SADRŽAJ

Predgovor xi

Deo I Početak

Poglavlje 1
Odakle da počnem? 3

Odakle da počnem? 4
Šta radi veb dizajner? 4
Koje jezike moram da naučim? 11
Šta treba da kupim? 14
Šta ste naučili 19
Proverite svoje znanje 20

Poglavlje 2
Kako funkcioniše veb 21

Internet naspram veba 21
Serviranje informacija 21
Par reči o čitačima veba 23
Adrese veb strana (URL) 24
Anatomija veb strane 26
Sklapanje u celinu 30
Proverite svoje znanje 32

Poglavlje 3
Opšti koncepti koje treba da poznajete 33

Vrtoglavo mnoštvo uređaja 34
Pridržavajte se standarda 36
Progresivno poboljšavanje 36
Responsivan veb dizajn 38

Jedan veb za sve (Dostupnost) ... 41
Potreba za brzinom (Performanse veb lokacije) ... 43
Proverite svoje znanje ... 45

Deo II HTML: struktura veb strane

Poglavlje 4
Izrada jednostavne veb strane ... 49

Veb strana, korak po korak ... 49
Prvo treba pokrenuti editor teksta ... 50
Korak 1: počnite sa sadržajem ... 53
Korak 2: strukturirajte dokument ... 55
Korak 3: obeležite tekstualne elemente ... 58
Korak 4: dodajte sliku ... 61
Korak 5: izmenite izgled strane pomoću stilova ... 64
Kada se dobre veb strane pokvare ... 65
Proverite ispravnost svojih dokumenata ... 66
Proverite svoje znanje ... 67
Pregled elemenata: struktura dokumenta ... 68

Poglavlje 5
Označavanje teksta ... 69

Pasusi ... 70
Naslovi ... 70
Liste ... 73
Još elemenata za sadržaj ... 76
Organizovanje sadržaja strane ... 79
Umetnuti elementi ... 84
Generički elementi (div i span) ... 95
Neki specijalni znakovi ... 99
Sklapanje u celinu ... 100
Proverite svoje znanje ... 102
Pregled elemenata: tekst ... 104

Poglavlje 6
Dodavanje veza ... 105

Atribut href ... 106
Veze ka stranama na vebu ... 107
Veze ka stranama na istoj veb lokaciji ... 108
Usmeravanje ka novom prozoru čitača ... 118
Veze za e-poštu ... 119
Telefonski brojevi kao veze ... 120

Proverite svoje znanje 121
Pregled elemenata: veze 122

Poglavlje 7
Dodavanje slika 123
Par reči o formatima slika 123
Element img 124
Prozor u prozoru 130
Proverite svoje znanje 131
Pregled elemenata: slike 132

Poglavlje 8
Označavanje tabela 133
Kako se koriste tabele 133
Minimalna struktura tabele 135
Zaglavlja tabele 138
Opseg prostiranja ćelija 139
Pristupačnost tabele 142
Šta se saznali o tabelama 144
Proverite svoje znanje 146
Pregled elemenata: tabele 146

Poglavlje 9
Obrasci 147
Kako rade obrasci 147
Element form 149
Promenljive i sadržaj 151
Kontrole na obrascu 152
Dostupnost obrazaca 171
Raspored elemenata i dizajn obrasca 173
Proverite svoje znanje 175
Pregled elemenata: obrasci 176

Poglavlje 10
Šta je suština verzije HTML5 181
Nešto čudno se dogodilo na putu za XHTML 2 182
Označavanje 185
Upoznajte API-je 189
Video i audio 192
Oblast za crtanje (canvas) 198
Završna reč 202
Proverite svoje znanje 203

Deo III CSS: prezentacija veb strane

Poglavlje 11
Osnovni koncepti CSS stilova 207

Prednosti stilova 207
Kako funkcionišu stilovi 209
Veliki koncepti 214
Usavršite rad sa stilovima 221
Proverite svoje znanje 223

Poglavlje 12
Formatiranje teksta 225

Svojstva fontova 225
Promena boje teksta 243
Dodatni tipovi selektora 244
Podešavanje reda teksta 249
Podvlačenje i ostali „ukrasi" 252
Prelazak s velikih slova na mala i obrnuto 252
Da se razmaknemo 253
Senke za tekst 254
Menjanje bulita i brojeva za liste 259
Proverite svoje znanje 261
Pregled CSS-a: svojstva fontova i teksta 263

Poglavlje 13
Boje i pozadine 265

Zadavanje vrednosti boja 265
Boja prednjeg plana 272
Boja pozadine 273
Igranje s providnošću 275
Predstavljamo još jednu vrstu selektora: pseudoklase 276
Još selektora: pseudoelementi 279
Selektori atributa 281
Pozadinske slike 284
Objedinjeno svojstvo background 293
Kao duga (prelivi) 296
Konačno, spoljni opisi stilova 300
Proverite svoje znanje 303
Pregled CSS-a: svojstva za boje i pozadine 304

Poglavlje 14
Razmišljajte uokvireno 305
Okvir elementa 305
Zadavanje dimenzija okvira 306
Prostor oko sadržaja 312
Ivice 316
Margine 328
Dodeljivanje uloga elementima pomoću svojstva display 333
Dodavanje senki okvirima 335
Proverite svoje znanje 336
Pregled CSS-a: osnovna svojstva okvira elementa 338

Poglavlje 15
Plutanje i pozicioniranje 341
Normalan tok 341
Plutanje 342
Osnove pozicioniranja 356
Relativno pozicioniranje 358
Apsolutno pozicioniranje 359
Fiksno pozicioniranje 368
Proverite svoje znanje 370
Pregled CSS-a: svojstva za plutanje i pozicioniranje 371

Poglavlje 16
CSS i raspoređivanje elemenata na strani 373
Strategije za izradu rasporeda 373
Tehnike raspoređivanja elemenata na strani 380
Izrada višestubačnih rasporeda pomoću plutajućih objekata 381
Pozicionirani raspored 392
Pozadine stubaca koji se protežu od vrha do dna strane 395
Proverite svoje znanje 398

Poglavlje 17
Prelazi, transformacije i animacije 399
CSS prelazi 399
CSS transformacije 410
Animacija pomoću ključnih kadrova 420
Proverite svoje znanje 423
Pregled CSS-a: transformacije i animacije 426

Poglavlje 18
Tehnike rada sa CSS stilovima 427
Čisto polazno stanje (resetovanje CSS stilova) 427
Tehnike zamene teksta slikom 429
CSS sprajtovi 430
Stilizovanje obrazaca 434
Stilizovanje tabela 441
Osnove responsivnog veb dizajna 444
Sažetak o opisima stilova 454
Proverite svoje znanje 454
Pregled CSS-a: svojstva za tabele 456

Deo IV JavaScript: ponašanje veb strana

Poglavlje 19
Uvod u JavaScript 459
Šta je JavaScript? 459
Dodavanje JavaScript koda veb strani 463
Anatomija skripta 463
Objekat window čitača veba 478
Događaji 478
Sklapanje u celinu 481
Proverite svoje znanje 483

Poglavlje 20
Korišćenje JavaScripta 485
Upoznajte DOM 485
Polifili 493
JavaScript biblioteke 497
Sažetak 501
Proverite svoje znanje 502

Deo V Izrada veb grafike

Poglavlje 21
Osnove veb grafike 507
Izvori slika 507
Upoznajte formate 510
Veličina i rezolucija slike 522
Rad s providnim slikama 526
Uvod u format SVG 533
Sažetak o slikama za veb 538
Proverite svoje znanje 539

Poglavlje 22
Izrada optimalnih slika za veb 541
Opšte strategije optimizacije 542
Optimizacija GIF slika 543
Optimizacija JPEG slika 547
Optimizacija PNG slika 552
Optimizacija do željene veličine datoteke 553
Pregled optimizacije 554
Proverite svoje znanje 555

Dodatak A
Odgovori 557

Dodatak B
Selektori u jeziku CSS3 583

Indeks 587

PREDGOVOR

Pred vama je prevod četvrtog izdanja knjige *Naučite veb dizajn*.

Toliko toga se desilo od prethodnog izdanja! Baš kada je izgledalo da se situacija smiruje nakon što su proizvođači čitača veba i zajednica programera prihvatili standarde za veb, pojavio se „mobilni veb“ da ponovo sve poremeti. S pojavom pametnih telefona i tabličnih računara (tableta), veb pronalazi put do malih ekrana i konteksta korisnika u pokretu, gde ga ranije nije bilo. To je postavilo ozbiljne izazove pred nas – veb dizajnere i programere – dok pokušavamo da korišćenje naših veb lokacija učinimo što prijatnijim bez obzira na to kako će im se pristupati.

Dok ovo pišem, mnogi od tih izazova – na primer, kako svakom uređaju poslati odgovarajuću sliku – još se razmatraju. Ovo je neverovatno dinamično doba za veb dizajn, prepuno eksperimentisanja i sadarnje. Donekle me podseća na doba „divljeg zapada“ veba, negde iz 1993. godine kada sam započela karijeru veb dizajnera. Toliko toga treba otkriti! Toliko mogućnosti! Iskreno, i sada je nezgodno smestiti sve te tehnologije i tehnike za takvo mnoštvo najrazličitijih uređaja u jednu knjigu. Imajući to u vidu, potrudila sam se da ukažem na teme koje su u središtu promena i na mrežne resurse koji će vam obezbediti ažurne informacije.

Tu su i dva nova strandarda – HTML5 (peta velika revizija jezika za označavanje hiperteksta) i CSS3 (Cascading Style Sheets, Level 3) – koji su nam sada dostupni, a bili su samo glasine kada sam pisala prethodno izdanje. Deo knjige u kome se obrađuje HTML, sada se odnosi na tekući standard, HTML5. Obuhvatila sam i one delove standarda za CSS3 koji su usvojeni, uključujući i novo poglavlje o dodavanju pokreta i interaktivnosti pomoću prelaza i transformacija. Alatke koje su nam na raspolaganju omogućavaju nam da radimo mnogo efikasnije i uradimo mnogo više nego pre samo nekoliko godina.

Na kraju, pošto je JavaScript postao toliko značajan deo programiranja za veb, ovo izdanje sadrži dva poglavlja sa uvodom u sintaksu JavaScripta i nekoliko primera njegove upotrebe. Nisam stručnjak za JavaScript, ali sam imala sreće da nađem nekog ko jeste. Poglavlja o JavaScriptu napisao je Met „Wilto“ Markvis, dizajner i programer u organizaciji Filament Group, član tima jQuery Mobile, i tehnički urednik veb lokacije A List Apart.

PRATEĆA LOKACIJA

Obavezno posetite prateću veb lokaciju ove knjige, na adresi learningwebdesign.com. Tu ćete naći materijal za vežbe, članke koje možete preuzeti, spisak hiperveza iz knjige, izmene i druge korisne stvari.

Kao i u prva tri izdanja, pažnja je posvećena konkretnim problemima i potrebama početnika svih vrsta – od klasičnih grafičkih dizajnera, preko programera koji žele da budu kreativniji i kancelarijskih službenika, do studenata, ljudi koje prave lične lokacije, i svih drugih koji žele da nauče kako da prave veb strane. Potrudila sam se da u knjigu unesem iskustvo koje sam stekla na svojim kursevima veb dizajna za početnike, i unela sam brojne vežbe i testove preko kojih ćete moći da se oprobate u praktičnom radu i proverite kako napredujete.

Bez obzira na to da li ovu knjigu čitate samostalno ili je koristite uz neki kurs veb dizajna, nadam se da će vam biti dobra odskočna daska i da će vam uz nju veb dizajn biti zabavan.

Kako je organizovana ova knjiga

Četvrto izdanje knjige *Naučite veb dizajn* podeljeno je na pet delova, od kojih se svaki bavi po jednim važnim aspektom izrade veb strana.

Deo I • Početak

U prvom delu postavljeni su temelji za sve što sledi u knjizi. Počinjem s nekim važnim opštim informacijama o okruženju za veb dizajn, opisujući – između ostalog – različite uloge koje možete imati, tehnologije koje možete savladati i alatke koje su vam na raspolaganju. Odmah ćete se sresti s HTML-om i CSS-om i naučiti opšte stvari o tome kako funkcionišu veb i veb strane. Uz to, predstavljam i neke važne koncepte zbog kojih ćete početi da razmišljate onako kako savremeni veb dizajneri razmišljaju o svom poslu.

Deo II • HTML: struktura veb strane

U poglavljima iz drugog dela knjige razmotreni su detalji svakog elementa i atributa, kako bi se opisala njihova semantička struktura, a obuhvaćeni su i novi elementi iz verzije HTML5. Obrađeno je označavanje teksta, veza, slika, tabela i obrazaca. Deo II se završava temeljnim razmatranjem jezika HTML5 i njegovim razlikama u odnosu na prethodne standarde.

Deo III • CSS: prezentacija veb strane

U trećem delu knjige proći ćete put od upoznavanja sa osnovama korišćenja CSS stilova za promenu načina predstavljanja teksta, do izrade višestubačnih rasporeda, pa čak i dodavanja vremenski programiranih animacija i interaktivnosti veb strani. Opisane su i uobičajene tehnike rada u CSS-u, uključujući i to kako da napravite veb stranu primenjujući responsivan veb dizajn.

Deo IV • JavaScript: ponašanje veb strane

Met Markvis započinje četvrti deo knjige pregledom sintakse JavaScripta, kako biste naučili da razlikujete promenljivu od funkcije. Upoznaćete i neke načine korišćenja JavaScripta, uključujući i DOM skriptovanje, i postojeće JavaScript alatke – kao što su polifili i biblioteke – zahvaljujući kojima možete brzo iskoristiti JavaScript, čak i ako niste još spremni da pišete sopstveni JavaScript kôd od nule.

Konvencije korišćene u knjizi

U knjizi su korišćene sledeće tipografske konvencije:

Kurziv

Za URL adrese, adrese e-pošte, imena datoteka i direktorijuma, naglašavanje, i stručne termine na srpskom i engleskom.

`Font konstantne širine`

Za primere koda i komande koji se unose sa tastature.

`Podebljan font konstantne širine`

Za naglašavanje u primerima koda.

`Kurziv konstantne širine`

Za tekst koji treba zameniti konkretnim vrednostima svojstava i stilova.

Deo V • Izrada veb grafike
U petom delu opisani su različiti formati datoteka pogodni za veb i kako ih optimizovati da bi datoteke sa slikama bile što manje.

Zahvalnica

Želim da se zahvalim svom uredniku, Sajmonu sen Lorenu, s kojim sam sarađivala na više projekata, a radujem se i novim. Hvala i koautoru, Metu Markvisu (*matmarquis.com*), što je učinio JavaScript zabavnim i što je zadržao dobro raspoloženje radeći s nekim ko želi sve da kontroliše.

Pri radu na ovoj knjizi, podržavali su me mnogi pametni i divni ljudi. Zahvaljujem se glavnim recenzentima, Aronu Gustafsonu (*easy-designs.net*), Džoelu Maršu (*thehipperelement.com*) i Metu Menceru što su odvojili toliko vremena od svojih poslova kako bi obezbedili da i najsitnije informacije u poglavljima budu potpuno tačne. Zahvalnost za hirurški precizne revizije zaslužuju i sledeći ljudi: Antoni Kalcadila, Deni Čepman, Met Hohi, Džerald Luis, Džejson Pamental i Stefani Riger.

Veoma sam srećna što poznajem toliko mnogo vodećih stručnjaka u ovoj oblasi, čije su knjige, članci, prezentacije, kompleti slajdova i lični kontakti bili pogonsko gorivo za moj rad. Ne bih uspela bez pomoći ovih genija (po abecednom redu njihovih imena na engleskom): Den Sederholm, Džoš Klark, Endi Klark, Kris Kojer, Bred Frost, Liza Gardner, Džejson Grigsbi, Stiven Hej, Skot Jel, Skot Dženson, Tim Kadlek, Džeremi Kit, Sanders Klajnfeld, Piter-Pol Koh, Brus Loson, Itan Markot, Erik Mejer, Karen Mekgrejn, Šeli Pauers, Brajan Riger, Stefani Riger, Remi Šarp, Luk Vroblevski i Džefri Zeldman.

Radu na knjizi doprinose mnogi ljudi, pa moju zahvalnost zaslužuju i Melani Jarbro (urednik izdanja i korektor), Ženevjev d'Entremon (lektor), Rebeka Demarest (izrada slika), Njudžen (prelom strana), Elen Trutmen Cajg (indeks), Rendi Komer (dizajn korica) i Ron Bilodo (dizajn unutrašnjosti knjige).

Na kraju, zahvaljujem se Edi Fridman (najboljem šefu svih vremena) jer je bila strpljiva kada sam upala u probleme. Hvala i mojim najdražima, Džefu i Arlou; srećna sam što napokon mogu da kažem: „Vratila sam se kući".

O autoru

Dženifer Niderst Robins (Jennifer Niederst Robbins) počela je da se bavi veb dizajnom 1993. godine, kada je napravila prvu komercijalnu veb lokaciju, Global Network Navigator. Pored ove knjige, napisala je i *Web Design in a Nutshell* i *HTML5 Pocket Reference* (koja postoji i kao aplikacija za iOS), a obe je objavio O'Reilly. Ranije je izlagala na mnogim konferencijama, uključujući Seybold i South By Southwest, a predavala je i uvod u veb dizajn na Univerzitetu Johnson and Wales u Providensu, SAD. Trenutno radi kao dizajner digitalnih proizvoda za kompaniju O'Reilly Media, gde se bavi arhitekturom informacija i dizajnom interakcija, i stara se da korišćenje veb lokacija, aplikacija i e-knjiga bude prijatno. U slobodno vreme, Dženifer uživa u kreativnim aktivnostima, indi roku, kuvanju i materinstvu.

Korišćenje primera koda

Namena ove knjige je da vam pomogne da obavite posao. U opštem slučaju, kôd iz knjige možete koristiti u svojim programima i dokumentaciji. Ne morate tražiti dozvolu od nas, osim ako reprodukujete značajan deo koda. Na primer, za pisanje programa u kome se koristi nekoliko segmenata koda iz ove knjige, ne treba vam dozvola. Za prodavanje ili distribuciju CD-ROM diskova s primerima iz knjiga izdavačke kuće O'Reilly, dozvola je neophodna. Dozvola vam ne treba kada odgovarate na pitanja tako što citirate ovu knjigu i navodite primer koda iz nje. S druge strane, obavezno nabavite dozvolu ukoliko u dokumentaciji svog proizvoda citirate značajan deo primera koda iz ove knjige.

Ne zahtevamo, ali svakako cenimo navođenje izvora. To obično uključuje sledeće podatke: autor, izdavač i ISBN. Na primer: *Learning Web Design*, Fourth Edition, Jennifer Niederst Robbins. Copyright 2012 Littlechair, Inc., 978-1-449-31927-4

Ako smatrate da se upotreba primera koda u vašem slučaju ne uklapa u gore navedene uslove fer korišćenja ili dozvole, slobodno nam pišite na adresu *permissions@oreilly.com*.

Komentari i pitanja

Molimo vas da svoje komentare i pitanja o ovoj knjizi uputite izdavaču originalnog izdanja:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472, USA
bookquestions@oreilly.com

ili izdavaču prevedenog izdanja:
redakcija@mikroknjiga.rs

Ova knjiga ima i svoju veb lokaciju na kojoj se objavljuju (na engleskom) ispravke uočenih grešaka, primeri i dodatne informacije. Adresa je:
http://oreil.ly/learn_web_design_4e

Više informacija o knjigama, konferencijama, centrima s resursima i mreži O'Reilly Network, naći ćete na veb lokaciji kompanije O'Reilly:
http://www.oreilly.com

Više informacija o izdanjima Mikro knjige potražite na adresi:
http://www.mikroknjiga.rs

Kolofon

Izgled ove knjige je rezultat komentara čitalaca, naših eksperimenata i povratnih informacija od distributera. Jedinstvene korice ove knjige prate naš jedinstven pristup tehničkim temama – trudimo se da udahnemo život potencijalno suvoparnim temama. Tekst je pisan u fontu Linotype Birka, a font u naslovima je Adobe Myriad Pro.

POČETAK

DEO I

U OVOM DELU

Poglavlje 1
Odakle da počnem?

Poglavlje 2
Kako funkcioniše veb

Poglavlje 3
Opšti koncepti koje treba da poznajete

ODAKLE DA POČNEM?

Veb postoji već više od dvadeset godina i do sada je prošao kroz euforičan rani razvoj, zastoj izazvan ekonomskim problemima i ponovni uspon pokrenut inovacijama, za sve to vreme konstantno evoluirajući. Jedna stvar je izvesna: kao komunikacioni i poslovni medij, veb će trajno ostati na sceni. I ne samo to, već je pronašao put i ka uređajima kao što su pametni telefoni, tableti (tablični računari), TV i mnogi drugi. Nikada nije bilo više prilika i mogućnosti da se primeni znanje o veb dizajnu.

Držeći kurseve i radionice za veb dizajn, imala sam priliku da se sretnem s ljudima iz različitih oblasti, zainteresovanim da nauče kako se prave veb strane. Evo šta su rekli neki od njih:

> *„Već sedamnaest godina radim dizajn za klasičnu štampu, a sada osećam pritisak da moram da ponudim i usluge veb dizajna."*
>
> *„Sekretarica sam u maloj firmi. Šef je tražio da napravim malu veb lokaciju na kojoj će svi zaposleni moći da razmenjuju informacije."*
>
> *„Godinama se bavim programiranjem i sada bih hteo da se okušam i u dizajnu. Mislim da mi veb pruža dobru priliku za ovladavanje novim veštinama."*
>
> *„Umetnik sam i želim da saznam kako mogu da postavim uzorke svojih slika i skulptura na internet."*
>
> *„Igrao sam se malo s veb stranama u srednjoj školi i čini mi se da bih voleo da se time bavim profesionalno."*

Bez obzira na razlog, prvo pitanje koje ljudi postave uvek je isto: „Odakle da počnem?" Možda će vam se učiniti da ima previše toga što treba naučiti i da nije lako znati odakle početi. Ali, odnekud mora da se krene.

U ovom poglavlju opisaću krivu učenja – navešću odgovore na tipična pitanja koja mi ljudi postavljaju kada odluče da se upuste u veb dizajn. Napraviću uvod u discipline, tehnologije i alatke koje se koriste u veb dizajnu.

U OVOM POGLAVLJU

Odakle da počnem?

Šta radi veb dizajner?

Koje jezike moram da naučim?

Koje programe i opremu treba da nabavim?

Odakle da počnem?

Vaša konkretna polazna tačka zavisiće od vaše profesije i ciljeva. U svakom slučaju, dobar prvi korak je da naučite osnovne stvari o funkcionisanju veba i veb strana. Ova knjiga će vam dati te osnove. Kada naučite osnove, na samom vebu ili u knjižarama biće puno materijala za dalje učenje, zavisno od oblasti koje vas zanimaju.

Postoji i mnogo nivoa bavljenja veb dizajnom – od izrade jednostavne lokacije za lične potrebe do izgradnje ozbiljne karijere u ovom poslu. Možda će vam se dopasti da postanete veb programer, ili da se specijalizujete samo za jedan aspekt ovog posla. Mogućnosti su velike.

Ako rešite da se veb dizajnom bavite isključivo iz hobija, ili zato što želite da objavite par svojih projekata, možda ćete uvideti da treba uložiti samo malo truda (npr. pročitati ovu knjigu), iskoristiti postojeće šablone i, eventualno, kupiti neku vizuelnu alatku za veb dizajn (recimo, Adobe Dreamweaver). Možete se i upisati na neki početni kurs veb dizajna.

Ukoliko planirate da se profesionalno bavite veb dizajnom ili produkcijom, moraćete da steknete više znanja. Poslodavci možda ne traže univerzitetsku diplomu iz veb dizajna, ali očekuju da vide funkcionalne veb lokacije koje ste već napravili i koje će im pokazati koliko znate. Te lokacije mogu da budu rezultat vežbi s kursa iz veb dizajna, lični projekti, ili jednostavna lokacija pravljena za malu firmu ili organizaciju. Važno je samo da izgledaju profesionalno i da su zasnovane na dobro napisanom HTML kodu, CSS stilovima i – eventualno – skriptovima. Odličan način da naučite više jeste da nađete posao na nekom početnom nivou i postanete član tima koji pravi velike prezentacije. To će vam pomoći da lakše odlučite kojim aspektom veb dizajna kasnije želite da se bavite.

Ja hoću samo blog!

Da biste počeli da objavljujete svoje tekstove i slike na vebu, ne morate biti veb dizajner. Svoj blog, tj. ličnu dnevničku lokaciju, možete da pokrenete pomoću nekog od besplatnih ili jeftinih hosting servisa za blogove. Ti servisi nude standardne šablone, pa ne morate da učite HTML (mada vam to sigurno neće škoditi). Nekoliko najpopularnijih su:

- WordPress (*www.wordpress.com*)
- Blogger (*www.blogger.com*)
- Tumblr (*www.tumblr.com*)

Još jedan hosting servis koji omogućava izradu veb lokacija pomoću miša (povlačenjem i puštanjem) i koji nije namenjen samo za blogove, jeste Sqarespace (*www.squarespace.com*).

Šta radi veb dizajner?

Tokom godina, izraz „veb dizajn“ postao je opšti pojam za proces koji obuhvata nekoliko disciplina – od dizajna od kojeg zavisi kakvo će iskustvo imati korisnik veb lokacije, preko označavanja dokumenata, do ozbiljnog programiranja. U ovom odeljku opisane su neke od najčešćih uloga veb dizajnera.

Ako samostalno pravite malu veb lokaciju, moraćete da promenite nekoliko uloga. Srećom, to nećete ni primetiti. Napravićemo analogiju s kućnim poslovima, gde se od vas zahteva da budete i kuvar i čistač i računovođa i diplomata i baštovan i građevinski radnik – iako su za vas sve vreme to samo obični kućni poslovi. Isto vam je i kao samostalnom veb dizajneru. Bićete malo grafički dizajner, malo pisac, malo HTML programer, malo novinar, malo urednik; a sve vreme ćete imati utisak da vi samo „pravite veb strane“. Nema razloga za brigu.

UKRATKO

„Veb dizajn“ je zapravo kombinacija nekoliko disciplina, među kojima su:

- Vizuelni (grafički) dizajn
- Dizajn korisničkog interfejsa i ukupnog doživljaja
- Izrada veb dokumenata i lista stilova
- Pisanje skriptova i programiranje
- Strategija obezbeđivanja sadržaja
- Multimedija

Postoje i specijalisti koje možete da pozovete ako nešto ne umete sami da uradite. Na primer, ja pravim veb lokacije od 1993. godine i još uvek angažujem programere i projektante za multimediju kada moji klijenti traže interaktivne funkcije. To mi omogućava da se koncentrišem na one delove posla koje najbolje radim (u mom slučaju, to su organizacija sadržaja, interfejs i vizuelni dizajn).

Ako ne želite da postanete samostalan veb dizajner i ekspert za sve oblasti, opredelite se za određenu oblast i postanite deo većeg tima ili radite po ugovoru.

Obimne veb lokacije gotovo uvek prave timovi ljudi, od njih par do nekoliko stotina. Tada se svaki član tima bavi jednim aspektom procesa izgradnje lokacije. Ako je to vaš slučaj, onda možete svoje postojeće veštine (pisanje, Photoshop, programiranje) i interesovanja samo prilagoditi novom mediju.

Brojne uloge i odgovornosti koje se obično podrazumevaju kada se kaže „veb dizajn", podelila sam u četiri veoma široke kategorije: dizajn, razvoj (projektovanje), sadržaj i multimedija.

Dizajn

Ah, dizajn! Zvuči prilično jasno, ali je čak i taj jednostavan zahtev podeljen na niz specijalizacija kada je reč o izradi veb lokacija. Sledi opis nekoliko novih poslova koji se odnose na izradu veb lokacije, ali imajte na umu da se discipline često preklapaju i da je osoba koja sebe naziva dizajnerom često odgovorna za više od jednog navedenog posla (ako ne i za sve).

Iskustvo koje se pruža korisniku, interaktivnost i korisnički interfejs

Kada pomislimo na dizajn, često mislimo na to kako nešto izgleda. Međutim, na vebu, prvo se treba pobrinuti za to kako lokacija *radi*. Pre nego što izaberemo boje i fontove, važno je da prepoznamo ciljeve lokacije, kako će se ona koristiti i kako će se posetioci kretati po njoj. Ti zadaci pripadaju disciplinama dizajna interaktivnosti (engl. *Interaction Design, IxD*), dizajna korisničkog interfejsa (engl. *User Interface, UI*) i dizajna korisnikovog celokupnog doživljaja, tj. iskustva u radu s lokacijom (engl. *User Experience, UX*). Između tih aspekata ima dosta preklapanja, i nije neuobičajeno da je za sve njih zadužena jedna osoba ili jedan tim.

Cilj dizajnera interaktivnosti je da korišćenje lokacije bude što lakše, efikasnije i prijatnije. U bliskoj vezi s dizajnom interaktivnosti jeste dizajn korisničkog interfejsa, koji je obično uže fokusiran na funkcionalno organizovanje strane i na određene alatke (dugmad, hiperveze, menije itd.) koje posetioci koriste za kretanje po sadržaju lokacije ili obavljanje zadataka.

Skorije zvanje u svetu veb dizajna jeste dizajner korisnikovog doživljaja. Tog dizajnera karakteriše holistički pristup – on obezbeđuje da celokupno iskustvo korišćenja veb lokacije bude zadovoljavajuće. UX dizajn se zasniva na dobrom razumevanju korisnika i njihovih potreba na osnovu posmatranja i intervjuisanja. Prema Donaldu Normanu (koji je skovao ovaj termin) dizajn korisnikovog doživljaja obuhvata „sve aspekte korisnikove interakcije s datim

proizvodom: kako se on doživljava, savladava i koristi". U slučaju veb lokacije ili aplikacije, to uključuje izgled, korisnički interfejs, kvalitet i poruku sadržaja, pa čak i ukupne performanse lokacije. Doživljaj mora da bude u skladu s poslovnim i opštim ciljevima date organizacije da bi lokacija bila uspešna.

U dokumente koje bi mogli da proizvedu IxD, UI ili UX dizajneri, spadaju i:

Izveštaji o proučavanju korisnika i testiranju

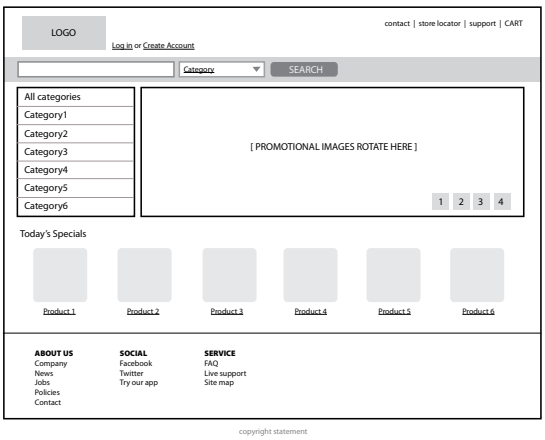
Razumevanje potreba, želja i ograničenja korisnika ključno je za uspeh dizajna veb lokacije ili aplikacije. Taj pristup koncentrisanja dizajna oko korisnikovih potreba, naziva se User Centered Design (UCD), i dominantan je u savremenom dizajnu. Izrada veb lokacije često počinje proučavanjem korisnika, koje uključuje intervjue i posmatranje, da bi se bolje razumelo kako lokacija može da rešava probleme ili kako će se koristiti. Uobičajeno je da dizajneri obave ciklus korisničkog testiranja u svakoj fazi procesa izrade lokacije kako bi osigurali upotrebljivost svojih rešenja. Ako se korisnici muče da bi otkrili određeni sadržaj ili kako da pređu na sledeći korak u nekom postupku, vreme je da se dizajneri vrate za crtaću tablu.

Žičani dijagrami

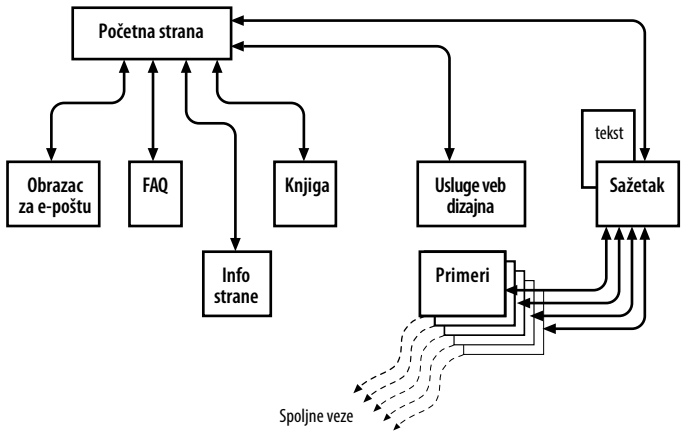
Žičani dijagram (engl. *wireframe diagram*) prikazuje strukturu veb strane koristeći samo obrise za svaki tip sadržaja (slika 1-1). Svrha žičanog dijagrama je da pokaže kako je podeljen prostor na ekranu i ukaže na to gde se nalaze funkcionalnosti i sadržaji kao što su navigacija, polja za pretraživanje, elementi obrasca itd., bez ikakvih ukrasa ili grafičkog dizajna. Obično sadrže i uputstva o tome kako sve treba da funkcioniše da bi razvojni tim znao šta da pravi.

Dijagram lokacije

Dijagram lokacije (engl. *site diagram*) prikazuje strukturu veb lokacije kao celine i međusoban odnos pojedinačnih strana. Na slici 1-2 prikazan je dijagram veoma jednostavne lokacije. Dijagrami nekih lokacija zauzimaju čitave zidove!



Slika 1-1. *Žičani dijagram veb lokacije.*



Slika 1-2. *Dijagram jednostavne veb lokacije.*

HTML: STRUKTURA VEB STRANE

DEO II

U OVOM DELU

Poglavlje 4
Izrada jednostavne veb strane (Uvod u HTML)

Poglavlje 5
Označavanje teksta

Poglavlje 6
Dodavanje veza

Poglavlje 7
Dodavanje slika

Poglavlje 8
Označavanje tabela

Poglavlje 9
Obrasci

Poglavlje 10
Šta je suština verzije HTML5

IZRADA JEDNOSTAVNE VEB STRANE
(Uvod u HTML)

U prvom delu knjige upoznali ste se sa osnovama veb dizajna. Pošto smo razmotrili najvažnije koncepte, vreme je da zasučemo rukave i bacimo se na izradu jedne prave veb strane. Napravićemo krajnje jednostavnu stranu, ali po istim principima po kojima se prave i one najsloženije.

U ovom poglavlju ići ćemo korak po korak da biste stekli osećaj za označavanje dokumenta HTML oznakama. U tom cilju koristićete priložene vežbe.

Evo šta želim da postignete u ovom poglavlju:

- Steknete osećaj za to kako se koristi označavanje u HTML-u i razumete elemente i atribute.
- Vidite kako čitači veba čitaju HTML dokumente.
- Naučite osnovnu strukturu HTML dokumenta.
- Steknete prvi uvid u to kako funkcionišu CSS stilovi.

Neka vas ne brine to što na ovom mestu nećemo učiti konkretne tekstualne elemente ili pravila u vezi sa stilovima; o njima će više reči biti u narednim poglavljima. Za sada je važno koncentrisati se na proceduru, opštu strukturu dokumenta i novu terminologiju.

U OVOM POGLAVLJU

Uvod u elemente i atribute

Postupno uputstvo za označavanje jednostavne veb strane

Elementi koji čine strukturu dokumenta

Jednostavni stilovi

Rešavanje problema s veb stranama

Veb strana, korak po korak

U poglavlju 2, *Kako funkcioniše veb*, videli ste kako izgleda HTML dokument. Sada ćete sami napraviti jedan dokument i doznati kako se on ponaša u čitaču veba. Primer u ovom poglavlju sastoji se od pet osnovnih koraka izrade veb strane.

Korak 1: Počnite sa sadržajem. Prvo ćete napisati sirov tekstualni sadržaj da biste videli šta čitači veba rade s njim.

Korak 2: Strukturirajte dokument. Naučićete sintaksu HTML elemenata i elemente koji dokumentu daju strukturu.

HTML težim putem

Ja se držim svoje klasične, staromodne metode učenja HTML-a – sama pišem kôd. Nema boljeg načina da istinski shvatite kako funkcioniše označavanje od ručnog pisanja dokumenta, oznaku po oznaku, i njegovog otvaranja u čitaču veba. Ne treba mnogo vremena da se stekne osećaj za ispravno označavanje dokumenata.

Mada možete odlučiti da koristite neki od gotovih programa za veb dizajn, lakše ćete ih i efikasnije koristiti ukoliko naučite HTML i razumete kako funkcioniše. Osim toga, biće vam drago što možete da pogledate izvornu datoteku i znate šta gledate. I još nešto – HTML je dobro znati ako tražite rešenje problema, ukoliko neka veb strana ne radi dobro, ili ako hoćete da izmenite podrazumevano formatiranje koje je generisao program za izradu veb strana.

Profesionalni veb dizajneri i programeri svakako teže ka tome da ručno označavaju sadržaj zato što tako imaju bolju kontrolu nad kodom i mogu sami da odlučuju koji elementi će se koristiti.

Korak 3: Obeležite elemente teksta. Sadržaj ćete opisati odgovarajućim tekstualnim elementima, a zatim ćete naučiti kako se koristi HTML.

Korak 4: Dodajte sliku. Dodaćete sliku na stranu i uz to naučiti kako se koriste atributi i prazni elementi.

Korak 5: Izmenite izgled strane pomoću stilova. U ovoj vežbi videćete kako se pomoću CSS-a formatira sadržaj.

Na kraju vežbe imaćete napisan izvorni dokument za veb stranu prikazanu na slici 4-1. Nije nešto, ali je početak.

Tokom vežbe, često ćemo proveravati naš rad u čitaču – verovatno više nego u stvarnom životu. Pošto je ovo uvod u HTML, korisno je videti kako svaka, pa i najmanja promena utiče na izvornu datoteku.

Prvo treba pokrenuti editor teksta

U ovom poglavlju, a i u celoj knjizi, HTML dokumente ćete pisati ručno, pa zato prvo treba da pokrenete editor teksta. Onaj koji ste dobili sa operativnim sistemom, na primer Notepad (Windows) ili TextEdit (Macintosh), biće sasvim dovoljan. I drugi editori teksta su u redu ako mogu da čuvaju čisto tekstualne datoteke sa oznakom tipa *.html*. WYSIWYG programi za izradu veb strana, kao što je Dreamweaver, za sada vam neće trebati. Namera mi je da vas ovde naviknem na ručno označavanje teksta (videti izdvojen odeljak „HTML težim putem“).

U nastavku teksta videćete kako se u Notepadu i TextEditu otvaraju novi dokumenti. Čak i ako ste ove programe već koristili, vredi se na brzinu podsetiti kako se primenjuju pojedini parametri, da biste vežbe koje slede uradili što bolje. Počećemo s Notepadom; korisnici Maca mogu da preskoče sledeći odeljak.



Slika 4-1. *U ovom poglavlju ćemo napisati izvorni dokument za ovu veb stranu, korak po korak.*

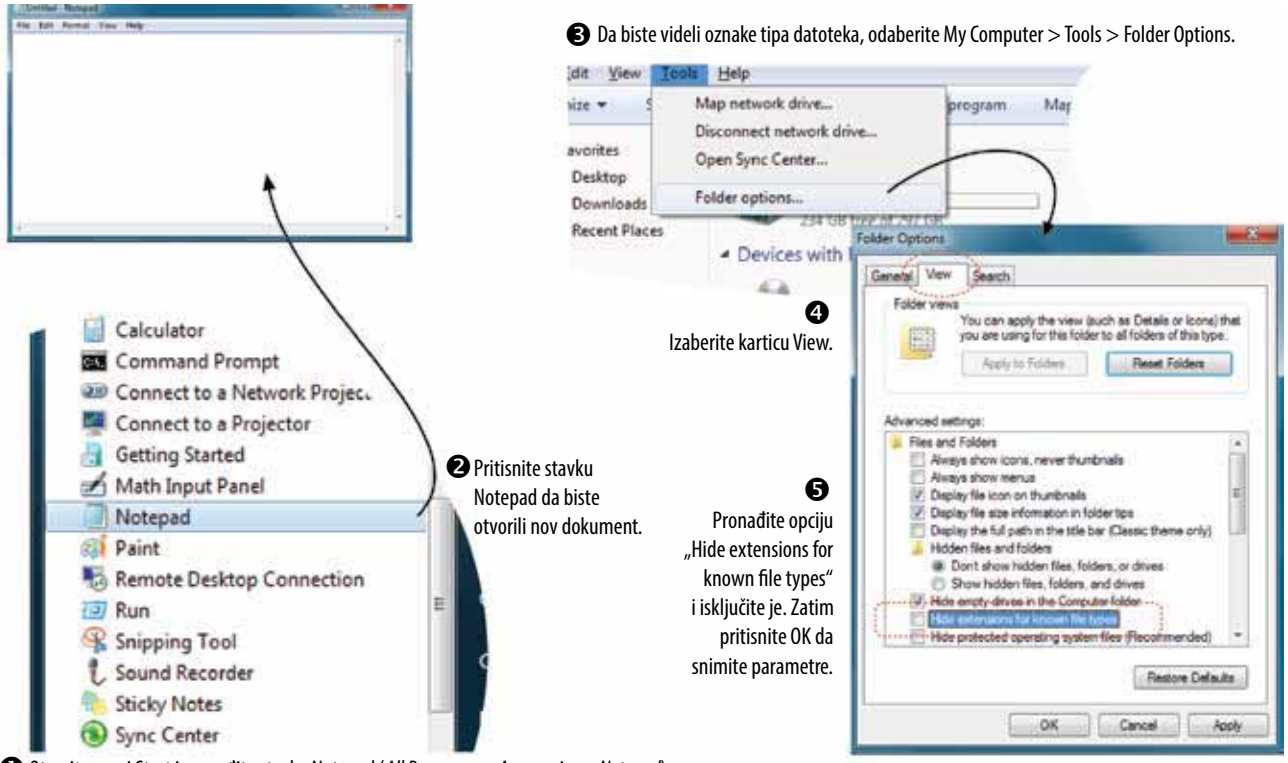
Izrada novog dokumenta u Notepadu (Windows)

Ovo su koraci za izradu novog dokumenta u Notepadu za Windows 7 (slika 4-2):

1. Otvorite meni Start i spustite se do Notepada (u podmeniju Accessories) ❶.
2. Pritisnite Notepad; otvoriće se prozor novog dokumenta. Možete početi da pišete ❷.
3. Sledeći korak je omogućiti da se vide oznake tipa datoteka. Ovaj korak nije neophodan prilikom izrade HTML dokumenata, ali je koristan da bi se tipovi datoteka jasnije i brže uočili. Izaberite „Folder Options...“ iz menija Tools ❸, a zatim karticu „View“ ❹. Pronađite opciju „Hide extensions for known file types“ i isključite je ❺. Pritisnite OK da biste sačuvali ovo podešavanje. Nadalje će se oznake tipa datoteka videti.

NAPOMENA

U Windowsu 7 pritisnite taster Alt da biste otvorili meni Tools i pristupili stavci Folder Options. U Windowsu 8.1, ova opcija se zove „Change folder and search options“ i nalazi se pod stavkom Options na kartici View.



Slika 4-2. *Izrada novog dokumenta u Notepadu.*

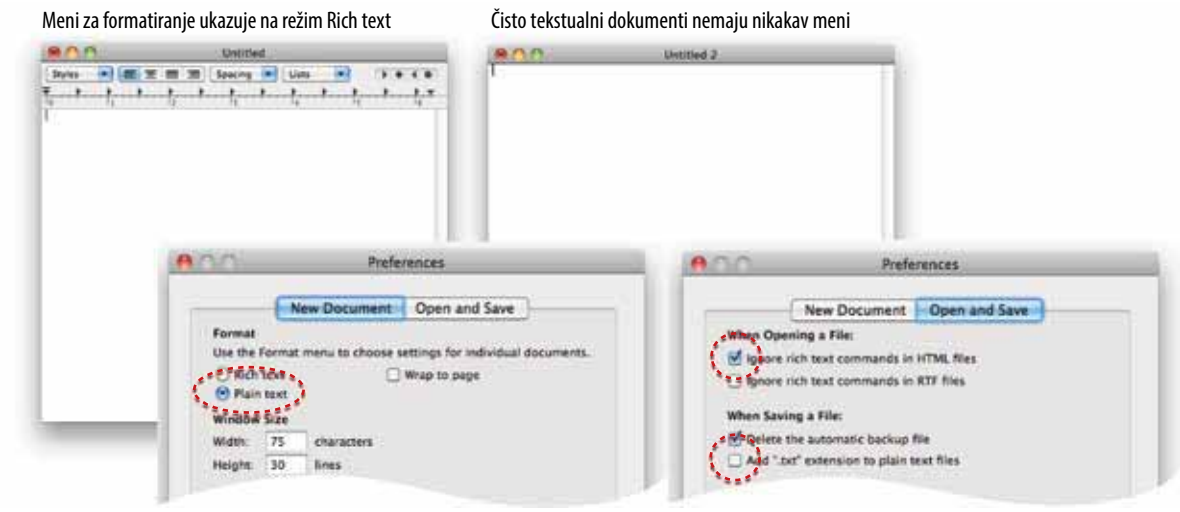
Izrada novog dokumenta u TextEditu (Mac OS X)

TextEdit automatski pravi dokumente u formatu „rich text“, sa skrivenim naredbama za formatiranje pomoću kojih se dobija podebljan tekst, podešava veličina fonta itd. TextEdit je u režimu „rich text“ kada se na vrhu prozora nalazi paleta za formatiranje (u režimu običnog teksta, ta paleta nije prikazana). HTML dokumenti moraju biti čisto tekstualni, pa treba izmeniti opciju Format, kao u primeru (slika 4-3).

1. Preko Findera pronađite TextEdit u direktorijumu *Applications*. Zatim pritisnite dvaput ime ili ikonicu TextEdit da biste pokrenuli istoimeni program.
2. TextEdit otvara nov dokument. Po paleti za formatiranje videćete da se nalazite u režimu Rich Text. Evo kako da ga promenite.
3. Otvorite okvir za dijalog Preferences u meniju TextEdit.
4. Treba podesiti tri parametra:

   Na kartici „New Document“ izaberite „Plain text“.

   Na kartici „Open and Save“ izaberite „Ignore rich text commands in HTML files“ i isključite opciju „ Add ‘.txt’ extensions to plain text files“.
5. Kada završite, pritisnite crveno dugme u gornjem levom uglu.
6. Nakon što napravite nov dokument, paleta za formatiranje se neće više prikazivati, pa možete da snimite svoj tekst kao HTML dokument. Svaki dokument uvek možete vratiti u Rich Text format tako što ćete izabrati Format → Make Rich Text kada TextEdit ne koristite za HTML.



Slika 4-3. *Kako da pokrenete TextEdit i u meniju Preferences izaberete format Plain Text.*

Korak 1: počnite sa sadržajem

Pošto smo otvorili novi dokument, vreme je da započnemo pisanje. Veb strana uvek počinje sadržajem, pa ćemo i mi tu početi. Vežba 4-1 služi da naučite kako se unosi sirov tekstualni sadržaj i snima dokument u nov direktorijum.

Vežba 4-1 | **Unesite sadržaj**

1. Pokrenite editor teksta i unesite dole navedeni sadržaj početne strane u novi dokument. Kopirajte ga onako kako ga ovde vidite – ne menjajte prelom redova. Sirovi tekst za ovu vežbu dostupan je na adresi *www.learningwebdesign.com/4e/materials*.

```
Black Goose Bistro

The Restaurant
The Black Goose Bistro offers casual lunch and dinner fare in
a hip atmosphere. The menu changes regularly to highlight the
freshest ingredients.

Catering
You have fun... we'll handle the cooking. Black Goose Catering
can handle events from snacks for bridge club to elegant corporate
fundraisers.

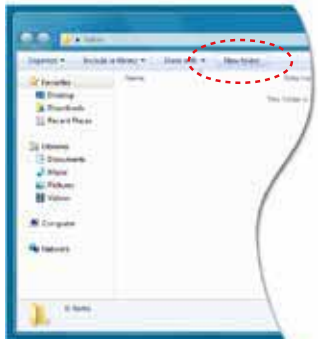
Location and Hours
Seekonk, Massachusetts;
Monday through Thursday 11am to 9pm, Friday and Saturday, 11am to
midnight
```

2. U meniju File izaberite „Save" ili „Save as" da biste otvorili okvir za dijalog Save As (slika 4-4). Prvo treba da napravite nov direktorijum u kojem će biti sve datoteke za vašu veb lokaciju (drugim rečima, treba da napravite lokalni osnovni direktorijum).

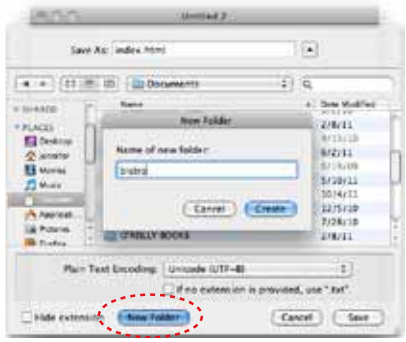
   Windows: Pritisnite ikonicu na vrhu da biste napravili nov direktorijum.

   Mac: Pritisnite dugme „New Folder".

Windows 7



Mac OS X



Slika 4-4. *Snimite datoteku index.html u nov direktorijum pod nazivom „bistro".*

Pravila za davanje imena

Prilikom davanja imena datotekama važno je poštovati sledeća pravila:

Koristite pravilne nastavke za datoteke. HTML i XHTML datoteke moraju da se završavaju nastavkom *.html*. Slike za veb moraju biti obeležene u skladu s formatom njihove datoteke: *.gif*, *.png* ili *.jpg* (*.jpeg* je takođe prihvatljiv nastavak).

Nikada ne koristite razmake u imenima datoteka. Za razdvajanje reči u imenima najčešće se koriste znak za podvlačenje ili crtica, na primer, *robins_bio.html* ili *robins-bio.html*.

Izbegavajte specijalne znakove kao što su **?**, **%**, **#**, **/**, **:**, **;**, • itd. Pri imenovanju datoteka koristite samo slova, brojeve, znak za podvlačenje, crtice i tačke.

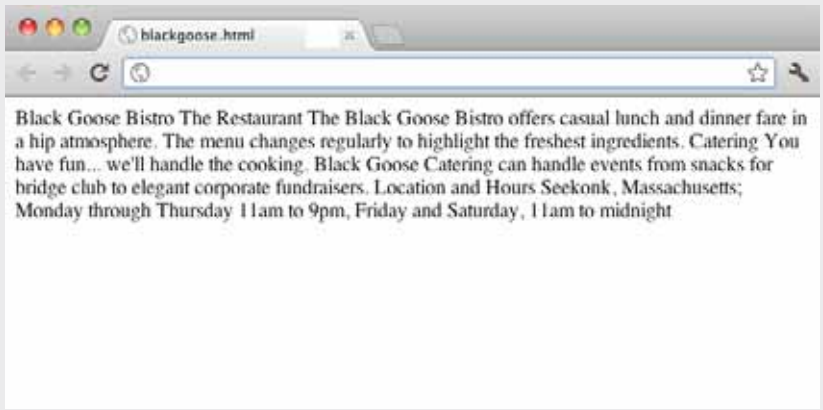
Neki serveri prave razliku između malih i velikih slova u imenima datoteka, neki ne. Zato je uvek bolje koristiti samo mala slova u imenima datoteka, jer je njima lakše upravljati.

Neka imena budu kratka. Kada su imena kratka, manja je i datoteka. Ako baš morate da zadate dugačka imena od više reči, razdvajajte reči crticama, na primer *veoma-dug-naslov-dokumenta.html*, radi čitljivosti.

Uvedite sopstvena pavila. Za velike veb lokacije korisno je razviti sopstvena pravila za davanje imena datotekama. Na primer, jedno pravilo može biti da uvek koristite mala slova i da reči razdvajate crticama. Kasnije nećete morati da nagađate kako se beše zove datoteka koju hoćete da povežete s drugim datotekama.

Dajte novom direktorijumu ime ***bistro*** i snimite tekstualnu datoteku ***index.html*** u taj direktorijum. Korisnici Windowsa će morati da izaberu opciju All Files posle Save as type kako bi sprečili da Notepad doda nastavak .txt imenu datoteke. Ime datoteke mora da se završava sa ***.html*** da bi čitač veba mogao da je prepozna kao veb dokument. Dodatne savete o imenovanju datoteka, videti u izdvojenom odeljku „Pravila za davanje imena".

Da vidimo sada kako datoteka ***index.html*** izgleda u čitaču. Pokrenite svoj omiljeni čitač (ja koristim Firefox) i u meniju File izaberite Open ili Open File. Odaberite datoteku ***index.html*** da biste je otvorili u čitaču. Videćete nešto kao na slici 4-5. O rezultatima će više reči biti u narednom odeljku.



Slika 4-5. *Sadržaj početne strane u čitaču veba.*

Šta čitači veba ignorišu

Pojedine informacije iz izvornog dokumenta, čitač će ignorisati. U te informacije spadaju:

Više belina (razmaka).

Kada čitač naiđe na više uzastopnih razmaka, on prikazuje samo jedan razmak. Znači, ako u dokumentu piše

```
u stara,    dobra    vremena
```

čitač prikazuje

u stara, dobra vremena

Prekid reda (znak za početak reda).

Čitači pretvaraju znakove za prelazak na početak reda (engl. *carriage returm, CR*) u beline, pa prema prethodno navedenom pravilu o ignorisanju više razmaka, prekidi redova ne utiču na formatiranje strane. Tekst i elementi se nastavljaju sve dok čitač ne naiđe na nov blok-element u dokumentu, kao što su naslov (**h1**), pasus (**p**) ili znak za prekid reda (**br**).

Tabulator.

Tabulatori se takođe prevode u znakove razmaka – znači, beskorisni su.

Nepoznate oznake.

Čitač ignoriše oznake koje ne prepoznaje ili koje su nepravilno napisane. Zavisno od elementa i čitača, to može dovesti do različitih rezultata. Čitač može da ne prikaže ništa, ili može da prikaže sadržaj oznake kao da je normalan tekst.

Tekst u komentarima.

Čitač neće prikazati tekst naveden između specijalnih oznaka **<!--** i **-->** koje se koriste za označavanje komentara. Videti izdvojen odeljak *Dodavanje skrivenih komentara*, kasnije u ovom poglavlju.

Čemu nas uči korak 1

Sadržaj koji smo uneli ne izgleda baš najbolje (slika 4-5). Tekst je spojen – a nije bio takav u originalnom dokumentu. Iz ovoga treba izvući par pouka. Prva je da čitač veba ignoriše prekide reda iz izvornog dokumenta. (U izdvojenom odeljku „Šta čitači veba ignorišu" dat je spisak ostalih informacija iz izvornog dokumenta koje se ne prikazuju u prozoru čitača.)

Druga pouka je da nije dovoljno samo ispisati sadržaj i nazvati dokument *.html*. Čitač može da prikazuje tekstualni sadržaj datoteke, ali taj sadržaj još uvek nema *strukturu*. Tu na scenu stupa HTML. Pomoću jezika za označavanje dodaćemo strukturu: prvo samom HTML dokumentu (korak 2), a zatim i sadržaju strane (korak 3). Kada zna strukturu dokumenta, čitač veba može bolje da prikaže stranu.

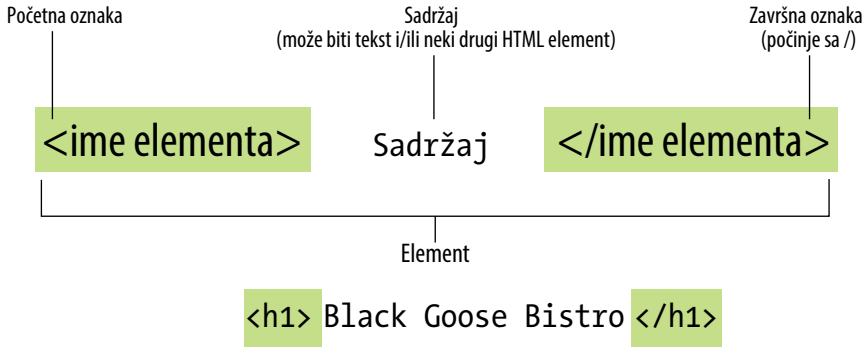
Korak 2: strukturirajte dokument

Snimili smo sadržaj u dokument sa oznakom tipa *.html* – sada možemo da ga označavamo.

Predstavljamo HTML elemente

U poglavlju 2, *Kako funkcioniše veb*, videli ste primere HTML elemenata s početnom oznakom (za pasus, na primer) i završnom oznakom ( ). Pre nego što počnemo da dodajemo oznake našem dokumentu, pogledaćemo anatomiju jednog HTML elementa (njegovu *sintaksu*) i proći kroz najvažniju terminologiju. Generički kontejnerski element prikazan je na slici 4-6.

Element čine i sadržaj i oznake.



Slika 4-6. *Delovi kontejnerskog HTML elementa.*

Elementi se prepoznaju po *oznakama* (engl. *tags*) u izvornom tekstu. Oznaka se sastoji od imena elementa (obično je to skraćenica nekog dužeg, opisnog imena) unutar ugaonih zagrada (< >). Čitač veba zna da je tekst unutar zagrada skriven i ne prikazuje ga u svom prozoru.

Ime elementa se pojavljuje u *početnoj oznaci* (engl. *opening tag, start tag*) i zatim u *završnoj oznaci* (engl. *closing tag, end tag*) ispred koje je kosa crta. Završna oznaka deluje kao prekidač za isključivanje elementa. Pazite da ne koristite obrnutu kosu crtu (engl. *backslash*) u završnim oznakama (videti savet „Kosa crta i obrnuta kosa crta").

Oznake dodate oko sadržaja nazivaju se *označavanje* (engl. *markup)*. Ne zaboravite da se *element* sastoji od sadržaja *i* svog označavanja (početne i završne oznake). Nemaju, međutim, svi elementi sadržaj. Neki su po definiciji *prazni*, na primer element `img`, koji se koristi za dodavanje slike strani. O praznim elementima čitaćete malo kasnije u ovom poglavlju.

Još jedna stvar ...velika slova. U HTML-u nije važno da li su imena elemenata napisana velikim ili malim slovima. Znači, što se tiče čitača veba, **<img>**, **<Img>** i **<IMG>** jednaki su. Međutim, u XHTML-u (strožoj verziji HTML-a) imena svih elemenata moraju biti napisana malim slovima. Mnogim veb programerima i dizajnerima sviđa se urednost XHTML-a, pa sve pišu malim slovima – što ću i ja raditi u ovoj knjizi.

SAVET

Kosa crta i obrnuta kosa crta

U HTML oznakama i URL adresama koristi se kosa crta (/). Ona se na standardnoj QWERTY tastaturi nalazi ispod upitnika (?).

Kosu crtu je lako pomešati sa obrnutom kosom crtom (\), koja se nalazi ispod znaka za uspravnu crtu (|). Obrnuta kosa crta ne funkcioniše ni u oznakama ni u URL adresama, pa je nemojte koristiti.



JAVASCRIPT: PONAŠANJE VEB STRANA

DEO IV

U OVOM DELU

Poglavlje 19
Uvod u JavaScript

Poglavlje 20
Korišćenje JavaScripta

UVOD U JAVASCRIPT

Autor: Mat Marquis

U OVOM POGLAVLJU

Šta JavaScript jeste a šta nije
Promenljive i nizovi
Naredbe if/else i petlje
Ugrađene i namenske funkcije
Objekti čitača veba
Procedure za obradu događaja

Ovo poglavlje sadrži uvod u JavaScript. Ukoliko zazirete od toga, razumljivo je. Prelazimo na „opasnu teritoriju“ pravog programskog jezika, a to može biti pomalo zastrašujuće. Ipak, nije tako strašno, obećavam!

Počećemo od toga šta JavaScript jeste – a šta nije – i razmotriti neke načine na koje se on koristi. U najvećem delu poglavlja upoznajete se sa sintaksom JavaScripta – promenljivama, funkcijama, operatorima, petljama i slično. Da li ćete pisati kôd kada završite čitanje ovog poglavlja? Verovatno nećete.

Ali, steći ćete dobru osnovu i bolje ćete razumeti šta se dešava u skriptu na koji naiđete. Poglavlje se završava razmatranjem nekih načina na koje možete da manipulišete prozorom čitača i da povezujete skriptove sa akcijama korisnika kao što su pritiskanje mišem ili slanje obrasca.

Šta je JavaScript?

Ako ste stigli do ovog dela knjige, verovatno već znate da je JavaScript programski jezik koji je zaslužan za interaktivnost i namenski definisana ponašanja veb lokacija. On je *jezik za skriptovanje na klijentskoj strani*, što znači da se izvršava na korisnikovom računaru a ne na serveru kao drugi programski jezici, npr. PHP i Ruby. To znači da se JavaScript (i načini na koje ga koristimo) oslanja na mogućnosti i parametre čitača veba. Moguće je čak i da ne bude dostupan, i to ili zato što ga je korisnik isključio ili zato što ga dati uređaj ne podržava – što je nešto čega su dobri programeri svesni i o čemu vode računa. JavaScript je poznat i kao *dinamički* i *labavo tipiziran* (engl. *loosely typed*) programski jezik. Ne razmišljajte mnogo o ovim opisima – kasnije ću objasniti šta oni znače.

Najpre želim da pokažem da je JavaScript donekle pogrešno shvaćen.

Šta JavaScript nije

Kao prvo, njegovo ime prilično zbunjuje jer JavaScript nema nikakve veze s Javom. Napravio ga je Brendan Ajh u kompaniji Netscape 1995. godine i prvobitno ga nazvao „LiveScript". Ali, pošto je u to vreme Java bila glavni hit, „LiveScript" je iz marketinških razloga postao „JavaScript". Ili samo „JS", ukoliko želite da zvučite najopuštenije moguće dok pričate o JavaScriptu.

JS ima i pomalo lošu reputaciju. Jedno vreme je bio sinonim za najrazličitije bezočne prevare na internetu – neželjene redirekcije, dosadne iskačuće prozore i brojna nebulozna „ugrožavanja sigurnosti", da pomenemo samo neke. Nekada je JavaScript omogućavao nesavesnim programerima da rade sve to (pa i gore stvari), ali su savremeni čitači veba uglavnom uspeli da se izbore s tom tamnom stranom programiranja u JavaScriptu i onemoguće je. Ipak, ne bi trebalo da okrivljujemo sâm JavaScript za takvo stanje. Kao što ne tako stara izreka kaže: „Uz veliku moć ide i velika odgovornost". JavaScript je oduvek pružao programerima ogromnu mogućnost kontrole nad načinom prikazivanja strana i ponašanjem čitača, pa je na nama da je koristimo odgovorno.

Šta JavaScript jeste

NAPOMENA

JavaScript je 1996. godine standardizovalo Evropsko udruženje proizvođača računara (European Computer Manufacturer's Association, ECMA), pa ga ponekad zovu i ECMAScript.

Sada znamo šta JavaScript nije: nije povezan s Javom i nije brkati zlikovac koji vas vreba iz čitača veba, kršeći ruke i čekajući da vas obavesti o „atraktivnim samcima u vašoj okolini". Hajde da vidimo šta JavaScript jeste.

JavaScript je ne mnogo složen ali izuzetno moćan jezik za pisanje skriptova. Najčešće ga srećemo posredstvom čitača veba, ali se probio svuda – od ugrađenih aplikacija, preko PDF-ova do e-knjiga. Čak i veb servere može pokretati JavaScript.

Kao *dinamički programski jezik*, JavaScript ne mora da se izvršava uz posredovanje bilo kakvog kompajlera koji prevodi naš kôd razumljiv ljudima u nešto što razume čitač veba. Čitač efikasno čita kôd isto kao i mi, i interpretira ga u hodu.

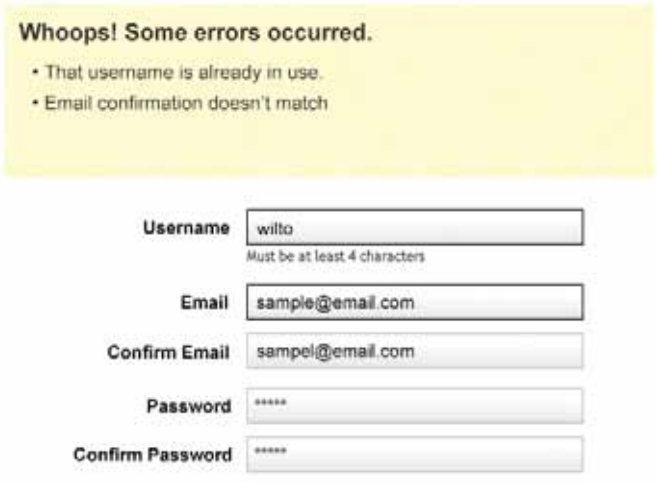
JavaScript je, takođe, *labavo tipiziran jezik*. To znači da ne moramo da saopštimo JavaScriptu tip promenljive. Ako promenljivoj dodeljujemo vrednost 5, ne moramo programski definisati tu promenljivu kao broj. Kao što ste možda i sami primetili, 5 je broj i JavaScript ga prepoznaje kao takvog.

Ipak, ne morate pamtiti pomenute termine da biste počeli da pišete JS kôd – iskreno, ja nisam. Navodim ih samo da bih vam predstavio nekoliko pojmova koje ćete često sretati dok budete učili JavaScript, i koji će s vremenom postajati sve smisleniji.

Šta JavaScript može

JavaScript se najčešće koristi za dodavanje interaktivnosti veb strani. Dok „strukturni“ sloj strane čine HTML oznake a „prezentacioni“ CSS stilovi, za treći sloj – „ponašanje“ – zaslužan je JavaScript. Skriptovi mogu pristupati svim elementima, atributima i tekstu na veb strani pomoću objektnog modela dokumenta (engl. *Document Object Model, DOM*), koji ćemo opisati u poglavlju 20, *Korišćenje JavaScripta*. Možemo pisati i skriptove koji reaguju na korisnikov unos, u hodu menjajući sadržaj strane, CSS stilove ili ponašanje čitača.

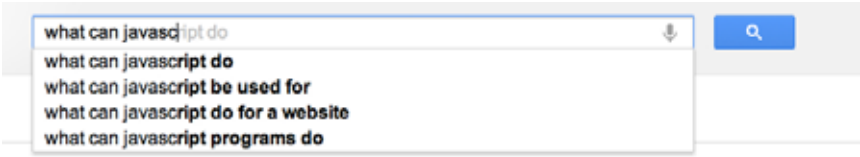
Verovatno ste videli kako to funkcioniše ako ste ikada pokušali da se registrujete na neku veb lokaciju, uneli korisničko ime i odmah dobili povratnu informaciju da to ime već koristi neko drugi (slika 19-1). Crno uokvireno polje za unos teksta i prikazivanje poruke tipa „sorry, this username is already in use“, primeri su toga kako JavaScript menja sadržaj strane i sprečava slanje obrasca, menjajući tako podrazumevano ponašanje čitača veba.



Slika 19-1. *JavaScript prepoznaje da korisničko ime nije na raspolaganju, i zatim umeće poruku i menja stilove kako bi problem postao očigledan.*

Ukratko, JavaScript vam omogućava da napravite interfejse veoma brzog odziva, koji poboljšavaju ukupno korisničko iskustvo i obezbeđuju dinamičko funkcionisanje bez čekanja da server učita novu stranu. Na primer, pomoću JavaScripta možemo uraditi bilo šta od sledećeg:

- Predložiti kompletan pojam koji je korisnik počeo da unosi u polje za pretragu – dok ga unosi. To možete videti na Google.com (slika 19-2).



Slika 19-2. *Google.com koristi JavaScript za automatsko dopunjavanje traženog pojma dok se on unosi.*



Slika 19-3. *JavaScript se može koristiti za prikazivanje i skrivanje delova sadržaja.*

- Tražiti sadržaj i informacije od servera i ubaciti ih u tekući dokument prema potrebi, ne učitavajući ponovo celu stranu – što je poznato i kao tehnologija „Ajax".
- Prikazati i sakriti sadržaj kada korisnik pritisne vezu ili naslov, i tako dobiti „stapajuću" oblast za sadržaj (slika 19-3).
- Ispitati osobine i mogućnosti pojedinačnih čitača. Na primer, možete ispitati postoje li „događaji dodira", koji ukazuju na interakciju korisnika sa stranom preko čitača veba na mobilnom uređaju, pa dodati pogodnije stilove i metode interakcije.
- Popuniti praznine tamo gde ugrađene mogućnosti čitača ne zadovoljavaju potrebe, ili starijim čitačima dodati neke mogućnosti iz novijih. Takve vrste skriptova obično se nazivaju *shims* ili *polyfills*.
- Učitati sliku ili sadržaj u namenski stilizovan, modalni pano za prikaz (engl. *lightbox*) – izolovan na strani pomoću CSS-a – nakon što korisnik pritisne umanjenu verziju date slike (slika 19-4).



Slika 19-4. *JavaScript se može koristiti za učitavanje slika u galerije s panoima za prikaz.*

Ova lista nije ni izbliza detaljna!

Dodavanje JavaScript koda veb strani

Kao i u slučaju CSS-a, skript možete ugraditi direktno u dokument ili ga držati u spoljnoj datoteci i povezati sa stranom. U obe metode koristi se element **script**.

Ugrađen skript

Da biste skript ugradili u stranu, samo dodajte odgovarajući kôd kao sadržaj elementa **script**:

```
<script>
… Ovde ide JavaScript kôd
</script>
```

Spoljni skriptovi

U drugoj metodi koristi se atribut **src** za pokazivanje na datoteku sa skriptom (sa nastavkom *.js*) pomoću njenog URL-a. U ovom slučaju, element **script** je prazan.

```
<script src="my_script.js"></script>
```

Prednost spoljnih skriptova je to što isti skript možete primeniti na više strana (ista prednost koju pružaju i spoljni opisi stilova). Naravno, nedostatak je to što se za svaki skript upućuje zaseban HTTP zahtev serveru, što usporava rad.

Položaj skripta

Element **script** može da se nalazi bilo gde u dokumentu, ali se skriptovi najčešće smeštaju unutar eelementa **head** dokumenta i na sam kraj elementa **body**. Preporučljivo je da ih ne razbacijute svuda po dokumentu jer bi se onda teško pronalazili i održavali.

Za većinu skriptova, najpogodnije mesto je kraj dokumenta, neposredno ispred oznake **</body>**, zato što će čitač veba tada već završiti s raščlanjavanjem (analiziranjem) dokumenta i njegove DOM strukture. Znači, te informacije će biti spremne i dostupne u vreme kada čitač dođe do skriptova, pa se oni mogu brže izvršiti. Osim toga, učitavanje i izvršavanje skriptova blokira iscrtavanje strane na ekranu, pa performanse strane posetiocu deluju bolje kada se skriptovi izvršavaju na kraju. Ipak, možda ćete ponekad želeti da skript obavi nešto pre nego što se telo dokumenta potpuno učita, pa ćete smeštanjem skripta u element **head** dobiti bolje performanse.

NAPOMENA

U verziji HTML 4.01, oznaka **script** *mora da sadrži atribut* **type** *da bi bila ispravna:*

```
<script type="text/
javascript">…</script>
```

U XHTML dokumentima, sadržaj elementa **script** *morate identifikovati kao CDATA kôd u sledećem omotaču:*

```
<script type="text/javascript">
  // <![CDATA[
  …Ovde ide JavaScript kôd
  // ]]>
</script>
```

Anatomija skripta

Postoji razlog zbog kojeg knjiga Dejvida Flanagana *JavaScript: sveobuhvatni vodič* (O'Reilly, Mikro knjiga) ima preko 1000 strana. Ima mnogo toga da se kaže o JavaScriptu! U ovom odeljku, na raspolaganju nam je samo nekoliko strana da vas upoznamo sa osnovnim gradivnim elementima JavaScripta

kako biste razumeli skriptove kada naiđete na njih. Mnogi veb programeri su sami naučili da koriste JavaScript tako što su pronalazili postojeće skriptove i prilagođavali ih svojim potrebama. Nakon malo prakse, bili su spremni da počnu da pišu sopstvene skriptove od nule. Možda i vi želite da naučite da sami pišete JavaScript kôd kako biste zaokružili svoja umeća veb dizajnera. Prvi korak ka tome je prepoznavanje delova skripta, pa ćemo odatle i početi.

```
alert("Hi there");
```




```
confirm("I'm gonna do something, okay?");
```




```
prompt("What should I do?");
```



Slika 19-5. *Ugrađene JavaScript funkcije:* **`alert()`** *(gore),* **`confirm()`** *(u sredini) i* **`prompt()`** *(dole).*

Funkcionalnost JavaScripta prvobitno je bila uglavnom ograničena na sirove metode interakcije s korisnikom. Mogli smo koristiti nekoliko ugrađenih funkcija JavaScripta (slika 19-5) da bismo omogućili dobijanje povratnih informacija od korisnika; takve su bile funkcija **`alert()`** za slanje obaveštenja korisniku i **`confirm()`** pomoću koje smo tražili da korisnik odobri ili odbije akciju. Kada je trebalo zahtevati da korisnik unese nešto, najčešće smo bili ograničeni na ugrađenu funkciju **`prompt()`**. Mada ove metode i danas imaju svoje mesto, one predstavljaju neprijatan, nametljiv i – bar po uobičajenom shvatanju – prilično grozan način interakcije s korisnikom. S razvojem JavaScripta, dobili smo mnogo suptilnije načine za definisanje ponašanja veb strana, što je doprinelo i boljem iskustvu korisnika u radu s njima.

Da bismo iskoristili prednosti tih metoda interakcije, prvo moramo razumeti logiku na kojoj se zasniva skriptovanje. To su logički obrasci (engl. *logic patterns*) zajednički svim programskim jezicima, bez obzira na njihovu različitu sintaksu (slično kao što se reči u govornim jezicima najčešće razlikuju, ali većina njih deli mnoge gramatičke obrasce).

Do kraja ovog odeljka, upoznaćete promenljive, nizove, operatore poređenja, naredbe if/else, petlje, funkcije i još mnogo toga. Spremni?

Osnove

Nekoliko uobičajenih sintaksnih pravila važi za ceo JavaScript.

Važno je znati da JavaScript *razlikuje mala i velika slova*. Promenljive po imenu „mojaPromenljiva“, „mojapromenljiva“ i „MOJAPromenljiva“ tretiraće se kao tri različita objekta. Pored toga, beline kao što su tabulatori i razmaci se ignorišu, osim ako su deo znakovnog niza (teksta) i obuhvaćeni navodnicima.

JavaScript razlikuje mala i velika slova.

Naredbe

Skript se sastoji od niza *naredaba* (engl. *statements*). Naredba je komanda kojom se nalaže čitaču šta da radi. Evo jednostavne naredbe koja čini da čitač prikaže poruku „Thank you.“

```
alert("Thank you.");
```

Tačka i zarez na kraju naredbe kazuju JavaScriptu da je komanda završena, isto kao što tačka označava kraj rečenice. Prema JavaScript standardu, znak za prelom reda takođe će označiti kraj komande, ali se završavanje svake naredbe tačkom i zarezom smatra dobrom praksom.

Komentari

JavaScript omogućava da ostavljate komentare koji će se zanemarivati u vreme izvršavanja skripta, tako da možete ubacivati podsetnike i objašnjenja kroz ceo kôd. To je posebno korisno ukoliko će vaš kôd u budućnosti menjati neki drugi programer.

Dva su načina korišćenja komentara. Za jednoredne komentare koristite dve kose crte (**//**) na početku reda. Jednoredni komentar možete pisati u istom redu kao naredbu, pod uslovom da je iza nje. Komentar ne morate zatvarati jer ga zatvara znak za prelom reda.

```
// Ovo je jednoredni komentar.
```

Za višeredne komentare koristi se ista sintaksa kao u CSS-u. Čitač ignoriše sve što se nalazi između znakova **/* */**. Možete ih koristiti da biste „isključili“ napomene, pa i delove skripta, kada ispitujete ima li grešaka.

```
/* Ovo je višeredni komentar.
Sve što se nalazi između ovih znakova potpuno
će se zanemariti pri izvršavanju skripta.
Ovaj oblik komentara se mora zatvoriti. */
```

Koristiću jednoredne komentare da bih dodao kratka objašnjenja u primere koda, a ranije opisanu funkciju **alert()** (slika 19-5) upotrebiću da bismo na brzinu pogledali rezultate našeg rada.

Promenljive

Promenljivu zamislite kao kontejner za podatke.

Ako ste slični meni, i sâm termin „promenljive“ pokreće neprijatna sećanja na srednjoškolske časove matematike. Suština je uglavnom ista, mada sada nema vašeg smešno začešljanog profesora.

Promenljiva je kontejner za podatke. Date joj ime a zatim dodelite vrednost, koja može biti broj, znakovni niz, element DOM-a ili funkcija – u stvari, bilo šta. Tako dobijamo zgodan način da je kasnije referenciramo po imenu. Sama vrednost se može menjati i ponovo dodeljivati kako god nalaže logika skripta.

Sledećom deklaracijom pravi se promenljiva po imenu „foo“ i dodeljuje joj se vrednost 5:

```
var foo = 5;
```

Počinjemo tako što deklarišemo promenljivu pomoću rezervisane reči **var**. Jedan znak jednakosti (=) ukazuje na to da joj dodeljujemo vrednost. Pošto je to kraj naše naredbe, red završavamo tačkom i zarezom. Promenljive se mogu deklarisati i bez rezervisane reči **var**, što utiče na to koji će deo skripta imati pristup njihovom sadržaju. To ćemo detaljnije razmotriti u odeljku „Oblast važenja promenljivih i rezervisana reč var“, kasnije u ovom poglavlju.

Kao ime promenljive možete koristiti bilo šta, ali obavezno izaberite neko koje će vam i kasnije imati smisla. Svakako ne biste želeli da promenljivoj date ime „podaci“; ono bi trebalo da opisuje podatke koje promenljiva sadrži. U našem