTML dokumentima

10.1. Veličina slika

Pored mnoštva drugih stvari, World Wide Web je i masovni distributer slika. Umjetnička djela, fotografije, crteži, mape i drugi tipovi grafike sastavni su dio Weba i često znaju namamiti čitatelje da dovlače i čitave megabajte grafičkih elemenata na svoje računar. Arhive slika imaju svoju funkcionalnu ulogu na Webu, no organizacije i distributeri tih tipično ogromnih servera često će vas natjerati da nešto poželite, ali ne uspijete pogledati.

Kako onda možete objavljivati velike grafičke datoteke a da ne izgubite dio svojeg važnog auditorija? Ako ste Web administrator neke umjetničke galerije, kako ćete posljednju izložbu prezentirati na Mreži? Ili ako ste administrator poslovnog poduzeća, kako ćete objavljivati grafikone s trgovačkim profitima i gubicima? Odgovor je jednostavan: koristit ćete manje slike. Postoje dvije mjerljive veličine na Webu koje biste trebali poznavati: količina ekrana koju će zauzeti neka slika i veličina datoteke u kojoj je pohranjena.

Sve čemu se najobičniji korisnik Web stranica nada su brzina i rezultati, odnosno ako je moguće - brzi rezultati. Kako biste zadovoljili toj dominantnoj potrebi za efikasnošću, razmislite o upotrebi tzv. *thumbnail* sličica unutar Web stranica, koje će predstavljati hiperveze prema dokumentima unutar kojih se nalaze slike u svojoj punoj veličini.

Thumbnail sličice, što se može prevesti kao sličice veličine palčanog nokta, male su grafičke skice koje će korisnika hipertekstualno povezati s pravim slikama. One obično zauzimaju između dva i tri kvadratna centimetra ekrana, a svaka je velika svega pet do deset kilobajta. Čak i oni korisnici koji posjeduju vrlo sporu vezu s Internetom mogu tolerirati desetak takvih sličica unutar jedne Web stranice, no ukoliko ih posjedujete puno više, razmislite o tome da ih razvrstate u kategorije.

Ukoliko želite koristiti velike slike, iskoristite Netscape proširenje naredbe IMG koje je realizirano kroz argument LOWSRC. Ubacivanje slike unutar HTML dokumenta pomoću ovog proširenja, koji prepoznaju i drugi poznati pretraživači, obavlja se ovom sintaksom:

```
<IMG SRC="velika.gif" LOWSRC="mala.gif">
```

Time se pomoću jedne naredbe navode putanje i imena datoteka i manje i veće verzije iste slike. Prilikom prikazivanja stranice koja sadrži ovakvu naredbu, pretraživački će program prikazati sliku koja je dodijeljena argumentu LOWSRC. Nakon što se učita manja sličica i sav potreban tekst, pretraživač će se vratiti do mjesta gdje je navedena SRC naredba, te će započeti s učitavanjem veće slike. Ovaj proces koji prolazi kroz dva koraka omogućava čitateljima da vide sliku koja izgleda bolje nakon što su stekli ideju o potpunom sadržaju stranice.

Korisnik će također moći kliknuti na neku od hipertekstualnih veza na stranici bez čekanja da se sve učita, ali će svejedno znati što je bilo prikazano na slikama.

Pogodnost ovog Netscape proširenja je i u tome što pretraživački programi koji ga ne prepoznaju neće stvarati nikakve probleme, već će automatski učitati veću inačicu slike. U nastavku su dana tri zanimljiva trika koja možete postići upotrebom argumenta LOWSRC.

- **Trik#1:** Kao LOWSRC sliku upotrijebite crno-bijelu kopiju originalne obojene slike koju ste dodijelili SRC argumentu. Neka formati obiju slika budu jednaki: Time ćete postići zanimljiv efekt osvjetljivanja.
- **Trik#2:** Definirajte LOWSRC sliku kao prvi korak u animaciji koja se sastoji od dvije slike. LOWSRC slika, na primjer, možete prikazivati šalicu čaja, a SRC slika tu istu šalicu iz koje izlazi para. Tijekom učitavanja stranice, najprije će se na ekranu prikazati šalica, a potom i para koja iz nje izlazi. Time ćete u svojoj stranice unijeti dozu dinamike, bez pretjerano velikog truda.
- **Trik#3:** Definirajte LOWSRC sliku kao pozadinsku sliku, naprimjer, kao neku teksturu ili gradacijski prijelaz boja. SRC sliku kreirajte tako da bude manjeg formata od LOWSRC slike. Kada se SRC datoteka učitava, LOWSRC datoteka će izgledati kao pozadinski okvir kojim je obrubljena manja slika.

10.2. Korištenje manjih datoteka

Slike manjeg formata brže se učitavaju u pretraživački program. Da biste što uspješnije smanjili datoteku unutar koje je pohranjena vaša slika, razmislite o upotrebi JPEG formata. JPEG kodiranje sažimlje sliku na jednu trećinu (atkada i na jednu petinu) ukupne veličine originalne GIF ili BMP slike.

Prije prevođenja svih velikih slika u JPEG format morate razumjeti da prilikom sažimanja JPEG odbacuje dijelove koje smatra manje važnim za kvalitetni prikaz slike. Što je veći stupanj sažimanja, više ćete izgubiti na kvaliteti slike, što znači da će vaša fotografija, crtež ili nešto treće najvjerojatnije izgubiti na broju boja ili će se negdje pojaviti tačkasti prijelazi. Ipak, uz ne prevelike stupnjeve sažimanja, JPEG slike gledane prostim okom ne razlikuju se od GIF ili BMP slika, a datoteke u kojoj su pohranjene znatno su manje.

Da biste još više smanjili veličinu datoteke u kojoj je pohranjena slika, ograničite broj boja koji se koristi. Svaka slika koju kreirate posjeduje paletu koja se koristi u zapisu datoteke. GIF format podržava palete do 256 boja, dok JPEG zapisi mogu raditi s više od 16 miliona različitih nijansi svih mogućih boja.

Odbacivanjem dijelova palete za boje koje se ne pojavljuju na slici, smanjit ćete veličinu datoteke za onaj dio koji je bio namijenjen tom dijelu palete. Ograničavanjem količine boja za neku sliku (64 do 100 za GIF datoteke i 1 do 2000 za JPEG datoteke), prilično ćete smanjiti čitav zapis. Programi kakvi su PhotoShop, Corel PhotoPaint, Paint Shop Pro, i slični, omogućavaju vam kontroliranje palete boja kod vaših grafičkih datoteka.

10.3. Prozirne i isprepletene GIF slike

Budući da jezik HTML nudi samo vrlo ograničene mogućnosti manipuliranja i rukovanja grafikom, potrebno je neke naprednije prilagodbe izvršiti prije no što se slike objave na Webu. Kreiranje isprepletenih (interlaced GIF slike) i prozirnih GIF slika zahtijeva od vas da upotrijebite neke posebne programe.

Kada radi s isprepletenim slikama, pretraživački program prikazuje sliku kroz više koraka, kako bi korisniku dao naznaku o tome što je prikazano u najkraćem mogućem vremenu. Za vrijeme prikazivanja slike korisnik stječe dojam da je slika horizontalno isprepletana, odnosno da su vodoravne trake slike preklapljene jedna preko druge, te da u svakim korakom sve više i više dolaze na svoje pravo mjesto.

Isprepletene slike korisniku se također mogu učiniti kao da su pohranjene u malom formatu, te potom "na silu" razvučene na veće područje, no taj se efekt svakim korakom prikazivanja sve više gubi. Situacija je slična fokusiranju slike pri radu s fotoaparatom: najprije ćete gledajući kroz aparat vidjeti mutnu sliku, no kada se ona ručno ili automatski izoštri, bit će potpuno jasna.

Prozirne slike mogu se pohranjivati u GIF89a format zapisa. Stari format, GIF87a, ne podržava rad s prozirnim slikama. Da biste jednu takvu sliku konstruirali, jednu od boja iz palete morate proglasiti pozadinskom, pa će ona u GIF89a datoteci odigrati ulogu prozirnog kanala. Kada pretraživački program učitava takvu datoteku, prozirni dijelovi slike bit će zamijenjeni s onim što se nalazilo u pozadini (najčešće je to neka pozadinska boja, odnosno uzorak s kojim ste ispunili stranicu). Upotrebom prozirnih slika postiže se efekt "plutajuće" grafike koja izgleda kao da se nalazi iznad pozadine, a ne kao uobičajeni pravokutnik s okvirom. Razliku između običnih i prozirnih slika pogledajte na slici 78.



Slika 78: Kroz sve dijelove slike koji su pokriveni prozirnom bojom vidi se pozadina. Zbog toga morate biti pažljivi koju ćete boju odabrati

11. CGI programiranje

Jedna od najsnažnijih mogućnosti World Wide Weba je sposobnost kreiranja dinamičkih dokumenata pomoću izvršnih programa, poznatih pod imenom CGI skripte (CGI je skraćenica od Common Gateway Interface). Te skripte mogu biti vrlo jednostavne, poput onih koje ispisuju razne poruke slučajnim odabirom, ali i prilično složene, pružajući korisniku mogućnost pretraživanja baza podataka ili rada s pristupnicama za druge Internetske servise kakav je, naprimjer, E-mail. Unatoč njihovom imenu, skripte se mogu pisati u bilo kojem programskom jeziku. Neke su doista prave skripte, napisane u interpreterskim jezicima poput Perla, dok su druge napisane u kompajlerskim jezicima kakav je C.

11.1. Upoznavanje sa skriptama

CGI skripte su programi koje pokreće server na zahtjev klijenta. Kada korisnik zatraži URL adresu koja pokazuje na neka skripta, server će je izvršiti. Svi izlazni podaci koje skripta proizvodi prikazat će se u prozoru pretraživačkog programa. URL adrese kojima se pozivaju skripte izgledaju kao i svake druge URL adrese s HTTP protokolom. Na serveru tvrtke NCSA, naprimjer, instalirana je skripta fortune koja je označena URL adresom:

<http://hoohoo.ncsa.uiuc.edu/cgi-bin/fortune>.

Kada serveru stigne zahtjev za ovom URL adresom, on će izvršiti UNIX naredbu fortune. Ta naredba vraća drugačiju duhovitu dosjetku svaki puta kada je pozovete.

Kako server zna da skriptu treba izvršiti, a ne prikazati njen sadržaj? Postoje dvije metode utvrđivanja. Najviše se koristi ona kojom se određeni direktorij na serveru posebno obilježi kao direktorij za skripte. Server sve datoteke u tom direktoriju i svim njegovim poddirektorijima smatra izvršnim skriptama, a ne običnim dokumentima. Čak je moguće kreirati nekoliko direktorija za skripte, navodeći ih u konfiguracijskoj datoteci Web servera. Web serveri najčešće su konfigurirani tako da im se direktorij za skripte nalazi odmah u korijenskom direktoriju servera, te da mu je dodijeljeno ime cgi-bin.

Drugi način utvrđivanja radi li se o običnom dokumentu ili izvršnom programu je upotreba tačno određenih ekstenzija za skripte, najčešće .cgi. Kada su skripte obilježene ovim postupkom, nije ih potrebno smještati u tačno određene direktorije, već se one mogu nalaziti bilo gdje u serverevu stablu direktorija. Ako radite s Web serverom koji podržava obje spomenute metode, moći ćete ih kombinirati, postavljajući neke programe u direktorij za skripte, a neke bilo gdje u stablu, dodjeljujući im posebnu ekstenziju.

Najčešće nije potrebno obavljati nikakav drugi konfiguracijski posao oko skripti. Jednostavno se uvjerite da su izvršive i da se ili nalaze u posebnom direktoriju, ili da im je dodijeljena posebna ekstenzija. Potom kreirajte hipertekstualnu vezu koja poziva skriptu, ili povežite svoj dokument sa skriptom na neki drugi način, kao, na

primjer, pomoću obrasca. Ako vam se učini da se u direktoriju za skripte nalazi previše programa i da nastaje prevelika gužva, slobodno kreirajte poddirektorije kako biste uveli malo reda. Server će svaku URL adresu koja sadrži ime cgi-bin shvaćati kao poziv za izvršavanje skripte, čak i ako je ona zakopana duboko u stablu poddirektorija.

Da biste mogli pisati skripte, najprije morate naučiti na kojem principu rade. Kada HTTP server primi zahtjev s URL adresom vaše skripte, on kreira skup globalnih varijabli koje sadrže korisne informacije o serveru, pretraživačkom programu i o trenutnom zahtjevu. Potom izvršava skriptu, "hvata" sve što ona izbacuje kao izlazne podatke, te uhvaćeni plijen šalje pretraživaču na prikazivanje.

Vaša skripta je zapravo dokument kao i svaki drugi, pa tako posjeduje sve slabosti i vrline pravih dokumenata. Ona može komunicirati izravno s korisnikom kao što to mogu i najobičniji računarski programi. Kako bi se ostvario ovakav dvosmjerni razgovor, morat ćete svojem korisniku poslati dokument unutar kojega je omogućen unos nekakvih podataka namijenjenih obrađivanju pomoću skripti. Kako je HTTP prilično jednostavan protokol, skripta ne može znati da li je isti korisnik već koji put pozivao tu skriptu, pa se pokreće na isti način svaki put ispočetka kada serveru stigne zahtjev s njenom URL adresom. Skripta se izvršava sve dok se ne proizvede dokument koji će se korisniku poslati kao rezultat. Sva komunikacija koja se izvršava između klijenta i CGI programa mora se stoga obavljati preko standardnog I/O sistema, dakle pomoću uobičajenih naredbi za kontroliranje standardnog ulaza i izlaza podataka. Ne postoji način na koji možete kontrolirati izgled korisničkog ekrana, kreiranje okvira za dijalog, dugmadi i drugih lijepih stvari kojima obiluju suvremene aplikacije koje rade pod grafičkim operativnim sistemima. Sve što vaš program mora proizvesti su podaci u običnom tekstualnom obliku.

Kao i svaki dokument na Webu, tako i dokument koji će proizvesti vašu skriptu mora posjedovati pridruženi MIME tip. Najčešće su dokumenti koje proizvodite HTML dokumenti tipa text/html, no dozvoljena je upotreba svih tipova, uključivši grafiku i zvuk. Kako biste rekli pretraživačkom programu koji MIME tip da očekuje, vaš će skript morati ispisati jedan redak koji sačinjava standardno HTTP zaglavlje, poput:

```
Content-type: neki/mime-tip
```

Zaglavlje koje šalje vaša skripta može sadržavati i druga polja, no ne obavezno, budući da će server umjesto vas automatski popuniti sva neophodna polja. Iza zaglavlja mora se ostaviti jedan prazan redak, te tada započeti s ispisivanjem željenih podataka.

Katkada nećete htjeti da vaš CGI program kreira nove dokumente za prikazivanje na klijentskom računaru, već ćete, naprimjer, htjeti da iz nekog skupa URL adresa odabere onu pravu i učita je u pretraživački program vašeg korisnika. Na tom principu rade serverske slikovne mape. Također, možda ćete htjeti prikazati jedan dokument spoljnim čitateljima, a neki drugi dokument čitateljima koji se nalaze u

lokalnoj mreži. U tom slučaju treba umjesto polja Content-type u zaglavlju ispisati polje Location koje pokazuje na željenu URL adresu, kao u ovom primjeru:

```
Location: http://neki.računar/neka/URL/adresa
```

Kada server naiđe na ovakvo polje u zaglavlju skripte, automatski će pretraživačkom programu poslati naredbu za preusmjeravanje klijenta na novu URL adresu čije će učitavanje odmah započeti. Ako se u ovom slučaju ne proizvodi nikakav dokument, sve što vaš program mora ispisati poslije ovog zaglavlja je jedan prazan redak. Prema tome, zahtjevi za CGI skripte su jednostavni:

- Skripta mora biti izvršiva i instaliran na HTTP serveru na standardni način, unutar direktorija cgi-bin (odnosno nekog drugog direktorija sa skriptama), ili negdje drugdje uz ekstenziju.cgi.
- Kada se izvršava, on mora ispisati HTTP zaglavlje iza kojega slijedi jedan prazan redak. To zaglavlje mora sadržavati polje Content-type ili polje Location.
- Potom mora ispisati datoteku namijenjenu prikazivanju.

Najjednostavnija skripta koju možemo napisati je ona koja od korisnika ne zahtjeva nikakav unos. U nastavku je dan ispis skripte koju smo nazvali malena.pl, a koja koristi jezik Perl za prikazivanje poruke u pretraživačkom programu. Perl je izabran za ispise primjera u ovom poglavlju zbog toga što je vrlo popularan na Webu i što je dostupan besplatno za sve poznate operativne sisteme.

Ako su vam poznate osnove programiranja, ali niste upoznati s Perlom, pročitajte ovih nekoliko natuknica koje će vam pomoći da razumijete o čemu se radi u ovom primjeru.

- Sintaksa Perla slična je sintaksi jezika C. Još je sličnija sintaksi raznih skriptnih jezika za programiranje UNIX ljsuke.
- Postoje samo tri osnovne strukture podataka: brojevi, nizovi (indeksirani cijelim brojevima) i asocijativni nizovi (indeksirani stringovima). Te se tri strukture razlikuju po prefiksu koji se piše uz varijable:

\$varijabla – skalar (broj)

@varijabla – niz

%varijabla – asocijativni niz

- Pored skalara, nizova i asocijativnih nizova, Perl ne prepoznaje tipove podataka. Stringovi, cjelobrojne vrijednosti, decimalni brojevi i drugi tipovi pripadaju među tri podržane strukture, ovisno o kontekstu. Naprimjer, potpuno je ispravan sljedeći izraz:

```
$suma = "1" + 2 + 3.01;
```

- Kada se koristi kao indeks niza, prefiks varijable označava tip elementa koji je pohranjen u nizu. Uglate zagrade koriste se za obične nizove, a vitičaste za asocijativne nizove.

\$niz [0] – prvi element niza @niz

\$niz [1] – drugi element niza @niz

\$niz {'jurica'} – vrijednost asocijativnog niza @niz na indeksu 'jurica'

\$niz {jurica}- isto kao i gore

- Uobičajeni izraz \$niz = <> koristi se za uzimanje jednog retka teksta sa standardnog ulaza. Navođenje samo oznaka <> pohranit će redak teksta sa standardnog ulaza na varijablu \$_.
- Asocijativni niz %ENV sadrži trenutne globalne varijable sistema kako bi im se moglo pristupiti. Primjeri članova niza bili bi: \$ENV {'PATH'} i \$ENV {'HOME'}.
- Izraz \$varijabla=~ /podudaranje_po_uzorku/ koristi se za operacije koje se moraju podudarati po nekom zadanom uzorku. On se, naprimjer, može koristiti da se provjeri sadrži li varijabla \$varijabla određene vrijednosti koje su navedene iza tilde (~) kao uzorak.
- Prefiks & koristi se za pozivanje potprograma. Na primjer: &sumiraj (\$marke, \$lire).
- Poziv system ("Neka naredba") izvršit će određenu naredbu u UNIX ljusci, ili ljusci nekog drugog operativnog sistema. Poziv eval ("neki Perl izraz") izračunat će navedeni izraz i vratiti rezultat.
- Perl varijablama koje su navedene unutar teksta obuhvaćenog dvostrukim znacima navoda biti će ispisana vrijednost. Ako se u tekstu obuhvaćenom znacima navoda nalazi simbol \n, na tom će mjestu ispis prijeći u novi redak. Varijable unutar jednostrukih znakova navoda se ignoriraju. Evo primjera:

```
$varijabla = 'jurica';
```

```
print "Moje ime je $varijabla.";
```

ispis je: Moje ime je jurica.

```
print 'Moje ime je $varijabla.';
```

ispis je: Moje ime je \$varijabla.

- Obrnuti jednostruki znaci navoda, poput 'date', uzrokovat će izvršavanje navedenog programa (u ovom slučaju naredbe date). Rezultati koje će program poslati na standardni izlaz bit će vraćene skripti. Ovo se najčešće koristi u kombinaciji s naredbom chop() kako bi se izbrisao znak za prijelaz teksta u novi redak s kraja izlaznih podataka:

```
chop($datum='date'; # $datum postaje "Mon September 18 2006"
```

- Još malo sintakse:

```
Print <<END
```

Ovo je tekst obuhvaćen dvostrukim znacima navoda koji će biti ispisan zajedno s prijelazima teksta u novi redak i ispisima vrijednosti svih varijabli...

```
END
```

Sav tekst između prvog retka i riječi END smatra se tekstom obuhvaćenim dvostrukim navodnim znacima. Riječ END nije ključna, pa možete koristiti i neku drugu, no među programerima je vrlo uobičajena. Gornji kod istovjetan je sljedećoj naredbi:

```
Print "Ovo je tekst obuhvaćen dvostrukim znacima navoda koji će biti ispisan zajedno s prijelazima teksta u novi red i ispisima vrijednosti svih varijabli...";
```

Naravno, nije teško zaključiti da se gornji primjer puno lakše čita.

Primjer skripte:

```
#!/usr/local/bin/perl
# Datoteka: malena.pl
print <<END
Content-type: text/plain

Maleni dokument:

Ovo je najobičniji tekst. Jednostavan, bez
značenja!

END
```

Ovaj program neće napraviti puno toga. On se sastoji od samo jedne print() naredbe koja na standardni izlaz ispisuje HTTP zaglavlje dokumenta iza čega slijedi jednostavan tekst. Kako biste instalirali ovu skriptu i isprobali je u živo, prilagodite putanju do Perl interpretera u prvom retku koda kako bi odgovarala konfiguraciji vašeg sistema, napravite skript izvršnim i instalirajte ga u direktorij za skripte. Pozovite ga iz svojeg pretraživačkog programa pomoću URL adrese <http://vaše.računar/cgi-bin/malena.pl>. Vaš će pretraživač ispisati sljedeći tekst:

```
Maleni dokument:

Ovo je najobičniji tekst. Jednostavan, bez
značenja!
```

Najvažniji dio koji morate uočiti u ovom primjeru je zaglavlje kojim se deklarira text/plain kao MINE tip dokumenta. Bez prisustva ovog retka i praznog retka koji slijedi, server će prijaviti pogrešku.

Primjer s HTML naredbama:

```
#!/usr/local/bin/perl
# Datoteka: malena.pl
print <<END;
Content-type: text/html
<HTML>
<HEAD>
    <TITLE>Maleni dokument</TITLE>
</HEAD>
<BODY>
    <H1>Ovo je najobičniji tekst.</H1>
    Jednostavan, bez značenja! <BR>
    A zatim se pređe u novi red...
    <P><HR><P>
    Povratak na <A HREF="/">početnu stranicu</A>
</BODY>
</HTML>
END
```

Sasvim je jasno što će biti prikazano u pretraživačkom programu. Uočite, međutim, da smo promijenili redak sa zaglavljem Content-type u novi tip dokumenta. Ne postoje nikakva ograničenja za upotrebu HTML elemenata unutar CGI skripata. Hipertekstualne veze, slike, multimedija, obrasci i sve ostalo radi na potpuno jednak način kao da se radi o običnom statičkom HTML dokumentu. Relativne URL adrese relativne su u odnosu na direktorij u kojem se nalazi CGI program.

Sada ćemo skriptu učiniti nešto zanimljivijom pretvarajući je u pristupnik za UNIX naredbu cal kojom se ispisiuje kalendar za godinu koja se navede kao parametar. U nastavku je dan kod ovog programa.

```
#!/usr/local/bin/perl
# Datoteka: kalendar1.pl
$CAL='/usr/bin/cal';
$DATE='/bin/date';
#Otkrivanje trenutne godine pomoću UNIX naredbe "date"
chop ($godina='$DATE +%Y');
#Kreiranje sadržaja kalendara upotrebom naredbe "cal"
```

```
chop ($kalendar='$CAL $godina');
#Ispisivanje svega u obliku HTML stranice
print <<END;
Content-type: text/html
<HTML>
<HEAD>
    <TITLE>Kalendar za godinu $godina</TITLE>
</HEAD>
<BODY>
    <H1>Kalendar za godinu $godina</H1>
    <PRE>
        $kalendar
    </PRE>
    <HR>
    <A HREF="/">Početna stranica</A>
</BODY>
</HTML>
END
```

U programu smo dva puta pozivali UNIX naredbe. Prvi poziv, naredba `date`, vratit će trenutnu godinu. Drugi poziv koristi godinu dobivenu naredbom `date` za izvršavanje naredbe `cal`. Tekst koji je nastao upotrebom naredbi `date` i `cal` potom se ubacuje na odgovarajuća mjesta u HTML dokumentu i ispisuje. Kako naredbe `date` i `cal` mogu biti smještene u drugačijim direktorijima na različitim sistemima, pune putanje tih programa pohranili smo u Perl varijable pri vrhu skripta kako biste ih kasnije lako mogli pronaći i promijeniti u odgovarajuće vrijednosti. Također ne smijemo zaboraviti ispis kalendara smjestiti unutar naredbe `<PRE>`, jer će se u suprotnom izgubiti razmaci i prelasci u novi redak koje je kreirala naredba `cal`.

12. JavaScript

World Wide Web u samim počecima nije imao riješen problem povezivanja Web stranica i drugih alata, poput baza podataka. Takve stvari mogao je riješiti CGI. No, vrlo brzo pojavila se potreba da dokumenti obavljani na Webu budu dinamički promjenljivi, odnosno da prikazuju najsvježije informacije o nekoj temi ili informacije koje se često mijenjaju, a da administrator Weba ne mora u to ulagati previše truda. Osim toga, razvoj Weba išao je u pravcu stvaranja što boljih sučelja, tako da je ubrzo postalo gotovo neophodno imati na svojoj Web stranici ikone, dugmad, padajuće popise s opcijama, radio dugmad i ostale elemente modernih sučelja. Budući da se CGI programi pozivaju na serveru, oni nisu mogli ponuditi način da se ove posljednje zamisli riješe. Ostali nedostaci CGI programiranja bili su:

- potreba za učenjem novog programskog jezika, najčešće Perla;
- nemogućnost da CGI skripta vrati pretraživačkom programu bilo šta drugo osim čistog teksta ili neke datoteke;
- sporost komunikacije s korisnikom, jer pretraživač mora ponovno kontaktirati servera da bi poslao podatke koje je korisnik upisao, a tek nakon toga poziva se CGI skripta i ona šalje rezultate;
- izvršavanje CGI skripti na serveru nepotrebno opterećuje server dodatnim zahtjevima, jer se za svaku CGI skriptu mora pokrenuti dodatni proces;
- relativno velik sigurnosni rizik, jer neka skripta s pogreškama može stvoriti velike probleme;
- CGI skripta može teoretski dobiti vrlo veliku razinu "dopuštenja" što smije raditi na serverskom računaru, zbog toga što se serverski program najčešće izvršava s vrlo visokim korisničkim statusom, što sve zajedno omogućava hakerima da preko CGI-a pronađu način da provale u vaše računare;
- Nemogućnost stvaranja korisničkih sučelja i općenito vrlo skučen popis stvari koje se mogu riješiti pomoću CGI-a

Zanimljivo je kako je mnogo vremena trebalo proći dok se nisu pojavila nova i bolja rješenja. Budući da CGI s tehničke strane ne predstavlja ništa posebno (radi se o najobičnijem pozivanju UNIX skripti i preusmjeravanju njihovog ispisa u pretraživački program korisnika), tjera na razmišljanje činjenica da je Web čak svojih prvih nekoliko godina bio ograničen isključivo na CGI.

Zatim je došla Java i revolucija je krenula. S Javom, autori Web stranica mogli su uz minimalno programiranje stvarati efektne programe, a sam programski jezik na sebe je preuzimao sve brige o sigurnosti, multimediji, prikazu grafike i sviranju zvučnih datoteka, te korištenju Internet protokola za komuniciranje s drugim računarima. Međutim, pojavom Java Web još uvijek nije dobio alat koji bi omogućavao bolju interaktivnost na Web stranicama, a da napisani programi budu

kratki i jednostavni. U Javi možete stvoriti animaciju ili nacrtati cijelo korisničko sučelje za svoju Web stranicu, ali trebalo je napraviti neko rješenje koje će ipak biti jednostavnije za širu primjenu. Tako je rupa između CGI-a i Jave i dalje postojala, ali je pojava Jave pokazala da će se sve više poslova moći obavljati na klijentima, a ne na serveru.

Rješenje su bili skriptni jezici. Zamisao je bila jednostavna: trebalo je napraviti jednostavan programski jezik čiji bi se "izvorni kod" ugrađivao neposredno u HTML stranicu (za razliku od Java aplikacija što se samo pozivaju iz HTML dokumenta i CGI programa koji se moraju pozvati s servera i izvršavati na njemu). Jezik bi trebao biti interpretiran, jer nikakvo kompiliranje ne može dolaziti u obzir već i zbog same činjenice da se klijentu prenosi izvorni kod, pa bi cijeli proces kompiliranja na računaru korisnika predugo trajao. Također, za takve poslove neprihvatljivo je rješenje koje donosi Java, jer je njen bajt-kod isključivo binarna datoteka koja se ne može ugraditi u HTML dokument. Posljednji - i najvažniji - zahtjev koji se postavlja pred skriptni jezik je mogućnost da on ipak obavi i mnogo složenije poslove od onoga što bi se od njega naizgled očekivalo: stvaranje korisničkih sučelja, ispis poruka, komunikacija s serverom i crtanje grafike sako su neka od svojstava koje mora imati svaki dobar skriptni jezik.

Prvi su na razvoju skriptnog programskog jezika počeli raditi u tvrtki Netscape Communications. Iako su oni samostalno počeli razvijati skriptni jezik (s početnim imenom Live Script), uskoro su se udružili sa Sunom i dogovorili se da JavaScript bude nalik Javi. I danas je tako: JavaScript je sličan Javi, ali je mnogo jednostavniji za korištenje u nekim svakodnevnim situacijama. Sve što vam treba za prikaz ili programiranje JavaScript skriptata je pretraživački program koji podržava JavaScript, a danas su to Netscape Navigator i Microsoft Internet Explorer, tako da, kao autor Web stranica, možete računati da 99 % korisnika može vidjeti JavaScript na djelu.

12.1. Java i JavaScript

Nekoliko je glavnih razlika između Jave i JavaScripta. Već iz ovih razlika moći ćete zaključiti u kojim je primjenama dovoljno koristiti JavaScript, a kada trebate uzeti "pravu stvar" i prihvatiti se pisanja programa u Javi.

Prvi kriterij po kojem treba usporediti oba jezika je lakoća programiranja. JavaScript je relativno jednostavan za korištenje. Mnogi će vam reći da je Java komplicirana i da je mogu savladati samo iskusni programeri, što nije nimalo tačno. Međutim, za Javu je ipak potrebno veće znanje nego za JavaScript, ali to je neizbježna posljedica svojstva samog programskog jezika i onoga što on može ponuditi. Java je potpuni programski jezik, sa svojstvima kao što su višenitnost i hvatanje pogrešaka, dok JavaScript to nema.

Drugo po čemu se obično uspoređuju ta dva jezika su alati za razvoj aplikacija. Za Javu vam treba Sunov Java Development Kit ili neki od ostalih razvojnih alata koje

smo spomenuli u prethodnom poglavlju; za pisanje JavaScript skripti bit će dovoljan pretraživački program koji ih može pokrenuti i izvršavati.

Java aplikacije pozivaju se iz Web stranice pomoću naredbe <APPLET>, putem koje se programu prosljeđuju i svi parametri potrebni za njegov rad. JavaScript skripte ugrađuju se direktno u HTML kod stranice.

Brzina izvršavanja programa danas ide uglavnom u prilog JavaScriptu. Iako se mnogo radi na tome da se Java ubrza, programi pisani u Javi još se uvijek izvršavaju relativno sporo. To je djelomično posljedica potrebe da se bajt-kod kompilira prije izvršavanja. S druge strane, JavaScript skripte obično se izvršavaju vrlo brzo.

Java je objektno orijentirani jezik. Programeri mogu iznova koristiti već napisane dijelove programa - objekte - a dodatno programirati samo onoliko koliko je potrebno da se uskladi rad tih objekata. S druge strane, JavaScript je jezik zasnovan na objektima, što je sasvim druga priča. JavaScript programi mogu pozivati objekte koji su ugrađeni u pretraživački program koji izvršava JavaScript, ali ne postoji mogućnost pravog rada s objektima, niti postoje klase i nasljeđivanja.

Druga značajna razlika što se tiče koncepcije samog jezika je specifikacija. Specifikacija programskog jezika je skup pravila kojih se programeri moraju pridržavati da bi napisali sintaktički i leksički ispravan program. Java je tu, kao i svi ostali pravi programski jezici, vrlo stroga, ali to je nešto na što su programeri navikli. Varijable moraju biti deklarirane prije korištenja, a i svi ostali objekti moraju prethodno biti stvoreni. JavaScript je ovdje sličniji Perlu, koji se najčešće koristi u CGI programiranju, jer nije toliko rigorozan po pitanju načina stvaranja i korištenja varijabli. Neki će reći da je to argument u prilog JavaScriptu, jer zašto na početku programa razmišljati o varijablama koje morate deklarirati, kad je prirodnije nove varijable uključivati u program "u letu"? Pa, ovaj drugi način da uvijek je podložen pogreškama, koje ne moraju biti samo do programera, već se može raditi i o nedostacima izvršnog sistema za JavaScript. Iako je početnicima lakše "zaboraviti" na potrebu deklariranja varijabli, nama se ta ideja ne dopada. Većina programera će s JavaScripta prijeći na Javu, a starih se navika teško odreći.

12.2. Kada JavaScript? Kada Java? Kada CGI?

JavaScript ćete koristiti za sve ono što ste željeli napisati u CGI-u, ali niste mogli ili nije bilo elegantno. JavaScript ćete koristiti za jednostavne radnje čije bi programiranje u Javi oduzelo previše vremena. To su, naprimjer, ispisivanje trenutnog datuma ili vremena na Web stranici, ispisivanje datuma zadnje promjene, ispis poruka u statusnom retku pretraživačkog programa, otvaranje novih prozora u pretraživaču i prikaz novih HTML dokumenata, provjeru ispravnosti podataka koji su upisani u obrasce, pa čak i ograničeno pretraživanje vrlo jednostavnih baza podataka.

Ako je potrebno napraviti efektanu animaciju, multimedijску prezentaciju ili interaktivnu aplikaciju na Webu koja stalno komunicira s korisnikom, Java je jedini dobar odabir. Ako program mora obavljati složena preračunavanja, s Javom možete postići da se taj proces obavlja u pozadini zahvaljujući Javinom ugrađenom sistemu višenitnosti; a ako je potrebno iz vlastitog programa intenzivno koristiti druge mrežne protokole (poput pristupanja FTP serverima ili drugim uslugama), Java već sadrži sve što vam treba. Dakle, za bilo pisanje bilo kojeg programa koji bi trebao imati dulji životni vijek i koji se neće izvršiti samo jedan jedini put kada netko posjeti Web stranicu, trebali biste odabrati Javu.

CGI-u će, s obzirom na razvoj situacije u svijetu skriptnih jezika, ostati vrlo malo prostora. Praktički jedine radnje koje ne možete izvesti sa skriptnim jezicima su pretraživanje baza podataka. Naime, baze podataka najčešće se nalaze na serveru, a Java i JavaScript im tamo ne mogu pristupati, pa CGI ostaje jedino rješenje.

12.3. JavaScript i HTML

Svi skriptni programa i ugrađuju se u HTML dokumente pomoću naredbe `<SCRIPT>`. Unutar tog bloka potrebno je jednostavno napisati cijeli kod JavaScript programa i to je sve. Dakle, da bi pretraživački program prepoznao da u dokumentu postoji JavaScript, trebate koristiti par naredbi:

```
<SCRIPT>
...
</SCRIPT>
```

Naravno, to je najjednostavnija varijanta te naredbe, ali se u tom obliku ne može koristiti jer je potrebno naznačiti i o kojem se skriptnom jeziku radi. Naprimjer, postoji i VBScript, pa pretraživač mora znati hoće li pokušati intepretirati JavaScript ili VBScript. U slučaju JavaScripta, naredba `<SCRIPT>` treba glasiti:

```
<SCRIPT LANGUAGE="JavaScript">
...
</SCRIPT>
```

Trenutačno se kao argumenti za parametar `LANGUAGE` mogu koristiti samo imena `!JavaScript`!, `VBScript` i `LiveScript`. LiveScript je ostatak iz prošlosti, pa preostaju samo prva dva skriptna jezika. Međutim, kako se stvari na Webu vrlo brzo razvijaju, moglo bi se dogoditi da će se svi skripte, sve Java aplikacije i sve ActiveX kontrole ubuduće u HTML dokumente ugrađivati uporabom jedinstvene naredbe, `<OBJECT>`, koja se i danas primjenjuje za povezivanje ActiveX kontrola i katkada plug-in dodataka ugrađenih u pretraživač. I u interesu samih tvrtki je da se konačno o tome dogovore, jer nema smisla da postoje tri različita načina ugradnje vrlo sličnih objekata u stranicu.

Parametar SRC je opcionalan u naredbi <SCRIPT>. Možete, naime, odrediti da se vaša skripta nalazi na nekom drugom računaru ili u nekom drugom direktoriju. Tada se sama skripta ne uključuje u HTML dokument, ali mora postojati na URL adresi koja se navede. Sintaksa je:

```
<SCRIPT LANGUAGE="JavaScript"
SRC="http://www.unze.ba/js/script.js">

...

</SCRIPT>
```

Sada ste spremni napisati i svoj prvi program u JavaScriptu. Napraviti ćemo jednostavnu Web stranicu koja sadrži skriptu koja će samo ispisati poruku:

```
<HTML>

<HEAD>

<SCRIPT LANGUAGE="JavaScript">

document.write ("<H1>Moja prva skript</H1">)

</SCRIPT>

<HEAD>

<BODY>

<H2>Ovo je običan HTML...</H2>

</BODY>

</HTML>
```

Ipak, ovdje se pojavljuje jedan problem. Gornji će primjer raditi u svakom pretraživaču koji zna izvršavati JavaScript (Netscape, Internet Explorer), ali korisnici ostalih pretraživača (npr. Lynxa ili Mosaica) ne samo da neće moći vidjeti skript na djelu, već će njihov pretraživač zaključivati po logici "ja nemam pojma čemu služi ovaj <SCRIPT>, ću to ispisati na ekran", pa će ispisati cijeli kod skripta kao i ostala Web stranice. To nije ono što želite.

Zbog toga je dogovoreno da se sama skripta obuhvati još i HTML naredbama za komentar: <!-- i -->. Međutim, prije ove završne naredbe za komentar treba postaviti još i oznaku //, kako bi sam JavaScript smatrao da je ostatak retka komentar, jer će inače on prijaviti pogrešku da ne razumije naredbu. Pretraživači koji podržavaju JavaScript zanemarit će oznake komentara i nastaviti će s izvršavanjem programa, dok će ostali pomisliti da se ovdje zaista radi o komentaru i potražiti će prvu sljedeću HTML naredbu koju poznaju. Naš gornji program nakon učinjenih promjena postaje:

```
<HTML>

<HEAD>

<SCRIPT LANGUAGE="JavaScript">
```

```
<!-- Početak komentara kojim "skrivamo" skript
document.write ("<H1>Moja prva skripta</H1>")
//-->
</SCRIPT>

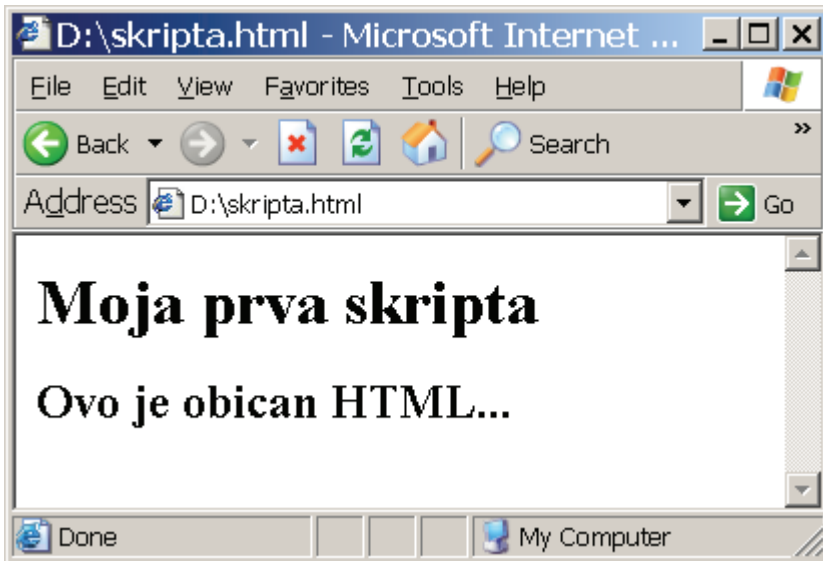
<HEAD>

<BODY>

<H2>Ovo je običan HTML...</H2>

</BODY>

</HTML>
```



Slika 79. Rezultat izvršavanja skripte u prozoru pretraživačkog programa

12.4. Svojstva JavaScript jezika

12.4.1. Varijable i automatsko pretvaranje tipova

Varijable u JavaScriptu mogu se zvati kako god želite, ali postoji ograničenje kao i u Javi, a to je da ne smiju počinjati brojevima. Dakle, prvi znak smije biti samo slovo ili oznaka podvlake (-). Ne postoji ograničenje duljine imena varijable.

Za deklariranje varijabli u JavaScriptu koristi se naredba `var`. JavaScript za početak ne zahtijeva da varijable budu deklarirane na početku programa, a osim toga u JavaScriptu ne postoje tipovi podataka. Potpuno jednako kao i Perl, JavaScript obavlja dinamičku pretvorbu podataka tijekom izvršavanja skripte, pa programeri mogu pisati skripte bez razbijanja glave o tome smiju li pretvoriti jedan tip podataka u drugi, a da pritom ne izgube na preciznosti. Osim toga, JavaScript će za

vas sam odrediti kojeg tipa treba biti pojedina varijabla onog trenutka kada je uvedete u program.

Na primjer, ako u svom programu koristite dvije varijable koje su prvi puta spomenute s:

```
Varijabla1="15"
```

```
Varijabla2=5
```

Tada će varijabla1 biti string (zbog toga što je njen sadržaj obuhvaćen navodnicima), dok će varijabla2 biti cjelobrojnog tipa.

Ako kasnije u programu napišemo, recimo:

```
X = varijabla1 + varijabla2
```

```
Y = varijabla2 + varijabla1
```

Kako će se računati vrijednosti varijabli x i y? Za početak, budući da varijable sasvim očito nisu jednakog tipa, JavaScript mora obaviti pretvorbu. Pravilo za pretvaranje tipova podataka vrlo je jednostavno: pretvaranje se obavlja slijeva nadesno, odnosno onaj operand s desne strane operatora (običnim riječima, "ono desno od plusa") bit će pretvoren u tip operanda s lijeve strane operatora.

Tako će u prvom gornjem primjeru varijabla2 prvo biti pretvorena u string, a zatim će taj string biti "zbrojen" sa stringom pohranjenim u varijabli varijabla1. Budući da zbrajanje stringova rezultira njihovim spajanjem, nova vrijednost varijable x (koja je u međuvremenu također postala tip string) bit će "155".

U drugom slučaju, varijabla1 se iz stringa mora pretvoriti u cijeli broj i zatim pomnožiti s vrijednošću pohranjenom u varijabli varijabla2. Varijabla y bit će cjelobrojnog tipa, a sadržavat će rezultat množenja 5 sa 15, odnosno broj 75.

Naravno, ne možete konvertirati iz bilo čega u bilo šta. Ako se oslanjate na automatsko pretvaranje varijabli, morate dobro razmisliti o tome može li se neka pretvorba izvesti. Naprimjer, string se može pretvoriti u cijeli broj samo ako nije sadržavao niti jedno slovo, što je tek samo jedan od posebnih slučajeva stringova, budući da oni najčešće sadrže i druge znakove. U tom biste primjeru iz stringa prvo morali sami pobrisati ono što ne želite, a tek potom obaviti pretvorbu. Drugi problem je u nedefiniranim operacijama: da smo prvu od ranijih naredbi napisali kao `x = varijabla1 - varijabla2`, što bi se dogodilo? Varijabla varijabla2 očito se mora pretvoriti u string, ali operacija oduzimanja stringova nije definirana. U svim će takvim slučajevima doći do ispisivanja pogreške u prozoru pretraživačkog programa.

12.4.2. Pisanje vrijednosti i komentara

Vrijednosti se u JavaScriptu pišu jednako kao i u Javi. Stringovi se navode pod navodnicima, koji mogu biti i jednostruki (tzv. literali) i dvostruki. Primjeri stringova su "Znak" i 'Znak'.

Posebne znakove potrebno je unutar stringova navoditi kao prekidne nizove (escape sequeunce).

Tabela 6. Escape sekvence za JavaScript i za Javu

Prekidni niz (Escape sekvenca)	Značenje	Opis
\	novi red	nastavak u novom redu
\n	LF	prelazak u novi red (line feed)
\t	HT	tabulator
\b	BS	backspace
\r	CR	prelazak u novi red (carriage return)
\f	FF	prelazak na novu stranu (form feed)
\\	\	kosa crta
\'	'	apostrof
\"	"	navodnik
\aaa	0aaa	oktalni broj
\xaa	0xaa	heksadecimalni broj
\uaaaa	0xaaaa	kod znaka u Unicode skupu

Cijeli brojevi mogu se pisati u decimalnom, oktalanom ili heksadecimalnom obliku. Isto kao i u Javi, oktalni brojevi pišu se s vodećom nulom (045), a heksadecimalni uz prefiks 0x (0x2FC4). Brojevi s pomičnim zarezom mogu se naravno pisati i u eksponencijalnom obliku, npr. 2E-5.

Logičke varijable mogu poprimiti vrijednost TRUE ili FALSE.

Za upoređivanje vrijednosti na raspolaganju su uobičajeni operatori: jednakost (=), nejednakost(!=), veće od (>), manje od (<), manje ili jednako (<=) i veće ili jednako (>=).

Komentari se pišu između para oznaka /* i */, pri čemu se mogu protezati i kroz više redaka, ili se navode iza oznake //, pa se u tom slučaju komentarom smatra sve što se nalazi do kraja retka.

12.4.3. Svojstva objekata i nizovi

Već smo rekli da JavaScript ne podržava u potpunosti rad s objektima, njihovo stvaranje i uređivanje, ali se cijelo programiranje zapravo svodi na pozivanje postojećih objekata. To su objekti koji se već nalaze u interpreteru za JavaScript u vašem pretraživaču i koji obavljaju određene radnje. Objektima se pristupa tako da

se prvo navede njegovo ime, a zatim i njegovo svojstvo koje se želi promijeniti ili kojem se želi pristupiti. Naprimjer:

```
Knjiga.stranica=1050
```

će svojstvu stranica od objekta knjiga pridružiti navedenu vrijednost. Ako svojstvo stranica ne postoji u objektu knjiga, ono će mu biti dodato.

S nizovima se u JavaScriptu radi na sličan način kao što se pristupalo objektima. Razlog tome nije teško otkriti ako se malo promisli: budući da JavaScript na isti način čuva u memoriji nizove i objekte, možete pristupati nizovima na jednak način kao što pristupate objektima i obrnuto. Tako bismo gornju naredbu mogli pisati i kao:

```
Knjiga ["stranica"]=1050
```

Jer se nizovima općenito pristupa tako da se unutar uglatih zagrada navede indeks člana koji se želi promijeniti ili čija se vrijednost želi dobiti.

12.4.4. Kontrola tijeka programa

Naredba `if ... else` za postavljanje logičkih uvjeta ima istu sintaksu kao u Javi ili C++. Umjesto naredbe `if ... else` može se opet koristiti i skraćeno pisanje pomoću upitnika i dvotački.

```
if (logički izraz) {  
    ...  
}  
else {  
    ...  
}
```

Naredbe za stvaranje petlji, `for` i `while` također se pišu jednako kao u Javi ili C++.

```
for (izraz; logički izraz; izraz) {  
    ...  
}
```

Naredbe `while` i `continue` mogu se, kao i u Javi i C++, koristiti za iskakanje iz petlje i nastavljnje izvršavanja od neke tačke programa.

12.4.5. Operatori i njihovi prioriteti

Operatori koji su na raspolaganju identični su onima u Javi i C++, a njihovi su međusobni odnosi u prioritetima jednaki.

Tabela 6. Operatori za JavaScript i za Javu

Prioritet	Operator	Opis
1	()	zagrada se koriste za promjenu prioriteta
2	++	inkrementiranje (unarni operator)
	--	dekrementiranje
	!	logički komplement (unarni operator)
	~	unarni komplement (unarna negacija)
	+	unarni pozitivni predznak
	-	unarna negacija
3	*	množenje
	/	dijeljenje
	%	ostatak cjelobrojnog dijeljenja (modulo)
4	+	sabiranje
	-	oduzimanje
5	<<	shift bitova ulijevo
	>>	shift bitova udesno
	>>>	shift bitova udesno uz popunjavanje nulama
6	<	manje od
	<=	manje ili jednako
	>	veće od
	>=	veće ili jednako
7	==	jednako
	!=	različito od (nejednako)
8	&	AND
9		OR
10	:=	dodjeljivanje vrijednosti

12.4.6. Funkcije

Osnovna jedinica programa koja obavlja neki koristan posao u JavaScriptu se, kao i u C-u naziva funkcijom.

Da biste stvorili vlastitu funkciju, koristit ćete ključnu riječ `function`. Naprimjer, sljedeća funkcija ispisat će tekst koji primi kao parametar masnim slovima:

```
function ispisuj_masno (tekst) {
    document.write ("<B>" + tekst + "</B>");
}
```

Zgodno je odmah usput primijetiti da u JavaScriptu možete formatirati ekranski ispis koristeći HTML naredbe, kao u gornjem primjeru. Vaš će skript jednostavno pretraživaču poslati cijeli string, koji uključuje i HTML naredbe za određivanje stila ispisa, a pretraživač će to ispisati po njemu poznatim pravilima, koja kažu da sve između `<B>` i `</B>` mora biti masno.

Svoju novu funkciju pozvat ćete navođenjem imena i parametra:

```
Ispisuj_masno ("Želim ocjenu 10!");
```

JavaScript se brine za sva potrebna pretvaranja tipova. Tako naša funkcija može ispisivati i cijele brojeve, i brojeve s pomičnim zarezom te stringove i sve ostalo. U C-u je takvo nešto jako teško izvedivo (treba se mnogo oslanjati na pokazivač), a u Javi je elegantno riješeno mogućnošću preopterećivanja metoda. Ipak, mogućnost pozivanja funkcija na ovaj način samo je posljedica toga da JavaScript ionako automatski pretvara podatke među različitim tipovima.

Ako funkcija prima više parametara, prilikom njenog pozivanja morate joj poslati onoliko njih koliko ih očekuje!

12.5. Stvaranje objekata

Novi objekt možete stvoriti vrlo jednostavno. Prvo je potrebno napisati funkciju:

```
function knjiga (autor, izdovac, stranica, CD, sifra) {  
    this.autor=autor;  
    this.izdovac=izdovac;  
    this.stranica=stranica;  
    this.CD=CD;  
    this.sifra=sifra;  
}
```

koja će pojedinim svojstvima objekta pridružiti odgovarajuće vrijednosti. Oznaka "this" samo naznačava da se mijenjaju svojstva trenutnog objekta, a ne nekog drugog.

Nakon toga možete stvoriti novi objekt tijekom izvršavanja programa pomoću operatora new:

```
novaknjiga=new knjiga ("XY", "UNZE", 1050, true, 01);
```

12.6. Ugrađeni objekti

Kada Netscape Navigator ili Internet Explorer započinju izvršavati skriptu pisanu u JavaScriptu, oni joj stvaraju prividno okruženje sa svim potrebnim objektima koje ona može pozivati. Ti objekti sadrže sve informacije o trenutnoj stranici, poput naslova, sadržaja, informacija o boji pojedinih dijelova itd. Vaš program u JavaScriptu može koristiti samo te objekte i ne može stvarati svoje, ali to je ograničenje za koje smo rekli da je jedno od svojstava skriptnih jezika koji teže što većoj jednostavnosti.

12.6.1. Objekt Window

Objekt Window sadrži funkcije pomoću kojih možete utjecati na izgled pretraživačkog prozora u kojem se izvršava vaš JavaScript program. Slijede neke od zanimljivijih funkcija i primjeri njihovog korištenja.

```
window.frames ['okvir1'].document.bgColor="yellow"
```

mijenja boju pozadine u trenutnom dokumentu, ali samo u njegovom okviru okvir1. Prvi dio te naredbe, window.frames, koristi se za pristup nekom okviru, a nakon toga jednostavno treba pozvati funkciju kojom se postavlja nivoa boja. Dakle, prevedeno na ljudski jezik, gornja naredba kaže "okvir1" oboji u žuto.

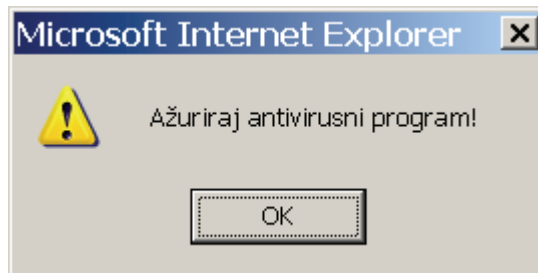
Možda vas u ovom trenutku još zbunjuje to što mi zapravo izrazu na lijevoj dodjeljujemo vrijednost s desne strane izraza na lijevoj da bismo obavili neku radnju. Međutim, JavaScript je i zamišljen tako da radi isključivo s objektima i njihovim svojstvima. Čim vi promijenite svojstvo objekta, to se odmah održava na odgovarajući način, ovisno o tome što ste učinili.

Da biste prikazali poruku u statusnom retku pretraživača, možete pozvati

```
window.status="Ne mogu zaspati bez neke TV serije!";
```

Ako želite prikazati okvir za dijalog s nekom upozorenjem, jednostavno pozovite

```
window.alert ("Ažuriraj antivirusni program!");
```



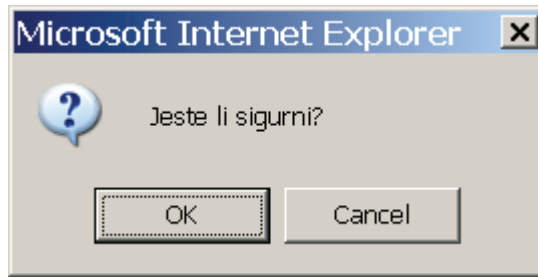
Slika 80. Okvir s upozorenjem

To će prikazati okvir s porukom i korisnik će morati kliknuti na dugme OK.

Ako želite prikazati okvir za dijalog u kojem korisnik može birati između potvrde i odbijanja pozovite funkciju

```
window.confirm ("Jeste li sigurni?");
```

Ako korisnik klikne dugme OK, vašem će se programu vratiti vrijednost true, a ako klikne na Cancel, dobit ćete nazad vrijednost false.



Slika 81. Okvir za potvrdu neke radnje

Objekt Window također se brine za otvaranje novih prozora. Da biste otvorili novi prozor i u njega učitali neki dokument, pozovite funkciju `open`:

```
open ("stranica.htm", "Window1", "toolbar=yes");
```

12.6.2. Objekt Document

Objekt Document daje osnovne informacije o trenutnom dokumentu i koristi se za sve ulazno-izlazne operacije.

Da biste ispisali neki tekst, upotrijebite naredbe `document.write` i `document.writeln`:

```
document.write("Moja prva skripta");  
document.writeln("Moja prva skripta");
```

Razlika između prve i druge naredbe je u tome što će se u drugom slučaju tačka od koje počinje ispis nekog sljedećeg stringa premjestiti u novi redak.

Da biste promijenili boje kojima se označavaju hipertekstualne veze (uključujući i one koje su već posjećene), upotrijebite:

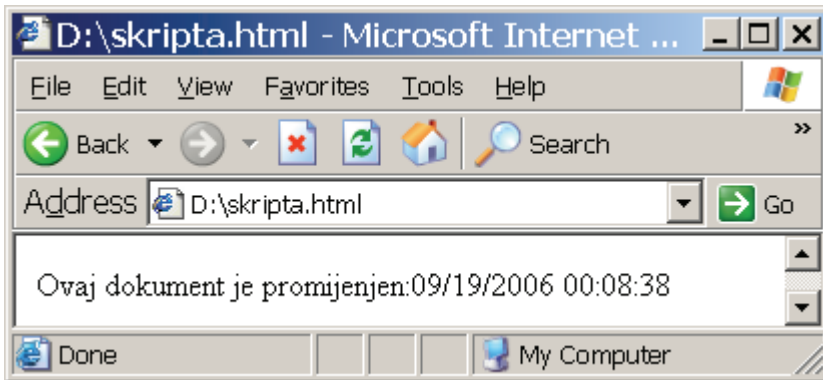
```
document.linkColor = 'red';  
document.alinkColor = 'yellow';  
document.vlinkColor = 'blue';
```

Koristeći JavaScript možete saznati i odakle je došao vaš posjetitelj, tj. s koje je stranice odabrao hipertekstualnu vezu koja je pokazivala na vašu stranicu:

```
document.write (document.referrer);
```

Ako želite ispisati datum zadnje promjene trenutnog dokumenta:

```
document.write ("Ovaj dokument je promijenjen:");  
document.write (document.lastModified);
```



Slika 82. Ispis podataka o zadnjoj promjeni dokumenta

12.6.3. Objekti Date i Math

Objekti Date i Math sadrže uobičajene funkcije za rukovanje datumima i vremenom, te obavljanje matematičkih operacija.

U objektu Math postoje sve funkcije kao i u Javinoj klasi Math (sin, cos, tan, asin, acos, atan, exp, log, sqrt, pow, round, random, abs, max, min). Funkcije se pozivaju na uobičajeni način:

```
rezultat = Math. Cos (2.3564);
```

Novi objekt koji sadrži datum stvorit ćete s:

```
datum = new Date (parametri);
```

Ako parametri nisu navedeni, objekt će sadržavati trenutni datum i vrijeme. Datum možete navesti u obliku stringa u obliku

```
mjesec dan, godina sati:minute:sekunde  
"12. 1, 1974 22:00:00"
```

Treća je mogućnost da navedete cijele brojeve koji redom predstavljaju godinu, mjesec, dan, sate, minute i sekunde, i to tačno tim redoslijedom. No, od navođenja mjeseca i dana ima jedna kvaka: oni se u JavaScriptu numeriraju od 0, a ne od 1!

12.7. Obrada događaja

Događaji su akcije koje obavlja korisnik. Kada korisnik nešto napravi (klikne mišem, odabere neku ikonu...), JavaScript stvara "događaj" (event). Smisao događaja u JavaScriptu sličan je smislu prekida (interrupt). Na taj način programer može biti siguran da će se neki određeni dio JavaScript programa izvršiti ako se nešto određeno dogodi.

Na primjer, mnoge stranice koriste JavaScript za provjeru ispravnosti popunjavanja obrazaca. Ako posjetitelj nije dobro popunio obrazac, treba ga na to upozoriti. Dakle, kada korisnik klikne na dugme koji šalje neke podatke, mi želimo da pretraživač izvrši neke naredbe u JavaScriptu. Da bi se to dogodilo, potrebno je u HTML naredbu za stvaranje dugmeta uključiti poziv tog događaja:

```
<INPUT TYPE="button" VALUE="posalji"
onClick="posalji(this.form)">
```

Novi parametar u naredbi <INPUT> govori koji događaj treba pratiti. Pretraživačima gornji parametar kaže da pozovu vašu JavaScript funkciju "posalji" nakon što korisnik klikne na taj dugme. Ukupno je 9 događaja koje možete pratiti, a oni su prikazani tablicom 7.

Tabela 7. Događaji u JavaScriptu i njihova imena

Događaj	Rukovatelj događaja	Do njega dolazi...
blur	onBlur	...kada korisnik premjesti fokus s trenutnog elementa na Web stranici
click	onClick	...kada korisnik klikne na neki element ili hipertekstualnu vezu
change	onChange	...kada korisnik promijeni neki tekst ili mišem označi neki element na Web stranici
focus	onFocus	...kada se korisnik usredotoči na neki element (npr. kada u obrascu za upis podataka klikne unutar nekog od polja)
load	onLoad	...kada korisnik učita stranicu u pretraživač
mouseover	onMouseOver	...kada korisnik mišem dođe na hipertekstualnu vezu
select	onSelect	...kada korisnik označi neke elemente u obrascu
submit	onSubmit	...kada korisnik pošalje podatke upisane u obrazac
unload	onUnload	...kada korisnik napušta stranicu

Umjesto da funkciju pozivate, ako se radi o vrlo kratkoj naredbi koju treba izvršiti nakon određenog događaja, možete je izravno uključiti u akciju koju treba obaviti nakon događaja. Naprimjer, da biste pokazali okvir s upozorenjem ako korisnik klikne na neki dugme:

```
<INPUT TYPE="button" name="Klikni!"
onClick="alert('Zašto bezveze klikaš?')">
```

Ili biste mogli na svojoj stranici napraviti dugme koji će pretraživaču narediti da se vrati dvije stranice nazad, ovisno o tome gdje je korisnik bio:

```
<INPUT TYPE="button" name="Klikni!"  
onClick="history.go(-2)">
```

Ako želite pozvati neku funkciju nakon što korisnik odluči poslati sadržaj popunjenog obrasca, upotrijebite događaj `onSubmit`:

```
<INPUT NAME="obrazac" ACTION="obrada.cgi"  
onSubmit="provjeri ( )">
```

Ovaj primjer će pozvati funkciju `provjeri()` u vašem programu, pa tako možete provjeriti je li korisnik ispravno popunio sva polja.

Kada korisnik učini neku promjenu u obrascu (npr. odabere neku drugu opciju između ponuđenih ili promijeni sadržaj nekog polja za upis teksta), JavaScript poziva rukovatelj događajem za događaj `onChange`:

```
<INPUT NAME="adresa" onChange="provjeriAdresu ( )">
```

Ovakva naredba pozvala bi funkciju `provjeriAdresu` svaki put nakon što korisnik promijeni "fokus", odnosno nakon što je upisao nešto u neko polje i premjestio se na drugo. Na ovaj način možete provjeriti je li dobro napisao svoje lične podatke (npr. telefonski broj ne može imati 3 cifre, ne mogu sve biti iste, a adresa mora sadržavati barem dva broja, broj ulice i poštanski broj grada). Ako korisnik nije podatke pravilno upisao, možete ga stalno vraćati na isto polje za upis sve dok ne ispravi pogrešku.

Ako želite da se neka poruka ispiše čim korisnik dođe na vašu Web stranicu, upotrijebite događaj `onLoad`. Njega je najbolje uključiti u naredbu `<BODY>` u HTML dokumentu. Sljedeći primjer otvorio bi okvir za dijalog i u njemu prikazao poruku odmah nakon što korisnik učita stranicu koja sadrži taj redak:

```
<BODY onLoad="alert('Dajemo popust za sve narudžbe na  
Webu!')">
```

A ako želite korisnika pozdraviti na odlasku ili obaviti neku drugu radnju, možete to na sličan način učiniti pomoću događaja `onUnload`.

```
<BODY onLoad="alert('Hvala na posjeti!')">
```

Događaj `onMouseOver` radi slično prethodnima: nakon što se mišem nađete iznad određenog objekta, JavaScript će obaviti neku radnju. Na primjer:

```
<INPUT TYPE="button" name="Klikni!"  
onMouseOver="help( )">
```

Na ovaj način biste mogli ispisivati kratke poruke pomoći koje korisniku objašnjavaju što pojedini elementi stranice rade.

12.8. Praktične primjene JavaScripta

Sada ste se upoznali s osnovnim načinima pisanja programa, te radnjama koje vaš skript može obavljati. U preostalom dijelu ovog poglavlja predstaviti ćemo vam neke česte skripte koje korisnici rado uključuju us voje Web stranice.

12.8.1. Pozdravne poruke

S pozdravnim porukama ne bi trebalo pretjerivati, jer korisnici ne vole kada im se pojavljuju prozori s nekakvim upozorenjima i napomenama niti u običnim programima, a na Webu to može biti posebno iritantno jer ometa normalan tijek pretraživanja Webu i skreće sve pažnju korisnika s njegovog željenog puta na vašu pozdravnu poruku. Poruka koju odlučite ispisati pojaviti će se u prozoru kao na slici 83. Ovaj način informiranja korisnika trebali biste koristiti samo ako se na vašim Web stranicama događa nešto zaista nevjerojatno ili nešto na što korisniku svakako treba skrenuti pažnju.

```
<SCRIPT LANGUAGE="JavaScript">

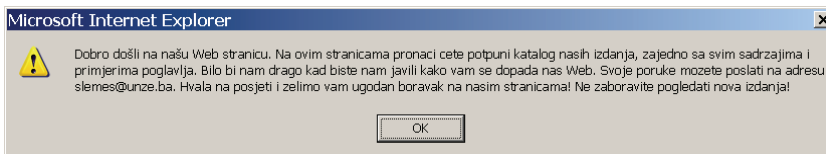
<!--

alert ("Dobro došli na našu Web stranicu. Na ovim
stranicama pronaci cete potpuni katalog nasih izdanja,
zajedno sa svim sadrzajima i primjerima poglavlja. Bilo
bi nam drago kad biste nam javili kako vam se dopada
nas Web. Svoje poruke mozete poslati na adresu
slmes@unze.ba. Hvala na posjeti i zelimo vam ugodan
boravak na nasim stranicama! Ne zaboravite pogledati
nova izdanja!")

//-->

</SCRIPT>

</HEAD>
```



Slika 83. Pozdravna poruka iz prikazanog primjera skripte

Neke verzije pretraživača ne mogu u okviru funkcije alert poslati više od 255 znakova teksta, niti im jedan redak programa u JavaScriptu smije biti toliko dugačak. Zbog toga je sigurnije primjer napisati ovako:

```
<SCRIPT LANGUAGE="JavaScript">

<!--
```

```

var por1=" Dobro došli na našu Web stranicu. Na ovim
stranicama pronaci cete potpuni katalog nasih izdanja,
zajedno sa svim sadrzajima i primjerima poglavlja. Bilo
bi nam drago kad biste nam javili kako vam se dopada
nas Web."

var por2=" Svoje poruke mozete poslati na adresu
slemes@unze.ba."

var por3="Hvala na posjeti i zelimo vam ugodan boravak
na nasim stranicama! Ne zaboravite pogledati nova
izdanja!"

alert(por1 + por2 + por3)

//-->

</SCRIPT>

</HEAD>

```

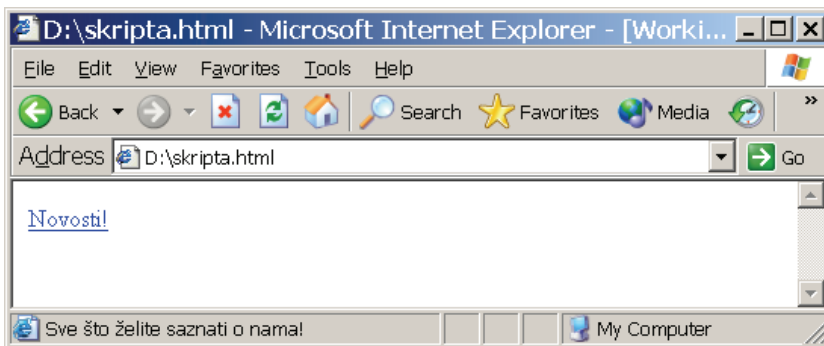
12.8.2. Ispisivanje pomoći

Kombinirajući događaj `onMouseOver` s pozivom funkcije `window.status`, možete u statusnom retku pretraživačkog programa ispisivati poruku koja pojašnjava čemu služi neki element stranice ili kamo vodi hipertekstualna veza. Na primjer, da biste umjesto same adrese, koja će se u statusnom retku pojaviti nakon što korisnik mišem dođe iznad hipertekstualne veze, ispisali detaljnije objašnjenje stranice, pisat ćete u svojem HTML dokumentu:

```

<A HREF="novost.htm" onMouseOver="window.status='Sve
što želite saznati o nama!';return true">Novosti!</A>

```



Slika 84. Tekst iz prethodnog primjera ispisuje se u statusnoj liniji pretraživača kad se mišem pokaže na link označen tekstem "Novosti!"

12.8.3. Automatsko preusmjeravanje korisnika

Ako se lokacija vaših Web stranica promijeni, trebate korisnike uputiti na novu lokaciju. Jedan način da se to učini je korištenje client pull/server push zahtjeva i naredbe <META HTTP-EQUIV="REFRESH"...>, dok je drugi JavaScript:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
alert("Promijenila se nasa URL adresa. Sada cemo vas
automatski prebaciti na nasu novu adresu. Postavite
novu adresu u svoje favorite, a Web administratora cije
su vas stranice uputile na staru adresu obavjestite o
promjeni. Hvala!")
//-->
</SCRIPT>
</HEAD>
<BODY
onload="location='http://nova.adresa.com/novi.htm'">
```

12.8.4. Automatsko preusmjeravanje uz potvrdu

Ako želite da korisnik potvrdi svoju želju da posjeti vaše Web stranice i na novoj adresi, možete malo promijeniti gornji skript tako da koristi funkciju confirm za stvaranje okvira za dijalog u kojem korisnik može kliknuti na dugme OK ili Cancel.

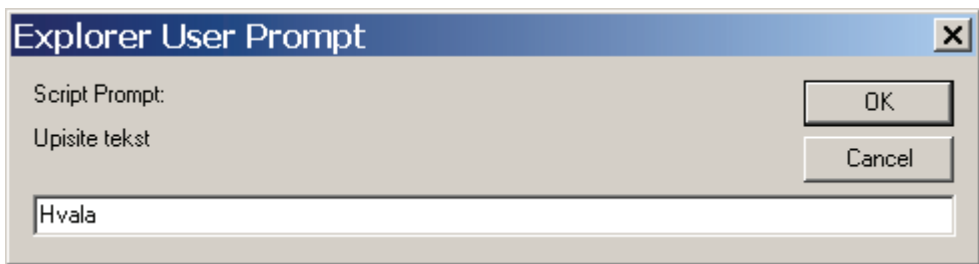
```
<SCRIPT LANGUAGE="JavaScript">
<!--
function redirect() {
    if (confirm ("Promijenila se nasa URL adresa.
Sada cemo vas automatski prebaciti na nasu novu
adresu. Pritisnite OK ako zelite posjetiti nasu
novu adresu."))
    {
        location='nova_stranica.htm'
    }
}
//-->
</SCRIPT>
<BODY onload="redirect()">
```

12.8.5. Stvaranje upita

Da biste na ekranu prikazali okvir za dijalog u koji korisnik treba upisati neki tekst, možete koristiti sljedeći primjer. Nakon što korisnik upiše tekst i klikne na dugme OK, vašoj varijabli dodijelit će se upisani tekst.

```
<SCRIPT LANGUAGE="JavaScript">
<!--
function function1() {
    var $TypedText
    $TypedText = prompt ("Upisite tekst", "")
    return $TypedText
}
//-->
</SCRIPT>
```

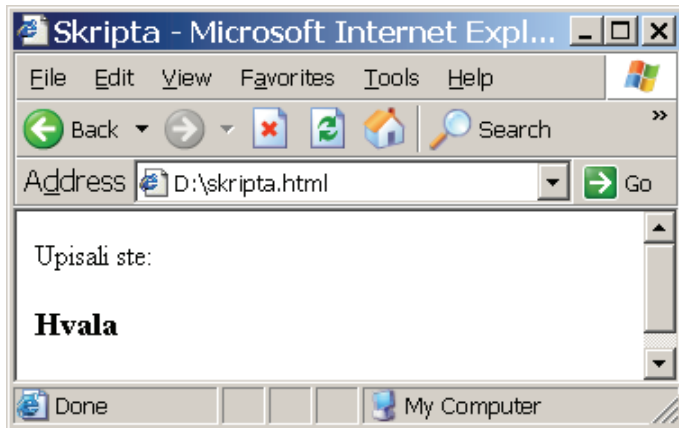
Gornji primjer otvara polje za upis teksta.



Slika 85. Stvaranje upita

Ako nakon upisa želimo ispisati što je korisnik napisao, unutar Web stranice upotrijebit ćemo funkciju `document.write` na onom mjestu gdje želimo ispis.

```
<SCRIPT LANGUAGE="JavaScript">
<!--
var $Text
$Text = function1 ( )
document.write("<P>Upisali ste: <P><H3>" + $Text +
"</H3><P>")
//-->
</SCRIPT>
```



Slika 86. Tekst preuzet iz dijaloga iz prethodnog primjera

12.8.6. Provjera starosti korisnika

Već smo rekli da se događaj `onBlur` koristi kako bi se vaša naredba u JavaScriptu mogla pokrenuti nakon što korisnik napusti neki element na Web stranici, npr. nakon što upiše svoje godine u odgovarajuće polje i prijeđe na sljedeće. Ako vas ne zadovoljava ono što je korisnik upisao, možete mu ispisati upozorenje i poruku o potvrdi upisa pomoću sljedeće funkcije:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
function testAge(form) {
var nAge = form.ageBox.value;
  if (nAge >= 18) {
    if (nAge >=1800) {
      alert("Upisali ste starost od " + nAge +
        "godina. Je li to sigurno tacno?")
    }
  }
  else {
    alert ("Morate imati vise od "  + nAge + "godina")
  }
}
//-->
</SCRIPT>
```

Naravno, u dokumentu mora postojati odgovarajući obrazac:

```
<FORM>

Upisite svoju starost: <INPUT TYPE="text" NAME="ageBox"
onblur="testAge(this.form)">

</FORM>
```

Isto bismo mogli učiniti i koristeći događaj onSubmit:

```
<SCRIPT LANGUAGE="JavaScript">

<!--

function testAge() {
var nAge = document.age.ageBox.value;
  if (nAge >= 18) {
    if (nAge >=100) {
      alert ("Upisali ste starost od " + nAge +
        " godina. Morate imati barem 18!")
      return false
    }
    else {
      return true
    }
  }
  else {
    alert ("Morate imati vise od " + nAge + "godina")
    return false
  }
}

//-->

</SCRIPT>
```

U ovom slučaju obrazac u dokumentu izgleda malo drugačije:

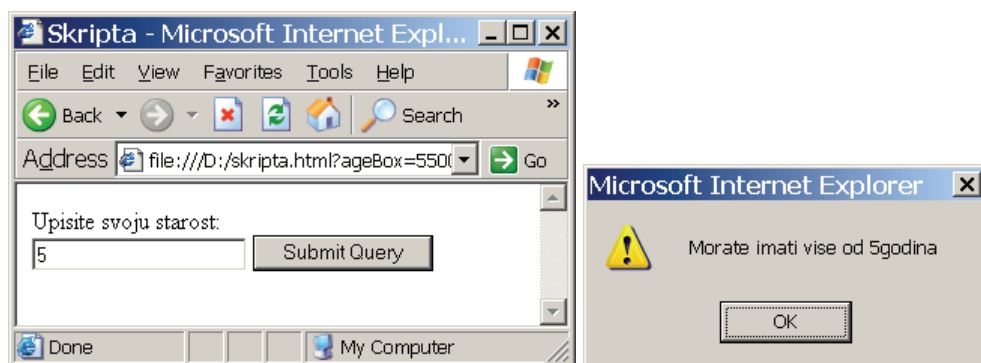
```
<FORM NAME="age" onsubmit="return testAge ( )">

Upisite svoju starost: <BR>

<INPUT TYPE="text" NAME="ageBox">

<INPUT TYPE="Submit" NAME="Submit">

</FORM>
```



Slika 87. Obrazac i poruka za provjeru starosti korisnika

12.9. Kolačići (cookies)

Svaki skriptni programski jezik, pa i JavaScript, ograničen je utoliko što samo ograničen pristup korisnikovom disku i datotekama. Za razliku od Jave, koja uz odgovarajuće sigurnosne mjere može čitati i pisati na disk korisnika (ali samo u dopuštene direktorije!), JavaScript ne može raditi niti to. Da je JavaScript programima dopušteno pisanje stvarnih datoteka na disk, to bi predstavljalo preveliki sigurnosni rizik, jer JavaScript nema tako razrađenu sigurnost kao Java. Mogućnost neposrednog pisanja u datoteku učinila bi pisanje skriptnih virusa vrlo jednostavnim; skript bi mogao zapisati virus na disk ili izbrisati neku vašu datoteku čim dođete na Web stranicu.

Ipak, postoji jedan način na koji JavaScript program može upisivati nešto na tvrdi disk korisnika, a to su kolačići (cookies). Vaša skripta može pretraživaču poslati neke podatke koje će on pohraniti u datoteku cookies.txt. Kasnije, vaša skripta može tražiti da joj pretraživač vrati neki sadržaj iz te datoteke.

Kolačići vam omogućavaju da neke stvari napravite bolje nego s CGI-em, a jednostavnije nego Javom. Naprimjer, kada korisnik naruči neke knjige, u datoteku s kolačićima zapišemo sve podatke koje nam je on poslao (ime, prezime, adresu, telefon) i upamtimo koje je knjige naručio. Sljedeći put kada posjeti naše Web stranice, iz datoteke s kolačićima pročitamo njegovo ime i sve podatke, te automatski popunimo cijeli formular, kako bismo ga oslobodili potrebe da nanovo upisuje mnoštvo podataka. Od korisnika možemo zatražiti samo potvrdu je li sve ispravno upisano, za slučaj da netko drugi sjedi za njegovim računarom i želi upisati svoje ime. A mogli bismo i pratiti koje je sve knjige posjetitelj do sada naručio.

Pomoću JavaScripta se može napraviti da korisnik odabere želi li dobivati obavijesti o novim knjigama čim posjeti naše Web stranice i da odabere želi li čitati popise pojedinih kategorija.

Kolačići imaju svoj "rok trajanja". Kada stvarate novi kolačić, morate navesti kada će mu isteći rok trajanja. Ako kolačić nema određen rok trajanja, pretraživač ga neće upisati u datoteku cookies.txt nakon izlaska iz pretraživača. (Inače svi pretraživači drže kolačiće u memoriji tijekom rada i spremaju ih na disk tek po izlasku iz programa.) Na taj način možete pamtit i neke podatke samo tijekom jednog posjeta Web stranici.

Ako želite da kolačić bude zapamćen na disku korisnika, što bi nam svakako trebalo ako želimo nakon nekog vremena vidjeti koje je sve knjige naručio, morat ćete zadati rok trajanja kolačića. Pretraživači automatski čiste datoteku s kolačićima i izbacuju one kojima je rok istekao. Ova je mjera neophodna zbog toga što postoje ograničenja broja kolačića i njihove veličine, tako da baš ne možete upisivati što god želite i kako god želite na korisnikov disk. Ograničenja nisu uvedena zato što se to ne bi moglo izvesti drugačije, nego zbog toga da se s previše kolačića ne bi popunio sav prostor na disku. Tako pretraživač ne smije čuvati više od 300 kolačića od kojih je svaki veličine najviše 4 KB. Ukupna veličina datoteke cookies.txt zbog toga neće nikad biti veći od 1,2 MB.

Da bismo stvorili i kasnije pročitati jednostavan kolačić, u svom programu ćemo napisati sljedeće dvije funkcije:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
function SetCookie(name, form) {
document.cookie = name + form.textCookie.value + "; "
form.textCookie.value = ""
}
function ShowCookie(form) {
form.textCookie.value = document.cookie
}
//-->
</SCRIPT>
```

Prva funkcija, SetCookie, uzima vrijednost polja name iz obrasca. Zatim mijenja svojstvo document.cookie, dodajući mu vrijednost tog polja, ali mu još dodaje i ostali tekst koji je korisnik napisao u polje form.txtCookie.value, nakon čega briše sadržaj tog polja. Druga funkcija, ShowCookie, pročitat će kolačić.

Sada moramo stvoriti odgovarajuće sučelje za upis podataka koji će se spremati kao kolačić:

```
<FORM>
<INPUT TYPE="Text" NAME="textCookie" size=50 ROWS=10
COLS=73><P>
```

```

<I>Kliknite na ovo dugme da biste poslali
kolacic:</I><BR>

<INPUT TYPE="button" VALUE="Create Cookie"
NAME="butSet" onclick="SetCookie('Cookie1',
this.form);"><P>

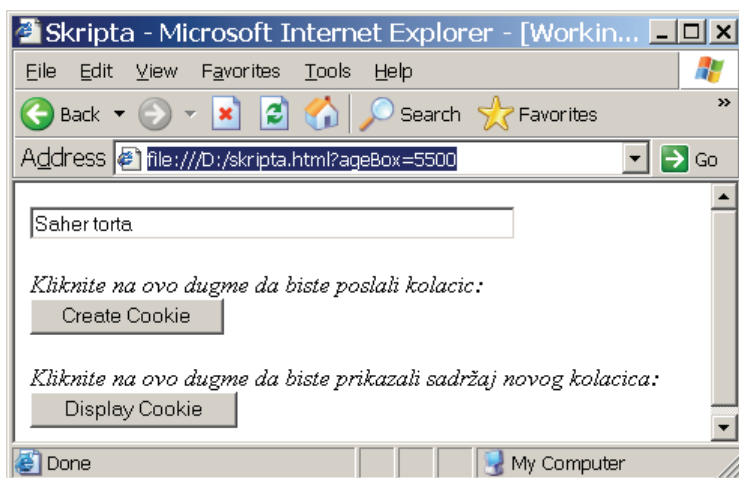
<I>Kliknite na ovo dugme da biste prikazali sadržaj
novog kolacica:</I><BR>

<INPUT TYPE="button" VALUE="Display Cookie"
NAME="butDisplay" onclick="ShowCookie(this.form);">

</FORM>

```

Da biste ovo isprobali, upišite neki tekst u polje za upis i kliknite dugme na Web stranici da biste poslali svoj odgovor skriptu. On će upamtiti upisane vrijednosti kao kolačiće, pa ih možete prikazati klikom na drugo dugme.



Slika 88. Stranica sa kolačićima

Korisnici Netscape Navigatora i Internet Explorer mogu isključiti primanje kolačića. Ako je ono uključeno, možete uključiti još i opciju za dobivanje upozorenja o tome da neka skripta želi postaviti kolačić. Tako možete odlučiti želite li da skripte s nekih Web stranica uopće nešto čuvaju na vašem disku.

13. PHP

13.1. Šta je PHP?

PHP (*skraćenica od PHP: Hypertext Preprocessor*) je skript-jezik koji se direktno upisuje u HTML stranice, tj. autor upisuje PHP naredbe skupa sa HTML-naredbama u jednu datoteku.

Prilikom pozivanja ove datoteke, PHP-naredbe se "*prevode*" uz pomoć jednog dodatnog software-a na serveru, i to korak po korak - tj. onako kako su napisane, i rezultat tog "*vrednovanja*" vraća se posmatraču kao rezultat.

Da bi to funkcionisalo, server mora biti "*PHP-sposoban*". Standardno uređen server ne prepoznaje ove PHP naredbe, tako da ih ne može prevesti ni prepoznati.

U zavisnosti od načina instalacije, ovaj dodatni software-a može "*prevoditi*" tj. razumije samo datoteke sa nastavkom "**.php**", ili također i datoteke sa nastavcima "**.html**", "**.htm**".

Nastanak PHP se vezuje za 1994 god. i od tada ima sve više i više pristalica. Poseban značaj ima korištenje PHP-a u vezi sa "*bazama podataka*".

PHP-jezik je dosta sličan programskom jeziku C, javi i perlu.

13.2. Ubacivanje PHP naredbi u HTML

Za ubacivanje PHP naredbi u HTML imamo 4 mogućnosti:

```
<?...?>
```

```
<?php...?>
```

```
<script language="php">...</script>
```

```
<%...%>
```

["... " stoji za PHP naredbe po volji.]

13.3. Prva PHP naredba

Prvi primjer, kao i uvijek je "Hello world!".

Naredba **echo** pokazuje tekst na ekranu, (tzv. strings):

```
<? echo "Hello world!" ; ?>
```

Na kraju svake PHP naredbe stoji "**tačka-zarez**";

U PHP-u postoji više "*specijalnih znakova*". Najvažniji od njih je znak "**\n**".

On označava novi red. (Ko je programirao u C, C++ ili Perlom - njemu je to već poznato). Prvi primjer: echo "Zdravo\n";

```
echo "Zdravo\n";
```

Ova naredba se razlikuje od sljedeće: `echo "Zdravo<BR>"`;

```
echo "Zdravo<BR>" ;
```

U prvom slučaju će u HTML tekstu koji server vraća biti izveden "novi red". U drugom slučaju HTML tekst sadrži HTML naredbu za novi red.

13.4. Varijable (promjenljive)

Da bi se mogli pisati smisleni programi i aplikacije potrebne su nam promjenljive. Sve promjenljive počinju sa znakom 'dolar' `$` !!!

```
$text = "Ja sam jedna recenica!" ;
echo $text,$text,$text;
$l = "duga";
$k = "kratka";
echo "Ja sam jedna $l$l$l$l$l$l$l recenica!";
echo "Ja sam jedna $k recenica!";
$i = 10;
$j = 5;
echo $i, "+"$j, "=", $i+$j;
```

Tip promjenljive (cijeli broj, decimalni broj, tekst ...) dobija se automatski prilikom primjene. (Za razliku od C i C++ gdje se tip promjenljive morao egzaktno i ručno odrediti). Ovaj "ručni" način određivanja tipa promjenljive vrši se samo u specijalnim slučajevima.

13.5. Pred-definisane (već određene) promjenljive

Kako je PHP specijalno razvijen za pravljenje dinamičkih web-stranica, samim tim je i obrada podataka koji vode porijeklo iz formulara lagana.

Primjer:

Pretpostavimo da u formularu stoji:

```
<FORM ACTION ="RadiNesto.html" METHOD = "POST">
<INPUT NAME="ImePoIzboru">
<INPUT TYPE="submit">
</FORM>
```

Prilikom "**klika**" na **SUBMIT** dugme, biće pozvana datoteka **RadiNesto.html** .

Ova datoteka može npr. ovako izgledati:

```
<?
echo "Vi ste ukucali $ImePoIzboru . ";
?>
```

Tekst koji je ukucan u formular-polje **ImePoIzboru**, sačuvan je (memorisan) u PHP promjenljivoj **\$ImePoIzboru**.

Početnici trebaju uvijek koristiti **METHOD=POST** prilikom izrade formulara !

13.6. Jednostavne računske operacije

Sljedeće računske operacije su na raspolaganju:

Znak	Radnja	Primjer
+	Sabiranje	$\$i + \j
-	Oduzimanje	$\$i - \j
*	Množenje	$\$i * \j
/	Dijeljenje	$\$i / \j
%	Ostatak	$\$i \% \j

Objašnjenje za radnju Ostatak: radnja izračunava ostatak dijeljenja, npr. $23\%17$ daje 6 , jer je 23 podijeljeno sa 17 jednako 1 i ostatak 6 .

13.7. Povezane riječi ('strings')

```
$d = "Dugacak";
$k = "Kratak";
echo $l.$k;
```

DugacakKratak

(pokazuje na ekranu riječ DugacakKratak)

Kao i u C i C++ i dalje važe već poznate skraćenice i olakšice:

```
$i++ uvećava varijablu $i za 1
++$i uvećava varijablu $i za 1
$i-- umanjuje varijablu $i za 1
--$i umanjuje varijablu $i za 1
```

Razlika između `$i++` i `++$i` :

```
$i=0;
echo $i++;
```

0

vraća 0 (nulu), tek nakon toga se vrijednost **\$i** uvećava za 1.

U drugom slučaju imamo:

```
$i=0;
echo ++$i;
```

1

U ovom slučaju će najprije **\$i** biti uvećano za jedan i tek onda biti pokazana vrijednost 1 na ekranu.

13.8. Petlje

Na samom početku jedan primjer:

```
$t = "Moram sam raditi domace zadatke !<BR>\n";
$i = 0;
while ($i<10)
{
    echo $t;
    $i++;
}
```

U ovom slučaju će biti 10 puta pokazan tekst koji je sadržan u promjenljivoj **\$t**. Na početku se varijabla **\$i** postavlja na vrijednost 0 (nula), i onda u svakom novom prolazu kroz petlju uvećava za 1 (jedan). Kada vrijednost dostigne 10, uslov **\$i<10** nije više ispunjen i petlja se napušta.

13.9. Daljnje naredbe za kontrolu programa

Umjesto :

```
$t = "Moram sam raditi domace zadatke !<BR>\n";
/* rečenica se memoriše u promjenljivu $t */
$i = 0;
```

```
// tzv. brojač se postavi na nulu
while ( $i<10)
    //sve dok je $i manje od 10
{
    echo $t;
    //pokaži sadržaj promjenljive $t
    $i++;
    //uvećaj i za 1
}
```

moglo bi se isto tako napisati:

```
$t = "Moram sam raditi domace zadatke !<BR>\n";
$i = 0;
    //brojač na nulu
do
    //izvrši !!!
{
    echo $t;
    //pokaži sadržaj $t, tj. memorisanu rečenicu
    $i++;
    //uvećaj brojač za 1
} while ( $i < 10 );
    // i to radi sve dok je $i manje od deset
```

Gdje je razlika? Razlika se najbolje može uočiti ako se kao početna vrijednost za brojač **\$i** umjesto nule **\$i=0**; stavi deset **\$i=10**; . U prvom slučaju nije ispunjen uslov ($i < 10$), tako da naredbe između "vitičastih" zagrada neće biti izvedene. U drugom slučaju biće najprije izvedene naredbe unutar "vitičastih" zagrada, i tek onda će biti provjereno da li je uslov još uvijek zadovoljen tj. da li važi **\$i<10** . Kako ovo nije slučaj, biće prekinuto izvođenje naredbi unutar petlje. Važno je primijetiti da će u ovom drugom slučaju, znači upotrebom **DO-WHILE** **petlje** naredba biti **BAREM JEDNOM izvedena**. Još jedna mogućnost za primjenu petlji u programu je tzv. **FOR** petlja.

Primjer:

```
$t = "Moram sam raditi domace zadatke !<BR>\n";
for ($i=0;$i<10;$i++)
{
    echo $t;
}
```

'for' petlja se sastoji od tri izraza. U opštem slučaju:

```
for (izraz1;izraz2;izraz3)
{
    ...
}
```

Sa vrijednošću **izraz1** inicijalizira se petlja, prosto rečeno "*brojač se stavlja na početnu vrijednost*". Vrijednost data u **izraz2** predstavlja krajnju vrijednost . Kada se ona dostigne izlazi se iz petlje. Kroz **izraz3** daje se tzv. "*korak*", tj. dajemo vrijednost za koju treba povećati ili umanjiti brojač. Prednost '**for**' naredbe je taj što sve naredbe potrebne za kontrolu petlje stoje u jednom redu (tačnije u jednoj zagradi). U idućem primjeru biće pokazana jedna petlja u kojoj se brojač umanjuje za određenu vrijednost, tj. brojat ćemo "*unazad*".

```
$t = "Moram sam raditi domace zadatke !<BR>\n";
    //promjenljiva $t
for ($i=10;$i>0;$i--)
    /* brojač se postavlja na 10. U svakom idućem koraku bit će
    umanjen za 1 sa $i--,... naredba se izvršava sve dok je $i>0 */
{
    echo $t;
}
```

13.10. Uslovno grananje programa

Ukoliko se program želi "usmjeriti" u nekom pravcu, i to u zavisnosti od izabranog slučaja, upotrijebit će se **if** naredba:

```
if ($i<0)
```

```
        //ako je $i manje od nule
    {
        echo "$i je manje od nule\n";
        //pokaži na ekranu tekst ...
    }

ili :

    if ($i<0)
        //ako je $i manje od nule
    {
        echo "$i je manje od nule\n";
        /*pokaži tekst */
    }
    else
        // u suprotnom slučaju...
    {
        echo "$i nije manje od nule\n";
        /* pokaži ovaj tekst*/
    }
}
```

Ove naredbe možemo i "uvezati" :

```
    if ($i<0)
        //ako je $i manje od nule
    {
        echo "$i je manje od nule\n";
        //pokaži ovaj tekst
    }
    else if ($i>0)
        //ako je $i veće od nule
    {
        echo "$i je vece od nule\n";
        //pokaži ovaj tekst
    }
}
```

```
}  
else  
    //ako $i nije ni manje ni veće od nule,...  
{  
echo "$i je nula\n";  
    // onda je sigurno $i=0, pa to pokaži...  
}
```

Ako jednu promjenljivu treba više puta testirati, onda je lakše upotrijebiti naredbu **switch**. Primjer:

```
switch ($ime)  
{  
case "Suljo":  
echo "Ja sam Suljo";  
break;  
case "Mujo":  
echo "Ja sam Mujo";  
break;  
case "Haso":  
echo "Ja sam Haso";  
break;  
default:  
echo "Nisam Suljo, Mujo a ni Haso, ja sam Huso";  
}
```

Ako je ime sadržano u promjenljivoj **\$ime** *Suljo*, (tj. ako promjenljiva **\$ime** sadržava vrijednost "Suljo") biće izvršena prva naredba: **echo "Ja sam Suljo"**; Nakon toga se prekida dalje izvršenje naredbi u petlji sa naredbom **break**; Ako ime nije Suljo, onda se nastavlja sa "*ispitivanjem slučajeva*" datim u **case**, u datom slučaju poređenjem sadržaja promjenljive **\$ime** i imena (riječi Mujo, Haso,...). Ukoliko ni jedan slučaj ne odgovara (tj. ako ime nije ni Suljo ni Mujo ni Haso), onda će biti izvršena naredba **default**; Pod **default** se uvijek stavlja ona naredba koja treba biti izvršena ako ni jedna od pretpostavki nije istina, u našem slučaju biće pokazan tekst: Nisam Suljo, Mujo a ni Haso, ja sam Huso.

13.12. Proširena poređenja

Do sada smo vidjeli osnovne mogućnosti usporedbe: provjeru na **veće**, **manje** ili **jednako** ($\$i < 10$, $\$i > 10$, $\$i = 0$) . Osim ovih osnovnih usporedbi, naravno da je moguće napraviti i "*proširene*" usporedbe kao npr.:

```
$i == 10
    // Da li je $i jednako 10 ?
$i <= 10
    // Da li je $i manje ili jednako 10 ?
$i => 10
    // Da li je $i veće ili jednako 10 ?
$i != 10
    // Da li $i nije jednako 10 ?
```

Ove upite možemo i koristiti u kombinaciji, kao npr.:

```
($i==10) && ($j>0)
    // Da li je $i jednako 10 i $j veće od 0 (nule)?
($i==10) || ($j==0)
    // Da li je $i jednako 10 ili $j jednako 0?
```

Prikazani slučajevi su identični (tačnije sintaksa) je ista kao i kod C jezika. Znači:

&& - znaci 'bool' i (**AND**)

|| - znaci 'bool' ili (**OR**)

13.13. Predefinisane promjenljive

U PHP datoteci, koja je pozvana iz nekog formulara, imena "*elemenata iz formulara*" stoje na raspolaganju kao promjenljive. Ovu osobinu možemo iskoristiti tako što ćemo sami formular i naredbe koje "*komanduju*" formularom (tj. šta i kad će biti izvedeno) memorisati u jednu datoteku. Pretpostavimo, naša datoteka form.html sadržava sljedeći formular:

```
<FORM ACTION="form.html" METHOD=POST>
<INPUT NAME="ImePoIzboru">
<INPUT TYPE="submit">
</FORM>
```

U tom slučaju pozivom (klikom) na submit bit će pozvan sama datoteka (znači form.html). U tom slučaju je promjenljivoj **ImePoIzboru** dato baš ime po izboru. Ovo se može iskoristiti:

```
<?
if (isset($nadimak))
    //ako egzistira (postoji) promjenljiva
    $ImePoIzboru
{
echo "Vi ste dali $ImePoIzboru\n";
    //pokaži na ekranu "Vi ... "
}
?>
<P>
<FORM ACTION="form.html" METHOD=POST>
<INPUT NAME="nadimak">
<INPUT TYPE="submit">
</FORM>
```

Prilikom prvog poziva datoteke form.html (tj. dok još nije kliknuto "dugme" submit), promjenljiva **\$ImePoIzboru** još ne postoji, što opet znači da izraz (**isset(\$nadimak)**) **NIJE ISTINIT**. Popunimo li formular i kliknemo na submit, varijabla **\$ImePoIzboru** će biti "napravljena", izraz (**isset(\$nadimak)**) je **ISTINIT**, tako da će biti pokazan tekst Vi ste dali **\$ImePoIzboru**. Ukoliko želimo prekinuti izvođenje naredbi, koristimo se naredbom **exit**.

13.14. Priključivanje datoteka

Još jedna sličnost sa C i C++. Naredbom

```
include ("NekaDatoteka");
```

bit će pročitana sadržaj datoteke **NekaDatoteka** kao kad bi cijela datoteka (cijeli njen sadržaj) stajala na tom mjestu. Na ovaj način je omogućeno postizanje istovjetnog "layout-a" i kod većih projekata.

13.15. Rad sa datotekama

U sljedećem slučaju htjeli bismo da "*čitamo*" iz neke datoteke koja leži na serveru, ili da u nju nešto dodamo. (Najčešći slučaj je vjerovatno upisivanje u "*knjigu*"

gostiju", ili povećanje za jedan na brojaču kod nove "*posjete*".) Iz sigurnosnih razloga je web-server na serveru dodijeljen korisniku (user - u) koji ima vrlo mala prava. Između ostalog, taj korisnik ne smije onako kako bi on htio pisati i brisati po vašim datotekama (tj. vašim htm ili html stranicama). Još manje ima pravo da sam polaže nove datoteke u vaš folder na web-serveru.

Kako položiti datoteku za "upisivanje" na web-serveru?

Da bismo položili datoteku **NekaDatoteka** u koju web-server smije upisivati podatke, potrebni su sljedeći koraci:

U nekom FTP programu kliknemo desnom tipkom na mišu i izaberemo **chmod**. Ukoliko nemamo izbor **chmod** sa desnim klikom, naredbu možemo poslati i direktno ftp-om:

site chmod xyz ImeDatoteke

Ovdje su **X**, **Y** i **Z** tri cifre koje služe za određivanje prava za ovu datoteku, i to:

X - stoji na prvom mjestu za prava koje ima vlasnik (onaj koji je položio datoteku)

Y - stoji na drugom mjestu za prava koja ima grupa kojoj vlasnik pripada;

Z - stoji na trećem mjestu za prava koja ima ostatak svijeta -tj. svi drugi.

Pri tome važi:

- 0 znači : nikakvo pravo (potpuna zabrana).
- 1 znači : pravo izvođenja (važno za programe i direktorije).
- 2 znači : pravo pisanja (što istovremeno znači i "pravo brisanja").
- 4 znači : pravo čitanja.

Ova prava se mogu sabirati, pojašnjenje će se dati u primjeru za datoteku *primjer.htm*. Htjeli bismo da postavimo sljedeća prava: Da vlasnik ima pravo čitanja (4) + pravo pisanja (2) + pravo izvođenja (1) što znači sva prava, slijedi:

$$4+2+1 = 7;$$

Da grupa kojoj vlasnik pripada ima pravo čitanja (4) i pravo izvođenja (1):

$$4+1=5;$$

I da svi ostali imaju samo pravo za izvođenje (1); Znači prava za vlasnika 7, za grupu 5 i ostale 1, pa je naredba:

site chmod 751 primjer.htm

Ako bismo stavili :

site chmod 777 primjer.htm

značilo bi da **SVI IMAJU SVA PRAVA ZA OVU DATOTEKU, A TO BI BILO VRLO OPASNO!!!**

Web-server radi u normanom slučaju pod pravima za "ostale", što znači da nema ni prava vlasnika, a ni prava grupe kojoj pripada vlasnik. (*Ovo se najčešće zaboravi prilikom prebacivanja CGI-skripte na server, pa se onda čudimo zašto skripta ne radi :)*) Pođimo od toga da sada imamo datoteku pod imenom **NekaDatoteka** iz koje želimo nešto čitati. Prije svega moramo otvoriti ovu datoteku. Pri tome je neophodno naznačiti koje akcije želimo preduzeti npr. čitati iz datoteke, upisivati u nju, brisati itd....

Naredba glasi:

```
$dato = fopen("NekaDatoteka", "r");
```

Ovo znači da otvaramo datoteku **NekaDatoteka** i da želimo iz nje čitati (" r " znači čitati - eng. read). "*Tehničke detalje*" ćemo prepustiti promjenljivoj **\$dato** , tj. kada budemo željeli čitati iz datoteke **NekaDatoteka** jednostavno ćemo upotrijebiti promjenljivu **\$dato**.

U drugom koraku čitat ćemo iz datoteke **NekaDatoteka** i to red po red , sve dok ne dođemo do kraja datoteke:

```
while (!feof($NekaDatoteka))
{
    $red = fgets($NekaDatoteka, 1000);
    echo $kraj;
}
```

- **feof(\$NekaDatoteka)** je istina kada dođemo na kraj datoteke. (EOF je skraćenica od *End Of File*).
- **\$red = fgets(\$dato, 1000);** Uz pomoć systemske funkcije **fgets** postizemo sljedeće: pročitaj maksimalno 1000 znakova u redu iz datoteke **NekaDatoteka**, prekini sa čitanjem kada naiđeš na znak za novi red ili na znak za kraj datoteke, i to "*pročitano*" upiši u promjenljivu pod imenom **\$red**.
(*Objašnjenje za broj 1000 ... Broj 1000 je izabran po slobodnoj volji jer je vrlo vjerovatno niko neće napisati 1000 znakova u jedan jedini red. Da smo htjeli mogli smo napisati i 1.000.000, onda bismo imali još veću vjerovatnoću da niko nije napisao milion znakova u jedan red :).*)
- **echo \$red;** jednostavno pokazuje sadržaj promjenljive **\$red** na ekranu.

Na kraju moramo još zatvoriti datoteku:

```
fclose($NekaDatoteka);
```

Prilikom otvaranja neke datoteke imamo sljedeće mogućnosti:

" **r** " : Samo čitanje, i to od početka datoteke

" **r+** " : Čitanje i pisanje od početka datoteke

" **w** " : Samo pisanje u datoteku. Ako datoteka već postoji, onda će dotadašnji sadržaj biti izbrisan. Ako datoteka ne postoji pokušat će se kreirati.

" **w+** " : Čitanje i pisanje. Ostalo važi kao i za "w".

" **a** " : Samo pisanje (bolje reći "dopisivanje") u datoteku. Znači ako datoteka već postoji bit će nastavljeno pisati u nju na kraju - tamo dokle se već stiglo. Ako datoteka ne postoji, pokušat će se kreirati nova datoteka.

" **a+** " : Čitanje i pisanje. Ostalo važi kao i za "a".

Kod zadnjeg slučaja "**a+**" se sa čitanjem počinje isto tako na kraju. A na kraju, naravno, ne stoji **NIŠTA**. To znači da neće biti ništa pročitano.

Sa naredbom:

```
fwrite($NekaDatoteka, "Ja ucim PHP");
```

može se upisati neki tekst (u primjeru upisujemo tekst "Ja ucim PHP" u datoteku pod imenom **\$NekaDatoteka**).

13.16. Obrada slika

U PHP postoji naredba **ImageGif()**. Uz pomoć ove naredbe može se "*napraviti*" neka GIF-sličica. Za to su nam još potrebne funkcije sadržane u slobodno dostupnoj (free) biblioteci gd. Firma **Unisys** ima patent-prava za *LZW* komprimiranje podataka, koje se, između ostalog, primjenjuje i kod pravljenja "**GIF-sličica**". Kako firma UNISYS zabranjuje pravnim putem upotrebu ovog LZW komprimiranja, u novijim verzijama gd biblioteke nije više moguće praviti GIF-sličice. Tako da je kao rezultat svega ovoga : PHP naredba ImageGif() se ne može više koristiti! Ali zato je korištenje funkcije koja polaže .jpg slike je **gotovo identična**. Osim toga, nove funkcije se dodaju gotovo dnevno, i u svakom slučaju ćete redovno morati (ili željeti?) da upoznate najnovije funkcije koje olakšavaju i omogućuju rad sa PHP-om.

PHP se ne koristi samo za stvaranje HTML teksta, isto tako je moguće dinamičko pravljenje slika. Npr. mogli bi položiti jednu datoteku pod imenom **slika.php3**, čijim bi pozivom dobili sliku. Ova datoteka ne bi sadržavala HTML, u njoj bi stajale **isključivo** PHP - naredbe. ISKLJUČIVO znači isto tako da prije prve <? naredbe **ne smije stajati ništa** (baš ništa, čak ni prazno mjesto ili red !!). U suprotnom slučaju bi "*browser*" očekivao nekakav tekstualni dokument.

Slijedi primjer:

```
<?
Header( "Content-type: image/gif");
$slicica = imagecreate(200,100);
$plava = ImageColorAllocate($slicica,
    0x2c,0x6D,0xAF);
ImageFilledRectangle($im,0,0,200,100,$plava);
ImageGif($slicica);
ImageDestroy($slicica);
?>
```

Ako pozovemo ovu datoteku, dobit ćemo jedan plavi pravougaonik. Naravno, koristeći:

```
<IMG SRC="slika.php3">
```

možemo povezati ovaj pravougaonik u nekoj HTML stranici. Šta se postiže gore navedenim naredbama?

Naredba:

```
Header( "Content-type: image/gif");
```

najavljuje browser-u da je ono što slijedi jedna GIF-slička. (Ako želimo spriječiti javljanje greške možemo privremeno ovaj red "izbaciti iz upotrebe" ostavljajući ga kao komentar stavljanjem znaka # na početak reda.)

Naredba:

```
$slicica = imagecreate(200,100);
```

pravi sličicu koja je 200 px (piksela) široka i 100 px visoka.

Svaka boja mora biti prije same upotrebe "stvorena" tj. određena. Ovo postizemo naredbom:

```
$plava = ImageColorAllocate($im, 0x2c,0x6D,0xAF);
```

ovdje je **\$slicica** slika za koju određujemo boju, ostala tri parametra predstavljaju heksadecimalne RGB-vrijednosti. (Obrati pažnju na sintaksu: **0x2c** odgovara decimalnom udjelu crvene boje od **36**).

Sa:

```
ImageFilledRectangle($slicica,0,0,200,100,$plava);
```

nacrtat ćemo jedan popunjeni pravougaonik , koji počinje u lijevom gornjem uglu (0,0), i završava 200 px desno i 100 px dole. Naravno, ispunjen je sa prije određenom (u promjenljivoj **\$plava**) plavom bojom.

Naredba:

```
ImageGif($slicica);
```

znači: najprije pravi jednu GIF-sličicu na osnovu "*unutrašnjeg predstavljanja*" i onda je pošalji nazad posmatraču, dok

```
ImageDestroy($slicica);
```

oslobađa interno zauzeto mjesto za memorisanje.

Postoji čitav niz mogućnosti za pravljenje slika. Npr. možemo upisati neki tekst u sličicu koju stvaramo:

```
$bijela = ImageColorAllocate($slicica,  
    255,255,255);  
  
imagestring($slicica,5,20,20,"Zdravo",$bijela);
```

Uz pomoć prve naredbe definišemo bijelu boju. (Ovdje se još da primijetiti da smo bijelu boju odredili sa decimalnim vrijednostima - 3 x 255). Drugom naredbom postizemo upisivanje teksta "*Zdravo*" veličine 5 koji počinje na mjestu 20,20 (20 px desno i 20 dole). Obje naredbe treba da stoje direktno ispred naredbe

```
ImageGif($slicica) .
```

Osim pravougaonika moguće je "**crtati**" i ostale uobičajene elemente kao što su krugovi, elipse, poligoni, lukovi itd.

13.17. Povezivanje sa bazama podataka

Jedno od glavnih područja u kojima se primjenjuje PHP je "**on-line**" korištenje baza podataka. Baza podataka sadržava odgovarajuće "**elemente**" ("*redove*" sa istovjetnim vrstama podataka npr. "*prezime, ime, ulica, broj kuće, poštanski broj, mjesto*"). U svakom redu stoje različiti upisi ("*promjenljive*") koje odgovaraju tipu podataka.... tako bi ime bilo jedna "*riječ*", dok bi broj kuće ili poštanski broj imali tip "*cijeli broj*". **Redovi** - "*upisi*" mogu biti pozvani, čitani, mijenjani i sl., moguće je dodavanje novih upisa, brisanje starih itd. Svi ovi postupci nazivaju se jednim imenom "**upiti**".

Najčešće korištena vrsta baza podataka su **SQL** (*skr. od Structured Query Language*) - baze koje mogu da se obrađuju pomoću jezika SQL. PHP se može koristiti za vezanje sa više vrsta baza podataka, a najčešća je **MySQL**. Prednosti MySQL baza podataka su relativno velika brzina i to što je njihovo korištenje besplatno.

Da bismo izveli neki SQL-upit sa PHP-om, moramo prije svega otvoriti bazu podataka - odnosno povezati se s njom. (*Što bi bilo identično sa čitanjem iz datoteke i upisivanje u datoteke*). Nakon toga bi se poslala SQL- naredba bazi podataka, bio bi preuzet odgovor od baze i na kraju bi se baza još morala zatvoriti.

13.17. Osnove SQL-a

SQL-baze podataka se sastoje od jedne ili više tzv. **tabela**. Jedan element u bazi odgovara jednom redu u tabeli. Primjer: Ocjenjivanje studenata.

Tabela studenata:

ID	Ime	Prezime	Ocjena	Komentar
1	Amir	Alić	9	samo usmeni
2	Edo	Mehić	8	praktični dio ispita
3				
4				

Radi jednostavnosti ograničit ćemo se samo na jednu tabelu u bazi.

Najčešće se koriste četiri vrste upita:

- pročitati: **SELECT**
- ubaciti : **INSERT**
- prepisati (promijeniti postojeći tekst) : **UPDATE**
- izbrisati: **DELETE**

Ako hoćemo da pogledamo sva prezimena (*Prezime*) studenata iz baze to postizemo SQL-naredbom:

```
SELECT Prezime FROM studenti;
```

Odgovor bi bio :

- Alić
- Mehić
- ...

Ako hoćemo da vidimo **ime** (*Ime*) i **prezime** (*Prezime*) studenata onda bi SQL-naredba bila:

```
SELECT Ime,Prezime FROM studenti;
```

Odgovor bi bio :

- Amir Alić
- Edo Mehić
-

Cijeli "**element**" tj. cijeli red dobili bi sa: **SELECT * FROM** studenti;

Odgovor bi bio

1. Amir Alić 9 samo usmeni

2. Edo Mehić 8 praktični dio ispita

3.

Moguće je iz tabele uzeti samo one podatke koji zadovoljavaju neki uslov. Npr. sa:

```
SELECT * FROM studenti WHERE Ocjena=9;
```

dobijamo samo studente koje u koloni Ocjena u bazi imaju upisanu vrijednost 9.

Odgovor bi bio

- Amir Alić 9 samo usmeni

Sa:

```
SELECT * FROM studenti ORDER BY Prezime;
```

Odgovor bi bio:

- Amir Alić 9 samo usmeni
- Edo Mehić 8 praktični dio ispita

dobijamo studente, poredane po prezimenu..

Nove "redove" u tabeli možemo dodati sa:

```
INSERT studenti (ID,Ime,Prezime,Ocjena,Komentar)
VALUES (3,'Ivo','Zec','9','');
```

Ta naredba bi dodala novi red u tabelu studenti i u taj red upisala podatke za studenta koji se zove "Ivo Zec" i ima ocjenu 9.

Postojeće "**elemente**" možemo promijeniti sa:

```
UPDATE studenti SET Ocjena='6',
Komentar='prolazna ocjena' WHERE ID=2;
```

13.18. SQL-naredbe sa PHP-om

Gore navedene SQL naredbe se mogu iskoristiti zajedno sa PHP-om. Za primjer bi trebalo pokazati sve elemente iz tabele studenti u datoteci pod imenom www2. Prije svega mora se uspostaviti veza između PHP-a i servera sa bazom podataka:

```
$veza = @mysql_connect("localhost","korisnik","");
```

localhost znači da se povezujemo sa istim računaram na kojem se nalazi i web-server. U našem slučaju to funkcioniše jer baza podataka i web-server leže na istom računaru. Moguće je povezivanje i sa bazom podataka koja leži na bilo kojem računaru na Internetu - naravno uz uslov da raspoložemo odgovarajućim pravom zahvata na taj računar.

korisnik je važeće korisničko ime za ovu bazu podataka (određeno od strane administratora baze).

"" (navodnici na kraju naredbe u zagradi) znače da ovaj korisnik ne treba lozinku (password).

Da li je veza uspostavljena možemo provjeriti sa:

```
if (!$veza)
{
    echo "Nije moguće uspostaviti vezu sa bazom!\n";
    exit;
}
```

Ako već imamo uspostavljenu vezu, možemo početi sa upitima. Radi bolje preglednosti, upisat ćemo upit u jednu promjenljivu (varijablu):

```
$pitanje = "SELECT Ime, Prezime FROM studenti";
```

Ovaj upit se onda šalje na server sa bazom podataka. Rezultat koji on vrati memorisat će se u promjenljivoj **\$rezultat**:

```
$rezultat = mysql_db_query("www2",$pitanje,$veza);
```

Prva promjenljiva sadržava ime baze podataka koju pitamo, druga sadržava SQL naredbu i treća promjenljiva sadržava vezu koju smo uspostavili sa `mysql_connect`.

Sad slijedi nešto teži dio posla. Treba da pročitamo "redove" koji su upisani u promjenljivoj **\$rezultat**. Da smo koristili neku drugu SQL naredbu (`INSERT`, `UPDATE` ili `DELETE`) bilo bi mnogo jednostavnije. Mogli bismo jednostavno provjeriti je li šta urađeno ili nije, je li naredba izvršena ili nije... Znači odgovor bi bio **ISTINA** (*TRUE*) ili **LAŽ** (*FALSE*). Ali kako naša SQL naredba ima u sebi **SELECT**, mi bismo htjeli i da vidimo ono što smo izabrali. Iz elemenata ("redova") sadržanih u tabeli studenti mi smo tražili da vidimo kolone Ime i Prezime. Sada ćemo ih čitati i to red po red:

```
list($Ime,$Prezime) = mysql_fetch_row($rezultat);
```

Promjenljiva **\$rezultat** poslije upita sadržava dvije kolone (u jednoj je ime, a u drugoj prezime).

```
mysql_fetch_row($rezultat)
```

nam vraća tačno jedan red. Na početku stoji jedna vrsta pokazivača (*pointer*) na prvi red. Izvođenjem naredbe **mysql_fetch_row(\$rezultat)**, ovaj pokazivač će da pokazuje na idući red itd. Ako bismo htjeli da pokažemo sve redove koji su selektirani u tabeli studenti i memorisani u promjenljivu **\$rezultat**, onda bismo to izveli uz pomoć jedne while-petlje:

```
while (list($Ime,$Prezime) =
        mysql_fetch_row($rezultat))
{
    echo "Student se zove $Prezime $Ime <BR>\n";
}
```

Znači, ono što želimo pokazati prebacujemo red po red u HTML tekst. Na kraju je još preporučljivo zatvoriti vezu sa bazom:

```
mysql_close($veza);
```

Ovo bi bile najjednostavnije MySQL naredbe u PHP-u.

13.19. SQL naredba za brisanje (DELETE)

Za naredbu DELETE koja se koristi za brisanje iz baze podataka, važno je uvijek tačno odrediti (i znati) koji element ili elementi treba da se izbrišu. U našem primjeru baze podataka "studenti" bi bilo najbolje i najsigurnije brisati na osnovu ID-a, jer u toj koloni imamo za svaki red jednoznačno određeni broj. Pretpostavimo da imamo ID=120 i da hoćemo da izbrišemo taj red (ovo znači cijeli red koji se nalazi pod tim brojem: ID, Ime, Prezime, ocjenu i komentar):

```
$brisi = "DELETE FROM studenti WHERE ID=120";
```

Pažnja: Ako zaboravimo WHERE dio, onda će biti izbrisani SVI elementi iz tabele.

Prije nego što se počne sa samim pravljenjem dinamičke web-stranice u vezi sa bazom podataka, uvijek se **MORA** dobro razmisliti šta će korisniku biti dozvoljeno, **a šta zabranjeno !!!**

13.20. Funkcije

Jedna od najvažnijih osobina svakog programskog jezika je i sposobnost "*sabiranja*" više naredbi u jednu jedinu naredbu, drugim riječima "*definisanje novih funkcija*". Razumljivo, PHP ima tu osobinu.

Primjer:

Za određeni - upisani broj dana treba izračunati ukupni broj sekundi!

```
function dan ($BrojDana)
{
    $sekunde = 3600 * 24 * $BrojDana;
    return $sekunde;
}

$x = dan(7);
echo "7 dana imaju $x sekundi !";
```

\$BrojDana ovdje predstavlja argument funkcije. Jedna funkcija može sadržavati više argumenata, ali isto tako i ni jedan argument:

```
function KamatniRacun ($procent,$VrijemeRacunanja)
{
$kapital = 10000.0;
$x = 1.0 + ($procent/100.0);
$y = $kapital * exp( $VrijemeRacunanja*log($x) );
return $y;
}

$proc = 6;
$vrijeme = 10;
$x = $KamatniRacun($proc,$vrijeme);
echo "Kapital iznosi $x KM-a";
```

Problematično u ovom navedenom primjeru je da se promjenljiva **\$kapital** ručno polaže, bolje rečeno unaprijed određuje. U normalnom slučaju se i ova vrijednost unosi u vrijeme izvršenja programa, tj. kao ARGUMENT predaje funkciji na obradu:

```
function KamatniRacun
($procent,$VrijemeRacunanja,$kapital) { ...
```

Mi posmatramo slučaj u kojem je i ova varijabla već određena, tj. kada je njena vrijednost upisana od strane programa u funkciju i kada se NE PREDAJE kao argument. U tom slučaju moramo eksplicitno navesti da ova promjenljiva dolazi "*spolja*". To postizemo korištenjem naredbe **global**:

```
$kapital = 10000.0;

function KamatniRacun($procent,$VrijemeRacunanja)
{
global $kapital;

$x = 1.0 + ($procent/100.0);
$y = $kapital * exp( $VrijemeRacunanja*log($x) );
return $y;
}

$proc = 6;
$vrijeme = 10;
$x = $KamatniRacun($proc,$VrijemeRacunanja);
echo "Kapital iznosi $x KM-a";
```

13.21. String-funkcije

Još jedno važno područje za primjenu PHP-a je obrada "*nizova znakova*" (string). Postoji mnogo naredbi za obradu "stringova" - ovdje će biti predstavljene samo najvažnije.

```
$x = " Ja sam jedna dosadna recenica! ";  
$y = trim($x);
```

Naredbe **ltrim()** i **chop()** otklanjaju "*whitespace*" tj "*prazna polja*" prije, odnosno poslije rečenice. Funkcije **strtoupper()** i **strtolower()** mijenjaju sve znakove jedne rečenice u velika, odnosno mala slova. Naredba **ucfirst()** mijenja prvi znak u redu riječi u veliko slovo (naravno, ako je to moguće). Naredba **ucwords()** mijenja u svakoj riječi jedne rečenice prvi znak u veliko slovo (opet ako je moguće).

Sljedeća naredba je vrlo važna kod pripreme teksta za WWW:

```
$x = nl2br($y);
```

Na ovaj način mijenjamo sve znakove za novi red (" \n ") rečenice \$y u naredbu
\n.

13.22. Cookies (kolačići)

Cookies sadržavaju sljedeće informacije:

Name: Ime

Value: Vrijednost

Expires: Datum isteka važnosti

Domain: Dio za adresiranje na serveru, koji cookie može ponovo pročitati

Path: Samo stranice, koje leže u tom direktoriju ili nekom od poddirektorija, smiju biti zatražene od cookie-a.

Secure: Ako je uključena (secure - sigurnost), cookie smije biti prenošen SAMO preko "*zaključanih*" informacijskih kanala.

Primjer: Želimo utvrditi da li je posjetilac već bio na nekoj stranici. U tu svrhu stavit ćemo prilikom prve posjete cookie: Ovo moramo uraditi još u samom zaglavlju HTML stranice (HTML-Head), odnosno na početku teksta. (Obavezno povesti računa da baš nikakav znak - ni prazan znak ne stoji prije ovog PHP-dijela).

```
<?  
$t = time()+3600*24*10;  
SetCookie("mojcookie", "Vec posjeceno", $t,  
          "/test", ".ba");  
?>
```

```
<HTML>
```

```
...
```

Ovom naredbom postavljamo jedan cookie pod imenom **mojcookie** sa vrijednošću "**Vec posjeceno**" sa rokom važnosti od **10 dana**. Čitanje se dozvoljava za svaki server sa nastavkom **.ba**, i to samo iz datoteka koje leže ili u direktoriju **/test** ili nekom od poddirektorija ovog direktorija **/test**. Rok važnosti jednog cookie-ja se daje u sekundama i to od 01.01.1970. Funkcija **time()** vraća sekunde od tog datuma do trenutnog vremena. Kako se vrijeme vrlo rijetko računa u sekundama, onda se koristi nekakva funkcija koja opet računa ovo vrijeme npr. u danima:

```
function dani ($BrojDana)
{
    $sekunde = time() + 3600 * 24 * $BrojDana;
    return $sekunde;
}
SetCookie( "mojcookie", "Vec posjecen", dani(10),
           "/test", ".ba" );
```

Čitanje cookies-a uz pomoć PHP-a se vrlo jednostavno realizuje: Sadržaj cookie-a se memoriše u jednu promjenljivu sa imenom tog cookie-a.

```
echo "$mycookie";
```

također daje riječ iz cookie-ja, u ovom slučaju to je tekst "Vec posjecen". Server može pomoću PHP-a ponovo otvoriti cookie, koristeći ime cookie-a kao argument funkcije:

```
SetCookie("mojcookie");
```

13.23. Promjenljive

U pojedinim slučajevima je potrebno "*pustiti*" petlju preko većeg broja promjenljivih. Npr. to je često slučaj kod obrade formulara u kojem se nalazi više polja za unosenje:

```
<INPUT NAME="polje0" >
<INPUT NAME="polje1" >
<INPUT NAME="polje2" >
...
<INPUT NAME="polje99" >
```

U ovom slučaju bi bilo vrlo naporno pojedinačno "obraditi" sva polja. Ako bismo htjeli pogledati sadržaj svih polja, onda bismo mogli umjesto sljedećih 100 naredbi:

```
echo "polje0: $polje0 <BR>\n";
echo "polje1: $polje1 <BR>\n";
...
echo "polje99: $polje99 <BR>\n";
```

preduzeti sljedeće korake:

Najprije bismo spojili izabrana imena promjenljivih :

```
for ($i=0;$i<100;$i++)
{
    $v = "polje".$i;
    echo "$v: ${$v} <BR>\n";
}
```

Znači sa **\$v** dobijamo imena promjenljivih, sa **\${\$v}** dobijamo zahvat na sadržaj promjenljive sa imenom **\$v**.

13.24. Polja (arrays) iz formulara

Na kraju još jedan trik:

Ako hoćemo u PHP-u da saznamo koji je "*kvadratić*" (check-box) izabran u jednom formularu, onda ne možemo koristiti dosadašnju uobičajenu strategiju:

```
<INPUT TYPE=CHECKBOX NAME="x" VALUE=1 > opcija 1
<INPUT TYPE=CHECKBOX NAME="x" VALUE=2 > opcija 2
...
```

Ako pošaljemo jedan ovakav formular, onda će PHP-promjenljiva **\$x** imati vrijednost prvog kliknutog "*kvadratića*".

Trik je u tome, da se u HTML-u "*kvadratić*" nazove drugačije, npr. ovako:

```
<INPUT TYPE=CHECKBOX NAME="x[]" VALUE=1 > opcija 1
<INPUT TYPE=CHECKBOX NAME="x[]" VALUE=2 > opcija 2
...
```

Na ovaj način dobijamo u PHP-u jedno polje sa imenom **\$x**. Sadržaj možemo pročitati na sljedeći način:

```
for ($i=0;$i<sizeof($x);$i++)
{
    echo "$x[$i] ";
}
```

14. ASP

14.1. Kompromis između CGI i ISAPI

Želimo li interaktivne web aplikacije, možemo koristiti CGI ili ISAPI. Svaki od njih ima problema. CGI koristi previše sistemskih resursa, pa nije prikladan za veći broj korisnika. Ako koristimo ISAPI, riješili smo problem resursa, ali ISAPI aplikacije mogu pisati samo iskusni C/C++ programeri.

Da bi zadovoljio apetite velike većine ostalih programera (npr. VisualBasic), Microsoft je kreirao ASP. ASP je tehnologija koja omogućuje brzo (ne trebate biti C/C++ programer) i efikasno (koristi DLL, a ne proces za svakog korisnika) pisanje web aplikacija. Jednostavno rečeno, ASP koristi skript jezik na strani servera da bi komunicirao sa drugom aplikacijom (koja, npr. obrađuje podatke). Novost kod ASP-a je da se taj skript izvodi na serveru neposredno prije nego li se stranica pošalje korisniku. Zapravo, rezultat izvođenja skripta je nova stranica. To je nešto slično kao predprocesorske naredbe u C/C++.

Vidimo dakle, da kod ASP-a kreiranje HTML stranice i programiranje (skript kod na serveru) postaje jedan te isti proces. Nemamo više posebno HTML stranice, a posebno programe koji povezuju te stranice sa bazama podataka ili drugim aplikacijama. Programiranje je u skript jeziku, npr. VisualBasic skript, pa je jednostavno. Pošto skript zna pozivati COM komponente, moguće je, i poželjno, kompleksne programske cjeline izdvojiti u COM komponente koje mogu biti programirane u praktično bilo kojem programskom jeziku. Na taj način dobivamo savršenu sintezu jednostavnosti (skript) i kompleksnosti (COM komponente).

Naglasimo još jednom cijeli proces:

1. Kreiramo HTML stranicu zajedno sa skript kodom
2. Skript kod se izvodi i kao rezultat nastaje nova stranica
3. Nova stranica se šalje korisniku; korisnik nema pojma da li je ta stranica statička ili je nastala kao rezultat nekog programa koji se izvodi na serveru.

14.2. Kako izgleda ASP stranica?

ASP stranice su vrlo slične HTML stranicama, jer sadrže HTML kod zajedno sa skript kodom. Osnovna razlika je dakle dodatni skript kod. ASP stranice imaju ekstenziju .ASP, dok HTML stranice imaju ekstenziju .HTM ili .HTML. Evo jednostavnog primjera:

```
<HTML>
<HEAD><TITLE>VBScript stranica</TITLE></Head>
<SCRIPT LANGUAGE=VBScript RUNAT=Server>
    MsgBox "ASP je zanimljiv!"
```

```
</SCRIPT>
<BODY>
<BR>
<B>Ovo je jednostavan HTML dokument.<B/>
</BODY>
</HTML>
```

Programski odsječak:

```
<SCRIPT LANGUAGE=VBScript RUNAT=Server>
  MsgBox "ASP je zanimljiv!"
</SCRIPT>
```

naglašava da je to skript kod, da je pisan u VisualBasic skriptu (LANGUAGE = VBScript), te da se izvodi na serveru (RUNAT = Server). Konkretna kod ispisuje tekst "ASP je zanimljiv!".

Kako je većina skript koda u VisualBasic skriptu (barem bi to Microsoft htio) i kako se taj kod uvijek izvodi na serveru, Microsoft je uveo i kraću notaciju za pisanje ovog koda:

```
<%MsgBox "ASP je zanimljiv!" %>
```

koju ćete najčešće sresti kod ASP stranica. Uočite da su delimiteri <% i %> zamijenili cijeli gornji blok. Ako želite da koristite iste delimitere, ali neki drugi skript jezik, npr. JavaScript, morate kao prvu liniju navesti:

```
<%@LANGUAGE=JavaScript%>
```

U tom slučaju morate, naravno, kodirati u JavaScript.

Kako je moguće da koristimo naredbu MsgBox i ispisujemo tekst na serveru? Ispis poruke na serveru ne znači baš ništa korisniku koji je daleko od servera. Radi toga se ovakve komande ne smiju koristiti na serveru. Ako želimo nešto napisati, koristit ćemo HTML format. Kako u našem jednostavnom primjeru imamo samo ispis teksta, on bi postao čista HTML stranica:

```
<HTML>
<HEAD><TITLE>VBScript stranica</TITLE></Head>
<BODY>
<P>ASP je zanimljiv!
<BR><B> Ovo je jednostavan HTML dokument.<B/>
</BODY>
</HTML>
```

HTML stranica se može smatrati kao specijalni slučaj ASP stranice kod koje jednostavno nema nikakvog koda. Server će ovakvu stranicu pročitati, zaključiti da nema potrebe da išta promijeni (jer nema skript koda) i generirati novu stranicu koja je 100% kopija originalne.

Evo sada malo kompliciraniji primjer, gdje ispisujemo poruke u svih sedam fontova:

```
<HTML>

<% For Count = 1 To 7 %>

<FONT SIZE = <%=Count %> >

ASP je zanimljiv <BR>

</FONT>

<% Next %>

</HTML>
```

Ovaj primjer pokazuje kako se ASP kod (podebljano) i HTML format (normalan tekst) mogu miješati na bilo kojem mjestu. Druga linija započinje programski odsječak:

```
<% For Count = 1 To 7 %>
```

gdje se vrti petlja od 1 do 7 (to su dozvoljene veličine fontova na Internetu). Treća linija

```
<FONT SIZE = <% =Count %> >
```

pokazuje da se na svakom mjestu u HTML formatu može ubaciti programski kod; naravno, mora biti unutar delimitera <% i %>. =Count predstavlja vrijednost varijable Count i pridružuje se atributu SIZE. Drugim riječima, specificiramo font. Nakon toga, linija 4 ispisuje tekst, a linija 5 završava font blok. Linija 6 povećava brojač petlje, kao i u normalnom Visual Basicu.

14.3. Kako testirati ASP stranicu?

Ako želite testirati gornje primjere, morate instalirati Internet (Web) server. Točni koraci za instaliranje i testiranje su sljedeći:

1. Instalirajte web server. Web server može biti ili Internet Information Server - IIS ili Personal Web Server - PWS.
2. Spremite gornji primjer u datoteku, npr. asptesl.asp.
3. Ako koristite IIS startajte Internet Service Manager program. Odaberite Default Web Site i dugme Properties. Nakon toga odaberite Home Directory tab i postavite Execute permission. Pritisnite OK i izađite iz Internet Service Manger programa. Ako koristite PWS, startajte Personal Web Manager i

pritisnite ikonu Advanced. Odaberite Home i zatim Edit Properties dugme. Selektirajte Execute item i pritisnite OK. Izadite iz Personal Web Manager.

4. Startajte Internet Explorer (ili Netscape Navigator) i upišite:

`http://ImeServera/asptes1.asp`

"ImeServera" je ime vašeg računara. Možete koristiti ime "Localhost".

5. U Internet Exploreru odaberite View i zatim Source. Vaša stranica bi trebala izgledati ovako:

```
<HTML>
<FONT SIZE = 1> ASP je zanimljiv!<BR> </FONT>
<FONT SIZE = 2> ASP je zanimljiv!<BR> </FONT>
<FONT SIZE = 3> ASP je zanimljiv!<BR> </FONT>
<FONT SIZE = 4> ASP je zanimljiv!<BR> </FONT>
<FONT SIZE = 5> ASP je zanimljiv!<BR> </FONT>
<FONT SIZE = 6> ASP je zanimljiv!<BR> </FONT>
<FONT SIZE = 7> ASP je zanimljiv!<BR> </FONT>
</HTML>
```

Ne zaboravite da je vaša originalna stranica najprije procesirana na strani servera i kao rezultat je generirana gornja stranica.

14.4. ASP objekti

Gornji primjer pokazuje kako početi programirati u ASP-u. Glavna snaga ASP-a proizlazi iz pet objekata koji su ugrađeni u model:

- Application object
- Request object
- Response object
- Server object
- Session object

14.4.1. Application objekt

Karakteristika web aplikacija je da one ne čuvaju stanje. Naime, kad se korisnik spoji na server i dobije željenu stranicu, veza se prekida pa svako pamćenje stanja na serveru nema smisla. To je i normalno, jer ako bi server pamtio stanja od svih korisnika, kojih može biti na hiljade, vrlo brzo bi istrošio memoriju i ostale resurse.

Ako ponekad stanje treba zapamtiti, služi između ostalog i Aplikacijski objekt. Aplikacijskom objektu se mogu specificirati varijable koje mogu koristiti (dijeliti) svi njezini korisnici. Tipičan primjer je brojač koji kaže koliko je korisnika posjetilo stranicu. Program za brojenje posjetilaca može izgledati kako slijedi:

```
<HTML>

...

<%

    Application.Lock

    Application("BrojPosjeta") =
        Application("BrojPosjeta") + 1

    Application.Unlock

%>

Ova strana je posjecena
<% = Application("BrojPosjeta") %> puta!

</HTML>
```

Application objektu može se pridružiti bilo koja željena varijabla na sljedeći način:

```
Application("ImeVarijable")=vrijednost
```

14.4.2. Request objekt

Request objekt služi da bi korisnik mogao predati parametre (vrijednosti) serveru. Radi toga se ponajviše koristi. Korisnik najčešće ispunjava formu da bi specificirao podatke koji se predaju serveru. U primjeru u prošlom broju koristili smo formu iz sljedećeg primjera:

```
<FORM METHOD="POST" ACTION="script/upitnik.idc"><BR>
Ime: <INPUT NAME="prezime"><BR>
Dob: <INPUT NAME="dob"><BR>
Spol: <INPUT NAME="spol" TYPE=RADIO VALUE="m">Muski
      <INPUT NAME="spol" TYPE=RADIO VALUE="z">Zenski
<P>Hvala na odgovorima. </P>
<INPUT TYPE=SUBMIT>
</FORM>
```

Request objekt služi za dohvaćanje upisanih podataka, npr.

```
<% Request.Form("ime") %>
```

ili skraćeno

```
<% Request("ime") %>
```

dohvaća podatak koji smo upisali kao ime. Ako smo kao ime napisali Hana, onda rezultat gornje operacije daje ime Hana. Slično je sa drugim podacima.

14.4.3. Response objekt

Ovaj objekt služi da se pošalje poruka korisniku, najčešće preko kolačića (*Cookies*). Kako Internet aplikacije ne pamte stanja, a ponekad je potrebno da se nešto spremi na strani korisnika, oni dolaze kao spas. Kolačić je podatak kojeg server sprema na strani korisnika (u obliku datoteke koja se kod Internet Explorera sprema u direktorij c:\Windows\Cookies), da bi ga kasnije mogao koristiti. Tako, npr. kolačić može specificirati jezik kojeg korisnik želi. Kako različiti ljudi mogu koristiti različite jezike, ova varijabla ne može biti zajednička za aplikaciju već je specifična za korisnika. Server može specificirati jezik preko kolačića na sljedeći način:

```
<% Response.Cookies("Jezik")="Hindu" %>
```

Kod izvođenja ovog koda server će generirati poseban HTTP header i poslati ga browseru. Kasnije je moguće pročitati vrijednost kukija:

```
Vi ste izabrali <% Request.Cookies("Jezik") %>jezik!
```

14.4.4. Server objekt

Najvažnija funkcija ovog objekta je da omogući kreiranje postojećih COM komponenti. Na taj način programiranje dobiva novu dimenziju, jer možemo povezivati praktički sve što želimo (ne zaboravite da postoji na hiljade gotovih COM komponenti). Tako, npr. ako se želimo povezati na bazu podataka preko ADO (ADO je COM komponenta koja omogućuje povezivanje na postojeću bazu podataka i često se koristi u VisualBasic aplikacijama) učinit ćemo to na slijedeći način:

```
<%  
    Dim dbConn  
    Set db = Server.CreateObject("ADODB.Connection")  
    ...  
%>
```

14.4.5. Session objekt

Ovaj objekt služi za čuvanje stanja (varijabli) koje neki korisnik želi samo za sebe tokom svoje sesije (za razliku od varijabli u Application objektu koje su dostupne svim korisnicima). Tako npr. možemo spremiti ime i dob korisnika u varijable Ime

i Dob i slobodno ih koristiti iz drugih stranica. Ove varijable postaju neaktivne ako zatvorimo Internet Explorer, ili ako smo neaktivni neko vrijeme (obično 20 minuta). Primjer izgleda ovako:

```
<%  
    Session("Korisnik") = "Jasmina"  
    Session("Dob") = 18  
    ...  
%>
```

Ove varijable možemo kasnije koristiti iz drugih ASP stranica:

```
Zdravo <%Session("Korisnik")%>.  
Ti imas <%Session("Dob")%> godina.
```

15. Pristup bazama podataka

15.1. API funkcije za pristup bazama podataka

Da bi aplikacija komunicirala sa bazom podataka, potrebno je određeno "sredstvo". To sredstvo u ovom slučaju su API funkcije za pristup bazama podataka.

Sama baza podataka nije ništa više nego "skladište" koje sadrži odgovarajuću reprezentaciju, odnosno apstrakciju procesa iz stvarnog života. Da bi podaci koji se čuvaju u bazi postali informacija od ikakvog značaja, obično nam je potrebna aplikacija koja zna kako da transformiše te podatke u oblik koji krajnji korisnik želi da vidi. Bilo da je takva aplikacija obični program ili pak web sajt, potrebno je obezbijediti sredstvo pomoću kojeg će aplikacija "pričati" sa bazom podataka. Programskim jezikom to "sredstvo" se naziva database API, a u ovom tekstu ćemo opisati neke od danas najraširenijih načina programabilnog pristupa bazama podataka.

Rastuća popularnost Windowsa (i drugih desktop operativnih sistema), pojava i velika važnost Interneta, te sve kompleksnije tehnologije kao što je OLAP ili Java uzrokovali su pojavu mnogih novih tehnologija za pristup bazama podataka.

Neke od ovih API tehnologija su tokom vremena postale *de facto* ili *de jure* standardne. U hronološkom redu njihovog pojavljivanja najbitniji API-ji su:

- Open Database Connectivity (ODBC),
- Object Linking and Embedding Database (OLE DB),
- Java Database Connectivity (JDBC) i
- ActiveX Data Objects (ADO).

Kao što se vidi, kreatori svih ovih API-ja su Microsoft i JavaSoft. Microsoft još koristi i termin Universal Data Access (UDA) kao krovni termin koji označava ODBC, OLE DB, ADO i Remote Data Service (RDS).

ODBC kao najstariji *de jure* standard je ujedno i najrašireniji i Microsoft i JavaSoft su prilikom pravljenja novijih standarda vodili računa o njihovoj kompatibilnosti sa ODBC. Novi API-ji imaju objekte ili komponente koje putem ODBC adaptera pristupaju SQL bazama podataka. Ipak, unatoč raširenosti ODBC-a, Microsoft i JavaSoft u posljednjih par godina pokušavaju navesti programere da migriraju prema drugim standardima. Logično, obje kompanije žele da se migracija dogodi ka tehnologijama koje su one same stvorile. Tako je Microsoft strategija - pomak ka OLE DB i ADO tehnologijama, dok JavaSoft želi pomak ka Javi i JDBC standardima.

ODBC je low-level SQL API koji omogućava pristup bazama uz pomoć biblioteke C funkcija. Zasnovan je na komponentama izgrađenim pomoću Microsoft COM tehnologije. Pomoću njega programeri mogu raditi sa tabelarnim podacima, kao što

su SQL baze podataka ili sa multidimenzionalnim podacima, kao što su OLAP kočke.

JDBC je stvoren kao objektni omotač pomoću kojeg su Java programeri mogli pristupati SQL bazama podataka. ADO dopušta programerima da koriste objekte podataka. Oni mogu biti raznih vrsta, kao što su SQL baze podataka, preko multidimenzionalne OLAP kočke, Lotus Notes, email sistemi, Active Directory i drugi - ukratko, sve što čuva bilo kakve podatke koji se mogu organizovati na neki način. Programeri koji koriste Active Server Pages (ASP) tehnologiju mogu koristiti preko HTTP i DCOM protokola.

RDS podržava diskonektovane klijente, klijent keširanje i ima mogućnost da sva ažuriranja na jednom klijentu prenese na sve druge distribuirane klijente. Tu je također i ActiveX kontrola pod imenom Advanced Data Control, koja se može postaviti u program ili web stranicu i koja može biti izvor podataka za ostale kontrole na stranici, kao što su liste ili tabele

15.1.1. ODBC

ODBC je najstariji i prvi prihvaćeni de jure database API standard. Dizajniran za maksimalnu interoperabilnost, daje mogućnost pristupa različitim sistemima za upravljanje bazama podataka (DBMS) na uniforman način. Aplikacije za upravljanje bazama podataka pozivaju funkcije u ODBC interfejsu, a ODBC putem svojih drajvera za baze podataka aplikaciji vraća podatke iz baze. Ovi drajveri su analogni drajverima za štampače.

Kada npr. program za obradu teksta nešto štampa, sve komande za štampanje se upućuju operativnom sistemu, koji upotrebljava odgovarajući drajver za željeni štampač. Isto tako, kada aplikacija za obradu podataka želi da pristupi bazi, ona ODBC-u upućuje zahtjev za podacima, a ODBC koristi odgovarajući drajver za pristup toj bazi podataka. Prednost ovakve arhitekture leži u tome da više nije potrebno prepravljati aplikaciju da bismo pristupili bazi podataka, već je samo potrebno instalirati novi drajver u ODBC sistem.

15.1.2. OLE DB

Ništa nije vječno, pa tako ni ODBC. Kada se pojavio, baze podataka su bile relativno jednostavne - homogeni tabelarno organizovani podaci kojima se lako moglo manipulirati pomoću SQL jezika. Ipak, u posljednjih nekoliko godina vidjeli smo ogroman porast u upotrebi baza podataka s jedne strane, a s druge strane ti su podaci postali kompleksniji i heterogeniji – pohranjeni u više organizaciono različitih izvora kao što su tekst datoteke, email sistemi, XML i mnogi drugi. Javio se problem pristupa takvih podacima, a kao odgovor na njega pojavio se OLE DB.

Najjednostavnije rečeno, OLE DB je nova verzija ODBC-a. Najjednostavnije, ali ne u potpunosti tačno! Dok OLE DB koristi ODBC za pristup tradicionalnim

relacionim bazama podataka, ne pokušava ga zamijeniti - u Microsoftu i dalje tvrde da je ODBC najbolji način za pristup SQL bazama podataka. OLE DB nadograđuje ODBC tako što, na jednako uniforman način na koji to radi ODBC sa relacionim bazama, omogućava pristup heterogenim izvorima podataka i vrši njihovu prividnu homogenizaciju, pojednostavljujući tako rad sa podacima.

Microsoft strateški vodi OLE DB kao novu tehnologiju za pristup svim vrstama podataka i kao takav će u budućnosti u potpunosti potisnuti ODBC.

15.1.3. JDBC

JDBC je ekvivalent ODBC tehnologije namijenjen upotrebi prilikom razvoja aplikacija u Java programskom jeziku. Nastao kao direktan odgovor na ODBC, JDBC koristi sličnu arhitekturu zasnovanu na API pozivima i drajverima za razne vrste sistema za upravljanje bazama podataka (DBMS). Interesantno je napomenuti da je Sun, kreator Java platforme, u JDBC ubacio i takozvani JDBC-ODBC "most" - specijalni drajver koji aplikaciji koja koristi JDBC omogućava pristup ODBC izvorima podataka. Obzirom da je većina baza podataka u trenutku pojave JDBC već imala gotove ODBC drajvere, ovo je značilo da Java aplikacije mogu pristupiti svim takvim bazama, iako JDBC drajveri za njih nisu (bili) dostupni.

15.1.4. DAO

ODB, OLE DB i JDBC su tehnologije koje aplikacijama obezbjeđuju pristup bazama podataka. Ipak, njihova upotreba zahtijeva od programera da do u najsitnije detalje ručno isprogramira svaki element pristupa bazi podataka. Ovakav pristup se u vremenu objektno-orijentisanih jezika pokazao nepotrebno komplikovanim i sklonim greškama, te je očito rješenje pronađeno u pravljenju objektnog modela za pristup bazama podataka.

Prvi takav model (barem za Windowse) je DAO (Data Access Objects), koji je sastavni dio VisualBasica već godinama. Originalno zamišljen kao set objekata za pristup MS Access bazama podataka preko MS Jet sistema za upravljanje podacima (Jet je prvi Microsoft DBMS na kojem je zasnovan Access), DAO također omogućava pristup ODBC izvorima podataka. I upravo tu leži njegov najveći nedostatak – pošto se oslanja na Access/Jet DBMS, prilikom pristupa ODBC bazama podataka sve naredbe za bazu podataka is vi podaci iz nje moraju proći kroz ovaj dodatni sloj, što može znatno ugroziti performanse aplikacije.

DAO je lakši za korištenje od standardnih ODBC API funkcija, ali ne pruža isti nivo kontrole kao ODBC APL.

15.1.5. ADO

ADO je nova tehnologija iz Microsofta čiji je cilj da zamijeni DAO kao standardni objektni model za pristup bazama podataka. Kao što smo rekli, DAO može pristupiti samo Access/Jet i ODBC bazama podataka. ADO, s druge strane, koristi OLE DB tehnologiju za pristup izvorima podataka. Kao što je OLE DB evolucija ODBC tehnologije, tako je ADO nadomjestak za DAO objektni model.

Baš kao što je slučaj sa DAO modelom, programiranje pomoću ADO modela ne pruža toliko kontrole kao direktan rad sa OLE DB pozitivima, ali je ADO dizajniran tako da je broj slojeva između aplikacija i baze podataka sveden na apsolutni minimum, što mu daje znatno bolje performanse. Dodatna prednost ADO modela leži u činjenici da je zasnovan na ActiveX tehnologiji, što ga čini upotrebljivim u svim jezicima koji podržavaju tehnologiju, uključujući i skriptne jezike poput VBScripta i JavaScripta, što sam OLE DB nikako ne može postići.

Od svih nabrojanih tehnologija, OLE DB i ADO imaju najsvjetliju budućnost. Microsoft, koji je jedan od glavnih aktera u ovom polju, svoju strategiju zasniva na ovim tehnologijama, što je razumljivo ako uzmemo u obzir njihov potencijal i smjer u kojem se kreću zahtjevi korisnika. ODBC i DAO tehnologije nisu ukinute, iako je sav dalji razvoj na njima obustavljen. Iz Microsofta kažu da će ove dvije tehnologije i dalje biti sastavni dio njihovih razvojnih alata, ali samo zbog kompatibilnosti sa već postojećim kodom, te preporučuju da se nove aplikacije razvijaju pomoću OLE DB i ADO tehnologija.

OLE DB nudi do sada neviđenu snagu i fleksibilnost za razvoj klijenata za baze podataka, ali uz veliku kompleksnost. ADO, pak, nudi fleksibilan i jednostavan objektni model sa pristojnim performansama, što ga u ovom trenutku čini najboljim načinom za razvoj samostalnih aplikacija za rad sa bazama podataka.

15.2. Baze podataka na web sajtovima

Sa server-side skripting jezicima, bazama podataka na serveru i dovoljno znanja možete napraviti moćne dinamične web sajtove bez (mnogo) muke.

Koristeći server-side skripting jezike, najčešće ASP i PHP, možete uljepšati vaš sajt i učiniti ga atraktivnijim. Recimo, možete napisati jednostavan brojač posjeta koji će posjete pamtit u tekstualnu datoteku i prikazivati ih na naslovnoj stranici, skriptu koja će od svih grafičkih datoteka u nekom direktoriju napraviti umanjene verzije (thumbnailove) koji predstavljaju linkove na sliku normalne veličine ili nešto slično. Obzirom da ne koristite ništa drugo do skriptnog jezika i HTML, kod ovakvih skripti jedina briga vam je pronalazak servera koji podržava scripting jezik koji koristite.

Ipak, želite li programirati zaista dinamične web stranice, orat ćete koristiti baze podataka. Snaga server-side skripting jezika dolazi do izražaja tek kada ih uparite sa bazama podataka. Solucija je mnogo i koju ćete od njih koristiti zavisi od toga kakav sajt želite, koliko posjetilaca planirate imati i koji vam je gornji limit sredstava koja možete uložiti. Ako pravite lični web sajt koji će sadržavati

informacije o vama, linkove na razne sajtove koji vam se sviđaju, foto album kojeg osvježavate svakog ljeta, povremeno obnavljanu bazu vaših literarnih dostignuća u vidu kratkih priča i pjesama i slične stvari, logično je da nećete zakupiti poseban (dedicated) server s ogromnim mjesečnim protokom (banda-with) i Microsoft SQL Serverom. Obzirom da je broj potencijalnih posjetilaca ograničen, a sadržaj neobičan, besplatni web server sa podrškom za ASP/PHP skripte i bilo koje baze podataka biće sasvim dovoljan. Ako pak namjeravate praviti e-zine, news sajt, pretraživač, web knjižaru ili bilo koji drugi sajt koji će imati dosta sadržaja i više od deset posjetilaca dnevno, situacija je potpuno drugačija. Ako programirate za Windows platformu, Microsoft SQL Server je imperativ, jer će baze kojim pristupate preko Access enginea početi štekati već nakon nekoliko simultanih upita. Za Linux platformu MySQL je najbolje rješenje - besplatan je, a performanse su mu u rangu već spomenutog MS SQL-a. Mada nije uobičajeno, ASP se može sasvim fino upariti i sa MySQL bazama, bez mnogo nekompatibilnosti i degradacije performansi.

15.2.1. Windows vs. LINUX vs. ASP vs. PHP

Iako se za razvoj web aplikacija mogu koristiti mnogi skriptni i programski jezici (Perl, Python, C), većina programera odlučuje se na PHP ili ASP. Koji od ova dva skriptna jezika je bolji još jedna je od vječitih rasprava koje se uglavnom vode na subjektivnoj razini.

ASP će kvalitetno raditi jedino na Windows platformi, iako postoje proizvođači koji će vam pokušati prodati Apache modul koji ASP-u donosi podršku i za Unix. Kad kažemo ASP, mislimo na ASP stranice napisane u VBscriptu, što je uobičajeni, ali ne i jedini način pisanja ASP aplikacija. Sa druge strane, PHP postoji i za Windows, gdje je donekle lišen funkcionalnosti (neke funkcije, doduše rijetke, raditi će samo na Unix verziji Apache web servera).

Za učenje PHP-a, Apache za Windowse ili pak PHP modul za PWS (Personal Web Server) ili IIS (Internet Information Server) poslužiti će sasvim dobro.

Oba skriptna jezika zapravo su jednostavni, no o vašem prethodnom znanju ovisi koji ćete lakše usvojiti. Na primjer, ako ste imali iskustva sa Visual Basicom, ASP doslovno nećete morati učiti. Programeri sa stažom u C-u ili Perlu, PHP će naći mnogo jednostavnijim.

Prednosti PHP-a prema ASP-u je u veća osnovna funkcionalnost (veći broj ugrađenih funkcija), no ASP za naprednije korisnike donosi veću skalabilnost zbog mogućnosti jednostavnog razvoja vlastitih serverskih komponenti, koje se jednostavno inicijaliziraju i koriste iz ASP koda. Primijetit ćete da uglavnom morate imati vlastiti Windows server za instaliranje dodatnih komponenti, što je u BiH prije izuzetak nego pravilo. Mnogo korisnih komponenti može se naći na Internetu i besplatno koristiti.

Na kraju se izbor skriptnog jezika svede na izbor serverske platforme. Naši Internet provajderi obično nude obje varijante, pri čemu je Linux varijanta po pravilu jeftinija.

15.3. Spajanje na bazu podataka

OK, znamo kakav sajt želimo, kakve baze podataka ćemo koristiti, ali kako se baze podataka, odnosno podaci iz njih, koriste na web stranicama? Prvo treba napraviti konekciju između baze i web stranice, odnosno spojiti se na bazu podataka. Za to postoji više načina na Windows i Linux platformama, a međusobno se razlikuju po mnogo čemu, a najbitnije su složenost i brzina pristupa.

Spajanje na baze bilo kojeg formata bilo iz ASP ili PHP-a u Windows OS-u svodi se na ODBC, ali nije nimalo svejedno koji način koristite.

Najjednostavniji i ujedno najgori način za spajanje na bazu podataka je preko sistemskog DSN-a. Njegova osnovna prednost jeste da na serveru, u ODBC Data Source Administratoru, definišete sve potrebne podatke - format, put ili ime baze podataka, koji se spremaju u Registry. Obzirom da ste sve već odredili, u skripti je potrebno samo navesti naziv DSN-a i eventualno login i password, ako je baza zaštićena. Ova prednost ujedno je i veliki nedostatak, jer ćete rijetko biti u stanju sami praviti i brisati sistemske DSN-ove na serveru. Osim ako nemate vlastiti Windows server u uredu, čime se može pohvaliti malo domaćih firmi. Drugi nedostatak ovakvog pristupanja bazama jeste sporost spajanja na bazu i manipulacija podacima (uzimanje, upisivanje, brisanje, pretraživanje i slično), bez obzira da li koristite brzi Microsoft SQL Server ili spori Access engine. Generalno, spajanje na baze podataka pomoću sistemskih DSN-ova preporučuje se samo početnicima i to samo za učenje, nikako za projekte koje namjeravaju postavljati na Internet.

Nešto bolja solucija bi bila tzv. DSN-less metoda, koja funkcioniše identično prethodno opisanoj DSN metodi, samo što server prilikom izvršavanja skripti ne uzima podatke iz Registry baze, nego ih vi upisujete direktno u kodu. Drugačije rečeno, umjesto da u finom interfejsu ODBC Data Source Administratora definišete format i naziv baze, to radite pišući komande u skripti. Razlike u konekcijskom stringu nisu velike, a možete ih vidjeti u tabeli. Prednost ovakvog pristupa bazama ogleda se u nešto većoj brzini i činjenici da ne morate imati direktan pristup serveru da biste dodavali ili brisali baze podataka.

Izvorni kod spajanja na bazu u ASP-u

DSN Metoda spajanja na bazu u ASP-u:	<pre>Set baza = Server.CreateObject ("ADODB.Connection") Baza.Open "DSN=naziv_dsn; UID=username; PWD=password"</pre>
DSN-less metoda spajanja na Accessovu bazu podataka u ASP-u:	<pre>Set baza = Server.CreateObject ("ADODB.Connection") Baza.Open "DRIVER={Microsoft Access Driver (*.mdb) }; DBQ=" & Server.MapPath ("naziv_baze.mdb") & "; UID=username; PWD=password"</pre>
DSN-less metoda spajanja na Microsoft SQL bazu podataka u ASP-u:	<pre>Set baza = Server.CreateObject ("ADODB.Connection") Baza.Open "DRIVER={SQL Server}; Server=ime_servera; Database=ime_baze; UID=username; PWD=password;</pre>
OLEDB metoda spajanja na Access bazu podataka u ASP-u:	<pre>Set baza = Server.CreateObject ("ADODB.Connection") Baza.Open "Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" & Server.MapPath ("naziv_baze.mdb") & "; User ID=admin; Password=";</pre>
OLEDB metoda spajanja na Microsoft SQL bazu podataka u ASP-u:	<pre>Set baza = Server.CreateObject ("ADODB.Connection") Baza.Open "Provider=SQLOLEDB; Data Source=ime_servera; Initial Catalog=ime_baze; User ID=username; Password=password;"</pre>
DSN metoda spajanja na bazu podataka u PHP-u:	<pre>\$baza = odbc_connect ('SYSTEM_DSN', 'username', 'password');</pre>
DSN-less metoda spajanja na Access bazu podataka iz PHP-a (Windows):	<pre>\$baza = new COM ("ADODB.Connection"); \$koneksi = "DRIVER={Microsoft Access Driver (*.mdb) }"; \$koneksi .= "DBQ="baza_podataka.mdb;uid=username;pwd=password;"; \$baza->open (\$koneksi);</pre>
DSN-less metoda spajanja na MySQL bazu preko ASP-a:	<pre>Konekcija = "driver={MySQL}; server=host; uid=username; pwd=password;database=baza" Set baza = Server.CreateObject ("ADODB.Connection") Baza.open Konekcija</pre>
Spajanje na MySQL bazu iz PHP-a (Unix):	<pre>\$link = mysql_connect ("host", "username", "password") Or die ("Neuspjesno spajanje na server"); Mysql_select_db ("baza")</pre>

Prethodno opisani način pristupa bazama se podrazumijevaju u ASP-u, ali nisu preporučljiv za sajtove sa većim brojem korisnika, odnosno sajtove koji puno manipulišu sa podacima u bazama. Mnogo bolji način je korištenje OLEDB objekta, čime preskačete jednu stepenicu u složenom sistemu komunikacije sa bazama podataka u Windowsima. Naime, sva komunikacija sa bazama podataka se svodi na ODBC, koji predstavlja prvi sloj. Iznad njega nalazi se OLEDB objekat, a na vrhu se nalazi ADO, koji se koristi u dva prethodno opisana načina. Direktno spajanje na OLEDB, bez posredovanja ADO-a, donosi značajne uštede u vremenu, što predstavlja dovoljan razlog za korištenje ove, ipak nešto složenije metode. Primjerice, za spajanje na bazu pomoću DSN metode potrebne su 82 milisekunde, a korištenjem OLEDB-a samo 18 milisekundi? Za ispis hiljade slogova iz baze pomoću prve metode trebaju 5,4 sekunde, dok ćete to isto sa OLEDB-om odraditi za 2,9 sekunde.

15.4. Manipulacija podacima

Dinamičke web stranice su napravljene upravo kako bi se prikazali određeni podaci, unijeli novi, izmijenili ili obrisali postojeći, jednom riječju - kako bi se manipuliralo podacima. Bez obzira koji skripting jezik koristili, kojeg je formata baza u kojoj se nalaze podaci ili koji način spajanja na bazu ste izabrali, za manipulaciju podacima se koristi jezik pod imenom SQL (Structured Query Language). Iz imena se već da naslutiti kako se koristi. Vi pišete upit korištenjem odgovarajućih komandi, zatim ga pošaljete bazi podataka i dobijete "odgovor" u obliku podataka iz baze. SQL je standard koji se koristi za pristup bilo kojoj bazi podataka, iz bilo kojeg programskog jezika na bilo kojoj platformi.

Generalno, dinamičke web stranice ćete najviše koristiti za prikaz podataka iz baze. Bilo da se radi o tekstovima, albumima slika, informacijama o knjigama, linkovima ili bilo kojem drugom sadržaju, princip je isti - potrebno je napraviti jednu (ili nekoliko, u zavisnosti od potreba i specifičnosti sajta) generičkih web stranica koje na osnovu parametara uzimaju potrebne podatke iz baze podataka i prikazuju ih posjetiocu. HTML kostur je isti, dok se sadržaji dinamički ubacuju iz baze na serveru i šalju korisniku ukomponovani u web stranicu.

Mijenjanje i brisanje podataka najčešće ne rade korisnici, nego administratori web sajta. Podatke možete dodavati, mijenjati i brisati i pomoću nekog klijentskog softvera, recimo Accessa, pa onda izmijenjenu bazu podataka uploadovati na server, ali postoje mnogo elegantnija rješenja. Svaki dinamički web sajt ima svoj administracijski dio, namijenjen samo određenom (malom) broju osoba, koji omogućava mijenjanje sadržaja baza direktno preko web-a - korištenjem HTML formi. Ako pak trebate omogućiti korisnicima da mijenjaju određene podatke u bazama, koristit ćete isti način, samo što ovaj put trebate voditi računa da stranice budu intuitivne i user-friendly. Kada smo kod dodavanja i editovanja podataka, jedno upozorenje programerima u ASP-u: bez obzira koju bazu podataka i konekciju koristite, sve manipulacije podacima radite isključivo i samo pomoću

SQL upita. Mada će vam se možda učiniti jednostavnijim koristiti ADO komande (ako koristite DSN ili DSN-less pristup bazama), vjerujte da će vam se to odbiti od glavu pri eventualnom prelasku na SQL server. A i performanse su mnogo bolje kada koristite SQL upite.

15.5. Korisni linkovi

<http://www.aspin.com/home/tutorial/database>

Mnogo linkova na razne tekstove, tutorijale i primjere za rad sa bazama podataka iz ASP-a.

<http://www.planet-source-code.com>

Ogroman web sajt na kome ćete pronaći mnogo primjera za rad sa bazama podataka u ASP-u i PHP-u, kao i primjere za razne SQL funkcije.

<http://www.aspl01.com/resources/>

Između dosta drugih članaka, na ovome sajtu možete pronaći i nekoliko kvalitetnih tutorijala za pristup bazama podataka iz ASP-a.

<http://www.phpbuilder.com/columns/>

Ovaj web sajt sadrži prilično obimnu sekciju koja se bavi bazama podataka u vezi sa PHP-om.

<http://www.weberdev.com/>

PHP-orijentiran web sajt koji ima prilično veliku arhivu tekstova posvećenih mySQL-u, Microsoft SQL serveru, ODBC-u, Informixu... i njihovom kombinovanju sa PHP-om.

<http://www.zend.com>

Odličan sajt posvećen razvoju PHP aplikacija

<http://www.php-resources.com>

Još jedan sajt posvećen PHP-u.

15.6. Razvoj PHP & MySQL aplikacije korištenjem Dreamweavera

Dreamweaver, pored podrške za PHP, ima i podršku za ostale web tehnologije (Cold Fusion, ASP, ASP.NET...), od čega je najizraženija podrška za ColdFusion, što je i logično, jer je Cold Fusion web serverska tehnologija razvijena od strane Macromedia-e.

15.6.1. Priprema

Potrebno je odlučiti kakvu ćemo aplikaciju izrađivati. Kako su novosti najčešće ažuriran sadržaj na većini web sajtova, uz pomoć DW-a, kreirat ćemo sistem za upravljanje novostima, simbolično nazvan infoNews, koji se kasnije lako može modificirati u knjigu gostiju ili skromniji CMS, infoNews će biti potpuno upotrebljiva news aplikacija, koja će se sastojati iz administracijskog (mogućnost unosa, izmjene, brisanja novosti, autorizacije korisnika) i korisničkog dijela (prilagodljiv prikaz novosti, pretraga...), a krajnja instalacija i upotreba će biti jednostavna. U proces pripreme bi trebala biti uključena izrada i osmišljavanje minimalističkog dizajna infoNews aplikacije.

15.6.2. Tabele

Planiranje pri izradi aplikacija je važan korak. Prvenstveno za potrebe infoNews aplikacije, trebamo napraviti plan baze podataka, tj. tabela. Za ovaj primjer potrebne su nam dvije tabele: sadržaj i kategorije. Tabela sadržaj će se koristiti za pohranjivanje podataka o novostima (naslov, kategorija, uvodni i cijeli tekst, datum), a tabela kategorije za pohranjivanje kategorija novosti (npr. Sport, Zabava, Edukacija, Tehnologija...).

Potrebno je imati tabelu kategorije, kao i kolonu kategorije u okviru tabele sadržaj. Razlog za to je što ovakva struktura omogućava lakše sortiranje i prikaz novosti iz baze.

Pokrenite phpMyAdmin (ili neki drugi alat za upravljanje MySQL bazom podataka) i kreirajte bazu imenujući je infoNews. Zatim, koristeći SQL opciju u phpMyAdmin-u, izvršite slijedeće SQL kodove, koji će u bazi infoNews kreirati dvije tabele, sadržaj i kategorije:

```
CREATE TABLE 'kategorije' (  
    'Id' int(11) unsigned NOT Null auto_increment,  
    'naziv' varchar(50) NOT NULL default",  
    PRIMARY KEY ('Id')  
) TYPE=MyISAM;  
  
CREATE TABLE 'sadrzaj' (  
    'Id' int(11) unsigned NOT Null auto_increment,  
    'kategorija' varchar(50) NOT NULL default",  
    'naslov' varchar(200) NOT NULL default",  
    'utekst' longtext NOT NULL,  
    'ctekst' longtext NULL,  
    'datum' varchar(10) NOT NULL default",
```

```

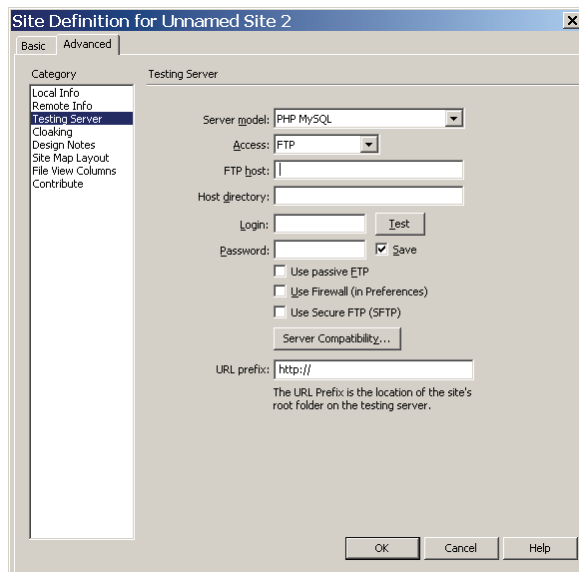
        UNIQUE KEY 'Id' ('Id')
    ) TYPE=MyISAM;

```

15.6.3. Definisanje sajta

Prije nego počnemo razvijati našu aplikaciju, potrebno je definisati postavke sajta. Definisanje je nešto drugačije od klasičnog, tj. definisanja sajta postavki pri izradi statičnih stranica, jer sada je potrebno odabrati koju tehnologiju ćemo koristiti pri izradi dinamičnih stranica - u našem slučaju PHP/MySQL, sajt direktorij - direktorij u koji se smještaju svi dokumenti vezani za sajt koji se razvija, podešavanje lokalnog servera...

Za početak, kreirajte PHP dokument i definirajte njegov naslov (Title) upisujući Infonews - Unos novosti koristeći Document toolbar. Na glavnom izborniku izaberite Site/Manage Sites i na vašem ekranu će se pojaviti Manage Sites prozor. Kliknite na dugme New i izaberite Site. Pojavit će se novi prozor Site Definition, u kojem definirate postavke sajta



Slika 89. Izbor serverske tehnologije za razvoj sajta

15.6.4. Konekcija na bazu

Da bi imali pristup podacima u bazi, ili imali mogućnost upisa, brisanja i izmjene podataka, potrebno je da uspostavimo konekciju (vezu) sa bazom. Pored navođenja imena baze, potrebno je navesti i korisničko ime, password, naziv konekcije...

15.6.5. Unos novosti

Kreirajte novi PHP dokument i implementirajte dizajn, i snimite ga sa nazivom unos_novosti.php. Potrebno je kreirati Insert Record, a to ćemo uraditi koristeći Record Insertion Form Wizard, koji će kreirati formu za unos, i generisati potrebni PHP kod i SQL upit. Prije nego krenemo sa radom na formi za unos, potrebno je kreirati jedan Recordset (Query). On će pokupiti sve kategorije iz tabele kategorije koje ćemo dinamički prikazati u listi kategorija. U okviru Application panela kliknite na Bindings/Znak +/Recordset (Query). Na ekranu će se pojaviti prozor za kreiranje Recordset-a u Simple modu. U polje Name upišite naziv Recordset-a (pokupi_kategorije), a pod Connection listom izaberite ranije definisanu infoNews konekciju. Nakon toga kliknite na dugme OK, čime završavate kreiranje upita na bazu.

Idući korak je da napravimo formu za unos novosti u bazu. U okviru Document toolbar-a izaberite Application, te kliknite na Record Insertion Form Wizard i na ekranu će se pojaviti prozor kao na slici 5. Pod listama Connection i Table izaberite infoNews, odnosno sadržaj. Koristeći dugme sa znakom minus u okviru Form fields panela izbrišite id kolonu. Zatim selektirajte kolonu kategorija i pod listom Display As izaberite menu. Nakon toga, pojavit će se dugme Menu Properties. Kliknite na njega i zatim izaberite From Databases. Pod listom Get Labels from izaberite naziv, a nakon toga kliknite na OK, ostavljajući podrazumijevane vrijednosti preostalih polja i listi.

Kolonama utekst i stekst pod Display As listama izaberite Text area, te Labele promijenite u Uvodni tekst i Cijeli tekst. Još je preostalo da koloni datum pod Display As izaberete Hidden Filed i u polje Default value upišete slijedeći PHP kod

```
<?php.echo (date("d.m.y.")).">
```

koji će pokazati vrijeme. Klikom na OK završavamo sa kreiranjem forme za unos novosti, koja je već spremna za testiranje. Ali, prije nego to učinite, kreirat ćemo JavaScript provjeru (validaciju) polja u formi. Pozovite Behaviors Panel kombinacijom tipki Shift+F3 i zatim označite dugme Insert Record, te kliknite na znak +/Validate Form. Nakon toga poljima naslov i utekst uključite opciju Required, ostavljajući izabranu opciju Anything.

Testirajte napravljeni dio aplikacije pritiskom na tipku F12, a rezultate unosa u bazu provjerite pomoću phpMyadmin-a, tražeći upis koji ste unijeli preko forme.

15.6.6. Zaštita administrativnog dijela baze

Zaštita se obavlja autorizacijom korisnika putem PHP sesija. Pristup administrativnom dijelu aplikacije treba omogućiti samo osobama koje administrator odredi, pa trebamo uvesti neki vid autorizacije.

Autorizacija korisnika ogleda se u slijedećem: administrator aplikacije kreira korisnički account, sa korisničkim imenom i šifrom, te rangom, odnosno nivoom privilegija korisnika pri korištenju skripte, nakon čega se korisnik prijavi sa ispravnim podacima. Generalno, autorizaciju korisnika u PHP-u je moguće realizirati na dva načina - pomoću Cookies-a (kolačića) ili sesija (session). Realizacija autorizacije korisnika preko sesija je puno sigurnija i efikasnija od autorizacije korisnika preko cookies-a. Kako su cookies usko povezani sa postavkama web browser-a, a podaci (npr. korisničko ime i šifra) se zapisuju na hard disk posjetioca, to predstavlja veliku nesigurnost i opasnost. Sesije su sigurnije, pa su izbor većine PHP programera. Dreamweaver MX omogućava pravljenje autorizacije preko sesija, tako da je logičan slijed da odaberemo ovaj put. Inače, postoji i kombinovana solucija, odnosno moguće je napraviti autorizaciju koristeći se i sesijama i cookies-ima, kako bi se povećala sigurnost aplikacije. Potrebe naše aplikacije to ne zahtijevaju, pa ćemo se opredijeliti za sesije.

Dakle, administrator aplikacije kreira novog korisnika aplikacije, upisujući njegovo korisničko ime i šifru, te rang. Naravno, sve to treba pohraniti u neku od tabela iz baze. Nama je potrebna nova tabela po imenu korisnici, sa četiri kolone (id, korisime, sifra, rang). Koristeći SQL opciju u phpMyAdmin-u (ili nekom drugom alatu za upravljanje MySQL bazom podataka), izvršite slijedeći SQL kod, koji će u bazi infoNews kreirati tabelu korisnici sa svim potrebnim kolonama:

```
CREATE TABLE 'korisnici' (  
  'id' int(11) NOT NULL auto_increment,  
  'korisime' varchar(20) NOT NULL default "",  
  'sifra' varchar(20) NOT NULL default "",  
  'rang' varchar(20) NOT NULL default "",  
  PRIMARY KEY 'korisime' ('korisime')  
) TYPE=MyISAM;
```

Napominjemo da smo koloni korisime dodijelili osobinu unique, jer se vrijednost u ovoj koloni ne smije ponavljati. To znači da u bazi ne možemo imati korisnike sa istim korisničkim imenom, što je i logično.

15.6.7. Kreiranje novih korisnika

Prvi korak u implementiranju autorizacije je da omogućimo kreiranje novih korisnika aplikacije. Naravno, za to nam je potrebna forma za unos, a nju, kao i sav potrebni PHP kod ćemo napisati koristeći se Record Insertion Form Wizard opcijom u okviru Application toolbar-a.

Kreirajte novu .php datoteku i imenujte je korisnici.php. U njoj će biti smještena forma za kreiranje korisnika, kao i lista svih kreiranih korisnika. Primijenite

šablone za dizajn i pozicionirajte kursor miša u površinu šablona koja se može uređivati. U okviru Application toolbar-a kliknite na Record Insertion Form Wizard - na ekranu će se pojaviti već dobro poznati prozor. Uspostavite konekciju sa bazom izborom infoNews u okviru liste Connection. Kako upisujemo informacije u tek kreiranu tabelu korisnici, pod listom Table izaberite opciju korisnici. U polje After inserting, go to navedite putanju ili klikom na dugme Browse izaberite datoteku korisnici.php, jer kako vam je već poznato, nakon izvršenja ovog dijela aplikacije, želimo da se izvrši redirekcija na spomenutu datoteku.

U okvirima Form Fields panela, koristeći se dugmetom sa oznakom minus (-), uklonite id kolonu, a zatim označite kolonu rang i pod Display as listom izaberite Menu. Zatim kliknite na dugme Menu Properties. Pojavit će se prozor koji služi za definisanje ponuđenih opcija jedne liste, u našem slučaju Rang liste, koja će imati dvije opcije: Administrator i Editor. Pošto je u pitanju lista čije opcije će se rijetko ili skoro nikako mijenjati, ostavite označeno Manually, a koristeći se Label i Value poljima, dodajte slijedeće opcije u listu: Administrator (sa vrijednošću admin) i Editor (sa vrijednošću editor). Korisnik sa rangom "editor" neće imati pristup modulima Postavke i Korisnici, odnosno neće imati ovlaštenje da pravi ili izmijeni novu kategoriju ili korisnika, dok će administrator imati sva ovlaštenja pri korištenju aplikacije. Kliknite na OK, a zatim još jedanput, kako bi završili sa kreiranjem forme za unos novih korisnika.

Idući korak je da kreiramo Recordset koji će prikazati sve kreirane korisnike. Kliknite na znak plus (+) na Application panelu, zatim izaberite Recordset (Query). U polje Name upišite prikazi_korisnike. Uspostavite konekciju sa bazom, a zatim pod listom Table izaberite korisnici. U okviru Columns panela označite Selected i selektujte (koristeći se tipkom Shift i mišem) kolone id, korisime, te rang. Za prikaz korisnika nam nije potrebna njegova šifra, pa nemamo razloga da je označimo ovim Recordset-om. Kliknite na OK, kako bi završili sa kreiranjem Recordset-a po imenu prikazi_korisnike.

Nakon uspješno kreiranog Recordset-a, idući korak je da prikažemo podatke (tačnije, korisnike) koje će prethodno kreirati Recordset označiti iz baze. Kliknite na opciju Dynamic Table u okviru Application toolbar-a (druga sa lijeve strane), a zatim u novootvorenom prozoru označite All Records, a u polja Border, Cell padding i Cell spacing, upišite 0, 2, 2.

Kliknite na OK, i po potrebi dodatno formatirajte tabelu u kojoj se prikazuju korisnici. Da bi u potpunosti kompletirali ovaj modul, potrebno je još omogućiti izmjenu i brisanje korisnika. Kako smo izmjenu i brisanje podataka objasnili već dva puta (brisanje i izmjena novosti, te kategorija), ostavljamo vam da sami kreirate dijelove skripte koji će vam omogućiti da izmijenite ili izbrišete korisnika. Logika i koraci su gotovo identični.

Prije nego krenemo sa realizacijom login skripte, testirajte kreiranje novih korisnika i upamtite podatke koje ste unijeli, jer će vam isti biti potrebni prilikom testiranja autorizacije.

15.6.8. Login forma

Počnemo sa konkretnom implementacijom autorizacije korisnika. Prvi korak u tome je da kreiramo jednostavnu login formu. Ona će se sastojati iz dva Text-Field-a (za korisničko ime i šifru) i jednog dugmeta (Button). Kreirajte novu .php datoteku i spasite pod imenom login.php. Ubacite formu, tačnije form tag, klikom na Insert (glavni izbornik), te na Form/Form. Zatim pozicionirajte kursor u okviru tek ubačene forme i ubacite dva TextFiled-a slijedećim koracima: Insert/Form/Text Field. Označite prvi i u okviru Properties panela ga imenujte sa korisime. Isto učinite i za drugi TextFiled, samo njega imenujte sa sifra. Na glavnom meniju kliknite na Insert / Form / Button, kao bi ubacili dugme za akciju, čiji ćete Label promijeniti u Prijavi me.

Sada imamo jednostavnu formu koja ne radi ništa. Trebamo je "oživjeti" pomoću opcije Log IN User, posljednje na Application toolbar-u. Označite dugme Prijavi me i kliknite na Log In User na Application toolbaru.

Get input from form lista služi za definisanje iz koje forme će aplikacija koristiti ulazne podatke. Pošto je kreirana samo jedna forma (čiji je naziv form1), spomenutu listu ostavite nepromijenjenom. U listi Username field izaberite opciju korisime, jer je to polje koje ćemo koristiti za unos korisničkog imena. Lista Password field služi za definisanje polja u koje se upisuje šifra za pristup. Kako smo mi naše polje za šifru imenovali sa sifra, logično je da izaberemo opciju sifra u spomenutoj listi. Naravno, potrebno je uspostaviti konekciju sa bazom, što možete učiniti izborom opcije infoNews u listi Validate using connection. Naravno, pod Table listom izaberite tabelu korisnici, jer aplikacija provjerava da li uneseni podaci postoje u toj tabeli. Stoga je potrebno definisati kolonu u kojoj se nalazi korisničko ime i kolonu u kojoj se nalazi šifra za pristup. To možemo uraditi koristeći se listama Username column i Password column, izborom odgovarajućih opcija (korisime i sifra).

Potrebno je navesti putanju do stranice na koju će skripta izvršiti redirekciju ukoliko su uneseni podaci pronađeni u bazi, kao i stranicu na koju će skripta izvršiti redirekciju, ukoliko je login bio neuspješan.

Kreirajte .php datoteku i nazovite je home.php, te primijenite predloške za dizajn. Radi se o stranici na koju će skripta preusmjeriti korisnika ukoliko login bude uspješan. Ujedno, to će biti i stranica u kojoj će se prikazivati statistički podaci (broj novosti, kategorija i kroisnika u bazi) aplikacije. Navedite putanju do datoteke home.php, upisujući je u polje if login succeeds, go to. U polje if login fails, go to navedite putanju do datoteke login.php, jer želimo da se korisniku ponovo prikaže forma za login, ukoliko je isti bio neuspješan. Još je samo preostalo da definišemo privilegije za pristup koji će biti realizovan na osnovu dva

spomenuta ranga korisnika, a to ćemo učiniti označavanjem opcije Username, password, and access level i izborom opcije rang u listi Get level from. Kliknite na OK, čime se završava pravljenje login skripte.

15.6.9. Ograničenje pristupa stranicama

Idući korak je da zabranimo pristup .php datotekama koje sačinjavaju administracijski dio aplikacije. Kreiranje restrikcije na stranice je jednostavno, a za to ćemo iskoristiti opciju Restrict Access To Page. Dakle, sve .php datoteke je potrebno zasebno otvoriti i izvršiti restrikciju, odnosno zabranu pristupa. Napominjemo da imamo dva ranga, admin i editor, te da admin ima pristup svim stranicama administracije, za razliku od editora, koji nema pristup modulima Postavke i Korisnici.

Prije svega, kreirajte obični .html dokument sa slijedećom porukom; "Nažalost pristup željenom modulu nije omogućen" i snimite ga pod imenom poruka.html. Radi se o stranici koja će se prikazati korisniku koji pokušava pristupiti modulu za koji nema dovoljno visok rang. Otvorite stranicu home.php, jer je to stranica na kojoj prvo treba izvršiti zabranu pristupa, tačnije provjeru autorizacije. Potrebno je definirati koji rang korisnika ima pristup izabranoj stranici. Označite opciju Username, password and access level, zatim kliknite na dugme Define i dodajte dva već ranije spominjana ranga (nivoa pristupa), pa kliknite na OK. Koristeći se tipkom Shift i mišem, označite oba ranga i klikom na dugme Browse, pored polja If access denied, go to, izaberite poruka.html datoteku, jer želimo da se korisniku prikaže poruka o nemogućnosti pristupa željenom dijelu aplikacije.

Ove korake je potrebno ponoviti za sve .php datoteke koje sačinjavaju administraciju, sa izuzecima datoteka koje sačinjavaju module Postavke i Korisnici, jer spomenutim modulima ima pristup samo korisnik sa rangom admin. Stoga, kada budete definisali restrikcije tim stranicama, ne zaboravite označiti samo admin rang.

15.6.10. Log out

Kako bi povećali sigurnost aplikacije, neophodno je napraviti i omogućiti odjavu (Log out) korisnika, sa ciljem sprečavanja mogućnosti zloupotrebe korisnika koji dijele isti računar. Naime, log out se bazira na poništavanju svih kreiranih session varijabli, da bi se pri ponovnom pokušaju logiranja zahtijevalo njihovo ponovno kreiranje. Otvorite predložak za dizajn i označite tekst Izlaz u navigaciji, pa kliknite na opciju Log Out User. Pojavit će se prozor na kojem je potrebno u listi Link clicked izabrati Selection: Izlaz i u polju When done, go to navesti putanju do login.php datoteke.

16. .NET

16.1. Šta je .NET Framework?

U posljednjih nekoliko godina, značaj Interneta je enormno porastao u gotovo svim oblastima računarstva i obrade informacija. Direktan rezultat ovoga rasta je taj što se od programera traži da kreiraju aplikacije koje koriste mogućnosti Interneta, bilo da je to proces naručivanja, isporuka robe, online saradnja ili bilo koja druga upotreba. Nažalost, razvojni alati koji su ranije bili dostupni programerima, kao što je Visual C++ 6 i Visual Basic 6, potiču iz pre-Internet faze. Bilo koja Internet mogućnost koju su imali je bila rezultat naknadnog dodavanja, koje je trebalo da zadovolji potrebe programera. Neke ranije tehnologije Internet programiranja su zapravo bile impresivne i mnoge odlične aplikacije su bile kreirane sa njima. Uprkos tome, to što Internet mogućnosti nisu bile ugrađene u ove alate za programiranje od samog početka vodilo je do nezaobilaznih problema kod efikasnosti pri razvoju, greškama i održavanju programa.

Microsoft-ov odgovor na ovu dilemu je da počne od početka, kreirajući novi strukturni okvir programerskih alata koji je dizajniran od prvog dana sa potpuno integrisanom Internet podrškom. .NET Framework se ne bavi samo Internet programiranjem; on također obezbeđuje razvoj tradicionalnih desktop aplikacija.

.NET ima dva glavna dijela:

- .NET Framework Software Development Kit, proširivi set klasa i interfejsa, zajedno sa raznim elementima podrške, je dizajniran da radi zajedno kako bi zadovoljio svaku razvojnu potrebu. .NET Framework mora biti instaliran na bilo kom sistemu koji će biti korišten za pokretanje .NET aplikacija ili za njihov razvoj. .NET SDK uključuje kompajlere za tri jezika: C++, novi jezik nazvan C# (C sharp), i poboljšani Visual Basic.
- Visual Studio razvojno okruženje. Ovaj alat obezbeđuje set programerskih alata, kao što su editori koda, dizajneri interfejsa i debagera koji dosta pojednostavljaju zadatak kreiranja .NET aplikacija.

16.2. Osnovni koncepti

Da bi se potpuno razumjeli listinzi koda u sljedećem tekstu, potrebno je razumjeti nekoliko ključnih izraza, uključujući common language runtime (CLR), skupove, opsege imena i hendlere događaja.

16.2.1. Common Language Runtime

Common language runtime (CLR) je infrastruktura koju .NET koristi za izvršavanje svih aplikacija. CLR uključuje ogroman set biblioteka klasa koje se mogu koristiti u aplikacijama. Neke interesantne klase koje nisu povezane sa

Internetom uključuju programe koji vam omogućuju da manipulišete Windows NT logom događaja, Active Directory-em, kriptografskim uslugama, brojačima performansi, COM+ uslugama, skladištenjem objekata i osnovnim Windows uslugama. U ovom tekstu fokusirat ćemo se na klase koje su povezane sa Internet programiranjem, kao što su Web kontrole, klase Web usluga i sigurnosne Web klase.

CLR također vodi brigu o kompajliranju "intermediate language" (međujezika) ili IL koda u osnovni mašinski kod. Osnovni proces funkcionira kao što slijedi:

1. Visual Basic kompajler konvertuje Vaš kod u IL. Što se tiče funkcionalnosti i kompleksnosti jezika, IL je negdje između assemblera i C-a. IL je optimiziran za brzo kompajliranje u mašinski kod. Možete zamisliti IL kao međuproizvod procesa kompajliranja.
2. Vašu Internet aplikaciju poziva korisnik, ili preciznije, Web server kao odgovor na zahtjev korisnika.
3. CLR preuzima i kompajlira IL u mašinski kod. Ovo se dešava funkcija po funkcija. Ako se funkcija koristi, ona se kompajlira u mašinski kod. Ako ne, ona ostaje kao IL.

Iako ovaj proces izgleda zaobilazno i čini vašu aplikaciju suviše sporom, on čini razvoj mnogo bržim, kao što je prvi korak mnogo brži od drugih metoda. Možete također kompajlirati vaš .NET kod direktno u mašinski kod korištenjem standardnih alata instaliranih sa .NET Framework-om.

CLR sadrži druge usluge, uključujući podršku za izvršavanje više .NET aplikacija unutar jednog procesa sa garancijom da se one neće miješati jedna sa drugom. Ovo je rezultat koncepta nazvanog type safety (sigurnost tipa). Bilo koji .NET kod mora garantovati type-safe kod, koji osigurava da, između ostalog, nizovima i kolekcijama nikada neće biti nekorektno pristupano. Visual Basic kompajler uvijek uspostavlja sigurnosti tipa kada kompajlira vaš kod u IL.

16.2.2. Skupovi-assemblies

Sav .NET kod se kompajlira u skupove. Izvana, skup izgleda tačno kao dynamic link library (.dll) ili izvršni program (.exe). Prvih nekoliko bajtova unutar svakog skupa sadrže mašinski kod koji aktivira common language runtime kada se skupu pristupi iz drugog programa ili kada mu pristupi korisnik.

Unutar skupa je mješavina IL-a i mašinskog koda, u zavisnosti koliki dio skupa je kompajliran CLR-om. Skupovi također sadrže metapodatke. Metapodaci za .NET program uključuju:

- Informaciju o svakoj javnoj klasi ili tipu, uključujući ime, iz kojih klasa je klasa razvijena, i svaki interfejs koji klasa implementira.

- Informacije o svakoj javnoj metodi u svakoj klasi, uključujući ime i povratnu vrijednost.
- Informacije o svakom javnom parametru za svaku metodu, uključujući ime svakog parametra i tip.
- Informacije o svakom javnom nabranjanju, uključujući imena i vrijednosti za svaki član nabranjanja.
- Informaciju o verziji specifikacije skupa.

Jedan problem koji imaju DLL datoteke je taj što nema načina da se pregleda koje tipove funkcija podržavaju; možete naći samo imena. COM datoteke ili komponente su obično izgrađene na vrhu DLL datoteka i imaju isti problem. Možete povezati biblioteke COM-tipova sa COM objektom, u zavisnosti od ogromnog broja ulaza Windows Registry-a koji vrlo lako mogu postati nekonzistentni.

Imajući tako obilje podataka u svakom skupu, .NET Framework nema potrebu da koristi Registry za pokretanje skupova. Ako .NET program ima potrebu da koristi drugi .NET skup, oni samo treba da budu locirani u istom direktoriju. Ovo znači da ako želite da razmjestite .NET program na drugom računaru, možete proći sa jednostavnim kopiranjem potrebnih datoteka.

Možete pisati .NET kod za pregledanje metapodataka drugog skupa da biste pronašli da li podržava funkcije koje su vam potrebne. Ovaj proces se naziva refleksija i Microsoft online dokumentacija sadrži veliki broj detalja.

16.2.3. Opsezi imena

Termin opseg imena je poznat iz jezika C++. Opseg imena se koristi u .NET jezicima da bi obezbijedio kontejner za set klasa. Ovaj kontejner vam omogućuje da kreirate klase sa istim imenima koje drugi programeri ili CLR-i koriste. Na primjer, možete koristiti set pomoćnih Visual Basic klasa koje ste preuzeli sa Web prezentacije. Ove pomoćne klase mogu sadržati klasu nazvanu `ErrorLogger`, koja ispisuje poruke o greškama u tekstualnu datoteku. Nažalost, već ste razvili klasu koja se također naziva `ErrorLogger`. Imati isto ime nije problem je pomoćna klasa koristi različit opseg imena od vašeg.

Pogledajte "HelloWorld.vb": Uobičajeni "Hello, world" program:

```
Imports System
Module Module1
    Sub Main ( )
        Console.WriteLine ("Hello, world")
    End Sub
```

End Module

Prva linija koristi Imports ključnu riječ, koja kaže kompajleru da uključi System opseg imena u ovaj program. Linija koja ispisuje Hello World (Zdravo svijete), može biti napisana različito. Da listing nije koristio Imports ključnu riječ u prvoj liniji, linija sa ispisom bi izgledala ovako:

```
System.Console.WriteLine ("Hello World.")
```

Kao što možete vidjeti, opsezi imena grupišu slične klase u jednu logičku grupu da bi spriječili konflikte sa imenima.

Druga linija deklarise modul za kod programa, jedan od načina na koji VB.NET organizuje svoj kod. Linije od Sub Main() do kraja definišu podproceduru nazvanu Main (), gdje izvršavanje počinje.

16.3. Konfigurisanje okruženja

Da biste izvršili prikazane primjere programa, potrebno je da imate instalirane tri komponente na računaru:

- Jedan od sljedećih operativnih sistema: Windows NT 4.0 Workstation ili Server, Windows 2000 Professional ili Server, ili Windows XP ili viši, kao što je Advanced Server ili Datacenter Server. Windows NT nije preporučljiv, jer je poznato da ima problema sa .NET-om na tom operativnom sistemu. Nema načina za izvršavanje .NET Internet razvoja na platformama kao što su Windows 95/98ME.
- Web server koji podržava ASP.NET. Ovo znači Microsoft Internet Information Services (IIS) 4.0 ili viši, ili Microsoft Personal Web Server (PWS) na NT 4.0 radnoj stranici.
- .NET Framework, koji uključuje CLR, Visual Basic kompajler i ASP.NET.

Iako je opcioni, preporučuje se instaliranje Visual Studio .NET (koji uključuje .NET Framework). Možete kreirati .NET aplikacije bez Visual Studio-a korištenjem bilo kog editora teksta i kompajlera sa komandne linije, ali Visual Studio čini čitav proces-naročito debugovanje-mnogo lakšim.

16.3.1. Instalacija IIS-a

IIS dolazi sa Windows-om 2000 Professional ali se ne instalira podrazumijevano. Da biste instalirali IIS, morate načiniti sljedeće korake, i morate imati pri ruci svoj Windows 2000 CD.

Pratite ove korake:

1. U Windows Control Panel-u, izaberite Add/Remove Programs.

2. Izaberite Add/Remove Windows Component na dnu Add/Remove Programs prozora.
3. Pogledajte polje za potvrdu pored Internet Information Services stavke.
4. Ubacite Windows 2000 CD u svoj računar.
5. Kliknite Next i sačekajte da se završi instalacija.

16.3.2. Instalacija Visual Studio .NET-a

Prije nego što pokrenete bilo koji kod od prikazanih, potrebno je dobro razumijevanje kako Web server izvršava Web stranicu ili Web Service. Internet aplikacije kao što su dinamičke Web strane ili korisnički Web Service-i brinu o jednom zadatku - obrada korisničkih zahtjeva. Ovi zahtjevi dolaze kao HTTP zahtjevi iz Web pretraživača klijenta ili kao SOAP zahtjevi iz klijent programa Web Service-a. Web Service-i mogu također biti nazvani iz Web pretraživača, pored toga što ih pozivaju korisnički klijent programi. U većini slučajeva, korisnički klijent je potreban da uradi bilo šta korisno sa podacima koje vraća Web Service.

Tipičan HTTP zahtjev izgleda ovako:

```
GET http://www.mycompany.com/index.htm HTTP 1.1
```

Ovaj zahtjev traži od Web servera koji se nalazi na računaru nazvanom "www.mycompany.com" da pošalje nazad stranu sa imenom index.htm.

SOAP znači Simple Object Access Protocol. Kao i HTTP, to je još jedan standard za slanje i primanje podataka sa server računara. U stvari, SOAP se oslanja na HTTP i dodaje dodatne nivoe sofisticiranosti da bi dozvolio kompleksnije interakcije. U većini slučajeva, SOAP zahtjevi se šalju Web serveru kako bi obradio zahtjev. SOAP zahtjevi i odgovori su prilagođeni pozivajućim funkcijama umjesto da vraćaju Web strane.

16.4. Pregled procesa dinamičkih strana

Šta se dešava kada korisnik zatraži preko Web pretraživača jednu od vaših strana? Web pretraživač generiše HTTP zahtjev za određenom stranom koju korisnik želi da vidi. Web server, koji osluškuje sve vrijeme za HTTP zahtjevima, preuzima zahtjev i traži datoteku za tu određenu stranu. Ako se strana završava sa .htm ili .html ekstenzijom, Web pretraživač jednostavno vraća sadržaj datoteke.

Naravno, Web serveri mogu uraditi više nego jednostavno prikazivanje datoteka koje se završavaju .htm ekstenzijom. Trenutno, možete naći Web prezentacije koje rade puno vrsta komplikovanih zadataka, kao što je usluga ispisivanja čekova kod online banaka, koja će vam omogućiti da plaćate svoje račune korištenjem samo Web pretraživača. Kako ove dinamičke Web strane rade?

Trik u procesu je da dinamičke strane koje sadrže kompleksnu programsku logiku koriste različite ekstenzije datoteka. Na primjer ASP.NET strane se završavaju sa .aspx ekstenzijom. Kada Web server vidi zahtjev za takvom stranom, on ne čita datoteku. Umjesto toga, zna da prosljedi HTTP zahtjev .NET-u (tačnije, ASP.NET procesu), i omogućiti ASP.NET-u da preuzme zadatak procesiranja strane i slanja odgovora. Rezultat je da ASP.NET izvršava vaš kod, koji izvršava bilo koji zadatak za koji je dizajniran.

16.5. Pregled obrade Web Service-a

Web Service-i izvršavaju sličan zadatak sa dinamičkim Web prezentacijama jer Web Service-i jednostavno odgovaraju na zahtjeve klijenata sa Interneta i vraćaju podatke. Kao što je pomenuto, umjesto da primaju HTTP zahtjeve, Web Service-i primaju SOAP zahtjeve. Svaki SOAP zahtjev sadrži ime naročite funkcije, kao što je GetMyAppointments, i parametar za funkciju, kao što je današnji datum. Svaki odgovor SOAP-a sadrži podatke koje vraća funkcija. Na primjer, metoda GetMyAppointment može da vrati strukturu podataka koja izlistava sve dogovorene sastanke korisnika za određeni dan.

SOAP omogućuje klijent programima da prosljeđuju skoro bilo koju vrstu strukture podataka Web Service-u; zbog toga WebService-i koriste SOAP. Slično ovome, Web Service-i mogu da vrate gotovo bilo koju vrstu strukture podataka klijent programu korištenjem SOAP-a

Većina Web Service-a još uvijek koriste Web server za komunikaciju sa spoljnim svijetom. Poslije svega, Web serveri su dobri za osluškivanje zahtjeva i vraćanje podataka. .NET Web Services koriste ASP.NET za obradu SOAP zahtjeva i prevode ih u strukture podataka koje oni mogu koristiti. Ovaj proces radi na isti način kao i dinamičke Web strane. Svi .NET Web Service-i završavaju sa ekstenzijom .aspx, Internet Information Service-i znaju da treba da prosljede sve zahtjeve koji se završavaju sa .aspx ASP.Net-u, koji zatim preuzima i obrađuje SOAP zahtjeve i prosljeđuje ih vašem korisničkom kodu Web Service-a.

16.6. Kreiranje Login.aspx strane

Kreirajte novu datoteku nazvanu Login.aspx korištenjem nekog editora teksta (npr. Notepad). ASP.NET strane završavaju sa .aspx ekstenzijom. ASP.NET datoteke mogu koristiti druge ekstenzije, kao što je .ascx za korisničke kontrole i .asmx datoteke za Web usluge.

Prva prikazana ASP.NET strana će biti jednostavan "login" ekran, kreiran korištenjem Visual Basic-a i ASP.NET-a

```
<%@ PageLanguage = "VB" %>
<script runat="server">
  sub Page_Load(Sender as Object, e as EventArgs)
    if IsPostBack then
      dim strUserName as string = Request ("username")
      dim strPassword as string = Request ("password")
      if (strUserName = "Joe Smith") and (strPassword =
        "secret") then
        Response.Write("<h2>Log on uspjesan za " &
          strUserName & "</h2>")
      else
        Response. Write("<h2>Vas username/password nisu
          pronadjeni.</h2>")
      end if
    end if
  end sub
</script>
<html><body>
  <h2>Dobrodošli na primjer stranice za Login</h2>
  <form runat ="server">
    <table><tr>
      <td>UserName:</td>
      <td><input type="text" name="username"></td>
    </tr>
    <tr>
      <td>Password</td>
      <td><input type="password" name="password"></td>
    </tr>
    <tr>
      <td></td>
      <td><input type="submit" value="Login"></td>
    </tr>
  </table></form>
</body></html>
```

Prva linija listinga kaže ASP.NET-u da koristite Visual Basic za kod na strani. Ova linija predstavlja direktivu strane - page directive.

Iako ASP.NET omogućuje da koristite bilo koji od .NET jezika, možete koristiti samo jedan tip jezika za svaku pojedinačnu stranu koda sa strane servera. Ako vaša strana sadrži kod sa strane klijenta, možete ga pisati sa različitim jezikom, kao što je JavaScript.

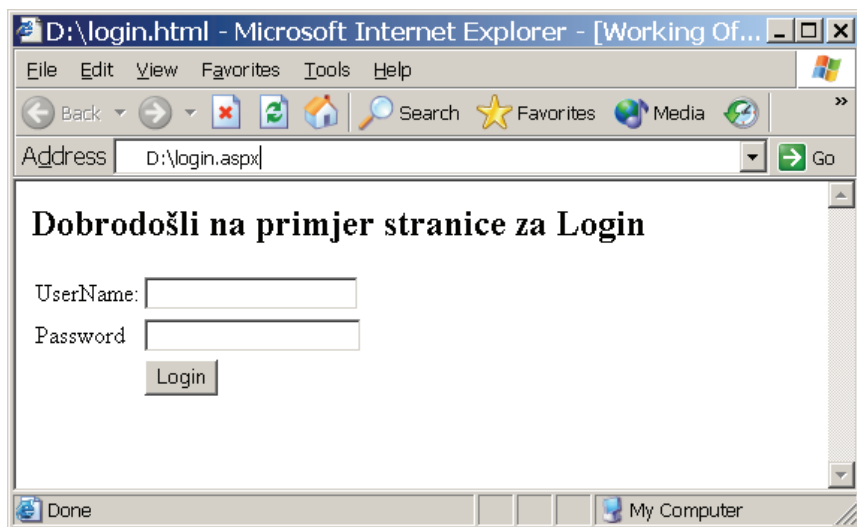
Druga linija označava početak koda skripte. Direktiva `runat="server"` obezbjeđuje da se kod u skript tagovima izvršava na Web serveru. Ovaj kod se nikada ne šalje klijent pretraživaču.

Prvi (i jedini) metod unutar `<skript>` taga se naziva `Page_Load`. Ovaj metod je ASP.NET hendler događaja. ASP.NET poziva ovu metodu svaki put kada se sastavlja strana. Ova metoda se također poziva prije bilo kog kreiranog HTML izlaza.

`Page_Load` metoda koristi kod da bi provjerila unijeto korisničko ime i password. Kod koristi `Request` i `Response` objekte za prijem i slanje koda klijent pretraživaču.

Test u prvom if iskazu koristi `IsPostBack` zastavicu. ASP.NET automatski podešava ovu zastavicu kada je strana pozvana pošto korisnik klikne submit dugme. Kod prvog poziva strane, ova zastavica (flag) je `False`, tako da se veći dio koda preskače. Ovo je ono što želimo-želimo da provjerimo korisničko ime i password samo ako je korisnik stvarno predao neke podatke.

HTML kod u preostalim linijama listinga je standardan izuzev `<form>` taga koji sadrži `runat="server"` atribut. Ovaj tag je prvi primjer koji ćete vidjeti na jednoj ASP.NET Web kontroli: ovaj tag funkcioniše kao običan tag forme, osim što obezbjeđuje da će ASP.NET pozivati `Page_Load` događaj.



Slika 90. Login strana kreirana korištenjem ASP.NET-a

Preostali HTML u listingu se sastoji od dva input okvira i submit dugmeta. Kad korisnik klikne "submit" dugme, sadržaj forme se šalje nazad login strani. Poslije ovog post back-a, kod u Load događaju će biti ponovo izvršen. Međutim, ovoga puta IsPostBack zastavica će biti True, i unos sa forme će biti provjeren.

16.7. Korištenje Web kontrola

Web kontrole nude mnoge prednosti nad HTML kontrolama u aplikacijama, uključujući svojstva, događaje i upravljanje stanjem. Možete konvertovati prethodno prikazani listing da koristi Web kontrole za input okvire i submit dugme.

```
<%@ Page language = "VB" %>
<script runat="server">
sub LoginClicked(Sender as Object, e as EventArgs)
    if IsPostBack then
        dim strUserName as string = Request ("username")
        dim strPassword as string = Request ("password")
        if (strUserName = "Joe Smith") and (strPassword =
            "secret") then
            Response.Write("<h2>Log on uspjesan za " &
                strUserName & "</h2>")
        else
            Response. Write("<h2>Vas username/password ne
                postoje.</h2>")
        end if
    end if
end sub
</script><html><body>
<h2>Dobrodošli na primjer Login strane</h2>
<form runat ="server">
<table>
<tr>
<td>UserName:</td>
<td>
    <asp:textbox id="username" runat="server"/>
</td>
</tr>
```

```
<tr>
  <td>Password</td>
  <td><asp:textbox id="password" textmode="password"
    runat="server"/></td>
</tr>
<tr>
  <td></td>
  <td>
    <asp:button text="Login" onClick="LoginClicked"
      runat="server"/>
  </td>
</tr>
</table>
</form></body></html>
```

Primijetite da su Text box korisničkog imena, Text box password-a i Login dugme Web kontrole. Web kontrole su označene sa "<asp:controlname" i završavaju se sa ">". Svaka definicija Web kontrole sadrži direktive za podešavanje atributa kontrola, kao što je textmode="Password" za polje password.

Listing ne koristi Page_Load događaj. Umjesto toga, koristi korisnički hendler događaja, LoginClicked, koji poziva button kontrola. Primijetite tekst programa onclick="LoginClicked" umjesto definicije Login dugmeta. Sve ASP.NET Web kontrole koriste događaje koje možete obraditi sopstvenim kodom na strani.

Svaka ASP kontrola na prikazanoj strani sadrži atribut runat="server". Ovaj atribut je od suštinskog značaja za bilo koju definiciju Web kontrole; bez njega, kontrola neće raditi pravilno.

Ako pokrenete kod u prikazanom listingu, možete primijetiti da polje korisničkog imena čuva svoju vrijednost čak i nakon što je Login dugme kliknuto. Polje password podrazumijevano ne čuva svoje stanje. ASP.NET Web kontrole čuvaju svoje stanje bez bilo kakvog programskog napora sa vaše strane.

16.7.1. Dostupne Web kontrole

Dostupne ugrađene (built-in) ASP.NET Web kontrole su:

Button	Table, TableCell, TableHeaderCell, TableRow Checkbox, RadioButton TextBox
HyperLink	AdRotator
Image, ImageButton	Calendar
Label	ListControl (Checkbox, Dropdown, RadioButton)
LinkButton	DataGrid, DataList
Panel	Validator kontrole

16.7.2. Definisanje Web formi

Primjer login strane u prethodnom dijelu je primjer ASP.NET Web forme. U Visual Basic-u, forma je prozor sa kojim korisnik stupa u interakciju; za ASP.NET, Web forma je Web strana koja stupa u interakciju sa korisnicima. Web forme imaju programiranje stanja, događaja i svojstava baš kao i prozor na desktopu.

16.7.3. Kodiranje iza formi u ASP.NET-u

Održavanje strana koje sadrže kombinaciju HTML i ASP ili sličnog koda je teško; kod je težak za razumijevanje zbog kombinovanja HTML-a i scripta. Možete lako praviti sintaksne greške u skriptu koje prouzrokuju grešku čitave strane, i često ASP automat ne daje dobru informaciju greške o greškama pri kodiranju.

Da bi se sav kod i logika stavili u jednu datoteku, a sav HTML i elementi korisničkog interfejsa u drugu, to se može uraditi korištenjem "kod-iza" osobine ASP.NET-a.

Promijenimo Login page primjer da koristi kod-iza, prikazan na sljedeća dva listinga.

LoginCodeBehind.aspx (samo elementi HTML-a i korisničkog interfejsa):

```
<%@ Page language="VB" inherits="MyLoginPage"
src="Login.vb" %>

<html><body>

  <h2>Welcome to the Sample Login Page</h2>

  <form method="post" runat="server">

    <table>

      <tr>
```

```
<td>UserName:</td>
<td><asp:textbox id="UserName" runat="server" /><td>
</tr>
<tr>
<td>Password</td>
<td><asp:textbox id="Password" textmode="Password"
    runat="server" /></td>
</tr>
<tr>
<td></td>
<td>
    <asp:button id="LoginButton" text="Login"
        onclick="LoginClicked" runat="server" />
</td>
</tr>
</table>
</form>
</body></html>
```

Prva linija listinga koristi `inherits` i `src` atribut unutar direktive strane. Atribut `inherits` definiše kod iza klase koji će koristiti ova Web forma (ASP.NET strana). Svaka Web forma koristi samo jednu klasu.

Direktiva `src` kaže ASP.NET-u gdje da locira izvorni kod-iza datoteku za ovu Web formu. Preostale linije u ovom listingu su identične sa prethodnom verzijom login strane, izuzev što je sav kod uklonjen. Ovo znači da je sav kod korisničkog interfejsa na jednom mjestu.

Login.vb (Visual Basic datoteka koja sadrži sav kod za login stranu):

```
imports System
imports System.Web
imports System.Web.UI.WebControls
public class MyLoginPage
inherits Page
protected UserName as TextBox
protected Password as TextBox
protected LoginButton as Button
```

```
public sub LoginClicked(Sender as Object, e as
EventArgs)

dim strUserName as string = UserName.Text
dim strPassword as string = Password.Text
if (strUserName = "Joe Smith") and (strPassword
="secret") then

    Response.Write("<h2>Log on uspjesan za " &
strUserName & "</h2>")

else

    Response.Write("<h2>Vas username/password ne
postoji.</h2>")

end if
end sub
end class
```

Ovakav kod deklarira klasu razvijenu iz Page klase, MyLoginPage. MyLoginPage je ime klase koje se koristi u inherits atributu u prethodnom listingu. Sav kod iza klase za Web forme mora biti razvijen iz Page klase.

Primijetite da MyLoginPage klasa deklarira svaku Web kontrolu korištenu u .aspx datoteci kao zaštićene članove klase. Napravili smo ove deklaracije jer koristimo kontrole u LoginClicked metodi za rukovanje događajem.

Također smo dodali imports direktive za svaki opseg imena koji MyLoginPage koristi.

Iako je kombinovani kod iz ova dva listinga duži od originalne verzije koja nije koristila kod-iza, ova nova implementacija je bolje organizovana. Takva organizacija postaje značajnija kada Web forme porastu, kao što se to dešava u praktičnim aplikacijama.

16.8. ASP.NET

Microsoft je nedavno predstavio Visual Studio .NET - kamen temeljac svoje .NET strategije, čime je ona zapravo otpočela svoje postojanje. Tek nešto ranije, web developerima širom svijeta postao je dostupan nasljednik ASP-a, pod nazivom ASP.NET. Nazvati ASP.NET logičnom nadogradnjom klasičnog ASP-a 3.0 bilo bi pogrešno. ASP.NET, kao uostalom i cijela .NET strategija, uvodi potpuno novu paradigmu u programiranje web skripti. Zapravo, od ASP-a je preuzeta tek namjena (stvaranje dinamičnih web stranica) i dio imena, a uvedene su dvije bitne stvari: mnoga programska poboljšanja na jednoj strani, a na drugoj potpuna promjena osnovna programiranja, odnosno način na koji pišete kod za .ASP stranice.

16.8.1. Način pisanja koda

Klasične ASP stranice pišu se proceduralno, što znači da je svaki zadatak koji je potrebno obaviti najčešće smješten u posebnoj ASP stranici, a redoslijed kojim se izvršavaju pojedine komande unutar stranice određen je njihovim položajem (komande se izvršavaju odozgo prema dole). Kod ASP.NET-a, situacija je drastično drugačija. Recimo, za jednostavan zadatak ispisa forme u koje korisnik treba upisati podatke i prikaza tih podataka nakon slanja forme, u klasičnom ASP-u potrebne su dvije stranice: jedna za ispis forme, druga za prikaz podataka nakon klika na submit dugme. Alternativa je korištenje jedne stranice, sa nekoliko if petlji za provjeru da li je forma poslana i da li su polja popunjena. Doda li se tome potreba za provjerom unosa (error checking), postaje jasno da to nije jednostavna stvar za isprogramirati (nije komplikovana, ali zahtijeva puno koda). U ASP.NET-u sve se to može uraditi pomoću jedne stranice. I ne samo to - korištenjem serverskih kontrola i ugrađenih rutina za provjeru grešaka moguće je gore navedeni primjer napisati u tek par linija koda. Da biste shvatili o čemu govorim, pogledajte kod:

ASP 3.0:

Datoteka: forma.asp

```
<form method="post" action="ispis.asp" id=form1
name=form1>

  <div align="center"> Ime i prezime: <br>

  <input type="text" name="Ime" size="20"
    maxlength="50">

  <br> E-mail adresa: <br>

  <input type="text" name="Email" size="20"
    maxlength="50">

  <br>

  <br>

  <input type="hidden" name="poslano" value="da">

  <input type="submit" name="Submit" value=" Ispisi
    podatke ">

  </div>

</form>
```

Datoteka: ispis.asp

```
<%

'da li je forma uopste poslana
If Request.Form ("poslano") ="da" Then
```

```
'ako jeste, da li su upisani podaci
    If Request.Form ("Ime") <> "" And Request.Form
("Email") <> "" Then
        'ako je sve ok, ispisi podatke
        Response.Write "Vase ime: " & Request.Form
("Ime")
        Response.Write "<br><br>Vasa e-mail adresa:" &
Request.Form ("Email")
        Else
            'ako nije
            Response.Write "Niste upisali potrebne
podatke!"
        End if
    End if
End if
%>
```

ASP.NET:

Datoteka: aspx.aspx

```
<form action="asp.ascx" runat="server">
    Ime i prezime: <br>
    <asp:textbox id="Ime" runat="server"/>
    <br>E-mail adresa:<br>
    <asp:textbox id="Email" runat="server"/><br><br>
    <asp:button text=" Ispisi podatke " runat="server"
onclick="Prikazi"/><P>
    <B><asp:label id="WelcomeMessage"
runat="server"/></B>
</FORM>
<script language="vb" runat="server">
    Sub Prikazi (sender as Object, e as EventArgs)
        WelcomeMessage.Text = "Pozdrav, vase ime je " &
Ime.Text & "!"
    End Sub
</script>
```

Kod više liči na dio nekog Visual Basic projekta, nego na web stranicu. Dok je kod klasičnog ASP-a teško izbjeći tzv. špageti-kod, ASP.NET kod je lako čitljiv, uspješno odvojen od dizajna stranice i uopšte mnogo sličniji pravom programskom jeziku.

16.8.2. Programska poboljšanja

Jedan od najvažnijih novosti jeste značajno ubrzanje izvršavanja skripti. Za razliku od dosadašnjih ASP skripti, koje su interpretirane prilikom svakog pozivanja od strane klijenta. ASP.NET skripte su u potpunosti kompajlirane! Kada se stranica prvi put pokrene biva kompajlirana pomoću CLR-kompatibilnog (Common Language Runtime, još jedna bitna novost .NET platforme) kompajlera i pri svakom sljedećem učitavanju pokreće se već kompajlirana verzija, koja se izvršava puno brže. Kada dođe do osvježavanja (update) stranice, ona pri prvom učitavanju biva rekompajlirana.

Kompajliranje skripti donosi određeno ubrzanje, ali najveći dio vremena izvršavanja neke skripte odlazi na pristupanje bazama i uzimanje podataka iz njih. Upravo zato ASP.NET donosi nekoliko tehnika keširanja podataka. Recimo da imate stranicu na kojoj prikazujete posljednjih 20 unosa iz baze i da toj stranici pristupa oko 100 ljudi u minuti. Bez obzira koju bazu podataka koristi i koliko jak server imali, ovi korisnici će prouzrokovati prilično usporenje servera. U ASP.NET skripti koja obavlja taj zadatak dovoljno je upisati naredbu koja će keširati podatke za nekoliko minuta (nekoliko sati ili dana, zavisi od frekvencije dodavanja novih podataka u bazu), tako da će skripta samo za prvog korisnika uzeti podatke iz baze, a ostali će vidjeti keširanu stranicu! Moguće je i fragmentno keširanje, tako da možete recimo odrediti da se reklame ne keširaju (uvijek se učitavaju iz baze), ali da se ispisi onih 20 unosa iz baze keširaju.

Još jedna bitna novost jesu serverske kontrole. Dok ste u ASP-u za ispis forme koristili standardne HTML kontrole (textbox, password, select, button i sl.), u ASP.NET-u koristite ekvivalentne serverske kontrole, koje se kasnije renderišu u obični HTML, u zavisnosti od browsera. Dakle, problemi sa kompatibilnošću browsera su gotovi! U svojoj skripti pišete naredbe koje su uvijek iste (recimo `asp:textbox` ili `asp:button`), a server vraća klijentu HTML kod u zavisnosti od njegovog browsera. Sa ovim kontrolama možete kombinovati kontrole za provjeru (validation), tako da kompleksne if petlje potrebne za provjeru unosa u formi konačno odlaze u prošlost.

Jedan od najvećih problema sa klasičnim ASP skriptama jeste njihovo postavljanje na server. Ako se radi o jednostavnim skriptama koje ne koriste nikakve serverske objekte ili posebno postavke IIS-a, onda je postavljanje na server čas posla. Ali ako vaša skripta zahtijeva posebne COM objekte ili IIS postavke postavljanje na server zna biti mučno. Potrebno je provjeriti da li ti objekti postoje na serveru, da li su registrovani, provjeriti kakve su postavke servera... Sve ove probleme rješava

ASP.NET. COM objekte koje vaša skripta koristi više uopšte nije potrebno registrovati na serveru! Dovoljno ih je kopirati na server u odgovarajući folder i skripta ih može koristiti! Također, sa ASP.NET-om stiže i tekstualna konfiguracija datoteka u kojoj možete podesiti bitne postavke, koje su se do sada mogle mijenjati jedino preko IIS administracijskih stranica.

Iako je ASP po tom pitanju veći bio iznad konkurencije, ASP.NET uvodi još nekoliko jezika koji se mogu koristiti za programiranje skripti. Umjesto oslabljene verzije VB-a - VBScript-a, koja je do sada predstavljala default jezik za programiranje ASP stranica, ASP.NET stranice se mogu programirati u dva "prava" programska jezika - VB.NET i C#. Pored njih, direktno je podržan Jscript, a Visual StudioNET (samim tim i ASP.NET) će podržavati oko 20 jezika! VBNET i C# su moderni jezici, koji ne samo da podržavaju polimorfiranje i nasljeđivanje objekata, nego se mogu i kombinovati - VBNET može nasljeđivati C# objekt i obratno.

17. XML

XML (eXtensible Markup Language) se možda čini jednim od onih W3C-ovih standarda za daleku budućnost. Ipak, on ima praktičnu prijem već danas. XML se može upotrijebiti u raznim situacijama za rješavanje raznih problema. Zato nemojte čekati da bude podržan u vašem omiljenom web editoru prije nego ga isprobate.

Poput HTML-a, XML vodi porijeklo od pradjeda svih markup jezika: SGML-a (Standard Generalized Markup Language). SGML je metajezik, odnosno sistem za definisanje drugih jezika. I XML je metajezik, mogli bismo reći podskup SGML-a namijenjen upotrebi na webu.

Ljudi iz World Wide Web Consortiuma (ili kraće W3C) - tijela zaduženog za donošenje standarda na polju Interneta, shvatili su da zbog njegovog rasta mora postojati način da se odvoji sadržaj od načina njegovog prikazivanja. I tako se rodio XML, kojeg W3C definiše kao "jezik za definisanje struktura podataka". Jednostavno rečeno, XML je zadužen samo za struktuiranje podataka, ali ne i njihovo prikazivanje. Pošto je XML markup jezik, sastoji se od tagova, odnosno elemenata, kako se oni zovu u XML-u. Za razliku od HTML-a gdje imate unaprijed određene tagove i njihove funkcije (na primjer, <h2> će uvijek tekst koji slijedi prikazati kao naslov), u XML-u sami definiše tagove koji vam trebaju (na primjer, tag <cijena> može sadržavati u sebi cijenu, tag <opis> može sadržavati opis nekog proizvoda itd.). Pošto jednom napisani XML dokument može biti prikazan na mnogo različitih načina, bez potrebe za ikakvim promjenama u tom XML dokumentu, on će biti čitljiv na svim platformama. To ovome jeziku daje dodatnu vrijednost.

Primjer XML programa:

```
<?xml version = "1.0"?>
<lista>
  <cd-disk>
    <izvodjac> Metalica </izvodjac>
    <naziv> ...And justice for all </naziv>
    <godina> 1988 </godina>
    <cijena> 15.92 EUR </cijena>
  </cd-disk>
  <cd-disk>
    <izvodjac> Metalica </izvodjac>
    <naziv> Black </naziv>
    <godina> 1991 </godina>
    <cijena> 17,55 EUR </cijena>
```

```
</cd-disk>
<cd-disk>
    <izvodjac> Red Hot Chilli Peppers </izvodjac>
    <naziv> Californication </naziv>
    <godina> 2000 </godina>
    <cijena> 14,99 EUR </cijena>
</cd-disk>
<cd-disk>
    <izvodjac> Kom </izvodjac>
    <naziv> Issues </naziv>
    <godina> 2000 </godina>
    <cijena> 15,99 EUR </cijena>
</cd-disk>
</lista>
```

Kao što vidite, dokument je dosta sličan HTML-u. Sa jednom velikom razlikom - tagovi su na našem jeziku! U XML-u sami definišete tagove koji imaju nazive kakve odredite (obično su nazivi logički povezani sa sadržajem). Još jedna razlika u odnosu na HTML jeste da u ovome dokumentu ne postoje nikakve odrednice koje govore kako će podaci izgledati u browseru. Izgled dokumenta će se, kada budemo došli do toga, definisati iz posebne datoteke.

A zašto se u ovom primjeru nije jednostavno koristio obični HTML, sa tabelom popunjenom podacima? Zato jer XML daje mnogo više fleksibilnosti. Recimo da jednog dana poželite odštampati svoju listu CD-ova (ima ih preko stotinu). Morali biste ručno brisati sve HTML tagove da biste dobili čistu tekstualnu datoteku, ili biste morali mnogo koristiti copy i paste funkcije kako bi iz HTML dokumenta prenijeli podatke u neki tekst procesor, što prilično naporno i dugotrajno. Sa XML-om, može se napisati jednostavan mali program ili koristiti neki već dostupan, koji će "izvući" podatke iz dokumenta i automatski ih formirati u nekom tekst procesoru (na primjer, informacije iz taga <izvodjac> i <naziv> su podebljane, a ostale su ispisane normalnim fontom). Čak se može koristiti neki XML-enabled tekst procesor koji će začas odštampati ovaj XML dokument. To su samo neke od prednosti XML nad HTML-om u ovome slučaju.

Iako XML ostavlja autorima na slobodu da kreiraju vlastite tagove, odnosno elemente, prilikom pisanja XML dokumenta moraju se poštovati određena pravila. Svaki XML dokument mora biti pravilno napisan (pravilno formatiran) i ispravan.

Da biste pisali pravilne XML dokumente, trebate se pridržavati određenih pravila. Ta pravila nisu uopšte komplikovana, a mnoga od njih bi se trebala odnositi i na

HTML. Prije svega, na početku svakog XML dokumenta morate staviti naznaku da se radi o XML dokumentu, a to izgleda ovako:

```
<?xml version ="1.0"?>
```

Zatim u svakom XML dokumentu mora postojati root ili korijenski element unutar kojeg se nalaze svi ostali elementi. Kod HTML dokumenata taj element je <html>, a kod XML-a možete sami izabrati njegov naziv. Korijenski element u prikazanom primjeru je <lista>.

Svaki otvoreni element u XML dokumentu mora biti i zatvoren. Na primjer, ako otvorite element <cd>, morate ga i zatvoriti sa komandom </cd>. Dakle, svaki element se zatvara sa istom komandom kojom se i otvara, samo što prije nje stoji znak "/". U HTML-u mnogi tagovi su mogli ostati i nezatvoreni, a browser bi prikazao pravilno web stranicu. U XML-u pak, ukoliko neki tag nije zatvoren pravilno, dokument neće uopšte biti prikazan. Postoje neki tagovi koji u HTML-u nisu imali svoju verziju za zatvaranje, poput taga
. U XML se oni pišu sa znakom "/" nakon komande, u ovome slučaju
.

U XML dokumentu svi elementi moraju biti pravilno postavljeni, odnosno ne smije doći do njihovog "preklapanja". Na primjer, u HTML ste mogli slobodno napisati sljedeće:

```
<center><b><font face=verdana" size="2"> Tekst  
</center></font></b>
```

i to ne bi uopšte bilo nepravilno. U XML-u je to fatalna greška, i dokument više ne bi bio ispravno napisan. Pravilno napisani elementi izgledaju ovako:

```
<center><b><font face="verdana" size = "2"> Tekst  
</font></b></center>
```

Prilikom pisanja XML dokumenta, treba voditi pažnju o velikim i malim slovima. Dok bi u HTML-u tagovi (elementi) <cijena>, <Cijena> i <CIJENA> bili potpuno isti, u XML-u su to tri različita elementa.

Da ne bi došlo do pogreške, preporučujemo vam upotrebu ili samo velikih ili samo malih slova u vašim XML dokumentima. Postoji još dosta pravila, ali najvažnija su ova spomenuta.

17.1. DTD – Document Type Definition

Naravno, nešto ćemo uraditi sa informacijama iz ispravno napisanog XML dokumenta. Tačnije, uradit ćemo mnogo toga sa tim informacijama. Međutim, opasnost još uvijek vrijedi - postoji mogućnost da dokument, iako je pravilno napisan, ne sadrži neke ključne informacije. Pogledajmo slijedeći primjer.

```
<cd-disk>  
  
<izvodjac> Metalica
```

```
</izvodjac>
    <naziv> </naziv>
    <godina> 1988 </godina>
    <cijena> 15.92 EUR
</cijena>
</cd-disk>
```

Da je ovo jedan od stotinu dijelova jednog XML dokumenta, teško bi primijetili da ovome CD-u nedostaje naslov. Kako je dokument pravilno napisan, XML parser (svojevrzni XML kompajler) neće prijaviti nikakvu grešku. Svakome ko je ikada bio administrator čak i najjednostavnije baze, zna koliko su česte ljudske greške prilikom unošenja podataka u bazu. Ljudska je priroda takva da ćemo ako imamo priliku, ispustiti važne informacije i tako stvoriti različite nonsense. Zato su kreatori XML-a, znajući naše ljudske slabosti, napravili Document Type Definition, odnosno DTD. DTD u stvari osigurava da vaši XML dokumenti budu validni, odnosno sadrže informacije koje želite da sadrže. Primjer DTD-a za listu CD-ova glasi:

```
<!DOCTYPE lista [
    <!ELEMENT cd-disk (izvodjac, naziv, godina, cijena)>
    <!ELEMENT izvodjac (#PCDATA)>
    <!ELEMENT naziv (#PCDATA)>
    <!ELEMENT godina (#PCDATA)>
    <!ELEMENT cijena (#PCDATA)>
]>
```

Možda nije lako čitljivo na prvi pogled, ali kada se malo dublje analizira, ima smisla. Pogledajmo od čega se to sastoj DTD za prikazanu XML listu. Na osnovu ovoga primjera trebali biste dobiti predstavu kako se kreiraju i od čega se sastoje DTD-ovi za XML dokumente.

```
<!DOCTYPE lista [
```

Prva linija naznačava da sve unutar zagrade "[" predstavlja DTD za taj dokument sa root, odnosno korijenski elementom <lista>. Kao što je već spomenuto ranije, korijenski element sadrži u sebi sve ostale elemente.

```
<!ELEMENT cd-disk (izvodjac, naziv, godina, cijena)>
```

Ova linija definiše element <dd-disk>. U zagradi su naznačeni elementi koji se moraju nalaziti unutar svakog <cd-disk> elemenata, i to redom kojim su navedeni.

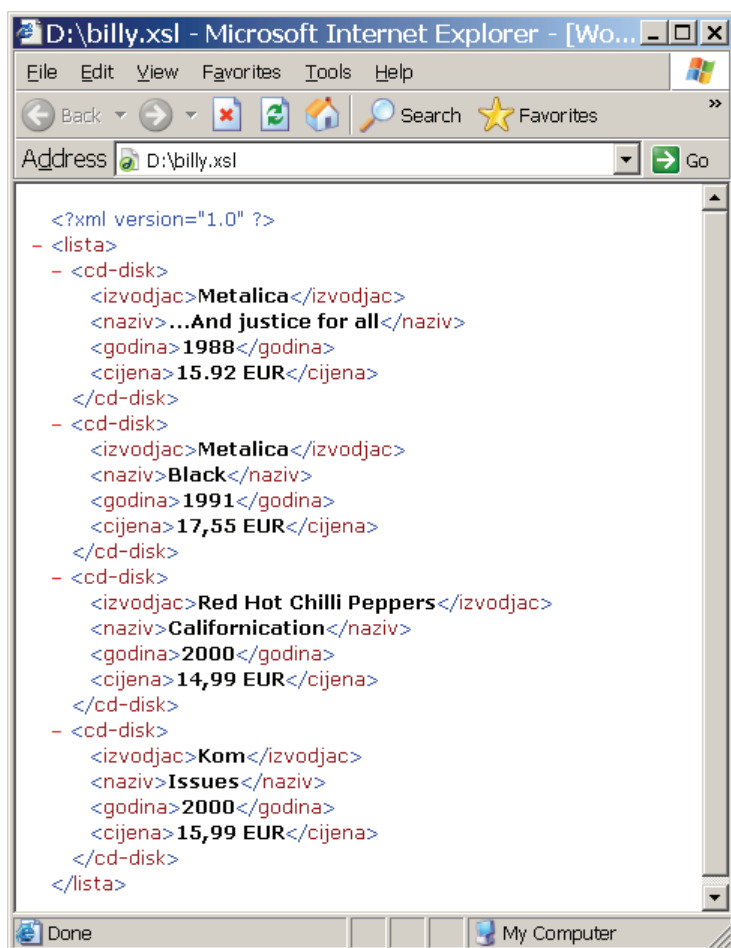
```
<!ELEMENT izvodjac (#PCDATA)>
```

U ovoj liniji se definiše element <izvodjac> koji, kako smo odredili u liniji iznad, mora biti unutar elementa <cd-disk>. Oznaka u zagradi označava da ovaj element

može sadržavati parsed character data (sve drugo osim binarnih podataka, kao što su slike), u našem slučaju će to biti tekst i brojevi.

17.2. Prikazivanje XML dokumenata u browseru

Ako učitate ovaj XML dokument u Internet Explorer, pod uslovom da ste sve pravilno uradili dobit ćete rezultat kao na slici 91. Ako se prisjetimo činjenice da je XML zadužen za definisanje strukture podataka, ali da u sebi ne sadrži nikakve informacije u tome kako će se ti podaci prikazati u korisnikovom browseru, stvari dolaze na svoje. Poseban jezik, koji se naziva XSL (eXtensible Style Language), omogućava autorima da formatiraju podatke sadržane u XML dokumentu. XSL koristi sintaksu XML-a, omogućujući XML autorima da stiliziraju svoje dokumente i podatke u njima bez potrebe za učenjem CSS-a ili nekog drugog stylesheet jezika (na primjer DSSSL).



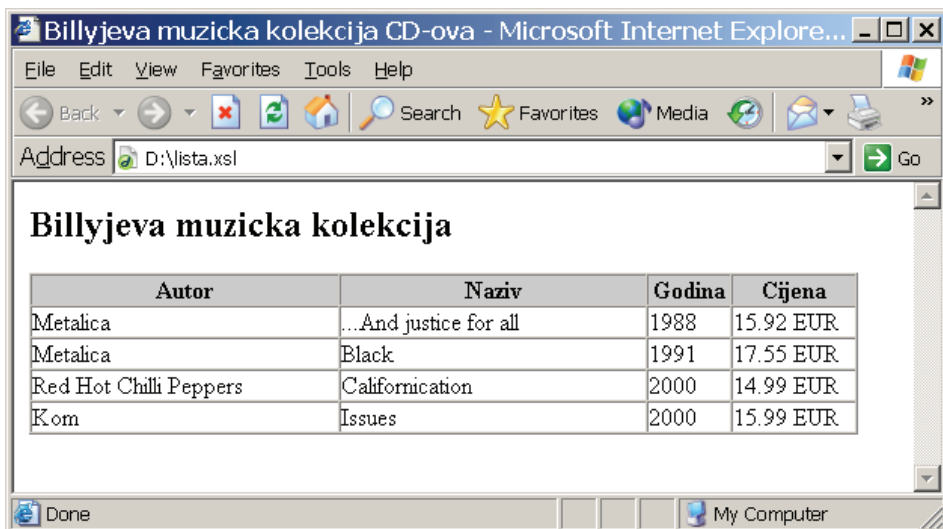
Slika 91. Prikaz XML dokumenta u browseru

Drugim riječima, XSL vam omogućava da izdvojite određene podatke iz XML dokumenta, a zatim te podatke formatirate i prikazete korisniku na način koji želite. Ovaj "transformacijski" dio XSL-a o kome govorimo se naziva XSLT (eXtensible Style Language for Transformation) i on će nas ovdje primarno zanimati, jer omogućava svojevrsnu transformaciju XML dokumenta i podataka sadržanih u njemu u HTML (i na taj način prikazivanje u korisnikovom browseru). Drugi dio XSL-a se još naziva i XSL FO (Formatting Objects), vrlo je kompleksan i omogućava prikazivanje XML-a u obliku bilo koje dvodimenzionalne prezentacije koju možete zamisliti (PDF dokument, izlaz za štampač itd.).

Šta se u stvari, jednostavnim jezikom rečeno, događa sa XML dokumentom prilikom formatiranja pomoću XSL-a? U XML dokumentu nalaze se podaci organizovani u elementima koje ste sami definisali. XSL pronalazi te elemente i pravi *source tree* (izvorno stablo dokumenta) XML dokumenta, a zatim primjenjuje pravila navedena u XSL datoteci na elemente i pravi *result tree* (konačno stablo dokumenta).

Konačno, XSL procesor (program koji primjenjuje XSL stilove na XML dokumente, a može biti na klijentskoj ili serverskoj strani) generiše HTML koji može biti prikazan u browseru. Dakle, podaci dolaze iz XML dokumenta, a formatiranje (koje može predstavljati i određeno restrukturiranje podataka) dolazi iz XSL datoteke. To je i bit XML-a; podaci i njihov prikaz su odvojeni. HTML nije jedini mogući "format" u kojeg se XML dokumenti mogu transformisati, ali nas u ovome tekstu jedino on zanima.

U nastavku možete vidjeti kako izgleda XSL datoteka za Billy-jevu muzičku kolekciju, a kako to sve izgleda kada se prikažu u browseru, možete vidjeti na slici 92.



Slika 92. Prikaz formatiranog XML dokumenta uz pomoć XSL-a

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="muzika.xsl"?>
<!DOCTYPE lista [
  <!ELEMENT cd-disk (izvodjac, naziv, godina, cijena)>
  <!ELEMENT izvodjac (#PCDATA)>
  <!ELEMENT naziv (#PCDATA)>
  <!ELEMENT godina (#PCDATA)>
  <!ELEMENT cijena (#PCDATA)>
]>
<lista>
  <cd-disk>
    <izvodjac> Metallica </izvodjac>
    <naziv> ...And justice for all </naziv>
    <godina> 1988 </godina>
    <cijena> 15.92 EUR </cijena>
  </cd-disk>
  <cd-disk>
    <izvodjac> Metallica </izvodjac>
    <naziv> Black </naziv>
    <godina> 1991 </godina>
    <cijena> 17.55 EUR </cijena>
  </cd-disk>
  <cd-disk>
    <izvodjac> Red Hot Chilli Peppers </izvodjac>
    <naziv> Californication </naziv>
    <godina> 2000 </godina>
    <cijena> 14.99 EUR </cijena>
  </cd-disk>
  <cd-disk>
    <izvodjac> Korn </izvodjac>
    <naziv> Issues </naziv>
    <godina> 2000 </godina>
    <cijena> 15.99 EUR </cijena>
```

```
</cd-disk>
</lista>
```

Od čega se sastoji korišteni XSL:

```
<?xml version="1.0!?">
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-
xsl/">
```

XSL je u stvari jezik definisan u XML-u, pa stoga to mora biti i naglašeno u prvom redu. Svaki XSL mora poštovati pravila pisanja ispravnih XML dokumenata i mora biti validan. Drugi red naznačava da se radi o XSL stylesheetu i morate ga navesti u svakoj XSL datoteci.

```
<xsl:template match="/">
```

Ovaj tag označava da se sve unutar njega (sva pravila) odnose na korijenski element u XML dokumentu. U ovom slučaju to bi bio element <lista>. Dalje možete vidjeti klasičan HTML koji susrećete na svakoj web stranici, a on definiše naslov stranice i postavlja tabelu na stranicu. Sljedeći zanimljiv XSL tag je:

```
<xsl:for-each select="lista/cd-disk">
```

On govori XSL procesoru da za svaki element <cd-disk> unutar korijenskog elementa <list> ubaci navedeni HTML kod. Kao što vidite u sljedećem kodu, taj HTML kod dodaje još jedan red tabeli koja je već ranije kreirana, a polja popunjava sa podacima iz XML dokumenta.

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-
xsl">
<xsl:template match="/">
<html>
<head>
<title> Billyjeva muzička kolekcija CD-ova </title>
</head>
<body>
<h2> Billyjeva muzička kolekcija </h2>
<table width="550" border="1" cellspacing="0"
cellpadding="0">
<tr bgcolor="#CCCCCC">
<td width="37%"><center><b>
Autor </b></center>
</td>
```

```
<td width="37%"><center><b>
    Naziv </b></center>
</td>
<td width="10%"><center><b>
    Godina </b></center>
</td>
<td width="15%"><center><b>
    Cijena </b></center>
</td>
</tr>
<xsl:for-each select="lista/cd-disk">
  <tr>
    <td width="37%">
      <xsl:value-of select="izvodjac"/>
    </td>
    <td width="37%">
      <xsl:value-of select="naziv"/>
    </td>
    <td width="10%">
      <xsl:value-of select="godina"/>
    </td>
    <td width="15%">
      <xsl:value-of select="cijena"/>
    </td>
  </tr>
</xsl:for-each>
</table></body></html>
</xsl:template>
</xsl:stylesheet>
```

Ovo je vrlo jednostavan XSL koji koristi samo osnovne komande. Pored for-each komande, moguće je koristiti if-then petlje i choose funkcije, koje su sigurno poznate programerima. Ove komande omogućavaju manipulaciju i prikazivanje podataka iz XML dokumenta na bilo koji način koji možete zamisliti (ili barem na

bilo koji praktičan način). Ako namjeravate upotrebljavati XML, onda svakako odvojite vremena da naučite XSL, jer bez njega nećete mnogo postići.

Primjer liste CD-ova u XML formatu je zaista jednostavan. XML dokumenti mogu biti mnogo složeniji, sa složenijim DTD-ovima i XSL-ovima.

Evo, uzmimo za primjer djelatnost gdje je razmjena podataka vitalna, kao što je bankarstvo. Svaka banka koristi neki interni sistem za praćenje transakcija, ali ako bi koristili standardizirani set XML elemenata preko weba, mogli bi lako opisati transakcije drugom bankovnom sistemu ili aplikaciji (kao što je MS Money), ili bi čak sve to mogli prikazati na Webu lijevo formatirano.

Mnogo je lakše i praktičnije voditi aukcijski web sajt pomoću XML-a i nekog serverskog scripting jezika, nego koristeći baze podataka, client i server-side scripting i HTML. Ako imate firmu koja prodaje robu i preko Weba želite da se zajedno sa promjenom cijena ažurira i cjenovnik na web sajtu, koristit ćete XML. Ako je cjenovnik u XML formatu, isti dokument može služiti za web (sa jednim XSL-om) i za štampanje cjenovnika (sa drugim XSL-om). Koristeći isti set elemenata (tagova) možete biti uvijek u toku sa svojim poslovnim partnerima, jer će razmjena cijena pomoću XML-a biti prava sitnica. Košaricu u kojoj se sprema sve što je posjetitelj kupio na web sajtu također je najlakše realizovati koristeći XML.

18. Budućnost: Ajax, Gadget,...

18.1. Ajax

Vremenom se u razvoju web aplikacija postavilo vrlo logično pitanje: kako dodati mogućnosti koje imaju klasične desktop aplikacije? I još pri tome poboljšati performanse i omogućiti potpunu neovisnost od browser-a sa istim brojem kontrola koje desktop aplikacije čine toliko boljim i jednostavnijim za programiranje. Ovo je zapravo ideja AJAX (Asynchronous JavaScript and HML) strategije za razvoj web-baziranih aplikacija.

Sam AJAX je nastao da bi se prevazišla ograničenja klasičnog request-response modela današnjih web aplikacija. Svaki put pri promjeni podataka, odnosno korisničkog interfejsa, stranica se mora poslati serveru kako bi se ažurirala. Međutim, velika količina ovih zahtjeva sa velikim skupom podataka može uzrokovati da korisnik čeka minutama za obradu jednog zahtjeva, što naravno nije poželjno: AJAX omogućava da klijentske skripte asinhrono (neovisno jedna od druge) pozivaju serverske aplikacije koristeći HML kao protokol. To za rezultat ima da se stranica mnogo brže učitava, jer se mnoge skripte mogu izvršavati paralelno, a obnavlja se samo onaj dio stranice koji se mijenja, bez potrebe za učitavanjem statičkog dijela stranice.

Drugi razlog za AJAX-om je i nedostatak kontrola u web aplikacijama, što je imalo za posljedicu da programeri naprave browser-specific kontrole čija je izrada prilično kompleksna i ograničava upotrebu web aplikacije isključivo u jednom browser-u ili čak u samo jednoj verziji nekog od browser-a. Većina programera, logično, izbjegava navedeni način razvoja kontrola, pa su opet korisnici ti koji su nezadovoljni mogućnostima web aplikacije. AJAX koristi JavaScript komponente koje daju više mogućnosti korisničkom interfejsu, pri čemu je kompatibilnost zagarantovana.

Izrada AJAX-baziranih aplikacija nažalost nije jednostavna i zahtijeva prilično znanja o JavaScript-u i dodatnim tehnologijama, premda postoji priličan broj biblioteka koje ovaj posao olakšavaju.

Da biste počeli raditi sa AJAX-om morate kombinovati HTML, DHTML, CSS XHTML i, naravno, JavaScript. Ukoliko dublje krenete u problematiku shvatit ćete da najviše posla imate sa DHTML-om, XMLHttpRequest ili odgovarajućim Microsoft-ovim ActiveX objektom. Moramo primijetiti da programer odmah ima probleme vezane za kompatibilnosti stranice u odgovarajućem browser-u, jer mora postojati detekcija u samoj stranici kako bi se koristio XMLHttpRequest ili Microsoft-ov ActiveX objekat.

Sljedeći problem je vezan za konceptualni dizajn web stranice, odnosno iskorištenje paralelnog izvršavanja skripti. Suština paralelizma se sastoji u tome da se inicijalno koristi minimalan broj JavaScript-i, a zatim postepeno povećava korištenje preostalih JavaScript-i, CSS-a i drugih tagova preko XMLHttpRequest objekta. Ova tehnika koristi veći broj konekcija prema serveru i značajno ubrzava

učitavanje web stranice, za razliku od tradicionalnog pristupa koji je sinhronizovan i koristi jednu konekciju. Ovdje opet možemo primijetiti da programer mora paziti i na redoslijed izvršavanja JavaScript-i, CSS i ostalih tehnologija koje su klijentu potrebne.

I kada pomislite da se problemi ovdje završavaju, prilikom programiranja dolazite do situacije da morate imati ponovnu detekciju u stranici. Ovaj put morate znati u kojem se stanju stranica nalazi, te u zavisnosti od toga pozvati odgovarajuću skriptu. Možete problem riješiti rekurzivno, međutim, održavanje postaje komplikovanije kako imate veći broj stanja stranice.

Ipak, održavanje nije uplašilo Google, koji nam je poklonio Gmail, Google Maps, Google Suggests i slične AJAX-bazirane servise. Ohrabreni ovim potezom, programeri su počeli AJAX prebacivati i na .NET pa ćete na Internetu pronaći i njegovu varijantu Ajax.NET.

Iskusniji arhitekti moraju biti i kritičari AJAX strategije razvoja web stranica. Naime, svaki informacioni sistem se može apstrahovati u tri logička dijela - prezentacijski dio sa kojim rade korisnici i administratori, biznis logika koja bi trebala biti transparentna i na kraju, naravno, perzistencija. AJAX spaja prezentaciju sa biznis logikom kako bi se povećavala efikasnost, brzina rada i jednostavnost upotrebe, ali znatno smanjuje kvalitete informacionog sistema, posebno njegovu fleksibilnost. Može se lako zaključiti da je AJAX idealno rješenje za male, jednostavne i često korištene servise.

Microsoft-ov odgovor na pitanje kako pojednostaviti AJAX i iskoristiti sve njegove mogućnosti je ASP.NET 2.0 Atlas tehnologija. Na klijentskoj strani Atlas pruža impresivan skup JavaScript biblioteka koje sadrže objektno-orijentisane mogućnosti, HTML/HTTP za pozive procedura na udaljenim računarima, te komponente sa event-ima i često korištenim metodama poput drag and drop, auto completion, popup i mouse hovering.

Na serverskoj strani Atlas se sastoji od ASP.NET serverske kontrole i ASP.NET servisa. Serverske kontrole su odgovorne za slanje klijentskih skripti na korisničku mašinu, čime se omogućava transparentan i neovisan rad od korisničkog browser-a. Servisi nam omogućavaju da čuvamo podatke o korisniku kao što je novi ASP.NET profiles koji je bio poznat pod nazivom Application i Session varijable. Personalizacija interfejsa, autentifikacija i uloge (roles) omogućuju da korisnik u zavisnosti od privilegija dobije manji ili veći broj kontrola na interfejsu i da iste posloži po svom nahođenju.

18.2. Ajax Agent 0.1

Ajax Agent je biblioteka napisana za PHP, koja na jednostavan način daje mogućnost korištenja Ajax tehnika u kombinaciji sa spomenutim skriptnim jezikom. Ajax (Asynchronous JavaScript and XML) predstavlja potpuno novi

pristup izgradnji web aplikacija korištenjem već postojećih metoda (Javascript, Document Object Model (DOM) i XMLHttpRequest).

Od mnogih biblioteka za Ajax posebno je zanimljiva ova, jer je jednostavna za korištenje i daje pregledan programski kod. Prva, verzija ove biblioteke se pojavila 3. februara 2006. pod GNU licencom, što znači da je potpuno besplatna. Čitavu biblioteku, koja iznosi 21 kB, možete preuzeti sa linka www.ajaxagent.org.

Ajax Agent predstavlja samo jedna datoteka (agent.php), a ostalo su testni primjeri. Instalacija je veoma jednostavna, dovoljno je da pomenutu datoteku smjestite u direktorij u kojem želite da kreirate PHP/Ajax baziranu aplikaciju.

Drugi korak je da kreirate neku PHP skriptu i u njoj priključite Ajax biblioteku na sljedeći način:

```
Include_once("agent.php");
```

Kako vidite, koristite običnu PHP komandu za pozivanje neke datoteke. Treći korak je komanda:

```
$agent->init();
```

Njom se inicira Agent klijent. Vrlo je važno spomenuti da se ova komanda mora upisivati između HTML tagova <head> i </head>. Četvrti korak je komanda:

```
Agent.call('url', 'server_fuction', 'client_handle',  
param1, param2, ...);
```

Ovo je najvažnija komanda i ona vrši spajanje server i klijent funkcija u jednu koja se ponaša kao klijent. Sastoji se od nekoliko parametara. Prvi je url - kako samo ime kaže, pokazuje na neku web adresu. Server_function predstavlja naziv server side funkcije, dok client_handle predstavlja naziv funkcije na strani klijenta. Param1, param2 može biti bilo šta; string, objekat, niz... Kako vidite, Agent je veoma jednostavan za korištenje, a da biste shvatili bolje kako ovo funkcioniše, evo malo praktičnih primjera.

Prvi primjer će biti ispisivanje nekog teksta. Najprije napravite datoteku tekst.php sa sljedećim sadržajem:

```
<html><head>  
<?php  
function server-tekst()  
{return "Koristite Ajax Agent";}  
include_once("agent.php");  
$Agent->init();  
?>  
<script>
```

```
function veza_tekst(str)
{
    agent.call(", 'server_tekst', klijent_tekst");
}
function klijent_tekst(str)
{
    dpci=document.getElementById("div_ispis").innerHTML=str;
}
</script>
</head><body>
<p><b><a href="#" onclick="veza_tekst()">Ispisi
tekst</a><b></p>
<div id="div_ispis"></div>
</body></html>
```

Ako je sve kako treba, nakon klika na link će se ispisati tekst (Koristite Ajax Agent) bez ponovnog učitavanja cijele stranice. Kako vidite, funkcija u sebi sadrži tri odvojene funkcije. Prva je server-side funkcija (server_tekst()) koja samo vraća vrijednost stringa. Druga je klijent funkcija (klijent_tekst) koja određuje gdje će se tekst ispisivati. U ovom slučaju to je div tag kod kojeg id parametar ima vrijednost div_ispis. Treća funkcija povezuje prethodne dvije (function veza:tekst) i ponaša se kao klijent server funkcija. Na kraju se poziv funkcije veza_tekst vrši kao i poziv bilo koje druge JavaScript funkcije (onclick="veza:tekst()"). Sve ostalo su obične PHP i JavaScript komande.

Nakon ovog napravite datoteku tekst2.php sa sljedećim sadržajem:

```
<html> >head>
<?php
function server_tekst()
{
    return "Koristite Ajax Agent";
}
include_once("agent.php");
$agent->init();
?>
<script>
function klijent_tekst()
{
    str = agent.call(", 'server_tekst',");
    alert(str);
}
</script>
</head><body>
```

```
<p><b><a href="#" onclick=" klijent_tekst  
() ">Ispisi tekst</a></b></P>  
</body></html>
```

Rezultat klika na link će biti ispis teksta, ali ovaj put ne na stranici, nego u alert prozoru. Možete vidjeti da ova skripta ima dvije funkcije. `Server_tekst()` isto kao i u prethodnom vraća vrijednost stringa, dok funkcija `klijent_tekst()` u ovom slučaju uzima vrijednost serverske funkcije (`str = agent.call('server-tekst',')`) i ispisuje poruku u alert prozoru (`alert(str)`).

Osim ovoga, Agent je sposoban da radi i sa parametrima, što možete vidjeti na sljedećem primjeru:

```
<html> <head>  
<?php  
    function server_tekst($ime, $mail)  
{ return " Vi se zovete: ".$ime, "<BR> Vas e-mail  
je:".$mail;}  
include_once("agent.php");  
$agent->init();  
?>  
<script>  
    function veza_tekst() {  
        ime = document.getElementById('ime').value;  
        mail = document.getElementById('mail').value;  
        agent.call(", 'server_tekst', 'klijent_tekst', ime, mail);  
    }  
    function klijent_tekst(str)  
    {document.getElementById('div_ispis').innerHTML=str,}  
</script>  
</head><body>  
<p><form>  
    Ime: <input name="ime" id="ime" size="4" type="text">  
    E-mail: <input name="mail" id="mail" size="4"  
type="text">  
    <input name="OK" value="Posalji" onclick="veza_tekst();  
return false; "type="submit">  
</form></p>
```

```
<div id="_ispis"></div>
</body></html>
```

Rezultat ovog primjera će biti ispis imena i mail-a koje se unese u formu bez osvježavanja web stranice. Tekst će se ispisati odmah ispod forme nakon klika na dugme. Kako vidite, skripta sadrži tri funkcije. Prva je server-side funkcija `server_tekst()` koja, za razliku od prethodnih primjera, ima dva parametra, ime i prezime. Zadatak joj je da vrati njihove vrijednosti. Druga funkcija je `klijent_tekst()` čiji je zadatak isti kao i u prethodnim primjerima - da ukaže gdje će se tekst ispisivati. U ovom slučaju to je `div` tag čiji parametar `id` ima vrijednost `div_ispis`. Funkcija `veza_tekst` daje vrijednost parametrima ime i prezime i onda ih povezuje sa gore navedenim funkcijama.

Sljedeća skripta će ujediniti sve do sada pomenuto. Napravite datoteku `glavna.php` sa sljedećim sadržajem:

```
<html> <head>
<?
    function server_glavna($src)
    {
        $str."<br/>";
        $str."<iframe width=100% height=6'' frameborder=0
marginwidth=0 src=$src></iframe>";
        return $str;
    }
    include_once("agent.php");
    $agent->init();
?>
<script>
function klijent_glavna(str)
{document.getElementById('div_ispis'.innerHTML=str;}
</script>
</head><body>
<a href="#"
onclick="agent.call(", 'server_glavna',
'klijent_glavna', 'tekst.php')">Tekst1</a>
<a href="#"
onclick="agent.call)", 'server_glavna',
'klijent_glavna', 'tekst2.php')">Test2</a>
```

```

<a href="#"
onclick="agent.call(", 'server_glavna',
'klijent_glavna', 'tekst3.php')">Tekst3</a>

<div id="div_ispis"></div>

</body></html>

```

Prilikom klika na neki od linkova prikazat će se jedan od prethodnih primjera bez osvježavanja stranice. U ovom slučaju imamo samo dvije funkcije, server-side funkciju `server_glavna()` koja pokazuje kako će se prikazivati datoteke, i klijent `glavna()` koja ukazuje gdje će se datoteke izlistati. I ovaj put je to `div` tag čiji parametar `id` ima vrijednost `div_ispis`. Sada komanda `agent.call` povezuje obje funkcije sa određenom datotekom (u ovom slučaju to su: `tekst.php`, `tekst2.php` i `tekst3.php`), a poziva se direktno komandom `onclick`.

18.3. Gadget

Riječ "gadget" (gizmo, gismo, widget) se koristi da označi uređaj koji je posebno koristan za neki posao, mali elektronski ili mehanički aparat koji predstavlja inovaciju, neimenovani ili hipotetički proizvod.

U programerskom žargonu, riječ gadget se koristi za kombinaciju grafičkih simbola i nešto programskog koda s ciljem izvršenja neke određene funkcije. Naprimjer, scroll-bar ili tipka predstavljaju gadgete.



Slika 93. Windows Vista Beta 2 Desktop sa Windows Sidebarom i gadgetima

Gadgeti su mini aplikacije koje se mogu povezivati sa internet servisima kako bi davale informacije o vremenskoj prognozi, novostima, auto-kartama, Internet radio stanicama i galerijama slika.

Gadgeti se također mogu integrisati sa mnogim aplikacijama da bi promijenili način interakcije s njima. Naprimjer, gadget može dati brzi pregled adresa iz chat programa, dnevne obaveze iz kalendara i sl. Gadgeti također mogu biti kalkulatori, igre, itd.

Microsoft Windows Vista će uz standardne elemente grafičkog interfejsa (desktop, taskbar, start meni) imati i osnovni set gadgeta. Dodatni gadgeti se mogu skinuti sa online galerije gadgeta (<http://gallery.microsoft.com/>). Windows Sidebar je novi grafički element Windows Vista desktopa koji se koristi za organizovanje gadgeta (slika 93). Pored sidebara, gadgeti mogu biti sastavni dijelovi Internet stranica.

Windows Sidebar je aplikacija koja omogućuje da se gadgeti prikazuju na sidebaru i na Windows desktopu. Svi gadgeti iz sidebara se pohranjuju lokalno. Gadgeti se mogu dobiti skidanjem sa nekog web sajta, kao e-mail attachment ili uz instalaciju neke aplikacije. To su datoteke sa ekstenzijom .gadget. Dvostrukim klikom na te datoteke pokrenut će se proces instalacije.



Slika 94. Gadget na desktopu

18.3.1. Kako rade gadgeti

Svaki gadget se razvija pomoću HTML i skriptnog jezika. Gadgeti također imaju pristup dodatnim informacijama kad se HTML pokreće kao gadget. To omogućuje gadgetima da imaju interakciju sa Windows datotekama i folderima, kao što su

prikazivanje slika sa lokalnog diska ili davanje informacija o bežičnoj konekciji. Složeni gadgeti koriste skriptni jezik bez korištenja drugih binarnih objekata (spoljnih datoteka).

18.3.1. Gadget datoteke

Gadget sadrži sljedeće datoteke:

Gadget Manifest	XML datoteka koja definiše osobine gadgeta, uključujući ime, ikonu i opis
HTML datoteka	Definiše osnovni kod za gadget
HTML datoteka postavki	Omogućuje da korisnik mijenja postavke
Slike, skriptu i Style Sheets	Za upotrebu u HTML
Ikona	Za upotrebu u programu za izbor gadgeta

18.3.2. Kreiranje gadgeta

Kreiranje gadgeta je proces vrlo sličan izradi web stranice:

- Kreiranje direktorija za smještaj gadget datoteka.
 - Kreiranje HTML stranice koja radi nešto zanimljivo
 - Kreiranje XML datoteke za gadget manifest
 - Testiranje novog gadgeta iz sidebara
 - Zatvaranje gadgeta radi eventualnih promjena
 - Otvaranje gadgeta da bi se vidjele te promjene
1. Otvoriti sidebar
Start, All Programs, Accessories, The Windows Sidebar
 2. Locirati vaš folder za gadgete
Start, Run,
%userprofile%\AppData\Local\Microsoft\Windows Sidebar\Gadgets
 3. Kreirati gadget folder:
Kreirajte folder pod imenom "HelloWorld.gadget" unutar foldera Gadgets. Ime gadget foldera mora imati ekstenziju .gadget da bi ga sidebar mogao prepoznati

4. Kreirajte HTML datoteku

HTML datoteka bi trebala sadržati sljedeće:

```
<html>
<head>
  <title>Hello, World!</title>
  <style>
    body {
      width:130;  height:50;
    }
  </style>
</head>
<body>
  <span id="gadgetContent" style="font-family: Tahoma;
font-size: 10pt;">Hello, World!</span>
</body>
</html>
```

5. Konačno, kreirajte gadget manifest:

Datoteka manifesta se mora zvati "gadget.xml"

Datoteka mora biti snimljena sa UTF-8 encoding.

XML datoteka bi trebala sadržati sljedeće:

```
<?xml version="1.0" encoding="utf-8" ?>
<gadget>
  <name>Hello World!</name>
  <namespace>Example.You</namespace>
  <version>1.0</version>
  <author name="Vase ime">
    <info url="www.example.com" />
  </author>
  <copyright>2006</copyright>
  <description>Moj prvi gadget</description>
  <hosts>
    <host name="sidebar">
      <base type="HTML" apiVersion="1.0.0"
        src="HelloWorld.html" />
    </host>
  </hosts>
</gadget>
```

```
<permissions>full</permissions>
<platform minPlatformVersion="0.3" />
</host>
</hosts>
</gadget>
```

Može se kreirati i datoteka za ikonu, kojoj treba dati isto ime koje ima i HTML datoteka (HelloWorld). Ako se ne kreira posebna, Windows sidebar će obezbijediti generički ikonu.

Kad se klikne na simbol "+" na dnu sidebara, u galeriji gadgeta trebao bi se pojaviti novi gadget.



Slika 95. Tipka "Add Gadget" za dodavanje novih gadgeta



Slika 96. Gadget Gallery sa gadgetom "Hello World!"

Datoteka sa slikom (naprimjer "Background.png") se može koristiti kao podloga gadgeta. Potrebno je samo izmijeniti CSS dio HTML datoteke "HelloWorld.html":

```
<style>
```

```
body {  
    width:130;  
    height:50;  
    background: url('Background.png');  
    padding-top: 5px;  
}  
</style>
```

18.3.3. Sistemski API-ji gadgeta

Platforma gadgeta obezbjeđuje objektni model koji omogućuje interakciju sa operativnim sistemom. Gadgeti mogu izvršavati radnje kao što je čitanje datoteka i fodlera, uzimanje informacija od hardvera, te prikaz informacija o mreži i Windows operativnom sistemu.

U sljedećem primjeru, gadget će se koristiti za prikaz vrijednosti sistemske varijable.

```
<html>  
<head>  
    <title>Hello, World!</title>  
    <style>  
        body{  
            width:130;  
            height:50;  
            background: url('Background.png'); }  
            padding-top: 5px;  
        </style>  
        <script>  
            var variableName = "ProgramFiles";  
            function setContentText() {  
                gadgetContent.innerText =  
                System.Environment.GetEnvironmentVariable(variableName)  
                ;  
            }  
        </script>  
    </head>
```

```
<body onload="setContentText()">
  <span id="gadgetContent" style="font-family: Tahoma;
font-size: 10pt;">Hello, World!</span>
</body>
</html>
```



Slika 97. Gadget sa sistemskom varijablom

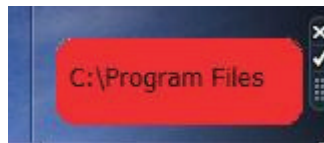
18.3.4. Postavke gadgeta

Gadjeti mogu prikazati dijalog za podešavanje postavki. Oni također mogu snimiti postavke sa APIjem.

Prvo je potrebno konfigurisati HTML stranicu koja će se pojaviti kad korisnik pritisne UI kontrolu za podešavanje postavki:

```
<script>
  var variableName = "ProgramFiles"
  function setContentText() {
    gadgetContent.innerText =
System.Environment.getEnvironmentVariable(variableName)
;
  }
  System.Gadget.settingsUI = "Settings.html";
</script>
```

To će omogućiti da oznaka na za podešavanje gadgetu postane vidljiva:



Slika 98. Gadget sa uključenom oznakom za postavke

sada je potrebno kreirati html stranicu koja će se koristiti za podešavanje postavki (datoteka sa imenom "Settings.html"):

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

```
        body{
            width:250;
            height:75;
        }
    </style>
    <script>
        function init() {
            var currentSetting =
System.Gadget.Settings.read("variableName");
            if (currentSetting != "")
                envVar.innerText = currentSetting;
        }

        System.Gadget.onSettingsClosing =
SettingsClosing;

        function SettingsClosing(event)
        {
            if (event.closeAction == event.Action.commit) {
                variableName = envVar.value;
                System.Gadget.Settings.write("variableName",
variableName);
            }

            event.cancel = false;
        }
    </script>
</head>
<body onload="init()">
    <span id="mySpan" style="font-family: Tahoma; font-
size: 10pt;">
        Environment Variable:<br>
        <input type="text" id="envVar" length="40">
    </span>
</body>
</html>
```

U ovoj html stranici postoji posebna funkcionalnost gadgeta:

1. Prvo se provjerava da li već postoji preferencija korisnika o tome koja sistemska varijabla treba da se koristi:

```
function init() {  
    var currentSetting =  
    System.Gadget.Settings.read("variableName");  
    if (currentSetting != "")  
        envVar.innerText = currentSetting;  
}
```

2. Odredi se funkcija koja će se koristiti kad korisnik klikne na OK:

```
System.Gadget.onSettingsClosing = SettingsClosing;  
function SettingsClosing(event)  
{  
    if (event.closeAction == event.Action.commit) {  
        variableName = envVar.value;  
        System.Gadget.Settings.write("variableName",  
        variableName);  
    }  
    event.cancel = false;  
}
```

3. Kreira se funkcija za pohranjivanje vrijednosti iz text boxa:

```
function SaveSettings() {  
    variableName = envVar.value;  
    System.Gadget.Settings.write("variableName",  
    variableName);  
}
```

Tada će se pojaviti dijalog za podešavanje postavki u kojem korisnik može unijeti naziv nove sistemske varijable koju treba prikazati (slika 99).



Slika 99. Gadget sa dijalogom za podešavanje postavki

Posljednji korak je ažuriranje (update) gadgeta kad korisnik promijeni postavku (u datoteci "HelloWorld.html"):

```
<script>
    var variableName = "ProgramFiles";
    function setContentText() {
        gadgetContent.innerHTML =
        System.Environment.getEnvironmentVariable(variableName)
        ;
    }
    System.Gadget.settingsUI = "Settings.html";
    System.Gadget.onSettingsClosed = SettingsClosed;
    function SettingsClosed() {
        variableName =
        System.Gadget.Settings.read("variableName");
        setContentText();
    }
</script>
```

19. Planiranje i procedura dizajna web prezentacije

Nakon što ste odlučili da se (konačno) pojavite na Internetu u vidu prezentacije firme, ponude određenih usluga ili u vidu ličnog predstavljanja, dolazi jedna od najbitnijih faza u izgradnji sajta - planiranje. Kao i svakom drugom poslu, od toga zavisi koliko konkretnog posla ćete imati u fazama koje slijede, a kvalitetno planiranje će doprinijeti ukupno boljem rezultatu.

Cjelokupno planiranje možemo podijeliti u dvije faze: osnovno planiranje u kojem "profilirano" sajt i planiranje dizajna, u kojem na osnovu tih informacija planiramo njegov izgled. Druga faza je bliža implementaciji - samom procesu izgradnje sajta.

19.1. Profiliranje

Prije svega, potrebno je definisati šta želite postići svojom pojavom na Internetu? Postoji puno razloga zbog kojih možete odlučiti "ići online"; od lične prezentacije ili objave portfolija, na koju se može odlučiti svaki Internet korisnik ili profesionalac u nekom poslu (slobodni dizajner/umjetnik, novinar i sl.), do prezentacije firme ili čak ponude određenih usluga ili aplikacija, što znači da ćete Internet iskoristiti kao poslovnu alatku.

Sve mogućnosti su tu, a na vama je da izaberete ono što želite. Ovdje ću odmah upozoriti da posjedovanje web sajta košta i da dobro razmislite da li vam se isplati. Jer, "trčati pred rudu" i napraviti sajt po svaku cijenu, a potom ga ne osvježavati, pogrešan je pristup i antireklama, pa ga izbjegavajte. Na žalost, domaći je web pun ovakvih loših primjera.

Od razloga zbog kojeg ćete napraviti sajt zavisi sve što dolazi poslije: izbor tehnologije koju ćete koristiti u njenoj izgradnji, izbor web hosting provajdera, pa na kraju i sam pojavni oblik, dizajn sajta. Jer, svima je jasno da će se stranica ličnog predstavljanja znatno razlikovati od prezentacije firme, web pretraživača ili web prodavnice, kako u dizajnu, tako i u korištenim tehnologijama. To na kraju najviše utiče na konačnu cijenu.

19.2. Planiranje dizajna

Nakon što ste odlučili kakav sajt želite imati, ulazite u drugu, konkretniju fazu. A u njoj je najbitnije pravilno strukturirati sajt i odrediti koje informacije ćete na njemu ponuditi. Svaki web sajt zasniva se na strukturi foldera, gdje svaki folder predstavlja jednu sekciju u koju se smještaju web stranice sa sadržajem. Veći sajtovi mogu imati puno sekcija, pa čak i podsekcije, što zavisi isključivo od količine ponuđenih informacija i dokumenata. U našim uslovima, najvjerovatnije je da ćete završiti samo sa sekcijama.

Najbolji prijatelji u ovoj fazi biće vam dobri stari "analogni" papir i olovka. Razmislite malo i skicirajte na papiru strukturu, odnosno podijelite informacije u

logične cjeline - sekcije. Sajt predstavljanja nekog dizajnera / fotografa i sl. sigurno će imati sekciju njegovih ličnih podataka, dio u kojem će posjetioci vidjeti njegove radove, sekciju za kontakt i sl. Nasuprot tome, stranica novinske kuće imat će puno podstranica sa raznim tekstovima - osvrtima urednika i autora, tekstove iz pojedinih rubrika i sl.

Obično se dizajnira naslovna stranica i jedan šablon (template) za podstranice, u koje se potom ubacuju informacije i dokumenti. Zavisno od dizajna, šabloni podstranica mogu biti različiti. Dizajn treba da oslikava namjenu stranice, znači treba da ima odgovarajuću šemu boja, izbor tipografije (pisma i veličine), određenu upotrebu slika i sl.

Po završetku ove faze, trebali bi na hard disku imati spremne digitalne dokumente (tekstove, slike i sl.) i jasnu strukturu sajta. Imate dakle sve preduslove da idete dalje - na konkretan razvoj u nekom od HTML alata.

Ako ste pažljivo išli korak po korak, logično zaključivali i sve fino zapisali, faza izgradnje biće uveliko olakšana, jer imate jasne vodilje u i cilj kojem stremite. Ostaje dio dizajniranja izgleda sajta.

19.3. Struktura sajta

Svaki web sajt sastoji se od glavne (home) stranice i podstranica, a svake od njih sadrže pojedine bitne elemente. Na osnovnoj (home) stranici trebaju biti izdvojeni dijelovi informacija iz svake od sekcija, sa linkovima koji vode na odgovarajuće podstranice. Ne morate izdvojiti sve informacije, nego treba izabrati one koje su bitne, zavisno od namjene sajta. Osnovna stranica ne treba biti prenatrpana, jer će prevelika količina informacija, ali i linkova zbuniti posjetioca, pa možda ga čak i otjerati sa stranice.

19.3.1. Slike

O korištenju slika treba dobro razmisliti. Više slika znači duže učitavanje, pa zato oprez! Bez slika se ne može, inače bi sve izgledalo kao tehnički priručnik, a ne kao dizajnersko djelo. Osnovnu stranicu nemojte preopteretiti, jer se ona učitava prva, a duže čekanje zna nervirati (znate po sebi - kada se neka stranica ne učitava u prvih pola minute, vjerovatno ćete otići na neku drugu). Na podstranicama si zato možete dati više slobode. Ako koristite više većih slika (nešto kao galerija), bolje je na jednoj stranici dati prikaz njihovih umanjenih verzija (brže se učitavaju), a klikom na svaku od slika dajte posjetiocu mogućnost da pogleda veću verziju. Veća slika se može učitati u istom dokumentu, ali je nekad bolje rješenje "iskačući" (popup) prozor. Nakon što posjetilac pogleda sliku i zatvori prozor, može nastaviti pregledati stranicu tamo gdje je stao. Opet, imajte u vidu i da neki posjetioci ne vole iskačuće prozore.

19.3.2. Fontovi

Fontovi su divna stvar, čije ljepote na žalost gotovo nikako ne možemo iskoristiti na Internetu. Kako nijedan korisnik nema instalirane iste fontove, to se pri izradi web stranica moraju koristiti samo oni za koje ste sigurni da su prisutni na svakom računar. Ovo obično sužava izbor na Arial, Times New Roman, Verdana i Courier New. Iako su prve specifikacije HTML-a predviđjele da browser odabire fontove, a dizajner samo bira vrstu (neproporcionalni, serif i sans serif), ovo se slabo koristi, jer većina dizajnera želi da ima što veću kontrolu nad onim što će biti prikazano na korisnikom računar. Zbog toga vodite računa koje ćete fontove koristiti. Naslove i neke detalje za koje vam treba neki kičasti font koji imate samo vi (i vaš prijatelj) prikažite kao slike, ali umjereno zbog veće spomenutih razloga.

19.3.3. Navigacija

Bitan element svakog web sajta i stranice je navigacija. Pod navigacijom se podrazumijeva skup linkova, predstavljen tekstualno ili grafički, koji su u svakom trenutku prisutni i mogu odvesti na glavne dijelove sajta. Sa svake podstranice treba biti dostupna svaka druga sekcija i jasno naglašena mogućnost povratka na prethodnu ili osnovnu stranicu. Iz ove definicije mogu se zaključiti osobine koje navigacija treba da posjeduje. Neprimjenjivanje svih pravila dovodi do konfuznog sajta i lošeg snalaženja u njemu.

Na svakoj stranici mora se nalaziti identična navigacija, koja će uvijek stajati na istom mjestu. Uvijek mora sadržavati linkove na ili sve ili glavne sekcije sajta. Navigacija mora biti uočljiva, ali ne prenapadna da odvlači pažnju korisnika od sadržaja. Najčešće se koriste boje kontrastne ostatku stranice. Iako se ovo ne mora uvijek striktno odnositi na navigaciju, korisnik uvijek mora biti u stanju pri prvom pogledu na stranicu znati koja je to stranica i na šta se odnosi. Ovo se najčešće radi drugačijim izgledom linka na navigaciji na željenu stranicu (koji više nije link).

Navigacija se najčešće smješta u lijevu ili gornju stranu: Dosta rjeđe srećemo rješenja navigacije s desne ili donje strane. Iako nema nekog pravila o tome kada staviti navigaciju na neko mjesto, ponekad neke stvari ne možemo izbjeći. Za veliki broj linkova preporučuje se smještanje navigacije u lijevu ivicu zbog puno više mjesta. Ako vam treba više sistema navigacije - jedan primami za cijeli sajt i jedan sekundami koji će biti prisutan samo na određenom skupu strana (jednoj rubrici), primarna navigacija se smješta uz gornju ivicu, a sekundama (kontekstno zavisna) uz lijevu ivicu. Kao što smo već naveli, nekog pravila kojeg bi se trebalo pridržavati nema, pa ako ste nesigurni - eksperimentišite!

Preporučljivo je da korištenjem neke tehnologije navigacija uvijek bude u jednoj datoteci koja će se uvijek prikazivati na stranici putem različitih tehnologija (server-side include, okviri...)

19.4. Izgradnja: sklapanje sadržaja u cjelinu

Suprotno popularnom mišljenju, dizajniranje sajta je tek pola (ili čak manje) posla. Zavisno od dizajna i tehnologija koje se odlučite primijeniti, sklapanje svih slika, tekstova, skripti, opcionalno baza, CSS datoteka i mnogočega drugoga može biti velik problem i postati noćna mora.

Spremili ste sve slike i tekstove koje ćete izbaciti na sajt, razradili dizajn i sada to želite pretočiti u oblik koji će browser prikazati. Iako nije neophodno, jako je preporučljivo napraviti dobru strukturu foldera po kojima ćete razmještati datoteke. Slike dobijaju svoj folder, svaka rubrika sajta svoj folder. Ako planirate imati više različitih CSS datoteka sa različitim stilovima, napravite poseban folder za njih. Tako umnogome olakšavate sami sebi snalaženje, a i drugim korisnicima ažuriranje podataka na sajtu.

Pretpostavit ćemo da ste već napravili neku bitmap sliku koja sadrži osnovni izgled svih vaših stranica, te sve grafičke elemente koje ćete koristiti na njima. Ukoliko to niste uradili, preporučujem da to učinite. Dizajn sajta najbolje je raditi u nekom od vektorskih alata, poput CorelDRAW-a ili Illustrator-a, jer je lakše. Tek kada ste potpuno zadovoljni izgledom stranice unutar nekog grafičkog programa, pređite na pravljenje HTML-a.

Prije nego počnete ovaj proces, odlučite koje ćete sve tehnologije koristiti. Da li isključivo tabele, okvire, slojeve.... Sve ima svoje prednosti i mane. Nakon toga u grafičkom programu prvo napravite svu grafiku. Uvijek je bolje staviti više manjih slika nego jednu veću. Postiže se brže učitavanje, posjetilac ima osjećaj da se nešto dešava i - što je najbitnije - mogu se koristiti različiti formati zapisa, što pravilnom primjenom umnogome smanjuje veličinu i poboljšava kvalitet.

Nakon toga, napravite osnovni šablon (template) po kojem ćete praviti sve ostale (pod)stranice. Time postićete konzistentnost, što je bitno kod web sajta, da bi on ostavio dojam kompaktnosti i da bi posjetilac prilikom istraživanja po njemu uvijek znao na kojem dijelu se nalazi.

Ako imate neke dijelove koji se ponavljaju (npr. navigacija, spisak linkova), gledajte da oni stoje u jednoj datoteci koju ćete poslije samo pozivati, upotrebom okvira, server-side include naredbi, različitih skripti i sl. Dosta ćete olakšati održavanje, naknadno ažuriranje podataka, ispravljanje grešaka...

Kada završite proces izgradnje kostura sajta, sačuvajte ga negdje, kao šablon, a tek nakon postojanja rezervne kopije počnite "punjenje" podacima. Dok se ovo može činiti glupo kod pravljenja manjih sajtova, svaka nadogradnja, dodavanje novih sadržaja može poći po zlu. Zato je rezervna kopija nešto što može uštediti puno živaca.

Napravili ste sajt upotrebom raznih alata i tehnologija, sada osigurajte da web server podržava sve tehnologije koje se na njemu koriste. ASP, server side scripting i sl. moraju biti podržane i ispravno konfigurisane da bi radile. Zavisno od servera i datoteka, moguća su ograničenja na imena datoteka koja ćete koristiti

(npr. ekstenzije datoteka sa server side include naredbama moraju imati .HTML ili .SHTML ekstenziju).

Nakon prebacivanja datoteka, sve naknadne promjene koje ćete uraditi na sajtu uradite i na lokalnoj kopiji, kao i na serveru. Uzrok mnogih grešaka u konfuzije je neusklađenost lokalnih podataka sa onim na serveru. Upravo zato treba imati identičnu strukturu foldera i lokalno i na serveru, te identične datoteke. Time olakšavate ispravljanje grešaka i održavanje linkova.

Nakon što ste izvršili kopiranje datoteka na funkcionalan web server (upload), vaš sajt je postao online - dostupan cijelom svijetu na razgledanje! Sav vaš trud konačno je pretočen u nešto "opipljivo" i nešto po čemu možete "klikati" mišem. No, posao još nije gotov, jer treba prekontrolisati sve linkove i podatke. Iako ste ovo već uradili na lokalnoj verziji sajta, greške se uvijek potkradaju i bolje je da svaku nađete vi, a ne posjetilac.

19.4.1. Flash ili ne?

Ako pravite prezentaciju firme ili nekog proizvoda i želite izvući "aaaaaaa" uzdahe iz posjetilaca, koristit ćete Flash. To je tehnologija koja omogućava da sajt obogatite multimedijom: animacijama, zvukom i raznim drugim za oko lijepim efektima. Praktično, u Flash-u možete napraviti cijeli sajt ili samo pojedine dijelove.

Budite pažljivi prilikom donošenja odluke o korištenju Flash-a, jer za njegov prikaz u browseru posjetilac treba imati instaliran Flash dodatak. Ovo je sve manji problem, jer noviji browseri dolaze sa insntaliranim dodatkom, a i veličina je stotinjak kB, pa se lako može kopirati s Interneta. Obazrivost ostaje, jer za neke sajtove Flash jednostavno nije primjeren. Treba ga izbjegavati kod sajtova koji nude puno informacija, državnih sajtova i sl. Nasuprot tome, može odlično poslužiti za predstavljanje proizvoda, za lične sajtove koji trebaju ostaviti dojam na posjetioca i sl. Mislim da vam je sada jasnije gdje treba koristiti Flash.

Po svoj prilici ćete kod "sklapanja" sajta koristiti neki od vizuelnih alata. Iako danas uglavnom daju dosta čist kod, preporučljivo je pokrenuti naredbu Clean Up HTML Code (u Dreamweaver-u – ili neku evivalentnu) i nakon toga ručno prepraviti kod. Dosta često se dešava da željeni izgled ne možete postići bez brisanja nekog HTML taga, koji vaš program ne želi da obriše. Upravo zato svi vizuelni editori imaju u sebi editor koda, da bi vam ovaj zadatak olakšali.

19.4.2. Veličina dokumenata

HTML dokumenti se prikazuju u prozoru web browsera, koji se iscertava na desktopu i zavisi od njegove veličine. Svaki korisnik koristi drugačiju radnu rezoluciju, pa stoga ne možete pretpostaviti kako će stranica izgledati u njegovom browseru. Prihvaćeno pravilo je da se stranice prave za pretpostavljenu najmanju

korištenu rezoluciju, a ona iznosi 800x600. Praktično, širina tabele u kojoj je smješten sadržaj ne smije prelaziti 770 piksela. Ako hoćete da budete sigurni da se stranica ispravno prikazuje i na manjim rezolucijama, širina tabele treba biti prilagođena rezoluciji 640x480, odnosno ne smije prelaziti 600 piksela.

Drugi pristup je korištenje tabele sa relativnom širinom, pri čemu se sadržaj prilagođava veličini prozora browsera.

Oprezni trebate biti i kod izbora prave veličine fonta, jer ako izaberete premalen font, na većim rezolucijama može biti nečitak. Računajte i da će kompletna stranica izgledati manja na većim rezolucijama.

Zbog svega ovoga se ne može dati neko generalno pravilo, već samo treba prilagoditi stranicu korisnicima kojima je namijenjena. Zato uvijek isprobajte sve.

19.4.3. Podloge

HTML omogućava da se mali ponavljajući uzorci (pattern) mogu postavljati u podlogu stranice. To su obično mali GIF ili JPEG dokumenti (32x32 ili 64x64 piksela) koji se brzo učitavaju i mogu poboljšati izgled sajta. Zahvaljujući CSS-u, moguće je postići da se slika pojavi samo jednom ili da se ponavlja samo horizontalno ili vertikalno (duž x ili y - ose).

Budite pažljivi, jer prejak uzorak može tekst iznad njega učiniti nečitkim. Koristite svjetlija pisma na tamnijim podlogama i obrnuto. Je, ne zaboravite - čitljivost je najbitnija!

19.4.4. GIF ili JPEG?

Dva se grafička formata koriste na Internetu: JPEG i GIF. JPEG je format s gubitkom (lossy) - radi postizanja što manje veličine datoteke, žrtvuje se određena količina informacija (kvalitet slike). Veličina slike je obrnuto proporcionalna kvalitetu. GIF nije format s gubitkom (non-lossy), pa čuva sve informacije uz kompresiju. Zbog toga GIF najčešće ima puno lošiju kompresiju. Pored toga, ima ograničen broj boja koje može čuvati (maksimalno 256). Najbolje će poslužiti za slike s malo boja ili slike male veličine. Pod ovim podrazumijevamo razne logotipe, natpise i sl. GIF posjeduje još jednu korisnu osobinu: transparentnost. Proglasimo jednu boju za providnu i umjesto nje se u browseru prikazuje ono što se nalazi ispod slike.

JPEG je puno pogodniji za fotografije i veće slike s puno boja, kod kojih GIF ne može puno učiniti. Najuočljiviji artefakti kod JPEG formata su na finim tonskim prelazima iz jedne boje u drugu. Zato, zlatno pravilo kod dizajna - eksperimentišite i pogledajte prije finalnog objavljivanja!

19.4.5. Web tehnologije

Okviri (frames)

Najkontroverznije pitanje na koje možete naići prilikom planiranja (i kasnije izgradnje) sajta sigurno su okviri. Pomoću njih se u istom prozoru browsera mogu prikazati dvije ili više HTML datoteke, ali tako da zajedno čine jednu cjelinu.

Okviri se najčešće koriste za navigaciju, pri čemu se u neki od njih smješta HTML dokumenti koji sadrži navigaciju: Tehnički - radi se o jednom HTML dokumentu u kojem je opisan način na koji je podijeljen prozor browsera, te pokazivač na druge HTML dokumente koji se učitavaju u svaki od okvira.

Prednost okvira je što je navigacija cijelo vrijeme dostupna posjetiocu stranice (učitana je u jedan od okvira), a ako sadrži puno grafike, štedi se propusnost i dobija na brzini učitavanja. Negativne strane su što programi koji prijavljuju stranice na Internet pretraživače (nazivaju se botovi) mogu prijaviti samo neku dokument stranicu, pa je moguće da prilikom pretrage posjetilac dobije samo stranicu dokumenta, bez navigacije. I na kraju - pošto je HTML sa navigacijom cijelo vrijeme učitano, dosta je teško postići da linkovi označavaju sekciju sajta na kojoj se nalazite, pa treba izabrati neki drugi način.

Okvire treba koristiti samo kada je neophodno - kada dizajn zahtijeva da cijelo vrijeme imate učitano navigaciju. Korisni su kod sajtova koji nude puno sadržaja, jer se ubrzava učitavanje i olakšava kretanje po sekcijama i dokumentima.

Slojevi (layers)

Ovo nema nikakve sličnosti sa layerima iz Photoshop-a ili Fireworks-a. Riječ je o okviru koji možemo smjestiti bilo gdje na stranici i kome možemo dati različita svojstva, a u njega staviti bilo koji sadržaj. Ovo omogućava dizajnerima veliku slobodu, ali na žalost, zbog lošije podrške među starijim browserima, layere nije preporučljivo uvijek koristiti. Neki browseri uopšte ne prikazuju layere, neki ih nisu u stanju složiti kako je to zadano... Za neke stvari su neizbježni (padajući meniji, recimo), no nastojte da njihovu upotrebu svedete na minimum zbog problema sa kompatibilnosti. Vremenom će ovi problemi nestati, što će upotrebu layera učiniti češćom.

Server-side include (SSI)

Radi se o tome da se određeni HTML dokument pohrani na serveru jednostavnom "include" komandom uključuje unutar drugog HTML dokumenta. I ova tehnologija se najčešće koristi za navigaciju, ali se može koristiti i za prikaz dijelova sajta koji se često mijenjaju. Na ovaj način izbjegavate promjenu "glavnog" HTML sa servera, već mijenjate samo pomenuti dokument, što u pojedinim situacijama može olakšati administriranje i osvježavanje sajta. Za programere, riječ je o običnoj include naredbi poznatoj u svakom programskom jeziku. Unutar HTML datoteke ubaci se link

```
<!-- #include virtual="ime datoteke"-->
```

na datoteku čiji sadržaj želimo ubaciti. Kada neki korisnik zatraži tu stranicu od servera, server će prvo pretražiti stranicu za ovakvim naredbama, i ako ih naše, umjesto linka na željenu datoteku, ubaciti će njen kod. Za dosta stvari dosta praktično rješenje, koje je teže implementirati kada to želimo uraditi sa skriptama koje imaju svoje fragmente na više mjesta unutar HTML koda (npr. onMouseover efekat za slike koji zahtijeva onLoad event unutar BODY taga, a deklaraciju funkcije u HEAD tagu). Web server mora podržavati ovu tehnologiju.

19.5. Postavljanje na server – web hosting

Da bi neki sajt bio vidljiv i dostupan korisnicima Interneta, mora biti postavljen na neki web server. Web server je računar na kojem se "vrti" web server softver i koji ima stalnu konekciju na Internet. Praktično, na njemu se napravi poseban folder u koji se kopira sajt sa svojom strukturom.

Usluga iznajmljivanja prostora na web serveru naziva se web hosting, od engleske riječi host, što znači domaćin. Proces kopiranja sajta na server naziva se *upload* i obično se obavlja putem FTP (File Transfer Protocol) protokola. Za tu će vam namjenu trebati neki FTP klijent – poseban softver namijenjen samo ovom poslu.

U zavisnosti od namjene i važnosti sajta zavisit će i izbor prostora na serveru. Za lične i testne sajtove, studentske i druge projekte, sasvim će dobri biti besplatni web serveri na Internetu, poput Geocities, Tripod i sl. Radi se o kompanijama koje nude "besplatno" prostor. "Kvaka" je u tome što stranice koje postavite na ove servere moraju sadržavati reklamne banere, od kojih ove kompanije zarađuju.

Bitna napomena: postavljanje stranice firme, organizacije ili bilo koje druge institucije na besplatni server je neprofesionalno i šteti njenom ugledu! Bolje nemojte imati stranicu uopšte, nego je imati na besplatnoj adresi tipa www.geocities.com/mojafirma.

Za komercijalne namjene (predstavljanje firme, projekta, proizvoda i sl.), najčešći izbor je iznajmljivanje prostora kod firmi koje se bave ovim poslom (web hosting ili web presence provajderi) ili kupovina vlastitog servera. Druga varijanta je puno skuplja i zahtijeva stalnu konekciju, pa će se na nju odlučiti samo veće firme koje već imaju potrebnu infrastrukturu. Naravno, ovo je najbolja opcija, jer imate najviše kontrole nad svojim sajtom.

Većina firmi će iz finansijskih razloga izabrati neku firmu na čijim će serverima zakupiti prostor. Kada dođe do toga, treba znati nekoliko stvari.

Prva bitna stvar je izbor platforme. Ovo se odnosi samo na aktivne stranice i zavisi od tehnologija oje ste koristili u razvoju sajta, jer se statične stranice mogu objaviti na bilo kojem serveru. Ako ste sajt razvili u ASP-u, morat ćete tražiti web hosting provajdera koji nudi prostor na Microsoft Internet Information Server (IIS) serveru. Analogno tome, ako ste koristili PHP ili Perl, morat ćete tražiti server na kojem se

"vrti" neki Unix, najčešće Linux ili neki BSD. Ovo je ujedno najbitnije pitanje koje morate razlučiti.

Većina domaćih web hostinga provajdera nude različite hosting "pakete" - određenu količinu megabajta, određeni broj e-mail adresa, podršku za različite tehnologije i skripting jezik. Bitno je da provajder nudi neki način uvida u statistiku (najbolje preko web browsera) posjeta sajta, te da su podržane tehnologije koje ste koristili na sajtu. Cijene su različite, pa ćete prema veličini sajta procijeniti šta vam stvarno treba. Pojedini provajderi u paketu nude i registraciju domene.

Najbitnija stvar, koju nećete saznati iz cjenovnika, je podrška. Ako imate važan sajt na kojem objavljujete puno novosti i drugih informacija, ne smijete dozvoliti ni sat njegove neprisutnosti na Internetu. Zato je bitno da tehničko lice iz web presence provajdera uvijek bude dostupno, bez obzira na doba dana ili noći. Ovo je teško saznati, ali je najbolje raspitati se kod drugih koji već koriste usluge određene firme. Na kraju krajeva, ako već nekoga i izaberete, iz iskustva ćete znati da li je njihov odnos dovoljno poslovan, a provajdera uvijek možete promijeniti.

Za većinu domaćih sajtova zbor nije toliko kritičan. Većina provajdera će ispuniti očekivanja i pokriti potrebe, pa onda ostaje da sinhronizujete potrebe sa ponudom.

19.6. Izbor domene

Ako surfate po domaćem web prostoru, sigurno ste primijetili sajtove sa čudnim domenama. Izbjegavajte ovaj pristup! Domena je naziv po kojem će neki posjetilac naći vašu stranicu. Ukucava se u URL polje u browseru, pa je bitno da bude što kraće (smanjuje mogućnost greške) i da jasno opisuje namjenu sajta. Za firme, najbolje je da oslikava njen naziv, a za projekte, potražite najkraću moguću riječ koja i dalje asocira na dati projekat. Primjer dobrih imena domena su; trazi.com, bosnia.ba, nt.ba i sl.

Za razumijevanje domena ključno je shvatiti da se pod domenom najčešće podrazumijeva ime tipa "mojafirma.ba". Dok je "mojafirma" proizvoljno ime koje vi birate (ako nije zauzeto). Zadnji desni dio imena (iza tačke) predstavlja takozvanu Top Level Domenu. TLD domene su: .com, .net, .gov, .org, .infom, .museum i neke druge. Postoje još jedan tip Top Level Domen-a, a to su državne domene. Za BiH, nacionalni nastavak je .ba i treba ga što više koristiti, posebno za lokalne firme, projekte, predstavljanje lokalnih informacija i sl. .hr je naravno Hrvatska, .si Slovenija, .de je Njemačka itd.

Domena .com dolazi do "commercial" i označava sve komercijalne projekte. Ovaj nastavak treba koristiti samo ukoliko se sajtom obraćate globalnoj Internet populaciji - ako ste firma koja usluge/proizvode nudi na međunarodnom nivou. Ostali nastavci koriste se za određivanje vrste organizacije koja stoji iza sajta ili vrste sajta. Sajtovi državnih institucija će imati .gov nastavak (od eng. Government = vlada), .org će koristiti različite organizacije i sl.

Da bi ipak uz vrstu sajta vezali pripadnost državi, nastale su kombinacije. Domaće državne institucije trebale bi koristiti gov.ba domen, domaće organizacije org.ba domen i sl. iz kojih je jasna namjena sajta i pripadnost određenoj državi.

19.7. Prijavljivanje na pretraživače

Konačno ste završili web sajt u koji ste uložili dosta vremena i truda, postavili ga na neku Internet lokaciju, ali niko ga ne posjećuje. Niko uopšte ne zna da on postoji. Jeste li ga prijavili na web pretraživače?

Zamislite sljedeću situaciju: proteklih mjesec dana svu svoju energiju, vrijeme i novac uložili ste u pravljenje web sajta. On je konačno tu, završen i spreman za start. Postavili ste ga na web, temeljito testirali i otklonili sve "404-file not found" greške, sredili uvodnu stranicu koja izgleda poput crvenog saga prostrtog ispred posjetitelja koji samo što nisu stigli. Ali, nakon početnog oduševljenja dolazi razočarenje - niko ne posjećuje vaš sajt!? Odnosno, niko osim vaše rodbine i prijatelja. Šta ste krivo napravili? Ne sviđa im se dizajn? Ne. Sadržaj nije dovoljno zanimljiv? Ne. Naslovnica je prenapadna? Ne. Stranice se predugo učitavaju? Ne. Pa šta je onda problem?

Niko uopšte i ne zna za vaš sajt! Među doslovce milijardama stranica koje postoje na Internetu i hiljadama koje se svakodnevno prave, šanse da ljudi posjete vašu web lokaciju tek onako, bez nekog specifičnog razloga i povoda, tako su malene da je o njima smiješno i govoriti. Potrebno je, dakle raširiti informaciju o vašem novootvorenom web sajtu, o tome gdje se on nalazi i šta nudi. Potrebno je omogućiti surferima da saznaju za vas.

Prema podacima kompanija koje se bave Internet istraživanjima tržišta, 50% do 80% surfera za pronalaženje web sajtova koristi Internet pretraživače. Drugačije rečeno, čak do 80% posjetitelja koji, barem u početku, posjete vaš sajt pronašli su ga preko nekog od velikog broja Internet pretraživača. Takva situacija je najlogičnija, obzirom da je osnova svrha Altaviste, Googlea i drugih upravo to - indeksiranje ogromnog broja stranica i omogućavanje surferima da dođu do njih. Iz toga nije teško zaključiti da je vaš primarni zadatak, nakon postavljanja web sajta na Internet, njegovo prijavljivanje na što veći broj pretraživača.

19.7.1. Meta tagovi

Većina Internet pretraživača ima robota (ili bota), skriptu čiji je osnovni (i jedini) zadatak da "surfa" po web stranicama širom Interneta i prikuplja podatke o njima. Kasnije se ti podaci obrađuju i ubacuju u bazu podataka pretraživača, gdje postaju dostupni svima za pretragu. Bitna stvar za vas, kao autora tek postavljenog i nepoznatog web sajta, jesu upravo podaci koje roboti prikupljaju o web stranicama. Iako to varira od robota do robota, najčešći podaci su URL (adresa web sajta), veličina stranica, datum zadnje izmjene i sadržaj TITLE i META tagova.

Unutar **TITLE** taga, naravno, upisan je naslov stranice. Preporučljivo je da on sadrži od 5 do 20 riječi i da bude što je više moguće opisan. U **META** tagovima su smješteni podaci koje će, pored naslova, najviše uticati na rangiranje sajta na pretraživačima, što znači da trebate obratiti posebnu pažnju pri njihovom pisanju. Meta tag description sadrži kratak opis vašeg sajta i ne bi trebao biti duži od 200 znakova. Mnogi autori jednostavno prepisu naslov stranice u ovaj tag, što je dobra praksa ako ste se odlučili za nešto duži naslov koji sadrži riječi koje karakterišu vaš sajt. Drugi bitan meta tag je **KEYWORDS**, čije ime sve govori - tu je potrebno upisati ključne riječi koje najbolje pristaju sajtu, odnosno njegovom sadržaju. Kada neko upiše jednu od tih riječi u pretraživač, među stranicama koje odgovaraju će dobiti kao rezultat i vašu. U ovaj tag zato upišite sve riječi koje možete povezati sa sadržajem koji nudite, odvojite ih zarezima i trudite se da istu riječ ne ponavljate više od pet puta. Prostornog ograničenja nema, ali nemojte pretjerivati i pisati više od 500 ključnih riječi, iz dva razloga: prvo - robot može takvu stranicu prepoznati kao spam i jednostavno je zaobići, a drugo - bez obzira koliko velik vaš sajt bio, 500 ključnih riječi je više nego dovoljno. Pogledajte primjer zaglavlja:

```
<HEAD>

<TITLE> INFO Online - online izdanje informatickog
magazina
INFO</TITLE>

<META name="description" content="INFO Online je online
izdanje bh. Informatickog magazina INFO, sa mnogim
ekskluzivnim sadrzajima">

<META name="keywords" content="infor, info online,
informatika, racunarstvo, software, hardware, internet,
forum, diskusije, muzika na struju, najnovije
racunarske novosti, cheat kodovi, top lista igara">

</HEAD>
```

Kada je riječ o robotima, treba još voditi računa o sadržaju prvog odjeljka naslovne stranice, eventualnim greškama u HTML kodu i postojanju `<noframe>` sekcije kod naslovnica koje su podijeljene na frejmove (okvire). Mnogi roboti, pored gore navedenog naslova i meta tagova, u obzir uzimaju i prvi odjeljak (nekoliko stotina znakova) na stranicama, tako da bi odlično bilo u tom dijelu imati što informativniji tekst. Postojanje grešaka u HTML kodu može vam se odbiti od glavu, jer nisu rijetki roboti koji jednostavno preskoče stranicu na kojoj naiđu na greške. Ako koristite okvire, svakako ubacite `<noframe>` tag, unutar kojeg možete izbjeći tako često viđeni "this browser does not support frames" opis stranice. Razlog za to leži u činjenici da roboti u stvari nisu klasični browseri (poput IE-a ili Netscapea), pa ne podržavaju okvire.

19.7.2. Prijava na pretraživače

Roboti i automatsko dodavanje na pretraživače su odlična stvar, samo što su očajno spori - zbog ogromnog broja novih stranica nekada zna proći i godinu dana dok robot ne obiđe vašu stranicu i uvrsti je u bazu podataka. A vi ste upravo postavili vaš sajt i želite dovući posjetitelje odmah, a ne za nekih godinu dana. Na sreću, svaki od pretraživača i web direktorija omogućava ručno dodavanje sajtova u baze. Na nešto manju sreću (da ne kažem na nesreću), danas postoji puno pretraživača i teško je na svaki od njih ručno dodavali sajt.

Rješenje za to su "Submission services" - servisi za prijavu web stranica na pretraživače. Obzirom da su podaci koji se upisuju prilikom prijave na različite pretraživače dosta slični, ljudi su se dosjetili da bi bilo korisno (i profitabilno) napraviti skriptu koja bi jednom unesene podatke prosljedila većem broju pretraživača. Tako jednim udarcem ubijete nekoliko (desetina) muha, ali i ostanete lakši za određenu svotu novca. Danas postoji dosta ovakvih servisa, među kojima vam barem polovica (ako ne i većina) neće naplatiti ništa za te usluge. Naravno, komercijalni servisi su kvalitetniji i obuhvaćaju više pretraživača, a nisu pretjerano skupi. Zato ih preporučujem za promociju web sajtova koji bi trebali donositi svojim kreatorima profit. Za lične i nekomercijalne web sajtove će sasvim dovoljan biti neki besplatan servis za prijavu:

- GetSubmitted
<http://www.getsubmitted.com>

Ovaj sajt nudi besplatnu prijavu vašeg sajta na veliki broj pretraživača uz minimalan trud (samo upisivanje potrebnih podataka i klik na dugme) ali i više od toga. Da biste mogli koristiti ovu uslugu, potrebno se prethodno registrovati (što je besplatno). Nakon registracije dobijate pristup administratorskom dijelu unutar kojeg se, pored skripte za prijavu sajtova na pretraživače, nalazi i obimna dokumentacija koja će vam biti od pomoći pri promociji. Analizator meta tagova će pomoći svima koji se žele provjeriti ispravnost meta tagova na svom sajtu. Posjedujete li više web sajtova, GetSubmitted će vam se svidjeti, jer nudi mogućnost prijave više sajtova, kao i postavljanje remindera glede termina prijave (recimo, svake 3-4 sedmice želite ponovo prijaviti web sajt, jer nije ušao u izbor mnogih pretraživača).

- AddMe
<http://www.addme.com>

AddMe će zanimati gotovo svakog webmastera, jer pored servisa za prijavu na pretraživače nudi i niz drugih usluga. Recimo, ovdje možete pročitati tekstove koji se tiču promocije na webu, ali i vođenja poslovnih sajtova i uspješne prodaje proizvoda preko Interneta. Treba li vam besplatna statistika, ova se adresa čini dobrim izborom. Ako ne znate sami ispisati meta tagove koji će optimalno "predstaviti" vaše stranice robotu, kliknite na Meta Tags Generator link u glavnom

meniju. Za postavljanje web shopa na sajt ne treba vam mnogo novaca i truda - sve potrebno naći ćete ovdje!

- JimTools
<http://www.jimtools.com>

Poput prethodno opisanog sajta, JimTools će ponuditi više od puke prijave na pretraživače. Prije svega, možete pročitati tutorijal o kreiranju pravilnih meta tagova ili koristiti njihov alat za pravljenje meta tagova. Nakon toga, Jim nudi prijavu na 42 pretraživača, 144 web direktorija i 1160 FFA stranica. Istina, većina tih lokacija nisu tako popularne i neće vam možda donijeti niti jednu posjetu, ali obzirom na vaše vrijeme i trud koji iziskuje, alat je više nego dobar.

Aklo želite vidjeti koliko je vaš sajt popularan, odnosno koliko se drugih sajtova linkuje na njega, preporučujem Link Popularity Analyzer. Position Tools će vam reći kako sajt stoji na glavnim pretraživačima i web direktorijima. Sve u svemu, sajt vrijedan posjeta.

- Website Promotion Center
<http://www.submit-away.com>

Kako naslov kaže, ovaj sajt teži da bude centar za vašu promociju na webu. Nećete tamo naći skriptu koja će vam omogućiti prijavu na popularne pretraživače, mada bi se to možda od sajta sa ovakvim imenom očekivalo: Ali ćete zato naići na vrijedne informacije i linkove koji će vam biti od velike pomoći. Koje su prednosti i nedostaci manualne (ručne) i automatske prijave na pretraživače? Šta sve trebate raditi, a šta ne smijete raditi prilikom promovisanja sajta? Koje greške treba izbjegavati po svaku cijenu? Odgovore na ova pitanja možete pronaći na gore navedenoj adresi.

Posebno treba izdvojiti čak 600 mjesta, odvojenih u kategorije, na koje možete prijaviti sajt. Naravno, to ćete morati učiniti ručno, ali su sajtovi lijepo opisani, tako da možete izabrati one koji vam najviše odgovaraju.

Iako će vam spomenuti servisi omogućiti prijavu na Yahoo i druge velike web direktorije (recimo Open Directory), mnogo je preporučljivije u ovim slučajevima ručno se prijaviti. Razlog za to treba tražiti u činjenici da je nemoguće automatski optimizirati prijavu na Yahoo za sve sajtove (jer svaki sajt ima specifičnosti), a ispravno popunjen obrazac za prijavu na Yahoo će vam kasnije donijeti barem 30% od ukupnih posjeta sa svih pretraživača. Još jedna stvar je bitna, a zove se vrijeme. Pravilno ispunjena i optimizirana prijava na Yahoo će vas dovesti u ovaj direktorij za oko tri sedmice (ako niste znali, ovaj vjerovatno najveći web direktorij na Internetu je ručno administriran).

19.7.3. Trikovi za prijavljivanje na Yahoo

Iako danas na Webu postoji dosta direktorija koji sadrže veliki broj stranica podijeljenih u više ili manje kategorija, Yahoo je do danas ostao najvažniji među njima. Zbog toga, ali i zbog činjenice da njihov web direktorij edituju ljudi kojima je to posao (pregledaju prijavljene slajdove, ocjenjuju ih i ubacuju u kategorije), pravilna prijava na Yahoo jedna je od najbitnijih stvari prilikom promocije web sajta.

Prije svega, morate imati strpljenja. Mora proći minimalno dvije sedmice da bi ljudi iz Yahoo uopšte pročitali vašu prijavu i eventualno uvrstili vaš sajt. Nekada se taj period zna odlučiti i na više od dva mjeseca. Važi pravilo - što tačnije i bolje popunite prijavu, brže ćete dospjeti u Yahoo indeks.

Dalje, nemojte ni razmišljati o prijavljivanju na Yahoo dok vaš sajt i sve stranice koje on sadrži ne bude u potpunosti završen. Završite sve "under construction" stranice, provjerite gramatičke i HTML greške, sredite dizajn što bolje možete i provjerite vrijeme učitavanja stranice. Sve ove stvari se broje, jer prijavljene sajtove pregledaju ljudi od krvi i mesa, a ne računarski programi. Oni odlučuju da li će sajt biti uvršten u kategoriju za koju je aplicirao ili ne.

Treća stvar na koju treba obratiti pažnju je ponavljanje prijave. Nemojte biti dosadni i prijavljivati sajt nanovo i nanovo svake dvije sedmice, jer će ga osoblje u Yahoo vjerovatno namjerno ignorisati. Budite uporni, ali kulturni. Ako vaš sajt ne bude uvršten nakon mjesec dana čekanja, ponovite prijavu.

Ispunjavanje prijave

Na početku izvršite pretragu na Yahoo Indeksu upisujući naziv vašeg sajta, da biste provjerili da li se on već nalazi u nekoj kategoriji. Ako bude pronađen, znači da ste već ubačeni, ali pretpostavljam da ćete željeti promijeniti prikazane informacije. U tom slučaju otidite na:

<http://add.yahoo.com/fast/change>

i ispunite formu. Uputstva koja će biti iznijeta dalje u tekstu vrijede i za ovu formu.

Ako niste pronašli vaš web sajt, pronađite kategoriju koja mu najbolje odgovara. Na dnu stranice bi se trebao nalaziti "Suggest a site" link na koji trebate kliknuti i ispuniti formu. Ukoliko linka nema, to znači da ste odabrali suviše generalnu kategoriju koja je zatvorena za dodavanje novih sajtova. Pronađite specifičniju kategoriju i kliknite na navedeni link. Prilikom prijave imate mogućnost koristiti Yahoo Express koji košta 299 USD, ali vam zato garantuje da će sajt biti pregledan u roku od sedam dana. Ono što ne garantuje jeste da će biti uvršten u indeks!?

Prije ispunjavanja forme dobro pročitajte uputstva iz Yahoo-a (neka od njih su već spomenuta u ovom tekstu). Slijedite ih doslovce, jer svaka greška vas može koštati. Nemojte da tekst unesen u polja predviđena za naslov i opis sajta izgleda poput promotivnog materijala! Budite što kraći i što opisniji. Preporučljivo je ispuniti

polje za komentar, u kome trebate upisati sve što bi moglo pomoći osoblju koje pregleda sajtove, a tako indirektno pomoći i vama da što prije budete primljeni.

Još jedna važna stvar je da Yahoo sortira sajtove u kategorijama po abecednom redu. Obzirom da kategorije znaju biti prilično pune, sajt koji počinje sa brojem ili nekim od prvih slova abecede će imati više klikova od nekog koji počinje sa recimo slovom Z. Ako već ne možete mijenjati naziv sajta, preporučujem da izaberete kategoriju sa što manje sajtova u njoj.

Naziv, opis i URL su informacije po kojima Yahoo pretražuje sajtove u bazi, što ih čini najvažnijima. Obratite posebnu pažnju na njih, posebno na opis. Pokušajte da on bude fino složena rečenica, sa što više ključnih riječi koje odgovaraju sajtu. Lista ključnih riječi odvojenih zarezom neće proći na Yahoo-u.

19.7.4. Domaći pretraživači

Sve iznijete informacije vrijede za strane web pretraživače, dok ćete se na domaće morati ručno prijavljivati. Na sreću (ili nesreću, prosudite sami), njih nema mnogo, tako da vam prijava novog sajta na sve važnije domaće pretraživače neće oduzeti više od sat vremena. Što je prilično isplativa investicija, posebno ako pravite sajt čiji je sadržaj namijenjen ili bi mogao biti interesantan posjetiteljima iz BiH. Obzirom da se slični jezici koriste u istočnom i zapadnom susjedstvu, radi postizanja veće popularnosti preporučujem prijavljivanje i na najpopularnije pretraživače u susjednim zemljama (Hrvatska, Srbija, Crna Gora).

Neki domaći pretraživači:

- <http://www.pogodak.ba>
- <http://www.trazi.com/>
- <http://dir.bosnia.ba/>
- <http://www.bosnia-online.com/>
- <http://www.bosnia.com>

19.8. Održavanje sajta

Za osnovno održavanje sajta koje bi uključivalo neke male izmjene na tekstu, pravljenje nekih novih stranica na osnovu šablona (template), mijenjanje cijena i slično, nije potrebno neko veliko znanje, tako da je ovaj vid održavanja, na Zapadu nazvan i self-maintenance, popularan, jer je pogodan za male i srednje firme. Još su dva velika razloga popularnosti ovog vida održavanja: novac i kontrola. Održavanje sajta je posao koji rijetko ko voli i zbog toga se od strane profesionalaca masno naplaćuje, što za male firme nije povoljno. S druge strane,

većina ljudi voli da ima kontrolu nad svojim drugim "ja" na Webu i želi da pravi izmjene kad hoće i koliko hoće, a da ne mora svaki čas zvati druge u pomoć.

Stranice je potrebno redovno održavati. Ružno je da posjetioci dođu na stranicu i tamo zateknu riječi: "zadnje osvježavanje: 23.maj 1997". Redovnost brige o stranici pokazuje da li se njen vlasnik brine o svojim posjetiocima i da li se trudi da im uvijek dostavi najsvježiji mogući sadržaj. Zato - naglasak na redovnost i to na dnevnoj bazi!

Važno je već u procesu planiranja stranice prilagoditi je kasnijem održavanju. Stoga web dizajneri, obratite pažnju! Da li je stranica koju ste napravili laka za izmjene? Da li je lako na nju dodavati tekst i slike? Nemojte da sami sebi u početku zagorčate život time što ćete poslije gubiti sate na održavanju. Ali isto tako, ukoliko nećete biti zaduženi za održavanje (što je najvjerovatnije), zapitajte se hoće li se taj koji će naslijediti stranicu od vas moći snaći u vašem kodu? Potrudite se da ostavite sve čisto iza sebe - to je najlakši put da zaradite poštovanje kolega.

Jedan od osnovnih zadataka je povratna informacija (feedback). Nakon što postavite sajt online, dobijat ćete masu e-mailova, na koje neizostavno morate odgovarati, što spada u aktivnosti održavanja. Ovo je posebno važno za vlasnike komercijalnih sajtova. Jedno istraživanje u SAD pokazalo je da prosječni korisnik, nakon što pošalje e-mail sa pitanjem, očekuje bilo kakav odgovor u roku od 1 sat! Stoga se trudite da odgovorite što brže možete. Ukoliko ne možete stići da odgovorite na sve, bilo bi dobro da napravite jedan šablonski tekst, koji ćete slati u obliku automatskog odgovora na svaki e-mail, a u kome ćete objasniti da je mail primljen id a će posjetitelj konkretan odgovor dobiti čim prije. Na taj način, on neće steći utisak da je zapostavljen i da govori u prazno.

Izuzetno važno je i da nemate tzv. mrtvih linkova, posebno na druge sajtove. Zato se oni redovno provjeravaju i održavaju da budu što je moguće više "up-todate". Za ovaj posao postoji nekoliko dobrih shareware alata, a i noviji editori imaju svoje alate koji provjeravaju i prijavljuju mrtve linkove. Držanje svježih i korisnih linkova na stranici može učiniti čuda, dovodeći ponovno posjetioce i povećavajući promet i prodornost sajta.

Koliko god da pažljivo kucate tekst, uvijek će se potkrasti barem nekoliko greškica (Murphy-jev zakon). U početnim stadijima održavanja zato je korisno pažljivo provjeravati tekst, tragajući za tip-felerima.

Kao i u svakom biznisu, ono što se od poslovnih partnera traži i cijeni je pouzdanost. I u održavanju sajtova, pouzdanost je važna i tražena vrlina. Kako? Na primjer, ako ste obećali da ćete staviti novi sadržaj na sajt svake sedmice, potrudite se da to i obavite! NIKAD, nikad ne obećavajte ono što ne možete ispuniti! To ruši vaš ugled.

Posjetiocu koji se ponovo, nakon nekog vremena, vraća na sajt može biti teško snaći se i razlučiti šta je novo i šta se promijenilo od prošle posjete. Pomozite mu na taj način što ćete jasno označiti nove stvari. Isto tako, razgraničite i sadržaj koji

je statičan, odnosno onaj koji se rijetko mijenja i onaj koji je dinamičniji i regularno se osvježava.

Ponekad se pojavi potreba da se promijeni adresa ili domena stranice. Ukoliko selite stranicu, potrudite se da ostavite redirekciju i link na staroj adresi. Nemojte dozvoljavati sebi da u suprotnom gubite posjetioce.

Kako proces polako napreduje, primijetit ćete kako je sajt polako evoluirao. Uvijek postoje mjesta za popravke i poboljšanja, pa ga zato redovno dotjerujete i dopunjavajte. Razmišljajte o informacijama koje stavljate na sajt, kao i o tome koliko često biste trebali raditi osvježavanje. Vijesti bi trebale biti što svježije, dok će se potreba da se izmjeni sadržaj npr. historije firme rijetko javiti.

Dok budete radili na održavanju, ideje o poboljšanjima će se često same javljati. Odolite iskušenju da velike izmjene pravite u malim koracima. Samo ćete sebe izludjeti, a proces održavanja će se pretvoriti u noćnu moru. Naročito ako imate dvije verzije sajta za dvije rezolucije. E, onda ste se uvalili u probleme. Zato velike izmjene pravite odjednom, dobro ih istestirajte, pa tek onda "dignite" na server.

Budite konzistentni u svom dizajnu! Prilikom pravljenja sajta, njegovom izgledu se u procesu dizajna daje jedan uniforman izgled koji je zajednički za sve stranice na sajtu. Prilikom izmjena, imperativ je da se tog izgleda držite, osim u situaciji kada vršite redizajn kompletnog sajta.

S vremena na vrijeme se HTML standardu dodaju nove komande. Za neke je privlačnost tehničkog izazova prevelika za odoljeti. Ali, prije nego što pokušate da upotrijebite nešto novo, zapitajte se barem da li je to stvarno potrebno.

20. Literatura

- [1] Dario Sušan, Dragan Petric, Velika knjiga o World Wide Webu, ZNAK, Zagreb, 1996, ISBN: 953-189-015-3
- [2] Peter Aitken, Phil Syme., Sams Teach Yourself Visual Basic .NET Web Programming in 21 Days, ISBN: 978-0-672-32236-5; 2001;
- [3] Khristine Annwn Page, Macromedia Dreamweaver MX: Training from the Source Macromedia Press; 2002, ISBN: 0201799294
- [4] H. M. Deitel, P. J. Deitel, A. B. Goldberg: Internet & World Wide Web How to Program (3rd Edition), ISBN: 0131450913
- [5] C.D. Knuckles: Introduction to Interactive Programming on the Internet: Using HTML and JavaScript, ISBN: 047138366X
- [6] S. Bishop: Macromedia Dreamweaver 8 Revealed, ISBN: 1418843083
- [7] H.E. Williams, D. Lane: Web Database Applications with PHP & MySQL, ISBN: 0596005431
- [8] Emir Jusić, BH Webmasters, <http://www.most-art.de/phptutorial/index.php>
- [9] Microsoft Windows Vista: Sidebar gadgets, <http://www.microsoft.com/windowsvista/features/foreveryone/sidebar.mspx>
- [10] Brian Teutsch, Microsoft Sidebar for Windows Vista Beta 2, Gadget Development Overview, Microsoft Windows White Paper, 2006, <http://microsoftgadgets.com/Sidebar/DevelopmentOverview.aspx>
- [11] članci iz magazina INFO, brojevi 26, 32, 33, 47, 49, 51, 87, 90-91, 99



financed by

Austrian
 Development Cooperation