

VEB DIZAJNI
MULTIMEDIJALNI
SISTEMI

Dragan Cvetković Nikola Savanović

UNIVERZITET SINGIDUNUM

**Dragan Cvetković
Nikola Savanović**

VEB DIZAJN I MULTIMEDIJALNI SISTEMI



BEOGRAD, 2025.

VEB DIZAJN I MULTIMEDIJALNI SISTEMI

AUTORI:

dr Dragan Cvetković, redovni profesor

dr Nikola Savanović, docent

RECENZENTI:

Prof. dr Aleksandar Bengin, Mašinski fakultet Univerziteta u Beogradu

Prof. dr Dejan Živković, UNIVERZITET SINGIDUNUM u Beogradu

Prof. dr Aleksandar Jevremović, UNIVERZITET SINGIDUNUM u Beogradu

IZDAVAČ:

UNIVERZITET SINGIDUNUM

Danijelova 32, Beograd

www.singidunum.ac.rs

ZA IZDAVAČA:

Prof. dr Milovan Stanišić

TEHNIČKA OBRADA:

Dragan Cvetković

DIZAJN KORICA:

Aleksandar Mihajlović, master

GODINA IZDANJA:

2025.

TIRAŽ:

1.500 primeraka

ŠTAMPA:

Birograf, Beograd

ISBN: 978-86-7912-795-2

© Sva prava zadržavaju autori. Nije dozvoljeno da, bez prethodne pismene dozvole autora, bilo koji deo ove knjige bude snimljen, emitovan ili reprodukovano, uključujući, ali ne i ograničavajući se na fotokopiranje, fotografiju, magnetni ili bilo koji drugi vid zapisa.

Sadržaj

Predgovor	ix
1 Tekst	1
1.1. Značenje i njegova snaga	2
1.2. O fontovima	2
1.2.1. Tipografski merni sistem	3
1.2.2. Serif i Sans Serif	7
1.3. Upotreba teksta u multimediji	8
1.3.1. Dizajniranje pomoću teksta	9
1.3.2. Izbor tekstualnih fontova	9
1.3.3. Meniji za navigaciju	11
1.3.4. Dugmad ili tasteri za interakciju	11
1.3.5. Polja za čitanje	12
1.3.6. HTML dokumenti	12
1.3.7. Simboli i ikonice	13
1.3.8. Animirani tekst	14
1.4. Kodni sistemi za prikazivanje znakova	14
1.4.1. Kodni sistem ASCII (ISO 646)	15
1.4.2. ISO 8859 porodica	16
1.4.3. <i>Windows</i> set znakova	18
1.4.4. <i>Unicode</i>	18
1.5. Jezici u svetu računara	19
1.6. Uređivanje teksta – pojmovi, pravila, saveti	20
1.7. Hipermedija i hipertekst	22
2 Slike	25
2.1. Kreiranje statičnih (nepokretnih) slika	25
2.1.1. Bitmape	26

2.1.2.	Softver za bitmape	26
2.1.3.	"Hvatanje" i editovanje slika	27
2.1.4.	Skeniranje slika	28
2.1.5.	Vektorska grafika	28
2.1.6.	3D crteži i renderovanje	29
2.2.	Boje	31
2.2.1.	Razumevanje boja	32
2.2.2.	RGB – osnovne aditivne boje	33
2.2.3.	CMY, CMYK i PANTONE modeli boja	35
2.2.4.	HSV, HSI i HSL modeli boja	36
2.2.5.	Još ponešto o bojama	39
2.3.	Grafički formati	39
2.3.1.	Često korišćeni <i>Windows</i> grafički formati	40
2.3.2.	<i>Macintosh</i> grafički formati	42
2.3.3.	Još neki grafički formati	42
2.4.	Licenciranje slika	43
3	Zvuk	47
3.1.	Prednosti zvuka	47
3.2.	Multimedijalni sistemski zvuci	48
3.3.	Digitalni audio	49
3.4.	Kreiranje MIDI audia	53
3.5.	Formati za audio datoteke	54
3.5.1.	Vrste audio formata	54
3.5.2.	Komprimovanje audio ili zvučnih podataka	56
3.6.	MIDI i digitalni audio	57
3.7.	Dodavanje zvuka u multimedijalni projekat	58
3.8.	Licenciranje	59
4	Animacija	63
4.1.	Principi animacije	63
4.2.	Tipovi animacije	64
4.3.	Vrste računarske animacije	64
4.4.	Budućnost računarske animacije	67

5	Video	69
5.1.	Kako video radi?	70
5.2.	Standardi za analogno prikazivanje	71
5.2.1.	NTSC	71
5.2.2.	PAL/SECAM	72
5.3.	Digitalni video	72
5.3.1.	HDTV	72
5.3.2.	Analogno i digitalno	73
5.3.3.	Digitalna obrada signala	75
5.3.4.	Prednosti digitalne obrade signala	76
5.4.	Standardi za digitalno prikazivanje	77
5.4.1.	Rezolucija	77
5.4.2.	Rezolucija 4K	77
5.4.3.	Rezolucija 8K	78
5.4.4.	Rezolucija 10K	78
5.4.5.	Broj slika u sekundi	78
5.4.6.	Kodeci	81
5.5.	Izrada video zapisa ili video klipa	81
5.5.1.	Merenje vremena u video klipu	82
5.5.2.	Kako vremenska baza i brzina slika utiču jedna na drugu?	82
5.5.3.	Merenje vremena korišćenjem vremenskog koda	83
5.5.4.	Televizijska slika	84
5.5.5.	Merenje veličine slike i rezolucije	85
5.5.6.	Proporcija	85
5.5.7.	Nekvadratni pikseli	85
5.5.8.	Veličina slike	86
5.5.9.	Preskeniranje i bezbedne zone	86
5.5.10.	Dubina bita	87
5.5.11.	Razumevanje komprimovanja video podataka	88
5.5.12.	Izbor metoda za komprimovanje video zapisa	89
5.5.13.	Prostorno komprimovanje	89
5.5.14.	Vremensko komprimovanje	90
5.5.15.	Komprimovanje sa očuvanjem kvaliteta	90
5.5.16.	Komprimovanje sa narušavanjem kvaliteta	91
5.5.17.	Asimetrično i simetrično komprimovanje	91

5.5.18. DV komprimovanje	91
5.5.19. Digitalno presnimavanje video zapisa	92
5.6. Komprimovanje AV podataka i optički medijumi	93
5.6.1. Zašto je digitalni video komprimovan?	93
5.6.2. Komprimovanje bez gubitaka	94
5.6.3. Komprimovanje sa gubitkom	94
5.6.4. MPEG-4 video	94
5.6.5. MP4 format	95
5.6.6. MPEG-4 audio	95
5.6.7. Neki tipovi digitalnih audio datoteka	96
5.6.8. Neki tipovi digitalnih video datoteka	96
5.7. Snimanje i uređivanje videa	97
5.7.1. Planiranje ili scenario	97
5.7.2. Platforma za snimanje	97
5.7.3. Osvetljenje	98
5.7.4. Hroma ključevi (<i>Chroma Keys</i>)	98
6 Virtuelna i proširena realnost	99
6.1. Definicija i princip virtualne realnosti	99
6.2. Uređaji za virtualnu realnost	101
6.3. Primene virtualne realnosti	106
6.4. Proširena realnost	108
7 Softverski alati za multimedijalni projekat	111
7.1. Obrada teksta	112
7.2. Optičko prepoznavanje znakova	113
7.3. Vektorska i rasterska grafika	114
7.4. 3D modelovanje i animacija	117
7.5. Uređenje slike	119
7.6. Uređenje zvuka	120
7.7. Uređenje animacija, videa i digitalnih filmova	121
7.8. Korisni pribori	121
8 Planiranje i troškovi multimedijalnog projekta	123
8.1. Analiza	123
8.2. Prethodno testiranje	127
8.3. Razvoj prototipa	130

8.4. Alfa razvoj	131
8.5. Beta razvoj	131
8.6. Isporuka/dostava	132
9 Dizajn i proizvodnja multimedijalnog projekta	133
9.1. Dizajn multimedijalnog projekta	134
9.1.1. Dizajniranje ili projektovanje strukture	134
9.1.2. Dizajniranje ili projektovanje korisničkog interfejsa	138
9.2. Proizvodnja multimedijalnog projekta	140
9.2.1. Početak	140
9.2.2. Rad sa klijentima	140
9.2.3. Praćenje i nadzor	141
9.2.4. Autorska prava	142
9.3. Tim za izradu multimedijalnog projekta	142
9.3.1. Rukovodilac ili menadžer projekta	142
9.3.2. Dizajner multimedije	142
9.3.3. Dizajner interfejsa	143
9.3.4. Pisac za multimediju	143
9.3.5. Stručnjak ili ekspert za video	143
9.3.6. Stručnjak ili ekspert za audio	143
9.3.7. Programer multimedije	144
9.3.8. Producent multimedije za veb	144
9.3.9. Programer	144
10 HTML i HTML5	145
10.1. HTML i osnove veb strana	146
10.2. HTML i tekst	152
10.3. HTML i slike	159
10.4. HTML i linkovi i/ili veze	169
10.5. HTML i tabele	175
10.6. HTML, zvuk i video	185
10.7. HTML i forme	188
10.8. HTML i sekcije (frejmovi)	196
10.9. Šta je doneo HTML5?	202
10.10. HTML5 i semantika	203
10.11. HTML5, zvuk i video	206

11 CSS i CSS3	209
11.1. CSS od početka	210
11.2. CSS i pozicioniranje	220
11.3. CSS i tekst	223
11.4. CSS i vizuelni efekti	240
11.5. CSS3 komande i noviteti	255
11.5.1. CSS3 i granični okviri	256
11.5.2. CSS3 i pozadine	257
11.5.3. CSS3 i gradijentni prelazi boja	258
11.5.4. CSS3 i efekti na tekstu	262
11.5.5. CSS3 i 2D (ravanske) transformacije	262
11.5.6. CSS3 i 3D (prostorne) transformacije	263
11.5.7. CSS3 i prelazni (tranzicioni) efekti	263
11.5.8. CSS3 i animacija	265
11.5.9. CSS3 i višestruke kolone	266
11.6. CSS modul <i>Flexible Box Layout</i>	268
11.6.1. Poravnavanje elemenata unutar fleksibilnog kontejnera	271
11.6.2. Raspoređivanje elemenata u fleksibilnom kontejneru	272
11.7. CSS sistem mreže	274
12 HTML5 i osnove računarske grafike	281
12.1. Pravougaoni elementi i površi	285
12.2. Linije, završeci linija i zatvorene konture	286
12.3. Kružnice i opšti kružni lukovi	288
12.4. Tekst	291
12.5. FOR petlja	292
12.6. Definisane boje i providnost	294
12.7. Gradijentni prelazi boja	296
12.8. Umetanje fotografija u grafičku oblast	297
12.9. Transformacije objekata	299
12.10. Kreiranje šablona	299
12.11. Kreiranje senki	300
12.12. Zadaci za vežbu	301

13 JavaScript	303
13.1. Istorija JavaScript tehnologije	303
13.2. Podrška JavaScript tehnologiji u pretraživačima	304
13.3. JavaScript u pretraživačima	304
13.4. Razvojna okruženja	305
13.5. Razvojna konzola veb pretraživača	305
13.6. Osnovni jezici na vebu	306
13.7. Upotreba JavaScript jezika i izlaz koda	307
13.8. Izlaz	308
13.9. Sintaksa	308
13.10. Instrukcije	308
13.11. Komentari	308
13.12. Operatori i aritmetika	309
13.13. Dodeljivanje vrednosti	313
13.14. Tipovi podataka	314
13.15. Funkcije	315
13.16. Objekti	317
13.17. Forme, događaji i osluškivači događaja	319
13.18. Brojevi	322
13.19. Matematički objekti	322
13.20. Nizovi	323
13.21. Uslovni iskazi	325
13.22. Petlje	328
13.23. Greške	331
13.24. Rezervisane (ključne) reči	332
13.25. Asinhroni JavaScript	332
13.26. Web API	333
13.27. Kompatibilnost	334
14 jQuery	335
14.1. Korišćenje	335
14.2. Verzije	335
14.3. Ugradnja jQuery koda u dokument	336
14.4. Selektori	336
14.5. jQuery događaji	336
14.6. Implementacija koda	337

14.7. Manipulacija DOM-a	337
15 Google analitika	339
15.1. Značaj analitike	339
15.2. GA4	341
16 Saveti za dobar UI/UX dizajn	345
16.1. Ciljna publika	347
16.2. Korisničko iskustvo i unapređenje stope konverzije	347
16.3. SEO	349
Literatura	351
Internet adrese	355

Predgovor

Kada se postavi pitanje o tome šta dizajn predstavlja, jednostavna definicija bi mogla da bude da se dizajn odnosi na postizanje ciljeva u okviru ograničenja. Ova definicija ne govori sve o dizajnu, ali pomaže da se čovek fokusira na sledeće elemente:

- **Ciljevi** – Šta je svrha dizajna i budućeg proizvoda? Za koga se to pravi? Zašto oni to žele?
- **Ograničenja** – Koje materijale bi trebalo koristiti? Koje standarde bi trebalo usvojiti? Koliko to može da košta? Koliko je potrebno vremena da se proizvod razvije? Da li postoje zdravstveni i bezbednosni problemi?
- **Razmena (kompromis)** – Trebalo bi izabrati i definisati koji ciljevi ili ograničenja mogu da se usvoje u blažoj formi, a koja ograničenja moraju da se poštuju do najsitnijih detalja.

Nemoguće je uvek ostvariti sve korisnikove ciljeve u okviru ograničenja, ali u životu je sve pitanje kompromisa, pa i u ovakvim slučajevima. Najbolji dizajni se kreiraju u sredinama gde dizajner razume kompromise i faktore koji utiču na njih.

Veb dizajn može da se shvati kao proces prikupljanja ideja i njihovog estetskog uređivanja i raspoređivanja radi postizanja određenog cilja, a to je izrada veb sajta. Krajnji korisnici i njihovo što lakše snalaženje na određenom veb sajtu ono je što bi trebalo da veb dizajn prvo uzima u obzir. Veb dizajn podrazumeva upotrebu raznih veština i disciplina koje se koriste za planiranje i izradu veb stranica i sajtova.

Multimedija je zajednički naziv za medije koji kombinuju više tipova pojedinačnih medija, da bi se stvorila jedna celina. U običnom govoru multimedija najčešće znači interaktivni računarski projekat u kome se koristi film, video, animacija, tekst i zvuk.

Proteklih godina, multimedija se sve više koristi u školstvu. Mogućnost ujedinjenja više komponenti (5 stubova multimedije – audio i video zapis, tekst, grafika i animacija) pomaže prosvetnim radnicima da prenesu učenicima i studentima znanje kroz jedinstven način. Učenici i studenti bolje i brže uče koristeći ove metode, a nastavni materijal je zanimljiviji i može biti zabavan.

Ova knjiga spada u grupu stručnih dela iz oblasti računarskih i inženjerskih nauka, koja pokriva značajno područje, koje se izučava na nivou dodiplomskih i posrediplomskih studija danas i u svetu i kod nas. Predviđena je da bude osnovni udžbenik iz predmeta **Veb dizajn i multimedijalni sistemi** ili, eventualno, **Multimedijalni sistemi** koji se izučavaju unutar

studijskih programa Informacione tehnologije i Softversko i informaciono inženjerstvo na UNIVERZITETU SINGIDUNUM u Beogradu. Ova knjiga može da se koristi i na ostalim visokoškolskim ustanovama za predmete koji imaju sličan sadržaj.

Tokom pisanja ove knjige ideja je bila da se materija izloži u što popularnijem stilu da bi bila dostupna i razumljiva i studentima sa manjim obimom predznanja iz ove oblasti. Knjiga je namenjena širem spektru potreba i širem krugu čitalaca.

Knjiga se sastoji iz šesnaest poglavlja. Prvi deo knjige, od prvog do devetog poglavlja, predstavlja multimediju kao izatkanu kombinaciju digitalno manipulisane teksta, grafike, zvuka, animacije i video elemenata. Danas tekst i sposobnost čitanja predstavljaju "prolaz" ka znanju i, eventualno, moći. Čitanje i pisanje su očekivane i neophodne veštine u većini modernih društava. Kao što je bio slučaj kroz istoriju, tekst i dalje predaje informacije mogu imati moćna značenja. Nakon "eksplozije" interneta tekst je postao važniji nego ikad. Prvo poglavlje se bavi ovom problematikom.

Ono što se vidi na ekranu multimedijalnog računara u bilo kom trenutku predstavlja mešavinu teksta, simbola, fotografskih bitmapa, vektorske grafike, trodimenzionalnih prikaza, različitih dugmadi koju treba kliknuti i prozora pokretnog videa. Neki delovi slike se mogu pomerati ili kretati tako da ekran nikad ne izgleda mirno i izaziva oči. Računarski ekran je akcija – sadrži mnogo više od poruke; to je, takođe, osnovna veza gledaoca sa svim sadržajima nekog projekta. U drugom poglavlju se nalazi priča "vezana" za slike.

Zvuk je verovatno najuzbudljiviji deo multimedije. To je univerzalan govor na bilo kom jeziku. Pruža zadovoljstvo slušanja muzike, a može i da iznenadi senzacionalnim specijalnim efektima ili da utiče na raspoloženje iz "drugog plana". Način na koji se koristi zvuk može da označi razliku između obične i profesionalne, spektakularne multimedijalne prezentacije. Treće poglavlje je posvećeno ovoj problematici.

Animacija je postupak stvaranja iluzije kretanja crtežima, modelima ili beživotnim stvarima. To je optička iluzija kretanja zahvaljujući fenomenu poznatom kao perzistencija vida i može se izvesti na više načina. Pokretni video je element multimedije koji može čvrsto da drži interes studenta za učenje pomoću računara. Digitalni video najviše i angažuje od svih multimedijalnih mogućnosti; to je vrlo moćno sredstvo da se korisnici računara približe realnom svetu. Četvrto i peto poglavlje se bave ovom vrstom problematike.

Stalni razvoj računara i informacionih tehnologija omogućuje realizaciju i primenu novih metoda i sistema kakve ranije nisu bile moguće. Jedan primer takvog razvoja je i tehnologija virtualne stvarnosti ili virtualne realnosti ili virtualnog okruženja. Tehnikama virtualne stvarnosti moguće je ostvariti realistične simulacije koje su korisne u mnogim područjima ljudske delatnosti. Simulacije su bile poznate i ranije, ali tehnike virtualne stvarnosti mogu ostvariti utisak "uplivavanja" čoveka u nepostojeći ili prividni svet. Ovom problematikom se bavi šesto poglavlje.

Softverski alati za izradu multimedijalnog projekta "govore" dostupnom hardveru šta treba da uradi. Na primer, da prikaže odgovarajuću boju, da pokrene određenu animaciju, da postavljeni baner "šeta" po monitoru sa jedne na drugu stranu, da pokrene odgovarajuću ili željenu muzičku numeru, da utiša zvuk, da pokrene željeni video materijal u određenoj oblasti i slično. Sedmo poglavlje se bavi ovom tematikom.

Pre nego što počnu sa radom na multimedijalnom projektu, autori moraju prvo da steknu utisak o obimu i sadržaju istog, kao i da stvore odgovarajuću sliku u sopstvenim glavama o dostupnim metodama i resursima kako bi preneli željene poruke do potencijalnih korisnika. Nakon toga, treba da razviju uređen nacrt i plan koji je racionalan u smislu veština, vremena, budžeta, alata i sredstava koje imaju pri ruci. Pravilno planiranje projekta je podjednako važno kao i planiranje rasporeda i sadržaja. Ovoj tematici je posvećeno osmo poglavlje.

Projektovanje ili dizajn i izgradnja ili proizvodnja multimedijalnih projekata su povezani procesi. Najbolji proizvodi su često posledica uvažavanja povratnih informacija i modifikacija koje se sprovode tokom procesa proizvodnje; projekti čiji je dizajn prerano priveden kraju imaju nedostatke i smanjene su im šanse po pitanju poboljšanja. Treba biti oprezan. Postoji opasnost da previše povratnih informacija i previše promena mogu "ubiti" projekat, trošeći nenamenski i vreme i novac. Uvek treba predlagati uravnotežene izmene i usaglasiti predložene izmene sa predviđenim budžetom. Pregled ove tematike je prikazan u devetom poglavlju.

Drugi deo knjige, od desetog do šesnaestog poglavlje posvećen je alatima i jezicima za kreiranje veb stranica i sajtova, kao i uključivanje svega navedenog u kompletan multimedijalni projekat.

Veb tehnologija je doživela značajni napredak od svog nastanka do danas. Maternji jezik veba je HTML (*Hypertext Markup Language*), prvobitno dizajniran da prikaže jednostavne tekstualne dokumente na računarskom ekranu, uz neku sliku ubačenu kao ilustraciju. Akademski članci, novinski članci, kompleksni priručnici i sadržaji kompletnih knjiga su sada dostupni da se čitaju preko interneta. Ugrađivanjem funkcija koje, na klik mišem, povezuju odabrane reči i fraze sa drugim sličnim i možda, detaljnijim materijalom, korisnik može da surfuje kroz medije koji su mnogo bogatiji nego papirne stranice knjige. Kao što je već pomenuto, jezik HTML (*Hypertext Markup Language*) je **standardizovani jezik** koji se koristi pri strukturiranju tekstova, medija i ugrađenih objekata u veb stranice i mejlove. Ovom jeziku je posvećeno deseto poglavlje, kao i najnovijoj verziji – HTML5.

"Ispod" svih veb stranica nalazi se stari i dobar HTML jezik, koji kontroliše stvari kao što su veličina fonta, boja teksta, gde slika ide, kao i informacije o drugim elementima na stranici. HTML jezik se ponekad naziva običan HTML jezik, da bi se razlikovale veb stranice izgrađene sofisticiranijim tehnikama, kao što su **liste stilova** (*style sheets*). Bez pomoći, HTML često stvara, zaista, dosadne stranice. Zbrka nastaje kada se pokušava da se napravi veb stranica koršćenjem samo HTML koda. Tu priskaču u pomoć liste stilova (*style sheets*). Jednostavan način da se razume svrha CSS-a je da se na njega gleda kao na dodatak HTML-u koji pomaže da se olakša i unapredi dizajn veb stranice. U stvari, neki CSS efekti nisu mogući bez pomoći HTML jezika. Još jedna prednost CSS-a je da omogućava da stil odredi jednom, a pretraživač može taj stil da primeni mnogo puta u dokumentu. Drugačije rečeno, CSS bi trebalo koristiti za definisanje opštih pravila o tome kako se elementi ponašaju unutar veb stranica i kako izgledaju, gde se nalaze, njihove veličine, transparentnost itd. Onda bi samo trebalo viditi računa o imenu pravila koje treba primeniti i koje treba sprovesti u pripadajućoj HTML stranici. U jedanaestom poglavlju obrađen je CSS, obrađena je i najnovija verzija – CSS3.

Element <canvas> jezika HTML5 omogućava korisniku da na lagan način definiše i koristi grafičke elemente korišćenjem jezika *JavaScript*. Mogu da se iscrtavaju grafikoni, da se prave kompozicije fotografija ili da se prave jednostavne (a i složenije) animacije. Dvanaesto poglavlje je posvećeno osnovama računarske grafike unutar HTML jezika i unutar predefinisane grafičke oblasti.

Prateći zipovani fajlovi za deseto, jedanaesto i dvanaesto poglavlje (to su fajlovi sa nazivima HTML i HTML5.zip, CSS i CSS3.zip i CANVAS.zip) biće priloženi na stranici predmeta, na sajtovima Fakulteta za informatiku i računarstvo ili Tehničkog fakulteta UNIVERZITETA SINGIDUNUM u Beogradu.

JavaScript predstavlja skript jezik, koji podržava i funkcionalno programiranje i objektno-orijentisanu paradigmu. Omogućava dodavanje različitih dinamičkih korisničkih interaktivnosti na veb stranicama i predstavlja jezik veba koji veb sajt čini **živim**. Ovom problematikom se bavi trinaesto poglavlje.

Četrnaesto poglavlje predstavlja izuzetno interaktivnu biblioteku *JQuery* koja može izvršavati različite manipulacije nad HTML dokumentima, kao npr. može rukovati različitim događajima, zatim, može raditi sa animacijama, dizajnom, vizuelnim prikazima itd.

Neizostavni deo predstavlja analitika veb sajtova. Fokus analitike je usmeren na sve korisničke procese i interakcije koje se odvijaju na veb sajtu. S tim u vezi, postoje korisnički procesi (najčešće prilikom kupovine nekog proizvoda ili usluge) koji imaju svoje odgovarajuće faze. Tri faze koje predstavljaju ove događaje jesu akvizicija, ponašanje i konverzija. Ovim elementima je posvećeno petnaesto poglavlje u skraćenom obliku.

Dizajniranje korisničkog interfejsa i korisničkog iskustva igra veoma važnu, pa čak i ključnu ulogu prilikom ostvarivanja uspeha u pogledu upotrebljivosti veb ili mobilne aplikacije. U digitalnom svetu, gde korisnici virtualnim putem mogu ostvariti interakcije sa kompanijama i korporacijama širom sveta, jednostavnost u pogledu korišćenja veb sajta ili mobilne aplikacije je od krucijalnog značaja. Ključno je da se obezbedi izuzetno kvalitetno dizajniranje i intuitivan korisnički interfejs, kako bi krajnji korisnik lako razumeo kako da pristupi željenom proizvodu ili informaciji. Ovome je posvećeno šesnaesto poglavlje.

Postoji pravilo kada se prave softverski proizvodi, a to pravilo glasi – testirati, a onda ga ponovo testirati. To je neizbežno pravilo. Mora se testirati i pregledati projekat ili veb sajt kako bi se osiguralo da nema grešaka, da je tačan i operativan i da je spreman da ispuni zahteve klijenta, čak i ako je klijent sâm autor.

Na kraju je prikazan spisak literature, s tim što treba napomenuti da je korišćena literatura prikazana na krajnje proizvoljan način; u ovom slučaju je poređana po godinama izdanja, gde je najnovija knjiga na vrhu, a najstarija na kraju spiska.

Tokom pisanja ove knjige dr Aleksandar Bengin, redovni profesor Mašinskog fakulteta u Beogradu, dr Dejan Živković i dr Aleksandar Jevremović, redovni profesori UNIVERZITETA SINGIDUNUM u Beogradu, pomogli su nam korisnim primedbama i sugestijama i mi koristimo ovu priliku da im se još jednom zahvalimo.

Glava 1

Tekst

Korišćenje teksta i simbola za komuniciranje je započelo pre skoro 6.000 godina u zemljama Mediterana (u Mesopotamiji, Egiptu, Sumeru i Vavilonu), kada su prvi simboli ispisivani na glinenim pločicama i ostavljani da se suše na suncu. Samo je članovima vladajuće klase i sveštenstva bilo dozvoljeno da čitaju i pišu piktografske znake i oznake na glinenim pločicama.



Slika 1.1. *Egipatski hijeroglifi*

Najranije poruke pisanim rečima su, obično, predstavljale informacije od vitalnog značaja za upravljanje ljudima, politikom i porezima. Pošto ovaj novi medij nije zahtevao da slabašna čovekova siva masa (mozak) uči napamet, pisane poruke su postale popularne među elitom. Ovakve nove poruke nisu bile u velikoj opasnosti da nestanu usled raznoraznih razloga. Čak i ako bi poruku presreli neprijatelji, ili konkurenti, ona bi i dalje bila nerazrešiva za sve, osim za one koji su već stekli veštinu čitanja.

U nekim ranijim dobima najveći prekršaj čoveka je bilo znanje čitanja, ako nije pripadao višoj društvenoj klasi, ili ako vladari za to nisu dali odobrenje.

Danas tekst i sposobnost čitanja predstavljaju prolaz ka znanju i, eventualno, moći. Čitanje i pisanje su očekivane i neophodne veštine u većini modernih društava. Kao što je bio slučaj kroz istoriju, tekst i dalje predaje informacije mogu imati moćna značenja.

Nakon "eksplozije" interneta tekst je postao važniji nego ikad. Maternji jezik veba je HTML (*Hypertext Markup Language*), prvobitno dizajniran da prikaže jednostavne tekstualne dokumente na računarskom ekranu, uz poneku sliku ubačenu kao ilustraciju. Akademski članci, novinski članci, kompleksni priručnici i sadržaji kompletnih knjiga su sada dostupni da se čitaju preko interneta. Ugrađivanjem funkcija koje, na klik mišem, povezuju odabrane reči i fraze sa drugim sličnim i možda, detaljnijim materijalom, korisnik može da surfuje kroz medije koji su mnogo bogatiji nego papirne stranice knjige.

Društveni uticaj ovog medija na način na koji ljudi pristupaju informacijama i koriste ih biće fundamentalan sa sazревanjem veba. Suprotno današnjem televizijskom mediju, koji se sastoji od zvuka i slike sa nekoliko naslova koji su usmereni na pasivnu publiku, veb nudi aktivno iskustvo sa dovoljno izbora. Više od televizije, sa svojih 5, 50, ili, čak, 500 kanala, veb nudi "istraživački raj" od ogromnog broja HTML dokumenata. Nedavno je firma *Google* izjavila da kroz njeno indeksiranje korisnik može da pronađe više od 22 milijarde dokumenata, u zavisnosti od toga šta traži. Pošto je reč o ozbiljnoj firmi, svakodnevno ažuriranje uključuje preko 35 milijardi veb dokumenata, 5 milijardi slika i preko 130 miliona audio i video fajlova.

1.1. Značenje i njegova snaga

Čak i obična reč može da ima više značenja, pa kada se počinje sa radom u nekom mediju važno je da se konkretne reči precizno definišu. U multimediji to su, uglavnom, reči koje će se pojaviti u naslovima, menijima i navigacionim pomoćnim elementima.

Mnogi primeri demonstriraju sledeći princip multimedije – **važno je dizajnirati oznake za naslove ekrana, menije i dugmad, koristeći reči sa najpreciznijim i najsnažnijim značenjima kako bi se izrazilo ono što zaista treba**. Treba shvatiti suptilne razlike i odgovorajuće nijanse. Izraz GO BACK! je jači i "oštriji" nego PREVIOUS; izraz QUIT je "jači" nego CLOSE. Na primer, odgovor FANTASTIČNO! može da bude mnogo bolji od TAJ ODGOVOR JE ISPRAVAN!

Reči i simboli u bilo kom obliku, izrečeni ili zapisani, najčešći su sistem komunikacije. Trebalo bi predočiti najšira značenja najvećem broju ljudi – precizno i u detalje. Zato su ti elementi od vitalne važnosti za multimedijalne menije, navigacione sisteme i sadržaj.

Da bi se poboljšala priča oko ovoga svega, u aplikacije se ugrađuju odgovorajući **elektronski rečnici**. Treba proveravati i prelistavati priložene rečnike. Korisnik može da se iznenadi brojem sinonima i srodnih reči od kojih može da počne, kako bi uradio ono što treba; sigurno je da će pronaći savršenu reč za konkretnu potrebu. Najveći broj današnjih procesora reči isporučuje se sa priloženim elektronskim rečnikom.

1.2. O fontovima

Pismo (*typeface*) je osnovni pojam u tipografiji i glavno oruđe tipografije. Sastoji se od svih slovnih i ostalih znakova (znakovi interpunkcije, brojke, posebni znakovi) koji imaju karakteristiku jednoobraznosti, tj. slični su po vizuelnim i ostalim karakteristikama.

Font je pojam koji je u digitalnom dobu, pogotovo od procvata stonog izdavaštva u osamdesetim godinama dvadesetog veka postao praktično istovetan sa pojmom pismo, te danas gotovo svi kada kažu font, zapravo misle na pismo. Međutim, kroz celu istoriju tipografije, font je predstavljao sve znakove jednog pisma u jednoj veličini.

Svako pismo, dolazi u više veličina (na računarima te veličine teoretski mogu rasti gotovo do beskonačnosti), i ako se uzme, na primer, da je početna veličina 4 pt, sledeća 5 pt, i tako dalje, recimo sve do 72 pt, svi znakovi svih veličina u tom rasponu čine pismo, ali, zato skup svih znakova veličine 4 pt čini jedan font tog pisma, skup svih znakova veličine 5 pt čini drugi font tog istog pisma itd.

Iako ova podela na font i pismo ima svoj smisao i praktičnu svrhu, danas se gotovo u potpunosti izgubila. Po pravilu, kod računarske pripreme za štampu, isključivo se koristi naziv **font**.

Pre nego što se krene u praktično istraživanje fenomena tipografije, potrebno je na početku definisati šta je to uopšte **tipografija**. Jer, iako je tipografija pojam koji je gotovo svakodnevno u govoru, mnogi ljudi ne znaju tačno da definišu tipografiju i prateće elemente.

Tipografija je pojam koji dolazi od dve grčke reči *typos* – žig, pečat + *graphein* – pisati, te se definiše na razne načine – kao nauka o slovima, umetnost korišćenja tipografskih slova, veština slaganja, izrade, oblikovanja i funkcionalnog korišćenja slova. Ipak, ono najvažnije se sastoji u tome da je glavni cilj tipografije što efikasniji način ponovnog korišćenja tipografskog materijala (olovnih i digitalnih znakova).

Najkraća definicija tipografije mogla bi da glasi – **umetnost, nauka i tehnika korišćenja i izrade slova i slovno materijala**.

Tipografija ima neka svoja određena tehnička, funkcionalna i estetska pravila, ali ona se u nekim slučajevima mogu i napustiti (na primer, u savremenom grafičkom dizajnu). U svakom slučaju tipografija je jedinstven spoj umetnosti i tehnike, koja se služi naizgled jednostavnim slovima i oblicima, no dobar tipograf i dizajner će od tih znakova moći načiniti tehničko-umetnička dela jedinstvene lepote.

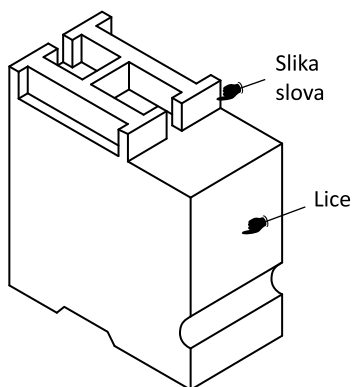
Tipografija se razvijala paralelno sa tehnološkim napretkom, te danas obuhvata širok spektar delovanja – od kaligrafije i rukopisa, preko klasičnih primena u grafičkoj industriji i grafičkom dizajnu, do korišćenja slova u veb dizajnu, korišćenju na raznim elektronskim uređajima (mobilni telefoni, razni plejeri i slično), korišćenju slova u TV produkciji, računarskim igrama i uopšteno svim aspektima u kojima se pojavljuju razni natpisi.

1.2.1. Tipografski merni sistem

Ovaj deo će korisnika možda malo zbuniti i u početku neće u potpunosti razumeti o čemu se radi. Veći deo ovde iznesenog se danas retko koristi i ovde se navodi uglavnom iz istorijskih razloga, ali ne sasvim, jer tipografski merni sistem i danas stoji u pozadini cele tipografije, a njen najvažniji pojam – **tipografska tačka**, danas se toliko široko koristi i osnova je za razumevanje tipografije.

Tipografski merni sistem bio je karakterističan, uglavnom, za olovni slog (slika 1.2). Prelaskom na ofset štampu polako izlazi iz upotrebe, međutim ima istorijsku važnost i uprkos

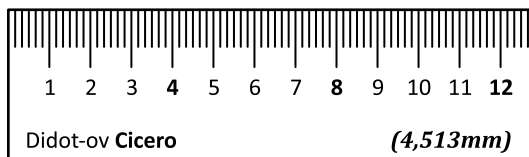
svemu i danas je na evropskom prostoru prihvaćen kao bazni sistem, što je potvrđeno primenom odgovarajućeg standarda.



Slika 1.2. Olovna slova i slog

Za početak malo istorije – prva potreba za uvođenjem jedinstvenog sistema kojim bi se mogla tačno odrediti veličina tipografskog i slagarskog materijala javlja se već u doba pronalazača savremenog štamparstva *Johannesa Guttenberga*. Prvi pokušaj standardizacije preduzeo je *Joseph Moxon* u Engleskoj 1683. godine. Ipak, prve praktične rezultate postigao je *Pierre Simon Fournier* koji je uveo tipografski merni sistem u kome je osnovna veličina **tipografska tačka** (*point typographique*) – pt. Tokom 1775. godine francuz *Francois-Ambroise Didot* sa sinom *Firminom* usavršava svoj tipografski sistem kome je osnova bila tipografska tačka koja preračunata u metrički sistem iznosi 0,3514 mm. Taj sistem je upravo gore spomenuti sistem koji je i danas prihvaćen kao evropski standard. Sistem je duodecimalan, tj. 12 tipografskih tačaka čini jedan **cicero**. Cicero je pojam koji se najčešće koristio kao standardna mera pre uvođenja standardizovanog mernog sistema.

Preračunavanje *Didot*-ovog sistema na metričku meru preduzeo je 1876. godine nemački slovovivac *Hermann Berthold*, te je na taj način utvrdio precizni tipometar čiju je dužinu od 30 cm podelio na 133 "podeoka" po 6 tipografskih tačaka, iz čega sledi da tipometar ima 798 tipografskih tačaka, odnosno $62\frac{1}{2}$ cicera.



Slika 1.3. Didot-ov tipometar

U anglosaksonskim zemljama upotrebljava se engleski tačkasti sistem u kome jedan *point* ima 0,352 mm odnosno 0.013832 inča, a engleski cicero (*pica*) ima 4,212 mm odnosno 0,1666 inča. Od 1866. godine tipografska mera se zasniva na dužini od 35 cm koja je podeljena na 166 "podeoka", odnosno 996 pt. Masovnim prodorom računara u područje

grafike počinje u novije vreme da se koristiti i merna jedinica poznata kao DTP tačka koja iznosi 0,35277 mm. Cicero, iako se danas više ne koristi tako često pod tim imenom, i u digitalno doba ostao je glavna veličina, jer je upravo najveća većina štampe na računarima ispisana upravo u veličini od 12 pt.

Iako je tipografski merni sistem standardizovan, ljudima je i dalje lakše da razmišljaju u metričkom sistemu. Zbog toga se koriste navedene formule za preračunavanje iz tipografskih u metričke jedinice i obrnuto.

Preračunavanje tipografskih tačaka u milimetre:

$$n [\text{pt}] \times \frac{3}{8} = n [\text{mm}]$$

Na primer, za 12 tipografskih tačaka:

$$12 \times \frac{3}{8} = 4,5 [\text{mm}]$$

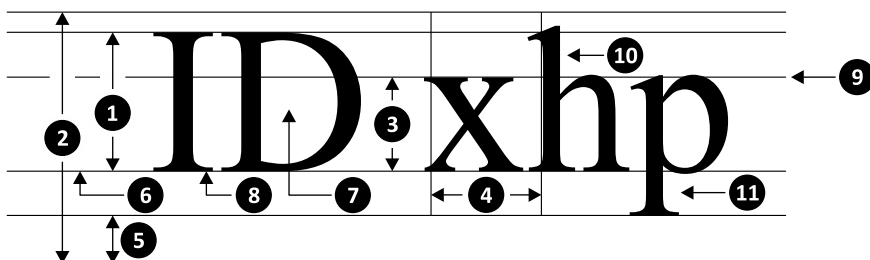
Preračunavanje milimetara u tipografske tačke:

$$n [\text{mm}] \times \frac{8}{3} = n [\text{pt}]$$

Na primer, za 6 mm

$$6 \times \frac{8}{3} = 16 [\text{pt}]$$

Veličina fonta ne opisuje tačno visinu ili širinu njegovih znakova. Razlog je činjenica da visina označena brojem 3 na slici 1.4 (visina malog slova x) u dva fonta može varirati, a visina velikih slova tih fontova je ista.

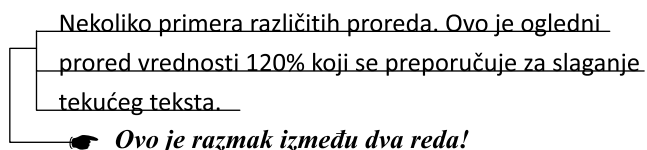


Slika 1.4. *Merenje veličine pisma*

Na slici 1.4 korišćene su sledeće oznake – *Cap Height* je visina velikog slova (1); *Point Size* predstavlja ukupnu visinu jednog reda (2); *x-Height* je visina malog slova x (3); *Set Width* predstavlja promenljivu širine predviđene za jedno slovo (4); *Shoulder* je dodatno rastojanje do naredne linije teksta (5); *Baseline* je osnovna linija po kojoj se piše tekst (6); *Counter* označava centralno slovo i brojač (7); *Serif* predstavljaju dodatne kvačice na slovima (8); *Mean Line* je glavna linija (9); *Ascender* predstavlja uzlazni potez (10); *Descender* je oznaka za silazni potez (11).

Računarski fontovi automatski dodaju prostor ispod donjeg dela slova (poneki put i iznad) da bi napravili odgovarajući razmak između redova; engleski izraz je *leading*, što vodi poreklo od tankih olovnih traka koje su tradicionalni slovoslagачi umetali između linija. Treba posebno napomenuti da mogu postojati znatne razlike između fontova.

Prored (*Leading*) može da se podešava u većini programa i na MAC OS i pod operativnim sistemom *Windows*. Obično će se naći ovo fino podešavanje u meniju *Text* programa za obradu slika ili u meniju *Paragraph* tekst procesora, mada to nije zvaničan standard. Bez obzira gde je to, korisnik mora da eksperimentiše sa ovom vrednošću da bi postigao najbolje rezultate za svoj font. Na slici 1.5 prikazano je nekoliko primera.



Sledeći primer je ovaj prored vrednosti 100% koji se, takođe, često koristi.

Nastavlja se sa proredom od 80% koji nije preporučljiv, ali u nedostatku prostora nema se drugog izbora.

I završava se sa proredom od 70% koji apsolutno nije od koristi, a ovde je prikazan čisto u edukativne svrhe.

Slika 1.5. *Definisanje veličine proreda*

Sa programima za editovanje fontova prilagođenja se, takođe, mogu praviti po horizontalnoj osi teksta – mogu se menjati metrika svakog znaka i kernovanje parova znakova, tj. selektivno menjati rastojanje između parova slova radi bolje čitljivosti i ravnomernosti razmaka između slova. **Metrika znaka** je opšta mera primenjena na pojedinačne znakove; **kernovanje** je razmak između parova znakova. Radeći sa fontovima kao što su PostScript, TrueType i Master, ali ne i sa bitmapiranim fontovima, metrika fonta se menja da bi se napravili interesantni efekti. Na primer, može se prilagoditi širina tela svakog znaka od *regular* preko *condensed* do *expanded*, kao što je prikazano na slici 1.6a.



Slika 1.6. *Podešavanje razmaka (a) i dodatno podešavanje razmaka (b)*

Pored ovoga, može da se prilagođava razmak između karaktera (*tracking*) i kernovanje između parova znakova, tj. karaktera (slika 1.6b).

Kada crta ili rasterizuje slovo **A** na ekranu ili štampanom izlazu (proces se naziva **rasterizacija**), računar mora da zna kako da prikaže slovo korišćenjem sićušnih kvadratnih

piksela (elementi slike) ili tačaka. To radi u skladu sa raspoloživim hardverom i po specifikaciji iz izbora raspoloživih pisama i fontova. Monitori visoke rezolucije i štampači mogu da kreiraju atraktivnije i raznovrsnije znakove. Današnji širok izbor softverskih fontova olakšava nalaženje pravog pisma i fonta za svačije potrebe. Na slici 1.7 nalazi se nekoliko primera istog slova prikazanog različitim fontovima.



Slika 1.7. *Isto slovo u različitim fontovima*

U vekovima kada se slagalo ručno, slova za pojedinačni font su uvek bila čuvana u dve posude ili kutije (*case*); gornja kutija je sadržala velika slova, a donja mala. Danas se na engleskom jeziku veliko slovo zove *uppercase*, a malo slovo je *lowercase*, a kod nas su odomaćeni nazivi **verzal** i **kurent**.

U nekim situacijama (na primer, za lozinke) računar pravi razliku između velikih i malih slova (*case sensitive*). U većini situacija koje zahtevaju unos sa tastature računar prepoznaje i mala i velika slova kao ista. Zbog toga se za računar kaže da je *case insensitive*, tj. "neosetljiv" na veličinu slova.

Nedavno su postali popularni nazivi kompanija i proizvoda poput *TheArt*, *FireWorks*, *PhotoDisc*, *PageMaker*, *MakeMotion*, *FileMaker* i *WebPage*. Postavljanje velikog slova u sredinu reči (*intercap*) je trend potekao od računarskog programiranja, gde su programeri otkrili da tako bolje vide reči koje upotrebljavaju za promenljive i naredbe.

1.2.2. Serif i Sans Serif

U svim slovnim znakovima mogu se pronaći neki oblici koji su više-manje zajednički. Ti oblici grade slovo i nazivaju se **elementi slovnih znakova**. Svaki znak se sastoji od elemenata, ali se ne nalaze svi elementi u svakom znaku.



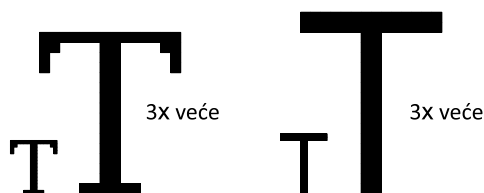
Slika 1.8. *Elementi slovnih znakova*

Pisma mogu biti opisana na mnogo načina. Pisma su bila opisivana kao ženska, muška, delikatna, formalna, kapriciozna, duhovita, komična, srećna, tehnička, novinska itd. Jedan

pristup kategorizovanju pisama je univerzalno poznat, manje se tiče lika slova, a više je reč o mehaničkim i istorijskim osobinama pisma. U ovom pristupu koriste se izrazi **serif** i **sans serif**.

Detaljno poznavanje ovih elemenata nije potrebno u praktičnom radu sa fontovima, važnije je samim dizajnerima i autorima pisama. Pri praktičnom radu najvažniji, praktično i jedini važan element je serif koji je vrlo koristan pri razlikovanju i odabiru pisama. Ipak, dobro je upoznati barem najosnovnije elemente.

Serif naspram sans serif je najjednostavniji način za kategorizaciju pisma. Pismo ili ima serife ili nema (na francuskom *sans* znači *bez*). Serif je mali ukras na kraju poteza kojim se ispisuje slovo. *Times New Roman*, *Times*, *Bookman*, *New Century Schoolbook* i *Palatino* su primeri serif-nih fontova. *Helvetica*, *Arial*, *Optima* i *Avant Garde* su sans serif-ni.



Slika 1.9. Elementi slovnih znakova – Serif i San Serif

Na štampanoj strani serif fontovi se tradicionalno koriste za osnovni tekst, jer se tvrdi da ovi fontovi pomažu u vođenju očiju čitaoca po redu ili redovima teksta. Sa druge strane, sans serif fontovi se koriste za naslove i važne izjave.

Računarski svet standardne monitorske rezolucije od 72 dpi nije isto što i svet štampe, što znači da može da se zastupa teza da su sans serif fontovi čitljiviji i privlačniji kada se koriste u malim veličinama tekstualnog polja na ekranu. Zaista, pažljiv izbor sans serif fontova dizajniranih da budu čitljivi u malim veličinama ima puno smisla kada se predstavlja velika količina teksta na ekranu. Font *Times* veličine 9pt može da izgleda "prezaposleno" i, zapravo, može da bude težak i umarajući za čitanje. Ogroman, polucrni serif font za naslov ili podnaslov može da prenese poruku elegancije i određenog stava.

Trebalo bi koristiti ono što je dobro za sistem isporučivanja, koji ne mora da bude istovremeno dobar i za štampanje materijala na papiru. Razlog je to što štampanje onoga što je kreirano na računarskom monitoru, WYSIWYG (*What You See Is What You Get*, šta vidiš to i dobijaš) više predstavlja cilj nego apsolutnu činjenicu.

1.3. Upotreba teksta u multimediji

Treba zamisliti projekat bez ikakvog teksta. Njegov sadržaj ne bi nikako mogao da bude kompleksan, a bilo bi potrebno mnogo slika i simbola kako bi se korisnici obučili da se kreću kroz projekat. Neko bi pomislio da je rešenje primena glasa i zvuka i da bi to moglo da vodi publiku, ali korisnici bi se brzo odvikli od toga – veći napor se traži pri obraćanju pažnje na izrečene reči, nego pri prelistavanju teksta.

Jedan jedini član tekstualnog menija praćen jednom akcijom (klik mišem, pritisak na taster na tastaturi ili prst uperen ka monitoru) zahteva malo obuke, a jednostavan je i neposredan. Treba koristiti tekst:

- **O čemu je uopšte reč?** – za naslove i podnaslove,
- **Gde da se ide?** – za menije,
- **Kako da se dođe do potrebne lokacije?** – za navigaciju i
- **Šta se vidi kada se tamo stigne?** – za sadržaj.

1.3.1. Dizajniranje pomoću teksta

Računarski ekrani pružaju vrlo mali radni prostor za razvoj kompleksnih ideja. U određenom trenutku korisnik će morati da isporuči "snažne" ili vrlo koncizne tekstualne poruke na ekranu u što kraćem obliku. Iz perspektive dizajna, izbor veličine fonta i broj naslova koji se postavljaju na konkretan ekran moraju imati neku vezu i sa kompleksnošću poruke i sa njenim delokrugom.

Ako su poruke deo interaktivnog projekta ili veb sajta gde se zna da će korisnik tražiti informacije, mogu se upakovati velike količine tekstualnih informacija na ekran, pre nego što to sve postane napadno. Većina korisnika voli "gusto spakovan" materijal i skrolovanje ("putovanje") kroz relevantan tekst, kao i proučavanje detalja. Tu bi morala da se nađe mera. Premalo teksta na ekranu zahteva beskonačno i iritirajuće prelistavanja stranica, nepotrebne klikove mišem i čekanja; previše teksta može da učini da ekran bude natrpan i neprijatan.

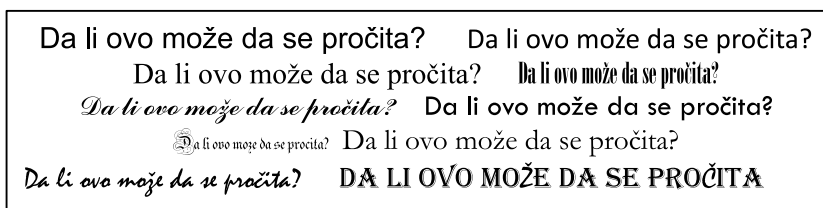
Sa druge strane, ako se daje javna podrška glasom, tekst se uključuje u "živu" prezentaciju i naglašava glavnu poruku. U takvom slučaju treba koristiti velike fontove i malo reči sa mnogo praznog prostora. Na taj način će se publika skoncentrisati na govornika na podijumu, umesto da troši vreme na čitanje poruka koje su projektovane na ekran.

1.3.2. Izbor tekstualnih fontova

Biranje fontova za upotrebu u multimedijalnoj prezentaciji može ponekad da bude teško zbog dizajna. Stvaralac multimedijalnog projekta u isto vreme mora da bude poeta, psiholog i grafički dizajner. Preporuka je da stvaralac treba da pokušava da oseti potencijalnu reakciju korisnika na ono što se dešava na ekranu. Sledi nekoliko dizajnerskih sugestija koje mogu biti od izuzetne pomoći:

- Za mala slova treba koristiti najčitljiviji raspoloživi font. Dekorativni fontovi koje je nemoguće pročitati su beskorisni (slika 1.10).
- Treba koristiti što manji broj različitih pisama u istom radu, ali trebalo bi menjati težinu i veličinu karaktera stilovima tipa **bold** i *italic*, gde god oni izgledaju dobro.
- U pasusima treba prilagoditi pored kako bi razmak između redova bio prijatan za oko. Redovi "spakovani" isuviše gusto se teško čitaju.

- Trebalo bi menjati veličinu fonta srazmerno važnosti poruke koja se isporučuje.



Slika 1.10. Isto pitanje, ista veličina (11pt) i različiti fontovi

- U naslovima sa velikim slovima trebalo bi podesiti razmak između slova (*kerning*) kako bi se stekao pravi osećaj.
- Trebalo bi istražiti efekte primene različitih boja kako bi stvaralac bio siguran da se slova ističu ili da bi bila čitljivija, a trebalo bi razmisliti i o postavljanju teksta na različite pozadine.
- Treba koristiti opciju *anti-aliasing* za tekst gde je potreban blag i stopljeni izgled za naslove i podnaslove. To odaje profesionalniji izgled. Pomenuta opcija stapa boje po ivicama slova (to se zove *dithering*) da bi se napravio mekan prelaz između slova i pozadine.
- Treba pokušavati sa upotrebom inicijala za pasuse (slika 1.11) i početnim velikim slovima kako bi se rečima obezbedila dodatna težina. Većina tekst procesora i editora teksta omogućavaju olakšanu primenu inicijala i umanjenih velikih slova (*SMALL CAPS*) u tekstu.

Trebalo bi iskoristiti inicijale za pasuse i početna velika slova kako bi se rečima “dodala” odgovarajuća težina. Trebalo bi iskoristiti inicijale za pasuse i početna velika slova kako bi se rečima “dodala” odgovarajuća težina.

Trebalo bi iskoristiti inicijale za pasuse i početna velika slova kako bi se rečima “dodala” odgovarajuća težina. Trebalo bi iskoristiti inicijale za pasuse i početna velika slova kako bi se rečima “dodala” odgovarajuća težina.

Slika 1.11. Primena inicijala na dva različita načina

- Ako se koriste centrirana slova u tekstualnom bloku, broj redova treba da bude što manji.
- Ako treba privući pažnju, stvaralac bi mogao da grafikom izmeni i modifikuje željeni tekst.
- Trebalo bi eksperimentisati sa primenom senki. Takav tekst može postati ubojitiji i imati mnogo više značenja. Na veb sajtovima osenčeni tekst i grafika na čistoj beloj pozadini dodaju dubinu strani.

- Naslove treba oivičiti sa mnogo praznog prostora. Beli prostor (*white space*) je izraz za prostrane prazne oblasti.
- Treba odabrati fontove koji izgledaju dobro za prenos poruke.
- Treba koristiti smislene reči i izraze za linkove i elemente menija.
- Tekstualni linkovi (*anchors*) na veb stranama mogu da promene težište poruke; istaknuti su bojom i podvučeni su. Trebalo bi koristiti boje za linkove na konzistentan način kroz ceo sajt i trebalo bi izbegavati "drečave" zelene, crvene ili ljubičaste boje na strani.
- Trebalo bi podebljati (**bold**) ili naglasiti tekst kako bi se istakle ideje ili koncepti, ali ne bi trebalo praviti tekst da izgleda kao link ili dugme kada on to nije.
- Na veb strani trebalo bi staviti najvažnije tekstualne elemente i elemente menija u prvih 320 piksela. Proučavanje surferskih navika otkriva da samo 10 ÷ 15% surfera prelistava veb stranu nadole.

1.3.3. Meniji za navigaciju

Interaktivni multimedijalni projekat ili veb sajt se obično sastoji iz tela informacije ili sadržaja kroz koji se korisnik kreće pritiskom na taster, klikom miša ili pritiskom na ekran sa detekcijom dodira. Najjednostavniji meniji sastoje se od tekstualne liste tema. Korisnici odaberu temu i kliknu je. Kako multimedija i grafički korisnički interfejsi (*Graphical User Interface*, GUI) postaju sveprisutni u računarskoj zajednici, izvesne intuitivne akcije se uče na sve više mesta.

Na primer, ako su na računarskom ekranu tri reči, tipičan odgovor od strane korisnika, a da računar ne postavi poseban podsticaj, je da klikne na jednu od tih reči da bi izazvao aktivnost. Ponekad su elementi menija okruženi okvirima ili napravljeni da liče na dugmad koju treba pritiskati. Da bi se prostor trošio što manje, tekst poput *Play Video* ili *Press to Quit* se često skraćuje u *Video* i *Quit*. Bez obzira na to, namera ostaje jasna korisniku.

1.3.4. Dugmad ili tasteri za interakciju

U multimediji **dugmad** su objekti, kao što su i blokovi teksta ili fotografija, koji prouzrokuju neku akciju pošto se klikne na njih. Izmišljena su samo da bi neko mogao da ih gurne ili pritisne kursorom, mišem, tasterom ili prstom – i da pokažu osobine kao što su osvetljavanje ili neki drugi vizuelni ili zvučni efekti, kako bi naznačili da je korisnik "pogodio" cilj. Na vebu tekst i grafika mogu biti dugmad. Za sada, trebalo bi zapamtiti da se pravila za pravilan izbor teksta i fontova u projektima odnose na dugmad, baš kao i na podnaslove, oznake pasusa i blokove teksta.

Pre nego što se krene u upotrebu odgovarajućeg fonta, njega mora da prepozna operativni sistem računara. Ako stvaralac multimedijalnog projekta želi da koristi neke druge fontove, a ne one koji su instalirani kad i operativni sistem, mora da ih naknadno instalira. Kada se naknadno instaliraju aplikacije, drugi fontovi obično bivaju dodati u postojeću kolekciju.

U većini autorizovanih sistema lako je napraviti sopstvenu dugmad od bitmapa ili nacrtanih objekata. U autorizovanim sistemima sa prenošenjem poruka, u kojima se skriptom opisuje aktivnost kada se dugme miša nađe iznad ili ispod objekta, može se brzo zameniti jedna bitmapa drugom, istaknutom ili osvetljenom verzijom, kako bi se pokazalo da li je dugme bilo "pritisnuto" ili da je miš iznad njega. Pravljenje sopstvenih dugmadi od bitmapa ili nacrtanih objekata daje autoru veću dizajnersku moć i slobodu i istovremeno predstavlja osiguranje od problema nedostajućih fontova. Sa druge strane, to može da zahteva i da oduzme mnogo više vremena. Interesantna tekstualna i grafička dugmad za veb mogu se praviti od GIF ili JPEG bitmapa, koje, kada se klikne na njih, vode ka drugim stranama.

1.3.5. Polja za čitanje

U startu, rad autora je već otežan kada dizajnira tekst koji treba čitati sa ekrana. Eksperimenti su pokazali da je čitanje teksta na računarskom ekranu sporije i teže od čitanja istog teksta sa papira ili iz knjige. Mnogi korisnici radije odštampaju izveštaje i elektronske poruke i čitaju ih sa papira, umesto da ih čitaju preko ekrana. Čitanje odštampanog materijala ostaje udobnije.

Ako cilj multimedijalnog projekta ili veb sajta nije da prikazuje velike blokove teksta, onda bi trebalo pokušati da se prezentira korisniku samo po nekoliko pasusa teksta po strani. Trebalo bi upotrebiti font koji se lako čita, umesto lepšeg fonta koji je nečitljiv. Trebalo bi pokušati da se prikažu celi pasusi na ekranu i treba izbegavati prekide gde korisnik mora da se "šeta" između strana da bi pročitao pasus.

1.3.6. HTML dokumenti

Jezik HTML je opisni jezik specijalno namenjen opisu veb stranica. Pomoću njega jednostavno se mogu odvojiti elementi kao što su naslovi, paragrafi, citati i slično. Pored toga, u HTML standard su ugrađeni elementi koji detaljnije opisuju sâm dokument kao što su kratak opis dokumenta, ključne reči, podaci o autoru i slično. Ovi podaci su opštepoznati kao **meta podaci** i jasno su odvojeni od sadržaja dokumenta.

U početku je bio prilično ograničen što se označavanja sadržaja tiče i pružao je uglavnom elementarne stvari za označavanje i formatiranje teksta (paragrafi, naslovi, citati itd.). Kako je veb rastao tako je rasla i potreba za bogatijim sadržajem te je u tom smeru razvijan i HTML standard. Tada su standardu dodati elementi za opis tabela, slika, slojeva, napredno formatiranje teksta itd.

Svi HTML dokumenti bi trebali da počinju sa definicijom tipa dokumenta (*Document Type Definition*, DTD) koji pretraživaču definiše po kom standardu je dokument pisan. Osnov HTML-a predstavljaju tagovi i atributi. Pomoću tagova se određeni deo dokumenta odvaja od ostatka i na njega se primenjuju pravila definisana samim tagom. Atributi se nalaze unutar tagova i omogućavaju da se pored samog imena taga i unapred definisanog ponašanja još bliže odredi način prikaza i ponašanja označenog dela dokumenta. Slede dva primera:

1. `<p>Tekst paragrafa.</p>`
2. `<p align="right">Tekst paragrafa</p>`

1.3.8. Animirani tekst

Ima mnogo načina da se zadrži pažnja posmatrača kada se prikazuje tekst. Na primer, korisnik može da animira označene pasuse i može da ih natera da "uleću" u ekran; može da povećava naslov znak po znak. Za govornike jednostavno isticanje važnog teksta dobro deluje kao sredstvo za pokazivanje. Kada treba napraviti nekoliko poenti, mogu da se stave ključne reči na gomilu i osvetljivati ih po određenom redosledu. Neke reči mogu da lete po ekranu, neke mogu da se rastapaju, rotiraju, sve dok se ne dobije dinamičan spisak reči interesantan za gledanje. Ipak, treba biti pažljiv – nema preterivanja sa specijalnim efektima ili će svi postati dosadni.



Slika 1.13. Neki primeri 3D teksta

1.4. Kodni sistemi za prikazivanje znakova

Kodiranje podrazumeva proces prikaza informacija u nekom obliku. Jezik ljudi je kodni sistem kojim se predstavljaju informacije kao nizovi leksičkih jedinica, a njih kao zvukove, odnosno, mimiku. Pisani jezik je izvedeni sistem kodiranja kojim te nizove leksičkih jedinica, zvukova i gestova predstavljaju grafičkim simbolima koji sačinjavaju neki sistem pisma.

U računarskim sistemima, kodira se pismo na način da predstavlja grafeme i druge elemente pisanog teksta kao nizove znakova, jedinica tekstualne informacije unutar nekog sistema za prikaz teksta. Ovi znakovi su u računaru prikazani na jedini način sa kojim računarski sistemi znaju baratati – binarnim brojevima. Kodni sistemi za prikaz znakova rade upravo to.

Svako kodiranje uključuje dve osnovne komponente – niz znakova i nekakav sistem za njihovo prikazivanje u obliku obradivih jedinica unutar računara. Naravno, ne postoji predodređen način na koji se ovo radi. ASCII standard je jedan od sistema, ali ne i jedini. Na taj način, isti niz brojeva može se protumačiti na beskonačno mnogo načina, zavisno od konvencija koje se pretpostavljaju. U najjednostavnijem slučaju, koji je još uvek najrašireniji,

jedan oktet bitova (bajt) odgovara jednom znaku prema nekoj kodnoj tabeli. Ovako je moguće prikazati 256 različitih znakova. Za pravilan način dekodiranja, mora biti poznato koja tabela se koristi.

Dok je ranije ASCII kodiranje bilo najučestalije, danas je raširen ISO Latin 1 način kodiranja, koji se može smatrati ekstenzijom ASCII-a. ASCII je bio dovoljan samo za prikaz teksta na američkom engleskom. Nije bio dovoljan za britanski engleski, a kamoli za izdavaštvo na engleskom govornom području ili gotovo bilo kom drugom jeziku. Ubrzo su prodrli novi standardi i to iz dva izvora – ustanova za standardizaciju i nezavisni proizvođači softvera i hardvera. Tako su firme IBM (codepage 437, 852, 855), *Apple* i *Microsoft* (codepage 1252 = Western = Latin 1) stvarali standarde usput kako su im zatrebali za proizvode koje su izdavali. Ako bi neki takav standard postao široko prihvaćen, nacionalno standardizacijsko telo ga je moglo proglasiti nacionalnim standardom, a takvi su standardi ponekad postajali i međunarodni. Uz celo mnoštvo ovako nastalih standarda, događalo se da neki standardi podržavaju određene znakove koji ne postoje u drugim.

Dešavalo se da su podržani isti setovi znakova, ali na nekompatibilan način. Ovi problemi se rešavaju *Unicode* standardom, ali nasleđeni standardi se ne mogu zanemariti.

Svaki komercijalni softverski proizvod je eksplicitno dizajniran da podržava određeni set standarda za kodiranje znakova. Sve njegove operacije sa tekstom biće izvedene pod pretpostavkom da je aktivan jedan od tih standarda. Ako podaci nisu tačno ili uopšte označeni kojim standardom su kodirani, doći će do neočekivanih rezultata.

Problem sa softverom baziranom na standardima se javlja ako je potrebno raditi sa kompletom znakova koji softver ne prepoznaje, onda se može reći da je korisnik "zapeo". Ovo je posebno bio problem za lingviste koji rade sa manje zastupljenim jezicima. Takvi onda kreiraju vlastita rešenja za svoje potrebe koja su sasvim dobra za vlastitu primenu, ali zakazuju kada dođe do potrebe za razmenom na višem nivou. Način za izbegavanje ovih problema je upotreba standardnog kodiranja koje uključuje sve znakove. Upravo takav tip rešenja nudi *Unicode* koji se razvija sa ciljem da ima univerzalni set znakova koji bi pokrivaio potrebe na svetskom nivou.

1.4.1. Kodni sistem ASCII (ISO 646)

ASCII (*American Standard Code for Information Interchange*) je oznaka za repertoar znakova, set kodiranih znakova i obrazac za kodiranje. Većina postojećih kodova sadrže ASCII kao svoj podskup u nekom smislu. Naziv ACSII se koristi naširoko i često pa se tako ponekad misli na tekst uopšteno, ili tekstualnu datoteku za razliku od binarne. ASCII po definiciji sadrži i set kontrolnih kodova, ali pravi repertoar znakova koji se sastoji od ispisivih znakova je prikazan na slici 1.14. Set kodiranih znakova definisan ASCII standardom je sledeći – kodne vrednosti su dodeljene znakovima redom koji je naveden prethodno, počevši od 32 za prazninu i završavajući sa 126 (tilda). Mesta $0 \div 31$ i 127 su rezervisana za kontrolne kodove koji imaju standardizovana imena i opise, ali im primena varira.

Obrazac za kodiranje je vrlo jednostavan i prilično očigledan za sve kodove znakova gde kodni brojevi ne prelaze 255 – svaki kodni broj je predstavljen kao oktet (bita) iste vrednosti. Okteti $128 \div 255$ se ne koriste u ASCII-u.

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00																
10																
20		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{		}	~	
80	Ç	ü	é	â	ä	à	â	ç	ê	ë	è	ï	î	ì	Ä	Å
90	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	₹	ƒ
A0	á	í	ó	ú	ñ	Ñ	ª	º	¿	¬	½	¼	í	«	»	
B0																
C0																
D0																
E0	α	β	Γ	π	Σ	σ	μ	τ	Φ	Θ	Ω	δ	∞	∅	ε	∩
F0	≡	±	≥	≤	∫	∫	÷	≈	°	•	•	√	n	2	■	

Slika 1.14. ASCII – ISO 646 proširen

1.4.2. ISO 8859 porodica

Postoji nekoliko sistema kodova znakova koji su ekstenzija ASCII-a na isti način kao što su to ISO 8859-1 i *Windows* set znakova. Sâm ISO 8859-1 je deo veće porodice znakovnih kodova ISO 8859 čiji kodovi proširuju repertoar ASCII-a na različite načine sa različitim posebnim znakovima (koji se koriste u raznim jezicima i kulturama).

ISO 8859-1 standard (koji je deo ISO 8859 porodice standarda) definiše repertoar znakova koji se naziva Latin alphabet No. 1, obično nazvan ISO Latin 1, te set kodiranih znakova. Standard specificira i obrazac za kodiranje koji je sličan onome u ASCII-u; svaki znak je predstavljen oktetom.

Kao dodatak ASCII znakovima, ISO Latin 1 sadrži različite znakove sa naglascima i druga slova potrebna za pisma zapadne Evrope i neke posebne znakove. Ovi znakovi zauzimaju mesta 160 ÷ 255 i prikazani su na slici 1.15.

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
80	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
90	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
A0		ı	ç	£	¤	¥	¦	§	¨	©	ª	«	¬	­	®	¯
B0	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C0	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D0	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E0	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F0	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Slika 1.15. ASCII – ISO 8859-1 – ISO Latin 1

Baš kao što ISO 8859-1 sadrži ASCII znakove i kolekciju znakova potrebnih za jezike zapadne (i severne) Evrope, tu je i ISO 8859-2 poznat i kao ISO Latin 2 (slika 1.16) koji je konstruisan slično za jezike srednje/istočne Evrope itd.

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
80	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
90	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
A0		À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î
B0	°	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î
C0	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D0	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E0	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F0	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Slika 1.16. ASCII – ISO 8859-2 – ISO Latin 2

ISO 8859 kodovi su izomorfni u sledećem smislu:

- kodni brojevi 0 ÷ 127 sadrže iste znakove kao i ASCII,
- mesta 128 ÷ 159 su neiskorišćena (rezervisana za kontrolne znakove),
- mesta 160 ÷ 255 su promenljivi deo koji se koristi različito u različitim izdanjima.

Iako se ISO 8859-1 koristio kao da je predodređen za kodiranje u mnogim primenama, on u principu nema neku posebnu ulogu. Očekivalo se da će ISO 8859-15 (poznat kao

ISO Latin 9 zameniti ISO 8859-1, budući da sadrži politički važan simbol za evro €), ipak se čini da nema veliku praktičnu primenu.

1.4.3. *Windows set znakova*

Tokom razvoja operativnog sistema *Windows* Američki nacionalni institut za standarde (ANSI) je razvijao standard koji je postao ISO 8859-1 "Latin". Kompanija *Microsoft* je kreirala svoju kodnu stranicu 1252 za zapadno evropske jezike na osnovi ranog nacrtu ANSI predloga i to nazvala "ANSI kodna stranica". Kodna stranica 1252 je finalizovana pre nego ISO 8859-1 i to dvoje nisu isti – 1252 je širi od ISO 8859-1. Kasnije je kompanija *Microsoft* počela da koristi ANSI u smislu bilo koje *Windows* kodne stranice, za razliku od *Unicode*. U ISO 8859-1 standardu, kodne pozicije 128 ÷ 159 su eksplicitno rezervisane za kontrolne svrhe. Takozvani *Windows set znakova* (WinLatin1 ili *Windows* kodna stranica 1252) koristi neke od tih pozicija za ispisive znakove. Korišćenje kodnih znakova pod brojevima 128 ÷ 159 u programu koji očekuje pravi ISO 8859-1 može izazvati različita ponašanja. Može se desiti da se ti znakovi ignorišu, da izgledaju kao nešto što ima smisla ili da se interpretiraju kao kontrolni znakovi.

1.4.4. *Unicode*

Unicode je standard, od strane *Unicode* konzorcijuma, koji definiše repertoar znakova i set kodiranih znakova koji su u potpunosti kompatibilni sa ISO 10646 i obrazac za kodiranje znakova.

	0020	0	0030	@	0040	P	0050	`	0060	p	0070	00A0	°	00B0	À	00C0	Ð	00D0	à	00E0	ð	00F0	
!	0021	1	0031	A	0041	Q	0051	a	0061	q	0071	i	00A1	±	00B1	Á	00C1	Ñ	00D1	á	00E1	ñ	00F1
"	0022	2	0032	B	0042	R	0052	b	0062	r	0072	ç	00A2	²	00B2	Â	00C2	Ò	00D2	â	00E2	ò	00F2
#	0023	3	0033	C	0043	S	0053	c	0063	s	0073	£	00A3	³	00B3	Ã	00C3	Ó	00D3	ã	00E3	ó	00F3
\$	0024	4	0034	D	0044	T	0054	d	0064	t	0074	¤	00A4	´	00B4	Ä	00C4	Ô	00D4	ä	00E4	ô	00F4
%	0025	5	0035	E	0045	U	0055	e	0065	u	0075	¥	00A5	µ	00B5	Å	00C5	Õ	00D5	å	00E5	õ	00F5
&	0026	6	0036	F	0046	V	0056	f	0066	v	0076	¦	00A6	¶	00B6	Æ	00C6	Ö	00D6	æ	00E6	ö	00F6
'	0027	7	0037	G	0047	W	0057	g	0067	w	0077	§	00A7	·	00B7	Ç	00C7	×	00D7	ç	00E7	÷	00F7
(	0028	8	0038	H	0048	X	0058	h	0068	x	0078	¨	00A8	¸	00B8	È	00C8	Ø	00D8	è	00E8	ø	00F8
)	0029	9	0039	I	0049	Y	0059	i	0069	y	0079	©	00A9	¹	00B9	É	00C9	Ù	00D9	é	00E9	ù	00F9
*	002A	:	003A	J	004A	Z	005A	j	006A	z	007A	ª	00AA	º	00BA	Ê	00CA	Ú	00DA	ê	00EA	ú	00FA
+	002B	;	003B	K	004B	[	005B	k	006B	{	007B	«	00AB	»	00BB	Ë	00CB	Û	00DB	ë	00EB	û	00FB
,	002C	<	003C	L	004C	\	005C	l	006C		007C	¬	00AC	¼	00BC	Ì	00CC	Ü	00DC	ì	00EC	ü	00FC
-	002D	=	003D	M	004D	]	005D	m	006D	}	007D	-	00AD	½	00BD	Í	00CD	Ý	00DD	í	00ED	ý	00FD
.	002E	>	003E	N	004E	^	005E	n	006E	~	007E	®	00AE	¾	00BE	Î	00CE	Þ	00DE	î	00EE	þ	00FE
/	002F	?	003F	O	004F	_	005F	o	006F		007F	¯	00AF	¿	00BF	Ï	00CF	ß	00DF	ï	00EF	ÿ	00FF

Slika 1.17. *Unicode*

ISO 10646 je uopšteniji i po prirodi apstraktniji, dok Unicode "nameće dodatna ograničenja na implementaciju da bi osigurao uniforman tretman znakova na svim platformama, programima i jezicima".

Unicode rešava sve one probleme oko jednoznačnosti, transformacija među različitim kodnim tabelama, raspoloživog opsega znakova koji su isticani u dosadašnjem tekstu. Ovaj standard je prihvaćen od vodećih industrijskih firmi kao što su – *Apple*, HP, IBM, *Microsoft*, *Oracle* – kao i mnoge druge. Unicode je potreban kod modernih standarda kao što su: XML, Java, JavaScript, WML itd. Podržan je od strane mnogih operativnih sistema i svih savremenih čitača.

UTF-32 kodira svaki kodni broj kao 32-bitni celi broj, tj. kao 4 bajta. Ovo je vrlo očito i jednostavno kodiranje. Sa druge strane, neefikasno je s obzirom na broj korišćenih bajtova. Ako se ima normalni engleski tekst ili neki drugi tekst koji sadrži samo znakove iz tablice ISO Latin 1, dužina kodiranog niza bajtova je 4 puta duža u Unicode-u nego u ISO 8859-1 kodu. UTF-32 se retko koristi, osim možda u internim operacijama.

Kod UTF-8 kodovi znakova manji od 128 (efektivni ASCII repertoar) se predstavljaju "kao takvi", koristeći jedan oktet za svaki kôd (znak). Svi drugi kodovi su prikazani prema relativno komplikovanoj metodi, tako da je jedan znak predstavljen kao niz od dva do četiri okteta, od kojih je svaki u rasponu $128 \div 255$. Ovo znači, da u nizu bajtova, oni koji su u rasponu $0 \div 127$ direktno predstavljaju ASCII znakove, dok bajtove u rasponu $128 \div 255$ treba interpretirati kao stvarno kodirane prezentacije znakova.

1.5. Jezici u svetu računara

U modernijim zapadnim jezicima reči se prave od simbola ili slova složenih zajedno, prezentujući kao celinu zvukove izgovorene reči. To nije tako u istočnjačkim jezicima, kao što su kineski, japanski i korejski (isto važi i za stare jezike Sumera, Egipta i Mesopotamije). U ovim jezicima ceo koncept bi mogao biti predstavljen simbolom od jedne reči, koji nema neke veze sa konkretnim fonetskim zvukom.

Slova ili simboli jezika su njegov alfabet. U engleskom alfabet se sastoji od 26 romanskih ili latinskih slova. U japanskom kanji alfabet ima više od 3.000 kana ili celih reči. Ruski alfabet, napravljen od ćiriličnih znakova zasnovanih na starom grčkom alfabetu, ima otprilike isti broj slova kao i romanski alfabet. Svi jezici imaju svoj posebni alfabet.

Najveći broj modernih alfabeta ima jedan veoma važan atribut – grafičke oblike i metode za pisanje arapskih cifara 0, 1, 2, 3, 4, 5, 6, 7, 8 i 9. To je jednostavan sistem za predstavljanje decimalnih brojeva, što omogućava lako čitanje, pisanje, manipulaciju i proračun. Izražavanje

$$16 + 32 = 48$$

je mnogo lakše u arapskim ciframa, nego u rimskim ili grčkim "brojevima":

$$\text{XVI} + \text{XXXII} = \text{XLVIII}$$

$$\iota \zeta + \lambda \beta = \mu \eta$$

Korišćenje arapske notacije se vremenom raširilo po svetu i zamenilo druge sisteme, iako se rimska notacija i danas koristi u zapadnom svetu u izvesnim formama i kontekstima.

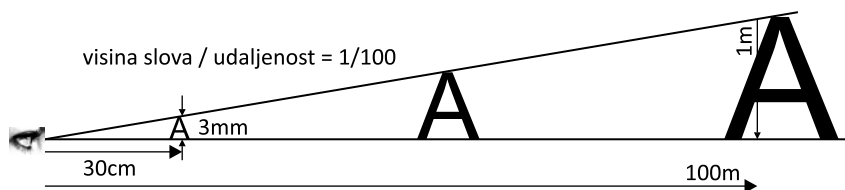
Prevođenje ili dizajniranje multimedije (ili bilo kog računarski zasnovanog materijala) u jezik koji se razlikuje od onog za koji je prvobitno pisan zove se **lokalizacija**. Taj proces se bavi svime – od redosleda mesec/dan/godina za izražavanje datuma do omogućavanja posebnih alfabetskih znakova na tastaturama i štampačima. Čak i u mnogim zapadnim jezicima kojima je romanski alfabet zajednički, postoje posebnosti i često se zahtevaju posebni znakovi da prikažu posebne zvuke.

1.6. Uređivanje teksta – pojmovi, pravila, saveti

Prelom teksta. Reč je o regulisanju prelaska u novi red. Cilj je postići podjednake razmake između reči na celoj stranici. U širem smislu pretvaranje čistog teksta u smislenu dizajniranu celinu sa svim grafičkim i ostalim elementima, na primer, knjigu.

Poravnanje (*alignment*), **isključivanje teksta.** Uređivanje teksta počinje definisanjem izgleda poravnavanja levog i desnog ruba (ivice). Tipografi tu aktivnost nazivaju "isključivanje teksta". Naša praksa najčešće dizajnira tekst razvučen na "puni format" (*justified*) sa poravnatim stupcima sa leve i desne strane. Uobičajena su još tri načina isključivanja: "u levo" – sa leve strane je poravnat tekst, a sa desne strane je nazupčena linija (neredovan kraj), "u desno" – sa leve strane je neredovan početak, a desna strana je poravnata, "u sredinu" – i sa leva i sa desna je tekst "jednako nazupčen" sa centriranim redovima. Ova tri načina najčešće su slaganja bez deljenja reči sa jednakim belinama među rečima u celom tekstu.

Čitljivost slova ili teksta. Ljudsko oko najbolje vidi slova na udaljenosti od približno $25 \div 30$ cm. Zato se ta udaljenost uzima kao daljina sa koje proveravamo da li su slova čitljiva. Najbolji odnos za čitanje slova je 1 : 100 (to je odnos udaljenosti oka od teksta i veličine slova, slika 1.18).



Slika 1.18. Najbolji odnos za čitanje slova je 1:100

Za idealnu čitljivost slovnih znakova uzima se 52 znaka po redu. Razmak između reči je slovni znak i ubraja se u pomenuti broj od 52 znaka ili karaktera.

Osnovni font/tekst (*body text*) je veličina i vrsta pisma u nekoj publikaciji kojim je rešena tipografija većine teksta. Zahteva se maksimalna čitljivost, funkcionalnost i prilagođenje sadržaju. Za knjižno čitanje sa udaljenosti oko 30 cm slova su najčešće veličine od 10 do 12 tačaka pismovnog reza bez proširenja, nisu kurzivna, nisu podebljana, nemaju nikakve efekte naglašavanja ili ukrašavanja.

Neki fontovi su jako nečitljivi (na primer, klasična *antiqua*) i zato se oni nikada ne koriste u službenim dokumentima, već se tamo koriste slova koja su izrazito čitljiva i lako prepoznatljiva – *serifi* (uspravni i još bitnije vodoravni potezi – male crtice na dnu i na vrhu fonta) vizuelno povlače crtu ispod reda teksta i time olakšavaju čitanje, odnosno praćenje linije teksta. Opšte je pravilo da u telo teksta ide *serifni* font (postoje izuzeci).

Masna (debela – *boldface*) slova nisu čitljiva ako su mala pa se koriste samo za naslove i to jako velika. Slova sa manje od 20% zacrnjenja koriste se isključivo na glatkoj površini, jer im je čitljivost smanjena.

Kurzivna pisma su manje čitljiva pa se upotrebljavaju za naglašavanje, isticanje, prekid ritma čitanja najčešće u osnovnom (*body*) fontu.

Sužena slova (*condensed*) se odnose na uže verzije fonta ili sužavanje slova. Koristi se u uređivanju teksta kada na primer korisnik želi da ugura više slova u red ili stisne veći tekst na ograničen prostor stranice.

Kontrast pozadine i slova (crna pozadina, bela slova) koristi se da se nešto jako istakne, jer oko nije naviknuto na takva slova.



Slika 1.19. Kontrasti (veliko – malo, usko – široko, geometrijsko – organsko, uspravno – nagnuto, svetlo – tamno)

Kod naslova i uopšte krupnijih slova ono što će svakako zatrebati jeste smanjivanje razmaka između krupnijih slova. Razlog ovom zbijanju nalazi se u prirodi ljudske percepcije, sa porastom veličine slova standardni razmak između njih doživljava se jače nego kod sitnijih slova. Jednostavnije rečeno, kod vrlo krupnih slova smanjivanje razmaka doprinosi čitljivosti. Takođe, u naslov treba staviti, na primer, font *Arial* ili *Chicago*, a ne, na primer, *Times* ili *Garamond*, jer su ovi potonji "lagani" fontovi i koliko god da se povećaju, oni će i dalje biti "lagani" i neće biti u stanju privući čitaocu pažnju kao *Chicago* ili *Arial* koji su poprilično "teški (masni)" fontovi.

Štampana slova (*small caps*). Sa stanovišta klasične tipografije upotreba velikih slova u osnovnom tekstu je nepoželjna pojava. Velika slova skreću pažnju od ostatka teksta tako da njihova nekritička primena ometa čitanje. Velika slova se, međutim, povremeno moraju upotrebiti za skraćenice, lična imena ili prezimena, uvodne reči u pasusu... Da bi se pojavni oblik velikih slova uskladio sa malim slovima upotrebljavaju se štampana slova ili umanjeni verzal (engleski termin je *small caps*). Štampana slova po obliku odgovaraju velikim slovima (verzalu), a po veličini malim slovima (kurentu). Ona nisu dobijena običnim smanjivanjem velikih slova jer su takve varijante *small caps* slova приметно tanje od ostatka teksta, a nema razloga da se koriste slovni oblici koji bespotrebno odudaraju od osnovnog teksta. Prava štampana slova su specijalno dizajnirani karakteri kod kojih je

debljina poteza i serifa, kao i veličina čitavih slova, proračunata tako da bude usklađena sa malim slovima. Upotreba štampanih slova je jedan od dokaza tipografske profinjenosti koja osigurava vrhunsku čitljivost teksta.

Linije. U tipografiji se linije upotrebljavaju kao okviri, za podcrtavanje, kao ukrasi i kao sigurnosne linije. Za podcrtavanje nekih važnih delova teksta se koriste ravne i valovite linije. Sigurnosne linije se koriste u službenim dokumentima. Ako se preko te linije napiše tekst, menjanjem teksta se menjaju i linije pa se zato te linije koriste za proveru autentičnosti; na primer, vlastoručnog potpisa.

Ukrasne linije su tipične za nacionalna obeležavanja. Uz njih (ili u njih) se mogu staviti i neki ukrasni elementi (zvezdice, trouglići,..).

Inicijalno slovo (*drop cap*). Reč je o početnom slovu odlomka ili paragrafa. Obično je to slovo veće ili iz drugog fonta ili je to slovo iz ukrasnog pisma. Nastoji se naglasiti početak novog sadržaja.

Predznak (*bullet*). Reč je o tački, crtici, kvadratiću ili drugom posebnom karakteru koji se stavlja na početak reda, služi za nabranje, naglašavanje novog podnaslova, naglašavanje sledećeg reda sa izuzetnim sadržajem.

Uniformnost dizajniranja dokumenta postiže se upotrebom fontova iz iste porodice. Pojedine tipografske kuće ili korporacije naručuju vlastitu familiju fonta. I pojedine grupe dizajnera komuniciraju na način da definišu porodicu fonta kao "tipografija".

1.7. Hipermedija i hipertekst

Multimedija (kombinacija teksta, slike i audio elemenata u jedinstvenu kolekciju ili prezentaciju) postaje interaktivna multimedija kada se korisniku dodeli neka kontrola nad tim koja se informacija gleda i kada se gleda. Interaktivna multimedija postaje **hipermedija** kada njen dizajner da strukturu povezanih elemenata kroz koje se korisnik kreće i sa kojima je u interaktivnoj vezi.

Kada hipermedijski projekat sadrži velike količine tekstualnog ili simboličkog sadržaja, taj sadržaj može biti indeksiran, a njegovi elementi povezani da bi se omogućilo brzo elektronsko vraćanje povezanih informacija. Kada je reč indeksirana ili povezana odgovorajućim ključem sa drugim rečima, onda je reč o **sistemu hiperteksta**. Tekstualni deo ovog izraza predstavlja sadržaj i značenje projekta, a ne grafičku prezentaciju teksta. World Wide Web nije ništa drugo nego hipertekst.

Kada tekst "oživi" u računaru umesto na štampanim stranama, mogućnost računara da brzo i efikasno obrađuje može da se upotrebi tako da tekst postane pristupačniji i smisljeniji. Tekst, onda, može da se nazove **hipertekst**. Pošto su reči, sekcije i misli povezane, korisnik može da se kreće kroz tekst na nelinearan način, brzo i intuitivno.

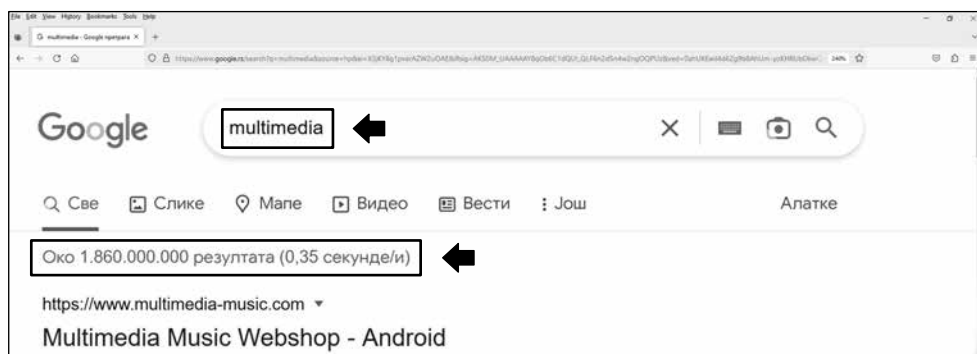
Koristeći sisteme hiperteksta, korisnik može elektronski da pretraži ceo tekst knjige na računaru, da pronađe reference na neku reč i da odmah vidi stranu gde je reč pronađena. Pored ovoga, može da pravi komplikovana logička pretraživanja (koristeći Bulove izraze kao što su AND, OR, NOT i BOTH) da bi se otkrio položaj povezanih reči. Treba napo-

menuti da mogu i celi dokumenti da se "vezuju" na druge dokumente. Pošto hipertekst predstavlja organizovanje unakrsnog povezivanja reči ne samo sa drugim rečima, nego i na srodne slike, video spotove, zvukove i ostale vrste prikaza, hipertekst često postaje dodatna osobina sveukupnog multimedijalnog dizajna. Kada se interakcija i unakrsno povezivanje dodaju multimediji, a navigacioni sistem je nelinearan, multimedija postaje hipermedija.

Moć ovakvih sistema za pretraživanje koju daje računar za ogromne količine podataka je velika, ali mora biti kanalisana na svrsishodne načine. Veze između reči i grupa informacija moraju biti dizajnirane da imaju smisla. Mora se zauzeti stav o vezama i načinu na koji je informacioni sadržaj organizovan i raspoloživ za korisnike. Filtere, kroz koje se ogromne količine podataka propuštaju, moraju "prizemljiti" i oblikovati oni koji dizajniraju sistem.

Multimedijalni dizajner upravlja mehanizmima za filtriranje u multimedijalnom projektu. Na primer, proizvođač koji predstavlja svoje proizvode kroz interaktivnu multimediju može predati mnoštvo informacija i prodajne "elemente" u ruke korisnika, uključujući pozadinske informacije, odgovarajući marketinški materijal, cenovne statistike i tehničke podatke. Dizajn projekta će, naravno, biti pristrasan – prodati što više proizvođačevih proizvoda i generisati više profita; ta pristrasnost se podrazumeva i razume u ovakvim okolnostima. Kada se pretpostavke u bilo kojoj informaciji pokažu kao pogrešne, kada se fiktivni ili nekompletni podaci prikazuju kao činjenice, prednosti i snaga multimedije i hipermedije mogu imati svoj najpogubniji uticaj.

Napravljeni su posebni programi za upravljanje informacijama i hipertekstom da bi se elektronski tekst, slike i drugi elementi predstavili u obliku baza podataka. Napravljeni su komercijalni sistemi velike i komplikovane mešavine teksta i slika – na primer, detaljan priručnik za opravke za avion Boeing 747, katalog delova za turbine Pratt & Whitney i elektronski referentni leksikoni za pravnike i biblioteke. Takve mašine za pretraživanje se uveliko koriste na vebu, gde softverski roboti posećuju milione veb stranica i indeksiraju kompletne sajtove. Hipertekstualne baze podataka zasnivaju se na sopstvenim indeksnim sistemima koji pažljivo pretražuje sav tekst i prave veoma brze unakrsno-referentne indekse koji pokazuju na položaj konkretnih reči, dokumenata i slika. Hipertekstualni indeks može biti veći za 50 do 100% od veličine originalnog dokumenta. Indeksi su od suštinske važnosti za brzo pretraživanje. Mašina za pretraživanje *Google* pravi 1.860.000.000 pogodaka (slika 1.20) za 0,35 sekundi!



Слика 1.20. *Pretraživač Google i upit "multimedia"*

Pretraga po rečima

Iako dizajner multimedijalne baze podataka pravi pretpostavke, on će dati korisniku i alate i adekvatan interfejs da proveri te pretpostavke. Pomoću interfejsa korisnici mogu da pretražuju baze po rečima da bi našli vrlo specifične kombinacije. Slede tipični metodi za pretragu po rečima u hipertekstualnom sistemu:

- **Kategorije.** Biranje ili ograničavanje dokumenata, stranica, polja teksta u okviru kojih treba tražiti reč ili reči.
- **Veza između reči.** Pretraživanje po rečima po njihovoj opštoj sličnosti i redosledu; na primer, korisnik može da traži *party* (zabava) i *bear* (pivo) samo kada su na istoj strani ili u istom pasusu.
- **Susedstvo.** Traženje reči koje su jedna iza druge, obično u izrazima ili ličnim imenima – na primer, pronaći pauka *widow* (udovica) samo ako je pre nje reč *black* (crna).
- **Naizmeničnost.** Primena kriterijuma OR da se traže dve ili više reči, kao što su *bacon* (slanina) i *eggs* (jaja).
- **Pridruženost.** Primena kriterijuma AND da se traže dve ili više reči, kao što su *skiff* (čamac jednosed), *tender* (izviđački brod) i *boat* (čamac).
- **Negacija.** Primena kriterijuma NOT da se isključivo traže reference na reč koja nije povezana sa njom – na primer, pronaći sva pojavljivanja reči *paste* kada reč *library* nije prisutna u istoj rečenici.
- **Skraćenje.** Traženje reči sa bilo kojim od mogućih sufiksa – na primer, da bi se pronašla sva pojavljivanja reči *girl* i *girls* može se zadati nešto nalik na *girl#*. Višestruki znakovni sufiksi se mogu nalaziti pomoću još jednog znaka, tako da *geo** može dati *geo*, *geology* i *geometry*, ali i *George*.
- **Umetnute reči.** Pretraživanje po rečima koje se nalaze između delova što bi normalno bile susedne reči – na primer, srednje ime ili inicijal u ličnom imenu.
- **Frekvencija.** Pretraživanje po traženim rečima, zasnovano na tome koliko se često pojavljuju – što se više puta izraz pomene u tekstu, to je dokument relevantniji za ovaj izraz.

Glava 2

Slike

Ono što se vidi na ekranu multimedijalnog računara u bilo kom trenutku predstavlja mešavinu teksta, simbola, fotografskih bitmapa, vektorske grafike, trodimenzionalnih prikaza, različitih dugmadi koju treba kliknuti i prozora pokretnog videa. Neki delovi ove slike se mogu pomerati ili kretati tako da ekran nikad ne izgleda mirno i izaziva oči. Računarski ekran sadrži mnogo više od poruke i predstavlja vezu gledaoca sa sadržajima projekta.

Grafički elementi se, obično, mogu skalirati na različite veličine, obojiti, išarati ili učiniti providnim, mogu biti ispred ili iza ostalih objekata, ili se naredbama mogu učiniti vidljivim ili nevidljivim. Kako se štampaju ti elementi, kako se biraju boje i fontovi, kojim trikovima se skreće pažnja, koliko je vešt u korišćenju alata – to su oznake veštine korisnika, talenta, znanja i kreativnosti, spojene u važnu vizuelnu vezu sa gledaocima.

2.1. Kreiranje statičnih (nepokretnih) slika

Statične (nepokretne) slike mogu biti male ili velike, čak i preko celog ekrana. Mogu biti obojene, mogu se postaviti na ekran na krajnje proizvoljan način, mogu biti ravnomernih ili čudnih geometrijskih oblika. U bilo kom obliku, statične slike se na računaru prave na dva načina: kao bitmape (**rasterska grafika**) i kao vektori (**vektorska grafika**).

Bitmape se koriste za fotorealistične slike i za kompleksne crteže koji zahtevaju fine detalje. Vektorski objekti se koriste za linije, okvire, krugove, poligone i druge matematičke oblike koji mogu matematički biti izraženi u uglovima, koordinatama i rastojanjima ili dužinama. Iscrtni objekat može biti ispunjen bojom i mustrom (*pattern*) i može da se markira (izabere) kao jedinstven objekat. Izgled obe vrste slika zavisi od rezolucije prikaza i mogućnosti grafičkog hardvera i monitora računara. Obe vrste slika se pamte u različitim formatima podataka i mogu se prevoditi iz jedne aplikacije u drugu ili sa jedne računarske platforme na drugu. Obično se slikovne datoteke komprimuju (sabijaju), radi uštede memorije i prostora na disku; mnogi formati slika već koriste kompresiju u okviru same datoteke – na primer, GIF, JPEG i PNG.

Statične slike mogu da budu najvažniji elemenat multimedijalnog projekta. Ako korisnik sâm dizajnira multimediju, trebalo bi da se postavi u ulogu crtača ili prelamača. Trebalo

bi utrošiti vreme da se nauče svi trikovi koje sadrži softver za crtanje. Kompetentnost i računarska opismenjenost u korišćenju grafičkog dizajna su od vitalnog značaja za uspeh projekta.

2.1.1. Bitmape

Bit je najjednostavniji element u digitalnom svetu i predstavlja elektronsku cifru koja može da ima vrednosti *on* ili *off*, može bude crna ili bela boja ili *true* (1) ili *false* (2). Referenciran je kao **binarni** sistem, jer samo dva stanja (*on* ili *off*) su na raspolaganju. **Mapa** predstavlja dvodimenzionalnu matricu bitova. To znači da bitmapa predstavlja jednostavnu matricu malih tačaka koje formiraju sliku, koja se ili prikazuje na monitoru ili se štampa.

Jednodimenziona matrica (*1-bit depth*) se koristi za prikaz monohromatskih slika (bitmapa) gde je, uobičajeno, svaki bit podešen da bude ili crn ili beo. U zavisnosti od korišćenog softvera, bilo koje dve boje mogu prezentuju stanja *on* ili *off* (1 ili 0). Potrebno je mnogo više podataka kako bi se opisala siva skala ili više od 16 miliona boja koje ima svaki element slike u boji.

Ovi elementi slike (poznati kao **pikseli**) mogu da imaju vrednosti *on* ili *off* unutar 1-bitne bitmape ili treba više bitova kako bi se oni opisali, ako prezentuju odgovarajuće nijanse boje (4 bita za 16, 8 bitova za 256, 16 bitova za 65.536 i 24 bita za 16.772.216 boja).

Zajedno, stanje svih piksela na računarskom ekranu sačinjava sliku koju gledalac vidi, bilo u kombinacijama crnog i belog ili obojenih piksela u liniji teksta, slici koja liči na fotografiju ili na jednostavan obrazac (šablon) pozadine.

Odakle dolaze bitmape? Kako se prave? Korisnik može da uradi sledeće:

- može da napravi bitmapu od nule programom za crtanje ili bojenje;
- može da "uhvati" bitmapu sa aktivnog računarskog ekrana programom za hvatanje ekrana (*Screen Capture*) i da je premesti u program;
- može da "uhvati" bitmapu sa fotografije, crteža ili televizijske slike korišćenjem skenera ili uređaja za "hvatanje videa" koji digitalizuje sliku i koja je, nakon toga, na raspolaganju korisniku za dalju upotrebu ili uređenje.

Kada se jednom napravi, bitmapa može da se kopira, menja, šalje elektronskom poštom ili da se koristi na mnoge kreativne načine.

2.1.2. Softver za bitmape

Mogućnosti i sposobnost programa za editovanje slika i za *Macintosh* i za *Windows* su u rasponu od jednostavnih do kompleksnih. Bolji editori postoje u verzijama koje se izvršavaju i izgledaju isto na obe platforme, a grafičke datoteke koje se pravi mogu da budu snimljene u mnogim formatima, i ti formati su čitljivi na obe platforme.

Macintosh se ne isporučuje sa alatom za bojenje, a *Windows* daje samo program *Paint*, mada za ozbiljnije poslove korisnik mora da nabavi dodatni softver za obradu slika. Većina multimedijalnih autorskih sistema nudi mogućnosti za editovanje bitmapa.

Za fotorealizam i dobijanje realističnih slika, trebalo bi skenirati fotografiju ili upotrebiti digitalnu kameru, pa, nakon toga, editivati slike u programima *Photoshop*, *Fireworks* ili nekom drugom programu za editovanje slika. Bukvalno je nemoguće "naslikati" realističnu fotografiju korišćenjem programa za bojenje i editovanje.

Alatke za umetnike i umetničko slikanje pružaju programi koji imaju u sebi na stotine vrsta četkica, sprejeva, mastila, vodenih boja i tekstura i koji omogućavaju da se to sve primeni u cilju kreiranja odgovarajuće bitmape.

Korisnik može da koristi editor slika da pravi originalne slike, kao što su stripovi, simboli, dugmad, bitmapiran tekst i apstraktne slike koje imaju rafiniran "grafički" izgled. Mnogi dizajneri kreiraju svoja rešenja primenom odgovarajućih programa za vektorsko crtanje, kao što su *CorelDRAW* ili *Adobe Illustrator*, i tako kreirana vektorska rešenja pretvaraju u bitmape.

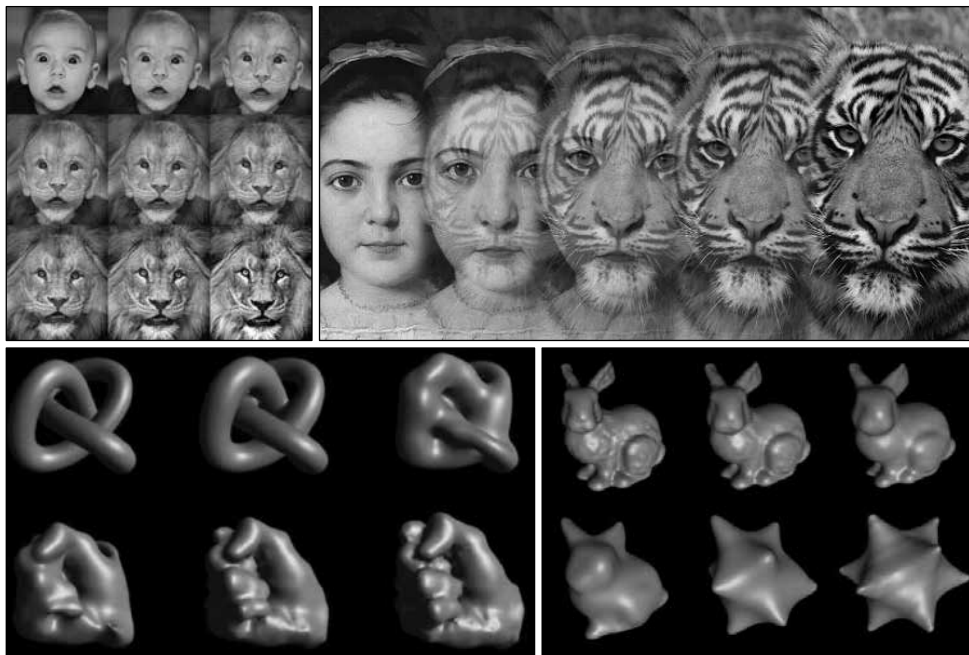
2.1.3. "Hvatanje" i editovanje slika

Slika koja se vidi na monitoru je digitalna bitmapa smeštena u video memoriji, osvežena na svakih 1/60 sekunde ili brže, zavisno od frekvencije skeniranja monitora. Dok korisnik skuplja slike za multimedijalni projekat, može često da ima potrebu da "hvata" (*capture*) i pamti sliku direktno sa ekrana monitora. Najjednostavniji način da se "uhvati" ono što se vidi na ekranu u nekom trenutku je da se pritisnu odgovarajući tasteri (ili kombinacija tastera) na tastaturi računara. To prouzrokuje konverziju iz ekranskog bafera u format koji može da se koristi.

- Radna okruženja i na Macintoshu i pod operativnim sistemom Windows imaju *Clipboard* – oblast u memoriji gde se podaci, kao što su tekst i slike, privremeno čuvaju kada se vrši isecanje (*Cut*) ili kopiranje (*Copy*) u nekoj aplikaciji. Unutar operativnog sistema *Windows*, kada se pritisne taster *Print Screen* ili *PrtScr*, primerak slike ekrana ide u klipbord. Iz klipborda korisnik može da prenese bitmapu u aplikaciju (kao što je *Paint*, koja dolazi uz *Windows*).
- Na *Macintoshu* kombinacija tastera *Command-Shift-3* pravi čitljivu datoteku pod nazivom *Picture*, u formatu *PNG* i smešta je u osnovni direktorijum aktivnog diska. Nakon toga, kreirana datoteka može da se importuje u multimedijalni sistem ili program za editovanje i bojenje. Pritiskom na taster *Command-Control-Shift-4* korisnik može da iscrta (definiše) pravougaonik na ekranu, "hvatajući" njegovu sadržinu i stavljaajući je u klipbord, za premeštanje u neki drugi program.
- Pomoćni programi za "hvatanje ekrana" (*Screen Capture*) za *Macintosh* i *Windows* idu korak dalje i neizostavni su za multimedijalne umetnike. Pritiskom na definisani taster omogućava se izbor oblasti ekrana i snimanje u različitim formatima.

Osim što dozvoljavaju uvećavanje i pravljenje složenih slika, alati za editovanje slika omogućavaju i izmenu i distorziju slika. Elementima na fotografiji u boji može da bude promenjena boja, mogu da se premeštaju, rastežu, deformišu i sve to u cilju kreiranja efektnijih slika. **Morfiranje** (*Morphing*) je još jedan efekat u nizu koji može da se koristi

za manipulaciju nepokretnim slikama ili da se prave interesantne, a često i čudne animirane transformacije (slika 2.1). Morfiranje omogućava glatko stapanje dve slike, tako da izgleda kao da se jedna slika utapa u sledeću, što često daje zabavne rezultate.



Slika 2.1. *Višetruki primeri morfiranja*

2.1.4. Skeniranje slika

Svakodnevni objekti mogu da se skeniraju i unutar programa za editovanje slika može da se manipuliše njima, da se naprave neobični efekti koji privlače pažnju. Treba otvoriti skeniranu sliku u programu za editovanje slika i eksperimentisati sa različitim filtrima, kontrastom i različitim specijalnim efektima. Treba biti kreativan, ne treba se plašiti da se probaju čudne kombinacije – poneki put greške daju najinteresantnije rezultate.

2.1.5. Vektorska grafika

Većina multimedijalnih sistema omogućava upotrebu vektorski nacrtanih objekata kao što su linije, pravougaonici, elipse, poligoni, kompleksni objekti kreirani od jednostavnijih i tekst.

- Programi za dizajn (projektovanje) pomoću računara (CAD) tradicionalno koriste vektorski nacrtane objektne sisteme da se naprave veoma kompleksni i geometrijski prikazi (renderi) potrebni arhitektama i inženjerima.
- Grafički umetnici koji dizajniraju za štampane medije koriste vektorski nacrtane objekte, zato što ista matematika koja stavlja pravougaonik na ekran može da

postavi taj pravougaonik (ili Bezierove krive pomoću dobre linijske ilustracije) na papir, bez "reckavosti". To zahteva veću rezoluciju štampača, upotrebom jezika za opis stranice, kao što je PostScript.

- Programi za 3D animaciju, takođe, koriste vektorsku grafiku. Na primer, različite promene u položaju, rotaciji i senčenju svetlosti koje se zahtevaju za obrtanje složenijeg sistema moraju da budu matematički izračunati.

Kako funkcioniše vektorska grafika (vektorsko crtanje)?

Vektor je linija opisana položajem svoje dve krajnje tačke. Kvadrat, na primer, može da se definiše kao:

```
RECT 0,0,200,200
```

Upotrebom Dekartovih koordinata, softver će nacrtati ovaj kvadrat, počevši od gornjeg levog ugla ekrana, 200 piksela horizontalno u desnu stranu i 200 piksela na dole. Za sledeću sintaksu:

```
RECT 0,0,200,200,RED,BLUE
```

softver će nacrtati isti kvadrat sa crvenom konturnom linijom i ispunio bi kvadrat plavom bojom. Korisnik uvek može da doda druge parametre kako bi opisao, na primer, šaru ili teksturu za ispunu zatvorene konture ili širinu granične linije.

Dekartove koordinate predstavljaju par brojeva koje opisuju tačku u dvodimenzionalnom prostoru (2D) i koja predstavlja presek horizontalnih i vertikalnih linija (x i y ose). Ovaj par je uređen i uvek se prikazuje u oblik (x, y) . U trodimenzionalnom prostoru (3D), treća dimenzija – dubina ili visina – definisana je z koordinatom i tačka je definisana kao (x, y, z) .

2.1.6. 3D crteži i renderovanje

Crtanje u perspektivi ili u 3D na dvodimenzionalnoj površini zahteva posebnu veštinu i talenat. Kreiranje 3D objekata koji se prikazuju na računarskom monitoru može biti izuzetno teško za dizajnere (projektante) koji su navikli da rade sa kvadratima, kružnicama i drugim ravanskim geometrijskim elementima. Postoji specijalizovani softver za prikazivanje trodimenzionalnih scena, kompletno sa usmerenim osvetljenjem i specijalnim efektima, ali korisnik mora da se pripremi za još učenja! Za put od pravljenja 3D teksta do stvaranja detaljnih šetnji kroz 3D prostor svaka aplikacija zahteva studiranje i vežbu, pre nego što korisnik postane efikasan u radu.

Proizvodne vrednosti multimedijalnih projekata su se dramatično uvećale, ali kako se kvalitet proizvodnje poboljšavao, tako su i očekivanja korisnika rasla. Ravni i bezbojni 2D ekrani više nisu dovoljni za uspešan komercijalni multimedijalni projekat. Grafika i animacija u 3D su postale opšte mesto, dajući "životnost" projektima. Na svu sreću dosta se promenilo – u oblasti gde su ranije radile samo moćne radne stanice, u današnje vreme jeftini PC računari i odličan softver su učinili 3D modelovanje pristupačnim većini multimedijalnih dizajnera. Danas su mnogi proizvodi – uključujući u tu celu priču i programe

*Carrera, form*Z* i *Cinema 4D* (slika 2.2) – označeni kao sredstva od suštinske važnosti za ilustraciju, animaciju i multimedijalnu produkciju.



Slika 2.2. 3D aplikacije omogućavaju poglede koji mogu da se prilagođavaju

Za 3D, dubina ili visina (**z dimenzija**) kocki i lopti mora biti izračunata i prikazana tako da perspektiva prikazanog objekta izgleda ispravno. Kao što je prikazano na slici 2.2, najveći broj 3D softverskih paketa daje prilagodljive poglede, tako da korisnik može da gleda svoj rad sa vrha, odozdo ili sa strane.

Ogromna količina informacija je potrebna da bi se prikazala 3D scena. **Scene** se sastoje od **objekata** koji se sami za sebe sastoje od mnoštva malih elemenata, kao što su blokovi, valjci, lopte ili kupe (opisani upotrebom matematičkih konstrukcija ili formula). Treba posebno napomenuti da što je više elemenata sadržano u objektu, to će komplikovanija biti njegova struktura, biće glatkiji, a njegova rezolucija finija.

Objekti i elementi u 3D prostoru nose sa sobom **svojstva** ili **osobine** kao što su oblik, boja, tekstura, senčenje i lokacija. Scena sadrži mnogo različitih objekata i korisnik može da uveća prikaz određenog dela scene kako bi video detalje. Objekti su sastavljeni od različitih blokova i pravougaonika i kreiraju se **modelovanjem** u kojem se koriste 3D aplikacije.

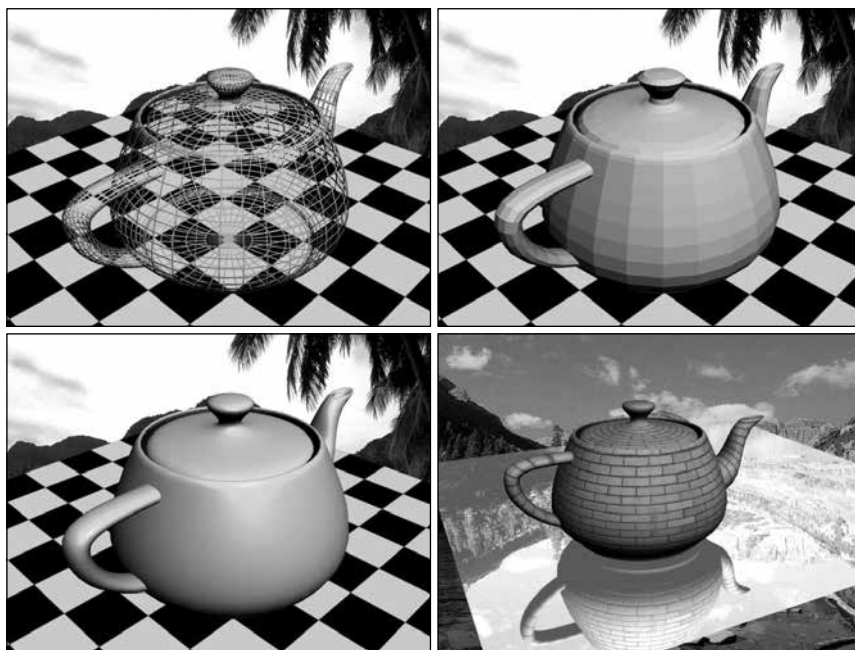
Da bi se modelovao objekat koji korisnik želi da postavi u scenu, mora da se počne sa **oblikom**. Korisnik može da kreira oblik od nule ili da uveze prethodno napravljene iz biblioteke geometrijskih oblika koji se nazivaju **primitivi** i obično je reč o blokovima, valjcima, loptama, kupama i piramidama. U većini 3D aplikacija korisnik može da napravi bilo koji 2D oblik alatom za crtanje ili postavljanjem konture, a zatim se 3D objekat dobija izvlačenjem i rotacijom u trećoj dimenziji duž ili oko z ose.

Pošto napravi 3D objekat, korisnik može da primeni na njega **teksturu** i boje da bi izgledao realnije – neravno i grubo, ili sjajno i glatko. Mogu da se primene boje, šare, ili, čak, i bit-

mapirane slike kako bi se napravila tekstura objekta. Da bi se izvršilo modelovanje scene, treba postaviti sve objekte u 3D prostor, bez obzira na njihov broj.

U modelovanu scenu korisnik može da postavi jedno ili više svetala koja će praviti difuznu ili oštru senku na objekte, a reflektovaće i blještaće tamo gde je svetlo najjače. Nakon toga, može da se doda pozadina i da se postave kamere kroz koje se gleda na finalno izrađenu scenu.

Senčenje i prikazivanje površina modela se, obično, može primeniti na nekoliko načina. Ravno senčenje (*flat shading*) je za računar najbrže i najčešće se koristi u režimu prethodnog pregleda (*Preview*). Ostala, kompleksnija senčenja (*Gouroud shading*, *Phong shading* i *Ray tracing*) traju duže, ali daju fotorealistične slike. Prikazivanje površina (slika 2.3) može da bude različito. Počev od prikazivanja žičanim modelom, preko površina koje su aproksimirane pravougaonim ili trougaonim poligonima i tim istim poligonima koji su "ispeglašani", pa do "zalepljene" teksture na željeni objekat.



Slika 2.3. *Različiti prikazi površine definisanog objekta*

Kada se kompletira modelovanje scene ili objekta u njoj, korisnik mora da prikaže finalni rezultat. Renderovanje (*rendering*) je proces kada računar konačno koristi ugrađene algoritme da primeni efekte koji su zadati za objekte koji su napravljeni.

2.2. Boje

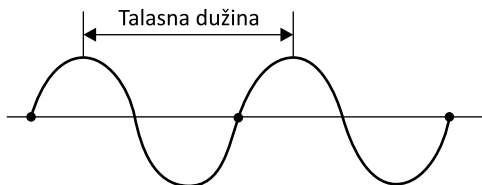
Računari su postali moćna alatka za brzo i ekonomično pravljenje slika i fotografija. U današnje vreme se više obraća pažnja na to da se sa potencijalnim kupcima komunicira

putem slika i fotografija, tj. putem primene boja. Primena boja i vizuelizacija svakog proizvoda je siguran put ka potencijalnom kupcu. To je pokazala praksa i na tome se intenzivno radi. Kao što se zna da su dizajn, grafika i profesionalna prezentacija putem fotografija bitni činioci, takođe se zna da je primena boja krucijalni deo procesa prodaje, jer je to bitan deo prilikom odlučivanja potencijalnog kupca o kupovini. Ako se boje efikasno primenjuju u proizvodnji i marketingu, onda je sigurno da će potencijalni kupci obratiti pažnju na takav proizvod. U ovom poglavlju će se obratiti pažnja na psihodinamiku boja, kao i na vezu između njih, na kulturološko značenje boja, na upotrebu realističnih paleta pseudoboja za slike koje su namenjene specifičnom auditorijumu, na strukturiranje pogleda ili prikaza za istinsko razumevanje, na modifikovanje slika za stvarni video prikaz ili štampu, na smeštanje ključnih informacija u boje ili druge vizuelne podatke, na upotrebu teksta u slikama radi prezentacije konteksta ili informacija u pozadini, kao i na vizuelnu korisničku povratnu spregu unutar grafičkih operacija.

2.2.1. Razumevanje boja

Da bi se razumelo kako se boja "meri", trebalo bi se pozabaviti fizičkim i fiziološkim osobinama boja. Boja predstavlja rezultat interakcije između svetlosti, objekta i posmatrača. Reč je o modifikovanju svetlosti na objektu, gde modifikovana svetlost dolazi do posmatrača i isporučuje mu odgovarajuću informaciju u vidu jasnih boja. Očigledno je da čovek mora, najpre, da razmotri neke stvari u vezi sa svetlošću da bi došao do odgovarajućih informacija o bojama.

Svetlost je vidljivi deo elektromagnetnog spektra. Svetlost se obično opisuje kao skup jednakih talasa. Svaki talas je definisan svojom talasnom dužinom (slika 2.4), gde talasna dužina predstavlja rastojanje od maksimuma jednog talasa do maksimuma drugog talasa. Talasne dužine se mere u milionitim delovima milimetra, nanometrима (nm).



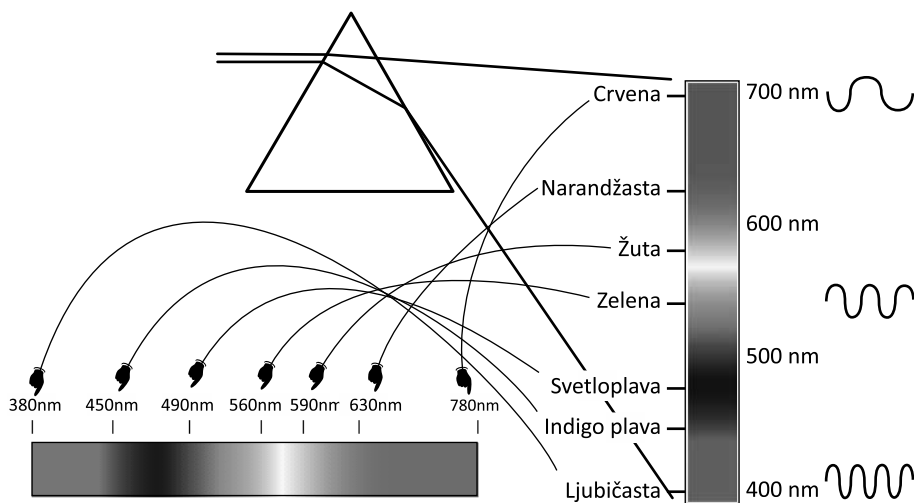
Slika 2.4. Grafički prikaz talasne dužine

Region elektromagnetskog spektra koji je vidljiv za ljudsko oko je u opsegu od 400 nm do 700 nm. Na primer, ako senzori registruju sve vidljive talasne dužine, onda to mozak shvata kao **belu** boju. Ako nema detektovanih vidljivih talasnih dužina, to znači da nema prisutne svetlosti i to mozak shvata kao **crnu** boju. Unutar celokupnog elektromagnetskog spektra postoje i talasi koji se ne vide, ali mogu da se iskoriste od strane ljudi. Na primer, mogu da se iskoriste talasi sa kratkom talasnom dužinom (X zraci) koji se koriste za rendgenske snimke, kao i talasi sa dugačkom talasnom dužinom, koji mogu da se upotrebe za radio i televiziju.

Kao što je već pomenuto, ljudsko oko registruje boje u opsegu od oko 400 nm do 700 nm, tj. tačnije vrednosti su od 380 nm do 780 nm. Do sada je zaključeno da ljudske oči i

mozak ili registruju prisustvo svetlosti (postoje talasi sa vidljivim talasnim dužinama) ili registruju da nema svetlosti uopšte (nema talasa sa vidljivim talasnim dužinama).

Kada se pusti zrak bele svetlosti kroz staklenu prizmu (slika 2.5), zrak svetlosti se razlaže i korisnik može da vidi kako oko reaguje na svaku pojedinačnu talasnu dužinu. Ovaj eksperiment dokazuje da različite talasne dužine omogućavaju da se vide različite boje. Na ovaj način mogu da se registruju dominantni opsezi crvene, narandžaste, žute, zelene, svetloplave, tamno (indigo) plave i ljubičaste boje vidljivog spektra, kao i "duge" ostalih boja između dominantnih opsega.



Slika 2.5. Vidljivi opseg elektromagnetnog spektra i dugine boje

Kada čulo vida ili vizuelni sistem detektuje talasnu dužinu vrednosti oko 700 nm, onda taj sistem vidi "crveno"; kada vizuelni sistem detektuje talasnu dužinu vrednosti koja se kreće oko 450 ÷ 500 nm, onda taj sistem vidi "plavo"; kada vizuelni sistem detektuje talasnu dužinu vrednosti oko 400 nm, onda taj sistem vidi "ljubičasto" itd. Uprošćeno posmatrano, moglo bi se reći da ljudsko može da vidi sve talasne dužine odjednom (vidi belu svetlost), ili može da vidi samo jednu talasnu dužinu. Međutim, svet boja je mnogo kompleksniji sistem od gore pomenutog, jer boja nije deo svetlosti, nego je boja svetlost.

2.2.2. RGB – osnovne aditivne boje

RGB model boja je dobio ime po početnim slovima tri osnovne boje koje učestvuju u ovom modelu: R je početno slovo od crvene boje (*Red*), G je početno slovo od zelene boje (*Green*) i B je početno slovo plave boje (*Blue*). Crvena, zelena i plava boja su osnovne (primarne) boje (osnovne boje se dodaju i mešaju kako bi se dobila željena boja i nijansa). Osnovne boje se prikazuju u 3D Dekartovom koordinatnom sistemu (slika 2.6).

Nacrtna prostorna dijagonala, od temena sa crnom bojom ka temenu sa belom bojom, predstavlja liniju gde je podjednako učešće svih osnovnih boja u željenoj nijansi, i ta dijagonala predstavlja različite nivoe sive boje. Naziv za nivoe sive boje je **siva skala**.



Slika 2.6. RGB model boja u obliku kocke

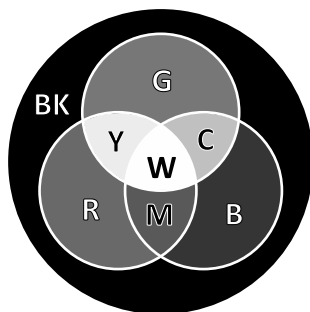
U tabeli 2.1 su prikazane RGB vrednosti za uobičajeni video test signal sa 100% amplitudom i sa 100% zasićenošću.

Tabela 2.1. RGB – 100% zasićenost boja i 100% amplituda

	Opseg	W	Y	C	G	M	R	B	BK
R	0 ÷ 255	255	255	0	0	255	255	0	0
G	0 ÷ 255	255	255	255	255	0	0	0	0
B	0 ÷ 255	255	0	255	0	255	0	255	0

RGB model boja je nadmoćan u odnosu na ostale modele boja po pitanju računarske grafike zbog toga što i monitori u boji koriste crvenu, zelenu i plavu boju kako bi napravili željenu boju. Iz tog razloga izbor RGB modela je pojednostavio arhitekturu i dizajn računara. Sistem koji je dizajniran da koristi RGB model boja je u prednosti u odnosu na druge sisteme zbog toga što se ovaj model boja nije menjao godinama.

Mešanje ovih osnovnih boja (RGB) u različitim kombinacijama dovodi do stvaranja nijansi boja koje su približne bojama u prirodi. Mešanjem crvene (R) i zelene boje (G) dobija se žuta (Y) boja, mešanjem crvene (R) i tamnoplave (B) boje dobija se ljubičasta (M) boja i mešanjem tamnoplave (B) i zelene (G) boje dobija se svetloplava (C) boja. Ako je odbijena svetlost "siromašna" primesama crvene, zelene i plave boje, onda to ljudsko oko shvata kao **belu** (W) boju. Ako svetlost nije prisutna, onda ljudsko oko to shvata kao **crnu** (BK) boju. Ove kombinacije su prikazane na slici 2.7, gde se vidi međusobni uticaj boja.



Slika 2.7. RGB – osnovne aditivne boje

Kombinovanjem dve "siromašne" osnovne boje za dodavanje (RGB) dobijaju se osnovne boje za "oduzimanje", tj. osnovne subtraktivne boje. Ove osnovne boje su cijan (C), magenta (M) i žuta (Y) i one stoje nasuprot crvenoj (R), zelenoj (G) i tamno plavoj (B) boji.

2.2.3. CMY, CMYK i PANTONE modeli boja

Ovi modeli boja bazirani su na apsorpciji svetlosti, tj. na osobini koju ima mastilo na papiru. Kako bela svetlost pada na poluprozirno mastilo, određene vidljive talasne dužine se apsorbuju, dok se ostale reflektuju nazad ka očima posmatrača. Kada se bela svetlost reflektuje sa papira i ako posmatrač vidi svetloplavu boju, to je zbog toga što svetlost apsorbuje crvenu boju i reflektuje svetloplavu (cijan). Slično tome, ljubičasta boja apsorbuje zelenu boju, a žuta apsorbuje plavu. To znači da svetloplava boja (C), ljubičasta (M) i žuta (Y) stvaraju **subtraktivni** model boja. U svakom modelu boja postoji tačka (komponenta) gde je nemoguće dodati bilo koju boju. U subtraktivnom modelu je reč o beloj boji. To znači da se željena boja dobija na taj način što se od bele boje oduzima neka boja, i zbog toga je ovaj model boja idealan za štampu na belom papiru. Ovakav model se naziva CMY model boja.

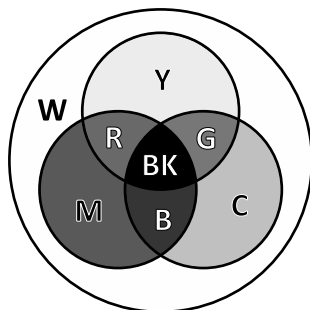
U teoriji, čiste komponente svetloplave boje (C), ljubičaste (M) i žute (Y) boje, kada se pomešaju, trebalo bi da apsorbuju svu svetlost i tako prikažu crnu boju. Međutim, sva štamparska mastila sadrže primese, tako da ova tri mastila (boje) proizvode tamnobraon nijansu umesto crne. To je dovelo do toga da mora da se uključi i crna boja u model boja kako bi ta boja figurisala u štampanom materijalu. Sa K se označava crna boja i na ovaj način se došlo do "novog" modela boja pod nazivom CMYK.

U režimu CMYK, svakom pikselu se dodeljuje procentualna vrednost svake boje. Najsvetlije boje sastoje se od malog procenta mastila (boja), dok tamnije sadrže veće procenete. Na primer, svetlocrvena boja može da sadrži 2% svetloplave boje (C), 93% ljubičaste boje (M), 90% žute boje (Y) i 0% crne boje (K). Na slikama koje su urađene u modelu boja CMYK, bela boja se dobija kada sve komponente imaju vrednost 0%.

Pored ovih modela boja, pojavljuje se i PANTONE (*Pantone Matching System*) sistem usklađivanja boja. To je standardni sistem specifikovanja boja mastila u grafičkoj umetnosti i štampanju, koji se sastoji od knjige uzoraka u kojoj je svakoj od oko 500 boja dodeljen broj. PANTONE može da se smatra međunarodnim sistemom boja koji se sastoji od 14 osnovnih boja, za dobijanje preko hiljadu različitih nijansi izvedenih boja po recepturi PANTONE uzoraka boja.

Monitori i skeneri koriste aditivne osnovne boje zbog toga što su emisijski uređaji, što znači da oni direktno dodaju R, G i B komponente svetla tamnoj pozadini. Štampači, u drugu ruku, moraju da renderuju boje na papiru, tako da oni rade sa reflektovanim svetlom. Da bi ovo uopšte moglo da se odradi, štampač mora da radi sa suptraktivnim osnovnim bojama – cijan, magentom i žutom.

Na slici 2.8 prikazane su kombinacije boja u modelima boja CMY i CMYK. Treba napomenuti da žuta (Y) boja apsorbuje tamnoplavu (B) boju, magenta (M) boja apsorbuje zelenu (G) boju i cijan (C) boja apsorbuje crvenu (R) boju.



Slika 2.8. *CMY i CMYK – osnovne subtraktivne boje*

Prevođenje CMY modela boja u RGB model boja može da se prikaže pomoću sledećeg izraza:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}.$$

2.2.4. HSV, HSI i HSL modeli boja

HSI (nijansa, zasićenost i intenzitet) i HSV (nijansa, zasićenost i vrednost) su modeli boja koji su razvijeni da bi korisnici lakše baratali bojama i ovi modeli su dizajnirani tako da aproksimiraju put kako čovekovo oko oseća i tumači boje. Vrednost H vraća korisnika u originalni spektar boja i može da ima vrednosti crvene, žute, zelene, svetloplave, plave i ljubičaste boje. Vrednost S definiše koliko se bele svetlosti sadrži u boji i ta vrednost se koristi za razlikovanje svetle boje od tamne. Ovi modeli su razvijeni u doba kada su se boje pravile "ručno", ali korisnici su navikli da ih koriste i u današnje vreme. HSL (nijansa, osvetljenost i zasićenost) je model boja sličan HSV modelu boja (samo što se koristi *osvetljenost* umesto *intenziteta*).

Razlika između HSI i HSV modela boja je u načinu izračunavanja komponente osvetljenosti ili sjajnosti (I ili V), pri čemu se određuju distribucija i dinamički opseg obe komponente (I ili V) i zasićenost (S). Model boja HSI je bolji kada se koriste tradicionalne funkcije za obradu slika kao što su savijanje, izjednačavanje, uravnotežavanje i histogrami; te funkcije manipulišu osvetljenjem, dok je parametar I zavisao od RGB komponenti. Model boja HSV omogućava manipulaciju nijansama i zasićenošću (dodavanjem ili regulisanjem količine boje) i na taj način se dolazi do većeg dinamičkog opsega zasićenosti.

HSV model boja je u obliku jednostruke šestostrane piramide. Vrh piramide predstavlja vrednost $V = 1$, što znači da je tu maksimalni intenzitet (jačina) boja. Tačka na tom mestu predstavlja belu boju. Tačka u osnovi piramide predstavlja crnu boju i njena vrednost je $V = 0$. Komplementarne boje se nalaze na suprotnim pozicijama jedna u odnosu na drugu, pod uglom od 180° . Vrednost ugla određuje osa H koja se rotira oko vertikalne V ose. Počinje se sa vrednošću 0° što je oznaka za crvenu boju. Vrednost ose S je prenosivi odnos, koji se kreće od vrednosti 0 na preseku osa S i V, pa do vrednosti 1 na stranicama piramide. Na primer, tačka sa vrednostima $S = 0$ i $V = 1$ predstavlja belu boju. Sve vrednosti na osi S mogu da se usaglase sa vrednošću $V = 0$. Sve vrednosti koje se nalaze

na osi V za vrednost $S = 0$ predstavljaju sivu skalu. Očigledno je da kada je $S = 0$, onda su vrednosti na osi H nebitne.

Sa tačke gledišta umetnika, boje sa koordinatama $V = 1$ i $S = 1$ su boje slabog intenziteta. Dodavanje bele boje se odvija smanjivanjem vrednosti S (bez promene vrednosti V), a dodavanje crne boje se vrši povećanjem vrednosti V (bez menjanja vrednosti S).

Sledi tabela 4.2 koja prikazuje parametre HSV modela boja, i to za već pominjane odnose u gornjim tabelama – 100% zasićenost boja i 75% amplituda signala.

Tabela 4.2. HSV model boja – 100% zasićenost boja i 75% amplituda

	Opseg	W	Y	C	G	M	R	B	BK
H	$0^\circ \div 360^\circ$	-	60°	180°	120°	300°	0°	240°	-
S	$0 \div 1$	0	1	1	1	1	1	1	0
V	$0 \div 1$	0,75	0,75	0,75	0,75	0,75	0,75	0,75	0

HSI model boja je u obliku dvostruke šestostrane piramide. Vrh ove piramide ima vrednost $I = 1$, što odgovara beloj boji. Najniža tačka piramide predstavlja crnu boju i tu je $I = 0$. Komplementarne boje se nalaze na suprotnim pozicijama jedna u odnosu na drugu, pod uglom od 180° .

Sledi tabela koja prikazuje parametre HSI modela boja, i to za već pominjane odnose u gornjim tabelama – 100% zasićenost boja i 75% amplituda signala.

Tabela 4.3. HSI model boja – 100% zasićenost boja i 75% amplituda

	Opseg	W	Y	C	G	M	R	B	BK
H	$0^\circ \div 360^\circ$	-	60°	180°	120°	300°	0°	240°	-
S	$0 \div 1$	0	1	1	1	1	1	1	0
V	$0 \div 1$	0,75	0,375	0,375	0,375	0,375	0,375	0,375	0

Konvertovanje CMYK u RGB

Da bi se izvršilo konvertovanje CMYK modela boja u RGB model boja, mora da se napravi međukorak. To znači da se najpre CMYK model boja konvertuje u CMY model boja, a onda se CMY model boja konvertuje u RGB model boja. Treba napomenuti da se ovde barata vektorima, što znači da se vrednosti svih komponenti svih modela boja nalaze u opsegu od 0 do 1. Da se ne bi stvorila zabuna, moraju da se uvedu sledeće oznake:

- komponente CMY modela biće označene indeksom 1, tj. CMY model će se označavati kao $C_1M_1Y_1$; i
- komponente CMYK modela biće označene indeksom 2, tj. CMYK model će se označavati kao $C_2M_2Y_2K$.

Konvertovanje CMYK modela u CMY model boja obavlja se pomoću izraza:

$$\begin{aligned}C_1 &= C_2 (1 - K) K \\M_1 &= M_2 (1 - K) K \\Y_1 &= Y_2 (1 - K) K\end{aligned}$$

Konvertovanje CMY modela u RGB model boja obavlja se pomoću sledećih izraza:

$$\begin{aligned}R &= 255 (1 - C_1) \\G &= 255 (1 - M_1) \\B &= 255 (1 - Y_1)\end{aligned}$$

Konvertovanje RGB u CMYK

Za konvertovanje RGB modela boja u CMYK model boja, takođe je potreban međukorak. To znači da se najpre RGB model boja konvertuje u CMY model boja, a onda se CMY model boja konvertuje u CMYK model boja. Treba još jednom napomenuti da se ovde barata vektorima, što znači da se vrednosti svih komponenti svih modela boja nalaze u opsegu od 0 do 1. Konvertovanje RGB modela u CMY model boja obavlja se pomoću izraza:

$$\begin{aligned}C_1 &= 1 - \frac{R}{255} \\M_1 &= 1 - \frac{G}{255} \\Y_1 &= 1 - \frac{B}{255}\end{aligned}$$

Sada sledi konvertovanje CMY modela u CMYK model boja, a to se obavlja pomoću sledećih izraza:

$$\begin{aligned}C_2 &= C_1 - (C_1, M_1, Y_1)_{\min} \\M_2 &= M_1 - (C_1, M_1, Y_1)_{\min} \\Y_2 &= Y_1 - (C_1, M_1, Y_1)_{\min} \\K &= K\end{aligned}$$

gde član $(C_1, M_1, Y_1)_{\min}$ predstavlja najmanju vrednost prikazane tri komponente i njihovih vrednosti. Na primer, ako su komponente CMY modela boja (0, 45; 0, 23; 0, 88), onda će taj član da ima vrednost:

$$(C_1, M_1, Y_1)_{\min} = 0,23$$

za sve tri komponente CMYK modela boja: C_2 , M_2 i Y_2 .

2.2.5. Još ponešto o bojama

Dubina boja pokazuje koliko boja i nijansi boja hardver i prateći softver mogu da prikazu. Uobičajeni sistemi nude 8-bitne (256 boja), 16-bitne (oko 64.000 boja) i 24-bitne (oko 16 miliona boja) dubine boja. Naravno, poneki put se dubina boja opisuje kao aktuelni broj boja koje sistem može da prikaže, kao 256 boja ili 16,7 miliona boja.

Hardver je ograničavajući faktor po pitanju boja. Internet pretraživač će iskoristiti onoliko boja koliko sistem može da upotrebi. Na primer, ako prikazanoj grafici treba 16,7 miliona boja, ali hardver podržava prikazivanje samo 256 boja, onda će pretraživač da svede broj prikazanih boja na uobičajenu paletu boja, koju diktira hardver (to često nije ugodno).

Sledeći ograničavajući faktor je tip slike koju treba prikazati. Na primer, GIF format dozvoljava da se prikaže slika sa samo 256 boja od mogućih 16,7 miliona. To je ograničenje formata, a ne slike! To znači da JPG format može da prikaže svih 16,7 miliona boja. Ovo je doprinelo tome da JPG postane najpopularniji format za prikazivanje realističnih slika.

8-bitnost boja omogućava prikazivanje 256 boja i to je minimalna mogućnost svakog sistema. Takvi su VGA sistemi i njihove mogućnosti su zato prilično ograničene. Kako se, uopšte, dolazi do broja boja? Do broja od 256 boja dolazi se na sledeći način:

$$2^8 = 256,$$

a to je posledica toga što postoji 8 bitova i 2 mogućnosti kao odgovor (ON i OFF ili 0 i 1). Sistemi sa 8-bitnim bojama daju siromašan prikaz i, uglavnom, su prevaziđeni.

16-bitnost boja omogućava prikazivanje oko 64.000 boja, što baš i nije tačan broj. Tačan broj je:

$$2^{16} = 65.536,$$

ali je odomaćen izraz od 64.000 boja ili 64 K boja. Ove boje su dobar izbor za sisteme sa ograničenim mogućnostima video memorije.

24-bitnost boja omogućava prikazivanje oko 16,7 miliona boja, što nije tačan broj. Tačan broj je:

$$2^{24} = 16.777.216,$$

ali je odomaćen izraz od 16,7 miliona boja. Ove boje su najbolje rešenje za pravljenje i prikazivanje slika na računaru. Korišćenjem 24-bitnih boja, na primer, najbolje se prikazuju gradijentni prelazi i najbolje se uklanjaju nazubljene ivice sa figure ili linije, pri radu sa grafikom.

2.3. Grafički formati

Svaka sačuavana slikovita prezentacija naziva se **grafičkom datotekom** ili **slikom**. Kada su u pitanju rasterski grafički sistemi, prikaz na displeju u boji je prezentovan u baferu kodova kao grupa piksela sa RGB vrednostima. Monohromatske slike se čuvaju u formatu bit mape (koristi se po jedan bit za svaki piksel). Uopšteno gledajući, bilo kakva rasterska prezentacija slike naziva se **rasterska datoteka**. Mnogi grafički formati su razvijeni, a razvijaju se i dan danas, kako bi se organizovale informacije za adekvatan prikaz

slike. Rasterske datoteke u "punoj" boji su velike datoteke, tako da su razvijeni neki grafički formati koji omogućavaju primenu odgovarajuće kompresije podataka kako bi se smanjila veličina datoteke, jer je manju datoteku lakše čuvati i može lakše da se "transportuje". Kao dodatak ovome, broj boja prikazanih na kolornoj slici mora da se smanjuje kada se takva slika prikazuje na sistemu sa ograničenim mogućnostima za prikazivanje boja, ili kada je slika sačuvana u formatu koji ne podržava 24 bita po pikselu. U ovom delu će se obratiti pažnja na upoznavanje sa najčešće korišćenim grafičkim formatima, kao i sa metodama za redukciju veličine i slika i broja boja koje treba prikazati.

Vrednosti boja svakog piksela unutar rasterske slike se čuvaju kao pozitivne vrednosti i te vrednosti se nalaze u opsegu koji je diktiran brojem dostupnih bitova po pikselu. Na primer, za "punu" boju (24 bita po pikselu) RGB slike, vrednost svake komponente boje smeštena je u jedan bajt, gde se vrednosti R, G i B boja nalaze u opsegu od 0 do 255. Ostali modeli boja, kao što su HSV, HSI ili HSL, koriste se u komprimovanim formatima.

2.3.1. Često korišćeni *Windows* grafički formati

Stotine i stotine formata je razvijeno za prezentaciju grafičkih podataka sa različitim kontekstom i za različite sisteme. Operativni sistemi, na primer, obično koriste određen broj specijalno dizajniranih formata sa različitim rutinama. Individualni formati su pravljeni za specifične aplikacije, kao što su aplikacije za trodimenzionalno modelovanje, za animaciju, za grafičke korisničke interfejsa, za video zapis, za vizuelizaciju načnih podataka, za slikanje i bojenje, za rad sa unakrsnim tabelama, za Internet komunikaciju, za difuzno emitovanje ili za transmisiju putem faksa. Trebalo bi pomenuti da su unutar ISO i ANSI standarda definisani određeni formati i određene metode kompresije podataka za opštu upotrebu.

Grafički format je specifičan format u kojem je datoteka sa slikom sačuvana (zapisana). Naziv datoteke se sastoji iz dva dela – naziv datoteke i ekstenzija. Format se identifikuje pomoću tri slova ekstenzije, a neki grafički formati imaju definisanu ekstenziju i sa četiri slova. Definisanjem samog formata datoteke postoji mogućnost da korisnik odredi broj bita po pikselu, kao i neke dodatne informacije.

Slike koje mogu da se koriste i obrađuju na personalnim računarima čuvaju se u različitim formatima. Različiti grafički formati su sposobni da sadrže različite "kvalitete" boja. Svaki format može da se klasifikuje prema broju "bitova po pikselu" koje taj format podržava, tj. da se klasifikuje prema "dubini boje":

- **1 bit po pikselu** upućuje na sliku sa **2 boje**;
- **4 bita po pikselu** upućuje na sliku sa **16 boja**;
- **8 bitova po pikselu** upućuje na sliku sa **256 boja**;
- **16 bitova po pikselu** upućuje na sliku sa **32.768 boja**;
- **24 bita po pikselu** upućuje na sliku sa **16.777.216 boja**.

Ovde će se pomenuti grafički formati koji se često koriste, jer je skoro nemoguće nabrojati sve grafičke formate koji postoje.

JPEG. Skraćenica JPEG potiče od engleskog naziva *Joint Photographics Experts Group*, što znači da je reč o formatu koji je formirala zajednička grupa fotografskih stručnjaka. Ako se realno pogleda, JPEG nije grafički format. nego je reč o izuzetnoj tehnici komprimovanja grafičkih podataka sa gubicima. Ova tehnika može, bez vidljivih gubitaka u kvalitetu slike za ljudsko oko, smanjiti veličinu datoteke i po nekoliko puta. To znači da JPEG pravi kompromis između kompresije i gubitaka – postiže koeficijent kompresije od 100 : 1 sa značajnim gubicima i 20 : 1 sa malim gubicima.

Očigledno je da JPEG koristi osobinu ljudskog oka da slabije uočava razlike u nijansama boje nego u intenzitetu svetlosti. Zato se ovom tehnikom najbolje komprimuje fotografija u punoj boji. Kada je fotografija "prevedena" u sivu skalu, onda su rezultati malo lošiji, jer su nijanse sive boje nijanse svetlosti. Kod crteža i kod teksta rezultati su najslabiji, jer se i pri manjem stepenu kompresije uočavaju defekti, tj. nedostaci.

Osnovni problem kod JPEG tehnike je što su oštećenja trajna i još gore je što se ti gubici akumuliraju prilikom višetrukog uređenja (editovanja) slike. Posle određenog broja uređenja jedne iste slike, slika više neće izgledati kako treba, tj. biće mnogo lošija od originala.

Zbog ovog problema ne bi trebalo uređivati (editovati) JPEG slike. Preporučuje se uređivanje slike u "sirovom" (nekomprimovanom) obliku ili slike koja se komprimuje bez gubitaka primenom odgovarajuće metode. To znači da korisnik može da se "igra" sa nekomprimovanim originalom koliko hoće, a JPEG sliku pravi posle svih izmena kada hoće da napravi malu datoteku koju treba poslati ili distribuirati.

BMP. Ovo je uobičajeni format za bitmapiranu grafiku unutar operativnog sistema *Windows*. Prilikom stvaranja ovog formata ili pretvaranja nekog drugog formata u BMP format, korisnik može da izabere "dubinu boje". Ovaj format podržava sve dubine boja i RLE algoritam za komprimovanje za slike sa 4 ili 8 bita po pikselu.

PCX. Ovaj format je razvijen unutar firme *ZSoft Corporation* za program *Paintbrush*. To je jedan od najstarijih formata za bitmapiranu grafiku, a koristi se i dan danas. To znači da nema grafičkog paketa koji ovaj format ne prepozna i ne može da ga koristi. Slike sačuvane u PCX formatu podržavaju sve dubine boja, a vrednosti piksela mogu da se specificiraju upotrebom RGB komponenti ili tabele boja. Kod PCX formata, podaci su uvek komprimovani i koriste RLE algoritam za kompresiju.

TIFF. Skraćenica TIFF potiče od engleskog naziva *Tag Image File Format*. Ovaj format je razvio konzorcijum računarskih kompanija u cilju transfera rasterskih slika između različitih aplikacija i različitih sistema. Iako je TIFF format izuzetno kompleksan, treba priznati da je krajnje prilagodljiv i da može da se prilagodi pojedinačnim aplikacijama. Trebalo bi pomenuti da ovaj format podržava i različite metode komprimovanja grafičkih podataka. Uobičajeno je da se slike u TIFF formatu koriste u stonom izdavaštvu, prilikom prikazivanja medicinskih slika, u grafičkim korisničkim interfejsima, prilikom prikazivanja satelitskih snimaka, kao i prilikom transmisije putem faksa.

PNG. Skraćenica PNG potiče od engleskog naziva *Portable Network Graphics*. Format PNG je novi format sa komprimovanim grafičkim podacima, koji može da koristi svako, bez ikakvih ograničenja. Ovaj format je predviđen da bude standardni grafički format na webu.

Kada se uporede PNG i GIF formati ima mnogo preplitanja, ali PNG format ima tri prednosti u odnosu na GIF. Te prednosti su alfa kanali (reč je o mogućnosti podešavanja nivoa transparentnosti), gama korekcija (to je mogućnost da se kontroliše koliko je slika svetla ili tamna) i 2D preplitanje (reč je o primenjenoj metodi za progresivno prikazivanje). Kada je reč o komprimovanju podataka, PNG format ima bolju kompresiju od GIF formata, ali ukupna razlika je između 5% i 25%. Format PNG nema podršku za animaciju.

GIF. Skraćenica GIF potiče od engleskog naziva *Graphics Interchange Format*. Ovaj format je jedan od starijih formata, mada se i dan danas koristi za prikazivanje jednostavnih slika na vebu. Dubina boje GIF formata se kreće od 1 do 8 bita po pikselu, što znači da ovaj format podržava maksimalno 256 boja. Slike sa GIF formatom su uvek komprimovane i primenjena je određena metoda komprimovanja grafičkih podataka.

GIF format je dobar za jednostavne crteže, crno-bele slike i za sitniji tekst. Jednostavne slike u GIF formatu se odlično komprimuju, tako da izlaz predstavljaju izuzetno mali fajlovi. To je jedan od razloga što se slike u GIF formatu dosta koriste na vebu. Ovaj format ima još neke osobine koje su razlog njegovog opstanka na tržištu, a to su transparentnost (providnost) i mogućnost animacije.

Pokazalo se da GIF format, pored statičnih, podržava i animirane slike. Dovoljno je kreirati nekoliko slika, koje posmatrane u nizu predstavljaju animaciju. To je najbolje odraditi kroz slojeve u programu *Photoshop*, a onda pomoću određenih programa formirati animaciju gde će svaki sloj biti pojedinačni kadar u animaciji. Prikazivanje svakog kadra na ekranu se naknadno podešava, tako da korisnik ima punu slobodu u kreiranju željene animacije.

TGA. Skraćenica TGA potiče od engleskog naziva *Truevision Graphics – Adapter* i ovaj format je razvijen od strane *Truevision* korporacije. Treba napomenuti da je TGA format u narodu poznat i kao *Targa* format. Format TGA se obično koristi za editovanje (uređivanje) video materijala. Ovaj format podržava sve dubine boja.

2.3.2. *Macintosh* grafički formati

Na *Macintoshu* praktično svaka aplikacija može da uveze ili izveze PICT datoteke. PICT je komplikovan, ali raznovrstan format – razvijen je kao zajednički format koji je uvek na raspolaganju korisnicima *Macintosha*. U PICT datoteci i bitmape i vektorski objekti mogu da egzistiraju jedan pored drugog. Mnogi programi za crtanje za *Macintosha*, kao što su *Illustrator* ili *Freehand*, dozvoliće uvoz bitmape, ali nemaju sposobnost da je edituju. Multimedijalni programi koji mogu da uvezu PICT slike možda neće koristiti iscrtane objekte koji su deo datoteke, ali će ih obično pretvoriti u bitmape.

2.3.3. Još neki grafički formati

Do sada je bilo reči o grafičkim formatima koje korisnik sreće svakog dana i sa kojima radi. Kako je rečeno, postoji još mnogo grafičkih formata i čovek ne može sve da ih pomeni. U ovom delu će se izvršiti nabranje još nekih formata sa osnovnim objašnjenjima. To su sledeći grafički formati koje korisnik može da sretne u radu sa odgovarajućim aplikacijama.

EPS. Reč je o skraćenici od engleskog naziva *Encapsulated PostScript*. Predstavlja standardan format datoteka za importovanje i eksportovanje postskript datoteka. Ono što je bitno za ovaj format je to da EPS datoteka može da bude "uključena" u drugu datoteku. Trebalo bi napomenuti da EPS datoteka može u sebi da sadrži kombinaciju teksta, grafike i slika.

AI. Skraćenica je od engleskog naziva programa *Adobe Illustrator*. Ovo je osnovna datoteka za crtanje unutar ovog programskog paketa i kompatibilna je sa većinom ostalih grafičkih paketa.

WMF. To je skraćenica od engleskog naziva *Windows MetaFile*. Ova oznaka predstavlja osrednji vektorski format za programe unutar operativnog sistema *Windows* i koristi se za lokalnu razmenu podataka.

CDR. Reč je o skraćenici od engleskog naziva programa *CorelDRAW*. Ovo je osnovna datoteka za crtanje unutar ovog programskog paketa i kompatibilna je sa većinom ostalih grafičkih paketa.

DWG. Ovo je grafički format programa za projektovanje *AutoCAD* firme *Autodesk*. Ovo je osnovna datoteka za crtanje unutar ovog programskog paketa i u poslednje vreme postaje kompatibilna sa većinom formata ostalih grafičkih aplikacija.

DXF. Reč je o skraćenici od engleskog naziva *Drawing eXchange Format*. DXF format je vektorski format programa za projektovanje *AutoCAD* firme *Autodesk* i možda je najpodržavaniji vektorski format u svetu danas. Većina grafičkih paketa ovaj format sadrže u sebi, tako da je moguća razmena podataka bez gubitaka pomoću ovog formata.

3DS. Predstavlja grafički format programa pod nazivom *3D Studio MAX*, koji je neprevaziđen po pitanju vizuelizacije i, eventualno, animacije. I grafički format 3DS postaje sve više kompatibilan sa novijom generacijom određene vrste grafičkih aplikacija.

Kao što je već rečeno, ovo su grafički formati koji se "sreću" u svakodnevnoj upotrebi računara i odgovarajućih grafičkih aplikacija, a o ostalim formatima drugi put više.

2.4. Licenciranje slika

Prava korišćenja slika ili grafičkih vizuala predstavljaju pravo na korišćenje takvog sadržaja, čiji je vlasnik drugo lice. Ukoliko je potrebno koristiti sliku, odnosno, grafički vizual u reklami, na veb sajtu, u članku i slično, moraju se dobiti prava korišćenja. Postoje specifični slučajevi kada navođenje autora nije neophodno; takve slike su u javnom vlasništvu i fer korišćenju, a pod okriljem su *Creative Commons* licence. Organizacija pod nazivom *Creative Commons* (CC), koja je posvećena promovisanju otvorenog pristupa intelektualnom vlasništvu, pruža korisnicima širom sveta mogućnost sticanja neophodnih besplatnih licenci za njihova autorska prava. Veb stranica ove institucije, koja je dostupna na adresi <https://creativecommons.org/>, omogućava pristup informacijama i resursima neophodnim za razumevanje i primenu ovih licenci.

Grafički vizuali, uključujući slike i fotografije, u savremenom svetu predstavljaju neodvojivi deo veb stranica, ali je ključno imati na umu da su podložni autorskim pravima. Kako bi

se izbegle potencijalne pravne posledice, neophodno je poštovati jasno definisana pravila koja regulišu njihovo korišćenje na internetu. Prilikom preuzimanja i upotrebe grafičkih vizuala sa globalne mreže, važno je da pojedinac pažljivo razmotri sledeće aspekte:

1. Ko je autor (vlasnik) grafičkog vizuala?
2. Koji je tip licence pod kojim se nalazi grafički vizual?
3. Da li su slike dozvoljene za komercijalnu ili samo za nekomercijalnu upotrebu?
4. Da li je dozvoljena izmena, tj. manipulacija nad preuzetim grafičkim vizualom?
5. Da li je neophodno navođenje i definisanje tačnih podataka o autoru (izvoru) slike?

Pravo na korišćenje adekvatnog grafičkog vizuala, kao što su slike ili fotografije, može biti ograničeno autorskim pravima koja nisu pod kontrolom korisnika. Svaki grafički vizual podleže relevantnim autorskim pravima. Na primer, postoje slike koje su javno dostupne i koje su definisane odgovarajućim licencama, kao što je *Creative Commons*. Neprikosnovo pravilo glasi – ukoliko korisnik ne poseduje pravo na korišćenje slike, zabranjeno je njeno korišćenje bez prethodne dozvole autora.

Autorima slika (fotografija i grafičkih vizuala) je omogućeno da definišu nivo dozvola za svoj autorski rad i njegovo korišćenje. Pored zadržavanja određenih autorskih prava, korisnicima širom globalne mreže se omogućuje izvršavanje određenih izmena, deljenje ili korišćenje odgovarajućeg autorskog materijala. Definisanjem nivoa licence se uspostavlja pravilo za distribuciju autorskog sadržaja na internetu. Važno je pažljivo proučiti smernice, uslove i instrukcije za svaki nivo licence, posebno ukoliko sadržaj podleže nekoj od pomenutih *Creative Commons* (CC) licenci. Nepoštovanje licenci, odnosno, kršenje jasno definisanih pravila, može rezultovati ozbiljnim sankcijama. Na primer, nepoštovanje autorskih prava i neovlašćeno korišćenje tuđeg materijala može rezultovati potpunim blokiranjem veb sajta.

Postoje različite tehnike istraživanja i moćni forenzički alati koji se koriste za otkrivanje neovlašćenog korišćenja autorskog materijala bez prethodnog odobrenja. U takvim slučajevima, vlasnik sajta može biti suočen sa pravnim merama. Prva mera koja se može preduzeti je upućivanje zahteva internet servis provajderu kod kojeg se nalazi hosting sajta za uklanjanje odgovarajućeg sadržaja, kao i podnošenje odgovarajuće tužbe protiv odgovornog lica, odnosno, vlasnika sajta.

Važno je napomenuti da svaki hosting provajder ima sopstvenu politiku korišćenja koju vlasnici veb sajtova moraju poštovati. U slučaju kršenja autorskih prava, hosting provajder zadržava potpuna prava u vezi sa preduzimanjem akcija i mera. To može uključivati upozorenja, brisanje sadržaja, privremeno obustavljanje rada veb sajta ili čak njegovo trajno brisanje.

Primeri različitih nivoa licenci su sledeći:

- CC BY – grafički vizuali koji su obuhvaćeni ovom licencom podležu nešto slobodnijem korišćenju. Za korišćenje autorskog dela označenog ovim nivoom licence, neophodno je navođenje autora.

- CC BY-NC – ova licenca zabranjuje korišćenje autorskog dela u komercijalne svrhe; dozvoljena je samo nekomercijalna upotreba.
- BY-ND – ova licenca zabranjuje bilo kakve izmene autorskog dela; obavezno je navođenje autora.

Grafička ilustracija navedenog tipa licenciranja može se videti na slici 2.9.



Slika 2.9. *Grafička intuitivna ilustracija CC licenci¹*

Preporuka jeste da se koriste slike za koje je dobijena pisana saglasnost (na primer, putem elektronske pošte), ili da se koriste samo one slike koje su nedvosmisleno označene kao potpuno besplatne za korišćenje na javnoj mreži, odnosno, internetu.

Upotreba grafičkih vizuala u komercijalne ili nekomercijalne svrhe pre svega zavisi od dozvole autora za takvu vrstu korišćenja svog autorskog dela. Često se dešava da autori svoje grafičke vizuale označe licencom CC BY-NC, koja zapravo zabranjuje korišćenje u bilo kojim komercijalnim kontekstima. Za korišćenje odgovarajućeg grafičkog vizuala u komercijalne svrhe, potrebna je pismena saglasnost autora i vlasnika autorskih prava.

Primena tehnike manipulacije grafičkog vizuala, odnosno izmena njenih delova, može podleći autorskim pravima. Licenca koja odgovara ovom nivou je CC BY-ND. Autori i vlasnici grafičkog vizuala imaju mogućnost da zabrane bilo kakve izmene. Takođe, vlasnik, tj. autor, može zahtevati da u konačnoj izmenjenoj verziji jasno stoji naznaka da je originalna slika podvrgnuta izmenama.

Precizno navođenje i definisanje relevantnih informacija o autoru (izvoru) grafičkog vizuala je imperativ i potrebno je da bude apsolutno tačno. U skladu sa vrstom vizuala, autor može zahtevati definisanje i navođenje različitih parametara, kao što su:

- Ime autora,
- Izvorni link ka originalnom grafičkom vizualu,
- Naziv licence koja je primenjena na vizual.

Kao alternativa, ukoliko korisnik grafičkog vizuala nije u potpunosti siguran u nivo licence kojoj određeni vizual pripada, mogu se koristiti servisi koji većinom nude besplatne elemente s jasno istaknutim licencama. Primeri besplatnih slika, ilustracija, vektora i drugih elemenata dostupni su na veb sajtovima, kao što su to sajtovi <https://pixabay.com/> ili <https://www.pexels.com/>. Korisnici ovih servisa moraju biti uvek svesni rizika koji se mogu javiti, jer se uslovi mogu promeniti ili se menjaju nakon određenog vremenskog perioda. Preporuka je da se korisnici pre postavljanja ili korišćenja slike na javnoj mreži detaljno upute u opšte uslove i sve odredbe koje su povezane sa slikom, kako bi se izbegle sve potencijalne pravne posledice.

¹Izvor: <https://creativecommons.org/about/cclicenses/>

U nastavku je prikazan primer sa veb sajta [pexels.com](https://www.pexels.com), gde su jasno obeležena mesta (strelice) gde se može pronaći više informacija o odgovarajućoj fotografiji i kojim pravima slika podleže. U ovom konkretnom primeru, slika je označena kao besplatna za korišćenje. Važno je naglasiti da navedeni primer ne obuhvata pravni, odnosno zakoniti status svake pojedinačne slike ili drugog grafičkog vizuala koji su dostupni putem ovih servisa. U trenutku pisanja ovog teksta, CC licenca za datu sliku bila je predstavljena kao potpuno besplatna za korišćenje.



Slika 2.10. *Ilustracija lokacije licence odabrane slike²*

²Izvor: <https://www.pexels.com/photo/woman-sitting-in-front-laptop-1181677/>

Glava 3

Zvuk

Zvuk je verovatno najuzbudljiviji deo multimedije. To je univerzalan govor na bilo kom jeziku, od šapata do zaglušujuće buke mlaznog aviona u poletanju. Pruža zadovoljstvo slušanja muzike, a može i da iznenadi senzacionalnim specijalnim efektima ili da utiče na raspoloženje iz drugog plana. Način na koji se koristi zvuk može da označi razliku između obične i profesionalne, spektakularne multimedijalne prezentacije. Pogrešna upotreba zvuka, međutim, može da upropasti projekat.

3.1. Prednosti zvuka

Kada nešto vibrira u vazduhu krećući se napred – nazad (na primer, konus zvučnika), stvaraju se talasi pritiska. Ovi talasi se šire kao nabori vode od kamička bačenog u bazen, a kada dosegnu bubne opne korisnika, korisnik će osetiti promene pritiska ili vibracije, kao zvuk. U vazduhu, talasanja se šire brzinom od $1.225 \frac{\text{km}}{\text{h}}$ ili 1 Mach na nivou mora. Zvučni talasi se menjaju zavisno od pritiska zvuka (amplituda) i frekvencije – visine zvuka. Mnoštvo zvučnih talasa spojenih u jedno čini "zvučnu oblast" simfonijske muzike, govora ili, jednostavno, buke.

Akustika je grana fizike koja proučava zvuk. Nivoi zvučnog pritiska (glasnoća ili jačina) mere se u **decibelima (dB)**; ta mera je, zapravo, proporcija između izabrane referentne tačke na logaritamskoj skali i nivoa koji se stvarno oseća. Kad se učetvorostruči snaga zvuka, to je povećanje od samo 6 dB; kada se napravi 100 puta jača buka, uvećanje u dB nije stotinu puta, nego samo 20 dB. Ova skala ima smisla, jer ljudi primaju (registruju) nivoe zvučnog pritiska kroz veoma širok dinamički raspon.

Zvuk je energija, kao talasi koji se lome o peskovitu plažu i prejak zvuk može za stalno oštetiti delikatne mehanizme za prijem iza bubnih opni, što obično smanjuje sluh na raspon od 6 kHz. U terminima jačine ono što korisnik čuje subjektivno nije i ono što čuje objektivno. Percepcija glasnosti (jačine) zavisi od frekvencije – na nižim frekvencijama zahteva se više snage kako bi se isporučila ista primljena zvučnost, kao i za zvuk u srednjim ili u višim frekventnim rasponima. Zvuk se može više osetiti, nego čuti. Na primer, kada je ambijentalni nivo buke viši od 90 dB na radnom mestu, ljudi obično prave veći broj grešaka u obavljanju osetljivih zadataka – naročito ako je u buci prisutna neka visoka frekven-

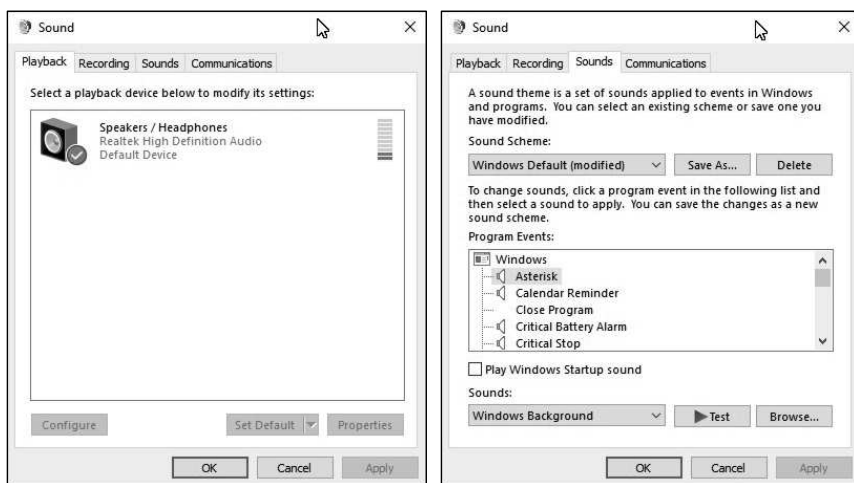
cija. Kada je nivo buke iznad 80 dB, potpuno je nemoguće koristiti telefon. Eksperimenti u naseljenim oblastima pokazali su da izvor buke od 45 dB ne proizvodi nikakvu reakciju kod suseda, od 45 do 55 dB prouzrokuje sporadične žalbe, od 50 do 60 dB žalbe na sve strane, od 55 do 65 dB pretnje da će okolina reagovati, a na više od 65 dB nastaje žestoka reakcija okoline. Ovo istraživanje je obavljeno oko 1950. godine i dan danas je koristan vodič za rok muzičare i za stvaraoce multimedije.

U akustici ima još mnogo toga, osim jačine i visine tona. Upotreba zvuka u multimedijalnim projektima neće verovatno zahtevati visokospecijalizovano poznavanje harmonika, intervala, sinusnih krivih linija, nota, oktava i fizike akustike i vibracija, ali stvaralac multimedijalnog projekta bi morao da zna:

- kako se proizvodi zvuk,
- kako se zvuk snima i menja na računaru i
- kako se zvukovi ugrađuju u multiimedijalni projekat.

3.2. Multimedijalni sistemski zvuci

Korisnik može odmah da koristi zvukove i na multimedijalnom PC-u pod operativnim sistemom *Windows* i na *Macintosh*-u, jer su bipovi i zvuci upozorenja raspoloživi čim se instalira operativni sistem. Treba otvoriti *Change System Sounds* kako bi se preslušali ovi zvuci i kako bi se postavili željeni ili napravili novi, prilagođeni zvuci (slika 3.1).



Slika 3.1. Kontrolni panel za zvuk i prateće efekte

Pod operativnim sistemom *Windows* sistemski zvuci su WAV datoteke, a nalaze se u direktorijumu *Windows > Media*. Na raspolaganju su mnogobrojni sistemski zvuci, a korisnik može da dodeli ove zvuke sistemskim događajima, kao što su startovanje operativnog sistema *Windows*, upozorenja iz drugih aplikacija ili klik van otvorenog dijaloga (što

prouzrokuje podrazumevani bip u *Windows-u*). Treba napomenuti da mogu da se prave i šeme zvukova i da se odabere posebna šema, prema raspoloženju. Korisnik može da doda svoje zvučne datoteke i da ih instalira da se aktiviraju prilikom nekog sistemskog događaja – treba postaviti WAV datoteke u direktorijum *Windows > Media* i odabrati ih unutar *Change System Sounds*.

Unutar operativnog sistema OS X na *Macintosh-u* korisnik može jedino da menja sistemski zvuk upozorenja. Korisnik može i ovde da doda svoje zvučne datoteke i da ih instalira da se aktiviraju prilikom pomenutog sistemskog događaja – treba postaviti AIF datoteke u direktorijum *Library > Sounds* i odabrati ih unutar *Sound Preference Panel*.

3.3. Digitalni audio

Digitalni audio se kreira kada se prezentuje zvučni talas brojevima – to je proces poznat pod nazivom **digitalizacija**. Korisnik može da digitalizuje zvuk sa mikrofona, sintisajzera, postojeće snimljene trake, "živog" radija i televizije, kao i sa popularnih CD ili DVD diskova. Praktično, korisnik može da digitalizuje zvuk sa bilo kog izvora, prirodnog ili veštačkog.

Digitalizovani zvuk je uzorkovani zvuk. Svakog n -tog delića sekunde **uzorak** zvuka se uzima i snima kao digitalna informacija u bitovima i bajtovima. Kvalitet ovog digitalnog zapisa zavisi od toga koliko često se uzimaju uzorci (**brzina uzorkovanja** ili frekvencija, merena u kilohercima – kHz ili hiljadama uzoraka u sekundi) i sa koliko se brojeva prezentuje vrednost svakog uzorka (**veličina uzorka**, *bitdepth*, rezolucija ili dinamički opseg). Što se češće uzima uzorak i što se više podataka zapamti o uzorku, biće finija rezolucija i veći kvalitet uhvaćenog zvuka prilikom reprodukcije. Pošto je kvalitet zvuka baziran na kvalitetu uređaja za snimanje, a ne od kvaliteta uređaja za reprodukciju, za digitalni audio se kaže je **nezavisan od uređaja**.

Tri frekvencije uzorkovanja se najčešće koriste u multimediji i to su sledeće frekvencije CD kvaliteta – 44,1 kHz, 22,05 kHz i 11,025 kHz. Veličine uzoraka su 8 ili 16 bitova. Treba napomenuti da što je veća veličina uzorka, prikupljeni podaci će bolje da opišu snimljeni zvuk. Osmobitni uzorak daje 256 jednakih jedinica za opis dinamičkog raspona (amplitude), nivo zvuka u tom trenutku, od dela "uhvaćenog" zvuka. Sa druge strane, šesnaestobitna veličina uzorka daje 65.536 jednakih jedinica za opis dinamičkog raspona. Delovi analognog talasnog oblika se uzorkuju u različitim frekvencijama, a svaki diskretni uzorak je onda zapamćen kao 8 ili kao 16 bitova podataka.

Vrednost svakog uzorka se zaokružuje na najbliži ceo broj (**kvantizacija**), a ako je amplituda veća od raspoloživih intervala, nastupa odsecanje na vrhu i dnu talasa. Kvantizacija može da napravi neželjenu pozadinsku buku, a odsecanje može oštro da izobliči zvuk.

Priprema digitalnih audio datoteka

Priprema digitalnih audio datoteka je prilično jednostavna. Ako korisnik ima analogne izvore materijala – muziku ili zvučne efekte koje je snimio na analognom mediju poput kasete – prvi korak je da se digitalizuje analogni materijal snimanjem u digitalni medij

koji računar može da pročita. U većini slučajeva to prosto znači reprodukovanje zvuka sa jednog uređaja, kao što je kasetofon, pravo u računar, upotrebom odgovarajućeg softvera za audio digitalizaciju.

Treba se skoncentrisati na dva bitna aspekta pripreme digitalnih audio datoteka:

- Balansiranje potrebe za visokim kvalitetom u odnosu na prateću veličinu datoteke. Bolji kvalitet, obično, znači da je i veća datoteka, koja zahteva duže vreme da se "skine" sa interneta i više prostora da se smesti na CD ili DVD.
- Postavljanje pogodnih nivoa snimanja da se dobije dobar, čist snimak.

Velicina datoteke i kvalitet

Mera uzorkovanja određuje frekvenciju snimka. Uzorkovanje na višim nivoima (kao što su 44,1 kHz ili 22,05 kHz) sigurnije hvata više frekvencije zvuka. **Audio rezolucija** (8-bitna ili 16-bitna) određuje preciznost sa kojom se zvuk može digitalizovati. Upotreba većeg broja bitova za veličinu uzorka daje snimak koji više zvuči kao original.

Tabela 3.1. *Snimanje digitalnog audia u trajanju od jednog minuta na uobičajenim frekvencijama i rezolucijama*

FU	Rezol.	St/Mo	Bajt/min	Komentar
44,1 kHz	16-bitna	Stereo	10,5Mb	CD kvalitet snimanja; prepoznavajući standard audio kvaliteta
44,1 kHz	16-bitna	Mono	5,25Mb	Dobra zamena za snimke visokog kvaliteta iz mono izvora kao što je glas naratora
44,1 kHz	8-bitna	Stereo	5,25Mb	Postiže najviši kvalitet reprodukcije na slabijim uređajima kao što su zvučne kartice na <i>Windows</i> računarima
44,1 kHz	8-bitna	Mono	2,6Mb	Odgovarajuća zamena za snimanje mono izvora
22,05 kHz	16-bitna	Stereo	5,25Mb	Preporučeno je za projekte za CD-ROM
22,05 kHz	16-bitna	Mono	2,5Mb	Solidna varijanta za govor, ali bolje je da se smanji vernost, a dobije prostora na disku smanjivanjem na 8 bita
22,05 kHz	8-bitna	Stereo	2,6Mb	Dobar izbor za stereo snimke solidnog kvaliteta
22,05 kHz	8-bitna	Mono	1,3Mb	Slično kao kada se sluša TV, a skoro svi računari mogu da reprodukuju ovu vrstu datoteke
11 kHz	8-bitna	Stereo	1,3Mb	Nema koristi od upotrebe sterea na niskoj frekvenciji
11 kHz	8-bitna	Mono	650Kb	U praksi niže ne treba ići, a da rezultati budu i dalje upotrebljivi
5,5 kHz	8-bitna	Stereo	650Kb	Stereo ne radi
5,5 kHz	8-bitna	Mono	325Kb	Dobro koliko i loša telefonska veza

Stereo snimci su realističniji, zato što ljudi imaju dva uвета. Mono snimci su dobri, ali zvuče "ravno" i neinteresantno kada se porede sa stereo snimcima. Logično, stereo snimci zahtevaju dva puta više prostora, nego mono datoteke za isto vreme reprodukcije.

U tabeli 3.1 date su neke česte frekvencije uzoraka i rezolucije, sa pripadajućim veličinama datoteka.

Evo formula za određivanje veličine (u bajtovima) digitalnog snimka. Za mono snimak:

$$VEL \text{ (bajt)} = FU \times TS \text{ (sek)} \times \frac{R(\text{bit})}{8} \times 1$$

gde su sledeće oznake u gornjoj formuli: FU - frekvencija uzorkovanja, TS(sek) - trajanje snimka u sekundama i R(bit) - rezolucija u bitovima.

Za stereo snimak:

$$VEL \text{ (bajt)} = FU \times TS \text{ (sek)} \times \frac{R(\text{bit})}{8} \times 2$$

Formula za snimak od 10 sekundi na 22,05 kHz u osmobitnoj rezoluciji glasi:

$$VEL = 22.050 \times 10 \times \frac{8}{8} \times 1 = 220.500 \text{ bajtova}$$

Formula za snimak od 10 sekundi na 44,1 kHz u šesnaestobitnoj rezoluciji glasi:

$$VEL = 44.100 \times 10 \times \frac{16}{8} \times 2 = 1.764.000 \text{ bajtova}$$

Formula za mono snimak od 40 sekundi na 11 kHz u osmobitnoj rezoluciji glasi:

$$VEL = 11.000 \times 40 \times \frac{8}{8} \times 1 = 440.000 \text{ bajtova}$$

Za potrošački nivo audio CD diskovi se snimaju kao stereo sa uzorkovanjem od 44,1 kHz i 16-bitnom rezolucijom.

Uređivanje digitalnih snimaka

Kada je snimak načinjen, gotovo je sigurno da će morati da se uređuje (edituje). Na slici 3.2 prikazan je program *Audacity* koji je besplatan i može da radi na operativnim sistemima *Windows*, *Macintosh* i *Linux*. Ovim alatom može da se prave zvučni snimci i da se vrše digitalni miksevi. Sledi opis osnovnih operacija uređenja koje su potrebne većini multimedijalnih stvaralaca.



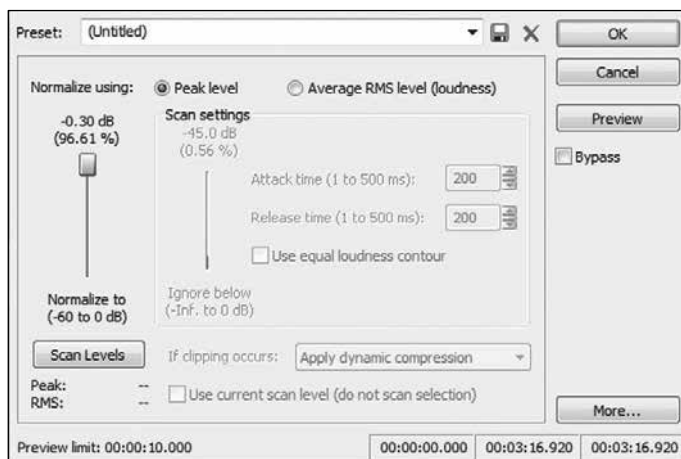
Slika 3.2. Osnovni ekran programa Audacity

Višestruke staze. Omogućeno je editovanje i kombinovanje više staza (za zvučne efekte, glas naratora, muziku, itd) i njihovo spajanje u staze i eksportovanje kao "finalni miks" u jednu audio datoteku i to je izuzetno važno.

Trimovanje ili odsecanje. Uklanjanje praznog prostora sa početka i svakog nepotrebnog dodatnog vremena sa kraja snimka je prvi zadatak u editovanju zvuka. Odsecanje čak i malog broja sekundi može da napravi ogromne razlike u veličini datoteka. Odsecanje se, obično, postiže prevlačenjem kursora miša preko grafičke prezentacije snimka i izborom komandi iz menija (na primer, *Cut*, *Clear*, *Erase* ili *Silence*).

Preplitanje i sastavljanje. Korišćenjem istih alata kao i za trimovanje, moguće je odstraniti dodatnu buku koja neminovno ulazi u snimak. Čak i u najkontrolisanim studijskim uslovima ponekad će biti potrebna ovakva intervencija. Osim toga, možda će biti potrebno da se sastavi duži snimak isecanjem i lepljenjem mnoštva malih snimaka.

Podešavanje jačine. Ako korisnik pokuša da sastavi 10 različitih snimaka u jedan, male su šanse da će svi segmenti biti iste jačine. Da bi se dobio konzistentan nivo jačine, treba označiti sve podatke u datoteci i povišiti ili sniziti ukupnu jačinu za izvestan iznos. Najbolje je upotrebiti zvučni editor za normalizovanje sastavljene audio datoteke na određeni nivo; na primer, 80 do 90% maksimuma (bez trimovanja) ili na oko -16 dB. Bez normalizacije finalni snimak će se reprodukovati ili isuviše jako ili isuviše slabo. Trebalo bi napomenuti da ni profesionalci ne mogu da izostave ovaj važan korak. Na slici 3.3 prikazan je proces normalizovanja u programu *Sound Forge*.



Slika 3.3. Normalizovanje nivoa zvuka u audio datoteci

Konverzija formata. U nekim slučajevima softver za digitalno editovanje može da čita format različit od onog koji je potreban za prezentaciju ili autorski program. Najveći broj programa za editovanje zvuka će sačuvati datoteke u željenom formatu koji multimedijalni sistem može da pročita.

Ponovno uzorkovanje. Ako je korisnik snimio i editovao zvuk na 16-bitnim frekvencijama, ali koristi niže frekvencije i rezolucije u projektu, morao bi ponovo da uzorkuje datoteku. Taj proces će uštedeti znatne količine prostora na disku.

Fade-in i Fade-out. Najveći broj programa nudi mogućnost obavljanja zvuka, što je korisno za duge sekcije za koje korisnik želi da se lagano pojačavaju (*fade-in*) ili smanjuju (*fade-out*) po jačini. Obavljanje je vrlo važno da se "ispeglažu" sâm početak i kraj zvučne datoteke.

Izjednačavanje. Neki programi nude **digitalno izjednačavanje (EQ)** kojim se menja frekvencija sadržaja, tako da snimak zvuči "svetlije" ili "tamnije".

Vremensko razvlačenje. Napredniji programi omogućavaju izmenu (u vremenu) zvučne datoteke, bez promene visine tona. To može biti vrlo korisno, ali treba paziti da većina algoritama za širenje može ozbiljno da degradira audio kvalitet datoteke ako se dužina promeni za više od nekoliko procenata u svakom pravcu.

Digitalna obrada signala. Neki programi omogućavaju obradu signala specijalnim efektima. Mogućnost da se zvučni izvor nadogradi efektima može mnogo da doprinese projektu. Stvaranje okoline stavljanjem zvuka u neku sobu, hol ili u katedralu može da unese dubinu i da doda dimenziju projektu. Ali, ne sme se preterivati sa zvučnim efektima! Kada se obradi i izmiksuje, snimak ne može dalje da se menja; uvek treba snimiti original da opet može da se menja ako korisnik nije zadovoljan.

Invertovanje zvuka. Još jedna jednostavna manipulacija je da se invertuje sve ili deo digitalnog audio snimka. Zvuci, naročito izgovoreni dijalozi, mogu napraviti nadrealne efekte kada se reprodukuju unatrag.

3.4. Kreiranje MIDI audia

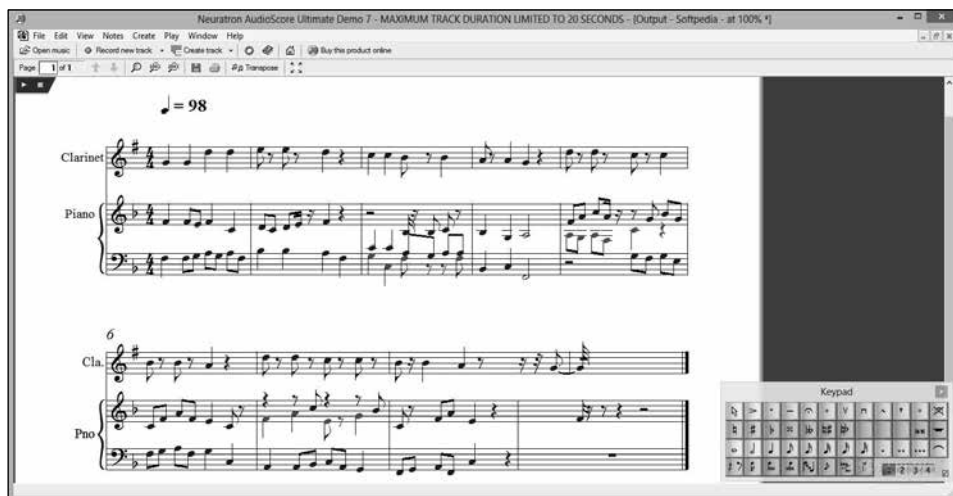
Komponovanje originalne muzike može biti jedan od najkreativnijih i najzahvalnijih aspekata pravljenja multimedijalnog projekta, a MIDI (*Musical Instrument Digital Interface*) je najbrži, najlakši i najfleksibilniji alat za to. Ipak, pravljenje originalne MIDI partiture je težak posao. Poznavanje elemenata muzike, sposobnost da se svira neki instrument i mnogo dobrih ideja su tek preduslovi za pravljenje dobre partiture.

Da bi se napravila MIDI kompozicija, potreban je **softver za sekvenciranje i sintetizator zvuka** (obično je ugrađen u zvučnu karticu PC računara, ali isto tako i dodatna kartica). **MIDI tastatura** je, takođe, izuzetno korisna, jer pojednostavljuje komponovanje, mada nije neophodna za reprodukciju i funkcioniše i ako ne postoji ugrađen sopstveni sistem za sekvenciranje. Softver za sekvenciranje omogućava snimanje i menjanje MIDI podataka i kvantifikovanje nota kako bi se ispravile vremenske neusaglašenosti. Softver snima akcije korisnika na MIDI tastaturi (ili na nekom drugom MIDI uređaju) u realnom vremenu i reprodukuje baš ono što je odsvirano na tastaturi; softver, takođe, može da odštampa lepo sređenu kopiju nota na papir.

Mnoge aplikacije za sekvenciranje omogućavaju štampanje MIDI nota na papiru. Program *AudioScore* nije samo program za sekvenciranje – može da se koristi i u suprotnom smeru, što znači da će korišćenjem skenera prepoznati notaciju i pretvoriti note u MIDI datoteke u računaru (slika 3.4).

Pravo mesto za MIDI u multimedijalnom radu može da bude kao proizvodni alat, umesto medija za disribuciju. MIDI je najbolji način da se kreira originalna muzika za multimedij-

jalne projekte, pa ga treba koristiti za dobijanje na fleksibilnosti i kreativnoj kontroli. Kada je muzika kompletirana i odgovara projektu, trebalo bi je zaključati radi distribucije pretvaranjem u digitalne audio podatke.



Slika 3.4. Program AudioScore pretvara note u MIDI datoteke

3.5. Formati za audio datoteke

Kada se pravi multimedija, korisnik će morati da se pozabavi formatima datoteka i prevodiocima za tekst, zvuk, slike, animacije i digitalne video spotove. Format zvučne datoteke je poznata metodologija za organizovanje bitova i bajtova digitalizovanog zvuka u datoteku podataka. Struktura datoteke mora biti poznata, naravno, pre nego što podaci budu snimljeni i kasnije učitani u računar da bi bili menjani i/ili odsvirani kao zvuk.

Fenomen digitalnih audio formata koji su nastali za upotrebu na računarima tu je manje od desetak godina. Uprkos tome, uspio je iz temelja da izmeni sve aspekte muzičke kulture. Popularnost MP3 datoteka je neosporna. Ovaj format je brzo osvojio Internet zajednicu zbog male veličine datoteka i zbog dobrog kvaliteta zvuka.

3.5.1. Vrste audio formata

Kao što je to slučaj sa mnogobrojnim grafičkim formatima koji definišu na koji način se zapisuju i čuvaju slike, formati audio datoteka koriste razne načine za skladištenje podataka. Korisnik najčešće može da se sretne sa tri vrste audio datoteka: WAV, MP3 i WMA.

Razlog za relativno velik broj različitih formata za zapis zvuka leži u činjenici da je zvuk memorijski vrlo zahtevan. Skladištenje informacija o nekom zvuku zahteva velike količine podataka. eki formati datoteka, poput WAV formata unutar operativnog sistema *Windows*, ne koriste nikakvo komprimovanje (sabijanje), pa su datoteke velike i neprimenjive za upotrebu na Internetu ili za slanje elektronskom poštom.

Drugi formati datoteka koriste komprimovanje ili sabijanje. Sabijanje može biti izvedeno na način da nema gubitka informacija (*lossless* sabijanje). Format datoteka koji nude ovakve algoritme sabijanja uvek imaju manju uspešnost, pa su datoteke veće. Prednost takvih datoteka je taj da je zvuk uvek moguće rekonstruisati u originalnom obliku. Danas su najpopularniji formati koji koriste tehniku sabijanja zvuka sa "gubitkom nekih informacija". Prilikom takvog sabijanja ne čuvaju se svi podaci o zvuku, već algoritam za sabijanje odbacuje manje važne podatke koji predstavljaju zvukove koje ljudsko uvo ne bi moglo da čuje. Konačna veličina takvih datoteka mnogo je manja u odnosu na nekomprimovane datoteke (ponekad i do 20 puta). Primer takvog zapisa je MP3.

Svaki tip računarske platforme i operativnog sistema koristi svoj jedinstven format audio zapisa za reprodukciju i snimanje. Neki od najpoznatijih audio formata koji se koriste na Internetu su: WMA, MP3 itd. Sledi spisak audio formata koji se najčešće sreću:

- MIDI (.MID) – MIDI format koji je nezavisan od platforme;
- Wave (.WAV) – format za digitalni audio unutar operativnog sistema *Windows*;
- Audio Interchange File (.AIF) – format za digitalni audio unutar operativnog sistema *Macintosh-a*;
- Sun Audio (.AU) – format za digitalni audio firme *Sun Microsystems*;
- Audio CD (.CDA) – format zvuka na audio CD-u;
- Windows Media Audio codec (.WMA) – komprimovan format za digitalni audio unutar operativnog sistema *Windows*;
- MPEG Audio Layer 3 (.MP3) – komprimovan format za digitalni audio koji je sve više u upotrebi.

WAV (skraćenica od engleske reči *wave* – talas) je matični format zvuka u operativnom sistemu *Windows*. Sve *Windows* aplikacije koje imaju ikakve veze sa zvukom znaju da rade sa WAV datotekama. Nekomprimovane WAV datoteke mogu da budu vrlo velike. Za jednu minutu muzike u WAV formatu, smeštene u CD kvalitetu, biće više od 10 Mb podataka na disku. Za skladištenje jednog muzičkog CD-a na hard disku u WAV formatu bi bilo potrebno oko 600 Mb. Čak i uz današnje kapacitete hard diskova ovo je vrlo mnogo.

MP3 je kraći naziv za algoritam kodiranja zvuka čije je puno ime *MPEG-1 Layer 3*. Skraćenica MPEG dolazi od *Moving Pictures Experts Group* što označava "grupu stručnjaka za pokretne slike", koja je deo Međunarodne organizacije za standarde ISO.

Jedna od najvećih prednosti formata MP3 nad ostalim formatima muzičkih datoteka jeste u tome što korisnici mogu sami da odaberu stepen komprimovanja (sabijanja). Logika je jasna – veći stepen sabijanja daje manju MP3 datoteku, ali i lošiji kvalitet zvuka, dok manji stepen sabijanja stvara veću datoteku, ali i vernije odgovara originalnom zvučnom zapisu iz kojeg je stvoren. Merilo kvaliteta MP3 zapisa je *bit rate*, dakle količina bitova kojom je predstavljena jedna sekunda zvučnog zapisa. Veći *bit rate* znači da se koristi manji stepen sabijanja i obrnuto.

Osnovne karakteristike MP3 formata i glavni razlog njegove popularnosti su velika kompresija uz očuvanje kvaliteta zvuka. Ipak, MP3 format koristi algoritam koji odstranjuje deo informacija iz ulaza sa ciljem smanjenja veličine datoteke. Ovaj gubitak je, međutim, neprimetan, budući da MP3 algoritam odstranjuje, uglavnom, delove koje ljudsko uvo ne registruje i ne primećuje. Na ovaj način, MP3 omogućava veliku kompresiju uz (naizgled) neprimetan gubitak kvaliteta zvuka.

WMA. Firma *Microsoft* suočena sa očitim nedostacima WAV datoteka, razvila je vlastiti komprimovani format muzičkih datoteka WMA (što je skraćenica za *Windows Media Audio*). Taj format primarno namenjen za upotrebu na programu *Windows Media Player*. Sve su češći i prenosni MP3/WMA *playeri*, a ovaj format zapisa muzike prihvatili su i divovi muzičke industrije poput firme *Sony*.

3.5.2. Komprimovanje audio ili zvučnih podataka

Većina ljudi nije ni svesna da se sa komprimovanjem audio podataka, tačnije sa smanjenjem protoka podataka, susreću svaki dan koristeći mobilni telefon. Uopšteno gledano, smanjenje protoka podataka nije nova stvar. Može se reći da je već kod telegrafa i prvog prenosa analognog signala bežičnim ili žičanim putem, postojala neka vrsta smanjenja protoka podataka. Međutim, kod svih komprimovanja postoji isti problem, a to je kako dovoljno smanjiti protok informacija, a da se ne izgubi na kvalitetu signala. Kod audio signala je to vrlo važno jer je ljudsko uvo dosta osetljivo na izobličenja i razne druge promene.

Zvuk koje ljudsko uvo čuje se kreće od 20 Hz do 20 kHz. To znači da sve frekvencije ispod ili iznad čovek ne primećuje u prirodi i nije potrebno njihova reprodukcija. Naime, ljudsko uvo prepoznaje pojedine frekventijske pojaseve kao grupe, odnosno tipove zvukova, u grubo – niske frekvencije, srednje frekvencije i visoke frekvencije. Niske frekvencije se približno nalaze u pojasu od 20 Hz do 400 Hz, srednje frekvencije od 400 Hz do 8.000 Hz, a visoke frekvencije od 8.000 Hz do 20.000 Hz. Prag čujnosti ljudskog uva je vrednost najmanje promene zvučnog pritiska koje uvo može da detektuje, i iznosi oko 20 mikropaskala. Ta vrednost je prosečna vrednost i različite osobe mogu bolje ili slabije čuti od te prosečne vrednosti. Prag čujnosti na frekvenciji 1 kHz uzeta je kao referentna vrednost i označena je sa 0 dB.

Karakteristično je za ljudsko uvo da subjektivno ne čuje sve frekvencije jednako. Ljudsko uvo je osetljivije na srednje frekvencije, nego na niske i visoke frekvencije. Što je visina signala viša, to se razlike između pojedinih frekvencija smanjuju. Tako, kada se čovek približi granici bola (otprilike 120 dB) ove razlike u čujnosti različitih frekvencija se smanjuju. Ova pojava je uzeta u obzir kod komprimovanja. Osim te pojave u obzir je uzet i efekat **maskiranja**. Tačnije, jedan od osnovnih koncepata na kojima se temelje svi psihoakustički algoritmi kodiranja zvučnih signala je maskiranje. Šta je to maskiranje? Maskiranje je akustički efekat kada snažniji zvuk ili zvuk sa nekim posebnim karakteristikama, potpuno prekriva neki drugi zvuk i čini ga nečujnim za slušaoca.

Kako je ljudsko uvo lako prevariti, može se iskoristiti mogućnost da, ako se pored signala jedne frekvencije nalazi signal druge, bliske frekvencije sa manjim intenzitetom, onda

će glasniji signal maskirati tiši. Laički rečeno, ovaj tiši signal se može izbaciti. Algoritam analizira signal u frekvencijskom domenu, tako da celi spektar deli na određeni broj delova. Ako se u nekom frekvencijskom pojasu uoči ovaj efekat maskiranja, onda se taj pojas neće ubaciti u niz podataka i na taj način se dobija smanjenje protoka podataka. Dinamika ljudskog sluha, odnosno razlika između praga čujnosti, odnosno najtiših signala i praga bola je otprilike 120 dB, ali samo na spomenutim srednjim frekvencijama.

3.6. MIDI i digitalni audio

MIDI (*Musical Instrument Digital Interface*) je komunikacioni standard razvijen za elektronske muzičke instrumente i računare. Dozvoljava da muzika i zvučni sintisajzeri različitih proizvođača komuniciraju među sobom duž kablova kojima su uređaji povezani. MIDI obezbeđuje protokol za prosleđivanje detaljnih opisa muzičke notacije, kao što su note, nizovi nota i koji instrument će svirati te note. Ali, MIDI podaci nisu digitalizovan zvuk; to je stenografsko predstavljanje muzike snimljene u numeričkom obliku. Digitalni audio je snimak, MIDI su note – prvo zavisi od mogućnosti zvučnog sistema koji se upotrebljava, a drugo od kvaliteta muzičkog instrumenta i mogućnosti sistema.

MIDI datoteka je spisak vremenski označenih naredbi koje su snimljene muzičke akcije (pristiskanje dirke na klavijaturi, održavanje pritisnute pedale ili pokret upravljačkog točka) koji, kada se upute MIDI uređaju za reprodukciju, proizvode zvuk. Koncizne MIDI poruke mogu da prouzrokuju kompleksne zvukove ili nizove zvukova na instrumentu ili sintisajzeru.

Nasuprot MIDI podacima, digitalni audio je stvarna prezentacija zvuka, snimljena u obliku hiljada individualnih brojeva (koji se zovu **uzorci**). Digitalni podaci predstavljaju trenutnu amplitudu zvuka (ili zvučnost) u diskretnim vremenskim intervalima. Pošto ne zavise od uređaja, digitalni audio zvuči uvek isto kad god se reprodukuje. Ta konzistencija ima svoju cenu – velike datoteke za čuvanje podataka. Digitalni zvuk se koristi za muzičke CD-ove i MP3 datoteke.

MIDI podaci su za digitalni audio isto što je vektorska grafika za bitmapiranu grafiku. MIDI zavisi od uređaja, a digitalni podaci ne zavise. Kao što se izgled vektorske grafike razlikuje u zavisnosti od štampača ili monitora, tako zvukovi koje prave MIDI datoteke zavise od konkretnog MIDI uređaja za reprodukciju. Digitalni podaci proizvode zvukove koji su manje ili više identični, bez obzira na reprodukcioni sistem. MIDI standard liči na *PostScript* jer omogućava instrumentima da komuniciraju na standardizovan način.

MIDI ima nekoliko prednosti u odnosu na digitalni audio i dve mane. Prednosti su:

- MIDI datoteke su mnogo kompaktnije, nego datoteke za digitalni audio, a veličina MIDI datoteke ne zavisi od kvaliteta reprodukcije. Generalno, MIDI datoteke će biti 200 do 1.000 puta manje od datoteka digitalnog audia CD kvaliteta. Pošto su MIDI datoteke male, ne zauzimaju previše RAM-a, prostora na disku i resursa centralnog procesora.
- Pošto su MIDI datoteke male, te datoteke ugrađene u veb strane učitavaju se i izvode brže nego njihovi digitalni ekvivalenti.

- U nekim slučajevima MIDI datoteke mogu zvučati bolje od digitalnog audia ako se koristi MIDI uređaj visokog kvaliteta.
- Dužina MIDI datoteke može da se menja (promenom tempa), bez promene visine tona ili gubljenja audio kvaliteta. MIDI podaci su kompletno promenljivi, i to sve do nivoa pojedinačne note. Korisnik je u stanju da manipuliše i najmanjim delom MIDI kompozicije, što je nemoguće sa digitalnim audiom.

Pomenuto je da MIDI ima dve mane u odnosu na digitalni audio i one su:

- Pošto MIDI podaci ne predstavljaju zvuk nego muzičke instrumente, onda je sigurno da će reprodukcija biti precizna samo ako je reprodukcioni uređaj identičan proizvodnom.
- MIDI ne može lako da se upotrebi za reprodukciju govornog dijaloga, iako postoje skupi i tehnički komplikovani digitalni sampleri.

Najvažnija prednost digitalnog audia je njegov stalni kvalitet izlaznog zvuka, gde je MIDI najmanje pouzdan! Sa digitalnim audiom korisnik može da ima više samopouzdanja da će audio snimak za multimedijalni projekat zvučati isto na kraju kao i na početku, kada je napravljen. Zbog toga, nije iznenađenje da se digitalni audio koristi mnogo češće, nego MIDI podaci za multimedijalne zvučne podloge.

Postoje još dva razloga da se radi sa digitalnim audiom, umesto sa MIDI podacima:

- Postoji širi izbor aplikacionog softvera i systemske podrške za digitalni audio, i za *Macintosh* i za *Windows*.
- Priprema i programiranje za pravljenje digitalnog audia ne zahtevaju poznavanje muzičke teorije; rad sa MIDI podacima obično zahteva solidnije znanje o muzičkim partiturama, klavijaturama, notama i audio proizvodnji.

3.7. Dodavanje zvuka u multimedijalni projekat

Bez obzira na kom operativnom sistemu korisnik radi, moraće da prođe izvesne korake kako bi "uveo" audio snimak u multimedijalni projekat. Evo kratkog pregleda procesa:

1. Treba odrediti koja vrsta zvuka se traži (kao što su muzika u pozadini, posebni zvučni efekti ili govorni dijalog).
2. Treba odrediti gde ovi zvučni događaji treba da se odigraju tokom projekta.
3. Treba uneti naznake položaja zvuka u scenario ili napraviti poseban spisak takvih naznaka.
4. Treba odlučiti kada i gde će se koristiti digitalni zvuk ili MIDI datoteke.
5. Treba nabaviti izvorni materijal praveći ga od nule ili kupovinom.
6. Treba editovati i uređivati zvučeve kako bi odgovarali projektu.

7. Treba testirati zvukove kako bi se videlo da li su ispravno povezani sa slikama iz projekta. To može da znači ponavljanje koraka od 1 do 6, dok se sve ne sinhronizuje.

Kada dođe vreme da se skupljeni i editovani zvuci uvezu u projekat, korisnik mora da zna kako konkretna multimedijalna softverska okolina radi sa zvučnim podacima. Svaki program to radi malo drugačije, ali proces je, uobičajeno, pravolinijski – softveru treba reći koju datoteku da svira i kada da je svira. To se, obično, ručno obavlja tokom procesa uvoženja ili "povezivanja" i tom prilikom se vrši i identifikacija datoteka.

Skript jezici, kao što su Lingo (*Director*) i ActionScript (*Flash*), daju veći nivo kontrole nad reprodukcijom zvuka, ali korisnik mora da zna taj jezik i sredinu. U određenim autorizovanim sredinama obično je potrebno jednostavno odsvirati zvuk kada korisnik pritisne dugme, ali to možda neće biti dovoljno. Ako korisnik menja ekrane dok svira dugačka datoteka, na primer, možda će morati da isprogramira da zvuk stane pre napuštanja aktuelnog ekrana. Ako datoteka koju treba odsvirati ne može da se pronađe na hard disku, možda će korisnik morati da kodira celu sekciju radi obrade grešaka i položaja datoteke. I u štampanoj dokumentaciji i u dokumentaciji u računaru obično su prikazane naredbe za reprodukciju zvuka.

3.8. Licenciranje

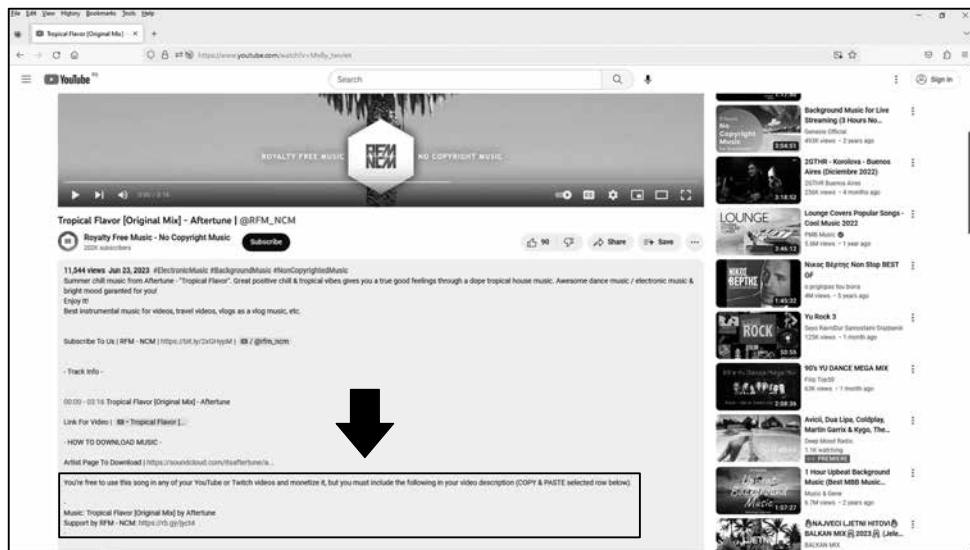
Muzička licenca predstavlja proces gde nosilac autorskih prava dozvoljava pravo na korišćenje svog dela. Korišćenje može biti javno, gde zauzvrat nosilac autorskih prava prima odgovarajuću novčanu nadoknadu. Takva nadoknada može biti u vidu fiksne sume novca, dogovorenog honorara ili na osnovu dogovorenog ugovora. Važno je istaći da licence mogu biti vremenski ograničene (mesec dana, pola godine, godinu i slično). Takođe, postoji tip licenci koje se izdaju na neograničeno vreme, poput otkupa od nosioca autorskih prava. Prilikom licenciranja zvuka i zvučnih zapisa (tonova, zvučnih efekata, melodija i drugih elemenata), principi su veoma slični licenciranju grafičkih vizuala. Međutim, postoje određene razlike u procesu licenciranja zbog specifičnosti same tehnike i drugih karakteristika ove oblasti. Uobičajeni primeri licenci koje se mogu pronaći na internetu su tradicionalne autorske licence, *Creative Commons* (CC) licence, *Royalty-Free* licence i druge. Suštinski, muzička licenca je, zapravo, mehanizam pozajmljivanja čime se omogućava javno korišćenje odgovarajuće muzičke numere, uz odgovarajuću naknadu. Bitna stavka je da se definisana autorska prava suštinski odnose na snimak (i eventualni tekst muzičke numere). Korisnik koji se pretplatio na odgovarajuću licencu je *de facto* pozajmio muzičku numeru, ali nije ostvario pravo nad intelektualnom svojinom.

Tradicionalne autorske licence zasnivaju se na autorskim pravima, što znači da se za korišćenje audio sadržaja mora imati odgovarajuća licenca od vlasnika, odnosno, autora tog audio sadržaja. Autor definiše pravila, moguće naknade i ograničenja korišćenja, i to mora da se poštuje.

Creative Commons licence su veoma popularne među korisnicima širom sveta, s obzirom na to da pružaju različite uslove korišćenja. Princip je gotovo identičan kao kod licenciranja slika, sa različitim verzijama CC licenci koje zahtevaju odgovarajuće i relevantno navođenje autora, kao i propisane načine korišćenja sadržaja. Ove licence omogućavaju

zaštitu intelektualne svojine, ali istovremeno omogućavaju korisnicima da koriste sadržaj u skladu s jasno definisanim pravilima i uslovima.

Royalty-Free (RF) licence su takođe popularne među korisnicima na internetu, jer omogućavaju korišćenje autorskog dela uz jednokratno plaćanje ili eventualno pretplatu. Ono što ih izdvaja je da omogućavaju široku upotrebu bez dodatnih naknada, ali uz poštovanje određenih specifičnih uslova. Na primer, na platformi *YouTube*, može se zahtevati da se u opisu video snimka navede autor audio zvuka ili muzičke pratnje, uz dodatne informacije.



Slika 3.5. Ilustracija potrebnog opisa koji je potrebno ugraditi u opis video snimka gde se muzička pratnja koristi

Izvorna lokacija primera na slici 3.5 je:

https://www.youtube.com/watch?v=Mv8y_tvwJek.

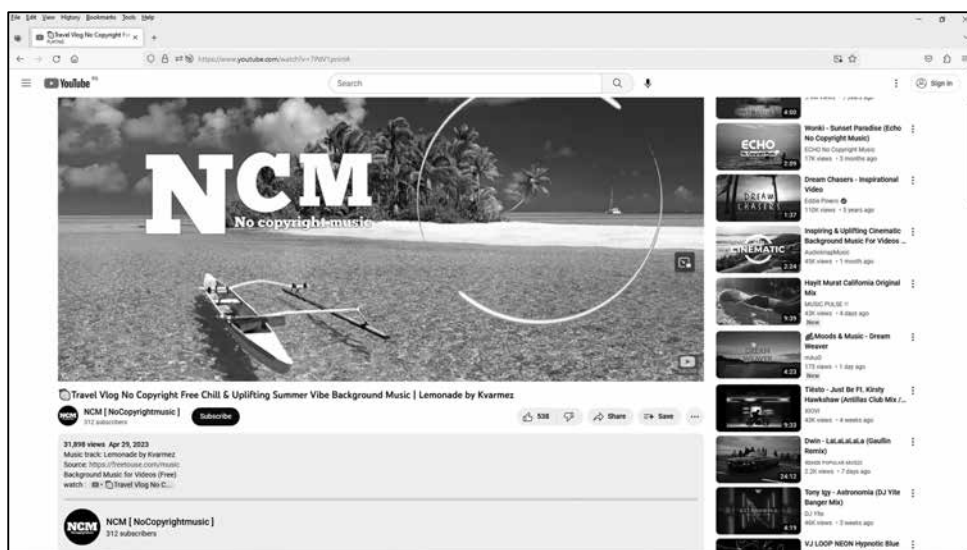
Ukoliko se koristi zvuk ili zvučni efekti, potrebno je voditi računa o autorskim pravima. U okviru projekta koji koristi odgovarajući zvučni efekat, potrebno je utvrditi da autor poseduje odgovarajuću licencu za korišćenje kako bi se poštovala intelektualna svojina autora. Takođe, ukoliko je potrebno koristiti zvučni zapis na veb sajtu, potrebno je proveriti da li postoji dozvola autora za korišćenje ili opšte pravo koje označava takav zvučni zapis kao besplatan za korišćenje.

Ukoliko se na veb sajtu koristi muzički zapis, potrebno je utvrditi da vlasnik veb sajta poseduje sve potrebne dozvole za korišćenje muzičkog zapisa. Muzički zapis može imati više vlasnika, stoga je neophodna saglasnost svakog od njih za korišćenje na veb sajtu. Ukoliko je nadležna distributerska kuća koja ima apsolutna prava nad muzičkim zapisom obezbedila taj zapis, licenca koju je korisnik dobio od te ustanove je u potpunosti validna.

Na globalnoj internet mreži mogu se koristiti zvučni efekti, koji su u osnovi kratki tonski zapisi. Trajanje zvučnih efekata varira od nekoliko sekundi do nekoliko desetina sekundi, u

zavisnosti od scenarija. Ti zapisi se prilagođavaju kontekstu u kojem se koriste kako bi stvorili odgovarajuću atmosferu ili ambijent. Na primer, na veb sajtu se može koristiti zvučni efekat tipkanja tastature, koji signalizira da će korisnički bot ili agent postaviti inicijalno pitanje u vidu dijaloga prikazanog u čet prozoru. Zvučni efekti podležu *Creative Commons* licencama. Pre korišćenja zvučnih efekata, potrebno je utvrditi nivo licence. Određeni autori i vlasnici zvučnih efekata mogu zahtevati navođenje autora. Takođe, važno je pridržavati se svih pravila navedenih u licenci vlasnika, budući da nepoštovanje može imati ozbiljne posledice.

Na primeru popularnog servisa Jutjub (eng. *YouTube*), ukoliko se koristi zvučni zapis kao pozadinska muzička pratnja u nekom video snimku, forenzički alati za zaštitu intelektualne svojine, odnosno autorskih prava, potrebno je svojim algoritmima prepoznavanja prepoznati poklapanje u odgovarajućim sekvencama. Navedeni servis ima nekoliko nivoa koji mogu uticati na korisnika; jedna od njih je mera u potpunosti ukidanja zvučnog zapisa u audio/video snimku, pri čemu ostalim korisnicima ostaje dostupan samo video snimak. Takođe, mera može biti u potpunosti blokiranje vidljivosti video snimka ili pak privremeno ili trajno blokiranje Jutjub kanala, što svakako zavisi od nekoliko faktora (npr. koliko puta je ignorisano upozorenje za kršenje autorskih prava, koliko puta je prekršeno pravilo korišćenja intelektualne svojine i drugo).



Slika 3.6. Video snimak na popularnom servisu Jutjub

Izvorna lokacija primera na slici 3.6 je:

<https://www.youtube.com/watch?v=7PdV1pnin1A>.

Postoji veliki broj biblioteka za dizajniranje i kreiranje zvuka, koje se često nazivaju VST dodaci (eng. *Virtual Studio Technology plugins*). Takve biblioteke mogu imati različite nivoje licenciranja, stoga je potrebno pridržavati se pravila korišćenja kako bi autor mogao

započeti postupak licenciranja, nakon što dizajnira odgovarajući zvučni efekat i pre nego što ga objavi širem audiotorijumu.

Svako kršenje ovih pravila nosi sa sobom izuzetno složene sankcije. Ukoliko dođe do kršenja autorskih prava (i intelektualne svojine), mogu se pokrenuti različiti pravni postupci koji obuhvataju podnošenje tužbi, a tuženom licu može biti izdato rešenje o finansijskoj nadoknadi štete.

Svaka država ima odgovarajuće vreme trajanja kraćeg dela tonskog zapisa (kao što su muzička numera, zvučni efekat i slično), koje se može koristiti bez saglasnosti autora ili vlasnika. Ovo prihvatljivo vreme je promenljivo i zavisi od specifičnosti i konteksta, kao i od zakona o autorskim pravima koji važe u toj državi.

Glava 4

Animacija

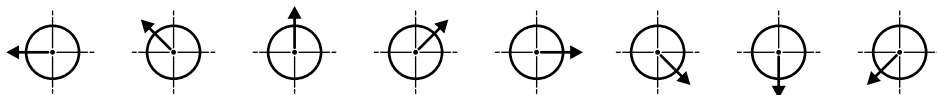
Animacija je postupak stvaranja iluzije kretanja crtežima, modelima ili beživotnim stvarima. To je optička iluzija kretanja zahvaljujući fenomenu poznatom kao perzistencija vida i može se izvesti na više načina. Najčešći metod prikazivanja animacije je film ili video program, mada postoji i nekoliko drugih formi predstavljanja animacije.

Moguće je animirati ceo projekat ili pojedine njegove delove, čime se oni naglašavaju ili im se daje poseban značaj. Za kratku demonstraciju proizvoda sa malo korisničke interakcije ima smisla da se dizajnira ceo projekat kao film i da se drži cela prezentacija stalno u pokretu. Za podršku govorniku može se animirati označeni tekst, može da se napravi da taj tekst uleće ili mogu da se naprave histogrami sa količinama koje rastu i smanjuju se; zatim, treba dati govorniku kontrolu nad ovim elementima koji privlače pažnju.

Vizuelni efekti kao što su brisanje, nestajanje, zumiranje i rastapanje su na raspolaganju u većini programskih paketa, a ponešto od toga može da se koristi za primitivnu animaciju.

4.1. Principi animacije

Animacija je moguća zbog biološkog fenomena poznatog kao perzistencija vida i odgovarajućeg psihološkog fenomena. Objekat koji vidi ljudsko oko ostaje hemijski preslikan na očnoj retini jedan kratak period posle viđenja. U kombinaciji sa potrebom ljudskog uma da konceptualno dovršava percipiranu akciju, omogućava se da se niz slika koje se menjaju vrlo malo i vrlo brzo, jedna za drugom, stopi u vizuelnu iluziju pokreta. Sledeća slika pokazuje nekoliko ćelija ili okvira u rotirajućem logotipu. Kada se slike progresivno i brzo menjaju, strelica kompasa se percipira kao da se vrti (slika 4.1).



Slika 4.1. Ključni kadrovi za kretanje kazaljke kompasa ili kazaljke na satu

Televizijski video gradi 30 kadrova (*frame*) ili slika svake sekunde; brzina kojom se svaki kadar zamenjuje sledećim omogućava da izgleda kao da se slike glatko spajaju u po-

kret. Film se obično snima brzinom od 24 kadra u sekundi, ali upotrebom projekcionih trikova (blenda u projektoru šalje svetlo dva puta kroz svaku sliku) treptanje se "penje" na 48 puta u sekundi i ljudsko oko to vidi kao pokretnu sliku. Na nekim projektorima svaki kadar se prikazuje tri puta pre nego što zupčanik povuče traku naniže do sledećeg kadra, za ukupnih 72 treptaja u sekundi, što pomaže da se eliminiše efekat treptanja – što je više prekida u sekundi, te zrak svetla izgleda neprekidniji. Brza promena posmatrane slike je princip animatične knjige za brzo prelistavanje, ili *zoetropa*. Da bi se objekat naterao da "putuje" preko ekrana dok menja oblik, treba mu samo promeniti oblik i translirati ga za nekoliko piksela u svakom kadru. Zatim, kada se ubrzano reprodukuju frejmovi, promene se spoje i pojavljuje se pokret i animacija.

4.2. Tipovi animacije

Od nastanka animacije do danas razvili su se razni oblici animacije. Neki su koristili u to vreme postojeće tehnike i dalje ih razvijali i prilagođavali svojim potrebama, drugi su stvarali i kasnije patentirali svoje tehnike animacije, tako da danas postoji više tehnika koje su u upotrebi, svaka za posebnu namenu.

Stop motion je vrsta animacije gde se pravi privid fizičkog kretanja objekata. Objekat se pomera u malim razmacima, svaki pokret se fotografiše, i kada se pusti kao kontinuirana sekvenca, stvara se iluzija kretanja. Na taj način mogu da se prave kretanja objekata koji su inače statični, kao što je kretanje čaše ili novčića po stolu.

Clay animacija je jedna od mnogih oblika *stop motion* animacije. Svaki animirani predmet se može deformisati, napravljen je obično od gline, oko žičanog skeleta. Kao u drugim oblicima animacije, objekat se formira, postavlja se na scenu, fotografiše se, zatim se ručno pomeri i modeluje, ponovo se fotografiše, i taj posao se ponavlja dok se ne dobije dovoljna dužina filma.

4.3. Vrste računarske animacije

2D animacija je proces stvaranja pokretnih slika u dvodimenzionalnom prostoru, bilo da je reč o tradicionalnoj animaciji ili računarskoj animaciji. Svaki frejm se posebno slika, a zatim se sve slike naizmenično prikazuju, jedna za drugom, brzinom od 24 sličice u sekundi ili brže. Tim postupkom oko može da se zavarava, da bi se stvorila iluzija pokreta.

Računarska animacija je digitalni naslednik *stop motion* animacije. Slike se stvaraju u računaru, animator pravi pokrete pomoću određenog programa. Program ne pravi svaki frejm, već se nacrtaju početna i krajnja pozicija objekta koji treba animirati, tj. prave se ključni frejmovi, a računar automatski izračunava frejmove koji se nalaze između dva ključna frejma. Ovakav način upotrebe računara dosta je skratio period koji je bio potreban da bi se uradio jedan sekund animacije na tradicionalan način, a mnogobrojne animatore zamenili su računari izvanrednih performansi.

CGI (Computer Generated Imagery) animacija je umetnost stvaranja pokretnih slika uz upotrebu računara. 3D animatori izrađuju modele uz pomoć posebnog programa, ti

modeli se "kače" na virtuelni skelet, zatim se ruke, noge, oči, usta, odeća figure pomeraju u ključnim frejmovima. Razliku između ključnih frejmova računar automatski proračunava u procesu koji je poznat kao *tweening* ili *morphing*. Konačno, animacija se renderuje.



Slika 4.2. Primeri 2D animacije

Najčešća upotreba CGI-a je u domenu 3D računarske grafike, specijalnim efektima na televiziji, filmu, u reklamama i raznim vrstama simulacija.

CGI se često upotrebljava za vizuelne efekte, jer se mogu bolje kontrolisati od nekih drugih fizičkih procesa kao što je izrada minijaturnih modela, scena ili unajmljivanje mnogobrojnih statista za masovne scene u filmovima. Ušteda u svemu, posebno u novcu, je velika.



Slika 4.3. CGI u punom sjaju

Ubrzani razvoj CGI softvera i povećana snaga računarskih procesora, omogućavaju umetnicima i malim kompanijama da proizvedu profesionalne filmove, igrice i razne umetničke forme, koristeći kućne računare.

Motion capture, *motion tracking* ili *mocap* su termini koji opisuju proces snimanja pokreta, a zatim prenošenje tih pokreta na digitalni model. Primena mu je raznovrsna, od vojne upotrebe, preko sveta zabave i sporta, do primena u medicini. U filmskoj industriji, *motion capture* se odnosi na snimanje jednog ili više glumaca u sceni, koji su obučeni u posebno odelo (slika 4.4). Na odelu su prikazani specijalni svetleći markeri, kamera snima njihove promene položaja i rotacije i tako snimljena scena se ubacuje u računar, koji uzima podatke sa markera i primenjuje ih na digitalno izrađeni model. Može da se prati *motion capture* kamere, da bi se napravila virtuelna kamera koja će moći da radi kao prava. Na taj način će digitalni karakteri imati istu perspektivu kao i glumac na sceni.



Slika 4.4. Glumci u specijalnim odelima sa svetlećim markerima

Ovaj sistem ima i svoje prednosti i svoje mane. Prednosti su sledeće:

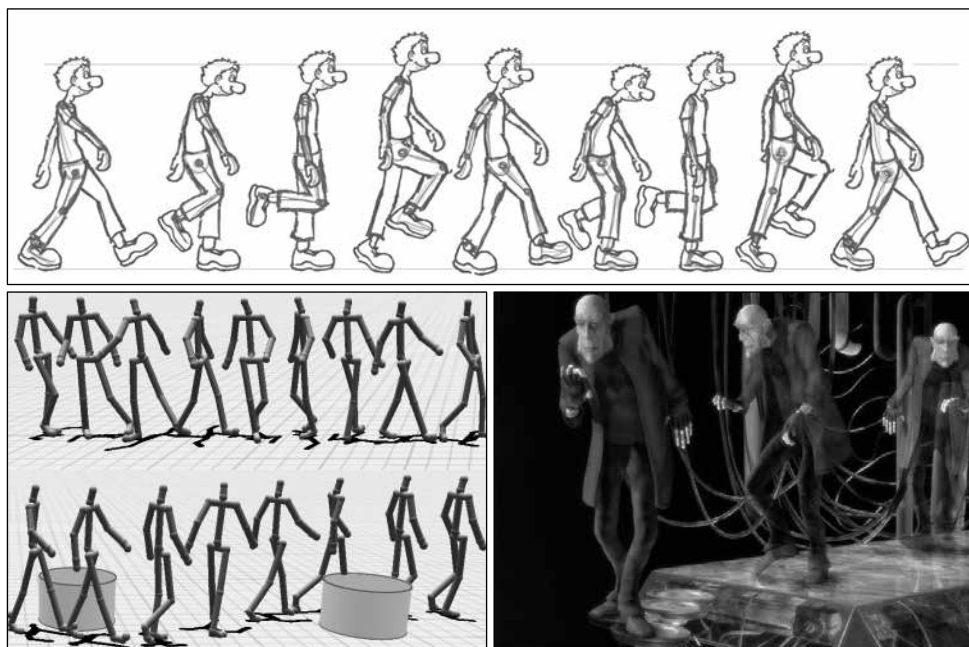
- Ovaj sistem može da prikaže rezultate rada gotovo u realnom vremenu, što znatno smanjuje troškove produkcije.
- Količina rada ne varira sa kompleksnošću izvođenja na sceni, kao kod tradicionalne animacije, što omogućava da se izvrše mnogobrojni testovi i različiti pristupi pre samog snimanja.
- Kompleksni i realistični pokreti mogu se snimiti sa velikom preciznošću.

Glavne mane sistema su:

- Potreban je poseban hardver i softver da bi se izvršilo procesuiranje podataka.
- Sama cena opreme i osoblja koji bi njom rukovali može biti ograničavajući faktor za manje produkcije.

- Rezultati su ograničeni snimljenim materijalom, naknadno se ne mogu dodavati novi podaci.
- Pokreti koji ne slede fizičke zakonitosti, ne mogu se snimiti.

Karakter animacija je specijalizovano područje animacije vezano za animaciju jednog ili više karaktera u animiranom filmu (slika 4.5). Predstavlja deo velike produkcije, i u umetničkom smislu je jedinstvena jer uključuje prikaz misli i emocija kao dodatak fizičkom pokretu. Iako je tipična namena ove vrste animacije u animiranom filmu, njena uloga u industriji računarskih igrica je ogromna. Ljudi koji razvijaju igrice, koriste komplikovane karaktere koji omogućuju korisniku da se u potpunosti sjedini sa likom u igri kako bi njegov doživljaj bio što potpuniji.



Slika 4.5. Nekoliko primera karakter animacije

4.4. Budućnost računarske animacije

Upotreba računara dovela je do novog pristupa animaciji. Počelo je tako što su računarski stručnjaci videli mogućnost u računarima kako bi izradili odlične slike, koje bi bile teško, ako ne i nemoguće napraviti na drugačiji način. Tokom druge polovine dvadesetog veka, računarski generisana grafika upotrebljena je veoma uspešno u velikim holivudskim filmovima Zvezdane staze, Park iz doba Jure, Gospodar prstenova, Hari Potter, Avatar itd.

CGI je upotrebljen za izradu impresivnih scena, za koje bi inače bilo potrebno mnogo novca za izradu realističnih maketa. Jedan od većih izazova u računarskoj animaciji, predstavlja

izrada fotorealističnih ljudskih karaktera. Do sada, uglavnom su pravljene animacije životinja, *science fiction* karaktera i crtanih ljudskih figura.

Neki filmovi su napravili pokušaj da prikažu realistična ljudska bića, kao što je to urađeno u filmu *Final Fantasy* (slika 4.6), ali zbog izuzetne kompleksnosti ljudskog tela, pokreta i mimike, realistična simulacija ljudi ostaje i dan danas jedan od najvećih izazova u računarskoj animaciji.



Slika 4.6. *Odredene scene iz filma Final Fantasy*

Budućnost animacije je teško zamisliti, a naročito predvideti. Svaki dan neki novi program, nova tehnika izlazi na tržište i postavlja nove kriterijume u računarskoj animaciji.

U budućnosti, neki ljudi očekuju da će **virtuelna realnost** zameniti sadašnje tehnike 2D i 3D animacije, što znači da će osoba koja gleda film, videti sebe u tom filmu kao i osoba koja taj film posmatra sa strane.

Glava 5

Video

Video je elemenat multimedije koji može čvrsto da drži interes studenta za učenje pomoću računara. Digitalni video najviše i angažuje od svih multimedijalnih mogućnosti; to je vrlo moćno sredstvo da se korisnici računara približe realnom svetu. To je, takođe, efikasan metod da se isporuči multimedija publici koja je odrasla uz televiziju. Sa video elementima u projektu, korisnik može efikasno da predstavi svoje poruke i da pojača priču, a gledaoci naginju ka tome da usvoje što više od onog što vide. Ali, treba biti krajnje oprezan! Video koji nije dobro osmišljen ili nije dobro produciran u stanju je da degradira prezentaciju.

Od svih multimedijalnih elemenata video postavlja najviše izvođačke zahteve pred računar i njegovu memoriju. Treba se setiti da nepokretna slika najvišeg kvaliteta na računarskom ekranu može da zahteva čak megabajt memorije. To treba pomnožiti brojem 30 (toliko puta u sekundi se slika menja da bi se dobio privid pokreta) i potrebno je 30 megabajta memorije po sekundi kako bi se reprodukovala, ili 1,8 gigabajta memorije u minutu, ili 108 gigabajta na sat. Samo premeštanje svih ovih slika iz memorije računara na ekran tom brzinom bilo bi problem i za super-računar. Neke od najinteresantnijih i najčudnijih multimedijalnih tehnologija i istraživanja danas se posvećuju komprimovanju podataka za slike digitalnog videa u upravljive tokove informacija.

Ako je korisnik u stanju da kontroliše platformu za isporučivanje za projekat, može da dobije najveće video performanse dodavanjem posebnih hardverskih i softverskih proširenja. Hardver za video kompresiju će obezbediti da se radi sa videom na celom ekranu, u punom pokretu. Razvijene audio kartice će omogućiti da se koristi zvuk CD kvaliteta. Korisnik može da instalira i superbrze RAID (*Redundant Array of Independent Disks*) diskove koji će podržavati velike brzine transfera, a u sistemu može da se definiše da se video po delovima unosi u RAM, radi brže reprodukcije.

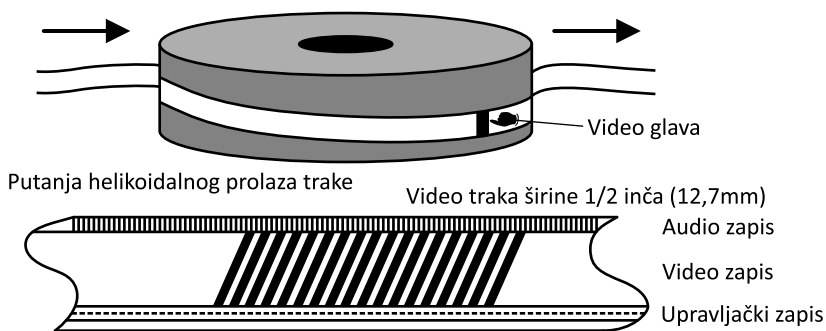
Pažljivo isplaniran, dobro izveden video spot može da napravi dramatičnu razliku u multimedijalnom projektu. Međutim, pre nego što se korisnik odluči da u svoj projekat doda video, od suštinske važnosti je da razume medij, njegova ograničenja i troškove.

Video standardi i formati se i dalje usavršavaju dok se tehnologije prenosa, skladištenja, kompresije i prikaza formiraju u laboratorijama i na tržištu, gde su uključeni oprema i postprocesiranje od analognog početka do digitalnog završetka, od "hvatanja" ekrana do

krajnjeg prikazivanja. Rad sa današnjim multimedijalnim videom može da se uporedi sa putom kroz pustinju – korisnik može da postavi šator na udobnom visokom zemljištu i da otkrije da je pokretni pesak, preko noći, "sahranio" i njegov pristup i njegovu investiciju. U današnje vreme tehnologije sve ima tendenciju da se brzo menja, tako da treba držati korak sa zahtevima i potrebama. To je tačno, posebno u današnja vremena kada se televizije kreću od NTSC (*National Television Standard Committee*) standarda ka novom DTV (*Digital Television*) standardu.

5.1. Kako video radi?

U analognim sistemima video signal iz kamere se isporučuje na *Video In* konektor video-rekordera, gde se snima na magnetsku video traku. Kamkorder može da kombinuje i kameru i snimanje na traku u jednom uređaju. Jedan ili dva kanala zvuka se, takođe, mogu snimiti na traku (mono ili stereo). Video signal se snima na traku obrtanjem glave za snimanje koja menja lokalne magnetske osobine površine trake u nizu dugačkih diagonalnih pruga. Pošto traka sledi helikoidalnu putanju, to se zove **prolazno helikoidalno snimanje**. Kao što je to ilustrovano na slici 5.1, svaka pruga predstavlja informaciju za jedno polje video okvira.



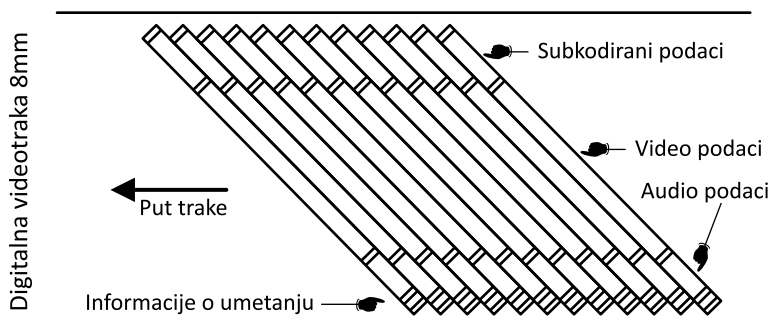
Slika 5.1. Putanja trake preko video glave za analognu snimanje

Pojedinačni video okvir se sastoji od dva isprepletena polja. Audio se snima na zasebnom pravolinijskom zapisu na vrhu videotrake, iako se u nekim sistemima za snimanje (za traku od 3/4 inča – 19,05 mm i za traku od 1/2 inča – 12,7 mm sa audiom visokog kvaliteta) zvuk snima helikoidalno između video pruga. Pri dnu trake je kontrolni zapis, koji sadrži impulse za regulisanje brzine. **Treking** (*tracking*) je fino podešavanje traka da se one ispravno poravnavaju kako se kreću po reprodukcionim glavama.

U digitalnim sistemima video signal iz kamere se najpre digitalizuje kao pojedinačni frejm i podaci se komprimuju pre nego što se upišu na traku u 1 od nekoliko formata – DV, DVCPRD ili DVCAM (slika 5.2). Postoje i druge konfiguracije video traka koje se koriste u profesionalnim produkcijama sa profesionalnom opremom.

Do pre izvesnog vremena, video rekorder je mogao da doda video i zvučne signale i da ih moduliše u radio frekvenciju (RF) u FM emisionom području. To su NTSC, PAL ili SECAM signali koji se dobijaju na *Antenna Out* konektoru rekordera. Obično korisnik može da

bira između određenih frekvencija, a rezultujući signal može da se vidi na televiziji. Mnogi televizijski aparati daju zaseban konektor za kompozitni signal da bi se izbegao nepotreban korak modulisanja i demodulisanja signala na frekventni opseg emitovanja. Video monitori imaju samo konektor za kompozitni signal i ne mogu da primaju RF signale.



Slika 5.2. Putanja trake preko video glave za digitalno snimanje

5.2. Standardi za analogno prikazivanje

Najčešće se koriste tri standarda za emitovanje videa i formata za snimanje – NTSC, PAL i SECAM. Pošto ove standarde i formate nije lako razmenjivati međusobno, važno je da korisnik zna gde će njegov multimedijalni projekat da se koristi. Video kasete snimljene u SAD neće moći da se reprodukuju na televizijskom aparatu u evropskim zemljama, uprkos tome što je način snimanja i stil kasete VHS. Slično tome, trake snimljene u evropskim formatima PAL ili SECAM neće moći da se prikažu na NTSC video rekorderu. Svaki sistem se zasniva na različitom standardu koji definiše kako se tekuća informacija enkodira da bi se napravio elektronski signal koji na kraju stvara televizijsku sliku. Multifformatni video rekorderi mogu da reprodukuju sva tri standarda, ali obično ne mogu da konvertuju sa jednog standarda na drugi. **Presnimavanje** (*dubbing*) u tu svrhu i dalje zahteva vrhunsku, specijalizovanu opremu.

5.2.1. NTSC

SAD, Kanada, Meksiko, Japan i mnoge druge zemlje koriste sistem za emitovanje i prikaz videa koji se zasniva na specifikacijama koje je 1952. godine postavio Nacionalni komitet za televizijske standarde (*National Television Standards Committee* – NTSC). Ti standardi definišu metod za enkodiranje informacije u elektronski signal koji na kraju stvara televizijsku sliku. Kao što je zacrtano u NTSC standardu, jedan frejm videa je napravljen od 525 horizontalno skeniranih linija, koje na unutrašnjoj strani fosforom premazane cevi svakih 1/30 delova sekunde iscrtava brzopokretni elektronski zrak. Iscrtavanje se dešava tako brzo da oči "vide" sliku kao stabilnu. Elektronski zrak pravi, u stvari, dva prolaza dok crta jedan video frejm – prvo sve neparne, zatim sve parne linije. Svaki od ovih prolaza (koji se dešavaju frekvencijom od 60 u sekundi, ili 60 Hz) crta polje. Proces pravljenja jednog frejma od dva polja zove se **preplitanje** (*interlacing*), i to je metod koji pomaže da se

spreči treptanje na TV ekranima. Računarski monitori koriste tehnologiju **progresivnog prolaženja** i crtaju linije celog frejma u jednom prolazu, bez preplitanja.

5.2.2. PAL/SECAM

Sistem *Phase Alternate Line* – PAL koristi se u Velikoj Britaniji, Zapadnoj Evropi, Australiji, Južnoj Africi, Kini i Severnoj Americi. PAL sistem je povećao rezoluciju ekrana na 625 linija frekvencijom od 25 frejmova u sekundi. Kao i kod NTSC sistema, parne i neparne linije su isprepletane; svako polje se crta za 1/50 sekunde (50 Hz).

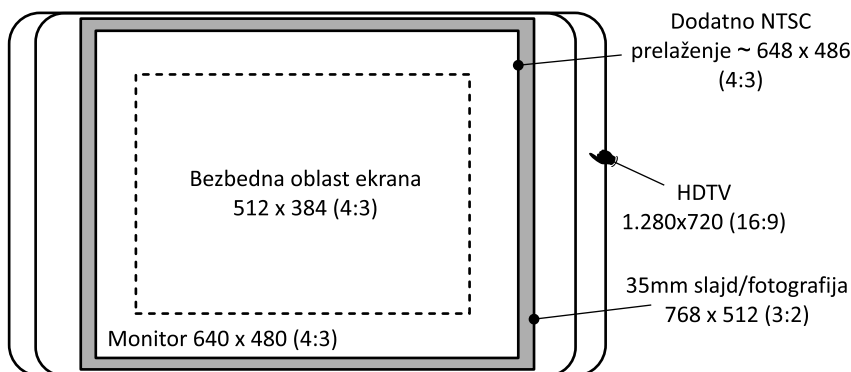
Sistem *Sequential Color and Memory* – SECAM se koristi u Francuskoj, Istočnoj Evropi, Rusiji i nekoliko drugih zemalja. Iako SECAM ima 625 linija na 50 Hz, mnogo se razlikuje i od NTSC i od PAL sistema po osnovama svoje tehnologije i metodu emitovanja. Međutim, TV aparati koji se prodaju u Evropi često koriste dualne komponente, pa mogu da koriste i PAL i SECAM sisteme.

5.3. Digitalni video

Puna integracija videa na računarima eliminiše analognu televizijsku formu videa sa platforme na koju će se isporučiti video. Ako je video spot snimljen kao podatak na tvrdi disk, CD ROM ili neki drugi uređaj masovne memorije, taj spot može biti reprodukovan na monitoru računara bez kartica za preklapanje video signala, čitača videodiskova ili drugog monitora. Reprodukcijski digitalnog videa postiže se softverskim arhitekturama, kao što su QuickTime ili AVI. Kao multimedijalni producent ili stvaralac, korisnik može da ima potrebu da konvertuje izvorni video materijal iz njegove još uvek česte analogne forme (video-traka) u digitalni oblik, kojim može upravljati njegov računarski sistem.

5.3.1. HDTV

Televizija visoke definicije (*High Definition Television* – HDTV) daje visoku rezoluciju u odnosu 16 : 9 (slika 5.3). Taj odnos omogućava gledanje *Cinemascope* i *Panavision* filmova.



Slika 5.3. Razlika u proporcijama između VGA i HDTV standarda

Postoji spor između industrije emitovanja i industrije računara da li da se koristi preplitanje ili progresivno prolaženje. Industrija emitovanja promovise format ultraviske rezolucije 1.920×1.080 sa preplitanjem, koji treba da postane okosnica nove generacije vrhunskih centara za zabavu, a industrija računara bi želela da se zaustavi na rezoluciji 1.280×720 sa progresivnim prolaženjem za HDTV. Format 1.920×1.080 daje više piksela nego 1.280×720 , ali frekvencije "osvežavanja" su sasvim drugačije. Format više rezolucije sa preplitanjem daje samo polovinu slike svakih $1/60$ u sekundi, a, zbog preplitanja, vrlo detaljne slike će treptati na 30 Hz. Nasuprot tome, računarski stručnjaci kažu da je kvalitet slike na 1.280×720 superioran i postojan.

5.3.2. Analogno i digitalno

Osnovni element računarske tehnologije u svakodnevnoj praksi je **mikroprocesor**, skup više miliona tranzistora integrisanih u malom kućištu ne većem od kutije šibica, okosnica mikroracunarskog sistema, koji se može **programirati** da izvrši zadatke prema željama korisnika. Procesor u suštini vrši obradu **podataka**, prema zadatom programu, te se na osnovu saznanja u postojećim podacima stiču nova saznanja kao proizvod njihove obrade.

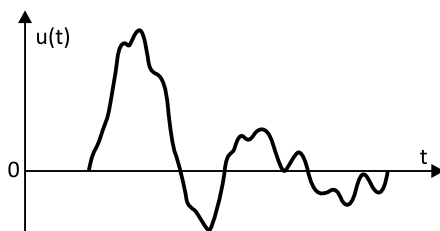
Podatak je u osnovi **poruka** koja se može i ne mora iskoristiti. Ako postoji i najmanja verovatnoća da se poruka jednoznačno i tačno iskoristi, te predstavlja neoborivu činjenicu, tada predstavlja **informaciju**. Dakle, svaka poruka može i ne mora sadržavati informaciju.

Nužno je da računar rezultat obrade podataka prikaže ili pošalje korisniku te podatke za obradu primi na odgovarajući način, odnosno treba da ostvari **komunikaciju** i izvrši razmenu informacija. Stanje ili proces materijalnog sistema koji prenosi informaciju (poruku) od izvora prema odredištu u obliku pogodnom za prenos naziva se **signal** (akustički, električni, svetlosni itd.), a medij kroz koji signal prolazi naziva se **prenosni put**. Pretvaranje poruke u signal vrši predajni uređaj, a pretvaranje signala u poruku vrši prijemni uređaj, a celokupni sistem prenosa poruke predstavlja komunikacijski sistem.

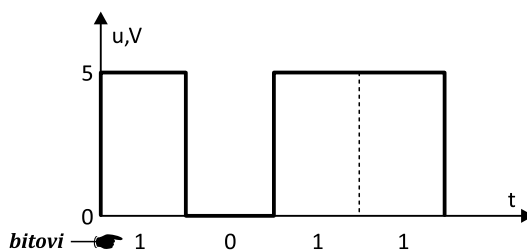
Različite vrste poruka, pa stoga i mogućih informacija, mogu se grupisati prema sledećem: *zvučna informacija* (na primer, telefon); *nepokretna slika* (na primer, telefaks); *pokretna slika* (na primer, televizija); *telesignalizacija* (na primer, daljinsko upravljanje procesima); *podaci* (grupe znakova, slika...).

Električni signal, kao neposredni nosilac informacije, od posebnog je značaja. Treba napomenuti da nije i jedini. Govorom u slušalicu telefonskog aparata (predaja), govorni organi proizvode **akustički signal** koji se pomoći mikrofona pretvara u **električni signal** koji sada u sebi sadrži govornu informaciju. Na prijemu se električni signal pomoću slušalice ponovo pretvara u akustički signal gotovo istovetan onome na izvoru, a koga sada obrađuju slušni organi (prijem).

Kako se oblik akustičkog signala govora slaže sa oblikom signala koji se prenosi komunikacijskim sistemom i pretvara u gotovo istovetan oblik akustičkog signala na odredištu, te po obliku pri prenosu ne doživljava izmene i verno prati promene izvora, takav signal naziva se **analogni signal**. Tokom prolaska kroz provodnik (prenosni put) signal je izložen smetnjama i izobličavanju što ponekad može imati za rezultat nerazumljivost poruke na odredištu. Razumljiva poruka, po kojoj sledi ispravna akcija, sadrži informaciju.

Slika 5.4. *Analogni signal*

Izvor informacije može predati poruku i na drugi način. Poruka se prema **dogovoru** izvora i odredišta pretvori u signal sastavljen od kombinacije impulsa, svetlosnih, akustičkih i slično, od kojih svaka pojedina kombinacija predstavlja neki znak, kao kod Morzeove abecede. Takav način prenosa impulsa nejednakog trajanja i dužine neprikladan je za brze komunikacije, te se u računarskoj tehnici prihvatio dosledniji način prenosa impulsa. Na primer, pritiskom na taster tastature računara (ili teleprintera) stvara se odgovarajući **niz** (povorka) **impulsa** jednakih po obliku i dužini trajanja niza (broja mesta za impulse), ali različit po broju impulsa u nizu i njihovom rasporedu unutar niza za svaki znak tastature. Takav niz se sa tastature šalje računaru i sadrži poruku o pritisnutom tasteru i o pripadnom znaku.

Slika 5.5. *Digitalni signal*

Na istovetan način će se razmenjivati poruke između računara prema dogovorenoj tablici signala za pojedine znakove. Kako su broj mesta, broj impulsa u nizu i njihov međusobni raspored unutar niza elementi nekog brojnog sistema, te računar primljeni niz prepoznaje kao odgovarajući **broj**, signal prikazan na prethodnoj slici naziva se **digitalni signal** (digit = broj).

U principu, postoje dva načina obrade – analogna obrada i digitalna obrada signala. Kao primer analogne obrade signala može se uzeti ljudsko telo. Ljudsko telo pretvara informacije primljene ušima, očima i kožom u električne signale različitih inteziteta, koji prolaze kroz nervni sistem do mozga. Ljudski mozak obrađuje te promenljive koje su u ovom slučaju kontinualni ili analogni signali. Zovu se kontinualni zato što mogu imati beskonačno mnogo vrednosti između minimalne i maksimalne vrednosti. Posle obrade tako primljenih informacija i donešenih odluka na osnovu njih, on šalje električne signale u druge delove tela kao odgovor na spoljne nadražaje. Digitalna obrada signala sadrži još jedan međukorak. Pošto računar koji vrši obradu signala "ne razume" analogne vrednosti, potrebno je iste pretvoriti u njemu prihvatljiv "jezik". Kada se usvoji stav da nisu važne sve vrednosti, već samo određen, unapred dogovoren broj nivoa analognog signala, dolazi se do

kvantovanja analognog signala. Na taj način "preveden" analogni signal je "razumljiv" za računar. Ulogu prevodioca ima na primer, mikrofoni koji pretvaraju energiju zvučnih talasa u električne, električni signali se zatim digitalizuju pomoću konvertora i kao takvi predaju računar. Posle obrade se preko zvučnika ponovo pretvaraju u zvučne signale.

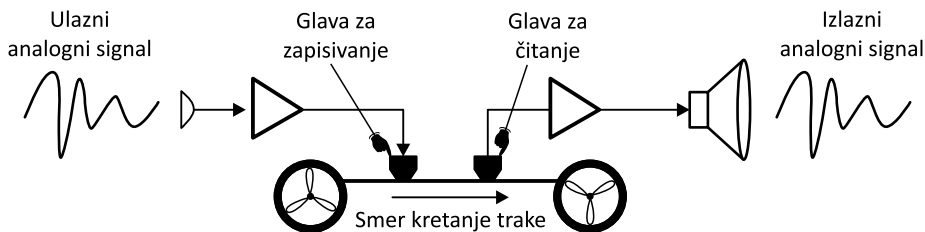
5.3.3. Digitalna obrada signala

Šta je digitalna obrada signala (*Digital Signal Processing* – DSP) uopšte i zašto treba da se koristi?

Termin DSP uopšteno označava da se koriste digitalni računari za obradu signala. Naravno, ovi signali se mogu obrađivati analognim procesima ali, iz raznih razloga, daje se prednost digitalnoj obradi.

Da bi se shvatile relativne prednosti jedne obrade u odnosu na drugu, pogodno je da se uporede ove dve tehnike kod neke opšte poznate aplikacije. Na slici 5.6 je prikazano analognu snimanje i ono radi na sledeći način:

- Zvučni talasi dopiru do mikrofona, gde se pretvaraju u električne impulse.
- Ovi električni impulsi se pojačavaju, zatim pretvaraju u magnetna polja pomoću glave za snimanje (upisivanje).
- Kako se magnetna traka pomera ispod glave za snimanje, intenzitet magnetnih polja se smešta na traku.



Slika 5.6. Snimanje analognog signala

Proces reprodukcije je inverzan procesu snimanja:

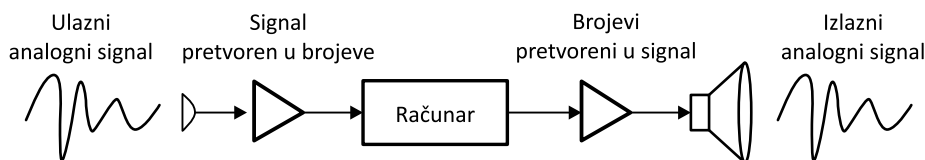
- Kako se magnetna traka pomera ispod glave za reprodukciju (čitanje), magnetno polje sa trake se pretvara u električni signal.
- Signal se zatim pojačava i šalje na zvučnike. Zvučnici pretvaraju ovako pojačan signal nazad u zvučni talas.

Prednost analognog procesa je dvostruka – on je sasvim jednostavan i analogni signal može da ima praktično beskonačan broj vrednosti unutar dinamičkog opsega signala. Nažalost, ovaj analogni proces je sam po sebi nestabilan. Pojačavači su izloženi varijaciji pojačanja u zavisnosti od temperature, vlažnosti i vremena. Magnetna traka se tokom vremena gužva i isteže, usled čega se signal izobličava. Magnetna polja će tokom vremena gubiti jačinu. Varijacije brzine motora koji premotava traku izazivaju dodatna izobličenja. Svi ovi

faktori utiču na pogoršanje kvaliteta izlaznog signala u odnosu na ulazni signal. Svaki put kada signal prolazi kroz sledeći analogni proces (presnimavanje trake), ovi neželjeni efekti se umnožavaju. Retko koji analogni sistem može da napravi više od 2 ili 3 kopije.

Sada treba pogledati digitalni proces snimanja (slika 5.7):

- Kao i kod analognog snimanja, zvučni talasi dopiru do mikrofona i pretvaraju se u električne signale. Ovi električni signali se zatim pojačavaju do upotrebljivog nivoa.
- Električni signali se mere ili, drugim rečima, pretvaraju u brojeve.
- Ovi brojevi sada mogu da se memorišu ili obrađuju u računaru kao bilo koji brojevi.
- Da bi se signal reprodukovao, brojevi se jednostavno pretvore nazad u električne signale. Kao i kod analogne tehnike, ovi signali se zatim odводе na zvučnike gde se pretvaraju u zvučne talase.



Slika 5.7. *Snimanje digitalnog signala*

Kod digitalnog procesa postoje dva očigledna nedostatka – prvo, on je mnogo komplikovaniji od analognog procesa; drugo, računari mogu da rade samo sa brojevima konačne rezolucije. Prema tome, gubi se potencijalna "beskonačna" rezolucija analognog signala.

5.3.4. Prednosti digitalne obrade signala

Očigledno, mora da postoje neke pogodnosti digitalnog procesa kada se on sve više i više koristi. Prvo, kada se signal pretvori u brojeve, signal je bezuslovno stabilan. Upotrebom raznih tehnika, kao što su detekcija i korekcija greške, moguće je memorisati, preneti i reprodukovati brojeve bez oštećenja. Dvadeseta generacija snimka je, prema tome, kvalitetna isto kao i prva.

Ova činjenica ima neke interesantne implikacije. Na primer, buduće generacije nikada neće znati kako su Bitlsi zaista zvučali. Raspoloživa analogna tehnologija šezdesetih godina dvadesetog veka jednostavno nije mogla precizno da snimi i reprodukuje signale. Nekoliko generacija analognih signala bilo je potrebno da bi se reprodukovao zvuk – prvo, trebalo je da se snimi tzv. master traka, a zatim da se miksuje i montira; od nje se pravila metalna matrica pomoću koje su se pravili plastični otisci (gramofonske ploče). Svaka etapa procesa imala je novu generaciju snimka, a svaka novo snimanje uticalo je na signal kao filter koji redukuje frekventni sadržaj i menja fazu signala.

Što se tiče muzičara, stvari danas izgledaju sasvim drugačije. Hiljadu godina od danas, naši potomci će moći verno da reprodukuju današnje digitalno pripremljene CD-ove. Sami kompakt diskovi mogu da se deformišu ili oštete, ali pre nego što se to desi, digitalni

brojevi na njima mogu da se kopiraju sa savršenom tačnošću na novi medijum. Signali koji su memorisani u digitalnom obliku su u stvari veliki nizovi brojeva. Kao takvi, oni su otporni na fizička ograničenja analognih signala.

Takođe, postoje i druge značajne prednosti digitalne obrade signala. Geofizičari su bili prvi koji su primenili tehnike obrade signala. Seizmički signali, koji su za njih interesantni, često su veoma niske frekvencije, od 0,01 Hz do 10 Hz. Teško je napraviti analogne filtre koji rade na ovim niskim frekvencijama. Vrednosti komponenti moraju biti toliko velike da je fizički nemoguće napraviti filter. Međutim, kada su signali pretvoreni u brojeve, sasvim je jednostavno napraviti program koji će obaviti filtriranje. Druga prednost digitalnih signala je njihova izdašnost. Na primer, DSP može da omogući veliki propusni opseg signalima koji treba da se šalju preko uskopojasnih kanala. Signal frekvencije 20 KHz može da se digitalizuje, a zatim šalje preko kanala od 5 KHz (učestanost nosioca). Signalu će biti potrebno četiri puta više vremena da se prenese preko uskopojasnog kanala, ali kada stigne na odredište, može da se rekonstruiše na svoj puni propusni opseg od 20 KHz.

Na isti način, sigurnost komunikacije može se znatno poboljšati preko DSP-a. Pošto se signal šalje kao niz brojeva, on može lako da se šifrira. Kada bude primljen na odredištu, brojevi se dešifruju i zatim reprodukuju kao originalni signal.

5.4. Standardi za digitalno prikazivanje

5.4.1. Rezolucija

U savremenim tehnološkim uslovima, Full HD (eng. *Full High Definition*) rezolucija se smatra standardom za video sadržaj visokog kvaliteta. Ova rezolucija se često označava kao 1080p, pri čemu se dimenzije prikaza izražavaju u pikselima (1.920×1.080 piksela), pri čemu prva vrednost predstavlja horizontalnu dimenziju, a druga vrednost predstavlja vertikalnu dimenziju.

Termin 1080p koristi se prilikom ukazivanja na sadržaj Full HD, gde broj 1080 ukazuje na broj vertikalnih linija piksela u sadržaju (tj. u slici), dok slovo p označava progresivno skeniranje sadržaja. Preciznije se može reći da slika u video sadržaju ima 1.920 horizontalnih i 1.080 vertikalnih piksela, te se može reći da se Full HD snimak sastoji od 2.073.600 piksela. Data rezolucija je visokog kvaliteta, što je čini pogodnom za gledanje sadržaja kao što su filmovi, televizijske emisije, sadržaj za prenos uživo (eng. *video streaming*) i druge slične vrste sadržaja.

Prethodna rezolucija od 1080p predstavlja standardnu definiciju 720p SD (eng. *Standard Definition*), čija je rezolucija 1.280×720 piksela. U poslednjim godinama kod korisnika širom sveta sve popularnije postaju rezolucije kao što su 4K ili 8K, koje nude mnogo viši kvalitet sadržaja u odnosu na 1080p.

5.4.2. Rezolucija 4K

U savremenoj tehnološkoj paradigmi, postoji respektabilna i unapređena rezolucija u odnosu na Full HD. Rezolucija poseduje dve dimenzije; dimenziju 3.840×2.160 pik-

sela (za standardne uređaje kao što su monitori i televizori) i 4.096×2.160 piksela za bioskopski format. Ova rezolucija pruža izuzetno ostru sliku koja, u suštini, sadrži četiri puta više piksela u odnosu na Full HD rezoluciju. Ova dimenzija omogućava prikazivanje detalja i boja značajno oštrijim u odnosu na manje formate. Pored toga, podržava HDR (eng. *High Dynamic Range*), koji suštinski proširuje opseg kombinovanja svetlosti sa tamnijim nijansama, postižući ravnotežu kontrasta i same slike.

5.4.3. Rezolucija 8K

Jedna od aktuelnih i zahtevnijih rezolucija koja se trenutno ističe je 8K, koja pruža superiornu gustinu piksela u odnosu na 4K rezoluciju. U poređenju sa Full HD, 8K rezolucija obezbeđuje video sadržaj sa osam puta većim brojem piksela. Numerička vrednost 8K rezolucije iznosi 7.680×4.320 piksela, zbog čega je sve više prisutna na modernim uređajima velikog formata. Ova rezolucija suštinski pruža visok stepen detaljnosti slike, omogućavajući izuzetno oštar prikaz video sadržaja. Međutim, zbog visokog kvaliteta video materijala, javlja se potreba za većim prostorom na hard disku. Takođe, reprodukcija ili prenos video sadržaja sa 8K rezolucijom može biti izuzetno zahtevna zbog potrebe za visokim performansama. Važno je napomenuti da 8K rezolucija još uvek nije postala standardna, te postoje različita ograničenja u pogledu platformi i uređaja koji pružaju podršku za ovakav tip video snimaka.

Ima veliki značaj naglasiti da na kvalitet video snimka, pored varijacija u raspoloživim rezolucijama, utiču i razni drugi faktori, uključujući: kvalitet snimanja, broj slika u sekundi (eng. *Frame Per Second – FPS*), kompresija koja se koristi, uređaj na kojem se prikazuje itd. Uzimajući u obzir ove faktore, neophodno je pažljivo proceniti sve aspekte koji utiču na kvalitet video snimka kako bi se postiglo optimalno vizuelno iskustvo tokom reprodukcije.

5.4.4. Rezolucija 10K

Jedna od trenutno dostupnih rezolucija koja se ističe je rezolucija 10K. Ova rezolucija se koristi za izuzetno široke ekrane koji prvobitno imaju format prikazivanja slike u srazmeri 21 : 9. Rezolucija 10K predstavlja izuzetno zahtevnu, kompleksnu i izrazito visoku rezoluciju video sadržaja. U horizontalnom pravcu, ova rezolucija može imati oko 10.000 piksela, u zavisnosti od specifičnog standarda koji se primenjuje. Navedena rezolucija je značajno veća od svojih prethodnika, 8K i 4K. Kod običnih korisnika ova rezolucija nema široku primenu, već se češće može sresti u bioskopima, profesionalnoj digitalnoj video produkciji i sličnim sredinama. Uređaji koji podržavaju prikazivanje ove rezolucije su još uvek retki, pa je za emitovanje video sadržaja u datoj rezoluciji potrebna tehnologija koja može pružiti adekvatnu podršku u smislu performansi prilikom reprodukcije.

5.4.5. Broj slika u sekundi

Proces emitovanja određenog broja slika (kadrova) u sekundi u video produkciji se definiše kao FPS (eng. *Frame Per Second*). FPS predstavlja brzinu prikazivanja slika u emitovanom video sadržaju, u jednoj sekundi. Navedeni izraz "brzina slika" se apsolutno koristi jednako

u računarskoj grafici, video produkciji, u sistemima za snimanje i detekciju pokreta i drugo. Važno je istaći da se brzina kadrova naziva i frekvencija slika (kadrova); jedinica mere se izražava u hercima (Hz).

Takođe, ima izuzetno visok stepen uticaja na korisničko iskustvo tokom gledanja video sadržaja. Važno je istaći da različite brzine video sadržaja značajno utiču na korisnička iskustva. Odabir odgovarajuće brzine uključuje više faktora, kao što je na primer da li je potrebno prikazati nešto realistično ili usporenog snimanja. Povećanjem broja FPS-a, video snimak dobija brži odziv i stvara utisak realističnosti pokreta. Na taj način veći broj prikazanih slika u sekundi doprinosi osećaju fluidnosti tokom gledanja video sadržaja.

Postoje uobičajene standardne brzine video sadržaja koje se često sreću, a koje su:

- 24 FPS – Brzina od 24 slike u sekundi (FPS) zastupljena je u kinematografiji i filmskoj industriji. Razlog leži u tome da navedena brzina približno odgovara prirodnom pokretu, odnosno, ljudskom oku. U modernoj kinematografiji se održava "standardan izgled filma" emulacijom sa kadrovima sa većim brojem slika u sekundi (npr. 60 ili 120 FPS-a).
- 25 FPS – Brzina od 25 slika u sekundi (FPS) predstavlja standardnu brzinu prikazivanja video sadržaja na mnogobrojnim TV sistemima. Kao i u prethodnom slučaju, ovaj format je zastupljen prilikom emitovanja različitih televizijskih formata (TV emisija, serija i drugih video formata), u cilju postizanja balansa između kvaliteta i visoke efikasnosti u pogledu iskorišćenosti resursa. Ovi se postiže da se kod gledaoca stiče fluidnost prilikom emitovanja sadržaja.
- 30 FPS – Brzina od 30 slika u sekundi (FPS) predstavlja uobičajenu brzinu prikazivanja video sadržaja, a koja je zastupljena u širokom spektru različitih striming platformi. Omogućava izuzetno visoko reprodukovanje pokreta u video sadržaju, te se time postiže efekat realistične vizualizacije objekata. U poređenju sa svojim prethodnicima, 30 slika u sekundi predstavlja značajno unapređenje u onosu na nešto niži broj slika u sekundi. Generalno se istakao kao podesan standard za žive TV programe i sportske događaje. Takođe, može se koristiti i za emitovanje televizijskih serija. U današnje vreme postao je standard i za snimanje video snimaka na pametnim telefonima. Naravno, ovaj segment je potrebno uzeti sa rezervom, u zavisnosti od brendova i njihovih podrazumevanih podešavanja.

Dodatno, za potrebe video igara, animacija i drugih vrsta video sadržaja, kao što su *JavaScript* animacije (*ThreeJS*, *PixiJS* i drugo) i drugi softverski grafički alati (na primer, *Blender*, *3DS Max* i drugo), mogu se koristiti više vrednosti frekvencije prikaza, poput:

- 50 FPS – Brzina od 50 slika u sekundi ima izuzetno jak uticaj na kontinualan prikaz pokreta. Povećanje učestalosti prikaza slika na 50 FPS, u odnosu na svoje prethodnike, postiže se preciznija reprodukcija pokreta u video sadržaju. Predstavlja duplu vrednost (*Double PAL*) u odnosu na svog para, tj. 25 FPS (PAL). Za države sa standardom napajanja 50 Hz, standard od 50 FPS-a je idealna brzina snimanja. U odnosu na svog para, snimci izgledaju izuzetno fluidnije. Ovakav format se koristi za snimanje usporenih snimaka. U svetu kinematografije smatra se da snimci snimani brzinom od 50 FPS-a i više, a da pri tome nisu usporeni, mogu izgledati vrlo

neprofesionalno. Shodno tome u zavisnosti od potreba i standarda u državi, koristi se odgovarajuća brzina. Nasuprot ovome, kada je u pitanju snimanje ekrana na računaru ili igranje video igre na računaru, ovaj standard može predstavljati bolji sveukupni doživljaj korisniku, pod uslovom da je napravljena kompatibilnost između osvežavanja monitora i broja slika u sekundi.

- 60 FPS – Brzina od 60 slika u sekundi pruža izuzetno visoku vizuelnu reprezentaciju dinamičnih akcija video sadržaja. Ovim standardnom se najčešće prikazuju izuzetno realistični video sadržaji ili se prikazuju računarski generisane animacije. Primeri takvih animacija mogu biti računarske video igre koja je kompatibilna sa osvežavanjem monitora. Takođe, ova brzina se često koristi i za usporeno snimanje, tj. usporeni pokret (eng. *slow motion*). U postprodukciji se video materijali snimljeni ovom brzinom usporavaju na svog para, tj. 30 FPS-a, kako bi se postigao efekat fluidnog usporenog snimanja.
- 100/120 FPS – Brzine od 100 ili 120 slika u sekundi predstavljaju standard u nekim državama (npr. 100 FPS-a u Velikoj Britaniji, dok je 120 FPS standard u SAD) ili više predstavljaju rezervisane kapacitete namenjene naprednim tehnologijama i specijalizovanim svrhama. Ove visoke brzine prikaza pružaju izuzetnu fluidnost. Njihova primena se posebno ističe u naprednim aplikacijama kao što su video igre, animacije i virtuelna stvarnost, gde je neophodan izuzetno precizan i realističan prikaz. Uz visok broj prikazanih kadrova u sekundi, ove brzine omogućavaju impresivno vizuelno iskustvo, obezbeđujući filmske sekvence usporenog snimanja različitih aktivnosti, a koje se izuzetno brzo odvijaju u realnom vremenu. To može biti primer snimanje kretanja automobila, atletičara i slično.

Može se izvesti zaključak da različite brzine mogu proizvesti drugačije iskustvo gledanja video sadržaja. Visoke brzine kadrova omogućavaju fluidan i realističan prikaz pokreta. Što je veći broj kadrova u sekundi, postiže se fluidnije prikazivanje pokreta, a samim tim postiže se i prikaz većeg broja detalja. Ipak, za mnoge holivudske reditelje brzina od 24 kadra u sekundi smatra se ukorenjenim standardom u 2D formatu. Kada je u pitanju 3D video produkcija u kinematografiji, bolje iskustvo je postiže se višim brzinama, poput brzine od 60 slika u sekundi.

Povećan broj frejmova u sekundi, tokom reprodukcije sadržaja, može pružiti korisniku unapređeno iskustvo pri gledanju video snimaka ili igranju video igara. Takođe, za video zapise iz prirode, više brzine ostvaruju iskustvo bliže stvarnosti. Ukoliko se želi postići realizam, onda je potrebno koristiti više brzine. Na primer, snimanje divljih životinja ili ineskata često nisu vidljive na brzinama od 24 FPS-a, te ih je neophodno snimiti na višim brzinama.

Važno je istaći da je ključna kompatibilnost između reprodukcije video sadržaja i prikaza istog. Konkretno, osvežavanje ekrana (eng. *refresh rate*) na kojem se prikazuje video snimak treba biti usklađen sa brzinom video sadržaja. Na primer, ukoliko je osvežavanje ekrana postavljeno na 60 Hz, najpodesniji broj FPS-a za prikazivanje bio bi 60, što omogućava potpunu fluidnost video sadržaja korisniku. Različiti standardi u video produkciji imaju različitu primenu. Takođe, odabir tehnologije, tj. brzine slika u sekundi bi trebalo da se zasniva na samom sadržaju i temi video snimka, kao i na umetničkoj slobodi.

5.4.6. Kodeci

Algoritam za kompresiju video zapisa, odnosno video kodek, predstavlja složen algoritam koji vrši odgovarajuću kompresiju nad video sadržajem. Cilj je brži i efikasniji prenos i prikaz video snimka, prilagođen uređaju na kojem se emituje. Kompresija se vrši odgovarajućim algoritmom kako bi se smanjila veličina video fajla i olakšalo deljenje putem različitih servisa.

U cilju postizanja kompresije video fajla, algoritam prvo utvrđuje redudansu, odnosno, suvišne informacije u video snimku. Koriste se tehnike kompresije radi efikasnog smanjenja viška podataka koji se nalaze u video fajlu. Količina podataka u ovom kontekstu odnosi se na podatke koji su bitni za reprodukciju video snimka. Nakon reprodukcije, aktivira se i koristi algoritam za dekompresiju kako bi se obnovili podaci u originalnom obliku.

Postoje kodeci sa gubicima, tj. algoritmi sa gubicima, i kodeci bez gubitaka, koji imaju suštinsku razliku i koriste se na različite načine. Kodeci sa gubicima komprimuju video podatke u cilju visoke kompresije video fajla, ali pri tome se gubi određeni broj podataka. S druge strane, kodeci bez gubitaka zadržavaju originalne informacije u video fajlu, ali ne postižu visoku kompresiju, već značajno nižu.

Ukratko, algoritmi sa gubicima i algoritmi bez gubitaka imaju svoje specifične primene u kompresiji video sadržaja. Izbor odgovarajućeg algoritma zavisi od potreba i zahteva za kompresijom i kvalitetom video materijala. U savremenim i poznatim kodecima koji su široko rasprostranjeni, kao što su H.264 (AVC), H.265 (HEVC), VP9, AV1, ProRes i drugi, može se primetiti različita karakteristika u vezi sa gubicima podataka.

H.264 kodek je izuzetno popularan algoritam sa gubicima. Njegova osnovna funkcija je kompresija visokokvalitetnih video snimaka i često se koristi na platformama za prenos video materijala uživo putem internet veze.

H.265 kodek je novija verzija prethodnog kodeka. Algoritam takođe vrši kompresiju video snimaka sa gubicima, ali nudi nešto bolju kompresiju u odnosu na H.264. Široko je rasprostranjen u izuzetno visokokvalitetnim video sadržajima i često se koristi u 4K video snimcima.

VP9 kodek, razvijen od strane kompanije *Google*, takođe koristi algoritam sa gubicima. Ovaj kodek je popularan na *Google* platformama, uključujući i *YouTube* za video snimke.

AV1 kodek takođe predstavlja algoritam sa gubicima. Osnovni karakteristika ovog kodeka je da je otvorenog standarda, pa se često koristi u video konferencijama i drugim platformama za prenos video sadržaja putem interneta.

ProRes kodek razvijen od strane kompanije *Apple*, ističe se svojom, gotovo, potpunom očuvanošću podataka, za razliku od prethodno pomenutih algoritama. Ovaj kodek se najčešće koristi u složenoj, kompleksnoj i profesionalnoj video produkciji, najčešće prilikom produkcije filmskog sadržaja.

5.5. Izrada video zapisa ili video klipa

Izrada video klipa (video zapisa) podrazumeva rad u tri opšte faze:

- **Preprodukcija** podrazumeva izradu skripta, vizuelizaciju scena skiciranjem u stripu knjige snimanja, kao i izradu produkcijskog rasporeda za snimanje pojedinih scena.
- **Produkcija** podrazumeva snimanje scena.
- **Postprodukcija** podrazumeva montiranje najboljih scena u konačni video program, korekciju i unapređenje video i audio zapisa prema potrebi. Montaža podrazumeva prvu verziju ili **grubu montažu** (*rough cut*) u kojoj može da se dobije okvirna ideja o mogućnostima koje autoru stoje na raspolaganju sa klipovima koje poseduje. U nastavku montaže video program će moći fino da se podešava uzastopnim intervencijama sve dok se ne dođe do konačnog rezultata. U tom trenutku nastaje **završni rez** (*final cut*). Programi za nelinearnu montažu su osmišljeni tako da obezbeđuju efikasno montiranje, korekciju i poboljšavanje izgleda klipova, čime predstavljaju vrlo dragocenu alatku u postprodukciji.

Očekuje se da korisnik svoje projekte realizuje samostalno, međutim ako projekat u bilo kom trenutku zahteva spoljne saradnike, kao što su, na primer, usluge postprodukcije, pre nego što uopšte započne dati projekat, najbolje je da se posavetuje sa njima. Oni mogu da pomognu da korisnik utvrdi koje parametre treba da koristi u različitim fazama projekta, kao i da pomognu da izbegne eventualne greške koje ga mogu koštati vremena i novca. Na primer, ako se izrađuje video klip za televizijsko emitovanje, potrebno je da se zna da li se isti izrađuje za standard NTSC, PAL ili SECAM.

Međutim, pre nego što se upusti u avanturu zvanu digitalni video, poželjno je da se upozna sa nekim pojmovima iz te oblasti i njihovim značenjem.

5.5.1. Merenje vremena u video klipu

U stvarnom svetu vreme se oseća kao stalni tok događaja. Međutim, u toku rada na video klipu zahteva se precizna sinhronizacija, tako da se vreme mora meriti korišćenjem preciznih brojeva. Uobičajene merne jedinice kao što su časovi, minuti i sekunde nisu dovoljno precizni za video montažu, pošto se u jednoj sekundi može naći više od jednog događaja. Video profesionalci mere vreme korišćenjem standardnih postupaka koji broje delove jedne sekunde u slikama.

5.5.2. Kako vremenska baza i brzina slika utiču jedna na drugu?

Određivanjem **vremenske baze** (*timebase*) nekog projekta određuje se način merenja vremena u njemu. Na primer, vremenska baza od 30 označava da je svaka sekunda podeljena na 30 jedinica. Tačno vreme u kome dolazi do montiranja zavisi od navedene vremenske baze, pošto do montiranja može doći samo u vremenskim jedinicama. Korišćenje različitih vremenskih baza dovodi do toga da vremenske jedinice nastupaju na različitim mestima. Vremenski intervali u izvornom klipu uslovljeni su **izvornom brzinom slika** (*source frame rate*). Na primer, kada se izvorni klipovi dobijaju korišćenjem kamere čija je brzina 30 slika u sekundi, to znači da kamera beleži radnju tako što snima po jednu sliku u svakom tridesetom delu jedne sekunde. Treba imati u vidu da sve ono što se dešavalo između ovih intervala u jednoj sekundi nije zabeleženo. Odatle manja brzina slika, na primer 15 fps

(*frames per second*) beleži manje informacija o nekoj radnji koja traje, dok veća brzina slika, na primer 30 fps, beleži više ovakvih informacija.

Određivanjem **brzine slika datog projekta** (*project frame rate*) određuje se koliko često aplikacija generiše slike iz projekta. Brzina slika nekog projekta od 30 slika u sekundi označava da će aplikacija izraditi 30 slika na osnovu svake sekunde projekta.

Da bi se postigla fina i konzistentna reprodukcija, vremenska baza, izvorna brzina slika i brzina slika datog projekta treba da budu istovetne.

Tabela 5.1. *Vrste video montaža i njihove brzine*

Vrsta video montaže	Slika u sekundi
Igrani film	24 fps
PAL i SECAM video klip	25 fps
NTSC video klip	29,97 fps
Veb ili CD-ROM	15 fps
Druge vrste video klipova, kao što su: montiranje bez odbacivanja slika, ED animacija	30 fps

Napomena: NTSC sistem je prvobitno izrađen za crno-bele slike brzine 30 fps, ali je, da bi se udovoljilo slikama u boji, modifikacijom signala sredinom dvadesetog veka standard NTSC izmenjen u 29,97 fps.

Ponekad se vremenski sistemi ne podudaraju. Na primer, od autora može de se zatraži da izradi video klip predviđen za distribuciju na kompakt diskovima u kome treba kombinovati izvorne klipove igranog filma koji su digitalizovani na 24 fps sa izvornim video klipovima digitalizovanim na 30 fps, korišćenjem vremenske baze od 30 da bi se dobila konačna brzina slika za kompakt disk od 15 fps. Kada se neke od ovih vrednosti ne podudaraju, matematička nužnost je da se pojedine slike ponavljaju ili izostavljaju. Dobijeni efekat može biti iritirajući ili neprimetan zavisno od razlika između vremenske baze i brzine slika koja je korišćena u projektu.

5.5.3. Merenje vremena korišćenjem vremenskog koda

Vremenski kôd (*timecode*) definiše način brojanja slika i odražava se na način prikazivanja vremena u nekom projektu, kao i njegovog određivanja. Vremenski kôd nikada ne menja vremensku bazu ili brzinu slika nekog klipa ili projekta, već jedino menja način na koji se ove slike broje.

Stil vremenskog koda određuje se na osnovu medijuma koji je najznačajniji za projekat. Kada se video klip montira za emitovanje na televiziji, način brojanja slika razlikuje se od načina brojanja slika prilikom montiranja video klipa za igrani film. Video aplikacije obično prikazuju vreme korišćenjem vremenskog koda za video SMPTE (*Society of Motion Picture and Television Engineers*), gde trajanje od 00 : 06 : 51 : 15 označava da je u toku 6. minut, 51. sekunda i 15. slika datog klipa. U bilo kom trenutku autor može da se prebaci na neki drugi sistem prikazivanja vremena, kao što su stope i slike u filmu od 16mm ili 35mm. Profesionalni uređaji za reprodukciju video traka i kamkorderi mogu da pročitaju

i upišu vremenski kôd direktno na video traku, što autoru omogućava da zvuk, video i montirani materijal sinhronizuje ili da montira van mreže.

Prilikom korišćenja vremenske baze standarda NTSC od 29,97, razlika između ove vremenske baze i vremenskog koda od 30fps dovodi do odstupanja između prikazanog trajanja datog programa i njegovog stvarnog trajanja. Iako je ovo neslaganje u početku neznatno, u toku trajanja programa ono se vremenom uvećava čime autoru onemogućava da precizno izradi program određene dužine. **Vremenski kôd sa odbacivanjem slike** (*drop-frame timecode*) predstavlja SMPTE standard za video od 29,97 koji eliminiše ovu grešku čime čuva tačnost NTSC vremena.

5.5.4. Televizijska slika

Slika na televiziji ili na računarskom monitoru sastoji se od horizontalnih linija. Ove linije mogu da se prikažu na više načina. U prikazu većine personalnih računara koristi se **progresivno skeniranje** (*progressive scan*) ili prikaz **bez preplitanja** (*Non-interlaced*) u kome su linije na nekoj slici prikazane odjednom odozgo nadole pre nego što se pojavi naredna slika. Televizijski standardi kao što su NTSC, PAL ili SECAM su **sa preplitanjem** (*interlaced*) pri čemu je svaka slika podeljena na dve **poluslike** (*field*). Svaka poluslika sadrži svaku drugu horizontalnu liniju na datoj slici.



Slika 5.8. Video sa preplitanjem predstavlja sliku u dva prolaza

TV prikazuje prvu polusliku naizmeničnih linija preko čitavog ekrana, a zatim prikazuje drugu polusliku i popunjava naizmenične procepe koje je napravila prva poluslika. Jedna slika u video standardu NTSC koja je prikazana približno na svaki trideseti deo sekunde sadrži dve prepletene poluslike, pri čemu se svaka prikazuje približno na svaki šezdeseti deo sekunde. Slike u video standardu PAL i SECAM prikazuju se približno na svaki dvadeset peti deo sekunde i sadrže dve prepletene poluslike, pri čemu se svaka prikazuje približno na svaki

pedeseti deo sekunde. Poluslika koja sadrži prvu ekransku liniju na vrhu date slike naziva se **gornja poluslika** (*upper field*), a druga poluslika se naziva **donja poluslika** (*lower field*). Prilikom reprodukcije ili izvoženja u video sa preplitanjem trebalo bi nastojati da se navedeni **redosled poluslika** podudara sa prijemnim video sistemom, jer će u suprotnom pokreti biti isprekidani, a ivice objekta na slici poprimiće izgled češlja.

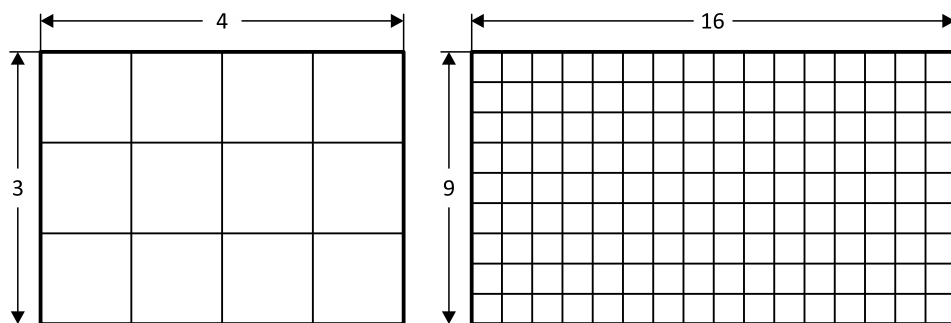
Napomena: Kada je reč o analognom videu, redosled poluslika mora da se podudara sa redosledom poluslika kartice za digitalizaciju (koji mora biti naveden u unapred definisanim parametrima). Kada je reč o digitalnom videu (DV), donja poluslika uvek dolazi prva u redosledu poluslika.

5.5.5. Merenje veličine slike i rezolucije

Prilikom digitalnog montiranja video klipova važno je imati u vidu nekoliko svojstava vezanih za **veličine slike** (*frame size*) – piksel i proporciju (razmeru) slike, rezoluciju klipa, veličinu slike u projektu i dubinu bita. **Piksel** (element slike) predstavlja najmanju jedinicu koja se može koristiti za izradu neke slike. Ništa što je manje od jednog piksela ne može precizno da se prikaže.

5.5.6. Proporcija

Proporcija (*aspect ratio*) neke slike označava odnos njene širine i njene visine. Na primer, proporcija video zapisa u NTSC standardu iznosi 4 : 3, dok se u video formatima DVD, HDTV i na nekim slikama u igranom filmu koristi produžena proporcija od 16 : 9.

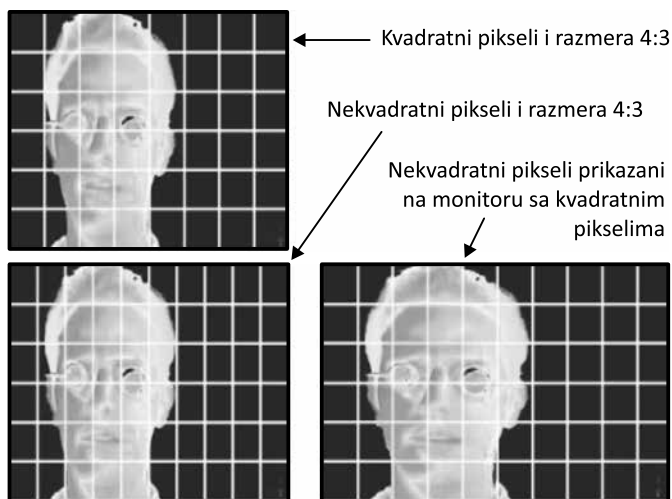


Slika 5.9. Slika sa proporcijom 4:3 (levo) i slika sa proporcijom 16:9 (desno)

U nekim video formatima koristi se različita proporcija (razmera) za piksele koji sačinjavaju datu sliku. Kada se video klip u kome su korišćeni **nekvadratni pikseli** (*non-square pixels*) (pikseli koji su duži nego širi, odnosno širi nego duži), prikaže na sistemu zasnovanom na kvadratnim pikselima, kao i obrnuto, dolazi do istezanja oblika i pokreta.

5.5.7. Nekvadratni pikseli

Većina programa za nelinearnu montažu ima podršku za različite razmere nekvadratnih piksela, uključujući i DV-ovu razmeru *Widescreen* (*Cinema*) za DV od 16 : 9, kao i razmeru piksela *Anamorphic* od 2 : 1.



Slika 5.10. *Kadar sa različitim pikselima*

Kada na ekranu svog računara korisnik gleda video klip sa nekvadratnim pikselima, aplikacija za montažu prikazuje odgovarajuću razmeru na monitoru računara kako slika ne bi bila izobličena. U parametrima za pokret i neprovidnost, kao i kod geometrijskih efekata takođe se koristi odgovarajuća razmera, tako da nema izobličenja posle montiranja ili generisanja videa.

5.5.8. Veličina slike

U programima za montažu po pravilu može da se definiše **veličina slike** za reprodukciju videa, a prema potrebi i za izvoženje videa u datoteku. Veličina slike iskazuje se horizontalnim i vertikalnim iznosima u pikselima. Na primer, 640 sa 480 piksela. Kod digitalne video montaže veličina slike označava se i kao **rezolucija**.

Uopšteno govoreći, veća rezolucija značajnije čuva detalje na slici i prilikom montiranja zahteva veću količinu memorije (RAM) i prostora na disku. Povećanjem dimenzija slike povećać se broj piksela koje aplikacija mora da obradi i sačuva za svaku sliku, tako da je vrlo važno znati koja se rezolucija zahteva za konačni video format. Na primer, slika u NTSC formatu čije su dimenzije 720 sa 480 piksela (DV standard) sadrži 345.600 piksela, dok slika u PAL formatu čije su dimenzije 720 sa 576 piksela sadrži 414.720 piksela. Ako se odredi suviše mala rezolucija, slika će izgledati grubo, a pikseli će biti uočljivi. Ako se odredi suviše velika rezolucija, upotrebiće se više memorije nego što je zaista neophodno. Prilikom promene veličine slike treba nastojati da dimenzije budu u skladu sa proporcijama izvornog video klipa.

5.5.9. Preskeniranje i bezbedne zone

Ako se video klip priprema za emitovanje na televiziji, veličina slike može da odvede autora u pogrešnom pravcu. Većina televizijskih prijemnika gledalaca u NTSC sistemu uve-

čava sliku. Na taj način spoljne ivice slike izlaze sa ekrana. Ovaj postupak se naziva **preskeniranje** (*overscan*). Kako stepen preskeniranja nije konzistentan na svim televizijama, poželjno je radnju i potpise (titlove) smestiti unutar dve bezbedne površine – zone bezbedne za radnju i titlove.



Slika 5.11. Zona bezbedna za akciju i zona bezbedna za potpis

Zona bezbedna za radnju je površina koja je približno za 10% manja od stvarne veličine slike. Zona bezbedna za titlove je približno za 20% manja od stvarne veličine slike. Da bi autor bio siguran da će svi značajni elementi video zapisa biti u potpunosti prikazani, treba da nastoji da se celokupna značajna radnja smesti unutar zone bezbedne za radnju, kao i da se sav tekst i važni grafički elementi nađu unutar zone bezbedne za titlove. Na taj način će izbeći izobličenja teksta i grafičkih objekata do kojih dolazi na ivicama mnogih televizijskih prijemnika. Moguće preskeniranje autor može uvek da predvidi korišćenjem bezbednih zona, pri čemu važna radnja i tekst ostaju unutar njih, a video klip bi trebalo isprobati na pravom televizijskom prijemniku.

5.5.10. Dubina bita

Verovatno je većina upoznata sa činjenicom da **bit** predstavlja osnovnu mernu jedinicu za čuvanje informacija u računaru. Što je veći broj bitova koji se koriste za opisivanje nečega, to će dati opis biti verniji. **Dubina bita** (*bit depth*) označava broj bitova koji su izdvojeni za opisivanje boje jednog piksela. Što je dubina bita veća, slika može da sadrži veći broj boja, a to omogućava precizniju reprodukciju boja i veći kvalitet slike. Na primer, slika koja čuva 8 bitova po slici (8-bitna slika) može da prikaže 256 boja, a 24-bitna slika može da prikaže više od 16 miliona boja.

U aplikacijama za montažu obično može da se koristi video klip koji je digitalizovan i sačuvan korišćenjem bilo koje dubine bita, ali one, uglavnom, klipove interno konvertuju u format boje YUV koji se koristi na televiziji. Format YUV može da sačuva kvalitetan video zapis korišćenjem 16 bitova po pikselu (format boje RGB zahteva 24 bita po pikselu). Time se postižu veći kvalitet i brzina obrade.

Prostor boje YUV osmišljen je ranih pedesetih godina dvadesetog veka kao način da se televizijskom programu obezbedi boja, a da se pri tom očuva kompatibilnost unazad sa crno-belim parovima i da se pri tom ograniči veličina tzv. propusnog opsega koji je neophodan za signal. U sistemu YUV, tri komponente boje sistema RGB matematički se konvertuju u svetlosni signal (tj. u crno-belu boju) plus u dva signala u boji. S obzirom na način na koji se izvodi ova konverzija, količina podataka za celokupni signal se znatno smanjuje, a da se pri tom ne gubi na kvalitetu.

Ako se video klip priprema za NTSC standard, trebalo bi da se ima na umu da je opseg boja NTSC standarda ograničen načinom na koji se prenose signali boja bez obzira na to što formati 16-bitni YUV i 24-bitni RGB obezbeđuju pun opseg boja. NTSC ne može precizno da reprodukuje zasićene boje i nežne prelive boja. Najbolji način da se spreče eventualni problemi kod boje u NTSC standardu, sastoji se u tome da video klip u toku montaže treba pregledati na pravilno kalibrisanom NTSC monitoru.

5.5.11. Razumevanje komprimovanja video podataka

Montiranje digitalnog video klipa podrazumeva čuvanje, premeštanje i izračunavanje izuzetno velikih količina podataka u poređenju sa drugim vrstama datoteka. Mnogi personalni računari, a posebno stariji modeli nisu adekvatno opremljeni za obrađivanje **podataka koji se prenose velikom brzinom** (količina video informacija koje se obrađuju u sekundi) i **velikih datoteka** koje sadrže nekomprimovane video zapise. Da bi se smanjila brzina prenosa podataka digitalnog video klipa i da bi bila svedena na meru koju računar može da obradi, potrebno je da se koristi **komprimovanje**.

Prilikom digitalizacije izvornog videa, pregledanja montiranog materijala, reprodukcije i kreiranja završnog materijala vrlo su značajni parametri za komprimovanje. Navedeni parametri neće biti isti u svim situacijama:

- Dobra je zamisao komprimovati video klip koji dolazi u računar. Cilj je da za potrebe montiranja očuva što bolji kvalitet slike, a da se pri tom brzina podataka održi u granicama mogućnosti računara.
- Video klip koji izlazi iz računara treba takođe komprimovati. Trebalo bi pokušati da se postigne što bolji kvalitet slike prilikom reprodukcije. Ako se izrađuje video traka, neka brzina podataka ostane u granicama računara koji će reprodukovati video zapis na video traku. Ako se izrađuje video zapis koji će biti reprodukovan na nekom drugom računaru, neka brzina podataka ostane u granicama mogućnosti planiranih modela računara. Ako se izrađuje video klip sa neometanim tokom podataka (*streamed*) sa veb servera, neka brzina podataka bude odgovarajuća za distribuciju na internetu.

- Primenjivanje parametara za komprimovanje predstavlja vrlo osetljivo pitanje, a izbor najboljih parametara može da se razlikuje od projekta do projekta. Ako se autor odluči za komprimovanje koje je suviše malo, brzina podataka će biti prevelika za dati sistem što će dovesti do grešaka, poput odbačenih slika. Ako se odluči za stepen komprimovanja koji je prevelik, čime se brzina podataka znatno smanjuje, neće moći u potpunosti da iskoristi mogućnosti svog sistema što će nepotrebno ići na uštrb kvaliteta slike.

Napomena: DV poseduje fiksnu brzinu podataka od 3,5 megabajta u sekundi, nominalno 25 megabita po sekundi. Stepenn standardnog komprimovanja za DV je 5 : 1.

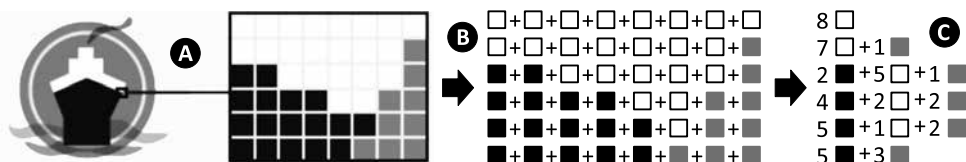
5.5.12. Izbor metoda za komprimovanje video zapisa

Sušтина komprimovanja podataka sastoji se u tome da se isti sadržaj predstavi sa što manje podataka. Da bi se upravljalo komprimovanjem na odgovarajući način, potrebno je da se navede kompresor/dekompresor, odnosno, **kodek**. Kodek može da koristi jednu ili više strategija za komprimovanje, pošto nijedna metoda sama po sebi ne predstavlja najbolje rešenje za sve moguće situacije. Ovde su opisane uobičajene strategije za komprimovanje koje koristi kodek, kao i vrste video klipova za čije komprimovanje su predviđene.

5.5.13. Prostorno komprimovanje

Prostorno komprimovanje (*spatial compression*) pronalazi načine za komprimovanje neke slike utvrđivanjem šablona i ponavljanja među pikselima. Na primer, umesto da opisuje svaki od nekoliko hiljada piksela na nekoj slici plavog neba, prostorno komprimovanje može da zabeleži znatno kraći opis, kao što je recimo ovaj: "Svi pikseli na ovoj površini su svetloplavi". **Kodovanje sa promenljivom dužinom kodne reči** (*run-length encoding*) predstavlja verziju ovog postupka koju koriste mnogi kodeci. Kodeci koji koriste prostorno komprimovanje, kao što su *QuickTime Animation* ili *Microsoft RLE* funkcionišu dobro na video klipovima koji sadrže velike površine jednoličnih boja, kao što je recimo animacija u crtanom filmu.

U načelu, povećanjem prostornog komprimovanja smanjuje se brzina podataka i veličina datoteke, a slika gubi oštrinu i definiciju. Međutim, korišćenjem pojedinih vrsta metode kodiranja promenljivom dužinom kodne reči u potpunosti se čuva kvalitet slike, ali se zahteva veća snaga za obrađivanje.



Slika 5.12. Primer prostornog komprimovanja

Kao što je prikazano na slici 5.12, digitalne slike se sastoje od određenog broja piksela (na slici označeno slovom A), koji zauzimaju mnogo prostora na disku ako nisu komprimovani

(označeno slovom B na slici); primenjivanjem metode kodovanja sa promenljivom dužinom kodne reči isti podaci na slici zauzimaju manje mesta (označeno slovom C na slici).

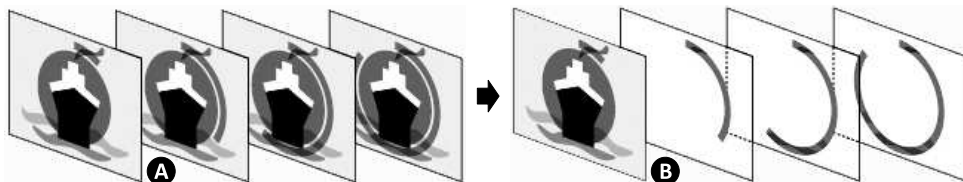
5.5.14. Vremensko komprimovanje

Vremenskim komprimovanjem sažimaju se izmene u sekvenci slika utvrđivanjem šablona i ponavljanja u toku vremenskog trajanja. U pojedinim video klipovima, kao što je recimo klip televizijskog spikera, temporalno komprimovanje će uočiti da se iz slike u sliku jedino menjaju oni pikseli koji formiraju lice spikera. Svi ostali pikseli se ne menjaju (kada je kamera nepokretna). Umesto da opisuje svaki piksel na svakoj slici, temporalnim komprimovanjem se opisuju svi pikseli na prvoj slici, a zatim za svaku sledeću sliku opisuju samo oni pikseli koji su različiti u odnosu na prvu sliku. Ovaj postupak se naziva **utvrđivanje razlike među slikama** (*frame differencing*).



Slika 5.13. U ovoj animaciji jedina izmena je krug koji se pomera oko broda

Kada se na nekoj slici veliki broj piksela razlikuje u odnosu na prethodnu sliku, preporučljivo je da se cela slika ponovo opiše. Svaka cela slika se naziva ključna slika koja uspostavlja novo polazište za utvrđivanje razlike među slikama. Aplikacije za montažu obično omogućuju da na neki način autor kontroliše način izrade ključnih slika.



Slika 5.14. Primer vremenskog komprimovanja

Kao što je prikazano na slici 5.14, klip beleži sve piksele na svim slikama (na slici označeno slovom A); primenjivanjem pomenutog komprimovanja izrađuje se ključna slika, a za naredne slike beleže se jedino izmene (označeno slovom B na slici).

Ako za kodek ne mogu da se odrede ključne slike, to znači da on najverovatnije i ne koristi temporalno komprimovanje. Temporalno komprimovanje pruža najbolje rezultate kada se velike površine u video klipovima ne menjaju, a mnogo je manje delotvorno kada se slika stalno menja, kao recimo u muzičkim video klipovima.

5.5.15. Komprimovanje sa očuvanjem kvaliteta

Neki kodeci koriste **komprimovanje sa očuvanjem kvaliteta** (*lossless*; bez gubitaka) koje obezbeđuje da se sve informacije – a samim tim i sav kvalitet – u izvornom klipu

sačuvaju posle komprimovanja. Međutim, očuvanje izvornog nivoa kvaliteta predstavlja ograničavajući faktor za stepen do koga se brzina podataka i veličina datoteke mogu smanjiti, a brzina podataka koja se dobija kao rezultat može biti prevelika za finu reprodukciju. Kodeci koji koriste ovaj metod komprimovanja, kao što je recimo *Animation* (parametar *Best quality*) upotrebljavaju se za očuvanje maksimalnog kvaliteta u toku montaže ili za zamrznute slike gde brzina podataka ne predstavlja problem.

Napomena: Da bi se obezbedila fina reprodukcija, za video klip koji sadrži sve slike u punoj veličini se u slučaju korišćenja ovog komprimovanja zahteva veoma veliki disk i vrlo brz računar koji je predviđen za brz protok podataka.

5.5.16. Komprimovanje sa narušavanjem kvaliteta

Većina kodeka koristi **komprimovanje sa narušavanjem kvaliteta** (*lossy*, sa gubitkom) koje prilikom komprimovanja odbacuje neke od izvornih podataka. Na primer, ako pikseli koji sačinjavaju nebo, zapravo sadrže 78 nijansi plave boje, kodek koji koristi ovu metodu komprimovanja može da zabeleži 60 nijansi plave boje. Iako sa jedne strane komprimovanje sa narušavanjem kvaliteta predstavlja svojevrstan kompromis u odnosu na kvalitet, ono kao rezultat nudi manje brzine podataka i veličine datoteka u odnosu na komprimovanje sa očuvanjem kvaliteta. U tom smislu su kodeci koji koriste ovaj metod komprimovanja, kao što su *Cinepak* ili *Sorenson Video* široko prihvaćeni u završnoj produkciji video klipa koji se isporučuju posredstvom kompakt diskova ili interneta.

5.5.17. Asimetrično i simetrično komprimovanje

Izabrani kodek bitno utiče na tok rada u produkciji, ne samo u pogledu veličine datoteke ili brzine reprodukcije, već i u pogledu vremena koje se zahteva da bi kodek komprimovao dati broj slika. Brza komprimovanja ubrzavaju video produkciju, a brza dekomprimovanja olakšavaju pregledanje, međutim za mnoge kodeke potrebno je mnogo više vremena za komprimovanje slika nego što je potrebno za njihovo dekomprimovanje u toku reprodukcije. U tom smislu je klip od 30 sekundi potrebno nekoliko minuta obrađivati pre same reprodukcije. Kodek se smatra **simetričnim** onda kada zahteva podjednaku količinu vremena za komprimovanje i dekomprimovanje nekog klipa. Kodek je **asimetričan** onda kada se vreme potrebno za komprimovanje razlikuje od vremena potrebnog za dekomprimovanje.

Komprimovanje video klipa je baš kao pakovanje kofera. Stvari se mogu zapakovati podjednako brzo kao i kod raspakivanja ako se jednostavno pobacaju u kofer, međutim ako se odvoji malo više vremena i lepo se složi i organizuje garderoba, u isti kofer će stati mnogo više stvari.

5.5.18. DV komprimovanje

DV je format koji koristi veliki broj digitalnih video kamera. DV, takođe, označava vrstu komprimovanja koju ovi kamkorderi koriste, a kojim se video zapis komprimuje u samoj kameri. U uobičajenoj formi DV komprimovanja koristi se fiksna brzina podataka od 25

megabita u sekundi (3,5 megabajta u sekundi) i stepen komprimovanja od 5 : 1. Ovo komprimovanje se naziva **DV25**. U većini video aplikacija postoji generička podrška za DV25 i druge DV kodeke, a izvorni digitalni video klip može da se pročita bez dodatnih konverzija.

Ni za jedan jedini kodek ne može da se kaže da je idealan za sve situacije. Kodek koji se koristi za izvoženje dovršenog programa mora biti dostupan širokom auditorijumu. Tako da iako je specijalizovani kodek koji dolazi u sastavu specifične kartice za digitalizaciju najbolje rešenje za digitalizovanje izvornih klipova, on svakako nije najbolje rešenje za izvoženje klipova, pošto je malo verovatno da će svaki pripadnik najšireg auditorijuma imati baš tu specifičnu karticu i specijalizovani kodek koji dolazi uz nju. Ovo je naročito značajno prilikom izvoženja tzv. striming medijuma (sa neometanim tokom podataka), pošto tri najpopularnije arhitekture za neometani tok podataka (*RealMedia*, *Windows Media* i *QuickTime*) koriste sopstvene kodeke u svojim programima za reprodukciju. Na primer, tok podataka *RealMedia* ne može uvek da se reprodukuje posredstvom programa za reprodukciju *Windows Media* i obrnuto. Da bi se izašlo u susret auditorijumu koji poseduje najrazličitije programe za reprodukciju u svojim pretraživačima veba, medijum sa neometanim tokom podataka obično se kodira u više formata.

5.5.19. Digitalno presnimavanje video zapisa

Da bi autor mogao da montira video program, svi izvorni klipovi moraju biti trenutno dostupni sa diska, a ne sa video trake. Izvorni klipovi se u računar uvoze sa izvornih video traka korišćenjem postproduksijske metode koja se naziva **digitalno presnimavanje video zapisa** (*video capture*). Shodno tome, na svom disku autor mora da ima dovoljno prostora da bi mogao da sačuva sve klipove koje želi da montira. Da bi uštedeo prostor na disku, trebalo bi da digitalizuje samo one klipove za koje zna da će ih koristiti.

Izvorni klipovi postoje u dva osnovna oblika:

- **Digitalni medijumi** – Ova vrsta medijuma je već izražena u obliku digitalnog formata koji računar može da pročita i direktno obradi. Mnogi noviji modeli kamkordera digitalizuju i čuvaju video zapis u digitalnom formatu direktno u kameri. Ovakvi kamkorderi koriste jedan od mnogobrojnih DV formata koji na izvorni materijal primenjuju standardnu količinu komprimovanja. Digitalne izvorne datoteke koje su sačuvane na DV traci ili nekom drugom digitalnom medijumu moraju da se **digitalno presnime** (prenesu) na pristupačan disk pre nego što se budu koristile na računaru u nekom video projektu. Najjednostavniji način da se digitalizuje DV sastoji se u tome da se DV uređaj, recimo kamkorder ili magnetoskop, (*deck*) poveže sa računarem posredstvom porta IEEE 1394 (poznatog i kao *FireWire* ili *i.Link*). Kada je reč o prefinjenijim zahtevima digitalizacije, neophodno je koristiti specijalizovane **DV kartice za digitalno presnimavanje**. Aplikacije za montažu podržavaju širok raspon DV uređaja i kartica za digitalizaciju čime se olakšava postupak digitalizacije DV izvornih datoteka.
- **Analogni medijumi** – Ova vrsta medijuma se mora **digitalizovati**. To znači da ona mora biti konvertovana u digitalni oblik i sačuvana u digitalnom formatu da bi

računar mogao da je sačuva i obradi. Klipovi sa **analognih** video traka, igrani film, konvencionalne audio trake i zamrznute slike kontinuiranih tonova (recimo slajdovi) predstavljaju primere analognih medijuma. Kada se analogni video uređaj (kao što je recimo analogna video kamera ili magnetoskop) u kombinaciji sa odgovarajućom **karticom za digitalizaciju** poveže sa računarom, aplikacija za montažu može da digitalizuje, komprimuje i prenese analogni izvorni materijal na disk u formi klipova koji se potom mogu dodati digitalnom video projektu.

Napomena: Hardverska oprema za digitalizaciju video zapisa nije uvek ugrađena u personalne računare, tako da se najčešće mora dodati sistemu instaliranjem kompatibilne kartice za digitalizaciju.

5.6. Komprimovanje AV podataka i optički medijumi

Prilično često se sreće termin **komprimovanje**, pa je vreme da se detaljno objasni i da se pokaže zašto je značajan. Komprimovanje bi najjednostavnije moglo da se definiše kao **sabijanje**. Treba zamisliti neki objekat kao što je hamburger. Pod hamburgerom se obično podrazumeva nekoliko različitih sastojaka kao što su zelena salata, krompirići, meso i zemičke. Kvalitetan hamburger ume da bude toliko veliki da ga je nekada nemoguće zagristi, a da se pre toga ne pritisne, kako bi uopšte mogao da stane u usta. Moglo bi se reći da to predstavlja neku vrstu komprimovanja koja prethodi žvakanju. Sastojci se sabijaju na različite načine – lepinja u sebi sadrži dosta vazduha tako sa da njom ide veoma lako. Međutim, meso može da se komprimuje samo malo zato što je već sabijeno u samu pljeskavicu. Sve u svemu, korisnik na kraju dobija manji hamburger u kome su neke komponente sabijene manje, a neke više.

Komprimovanje karakteristično za računare zasniva se na istim principima. Tekstualne datoteke su prepune "međuprostora" tako da se veoma uspešno komprimuju, tj. nakon sabijanja može se dobiti datoteka koja je 90% manja, što znači da je komprimovanje izuzetno veliko. Neki drugi tipovi datoteka kao što su video ili fotografije manje se komprimuju, jer se one nalaze u formatu koji je već komprimovan.

5.6.1. Zašto je digitalni video komprimovan?

Možda zvuči čudno, ali jedan od razloga zbog kojih je digitalni video komprimovan je taj što ljudska čula to ne primećuju. Poznato je da ljudsko uho može da registruje samo zvuke između 20 i 20.000 Hz. Slično tome, ljudsko oko može da registruje otprilike 1.024 nijanse iste boje, od milijardi boja do kojih je moguće doći matematičkim putem. Prema tome, ima li smisla zadržati baš sve boje ako naše oko ionako nije u stanju da ih razlikuje? Pored toga postoje i redundantne ili suvišne slike – ako se u svakoj sekundi nekog videa od 60 sekundi na istom mestu pojavljuje ista stolica, zašto bi se podaci o njoj snimali u svakom kadru video zapisa. Drugi razlog za postojanje komprimovanja jeste prostor koji je neophodan za skladištenje digitalnog videa. Veličina kadra K se određuje na osnovu sledećeg obrasca:

$$K = \frac{\text{širina u pikselima} \times \text{dužina u pikselima} \times \text{dubina bitova}}{8 \times 1.024}.$$

Ako se primeni gornji obrazac za jedan kadar nekomprimovanog video materijala rezolucije 720×576 , dobija se:

$$K = \frac{\frac{720 \times 576 \times 24}{8}}{1.024} = \frac{720 \times 576 \times 24}{8 \times 1.024} = 1.215.$$

Ovaj obrazac govori da samo jedan kadar sirovog, nekomprimovanog video materijala u standardnoj digitalnoj video rezoluciji (720×576) zauzima preko 1 Mb prostora. Pošto video snimci obično imaju oko 25 kadrova u svakoj sekundi, ovde se govori o preko 30 Mb prostora po svakoj sekundi (Mbps) video snimka. Za 60 minuta videa to je oko 120 Gb prostora. Kada se tome doda zvuk, ovaj broj je još veći. Drugim rečima, bez komprimovanja bi rad sa digitalnim videom bio izuzetno težak ili praktično nemoguć.

5.6.2. Komprimovanje bez gubitaka

Pod terminom **bez gubitaka** ovde se podrazumeva da nema i ne postoji gubitak podataka. Kada se neka datoteka komprimuje bez gubitaka, to znači da su apsolutno svi podaci još uvek tu. Dobar primer je zip format *Microsoft Word* dokumenta. Kada se raspakuje dokument, u njemu su sva ona slova koja je imao i pre postupka komprimovanja. Slično tome, i video materijal je moguće snimati bez gubitka podataka. U ovom slučaju se podaci samo sabijaju u manji format. U formate komprimovanja bez gubljenja podataka spadaju AVI, WAV i bitmapa (BMP datoteka).

5.6.3. Komprimovanje sa gubitkom

U komprimovanje kod kojih postoji gubitak podataka ubrajaju se formati WMV, WMA i MPEG. Prilikom svakog snimanja datoteka u nekom od ovih formata odbacuje se sve više podataka. Kod visokog stepena komprimovanja slika ili videa gubitak podataka postaje primetan, obično u vidu delimične deformacije oblika, tj. glatkih linija. Zato će autor, kad god neki video zapis snima u komprimovanom formatu, morati da balansira između kvaliteta i veličine.

5.6.4. MPEG-4 video

Sticajem okolnosti, neki elementi MPEG-4 standarda ušli su u upotrebu mnogo brže nego što su tvorci očekivali. Glavni razlog za to je gotovo neverovatna popularnost video-kodeka. Zamisao je bila da MPEG-4 omogući kvalitet slike približan DVD-u na daleko manjim propusnim opsezima. Teorijski, algoritam je trebalo da omogući DVD kvalitet u $3 \div 6$ puta manjem fajlu. Donja granica otprilike odgovara kapacitetu CD-a.

Prvu implementaciju napravio je *Microsoft*; ta verzija je bila dostupna u izvornom obliku ali i kao *Video for Windows* kodek. Tako je šira javnost prvi put i došla u kontakt sa MPEG-4 videom i uvidela njegove potencijale. *Microsoft* je želeo da primenu zadrži za sebe, pa je *Microsoft*-ov MPEG-4 kodek radio samo sa *Microsoft*-ovim zatvorenim formatom koji se zove ASF/VMW. Odgovor video entuzijasta bio je kreiranje kodeka – kako bi krekovani i originalni kodek mogli da rade u isto vreme, autori kreka su promenili ime

krekanog kodeka u DivX, po propalom sistemu za digitalne video diskove nalik DVD-u, ali u kome je diskovima isticao rok upotrebe nakon 48 sati.

DivX je ubrzo postao za filmove ono što je MP3 bio za muziku – filmski studiji i distributeri dobili su još jedan razlog za glavobolju, ali su autori kreka uvideli potencijale pa su formirali kompaniju koja se danas zove *Divx Networks*. Ova kompanija je razvila sopstvenu implementaciju MPEG-4 video-kodeka (Divx 4 i 5). Razvoj ovog kodeka jedno vreme je bio u formi *open source* projekta OpenDivx, iz čega je posle raznih peripetija stvoren projekat Xvid, čiji se kodek smatra jednom od najboljih i najkompletnijih implementacija MPEG-4 video-kodeka.

5.6.5. MP4 format

Divx, Xvid i ostali danas se najčešće koriste kao video-kodek u AVI formatu fajlova, ali je AVI zastareo format koji omogućava samo delić onoga što MPEG-4 nudi. Za punu podršku biće potrebno masovnije korišćenje MPEG-4 formata fajla. MPEG-4 je, slično formatima AVI ili MOV, po tipu "kontejnerski", što znači da je namenjen smeštanju raznih drugih informacija, u ovom slučaju video i audio materijala i drugih sadržaja koje MPEG-4 podržava. Posebno je zanimljivo što je za ovaj format usvojena ekstenzija MP4, čime je u startu dobijena besplatna reklama.

MPEG-4 format je nastao na osnovu programa *QuickTime* kompanije *Apple*, što ne treba da čudi. *QuickTime* tehnologija je u startu razvijana za profesionalne primene, pa je i format zapisa *QuickTime* omogućavao zanimljive stvari kao što je podrška za titlove ili spisak efekata koje pri reprodukciji treba izvesti nad slikom ili zvukom. *QuickTime* nije u široj upotrebi samo zato što je *Windows* implementacija katastrofalno loša, čime je *Apple* propustio da svojoj tehnologiji obezbedi veći broj korisnika.

MPEG-4 format daleko proširuje ove mogućnosti podrškom za različite tipove multimedijalnog sadržaja koji su deo MPEG-4 specifikacije, kao i jezikom za definisanje "scena", te kompozicije objekata, njihove međusobne interakcije i interakcije sa korisnikom. Uz sve ovo dolazi i obavezni mrežni *streaming*, kao i (za sada delimična) podrška za skaliranje u skladu sa raspoloživim propusnim opsegom.

5.6.6. MPEG-4 audio

Iz tonskog aspekta MPEG-4 standard ne donosi mnogo toga novog, ali se očekuje da će MPEG-4 audio postići tržišni uspeh. Glavni audio-kodek je već poznat iz MPEG-2 standarda kao AAC (*MPEG-2 Advanced Audio Coding*, originalno ime je bilo *MPEG-2 Non Backwards Compatible*). Glavne karakteristike su ostale iste – AAC omogućava do 48 audio kanala uz njihovo uparivanje po potrebi, 15 LFE kanala, ugrađivanje podataka, podršku za različite jezike i slično. Osim MPEG-4 AAC, MPEG-4 format može da primi i prethodne MPEG audio standarde (MP2, MP3, MPEG-2 AAC), i ima rezervisane tipove objekata za ove kodeke.

Mogućnost skladištenja i upotrebe sintetičkog sadržaja predstavlja glavnu razliku u odnosu na MPEG-1 i MPEG-2. Umesto realnog digitalizovanog zvuka i slike, objekti mogu biti i računarski generisani – umesto prave, snimljene i digitalizovane muzike, MPEG-4 može

da ima MIDI muziku koja će biti sintetizovana prilikom reprodukcije. Umesto pravog glasa, autor MPEG-4 sadržaja može navesti samo tekst koji treba da bude pročitao, te definisati (ili ostaviti korisniku da to uradi) uslove pod kojima će tekst biti pročitao – pol "spikera", boja glasa itd. Isto tako, vizuelni sadržaji mogu biti predstavljeni 2D ili 3D objektima.

MPEG-4 sadrži jezik koji služi za opis i upravljanje ovakvim zvučnim i vizuelnim objektima, koji je nazvan BIFS – *Binary Format For Scenes*. On omogućava manipulaciju objektima i dinamičke promene njihovih svojstava. Pandan BIFS-u su već pomenuti *Macromedia Flash* ili W3C jezik VRML. Razlika između njih je što je MPEG-4 otvoren standard, a velika prednost u odnosu na VRML je mogućnosti za *streaming* u realnom vremenu. MPEG-4 podržava i *Javu* kroz ekstenziju specifikacije nazvanu MPEG-J.

5.6.7. Neki tipovi digitalnih audio datoteka

Windows Audio Video (WAV) – Ove datoteke predstavljaju kompresiju bez gubitaka tako da su prilično velike. Pesme u ovom formatu obično su veće od 40 Mb, ali i izuzetno kvalitetne, pa se mogu snimati koliko god puta treba, bez ikakvog gubitka kvaliteta. Sve video aplikacije prihvataju ovaj format kao zvučni izvor.

Windows Media Audio (WMA) – Ovo je jedan od najboljih formata za komprimovanje digitalnog zvuka. WMA datoteke poseduju kvalitet MP3 formata, ali sa upola manjim datotekama i protokom podataka. Nažalost, dosta programa ne prihvata WMA format.

MP3 – Format MP3 ili MPEG Layer 3 je predak svih audio kompresija sa gubitkom. MP3 datoteke imaju sasvim prihvatljiv kvalitet zvuka, a deset puta su manje od WAV datoteka, što ih čini izuzetno popularnim. Danas gotovo svaki paket za uređivanje videa može da prihvati ovaj format zvučnog zapisa.

5.6.8. Neki tipovi digitalnih video datoteka

Audio Video Interleaved (AVI) – Ovaj format je jedan od najstarijih i najrasprostranjenijih video formata namenjenih PC platformama. AVI datoteke u osnovi predstavljaju "kontejnere" za video. Ako se pri kompresiji ne koristi neki moćan kodek (kompresor/dekompresor), ove datoteke su veoma velike.

Windows Media Video (WMV) – U ovom formatu se podaci odbacuju primenom posebnih psihovizuelnih kodeka nakon čega se dobija veoma kvalitetan video zapis koji zauzima iznenađujuće malo prostora. AVI datoteka komprimovana u WMV formatu može da zauzima i do 20 puta manje prostora, a da razlike u kvalitetu ne budu mnogo primetne.

Moving Picture Experts Group (MPEG) – Format MPEG je dugo prisutan i predstavlja osnovu mnogih drugih postojećih video formata. Ovo je prvi format namenjen PC računarima koji je omogućavao komprimovanje ogromnih AVI datoteka u mnogo manje datoteke uz zadržavanje pristojnog kvaliteta. Ove datoteke mogu da se reprodukuju u *Windows Media Player*-u, a u većini programa za uređivanje videa postoji mogućnost izbacivanja neke varijante MPEG formata.

MPEG2 – Ovaj format je novija verzija MPEG-a koja omogućava dobijanje kvalitetnijih video zapisa. MPEG2 datoteke se snimaju u kompresiji sa gubitkom podataka.

MPEG4 – Ovaj format je vremenom prerastao u zamenu originalnog MPEG video formata. Većina aktuelnih video formata se bazira na standardu MPEG-4, i on predstavlja jezgro najvećeg dela striming videa koji danas postoji.

RealVideo – Nekada je bio neprikosnoveni gospodar striminga. Mana mu je to što da bi mogao da se reprodukuje *RealVideo* materijal neophodno je da bude instaliran *RealOne* plejer, što je smetnja kada korisnik hoće da razmenjuje svoje snimke sa drugima.

QuickTime – Ovaj format posebno je popularan među filmskim kompanijama koje svoje trejlere objavljuju na webu. Video kvalitet ovog formata je izuzetan, a njegov striming veoma uspešan.

DivX – Ovaj format je relativno nov, i reč je o kodeku za AVI format baziranom na standardu MPEG-4 pomoću koga se dobijaju izuzetno komprimovane video datoteke vrhunskog kvaliteta. Posebno je popularan za razmenjivanje filmova. Ipak i ovaj format ima svoje nedostatke. Ako se korisniku pošalje elektronskom poštom AVI datoteka u DivX formatu, on će u svom *Windows Media Player*-u moći da čuje zvuk, ali neće moći da vidi sliku.

5.7. Snimanje i uređivanje videa

Da bi se multimedijalnom projektu dodao video preko celog ekrana i u punom pokretu, autor će morati da investira u specijalizovan hardver i softver ili da plati usluge profesionalnog studija za video produkciju. U mnogim slučajevima profesionalni studio će, takođe imati alate za editovanje i postprodukcione sposobnosti koje korisnik ne može da ima na *Macintosh*-u ili PC-u. Treba napomenuti da se može na zadovoljavajući način obaviti mnogo posla sa video kamerama potrošačkog nivoa, ako se razumeju ograničenja tehnologije.

5.7.1. Planiranje ili scenario

Planiranje ili **scenario** je faktor koji ne može biti ignorisan bez troškova zbog gubitka vremena, puno nepotrebnih pogoršanja i gubitka novca koji bi se bolje upotrebio na nekom drugom mestu. Uspešna video produkcija, bilo koje vrste, zaslužuje vreme koje je potrebno da bi se napravio plan da se sprovede. Možda će biti potrebno malo više vremena prvi put, ali je očigledno da je to korisna stvar na duge staze. Scenario predstavlja strip koji se čita svakodnevno. Za svaki dan postoje tri ili četiri panela koji prikazuju napredovanje priče. Treba naći vremena da se napravi struktura produkcije, a zatim sledi grupa crteža koji pokazuju kamere i scenu, uglove snimanja, osvetljenje, akcije, specijalne efekte i koliko objekata će se kretati kroz scenu, od početka do kraja. Scenario mogu svi učesnici dobiti brzo na jednoj jedinoj stranici.

5.7.2. Platforma za snimanje

Nikada se ne sme potcenjivati vrednost postojeane platforme za snimanje. Klasični simbol amaterskih kućnih snimaka je kamera koja se trese u ruci. Postavljanjem kamere na stabilnu osnovu, kao što je presavijeni džemper na haubi kola, može poboljšati snimak. Uz

malo pažnje i pažljivog prilagođavanja šrafova za učvršćivanje, metalni tronožac će učiniti čuda. Ako korisnik mora da snima iz ruke, trebalo bi da pokuša da koristi kameru sa stabilizacijom elektronske slike za nepokretne snimke ili da pokreće kameru i/ili ono što snima da bi zamaskirao nedostatak postojanosti.

5.7.3. Osvetljenje

Verovatno najveća razlika između profesionalnih i potrošačkih kamkordera je sposobnost snimanja po slabom svetlu. Upotrebom jednostavnog priručnog osvetljavanja ili, čak samo obezbeđivanjem da dnevna svetlost prodre u sobu korisnik može znatno da poboljša sliku. Kao u fotografiji, dobre tehnike osvetljavanja razdvajaju amatere od profesionalaca u snimanju videa.

5.7.4. Hroma ključevi (*Chroma Keys*)

Hroma ključevi ili *Chroma Keys* se koriste za brzo uklanjanje ili maskiranje dela slike. Mogu da se koriste za razne svrhe – uklanjanje pozadine sa slika, maskiranje slika, generisanje ikonica, isecanje/odsecanje delova slika, stvaranje nepravilnog oblika slike itd.



Slika 5.15. *Snimanje scena ispred plave pozadine*

Filmska industrija je rešila ovaj problem razvojem **plavog ekrana**, a kasnije i zelenog (slika 5.15). Koristeći ovaj metod, glumci (u prvom planu) su snimani ispred plave pozadine. Plava pozadina je kasnije zamenjena elektronskom ili drugom slikom ili videom. Plavi ekran je popularan metod za pravljenje multimedijalnih naslova, zato što se ne zahteva skupa scenografija. Neverovatne pozadine se mogu napraviti 3D modelovanjem i grafičkim softverom i na njih se mogu pogodno razmestiti jedan ili više glumaca, prevoznih sredstava ili drugih objekata. U današnje vreme mnoge aplikacije imaju tu mogućnost.

Kada se snima plavi ekran, treba proveriti da li je svetlo na zaslonu apsolutno ravnomerno; fluktuacije u jačini će učiniti da taj "ključ" izgleda kao brdovit ili slomljen. Snimanje po dnevnom svetlu (omogućavanje da sunce osvetli zaslon) ublažiće ovaj problem. Ako glumci stoje previše blizu zaslona, obojeno svetlo odbijeno od zaslona će se razliti po njima, pa će delovi njihovih tela "nestati". I dok podešavanja u većini programa omogućavaju da se ovakve greške isprave, treba imati na umu da su te mogućnosti ograničene.

Glava 6

Virtuelna i proširena realnost

Stalni razvoj računara i informatičkih tehnologija omogućuje realizaciju i primenu novih metoda i sistema kakve ranije nisu bile moguće. Jedan primer takvog razvoja je i tehnologija **virtualne stvarnosti** ili **virtualne realnosti** ili **virtualnog okruženja**. Tehnikama virtualne stvarnosti moguće je ostvariti realistične simulacije koje su korisne u mnogim područjima ljudske delatnosti. Simulacije su bile poznate i ranije, ali tehnike virtualne stvarnosti mogu ostvariti utisak "uplivavanja" čoveka u nepostojeći ili prividni svet.

Takav utisak prisutnosti u prividnom svetu moguće je ostvariti pomoću naprednih računara i uređaja za komunikaciju između čoveka i računara. Tehnike virtualne stvarnosti koriste i savremene računarske mreže da bi ostvarile komunikaciju između čoveka i od njega udaljene okoline sa svrhom ostvarivanja delovanja na daljinu.

Početkom devedesetih godina dvadesetog veka javno shvatanje virtualne stvarnosti prilično je iskrivljeno. Naime, zahvaljujući njenom odjeku u štampi, filmovima i televiziji, od virtualne stvarnosti su se doslovno očekivala čuda. Ali, iako su to neki predviđali, virtualna stvarnost nije ušla u široku primenu. Tako je kod dela javnosti, ali i kod nekih stručnjaka došlo do velike promene u stavovima o virtualnoj stvarnosti, pa su tu tehnologiju proglasili beskorisnom.

6.1. Definicija i princip virtualne realnosti

Da bi pojam i princip virtualne stvarnosti bio što je moguće jasniji, na samom početku potrebno je pojasniti pojam **percepcije**. Percepcija je proces u kojem se prikupljaju i interpretiraju informacije o svetu oko korisnika. U procesu percepcije učestvuju čula i mozak.

Postoje dve vrste čula – spoljašnja i unutrašnja. Spoljašnja detektuju pojave izvan organizma, a unutrašnja detektuju pojave unutar organizma (glad, umor, bol, žeđ...). Spoljašnja čula mogu još da se podele na **daljinska** (toplota, vid, sluh) i **kontaktna** (miris, dodir, ukus). Kada je u pitanju virtualna stvarnost, za sada su dobro razvijeni samo sistemi koji utiču na daljinska čula, mada se postupno razvijaju i sistemi koji utiču na kontaktna čula.

Čula prenose informacije iz okoline, dok mozak interpretira primljene informacije. Osim čula, na percepciju utiču i iskustva, znanje, emocije i motivacija. Da bi se prevario sistem

percepcije, osnovna ideja je da realne nadražaje koje primaju čovekova čula treba zameniti veštački generisanim nadražajima. Na taj način može realna okolina da se zameni prividnom okolinom. Kao posledica se javlja to da se u sistemu percepcije stvara utisak o prisutnosti osobe u prividnoj, nepostojećoj okolini.

Virtualna realnost (*Virtual Reality*, VR) je računarski kreirano senzorsko iskustvo koje omogućava korisniku da poveruje u prividnu stvarnost. Korisnik je tada ili potpuno okružen tim virtualnim svetom ili delomično uključen slušajući i gledajući aplikacije virtualne stvarnosti. Virtualna stvarnost je skup tehnologija koje korisnika "uranjaju" u virtualno okruženje. Pritom, u idealnom slučaju, korisnikova čula detektuju samo virtualne nadražaje proizvedene računarom, a uz to je ostvaren direktan unos korisnikovih pokreta u računar. Za postizanje ovih efekata koriste se različiti uređaji, o čemu će kasnije biti reči.

Virtualna okruženja (VO) se zasnivaju na predmetima koji su definisani u memoriji računara na takav način da računar kasnije može da prilaže te predmete na ekranu uz mogućnost interakcije. Kombinovanjem elemenata nestvarne (izmišljene) okoline i stvarne okoline (koja može biti i udaljena), kod korisnika se stvara utisak o prisutnosti u nekoj virtualnoj okolini. To je prikazano u sledećoj slici.



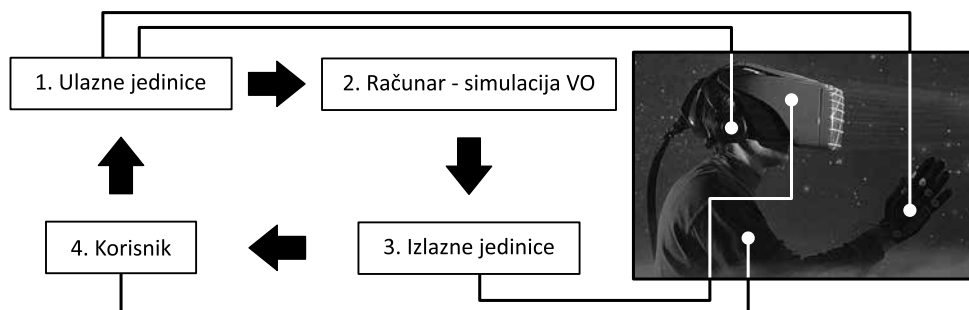
Slika 6.1. *Virtualna okruženja*

Slika 4.2 prikazuje osnovni princip virtualne stvarnosti. Korisnik se nalazi u zatvorenoj petlji i povezan je sa računarom pomoću ulaznih i izlaznih jedinica. Ulazne jedinice (1) prate pokrete korisnika i prosleđuju ih računaru (2), koji na osnovu tih i drugih podataka vrši simulaciju virtualnog okruženja (VO). Pomoću izlaznih jedinica (3) računar prikazuje virtualno okruženje korisniku, i to što je moguće direktnije.

U idealnom slučaju, korisnikova čula bi trebala da detektuju samo veštački generisane nadražaje (od računara), i time bi stvarni svet bio potpuno isključen. Tako u primeru na slici korisnik (4) vidi samo sliku stvorenu na računaru. Time je petlja zatvorena, a korisnik direktno vidi, čuje (i eventualno oseća, miriše, proba...) virtualno okruženje sa neposrednim rezultatima vlastitih pokreta.

Trenutni praktični dometi ove ideje ograničeni su nemogućnošću da se realna čula zamene veštačkim čulima, a posebno je teško preneti kontaktna čula (dodir, miris, ukus).

U praksi je granica između virtualne stvarnosti i "običnog" virtualnog okruženja vrlo relativna i slabo definisana, i zavisi od toga šta se uzima kao glavno svojstvo VR.



Slika 6.2. Princip virtualne realnosti

Ako se kao glavno svojstvo virtualne stvarnosti uzima to da korisnika ubacuje u virtualno okruženje, tj. da korisnik ima osećaj da se nalazi negde drugo, onda se ta granica može postaviti bilo gde. U tom slučaju se VR može smatrati i obična knjiga ili film, jer i u njih korisnik može toliko da se udubi da mu se čini da se nalazi negde drugde.

Ipak, najčešće se virtualnim okruženjem smatra interaktivna 3D grafika na personalnom računaru sa mišem, a virtualnom stvarnošću se smatraju sistemi koji korisnika još više "ubacuju" u virtualno okruženje korišćenjem raznih dodatnih uređaja.

6.2. Uređaji za virtualnu realnost

Uređaji za virtualnu stvarnost dele se, najpre, na **ulazne** i **izlazne**, a zatim na vrste i podvrste unutar svake kategorije. U **ulazne uređaje** spadaju:

- **Senzori pozicije/orijentacije** – elektromagnetski, akustički, optički, mehanički, inercijski.
- **Senzori sile/momenta sile** – *Spaceball* i slično.
- **Senzori položaja tela/ruke** – senzorska rukavica *DataGlove*, senzorsko odelo *BodySuit*.
- **Senzori pokreta** – pokretna traka, bicikl ergometar, veslački ergometar itd.
- **Ostalo** – upravljanje putem disanja, praćenje lica i očiju, prepoznavanje govora.

Elektromagnetski senzori koriste izvor elektromagnetskog polja, te elektromagnetske senzore, a položaj izvora je fiksna i poznat. Senzori koji se nalaze na glavi i rukama korisnika primaju elektromagnetski signal te ga prosleđuju središnjoj jedinici koja na osnovu primljenog signala računa položaj i orijentaciju senzora unutar elektromagnetskog polja, dakle položaj i orijentaciju u odnosu na izvor. Izvor i senzori povezani su sa središnjom jedinicom pomoću kablova.

Akustički senzori rade po sličnom principu kao i elektromagnetski, samo što umesto elektromagnetskih talasa svaki odašiljač šalje zvuk visoke frekvencije koji primaju prijemnici – specijalni mikrofoni. Prednosti ovog sistema su prihvatljiva cena i dobar domet, ali

i ovaj sistem ima svoje nedostatke. Između odašiljača i prijarnika ne sme biti nikakvih prepreka, sistem ne može da podržava veći broj istovremenih senzora, preciznost je lošija od magnetskih senzora, a i dimenzije prijarnika mogu predstavljati poteškoću za pojedine primene.

Optički senzori (*motion tracking, optical motion capture*) su deo sistema koji pomoću većeg broja kamera prate oznake (markere), te kombinovanjem položaja markera u vidnom polju svake kamere izračunavaju položaj markera u prostoru (3D). U radu ovih sistema koriste se različiti principi praćenja, najčešće uz pomoć markera od sjajnog materijala. Kamere, osetljive na infracrveno svetlo prate markere, odnosno pokrete tela u prostoru. Kamere moraju biti kalibrisane, tj. njihovi međusobni položaji i orijentacije poznate. Kombinovanjem zapisa 2D položaja markera (iz svih kamera), uz informaciju o položajima i karakteristikama kamera, precizno se utvrđuje 3D položaj markera. Glavne prednosti optičkog praćenja su izuzetno velika tačnost, velika količina uzorkovanja podataka, te mogućnost istovremenog korišćenja velikog broja senzora (čak i do stotinu). Ovakav način najčešće se koristi u biomehaničkim laboratorijama gde se analiziraju kinematičke i dinamičke komponente gibanja.

Mehanički senzori konceptualno su najjednostavniji. Postoji mehanička konstrukcija sa određenim brojem zglobova u kojima se meri ugao rotacije zgloba. Iz ovih uglova, poznatih dužina segmenata među zglobovima, jednostavnim rešavanjem direktne kinematike dobija se položaj zadnjeg segmenta u odnosu na prvi, fiksni segment konstrukcije. Upotrebom tzv. *exoskeletona* (mehaničke konstrukcije montirane na telo, slika 6.3) mehaničko praćenje može se koristiti za praćenje položaja zglobova u telu, a time i položaj čitavog tela. Prednosti ovakvih uređaja su visoka preciznost i velika frekvencija uzorkovanja, ali nedostatak je nepraktična izrada uzrokovana veličinom.



Slika 6.3. *Exoskeleton – mehanička konstrukcija montirana na telo*

Inercijski senzori mere ubrzanje i ugaono ubrzanje segmenata određujući na taj način njihov položaj, kao i međusobni položaj i orijentaciju. Zbog tačnosti potrebno je precizno odrediti početni položaj.

Senzori sile (momenta sile) su uređaji koji mere silu, odnosno moment sile, a mogu se nalaziti u uređajima poput *spaceball*-a (slika 6.4 sredina) ili su deo kompleksnijeg sistema za simulaciju sile ili dodira (slika 6.4 levo). Koriste se za intuitivniju manipulaciju 3D predmetima i prirodnije kretanje kroz virtualnu okolinu.

Senzori položaja tela sastoje se od velikog broja senzora (obično je to broj od 10 do 100) za registraciju položaja i orijentacije koji su integrisani u odelo. Informacije koje emituju senzori gotovo u potpunosti i u realnom vremenu opisuju kretanje segmenata u prostoru (slika 6.4 desno).



Slika 6.4. Senzori sile i položaja

Senzori položaja ruke (data glove) su ulazni uređaji u obliku rukavice koji putem mnogobrojnih senzora registruju položaj šake i poziciju prstiju mereći ugao pomaka (slika 6.5). Ovaj tip uređaja moguće je kombinovati sa simulatorima sile ili dodira čime rukavica postaje i haptički izlazni uređaj.



Slika 6.5. Senzori položaja ruke (Data Glove)

Uloga **senzora pokreta** sastoji se u obeležavanju pozicije objekata u realnom svetu, te prosledjivanje informacije do računara. Obradenu informaciju o poziciji tela računar sprema, grafički prikazuje ili dodeljuje objektu sa kojim je korisnik u interakciji.

Zanimljivo je kako konstrukcija senzora pokreta omogućuje njihovu upotrebu i u kineziološkoj praksi. Trenažeri poput bicikl ergometara, veslačkih ergometara, pokretnih traka i drugih različitih platformi mogu biti upotpunjeni takvim sensorima u svrhu detaljnijeg uvida u pokretnu strukturu.

Zajednička karakteristika navedenih sistema je mogućnost prikupljanja informacija (*motion capture*) o položaju referentnih tačaka u prostoru u realnom vremenu. Dobijeni podaci mogu poslužiti za vizualizaciju događaja u tri dimenzije i njihovu analizu. Različite aplikacije za modelovanje i animiranje 3D objekata u mogućnosti su da koriste informacije prikupljene sa senzora pri izradi animacija.

U **izlazne uređaje** spadaju:

- **Uređaji za 3D prikaz** – stereo naočare, *Head Mounted Display*, stereo ekrani (sa izmenjivanjem slike ili sa dvostrukom slikom), projekcioni sistemi (stereo projekcije na jedno platno, CAVE, širokougaone projekcije, virtualni radni sto).
- **Uređaji za generisanje 3D zvuka**
- **Uređaji za generisanje čula dodira i sile** – taktilni izlazni uređaji, uređaji za povratne sile, pomične platforme.
- **Ostalo** – miris, vetar, toplota.

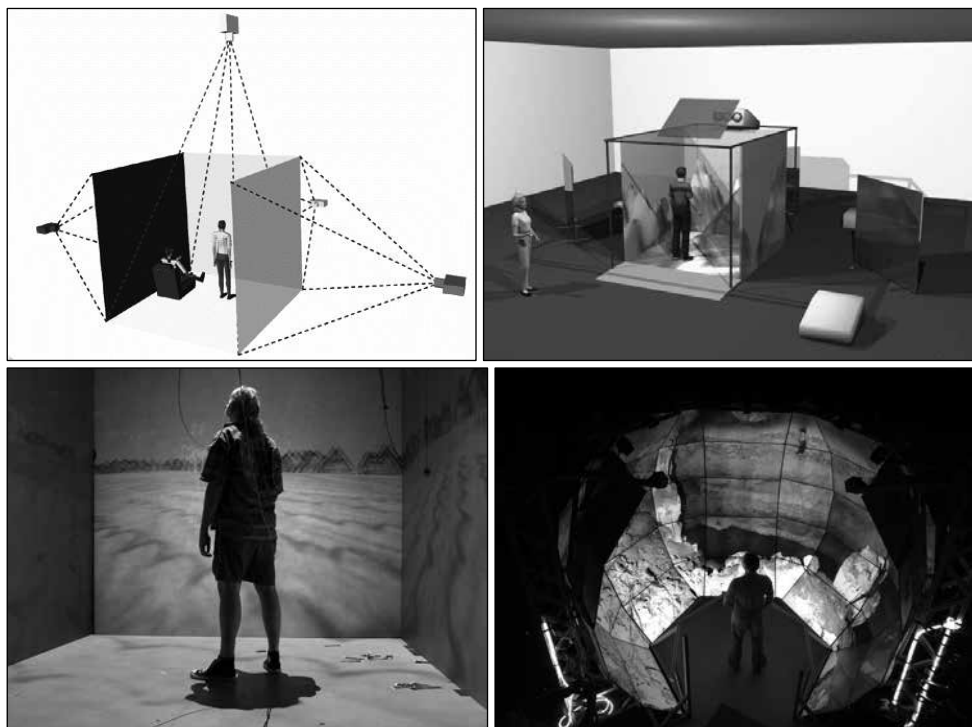
Da se postigne stereoskopski efekat potrebno je da se projektuju dve slike, na svako oko po jednu. *Head Mounted Display* – HMD ima za svako oko poseban zaslon (slika 6.6). Zbog malih dimenzija uređaja, zaslone su preblizu da bi ih neposredno posmatrali, pa se između oka i zaslona postavlja odgovarajući optički sistem koji omogućuje gledanje zaslona.



Slika 6.6. *HMD – Head Mounted Display*

Najvažnije karakteristike HMD-a, uz veličinu, težinu i udobnost, ogledaju se u veličini vidnog ugla i rezoluciji zaslona. Mogu se pronaći u raznim oblicima, od kaciga do naočara. Danas se teži minimalističkom pristupu i što većoj praktičnoj primenljivosti. Cilj je stvoriti dovoljno mali uređaj koji ne ometa slobodno izvođenje kretnji. Takvi uređaji mogu biti opremljeni, uz slušalice, i senzorima položaja i orijentacije.

Najnapredniji **projekcioni sistem** je CAVE (*Cave Automatic Virtual Environment*) sistem. Sastoji se od prostora omeđenog projekcionim platnima (koji kreiraju sobu u kojoj se nalazi korisnik) na koje se projektuje računarom generisana stereo slika (slika 6.7). Korisnik nosi naočare putem kojih se ostvaruje trodimenzionalni doživljaj pružajući pritom zadovoljavajući periferni vid. Doživljaj je izuzetno realističan da su neprimetni spojevi između postavljenih projekcionih platana.



Slika 6.7. Šematski i stvarni prikaz CAVE sistema

Zvučna simulacija uključuje reprodukciju ili generisanje zvuka u virtualnom okruženju. Uključivanjem trodimenzionalnosti zvuka dobija se dojam precizne lokacije izvora zvuka u prostoru. Efekat se postiže razlikom primljene jačine zvuka u levom i desnom uvu, na temelju odbijanja zvučnih talasa na ušnoj školjci i njenoj okolini.

Haptički uređaji omogućuju simulaciju dodira i/ili sile, čime se kod korisnika stvara osećaj kontakta (dodira) sa virtualnim predmetom.

Simulacija dodira (*tactile feedback*) obično se temelji na vibrirajućim ili termičkim elementima koje korisnik nosi na prstima i koji se aktiviraju kada korisnik "dodirne virtualni predmet", za što je potrebno precizno praćenje položaja korisnika, odnosno ruku.

Simulacija sile (*force feedback*) uključuje praćenje položaja uz uključivanje aktivnih elemenata (motora, elektromagneta, servo motora) koji vrše silu na korisnikovu ruku, delove tela ili na alat kojim barata.

Pomične platforme su haptički sistemi kojima se simulira pozicija korisnika pomicanjem čitave platforme na kojoj stoji ili sedi. Usklađenost pozicije sa vizualnim informacijama pojačava osećaj učešća u simulaciji.



Slika 6.8. *Razni simulatori*

Ovaj tip haptičkog sistema najčešće se koristi u kompleksnim simulatorima, simulatorima vožnje automobilom, avionom i slično (slika 6.8).

6.3. Primene virtualne realnosti

Virtualna stvarnost se najviše primenjuje u sledećim područjima – medicina, vojna primena, obrazovanje, zabava, dizajn i razvoj, marketing.

Medicina je jedno od najjačih područja primjene virtualne stvarnosti. Koristi se u području hirurgije, kako za obuku (učenje na virtualnim ljudskim modelima), tako i za planiranje hirurških zahvata. Iz medicinskih snimaka mogu se dobiti 3D prikazi (slike 6.9 i 6.10), što je sve češći slučaj na modernim uređajima u medicini. U psihijatriji se virtualna stvarnost koristi za lečenje raznih psihičkih poremećaja, počevši od straha od letenja do posttraumatskog stresnog poremećaja, gde se postižu vrlo dobri rezultati.



Slika 6.9. *Primeri kreirane vizuelizacije pomoću računara u medicini*



Slika 6.10. *Primeri vizuelizacije pomoću računara u medicini*

Jedan od najjačih ulagača u virtualnu stvarnost su vojne organizacije i mnoge VR tehnologije su ugrađene u simulatore raznih vojnih uređaja (slika 6.11).



Slika 6.11. *Simulator helikoptera, aviona, tenka i broda*

Simulacije raznoraznih vozila su među najčešćim primenama virtualne stvarnosti. Brojni stručnjaci se usavršavaju na različitim simulatorima, a posebno je važno da mogu da se uvežbavaju situacije koje se u stvarnosti retko dešavaju, na primer, spašavanje talaca.

Virtualna stvarnost je idealna za industriju zabave, zbog mogućnosti kreiranja iluzija. U zabavnim parkovima kao što je *Disneyland* postoji veliki broj atrakcija koje koriste tehnike virtualne stvarnosti. U salonima igara pojavljuje se sve više igara koje koriste ovu tehniku, i

pitanje je vremena kada će ova tehnologija postati dostupna i kućnim igračima računarskih igara.

Virtualna stvarnost može da se koristi i za prezentacije budućih projekata u arhitekturi, stvaranje prototipa budućih proizvoda, kao uspešan alat za promociju i marketing na izložbama, sajmovima i tako dalje, jer je sama pojava 3D projekcije još uvek dovoljno zanimljiva da privuče znatiželjnike.

Usprkos brojnim područjima primene, postoje i ograničenja. Iako je poslednjih godina došlo do znatnog napretka, oprema je i dalje nepraktična, velika, skupa i složena. Određene vrste virtualne stvarnosti mogu kod korisnika izazvati mučninu, a čak i ako je ne izazovu, previše su neudobne za dugotrajnu upotrebu. Mučnina koja se javlja kao posledica razlike između vizualnog nadražaja i signala centru za ravnotežu naziva se **simulatorska mučnina**.

6.4. Proširena realnost

Proširena realnost (*Augmented Reality – AR*) dodaje elemente virtualnog okruženja u stvarni svet tako da izgledaju kao deo stvarnog sveta. Time se korisnikovo viđenje sveta proširuje dodatnim informacijama koje su direktno ugrađene u stvarni svet. U nekim primenama nije potrebno potpuno zameniti realnost sa virtualnim svetom, nekad je potrebno samo nadopuniti ili poboljšati realnost nekim virtualnim delovima (slika 6.12).

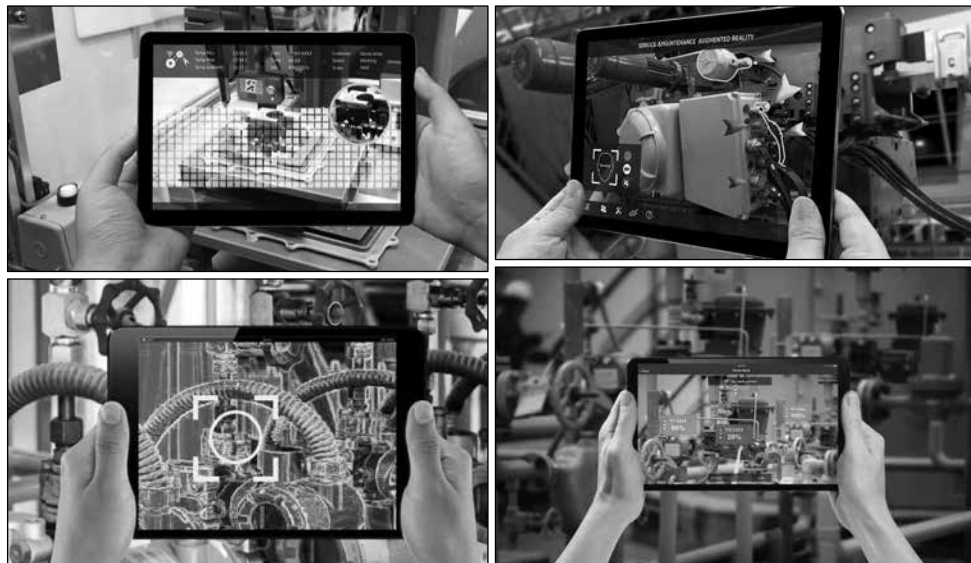


Slika 6.12. *Stvarnost i proširena realnost*

Proširena realnost je relativno novo područje. Iako se osnovna ideja javila još dvadesetih godina dvadesetog veka, tek devedesetih godina počinje intenzivno da se radi na njenom razvoju, i to je razlog što, uglavnom, još nije zrela za široku upotrebu. Ona pruža direktan pristup informacijama tako da su one prikazane u samom vidokrugu korisnika i isprepletene sa stvarnim svetom. Time se omogućuje brži, kvalitetniji i jednostavniji pristup informacijama. Moguća područja primene su – medicina, proizvodnja i održavanje, arhitektura, robotika, vojne primene, komercijalne primene, zabava.

Kada su u pitanju medicinske primene, medicinske slike se preklapaju sa pacijentom, čime se dobija vrsta virtualnog rendgena u realnom vremenu. Dobijeni efekat se ogleda u tome da lekar vidi organe pacijenta kao da je telo prozirno. Za sada nisu u širokoj primeni. Kod

proizvodnje i održavanja vizuelne instrukcije se prikazuju direktno na opremi/mašinama, pa operater, umesto da gleda dokumentaciju, ima sve potrebne informacije u pravo vreme na pravom mestu.



Slika 6.13. *Primeri proširene realnosti u proizvodnim pogonima*

Proširena realnost može da se koristi u dizajnu enterijera, vizualizaciji struktura ili instalacija. Na primer, može da se razmešta virtualni nameštaj po stvarnoj prostoriji, čime može da se stekne utisak o prostornim odnosima i o tome kako će prostorija zaista izgledati sa nekim nameštajem.



Slika 6.14. *Primeri proširene realnosti*

Pomoću proširene realnosti vojni piloti mogu dobijati dodatne informacije kao što su navođenje, prikaz ciljeva ili navođenje projektila. Prikaz je ugrađen u kacigu ili na kabinu.

Cilj proširene realnosti je da korisniku pruži jednostavan i intuitivan pristup podacima. Da bi se to ostvarilo, potrebne su prilično složene tehnologije. U optičaju su tri osnovna problema – mešanje slike, poravnanje i prikupljanje podataka. Mešanjem slike omogućuje se

istovremeni prikaz stvarne i virtualne slike. Poravnanje osigurava da se virtualni predmeti tačno poklapaju sa stvarnim.



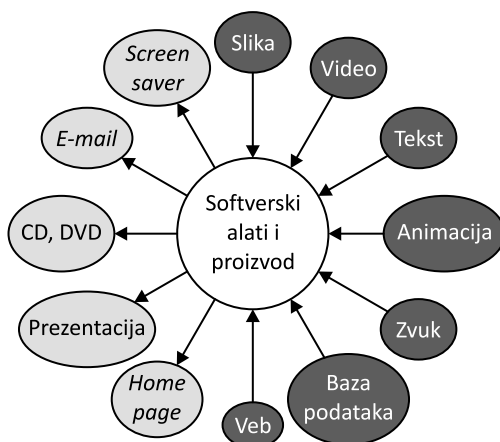
Slika 6.15. *Još primera proširene realnosti*

Proširena realnost ima veliki potencijal, ali današnji sistemi proširene realnosti (stvarnosti) su još uvek dosta nezgrapni, neprecizni i troše previše energije, pa je ove probleme potrebno rešiti.

Glava 7

Softverski alati za multimedijalni projekat

Multimedijalni softver "govori" hardveru šta treba da uradi. Na primer, da prikaže odgovarajuću boju, da pokrene određenu animaciju, da postavljeni baner "šeta" po monitoru sa jedne na drugu stranu, da pokrene odgovarajuću ili željenu muzičku numeru, da utiša zvuk i slično.



Slika 7.1. *Softverski alati – ulaz i izlaz*

Osnovni alat postavljen za izgradnju multimedijalnih projekata sadrži jedan ili više programa koji sadrže unapred programirane elemente za razvoj interaktivnih multimedijalnih softverskih elemenata (*authoring tools* ili *authoring systems*) i razne aplikacije za uređivanje teksta, slika, zvuka, animacija i videa. Prilikom razvoja obrazovnog softvera, jedan *authoring* sistem predstavlja program koji omogućava korisniku, koji nije programer, laku izradu softvera sa programskim funkcijama. Programske funkcije su **ugrađene** u sistem, ali su **skriven** iza tastera i drugih alata, tako da autor ne mora da zna kako se programira. Na ovaj način, authoring sistemi obezbeđuju grafiku, interakciju i druge alate koji su potrebni za kreiranje obrazovnog softvera. Takođe je korisno imati i koristiti i nekoliko dodatnih aplikacija za snimanje slika sa ekrana, prevođenje formata i distribuiranje fajlova između računara članova tima za razvoj multimedijalnog projekta.

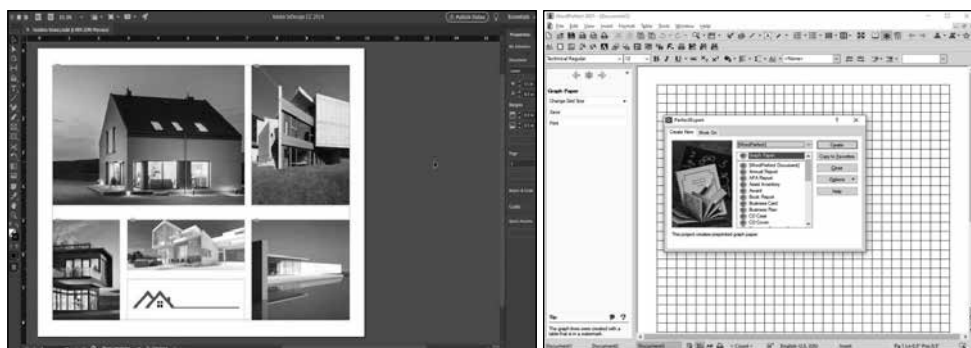
Softveri koji se upotrebljavaju za izradu multimedijalnog sadržaja i veštine autora da ih koriste određuju kakva će buduća multimedijalna aplikacija biti, šta će moći da radi i koliko će biti pristupačna ostalim korisnicima. Uspešan izbor svih potrebnih elemenata i alatki omogućava autoru ili autorima da naprave dobar multimedijalni sadržaj, koji će biti razumljiv i korisnicima sa manjim predznanjem iz ovih oblasti i koji će biti lak za upotrebu. Ako ne napravi uspešan izbor potrebnih alatki, autor će imati silnih problema i može da se desi da multimedijalni sadržaj ne završi i može da se desi da će biti prinuđen da obustavi započeti projekat.

Trebalo bi napomenuti da autor ne mora da bude programer ili naučnik da bi multimedija "radila" za njega, ali autor mora biti upoznat sa odgovarajućim uslovima i načinima kreiranja ili izgradnje multimedijalnog sadržaja. I najjednostavniji multimedijalni alati zahtevaju određen nivo znanja od strane autora, kako bi mogli da rade i kako bi omogućili da se postigne ono što autor želi. Biće stvari koje autor želi da uradi, ali neće znati kako ti alati rad. Očigledno je da autor treba da odvoji određeno vreme da nauči osnovne stvari po pitanju računara i multimedije, kako bi se lakše snalazio u daljem poslu.

Alati koji se koriste za pravljenje i uređivanje multimedijalnih elemenata, i na *Windows* i *Macintosh* platformama, uključuju alate za obradu teksta, alate za obradu slika i njihovo uređivanje, alate za crtanje i ilustrovanje, alate za projektovanje (CAD i 3D), alate za optičko prepoznavanje ili raspoznavanje znakova, alate za snimanje i uređivanje zvuka i videa, alate za kreiranje i uređivanje animacija i digitalnih filoma, kao i razne korisne dodatke ili pribore.

7.1. Obrada teksta

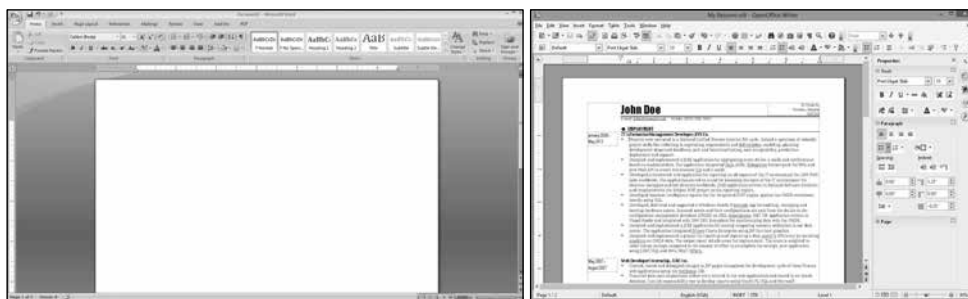
Program za obradu teksta (*word processor*) je obično prvi softverski alat koji korisnici računara treba da nauče. Od pisama, faktura, pisanja scenarija do projektovanja i kreiranja sadržaja, program za obradu teksta može da bude najčešće korišćen alat, kada se krene u projektovanje i izgradnju multimedijalnog projekta. Što je autor ili korisnik bolji u kucanju, to će lakše i efikasnije popunjavati odgovarajuća elementa i polja neophodna za multimedijalni projekat. Definitivno je najteži deo posla ukucati tekst i to po mogućstvu bez formatiranja, jer će se tekst podešavati kasnije u radu.



Slika 7.2. Adobe InDesign i Corel WordPerfect

Uobičajeno je da timovi unutar jedne kancelarije ili radne grupe izaberu zajednički program za obradu teksta, kako bi razmenjivali dokumenta u standardnom formatu. Najčešće se program za obradu teksta isporučuje u poslovnom paketu, u kojem se pored, pomenutog programa, nalaze i program za tabelarni proračun, baza podataka, veb pretraživač, program za elektronsku poštu i program za prezentovanje. Klasičan primer takvog poslovnog paketa je *Microsoft Office*.

Programi za obradu teksta, kao što su *Word* kompanije *Microsoft*, *WordPerfect* kompanije *Corel* ili *InDesign* kompanije *Adobe*, su ozbiljne aplikacije koje, pored ostalog, uključuju proveru pravopisa, formatiranje tabela, rečnike i unapred definisane šablone za mnoge elemente unutar lokalnih dokumenata i drugih zajedničkih dokumenata.



Slika 7.3. *Microsoft Word i OpenOffice Writer*

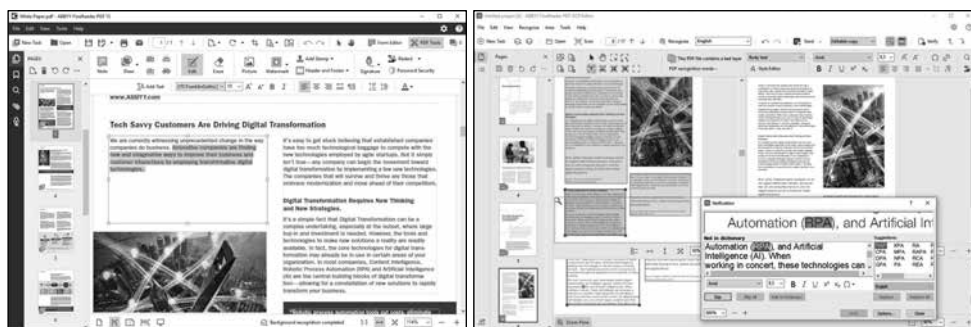
Mnogi su počeli da koriste *OpenOffice* za obradu teksta, tabelarni proračun, prezentacije, grafiku, baze podataka i još mnogo toga. Može se preuzeti i koristiti potpuno besplatno za bilo koju svrhu i dostupan je na više jezika. Mogu se čitati i dopisivati fajlovi iz drugih, skupljih, poslovnih paketa. U mnogim programima za obradu teksta mogu da se ugrade multimedijalni elementi kao što su zvuci, slike i video. Treba napomenuti da današnja deca u školama uče kucanje, pa čak i kucanje bez gledanja na tastaturu, tako da se vremenom smanjuje populacija stanovništva koja je u stanju da kuca samo sa dva prsta.

7.2. Optičko prepoznavanje znakova

Često se dolazi u situaciju da treba štampani materijal i drugi tekst uključiti u projekat, ali ne postoji elektronski tekstualni fajl. Sa **softverom za optičko prepoznavanje znakova** (*Optical Character Recognition, OCR*), kombinacija ravnanskog skenera i računara može da uštedi mnogo vremena, jer tekst ne mora da se prekucava i, na ovaj način, će se posao obaviti brže i preciznije. Softver OCR pretvara rasterske ili bitmapirane karaktere u elektronski prepoznatljiv ASCII tekst. Skener se obično koristi za kreiranje potrebnih bitmapa.

Softver OCR pretvara bitmapu u odgovarajući broj komada u zavisnosti da li sadrži tekst ili grafiku, uvidom u teksturu i gustinu bitmapiranog područja i otkrivanjem i selektovanjem ivica. Tekstualna oblast slike se zatim pretvara u ASCII znakove pomoću verovatnoće i algritama eksperstkih sistema. Većina proizvođača OCR aplikacija tvrdi da je prepoznavanje znakova sa 99% tačnosti kada se "čita" tekst sa slovima veličine od 8 do 36 pt i

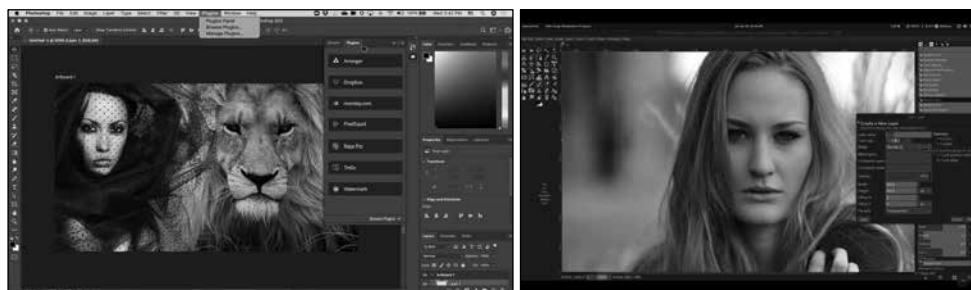
rezolucije 300 dpi i mogu da obrade oko 150 znakova u sekundi. Ovi programi, međutim, imaju problema u prepoznavanju znakova sa loših kopija originala, gde su ivice znakova neprepoznatljive ili problematične. Ponekad je lakše prekucati neki tekst, nego ga propustiti kroz OCR softver, koji će napraviti mnogo grešaka i korisnik će izgubiti više vremena za njihovo ispravljanje.



Slika 7.4. ABBYY FineReader

7.3. Vektorska i rasterska grafika

Programi za slikanje i crtanje, kao i za 3D modelovanje, su možda najznačajnije alatke jer, od svih multimedijalnih elemenata, grafički ugođaj projekta će verovatno imati najveći uticaj na krajnjeg korisnika. Ako je umetničko delo amaterski urađeno, ako je neinteresantno i nezanimljivo, onda je sigurno da će i autor i korisnici biti razočarani.



Slika 7.5. Adobe Photoshop i GIMP

Programi za obradu slika i slikanje, kao što su *Photoshop*, *Fireworks* ili *GIMP*, posvećeni su kreiranju i izradi bitmapiranih ili rasterskih slika. Aplikacije ili programi za uređivanje slika su specijalizovana i moćna su sredstva za poboljšanje i retuširanje postojećih bitmapiranih ili rasterskih slika. Ovi programi su neophodni za upotrebu slika koje se koriste u multimedijalnim prezentacijama. **Programi za crtanje**, kao što su *CorelDRAW*, *Illustrator*, *FreeHand* ili *Inkscape*, posvećeni su kreiranju i izradi "linijskih" ili vektorskih crteža koji mogu lako da se odštampaju na papiru u visokoj rezoluciji. Neke softverske aplikacije kombinuju mogućnosti i za crtanje i za slikanje, ali mnogi authoring sistemi mogu

da uvoze u svoj radni prostor samo bitmapirane ili rasterske slike. Tipično, bitmapirane ili rasterske slike obezbeđuju najveći izbor umetniku za pružanje i prikazivanje finih detalja i efekata i danas se rasterske slike češće koriste u multimediji umesto vektorskih.



Slika 7.6. *CorelDRAW, Inkscape, FreeHand i Illustrator*

Da bi određeni grafički paket (bez obzira da li se radi o rasterskom ili vektorskom paketu) bio zadovoljavajući za upotrebu unutar multimedijalnog projekta, treba potražiti da li može da prikaže sledeće elemente, da li može da ispoštuje sledeće stvari i da li ima sledeće funkcije:

- intuitivan grafički korisnički interfejs sa padajućim menijima, statusnim linijama¹, kontrolnim paletama i okvirima za dijalog, koji omogućavaju korisnika brz i logičan izbor;
- podesive dimenzije, kako bi korisnik mogao da promeni veličinu velikih i malih rasterskih slika, da može da ih "isteže" i da ih iskrivi ili "izobliči";
- alate za kreiranje geometrijskih oblika, od kvadrata do krugova i od krivih linija do kompleksnih poligona (pravilnih mnogouglova);
- sposobnost da ispuni bilo koju oblast jednom bojom, odgovarajućim šablonima jedne ili više boja, kao i gradijentnim prelazima (linernim i/ili radijalnim) dve i više boja;

¹Područje na dnu mnogih programskih prozora operativnog sistema *Microsoft Windows*, koje sadrži kratku tekstualnu poruku o tekućem stanju programa. Neki programi na statusnoj liniji prikazuju i objašnjenje trenutno izabrane komande iz menija.

- sposobnost da se oslikava određeni element upotrebom gotovih šablona i biblioteka gotovih sličica (*clip arts*);
- sposobnost da menja i podešava oblike i veličine olovaka za crtanje, kao i četkica za oslikavanje;
- alat za određivanje i prepoznavanje uzoraka boja (*eyedropper tool*);
- alat za automatsko opcrtavanje² (*autotrace tool*) koji omogućava korisniku da pretvori rastersku u vektorsku sliku, uz određena ograničenja;
- sposobnost da podržava promenu veličine familije fontova i sposobnost da primenjuje senčenje istih;
- sposobnost da više puta ponavlja odgovarajuću operaciju ili proces i da nema ograničenja po pitanju primene komande *Undo*;
- sposobnost da pamti šta rađeno do datog momenta, u slučaju da treba popraviti neke efekte, delove crteža ili slike i tekst;
- sposobnost da se vodi računa o primenjenim svojstvima ili osobinama;
- sposobnost da se snimi tekući ekran kao nezavisna slika koja može da se iskoristi u daljem radu;
- karakteristike oslikavanja ili slikanja kao što su ujednačavanje ili "glađenje" grubih ivica objekata u pozadini primenom uklanjanja nazubljenosti³ (*anti-aliasing*), četkice raznih veličina, oblika, gustine i dezena, mešanje boja kroz gradijentne prelaze, kao i odgovarajuće zaklanjanje ili maskiranje delova elemenata ili celih elemenata;
- sposobnost da podržava odgovarajuće specijalne efekte prijateljskih softvera ili kuća;
- sposobnost da objekte i elemente razvrsta i čuva na različitim slojevima, što omogućava korisniku da iste tretiraju kao nezavisne objekte ili elemente;
- sposobnost zumiranja (uvećanja ili umanjenja) radi uređivanja pojedinačnih piksela kod rasterskih slika;
- sposobnost da podržava zajedničke dubine boja: 1-bitne, 4-bitne, 8-bitne, 16-bitne, 24-bitne ili 32-bitna boje, kao i sivu skalu, bez ograničenja;
- sposobnost da se dobro upravlja bojama i da se omogućava primena međutonova (*dithering*) između različitih modela boja kao što su RGB, HSB i CMYK;
- sposobnost dobrog upravljanja paletama boja kada se nalazi u 8-bitnom radnom modu;

²Funkcija programa za crtanje koja isticava linije duž ivica slike predstavljene bitovima radi konverzije slike u objektno orijentisanu sliku.

³Softverska tehnika za izravnavanje nazubljenog izgleda krivih ili dijagonalnih linija koji je posledica slabe rezolucije ekrana. Izravnavanje linija se postiže umetanjem senki između piksela i manipulisanjem veličinom i horizontalnim poravnavanjem piksela.

- sposobnost nesmetanog uvoza i izvoza grafičkih formata, kao što su PIC, GIF, TGA, TIFF, PNG, WMF, JPG, PCX, EPS, PTN i BMP.

Ako se korisnik baš ne snalazi u multimediji i sa ovim alatima, onda mu treba malo vremena da ispita više od jednog grafičkog paketa, kako bi sagledao mogućnosti šta koji paket može da uradi i na koji paket može da se osloni. Ako ne može sâm da ovo odradi, onda takav korisnik treba da pronađe nekog ko je već upoznat sa grafičkim aplikacijama, jer korisnik može da provede nekoliko dana učeći da koristi softver za crtanje ili slikanje i da ne postigne dobre rezultate. To je već izgubljeno silno vreme, a rezultat je nula. Treba napomenuti da mnogi korisnici (umetnici) nauče da koriste jedan, moćan softverski alat, ali nauče da ga koriste izuzetno dobro.

7.4. 3D modelovanje i animacija

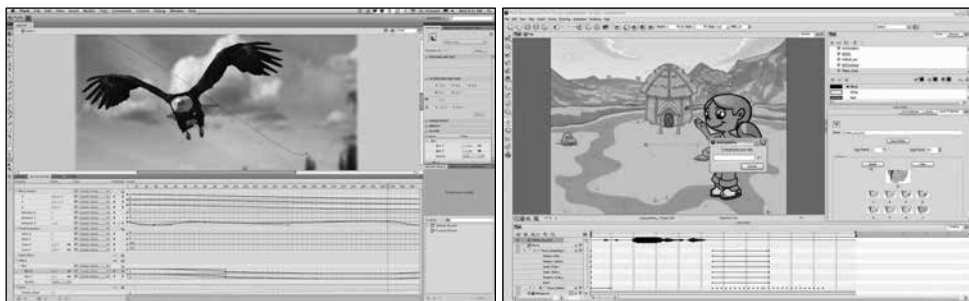
Softveri za 3D modelovanje su ušli u sferu grafičkog dizajna i na svoj specifičan način unapredili su kompletnu priču po pitanju multimedijalnog projekta. Skraćenica CAD (*Computer Aided Design*) "pokriva" softvere koji koriste inženjeri, arhitekte, umetnici i drugi u cilju stvaranja preciznih crteža ili tehničkih ilustracija. Mogu se koristiti za stvaranje dvodimenzionalnih (2D) ili trodimenzionalnih (3D) modela. CAD "slike" mogu da se "okreću" u prostoru, sa preciznim simuliranjem svetlosnih izvora i precizno definisanim i prikazanim senkama. Kada se 3D CAD modeli u pitanju, autor može da stoji ispred svog rada i da ga gleda iz bilo kog ugla i da, na osnovu toga, donese sud o njegovom dizajnu. U ovu kategoriju softvera spadaju *AutoCAD*, *3ds Max*, *Maya*, *SolidWorks*, *CATIA*, *Pro/ENGINEER* i još mnogi drugi.



Slika 7.7. *AutoCAD*, *3ds Max*, *SolidWorks* i *CATIA*

Oznaka 3D predstavlja skraćenicu za tri dimenzije ili trodimenzionalni izgled. Dok su u 2D grafičkom programu slike izrađene duž x (po horizontali ili u širinu) i y (po vertikali)

ili u visinu) osa, u 3D programima se dubina oslikava kroz dodavanje treće dimenzije duž z ose. Svaki 3D program koji je objekte smestio po slojevima (*layers*), mora da zna vrednosti objekta po z osi. Na primer, veb pretraživači objekte na ekranu smeštaju pomoću *z-index* CSS atributa. Neki softverski programi (kao što su *Adobe Flash* i *Toon Boom Studio*) mogu da simuliraju dubinu automatskim skaliranjem slike na osnovu vrednosti za z osu, kako bi se simulirao 3D efekat. Ovo se razlikuje od pravog 3D modelovanja i renderinga⁴, gde objekti mogu da se rotiraju i posmatraju iz bilo kog pravca ili ugla.



Slika 7.8. *Adobe Flash i Toon Boom Studio*

Kada se radi unutar softvera za 3D modelovanje, objekti renderovani u perspektivi su realniji; mogu da se stvore fantastične scene izborom pravog osvetljenja i perspektive za krajnju renderovanu sliku. Odlični paketi za modelovanje, kao što su *Vectorworks*, *Autodesk Maya*, *Strata* i *Autodesk SoftImage*, omogućavaju renderovanje "objekata" kao što su ljudi, nameštaj, zgrade, automobili, avioni, drveće i fabrike ili postrojenja. Softver *Blender* je odličan i besplatan program za 3D modelovanje i nudi širok skup korisnih funkcija. Program *Google SketchUp* je besplatan program za 3D modelovanje sa ograničenim mogućnostima, ali sa velikim onlajn bibliotekama komponenti. Izuzetno je važno za razvoj multimedijalnog projekta što mnoge aplikacije za 3D modelovanje uključuju mogućnost da izvezu putovanje kroz scenu ili određeno kretanje kao video materijal u obliku *QuickTime* ili *MPEG* fajlova.

Kreiranje svake završne 3D renderovane slike može da traje od nekoliko sekundi do nekoliko sati, u zavisnosti od složenosti crteža i broja nacrtanih objekata uključenih u taj crtež. Ako se pravi složen prolaz kroz definisanu scenu ili se simulira preletanje iste, onda mora da se odvoji mnogo sati rada računara da bi se odradio rendering te scene i da bi se dobio kvalitetan materijal za prikazivanje na računaru.

Dobar alat ili softver za 3D modelovanje trebalo bi da sadrži sledeće karakteristike:

- više prozora koji omogućavaju autoru da vidi svoj model u svakoj dimenziji (duž svake ose), iz perspektive kamere, kao i prethodni pregled (*preview*) renderovanog modela ili scene;

⁴Stvaranje slike koja sadrži geometrijske modele korišćenjem boja i senčenja za postizanje realističnog izgleda. Komanda *render*, koja je obično deo paketa za geometrijsko modelovanje kao što je *CAD*, koristi matematiku za definisanje lokacije svetlosnog izvora u odnosu na objekat i za proračun načina na koji će svetlost stvarati osvetljene delove, senke i varijacije boja. Stepenn realističnosti se kreće od neprozirnih, osenčenih poligona do slika koje se po složenosti približavaju fotografijama.

- sposobnost prevlačenja i puštanja (*drag and drop*) primitivnih ili jednostavnih oblika unutar scene;
- sposobnost da izmodeluje i kreira organske objekte od nule;
- funkcije za modelovanje objekata rotacijom inicijalne konture oko proizvoljne ose i izvlačenjem inicijalne konture duž proizvoljne ose;
- sposobnost mapiranja boja i tekstura;
- mogućnost dodavanja realističnih efekata kao što su transparentnost, senke i magla;
- mogućnost dodavanja posebnih, lokalnih i globalnih svetlosnih izvora, da ih postavi bilo gde i da manipuliše njima u cilju kreiranja specijalnih efekata osvetljenja;
- mogućnost primene kamere bez nekih ograničenja kada su hardverske komponente u pitanju;
- sposobnost da kreira glatke krive linije ili krivulje (*splines*) koje će služiti kao putanja za kretanje kamere i za kreiranje odgovarajuće animacije.

7.5. Uređenje slike

Aplikacije za uređenje slika su specijalizovane i predstavljaju moćno sredstvo za stvaranje, unapređenje, i retuširanje postojećih rasterskih ili bitmapiranih slika. Moderne verzije pomenutih aplikacija pružaju mnoge mogućnosti, kao i mnogo alata za slikanje, a mogu da se koriste i za stvaranje slika od nule, kao i slika digitalizovanih od strane skenera, digitalnih kamera ili datoteka sa umetničkim delima kreiranim u drugim paketima za slikanje ili crtanje.

Biće navedene neke karakteristike koje su tipične za aplikacije za uređivanje i obradu slika i od interesa su za razvijaoce multimedijalnih projekata:

- više prozora koji omogućavaju autoru da vidi više od jedne slike u isto vreme;
- mogućnost konverzije glavnih tipova slika i podataka i industrijskih standardnih formata;
- mogućnost direktnog učitavanja slike iz skenera ili video izvora;
- mogućnost upotrebe šeme virtualne memorije koja omogućava korišćenje prostora na hard disku kao RAM memorije za slike koje zahtevaju velike količine memorije;
- mogućnost upotrebe određenih alata za selekciju (na primer, *Rectangle*, *Lasso*, *Magic Wand selection tools*) u cilju izbora delova bitmapirane ili rasterske slike;
- mogućnost prikazivanja i kontrole osvetljenja, kontrasta i balansa boja;
- dobre karakteristike maskiranja;

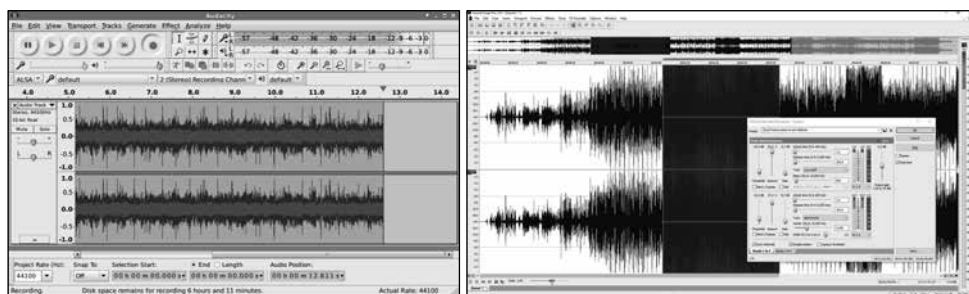
- mogućnost primene višestrukog poništavanja komandi i radnji i njihovo vraćanje na prvobitno stanje;
- sposobnost uklanjanja nazubljenosti (*anti-aliasing*), sposobnost izoštravanja i podešavanje oštine prikaza (*sharpening*), kao i sposobnost ujednačavanja samog prikaza (*smoothing*);
- sposobnost kontrolisanja mapiranja boja u cilju preciznog podešavanja balansa boja;
- alati za retuširanje (*retouching*), zamućenje (*blurring*), izoštravanje (*sharpening*), osvetljenje (*lightning*), zatamnjenje (*darkening*), razmazivanje (*smudging*) i nijansiranje (*tinting*);
- sposobnost primene geometrijskih transformacija, kao što su preslikavanje objekata gde novi objekat predstavlja sliku u ogledalu originala (*flip, mirror*), smicanje duž odgovarajućih osa ili ivica (*skew*), rotiranje (*rotate*), izobličenje (*distort*) i menjanje u cilju poboljšanja prikaza objekata ili celokupne scene u perspektivi;
- sposobnost da se promeni veličina slike (*resize*) i da se promeni broj piksela na slici (*resample*);
- sposobnost da podržava 24-bitne boje, 8-bitne ili 4-bitne indeksirane boje, 8-bitnu sivu skalu, crno-belu preraspodelu boja i prilagodljive palete boja, bez ograničenja;
- sposobnost da se stvori slika od nule, koristeći alate za linije, pravougaonike, kvadrate, krugove, elipse, poligone, sprejeve, četkice, olovke i gumice, sa prilagodljivim oblicima četkica i sa definisanim gradijentnim prelazima dve i više boja od strane korisnika ili autora;
- podržavanje različitih i višestrukih pisama, stilova i veličina, kao i rutinsko odrađivanje po pitanju manipulisanja i maskiranja istih;
- filteri za specijalne efekte;
- podrška dodataka za specijalne efekte koji dolaze "sa strane" ili od strane "trećih lica";
- sposobnost da dizajnira u različitim slojevima (*layers*) koji mogu da se kombinuju, skrivaju, uklanjaju i reorganizuju.

7.6. Uređenje zvuka

Alati za uređivanje zvuka, i digitalizovanog i MIDI (*Musical Instrument Digital Interface*), omogućavaju autoru i da "vidi" i da čuje muziku. Oslanjajući se na predstavljanje zvuka u obliku talasa, autor može da izreže (*Cut*), kopira (*Copy*), umetne (*Paste*) i uređuje (*Edit*) segmente zvuka sa velikom preciznošću i da kreira sopstvene zvučne efekte.

Korišćenje alata za uređivanje ili editovanje zvuka (na primer, *Audicity* i *Sound Forge*) traži od autora da zna sve što treba o svojim MIDI fajlovima (trebalo bi da bude muzički,

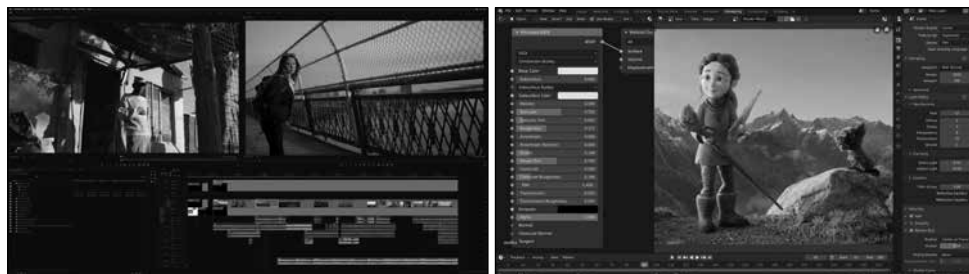
bar malo, "potkovan", da ne bi "lutao" u mraku), kao i da ima povezan MIDI sintisajzer ili drugi "muzički uređaj" sa računarom.



Slika 7.9. Audacity i SoundForge

7.7. Uređenje animacija, videa i digitalnih filmova

Animacije i digitalni video filmovi predstavljaju sekvence bitmapiranih grafičkih scena (kadrova, *frames*) koje se brzo reprodukuju. Trebalo bi napomenuti da animacije mogu, takođe, biti unutar *authoring* sistema gde se brzo menjaju lokacije ili položaji objekata ili sličica kako bi se stvorio utisak kretanja.



Slika 7.10. Adobe Premiere Pro i Blender

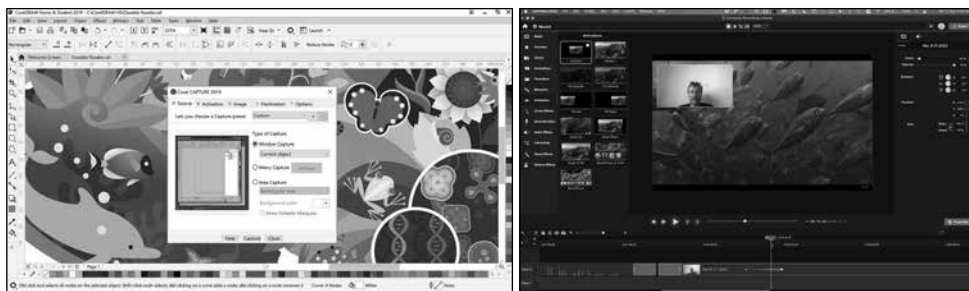
Alati za kreiranje filmova (na primer, *Premiere*, *Flash*, *Blender*, *OpenShot Video Editor* i *Windows Movie Maker*) dozvoljavaju autoru da uređuje, sastavlja i uklapa video snimke dobijene iz različitih kamera, animacija, skeniranih slika i segmenata drugih digitalizovanih filmova.

Kompletiranom snimku se često dodaju tranzicioni i vizuelni efekti, koji dolaze do izražaja prilikom reprodukovanja završenog materijala.

7.8. Korisni pribori

Nijedan multimedijalni alat nije potpun bez nekoliko neophodnih alata za obavljanje nekih čudnih, ali ponavljajućih zadatak. To su korisni pribori ili dodaci.

Na primer, mogućnost "skidanja" ili snimanja dela ekrana ili monitora je od suštinskog značaja, jer rasterske ili bitmapirane slike su toliko česte u multimediji, te je važno imati alat za "hvatanje" celog ili dela prikaza na ekranu. Autor može da "uveze" ovakve segmente u originalni sistem ili da ih kopira u aplikacije za uređivanje ili editovanje slika. U programe koji omogućavaju da se snimi ekran ili deo ekrana, pored ostalih, spadaju i *Screen Capturer*, *Corel CAPTURE*, *Camtasia Studio*, *CamStudio* itd.



Slika 7.11. *Corel CAPTURE i Camtasia Studio*

Dodatni neophodni alati u razvoju multimedijalnog projekat su **konvertori** ili **pretvarači formata** (*Format Converters*), jer izvorni materijali mogu da potiču sa *Macintosh* računara i pratećeg operativnog sistema, sa personalnih računara sa operativnim sistemom *Windows*, sa radnih stanica sa operativnim sistemom *Linux*, pa čak i sa velikih centralnih računara⁵ (*mainframe computers*). Ovo je pitanje posebno bitno za video i audio fajlove, jer postoji mnogo formata i mnogo šema komprimovanja.

⁵Računar visokog nivoa projektovan za najintenzivnije računarske zadatke. Velike centralne računare često zajednički koristi više korisnika posredstvom terminala povezanih sa računarom.

Glava 8

Planiranje i troškovi multimedijalnog projekta

Pre nego što počnu sa radom na multimedijalnom projektu, autori moraju prvo da steknu utisak o obimu i sadržaju istog, kao i da stvore odgovarajuću sliku u sopstvenim glavama o dostupnim metodama i resursima kako bi preneli željene poruke do potencijalnih korisnika. Nakon toga, treba da razviju uređen nacrt i plan koji je racionalan u smislu veština, vremena, budžeta, alata i sredstava koje imaju pri ruci. Pravilno planiranje projekta je podjednako važno kao i planiranje rasporeda i sadržaja. Ovi planovi treba da budu definisani i predstavljeni članovima tima pre nego što se počne sa radom na grafičkim elementima, zvukovima, video materijalima i drugim komponentama, a treba voditi računa i o ispunjenju pojedinačnih elemenata ili etapa tokom izvršenja projekta.

Preporuka je da se planira celokupni proces – počevši od prvih ideja, a zaključno sa završetkom i isporukom gotovog proizvoda. Što bi se reklo, treba gledati ukupnu ili širu sliku. Postepen proces izrade multimedijalnog projekta, kroz blok dijagram, prikazan je na slici 8.1. Treba koristiti ovaj blok dijagram kao pomoć pri izradi novog veb sajta ili DVD produkcije. U blok dijagramu su evidentne i povratne petlje za revizije zasnovane na testiranju i eksperimentima. Treba imati na umu i konstantno prisustvo **komisije za ocenjivanje** (koja može biti jednostavno menadžer projekta) da nadgleda celu priču.

8.1. Analiza

Treba posebno napomenuti da bi pre početka multimedijalnog projekta, trebalo utvrditi njegov obim i sadržaj. Balans ili ravnoteža i kompromis su ključni principi u analizi ideja. Cilj je da se stvori akcioni plan koji će postati mapa puta za kreiranje i proizvodnju. Treba voditi računa o osnovnim elementima. Kome treba pomenuti projekat? Da li je to vredno? Da li postoji materijal pri ruci da se izgradi? Da li članovi tima poseduju potrebne veštine da se izgradi? Ideja će biti izbalansirana ako se vodilo računa o odgovarajućim elementima:

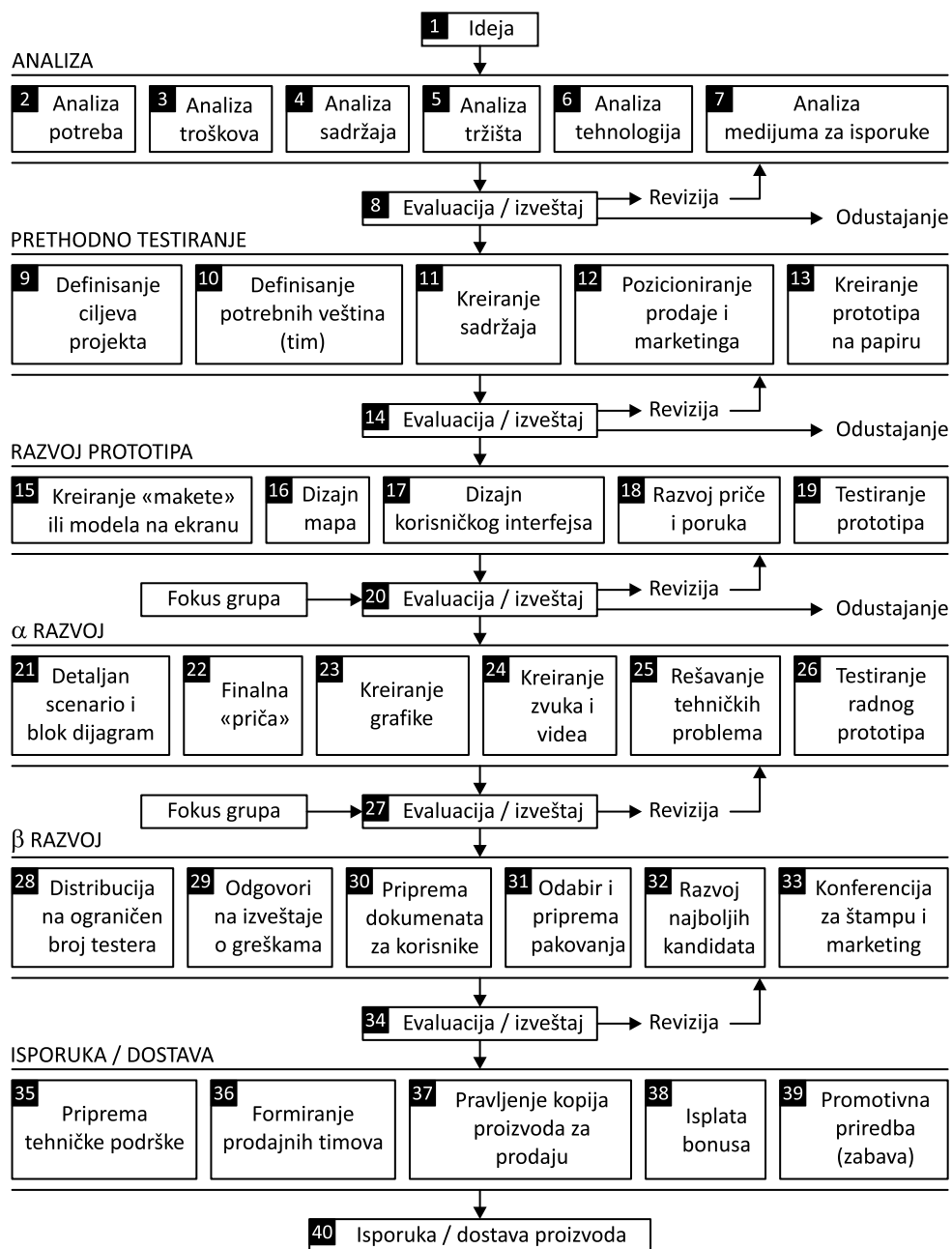
- Šta je suština onoga što autor želi da uradi? Koja je njegova svrha i poruka?

- Ko je ciljana publika? Ko će biti krajnji korisnici? Šta oni već znaju o ovoj temi? Da li oni razumeju određene izraze i termine (žargonski) i koje informacije trebaju da znaju da bi projekat "komunicirao" sa njima? Koje platforme će koristiti za reprodukciju multimedijalnih materijala i koje su minimalne tehničke mogućnosti ovih platformi?
- Da li postoji klijent i šta klijent želi?
- Kako organizovati projekat?
- Koji će multimedijalni elementi (tekst, zvuk i vizuelni elementi) najbolje preneti poruku autora?
- Da li već postoji određeni materijal koji može da se iskorist za projekat, kao što su stare video kasete ili video fajlovi, muzika, dokumenti, fotografije, logotipi, reklame, marketinški paketi i druga umetnička dela?
- Da li je potrebna interaktivnost?
- Da li autorova ideja potiče od postojećih tema koje se mogu pojačati multimedijalnim sadržajem ili će autor napraviti nešto potpuno novo?
- Koji hardver je na raspolaganju za razvoj projekta? Da li je dovoljno?
- Koliko prostora za skladištenje postoji? Koliko skladišnog prostora je potrebno?
- Koji multimedijalni softver je dostupan za izradu projekta?
- Koji su sposobnosti i veštine potrebne da bi se radilo i sa hardverom i sa softverom?
- Može li autor da odradio celokupan posao samostalno? Ko može da pomogne?
- Koliko se vremena ima da se uradi projekat?
- Koliko se novca ima da se uradi projekat?
- Kako će se distribuirati finalni projekat?
- Hoće li biti potrebno da se finalni proizvod povremeno ažurira?
- Da li je neophodna stručna podrška po pitanju finalnog proizvoda?

Neophodno je da se stalno **utvrđuje svrha** ili **cilj** za i/ili protiv izvodljivosti multimedijalnog projekta, kao i definisanje troškova proizvodnje i isporuke. To bi moglo da se uradi (doradi) dinamičkim putem, tj. "stalnim" **dodavanjem** ili **oduzimanjem** elemenata iz projekta. **Proces dodavanja** ili **aditivni proces** "kreće" od minimalnih mogućnosti i postepeno se dodaju elementi, ako je to moguće. **Proces oduzimanja** ili **subtraktivni proces** uključuje odbacivanje nepotrebnih elemenata iz potpuno razvijenog projekta. Oba procesa, i aditivni i subtraktivni, mogu da rade zajedno i mogu dati veoma korisne procene troškova i mapu puta za proizvodnju.

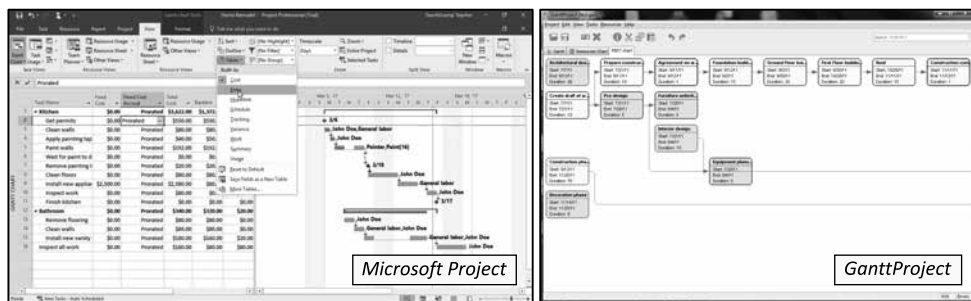
Vreme provedeno u definisanju sopstvenog projekta na ovaj način – sa jedne strane realnost i realni zahtevi, a sa druge strane testiranje po pitanju tehnologije i potrebnih znanja

i veština – može biti najvrednija investicija autora, čak i pre nego što upali računara. U svakom trenutku autor može da odluči da li ide napred ili će odustati.



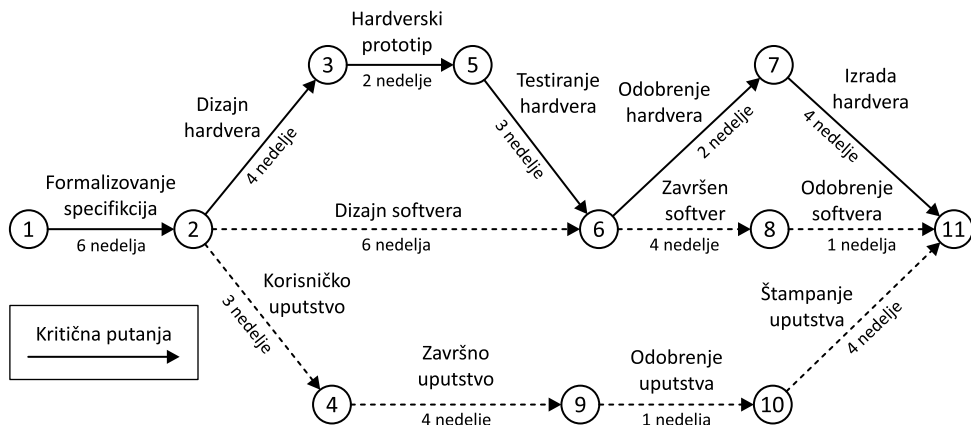
Slika 8.1. Proces izrade multimedijalnog projekta

Softverski programi kao što su *Microsoft Project* (slika 8.2 levo), *dotProject*, *OpenProj* i *GanttProject* (slika 8.2 desno) i programi za tabelarno izračunavanje, kao što je *Microsoft Excel*, mogu biti korisnik za uređenja ideja i raznih zadataka, radnih predmeta, sredstava zaposlenih i troškova neophodnih za realizaciju multimedijalnog projekta. Programski alati za upravljanje projektima obezbeđuju dodatnu korist u vidu analize koja će autoru pomoći da ostane unutar predviđenog rasporeda i budžeta tokom realizacije samog projekta.



Slika 8.2. Programski alati za upravljanje projektima

Softver za upravljanje projektom obično primenjuje **metodu kritičnog puta** (*Critical Path Method*, CPM) koja obezbeđuje funkcije za izračunavanje ukupnog trajanja projekta na osnovu svakog identifikovanog zadatka, izdvaja zadatke koji su kritični i ako se ti zadaci produže, dovešće do kašnjenja u završetku projekta. Program *Evaluation Review Technique* (PERT) obezbeđuje grafički prikaz odnosa zadataka, preduslova i redosleda zadataka koji moraju biti završeni pre nego što drugi mogu da počnu (slika 8.3).



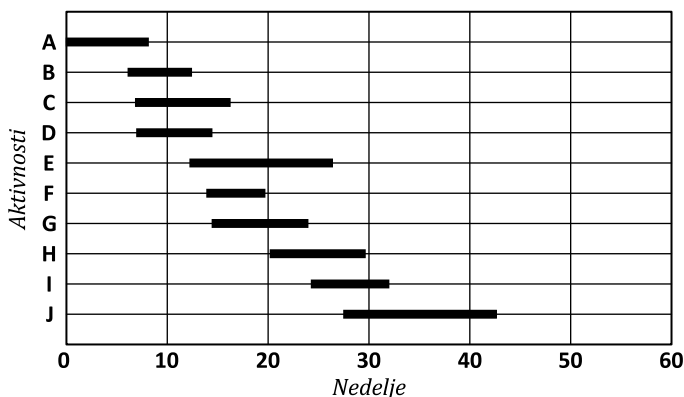
Slika 8.3. Primena PERT/CPM za projekat izrade PC ploče

Brojčane oznake sa slike 8.3 imaju sledeće značenje:

1. početak projekta;
2. sve specifikacije su završene;

3. dizajn hardvera je završen;
4. uputstvo za korisnike je završeno;
5. prototip radi;
6. hardver je u potpunosti funkcionalan;
7. PC ploča je završena;
8. softver je završen;
9. priručnik je završen;
10. priručnik je spreman za štampanje;
11. projekat je kompletan i završen.

Gantogram (*Gantt charts*) prikazuje sve zadatke duž vremenske linije. Gantogram daje grafički prikaz plana faza i aktivnosti projekta u vremenu, sa definisanim početkom, trajanjem, završetkom i neophodnim resursima potrebnim za njegovu realizaciju i prati izvršenja pojedinih aktivnosti (slika 8.4).



Slika 8.4. Osnovni prikaz gantograma

Osnovna slabost gantograma je u nemogućnosti preciznijeg definisanje aktivnosti, posebno njene važnosti, i istraživanja međuzavisnosti između različitih aktivnosti.

8.2. Prethodno testiranje

Ako je autor shvatio da projekat ima određenu "specifičnu težinu", onda treba krenuti na naredni korak, shodno slici 8.1. U narednom koraku treba definisati detaljnije ciljeve projekta, a treba definisati potrebne veštine (samim tim se definiše posrednim putem i tim), sadržaj i novac da bi se ispunili željeni ciljevi. Ako je zamišljen komercijalni proizvod, onda treba definisati i način kako ga prodati. Na kraju, treba izraditi prototip projekta na

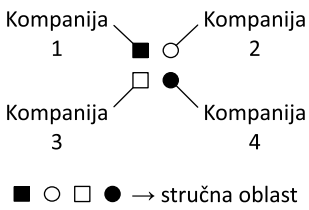
papiru, uz objašnjenje kako će raditi. Svi ovi koraci će pomoći autoru da organizuje svoju ideju i da je testira u okruženju realnog ili stvarnog sveta.

Kada se krene u multimedijalni projekat, može da bude mnogo zadataka koje treba izvršiti. Evo nekih stavki o kojima treba voditi računa i gde bi trebalo isplanirati šta i kada sa njima tokom realizacije definisanog projekta:

- dizajniranje razvojnog okruženja sa instrukcijama;
- određivanje platforme za isporuku;
- određivanje *authoring* platforme;
- ispitivanje dostupnog ili definisanog sadržaja;
- iscrtavanje navigacione mape;
- kreiranje scenarija;
- dizajniranje interfejsa;
- dizajniranje kontejnera ili spremišta sa informacijama ili za informacije;
- istraživanje i prikupljanje informacija za korektan sadržaj multimedijalnog projekta;
- izbor članova i kreiranje tima;
- izrađivanje prototipa;
- testiranje ponašanja korisnika;
- revidiranje dizajna, ako je to potrebno;
- kreiranje grafičkih elemenata, animacija, audio i video materijala;
- digitalizovanje audio i video materijala;
- pronalaženje i angažovanje odgovarajućeg fotografa;
- programiranje i autorstvo;
- testiranje funkcionalnosti;
- pronalaženje i fiksiranje ili ispravljanje grešaka;
- sprovođenje beta testiranja;
- kreiranje završnog materijala i pripremanje za pravljanje kopija;
- pravljanje potrebnog broja kopija;
- pripremanje odgovarajućeg pakovanja;
- isporučivanje materijala ili instaliranje istog na veb sajtu;

- određivanje bonusa koje treba podeliti zaslužnim članovima tima;
- promotivna priredba ili zabava.

Multimedija postaje narastajuća tehnologija koja zahteva potrebnih niz vještina, a taj niz je toliko veliki da i multimedija, sama po sebi, ostaje slabo definisana. Učesnici u ovoj tehnologiji dolaze iz svih oblasti računarstva i umetnosti, kao i iz raznih drugih disciplina, pa ako treba sastaviti tim, onda je to ozbiljan posao. Autor mora da zna koji i kakvi ljudi su mu potrebni, kao i potrebne veštine, da bi se krenulo u realizaciju multimedijalnog projekta i da bi se pomenuti projekat doveo do kraja.

Matrica iskustva i mogućnosti 	Trening ili obuka (proces)	Trening ili obuka (proizvod)	Prezentacije	Alati za bolje performanse	Simulacija proizvoda/prototip	Oglašavanje	Elektronska izdanja	Vizualne baze podataka/katalozi	Alati i aplikacije za korisnika	Gfraički korisnički interfejsi	Opšta edukacija	Edukacija krajnjeg korisnika	Edukacija po pitanju procesa	Štampani dokumenti/Uputstva	Pisani materijali (knjige)	Izveštavanje
Rukovodilac projekta	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
Istraživač	■		■	■	○	○	■	■	■	■	■	■		■	■	■
Pisac/Editor	■	■	■	■	○	■	■	■	■	■		■	■	■	■	■
Dizajner instrukcija	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■
Dizajner interfejsa	■	■	■	■	○	○	■	■	■	■	■	■	■	■	■	■
Stručnjak za ljudske resurse	■		■	■			■	■	■	■				■	■	■
Dizajner dokumenata	■		■	■		■	■	■	■	■				■	■	
Grafički dizajner	■	■	■	■	○	○	■	■	■	■	■	■		■	■	
Stručnjak za fotografije	■	■	■	■	○	○	■	■	■	■	■	○		■	■	
Ilustrator	■	■	■	■	○	○	■	■	■	■	■	○		■	■	
Specijalista za <i>authorware</i>	■	■	■	■	○	○	■	■	■	■	■	○				
Pisac/stručnjak za žargon	■	■	■	■	○	○	■	■	■	■	■	○	■		■	■
Programer	■		■	■	○	○	■	■	■	■	■	○				
Stručnjak za video		■	■	■	■	■		○	○	○	■	■	■		■	■
Fotograf	■	■	■	■	■	■	■	■	■	■	■	■	■	■		
Dizajner zvuka	■	■	■	■	■	■	■	■	■	■	■	■	■			

Slika 8.5. Matrični prikaz potrebnih i raspoloživih vještina za potencijalni projekat

Definisanje i izgradnja matrice potrebnih vještina (slika 8.5) često pomaže da se opiše sastav potrebnog tima. Treba napomenuti da potrebne veštine i potreban softver i nisu neko ograničenje, koliko ograničenje može da bude potreban hardver. Unutar budžeta uvek

može da se pronađe sredstava ako je potrebno obučiti deo tima za upotrebu nekog boljeg softvera, koji će omogućiti lakše i efikasnije realizovanje projekta. Treba napomenuti da je authoring softver, obično, potreban samo za razvoj projekta, a ne za njegovu reprodukciju ili isporuku, tako da troškovi korišćenja ovakvog softvera ili obuka članova tima za njegovo korišćenje ne spadaju u direktne troškove koji se zaračunavaju krajnjim korisnicima.

8.3. Razvoj prototipa

Kada autor odluči da je projekat u redu i da je vredan truda i izrade, trebalo bi da se razvije radni prototip. To je tačka u kojoj počinje ozbiljan rad na računaru, izgradnja interfejsa i pripadajućih menija i tastera. Kreće se u osmišljavanje načina kako će biti prezentovane željene poruke i kako će to uticati na krajnje korisnike. Za izgradnju prototipa, koji se ponekad naziva **konceptualni dizajn** ili **studija izvodljivosti**, može da se izabere samo mali deo velikog projekta i da taj deo radi kao da se nalazi u finalnom proizvodu. Nakon višestrukih različitih pristupa u izgradnji prototipova, autor može da predloži više od jednog mogućeg kandidata za finalni proizvod.



Slika 8.6. Konceptualni dizajn u "papiernoj" fazi

Treba testirati definisani prototip po pitanju nekoliko elemenata: **tehnologija** (da li će i kako raditi predložene platforme), **cena** (da li se može odraditi projekat u okviru budžetskih ograničenja), **tržište** (da li se finalni proizvod može prodati ili da li će se pravilno koristiti ako je reč o lokalnom karakteru projekta) i **ljudski interfejs** (da li je intuitivan i jednostavan za korišćenje). U ovom trenutku možda bi trebalo organizovati fokus grupu, gde autor može da "poveže" krajnje korisnike i prototip i da posmatra njihove reakcije. Cilj svakog prototipa je da testira početnu primenu autorove ideje i da je poboljša na osnovu rezultata testa. To, očigledno, znači da autor ne treba da bude sputavan ili da se "vezuje"

za neku opciju, nego treba da se bude spreman da eksperimentiše i da svojevolljno promene neke stvari.

Često se dolazi u situaciju da autor mora da ubedi klijenta da uloži ili potroši malu količinu novca i truda kako bi se izgradila verzija "skeleta" projekta, uključujući i neka umetnička dela, interaktivnu navigaciju i proveru performansi. Pokazalo se da mogu postojati neka vrlo specifična pitanja tehnologije koje je potrebno detaljno ispitati i potvrditi, pre nego što se, uopšte, krene u realnu procenu rada i potrebnih troškova. Na ovaj način će i autor i klijent moći da procene na pravi način ciljeve projekta i sredstva za njegovo ostvarenje.

Preporuka je da se pilot projekat iskoristi kao prva faza projekta. Na osnovu zaključaka pilot projekata, treba pripremiti izveštaj sa ključnim elementima i funkcionalnu demo verziju projekta. Autor će biti plaćen za dotadašnji rad, a klijent će dobiti pravi materijal za demonstraciju i prikazivanje njegovim šefovima i menadžerima. Ako je urađena demo verzija projekta dobra, to će biti ubedljiv argument u upravljačkoj hijerarhiji klijenta za prihvatanje sveobuhvatnog projekta.

Kao deo isporuke na kraju pilot faze, treba izvršiti preispitivanje procene potrebnih zadataka, kao i troškova. Treba pripremiti pismeni izveštaj i analizu budžeta, kao i predviđene dodatne troškove. Ovo je takođe pravo vreme da se razvije revidirani i detaljan plan projekta za klijenta. On omogućava klijentu određenu fleksibilnost i obezbeđuje proveru realnih elemenata za autora. U ovom trenutku može da se finalizira budžet i raspored plaćanja za nastavak projekta, kao i da se potpiše punovažni ugovor i utvrde procedure.

Teškoće mogu da nastanu ukoliko je klijent razočaran po pitanju količine isporučenog materijala ili na drugi način nije zadovoljan radom autora. Treba voditi strogu evidenciju o uloženom trudu i utrošenom vremenu, kako bi se izgladili potencijalni problemi. Treba imati stalno na umu da je u razvoj multimedijalnog projekta uključen veliki broj pokušaja (proba se jedna stvar, proba se druga stvar, opet se proba druga stvar sa nekim izmenama i tako redom), a kreativni proces neprimetno troši mnogo vremena i stvara mnogo troškova. Preporuka je da autor sluša reakciju klijenta kreiranog prototipa, jer mnogi problemi mogu brzo da se reše i svi konstruktivni komentari sigurno mogu biti uključeni u narednu fazu razvoja.

8.4. Alfa razvoj

Kako se napreduje sa projektom, treba stalno definisati zadatke unapred, jer mora da se vodi računa o potencijalnim problemima i teškoćama koji će se pojaviti tokom rada i treba se pripremiti za njih. Kako etapa **alfa razvoja** napreduje, očigledno je da se napori uvećavaju, a, u isto vreme, postaju više fokusirani. Više ljudi može da se uključi u projekat i fizionomija projekta polako počinje da biva vidljiva.

8.5. Beta razvoj

Do trenutka kada početna ideja dostigne **beta fazu razvoja**, autor je utrošio ozbiljno vreme, energiju i novac, i tada je, verovatno, prekasno za odustajanje. Došlo se do tačke

iz koje nema povratka na početak i treba nastaviti dalje. U ovoj fazi, u velikoj većini slučajeva, projekat izgleda dobro i funkcionalno. Većina karakteristika radi, a autor vrši distribuiranje projekta većem broju testera. Kada se pogleda sa strane, od ove tačke izgleda kao da projekat ide nizbrdo i autorova jedina briga treba da bude uspešno upravljanje projektom do unapred dobro zacrtanog cilja.

8.6. Isporuka/dostava

U vreme kada se dođe do ove faze, a reč je o **isporuci i dostavi**, autor može da računa na zaradu i dobit, jer se došlo do proizvodnje finalnog proizvoda. Brige autora su više okrenute ka tržištu i ka pitanju kako će projekat biti primljen od strane ciljane publike. Takođe, autor mora da se posveti velikom broju praktičnih detalja (na primer, ko će odgovarati na postavljena pitanja po pitanju podrške, gde treba locirati potrebni server i slično).

Glava 9

Dizajn i proizvodnja multimedijalnog projekta

Projektovanje i izgradnja multimedijalnih projekata su povezani procesi. Za projekte koji se distribuiraju putem CD-a i DVD-a, ulazni elementi projekata mogu da se menjaju sve dok se ne kaže da je gotovo i dok se projekti ne narežu na pomenute medije i isporuče. Najbolji proizvodi su često posledica uvažavanja povratnih informacija i modifikacija koje se sprovode tokom procesa proizvodnje; projekti čiji je dizajn prerano priveden kraju imaju nedostatke i smanjene su im šanse po pitanju poboljšanja. Treba biti oprezan. Postoji opasnost da previše povratnih informacija i previše promena mogu "ubiti" projekat, trošeći nenamenski i vreme i novac. Uvek treba predlagati uravnotežene izmene i usaglasiti predložene izmene sa predviđenim budžetom.

Za projekat koji je "vezan" za izradu veb sajta, dizajn može biti završen i realizovan, ali sadržaj ovog sajta može da se redovno ažurira i menja, tako da projekat (po sopstvenom dizajnu) nikada neće biti potpuno prekinut i završen ili "zamrznut". U takvim slučajevima, posebno je važno da se postavle jasni rokovi i smernice.

Kao što arhitekta stambenih zgrada moraju da shvate kako da koriste materijale koji su im dostupni, tako i autori multimedijalnih projekata moraju da razumeju prednosti i ograničenja elemenata koji ulaze u sastav njihovog projekta. Na primer, nema smisla za dizajn audio elemenata multimedijalnog projekta koristiti 16-bitni stereo zvuk od 44,1 kHz, kada na medijima za isporuku nema dovoljno prostora da se to snimi i preda klijentu; nema smisla praviti video snimke koji će se prikazivati brzinom od 30 fps putem interneta, kada krajnji korisnici koriste *dial-up* modeme (još uvek se negde koriste); nema smisla dizajnirati i optimizovati grafiku za rezoluciju 1.024×768 piksela za prikazivanje u školama, kada škole poseduju laptopove koji podržavaju rezoluciju od 800×600 piksela.

Dizajneri moraju da blisko sarađuju sa proizvođačima kako bi bili sigurni da će se njihove ideje pravilno shvatiti, a proizvođači treba da potvrde rezultate sa dizajnerima. Obostrana komunikacija je neophodna, jer neke stvari mogu da krene po zlu ili mnoge stvari treba korigovati u hodu (da li odgovaraju boje, može li da se promeni odgovarajuća sekvenca u animaciji, može li da se promeni određena funkcija koja usporava rad). Povratne sprege i dobra komunikacija između dizajna i proizvodnje su od ključnog značaja za uspeh projekta.

9.1. Dizajn multimedijalnog projekta

Dizajnerski deo projekta je mesto gde autorovo znanje i veštine rada na i sa računarima dolazi do izražaja; talenat u grafičkoj umetnosti, video i muzici; sposobnost da osmisli logične puteve kroz informacije kako bi svi bili fokusirani da stvore pravu stvar. Dizajn predstavlja sinergiju razmišljanja, izbora, odluka i rada. U to spadaju oblikovanje, ujednačavanje, prerađivanje, završno doterivanje, testiranje i uređivanje ili editovanje. Kada autor krene u dizajniranje svog projekta, njegove ideje i koncepti su postali jedan korak bliže realnosti. Nadležnost u fazi projektovanja je ono što odvaja amatere od profesionalaca u izradi multimedije.

U zavisnosti od obima projekta i veličine i sposobnosti tima, mogu da se primene dva pristupa u stvaranju originalnog interaktivnog multimedijalnog dizajna. Jedan pristup se ogleda u ulaganju velikih napora da se napravi detaljan **scenario**, opisujući projekat u tačnim detaljima – koji koristi reči i skice za svaki prikaz ekrana, zvukove i navigacione elemente, pa sve do specifičnih boja i nijansi, tekstualnih sadržaja, atributa i fontova, oblika tastera ili dugmadi, stilova, odgovora i glasovnih interpretacija. Ovakav pristup je posebno pogodan za timove koji su u stanju da brzo izgrade prototipove, a zatim da ih brzo pretvore u finalne proizvode. Drugi pristup se ogleda u kreiranju i upotrebi scenarija sa manje detalja i takvi scenariji se koriste kao grubi šematski vodiči, koji omogućavaju više slobode i kreativnosti u momentu rada i koji zahtevaju da se uloži mnogo više truda i rada prilikom izrade proizvoda na radnom mestu.

Metod koji se odabere zavisi od toga da li će isti ljudi raditi na kompletnom zadatku (i dizajniranje i projektovanje i implementacija), ili će implementacija biti zadatak koji će se prepustiti novom timu ili novim timovima, a onda je potrebno imati detaljne specifikacije (detaljan scenario i skice) koje će se proslediti novom timu ili novim timovima. Oba pristupa zahtevaju isto temeljno poznavanje alata i mogućnosti multimedije i oba pristupa zahtevaju scenario ili konture projekta. Prvi pristup je često favorizovan od strane klijenata koji žele da čvrsto kontrolišu troškove procesa proizvodnje i rada. Drugi pristup vodi ka bržoj realizaciji osnovnih elemenata, ali će se sigurno potrošiti više vremena na zadacima, jer će biti potrebno više iteracija i uređivanja kako bi se doterao projekat na željeni nivo tokom procesa rada. U svakom slučaju, što je više planiranja na papiru, biće bolje i lakše izgraditi projekat.

9.1.1. Dizajniranje ili projektovanje strukture

Multimedijalni projekat predstavlja više od uređenje teksta, slika, zvuka i video elemenata ili objekata. Način na koji se komponuju ovi elementi u interaktivnom smislu obliku potrebe samog autora i njegove poruke.

Kako se organizuje materijal, tako će projekat imati određeni uticaj na posmatrača po pitanju samog sadržaja. Zahvaljujući brzom rastu *World Wide Web*-a i omogućavanja pristupa milionima multimedijalnih elemenata; HTML dokumenti mogu da se povezuju sa ogromnim brojem drugih sličnih dokumenata u sajber prostoru, tako da novi dizajni i pronalasci mogu da doprinesu novoj medijskoj revoluciji – drugi ljudi mogu da otkriju rad autora i da vrše nadogradnje ideja i metoda, za sveopšte dobro, uz saglasnost autora.

Na nekim projektima, autor možda može da bude i dizajner i programer. To može da bude dobro zato što će razumeti dizajn na pravi način i što će izabrati odgovarajući put za njegovu implementaciju. Na ovaj način može da se osnaži predloženi dizajn, a programerske sposobnosti će mu omogućiti da lakše i za manje vreme ugradi potrebne funkcije kako bi ceo sistem radio kako treba.

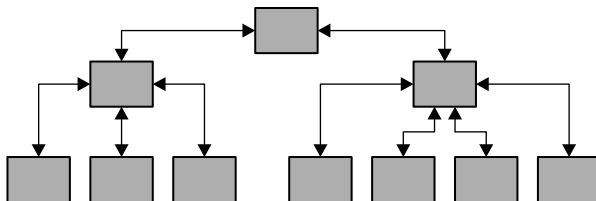
Navigacija

Mapiranje strukture projekta je zadatak koji treba da počne u fazi planiranja, jer na ovom mestu mape su u stanju da istaknu veze ili linkove između različitih oblasti sadržaja i mogu da pomognu da se organizuju sadržaj i poruke na pravi način. **Navigaciona mapa** (ili **mapa sajta**) omogućava tabelarni pregled sadržaja, kao i odgovarajući dijagram toka protoka informacija unutar interaktivnog interfejsa. Obično mapa sajta ima jednostavan hijerarhijski sadržaj gde je svaki naslov povezan sa pripadajućom stranicom, dok detaljniji dokument navigacione mape je kompleksniji i može biti od velike pomoći za projekat, navodeći multimedijalne objekte i opisujući šta se dešava kada korisnik reaguje na određenim mestima.

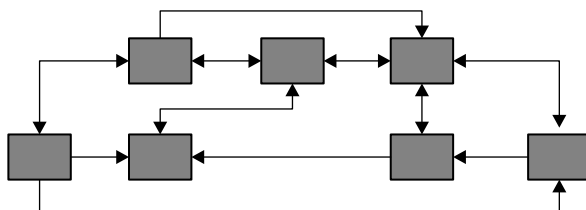
Linearna



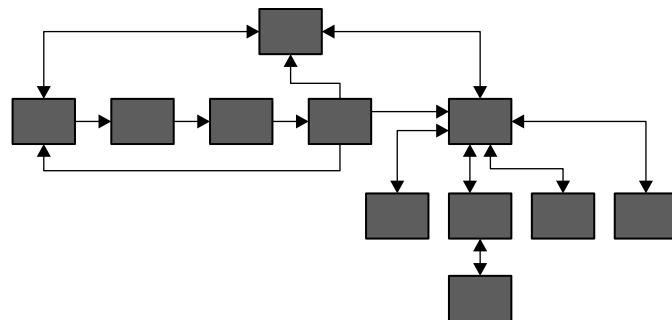
Hijerarhijska



Nelinearna



Kompozitna



Slika 9.1. Navigacione mape ili strukture

Nekoliko osnovnih struktura za multimedijalne projekte pokriva većinu slučajeva: linearna navigacija (struktura), hijerarhijska navigacija (struktura), nelinearna navigacija (struktura) i kompozitna navigacija (struktura). Na slici 9.1 prikazane su pomenute četiri osnovne organizacione strukture koje se koriste u multimedijalnim projektima, često u kombinaciji:

- **Linearna navigacija.** Korisnici se "kreću" redom, od jednog frejma sa informacijama do drugog.
- **Hijerarhijska navigacija.** Korisnici se "kreću" duž grana strukturalnog stabla, koje se uobličava prirodnom logikom sadržaja. Takođe se naziva i linearna navigacija ili struktura sa grananjem.
- **Nelinearna navigacija.** Korisnici se "kreću" slobodno kroz sadržaj, nevezano za prethodno utvrđene putanje ili puteve.
- **Kompozitna navigacija.** Korisnici mogu da se "kreću" nelinearno, ali povremeno su ograničeni linearnim prezentacijama podataka, jer je to najlogičniji put unutar definisane hijerarhije.

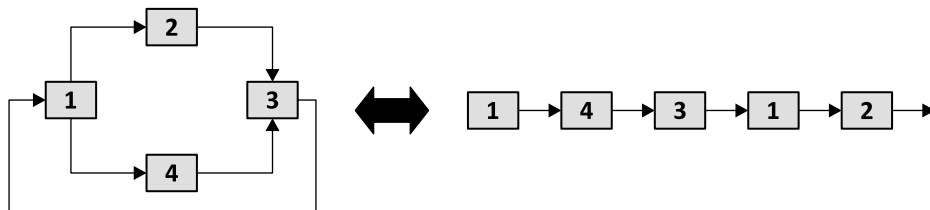
Metod koji obezbeđuje korisnicima navigaciju ili kretanje sa jednog mesta na drugo u projektu je deo korisničkog interfejsa. Uspeh korisničkog interfejsa ne zavisi samo od dizajna i implementiranih grafičkih elemenata, nego i od mnoštva tehničkih detalja (položaj interaktivnih tastera ili lokacija za aktiviranje ili aktuelnih lokacija¹ (*hot spots*) u odnosu na trenutne aktivnosti korisnika. Dobar korisnički interfejs je od ključnog značaja za ukupan uspeh projekta.

Dubina strukture

Korisnici mogu da projektuju svoje proizvode koristeći dve vrste strukture:

- **Dubinska struktura.** Predstavlja kompletnu navigacionu mapu i opisuje sve veze između svih komponenti projekta (slika 9.2, levo).
- **Površinska struktura.** Predstavlja strukture koje se, zapravo, realizuju od strane korisnika, dok prolazi "kroz" dubinsku strukturu (slika 9.2, desno).

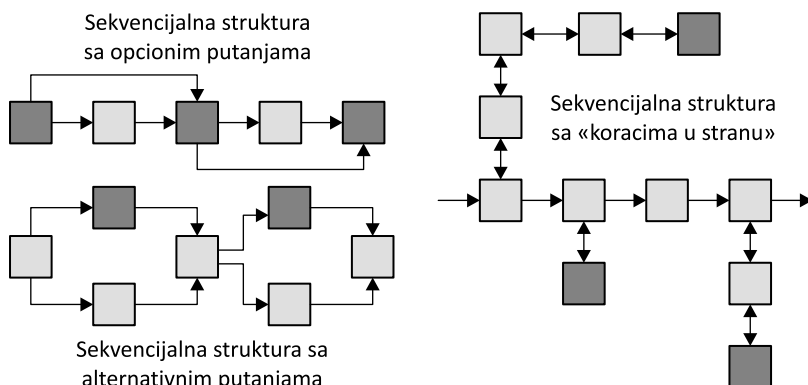
Očigledno je da dubinska struktura (slika 9.2, levo) može biti realizovana kao površinska struktura (slika 9.2, desno).



Slika 9.2. *Dubinska struktura (levo) i odgovarajuća površinska struktura (desno)*

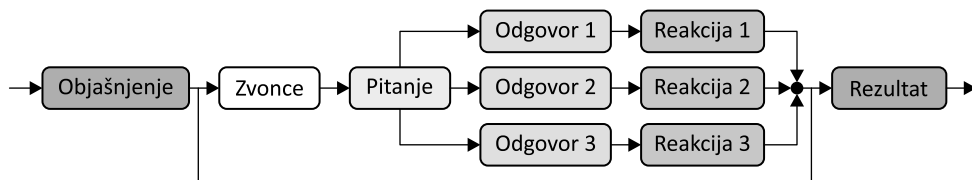
¹Položaj pokazivača miša, kao što su vrh strelice ili presek linija kod pokazivača u obliku krsta, koji označava tačnu lokaciju na koju će uticati radnja mišem, na primer pritisak na taster.

Neke površinske strukture generisane od strane korisnika mogu da imaju oblike prikazane na slici 9.3 i njihova primena zavisi od mnogo elemenata.



Slika 9.3. *Moguće površinske strukture generisane od strane korisnika*

Na slici 9.4 prikazan je dubinska struktura kviza koja se sastoji od tri moguće površinske strukture.



Slika 9.4. *Dubinska struktura koja se pretvara u tri površinske strukture*

Kada se dizajnira navigaciona mapa, treba razmišljati o površinskoj strukturi kako bi se video proizvod iz perspektive korisnika. Površinske strukture su od posebnog interesa za marketinške firme po pitanju praćenja korisničkih ruta kretanja kroz veb sajt kako bi se utvrdilo da li dizajn sajta efikasan i da li se uklapa u odgovarajuće preferencije samih korisnika. Kada su poznate preferencije korisnika, prilagođena iskustva korisnika po pitanju veb sajta mogu dinamički da se prilagode i da se dostave određenom korisniku ili grupi korisnika. Profilisanje korisnika i pripadajućih podataka je "vruća" tema, jer se zadire u privatnost korisnika i ovakve vrste podataka treba da se tretiraju kao medicinski i zdravstveni kartoni (nisu dostupni svakom i nisu dostupni bez odgovarajuće dozvole).

Navigacioni sistem treba projektovati na takav način da se korisniku daje mogućnost slobodnog izbora. "Tehnički" (u literaturi se nazivaju i "arhitektonski") crteži ili skice za multimedijalni projekat predstavljaju scenario i navigacione mape. Scenarija su "vezana" za navigacione mape tokom procesa dizajniranja ili projektovanja i pomažu da se vizualizuje informaciona arhitektura.

Aktuelne lokacije, veze, ikonice i tasteri

Većina multimedijalnih sistem omogućava da bilo koji deo monitora ili ekrana ili bilo koji objekat predstavlja **aktuelnu lokaciju** (*hot spot*). Kada korisnici prelaze kursorom miša

iznad aktuelne lokacije ili kliknu na nju, nešto se dešava, što čini multimediju ne samo interaktivnom, nego i uzbudljivom. Na osnovu odgovarajućih zaključaka, treba obratiti pažnju na sledeće elemente:

- Treba insistirati i dodavati interaktivnost multimedijalnom projektu kad god je to moguće i kada ne zahteva mnogo truda.
- U tri bitne kategorije za definisanje aktuelnih mesta spadaju tekst, grafika i ikonice.
- Najjednostavnije aktuelne lokacije na internetu su tekstualni linkovi koji povezuju dokument sa drugim delom istog dokumenta, dugim dokumentima ili drugim sajto-vima.
- Veće slike (*image maps*) su smeštene na odgovarajućim mestima sa pripadajućim linkovima.
- Ikonice (*icons*) su osnovni grafički objekti koji simbolizuju neke aktivnosti ili koncept (kada je operativni sistem *Windows* u pitanju veličina ikonice je 256×256 piksela, a kada je operativni sistem *Macintosh* u pitanju zastupljene su veličine ikonica 16×16 , 32×32 , 48×48 , 128×128 , 256×256 i 512×512 piksela).
- Dugmad ili tasteri (*buttons*) predstavljaju grafičku sliku aktuelne lokacije ili linka (preporuka je da se dodaju svetlosni efekti dugmadima jer je to najčešći način da željena dugmad "upadnu" u oči i budu vidljiva za korisnika).

9.1.2. Dizajniranje ili projektovanje korisničkog interfejsa

Korisnički interfejs multimedijalnog proizvoda je spoj pripadajućih grafičkih elemenata i sopstvenog sistema za navigaciju. Ako su poruke i sadržaj neorganizovani, ako ih je teško pronaći ili ako korisnici postanu dezorijentisani ili im tokom vremena postane dosadno, onda je očigledno da multimedijalni projekat možda neće uspeti. Loša grafika može da izazove dosadu. Loša navigacija može da utiče da se korisnici osećaju izgubljeno i da ne mogu da se povežu sa sadržajem. Najgore moguće je da korisnici, posle određenog vremenskog perioda, odustanu i napuste program. Kada se tako nešto desi, praksa je pokazala da se korisnici ne vraćaju na adresu sajta ili ne gledaju proizvode određene firme ili autora.

Interfejs za početnika i za iskusnog korisnika

Najjednostavnije rešenje za rukovanje korisničkim interfejsom od strane korisnika različitih nivoa stručnosti je obezbeđivanje **modularnog interfejsa**. U modularnom interfejsu, korisnik može jednostavno da klikne na dugme *Novice* (dugme koje će aktivirati određeni moduo gde i početnik može da se snađe) ili dugme *Expert* (dugme koje će aktivirati kompleksniji korisnički interfejs za iskusnog korisnika) i promeniće se pristup celom interfejsu.

Pokazalo se da modularni interfejsi **nisu pogodni** za multimedijalne projekte. Rešenje je da se izgradi projekat (korisnički interfejs) koji može da sadrži dovoljno jasnih navigacionih smernica, koje će pružiti pristup sadržaju i zadacima korisnicima na svim nivoima

njihovog znanja. Ne bi trebalo posebno napominjati da korisnički interfejs mora da bude jednostavan i razumljiv.

Grafički korisnički interfejs i grafički pristup

Grafički korisnički interfejs² (*Graphical User Interface*, GUI) kompanija *Microsoft* i *Macintosh* su uspešni zbog svoje jednostavnosti, doslednosti i olakšanog korišćenja. Uobičajeno je da GUI imaju ugrađene sisteme pomoći i pružaju standardne obrasce aktivnosti koji proizvode standardne očekivane rezultate.

Dizajniranje odličnog prikaza na računarskom monitoru zahteva poseban skup likovnih sposobnosti, a treba napomenuti da ne mogu svi programeri ili svi likovni umetnici da se bave kreiranjem računarske grafike. Kao programeri što moraju da drže korak sa aktuelnim operativnim sistemima i programskim jezicima, tako i eksperti za računarsku grafiku, takođe, moraju da ostanu informisani o brzomjenjajućim funkcijama, tehnikama, aplikacijama i kreativnim alatima. Grafičko umetničko delo mora biti prikladno ne samo za projekat, nego mora da bude dobro za korisnika.

Grafički pristupi koji su se **uspešno pokazali** u dosadašnjem radu i primenama su:

- uredno primenjeni kontrasti u raznim oblicima: veliko i malo, teško i lako, svetlo i mračno, tanko i debelo, usko i široko;
- jednostavni i čisti ekrani sa dosta belog prostora, tj. "neinformativnih oblasti";
- senčenje i primena senki u različitim nijansama;
- gradijentni prelazi;
- inverzna grafika da se naglasi bitan tekst ili slika;
- senčenje objekata i teksta u 2D i 3D;
- upotreba elemenata koji će privući pogled.

Grafički pristupi, koje **treba izbegavati**, uopšteno mogu da se smeste u naredne grupe:

- sukobi u boji (nekomplementarne boje);
- pretrpani ekrani koji zahtevaju više od dva klika na određeno dugme kako bi se napustio taj isti ekran;
- previše brojeva i reči;
- previše materijalnih elemenata koji se menjaju ili predstavljaju prebrzo.

²Okruženje koje programe, datoteke i opcije predstavlja pomoću ikona, menija i okvira za dijalog na ekranu. Korisnik bira i aktivira opcije postavljanjem pokazivača na nju i pritiskanjem tastera miša, a često i pomoću tastature. Određeni element (kao što je traka za pomeranje sadržaja) radi za korisnika na isti način u svim aplikacijama, jer grafički korisnički interfejs obezbeđuje standardne softverske rutine za rad sa ovim elementima i izveštavanje o akcijama korisnika (kao što je biranje njene ikone mišem, biranje mesta u tekstu, ili pritisak na taster); aplikacije pozivaju ove rutine sa određenim parametrima, a ne pokušavaju da ih generišu.

Upotreba zvuka

Kada je reč o upotrebi zvuka unutar korisničkog interfejsa, treba voditi računa o sledećem:

- Multimedijalni korisnički interfejsi ili grafički korisnički interfejsi mogu da sadrže elemente zvuka.
- Zvuci mogu biti pozadinska muzika, specijalni efekti za dugmad, efekti koji su sinhronizovani sa primenom animacijom itd.
- Uvek bi trebalo "ugraditi" dugme koje će omogućiti korisniku da "isključi" zvuk, jer ne vole svi prateće zvučne efekte.

9.2. Proizvodnja multimedijalnog projekta

Tokom razvoja ili faze proizvodnje, plan projekta postaje sistematsko uputstvo za upotrebu za izgradnju ili realizaciju projekta. Istovremeno, proizvodnja (produkcija) zahteva dobru organizaciju i detaljan nadzor upravljanja tokom celog procesa izgradnje ili realizacije definisanog plana ili projekta.

Proizvodnja je faza kada se multimedijalni projekat zapravo realizuje. Tokom ove faze autor će stalno morati da vodi računa o značajnim i stalnim organizacionim zadacima. Potreban je dobar sistem koji "vodi računa" koliko je vremena utrošeno za svaki element unutar projekta. Isti sistem je "dužan" da vodi evidenciju o vremenu provedenom na pojedinim zadacima.

9.2.1. Početak

Pre nego što autor počne sa radom na sopstvenom multimedijalnom projektu, važno je da proverí svoj razvojni hardver i softver i da pregleda i preispita organizaciono i administrativno podešavanje, čak i ako radi sâm. Ovo je ozbiljan zadatak u poslednjem trenutku.

Potencijalni problemi mogu da se izbegnu tako što bi trebalo odgovoriti na sledeća pitanja:

- Da li postoji dovoljno prostora za sve datoteke za skladištenje na disku?
- Da li su na raspolaganju stručni ljudi za sve faze projekta?
- Da li postoji sistem za pravljenje rezervnih kopija kritičnih datoteka?
- Da li su finansijski aranžmani obezbeđeni i bezbedni?
- Da li je komunikacija "otvorena" prema klijentima?

9.2.2. Rad sa klijentima

Izrada multimedijalnih projekata za klijente je poseban slučaj. Autor treba da bude siguran da njegov projekat sadrži sistem za dobru komunikaciju između njega i klijenta, kao i među ljudima koji učestvuju u izradi projekta. Mnogi projekti su se završili neslavno zbog loše

komunikacije između autora i klijenta. Da bi komunikacija sa strankama ili klijentima bila na dobrom nivou, treba obratiti pažnju na sledeće stvari:

- Potrebno je uspostaviti sistem za dobru komunikaciju između klijenta i ljudi koji, zapravo, učestvuju u projektu.
- Potrebno je kontrolisati proces pregleda od strane klijenta, kako bi se izbegle beskrajne povratne petlje.
- Potrebno je razviti šemu koja određuje broj i trajanje ciklusa odobrenja klijenta.
- Potrebno je obezbediti mehanizam za menjanje naloga, kada su određene promene ili izmene već odobrene i potpisane (verifikovane).

Pored ovoga, treba obratiti pažnju na medije za čuvanje i transport (dostavu) svih elemenata multimedijalnog projekta:

- Ovo je neophodno da bi klijent lako mogao da pregleda i preispita rad.
- Potrebno je usaglasiti medije, koji će biti korišćeni, i sistem za prenos podataka (moraju da odgovaraju jedan drugom).
- Potrebno je obezbediti visoki propusni opseg interneta, kako bi veza funkcionisala kako treba.
- Najisplativije i najbrže je koristiti CD ili DVD kao metode transporta urađenog posla, a u poslednje vreme je dobra priča "vezana" za *flash* memorije.

9.2.3. Praćenje i nadzor

Autor treba da organizuje način za praćenje prijema materijala koji će biti ugrađen u multimedijalni projekat. Čak i u malim projektima, biće mnogo pojedinačnih elemenata i o njima treba voditi računa:

- Potrebno je organizovati metod za praćenje dostavljenog materijala koji će se "ugraditi" u projekat.
- Potrebno je razviti konvenciju imenovanja datoteka, koje su specifične za usvojenu strukturu projekta.
- Potrebno je čuvati (skladištiti) fajlove u direktorijumima ili folderima sa logičkim imenima.
- Ako skladišni prostor dozvoljava, trebalo bi arhivirati sve iteracije (verzije) datoteka.
- Potrebno je, možda, razviti identifikacioni sistem datoteka koji koristi DOS konvenciju imenovanja fajlova (8 karaktera + dodatak od 3 karaktera u ekstenziji) koji je zgodan za adresiranje unutar definisane platforme.

9.2.4. Autorska prava

Najčešće korišćene *authoring* platforme mogu dozvoliti pristup softverskom programskom kodu ili skriptama koji pokreću određeni projekat. Izvorni kôd HTML stranice na vebu može se lako posmatrati. Kada su autorska prava u pitanju preporučeno je:

- Kontrolisanje verzija fajlova unutar projekta je veoma važno, posebno kada su veliki projekti u pitanju.
- Ako je moguće i ako je dovoljan raspoloživi skladišni prostor, trebalo bi arhivirati sve iteracije (verzije) datoteka.
- Potrebno je ubaciti izjavu o autorstvu unutar projekta, koja legalno označava kôd kao intelektualnu svojinu tvorca ili autora.
- Izjave o autorskim pravima i vlasništvu su ugrađeni u <meta> tagovima na vrhu HTML stranice.

9.3. Tim za izradu multimedijalnog projekta

Potreban je tim iskusnih pojedinaca kako bi se stvorio dobar multimedijalni projekat. **Kreiranje tima** ili **formiranje tima** se odnosi na aktivnosti koje pomažu grupi i njenim članovima da funkcionišu na optimalnom nivou. Raznovrsna ponuda veština potrebnih za projekat se zove **multimedijalni skup veština** (*multimedia skillset*). U ovom delu biće nabrojani samo "najbitniji" članovi multimedijalnog projekta i biće nabrojane karakteristike svakog od njih.

9.3.1. Rukovodilac ili menadžer projekta

Rukovodilac projekta je odgovoran za:

- ukupan razvoj projekta, implementaciju i definisanje rada na projektu na nivou dnevnih zaduženja;
- projektovanje i upravljanje projektom;
- razumevanje prednosti i ograničenja i hardvera i softvera;
- obezbeđenje ljudstva i za njihovo organizovanje;
- prenošenja informacija između tima i klijenta.

9.3.2. Dizajner multimedije

Pod nazivom dizajner multimedije se "krije" tim koji se sastoji od:

- **grafickih dizajnera, ilustratora, animatora i specijalista za obradu slika** koji se bave vizuelnim, čime se doprinosi da je projekat privlačan i estetski ugodan;

- **dizajnera instrukcija**, koji obezbeđuju i potvrđuju da su prikazani elementi "kristalno" jasni za ciljnu grupu;
- **dizajnera interfejsa**, koji smišljaju i navigacione puteve i sadržajne mape;
- **dizajnera informacija**, koji kreiraju strukturu sadržaja, određuju "puteve" kojima će korisnik "ići" da bi došao do potrebnih informacija, određuju povratne informacije i biraju najpogodniji medij za prezentovanje multimedijalnog projekta na kome se radi.

9.3.3. Dizajner interfejsa

Dizajner interfejsa je odgovoran za:

- kreiranje softverskog uređaja koji organizuje sadržaj, omogućava korisnicima da pristupe ili izmene sadržaj i prikazuje taj sadržaj na ekranu ili monitoru;
- kreiranje "prijateljskog" korisničkog interfejsa (*user-friendly interface*), koji je prepoznatljiv većini korisnika i koji će se osećati prijatno kada ga budu koristili.

9.3.4. Pisac za multimediju

Pisac za multimediju je odgovoran za:

- stvaranje likova, akcije, tačke gledišta i interaktivnost;
- pisanje predloga teksta i njegovo testiranje na ekranu ili monitoru;
- pisanje teksta za voditelja i glumce, kao i vođenje računa o naraciji i naglascima.

9.3.5. Stručnjak ili ekspert za video

Stručnjak za video je odgovoran za:

- isporuku video fajlova za DVD ili za veb;
- kvalitet video materijala;
- prenos originalnog video snimka na računar;
- uređenje (editovanje) snimka do krajnjeg proizvoda, korišćenjem digitalnog nelinearnog sistema za montiranje (*Non-Linear Editing System*, NLE).

9.3.6. Stručnjak ili ekspert za audio

Stručnjak za audio je odgovoran za:

- pronalaženje i izbor pogodnih muzičkih talenata;
- zakazivanje snimanja;
- digitalizaciju i uređivanje snimljenog materijala u računarske fajlove.

9.3.7. Programer multimedije

Programer multimedije (često se naziva i softverski inženjer) je odgovoran za:

- integraciju svih multimedijalnih elemenata u jednostavan projekat, koristeći *Authoring* sisteme ili neki programski jezik;
- pisanje kodova za prikaz multimedijalnih elemenata i za kontrolu različitih perifernih uređaja;
- upravljanje tajmingom, prelazima i čuvanjem snimljenog materijala.

9.3.8. Producent multimedije za veb

Producent za veb je odgovoran za:

- postavljanje koordiniranog seta (sadržaja) na odgovarajući sajt ili odgovarajuću veb stranicu;
- koordiniranje ažuriranja i adekvatnih ili potrebnih (traženih) promena.

9.3.9. Programer

Programer je odgovoran za:

- povezivanje odgovarajućih elemenata, ako je to bilo nemoguće obaviti na neki drugi način;
- optimizaciju rada, vizualizacije, ažuriranja i eventualnih promena;
- konstantno praćenje rada veb sajta ili veb stranice gde se nalazi multimedijalni materijal.

Glava 10

HTML i HTML5

Jezik HTML (*HyperText Markup Language*) je standardizovani jezik koji se koristi pri strukturiranju tekstova, medija i ugrađenih objekata u veb stranice i elektronsku poštu.

Jezik HTML je sačinjen od običnog teksta i tagova. Iako se HTML znakovi (*tags*) često nazivaju kodom, tehnički HTML nije kôd, jer računarski kodovi su instrukcije koje od računara traže da izvrši određenu operaciju. Svrha HTML tagova je da se struktura dokumenta "označi" tako da bi korisnikov agent (*user agent*), tj. internet pretraživač mogao da prepozna strukturu dokumenta i ispravno je prikaže u prozoru pretraživača kojeg korisnik koristi.

Jezik HTML koristi veoma "labavu" sintaksu, ali i dalje postoji nekoliko stvari o kojima bi trebalo voditi računa i kojih korisnik mora da bude svestan:

- HTML nije "osetljiv" na veličinu slova. Za elemente, attribute i vrednosti atributa mogu da se koriste proizvoljne veličine karaktera. Mogu da se koriste i kombinuju veličine (mala i velika slova), ali trebalo bi biti dosledan kako bi kôd bio čitljiviji.
- HTML je "neosetljiv" po pitanju belina. U okviru tagova, element mora odmah da prati otvorenu zagradu. Nakon toga, međutim, može da se ostavi "prazan" prostor ili "belinu" ukoliko korisnik hoće, između elementa i atributa, između atributa, a pre zatvaranja zagrade. Izvan tagova, korisnik može da koristi beline po nahođenju, jer nemaju uticaja na konačni izgled ekrana unutar aktivnog pretraživača.
- Kada se koriste atributi, vrednosti atributa mogu da se prilože između jednostrukih ili dvostrukih navodnika. Neki vrednosti atributa ne moraju da imaju navodnike, ali je lakše uvek ih koristiti, nego da se razmišlja kada su neophodni, a kada ne.

Za dodatno modifikovanje veb stranica postoje:

- CSS ili *Cascading Style Sheets* za izgled i položaj prezentacije;
- skriptni jezici (*JavaScript*, *VBScript*) za omogućavanje dinamičnosti i interaktivnosti na veb prezentacijama;
- DOM ili *Document Object Model*, koji označava vezu između skripti i elemenata na stranici.

10.1. HTML i osnove veb strana

Mnogi ljudi stvaraju veb stranice pomoću tekst editora ili tekst procesor. U popularne tekst editore spadaju *Notepad++ for Windows* i *SimpleText for Macintosh*, a u popularne tekst procesore *Microsoft WordPad* i *Microsoft Word*.

Da bi se podesila veb stranica, trebalo bi, najpre, uneti tekst koji će se pojaviti na pomenutoj stranici. Preporuka je da se tekst ne formatira koristeći tekst editor ili tekst procesor. Moraju da se dodaju HTML oznake (tagovi) kako bi se promenio izgled teksta. HTML tagovi pružaju uputstva za formatiranje i struktuiranje veb stranice.

Nakon unošenja teksta za veb stranicu, trebalo bi sačuvati istu kao tekstualni dokument i navesti ime za veb stranicu sa ekstenzijom `.html` ili `.htm`. Ime veb stranica može da sadrži slova i brojeve, ali bez razmaka. Glavna veb stranica ili početna stranica, na veb sajtu, se obično zove `index.html`.

Postoje neki osnovni HTML tagovi koje bi trebalo dodati svakoj veb stranici koja se kreira. Tagovi se pišu u okviru oznaka `< i >`. Većina tagova ima početni i završni tag. Završni tag se dobija dodavanjem znaka `/` i označava mesto na kome prestaje dejstvo početnog taga. HTML tagovi ne zavise od veličine slova kojim se pišu; mogu se da se pišu malim ili velikim slovima (*case insensitive*).

Oznaka ili tag `<HTML>` treba da se pojavi na vrhu svake veb stranice koja se kreira. Ovaj tag identifikuje dokument kao veb stranicu koja sadrži HTML kôd.

Oznaka `<HEAD>` treba da se pojavi neposredno ispod oznake `<HTML>`. Oznaka identifikuje deo glave veb stranice, koji sadrži informacije o veb stranici, kao što je naslov.

Oznaka `<TITLE>` se pojavljuje u odeljku glave veb stranice. Ovaj tag omogućava da se veb stranici dodelu naslov. Svaka veb stranica koja se pravi mora da ima naslov. Naslovi se obično pojavljuju u naslovnoj traci prozora veb pretraživača. Naslov veb stranica trebalo bi da sadrži samo slova i brojeve.

Oznaka `<BODY>` identifikuje telo veb stranice. U ovaj odeljak su uključeni sadržaj veb stranice, tekst, slike, tabele i drugi elementi koje korisnici žele da vide kada posete veb stranicu.

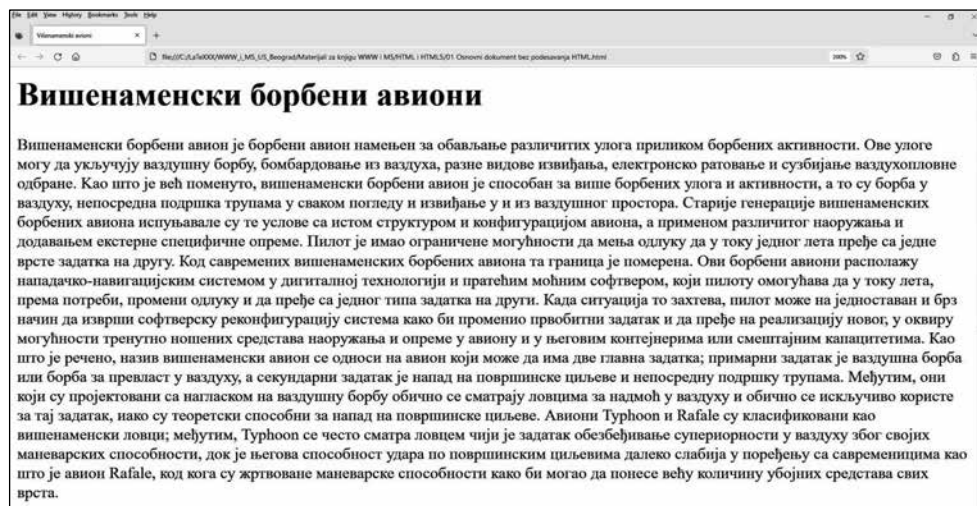
Minimalna struktura HTML dokumenta je prikazana ispod:

```
<html>
  <head>
    <meta charset="UTF-8">
    <!--nasa slova - KOMENTAR-->
    <title>
    <!--naziv stranice - KOMENTAR-->
    </title>
  </head>
<body>
<!--Izgled stranice i tekstna stranici - KOMENTAR-->
</body>
</html>
```

Prikazivanjem veb stranice u veb pretraživaču omogućeno je da se vidi kako će se pojaviti stranica na internetu. Potreban je veb pretraživač, kao što su, *Mozilla Firefox*, *Google Chrome*, *Microsoft Edge*, *Apple Safari*, *Opera*.

Veb pretraživač čita HTML kodove koji su korišćeni za kreiranje veb stranice i zatim prikazuje veb stranicu u prozoru veb pretraživača. Kako veb pretraživač čita svaki deo HTML koda, tako dobijene informacije prikazuje na ekranu. To znači da veb pretraživač može da prikaže deo veb stranice pre nego što je u potpunosti završena obrada stranice.

Kao primer svega rečenog, na slici 10.1 je prikazan sadržaj dokumenta sa primenjenim osnovnim tagom za naslov, a ostali tagovi nisu korišćeni.



Slika 10.1. *Tekst bez definisanih tagova, sem naslova*¹

Trebalo bi prikazati napravljenu veb stranicu u nekoliko veb pretraživačima, da bi se videlo kako će svaki pretraživač da prikaže veb stranicu. Svaki veb pretraživač može da tumači drugačije HTML oznake i neki veb pretraživači ne podržavaju sve HTML funkcije. Na primer, najnovije verzije veb pretraživača mogu da prepoznaju nove HTML funkcije, dok stariji veb pretraživači ne mogu pravilno da prikažu te funkcije.

Veb pretraživači obično ignorišu pritisak na taster Enter prilikom kucanja u tekst editoru ili tekst procesoru. Korisnik mora da koristite oznaku ili tag <P> da bi odredio gde želi da svaki novi **paragraf** ili **pasus** počne. Sintaksa ovog elementa je:

```
<p> Paragraf </p>
```

Kada se prikazuje veb stranica u veb pretraživaču, između svakog paragrafa će se pojaviti prazan red, koji je definisan upotrebom taga <P>. Ovo može da poboljša izgled i čitljivost veb stranice na osnovu "razbijanja" velikih delova teksta (slika 10.2).

Po pravilu, većina veb pretraživača poravnava tekst po levoj margini. Atribut ALIGN omogućava da se promeni poravnanje paragrafa. Paragrafi ili pasusi mogu da budu poravnati

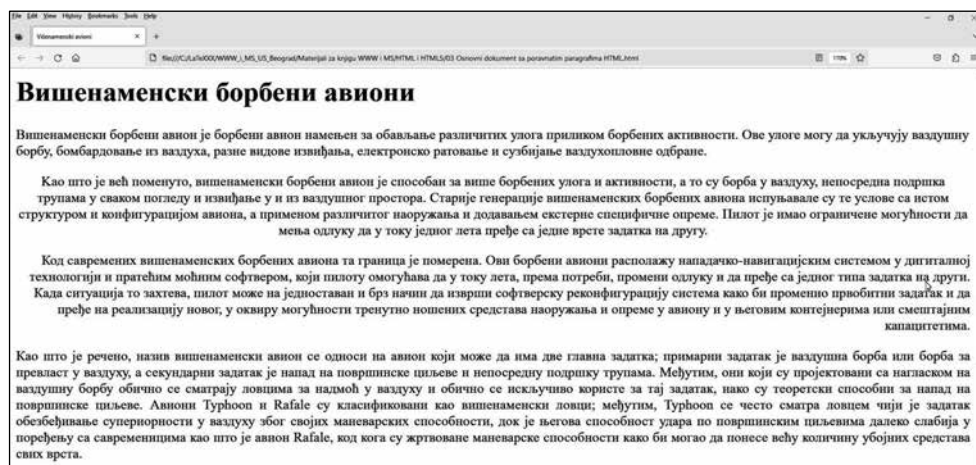
¹ 01 Osnovni dokument bez podesavanja HTML.html

po levoj margini (*left*), po desnoj margini (*right*), po obe margine(*justify*), a mogu da budu i centrirani (*center*).



Slika 10.2. Osnovni dokument sa tagovima za paragrafe²

Promena poravnanje važnog paragrafa može da ga izdvoji iz ostatka veb stranice. Na slici 10.3 su prikazana sva četiri poravnanja unutar istog dokumenta, gde svaki pasus ima drugačije poravnanje.



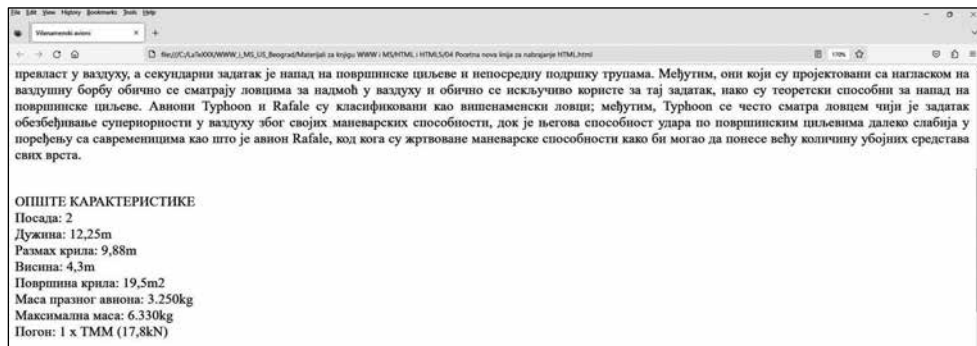
Slika 10.3. Osnovni dokument sa poravnatim paragrafima³

Tag
 određuje da veb pretraživač zaustavi postavljanje teksta i drugih elemenata u trenutnu liniju i da **počne novu liniju** u prozoru veb pretraživača. Ova oznaka je korisna za odvajanje kratkih redova teksta, kao što je tekst adrese ili pesme. Za razliku od mnogih HTML tagova, tag
 nema krajnji tag.

Tag
 je, takođe, koristan za povećanje količine prostora između elemenata na veb stranici. Na primer, može da se koristi ovaj tag da bi se dodala još jedna prazna linija između dva pasusa. Svaki tag
 će stvoriti dodatnu liniju prostora (slika 10.4).

² 02 Osnovni dokument sa paragrafima HTML.html

³ 03 Osnovni dokument sa poravnatim paragrafima HTML.html



Slika 10.4. Kreiranje novih linija za provizorno nabiranje⁴

Veb pretraživači obično ignorišu neke dodatne prostore kada se unosi tekst za veb stranicu. Ako korisnik želi da uključi dodatne prazne prostore na veb stranici, onda bi trebalo da koristi ` ` kôd. Ubacivanje praznih prostora je korisno za uvlačenje prve linije pasusa, kao što je prikazano na slici 10.5.



Slika 10.5. Kôd za uvlačenje prvog reda paraagrafa⁵

Razmaci su, takođe, korisni za odvajanje elemenata na veb stranici, kao što su slike. Na primer, ubacivanje praznog prostora između dve slike koje stoje jedan uz drugu, može da spreči da se slike pojavljuju kao jedna velika slika. Kôd ` ` stvara prostor bez prekida. Kucanjem ` ` umesto da se kuca razmak između dve reči će sprečiti da reči budu odvojene kada veb pretraživač završava tekst i pokušava da ga uklopi u prozor. Ovo osigurava da će se dve reči uvek pojaviti u istoj liniji.

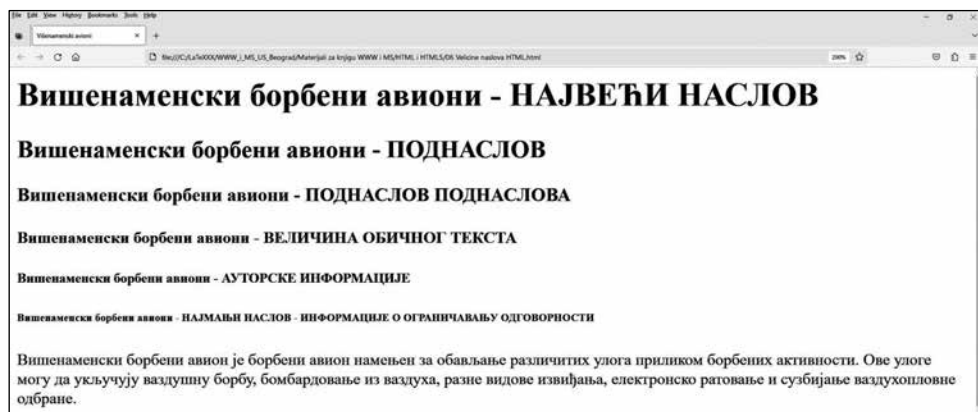
Naslovi (*headings*) mogu da pomognu da se organizuju informacije na veb stranici. Korisnici mogu da pregledom naslova na veb stranici brzo pronađu informacije od interesa.

Postoji **šest nivoa naslova** koji mogu da se koriste. Nivo 1 (`<H1>`) je najveći, a nivo 6 (`<H6>`) je najmanji (slika 10.6). Nivoi `<H1>`, `<H2>` i `<H3>` se često koriste za naslove veb

⁴ 04 Pocetna nova linija za nabiranje HTML.html

⁵ 05 Uvlacenje prvog reda paraagrafa HTML.html

stranica i naslove poglavlja, dok su nivoi <H5> i <H6> korisni za informacije o ograničavanju odgovornosti, kao i za autorske informacije. Nivo <H4> je obično iste veličine kao i glavni tekst na veb stranici. Korišćenje naslova u 6 nivoa će pomoći da korisnici razumeju strukturu veb stranice. Da bi se izbegla konfuzija, ne bi trebalo koristiti više od tri različita nivoa naslova na jednoj veb stranici.



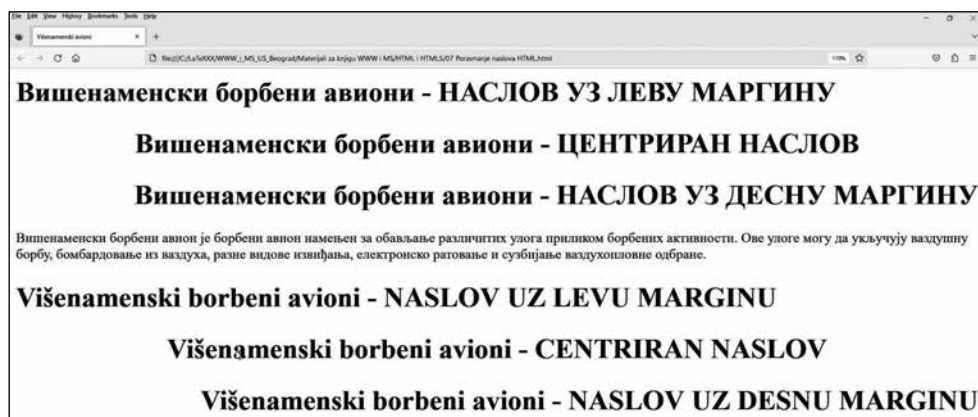
Slika 10.6. *Uporedno prikazivanje veličina naslova*⁶

Opšti oblik taga za naslov (slika 10.6) je:

<Hn> Naslov na nivou n </Hn>

gde n predstavlja nivo na kome se naslov nalazi.

Veb pretraživači, obično, prikazuju naslove masnim (**bold**) tekstom sa praznom linijom iznad i ispod naslova. U većini veb pretraživača, uobičajeno je da su naslovi poravnati po levoj margini. Atribut ALIGN omogućava da se promeni poravnanje naslova (slika 10.7).



Slika 10.7. *Poravnavanje naslova*⁷

⁶ 06 Velicine naslova HTML.html

⁷ 07 Poravnanje naslova HTML.html

Opšti oblik taga za naslove (slika 10.7) je:

```
<h1 align="left">NASLOV UZ LEVU MARGINU</h1>
<h1 align="center">CENTRIRAN NASLOV</h1>
<h1 align="right">NASLOV UZ DESNU MARGINU</h1>
```

gde broj predstavlja nivo na kome se naslov nalazi.

Veb pretraživač obično **ignorise blanko linije i razmake** koji se dodaju kada se unosi tekst u veb stranicu. Tag <PRE> nalaže veb pretraživaču da zadrži razmak koji se ukucava unutar tekst editora ili teskt procesora.

Veb pretraživači prikazuju preformatirani tekst u *monospace* fontu, kao što je *Courier*. Ova vrsta fonta je takva da svaki znak zauzima istu količinu prostora. Kada se preformatira tekst, veb pretraživač će prikazati tekst sa tačnim razmacima koji su uneti i neće "razvlačiti" tekst kako bi se na "silu" popunio prozor unutar veb pretraživača. Preformatirani tekst je koristan za kreiranje jednostavnih tabela.

Dodavanje komentara na veb stranici je korisno kada autor želi da se podseti da ažurira deo teksta ili da pokaže zašto je koristio određenu oznaku. Komentar mora da bude okružen oznakama <!-- i -->. Veb pretraživači će ignorisati informacije između ovih oznaka i sprečavaće komentare da se pojavljuju na veb stranici. Trebalo bi imati na umu da će korisnici koji gledaju HTML kôd moći da čitaju komentare "iza" veb stranice. Evo primera određenog koda:

```
<pre>
Posada: 2 <!-- Ovo je komentar!!!! -->
Duzina: 12,25m
Razmah krila: 9,88m
Visina: 4,3m <!-- Da li je dobra visina? -->
Povrsina krila: 19,5m2
</pre>
```

HTML standard omogućava kodovanje znakova za dodavanje specijalnih karaktera i/ili znakova na veb stranicu (slika 10.8). Ovo omogućava da se uključuju znakovi koji se ne pojavljuju na tastaturi, kao što su matematički simboli.



Slika 10.8. Specijalni karakteri⁸

⁸ 08 Dodavanje specifičnih znakova HTML.html

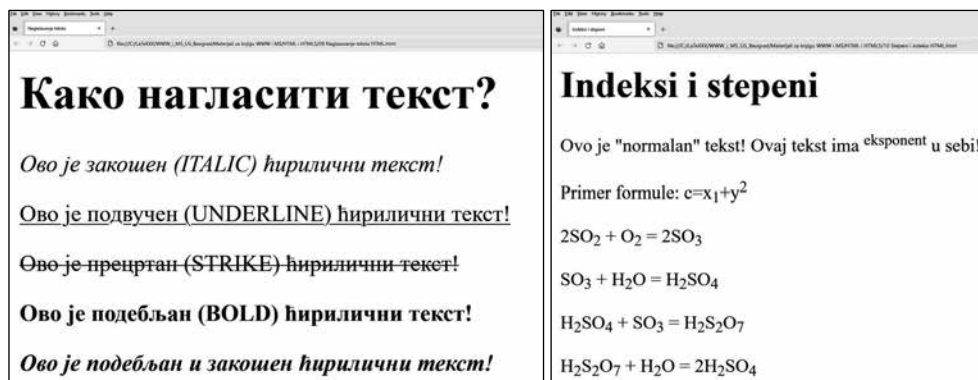
Kada se dodaje poseban karakter, za njega bi trebalo navesti broj koda ili ime koda. Broj koda ili brojčani kôd se sastoji od zagrada, broja znaka, broja karaktera i zareza. Ime koda se sastoji od zagrada, naziva karaktera i zareza. Kodovanje znakova je korisno za dodavanje znakova koji se koriste za izradu veb stranica, kao što su <, >, " i &, jer veb pretraživači mogu pogrešno protumačiti ove likove. Na primer, veb pretraživači mogu da protumače znak < kao početak taga.

Trebalo bi imati na umu da će pojava specijalnih znakova, dodatih na veb stranicu, zavisiti od konfiguracije veb pretraživača korisnika i instaliranih fontova na korisnikovom računaru.

10.2. HTML i tekst

Sledi kratka priča o tekstu, njegovom uređenju, biranju i definisanju određenih fontova, kao i odgovarajućih lista za nabranjanje unutar željene veb stranice.

Da bi se **naglasio tekst** na veb stranici, korisnik će možda da ga podeblja (*Bold*), zakosi (*Italic*), podvuče (*Underline*) ili "precrti" (*Strike*); prikazano na slici 10.9, levo.



Slika 10.9. Specifično naglašavanje teksta⁹ i indeksi i stepeni¹⁰

Oznaka omogućava ispisivanje "masnih" ili podebljanih slova, dok oznaka <I> omogućava ispisivanje zakošenog teksta (kurzivom). Podebljani i zakošeni tekst su korisni za uvođenje novih termina i isticanje važnih fraza na veb stranici.

Oznaka <U> podvlači tekst. Trebalo bi biti obazrivi prilikom podvlačenja teksta, jer korisnici mogu ovakav tekst da zamene za link. Oznaka <STRIKE> "postavlja" liniju kroz tekst (precrtava tekst). Ova oznaka je korisna za prikazivanje promena u informacijama na veb stranici.

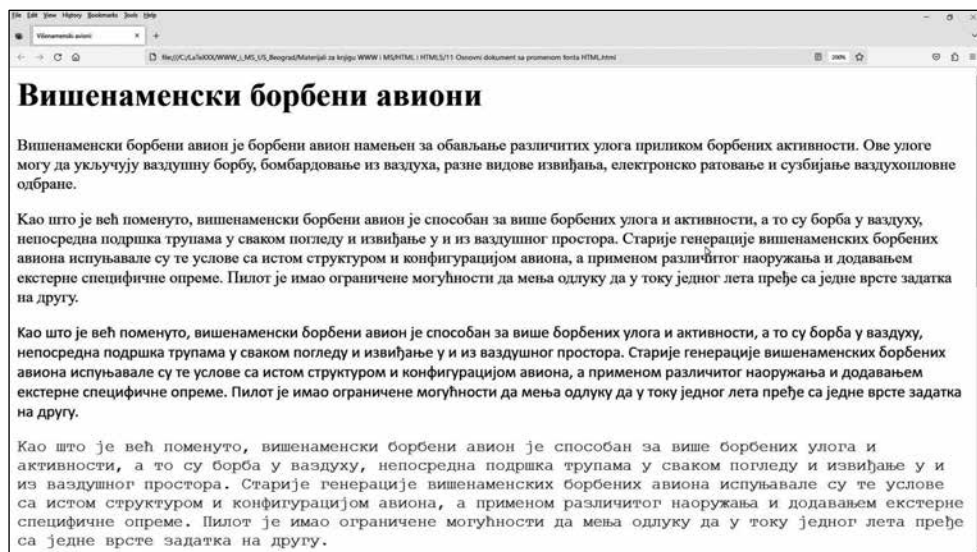
Oznaka <SUP> omogućava da se kreira **stepen** (eksponent) na veb stranici. Tekst stepena ili eksponenta se pojavljuje nešto iznad glavnog teksta na veb stranici. Oznaka <SUB> omogućava da se kreira **indeks** na veb stranici. Tekst indeksa se pojavljuje nešto niže od glavnog teksta na veb stranici (slika 10.9, desno).

⁹ 09 Naglasavanje teksta HTML.html

¹⁰ 10 Stepeni i indeksi HTML.html

EkspONENTI i INDEKSI su idealni za prikazivanje matematičkih jednačina, hemijskih formula, naučnih zapisa i fusnota. veb pretraživači mogu da prikazuju tekst indeksa i eksponenta u manjoj veličini fonta, nego što je veličina glavnog teksta na veb stranici. Preporuka je da se koristi dovoljna veličina glavnog fonta, kako bi se pravilno prikazali i videli i indeksi i eksponenti.

Korišćenjem oznake sa atributom <FACE> omogućena je **promena fonta** za jedan deo teksta na veb stranici. Može da se odredi novi font po imenu, kao što je *Courier*, ili prema vrsti, kao što je *monospace*. Ako se određuje font po imenu, onda bi trebalo da se navede više od jednog fonta u slučaju da prvi izbor nije dostupan na korisnikovom računaru. Jedan od fontova, koje bi trebalo navesti, je "zajednički font" za sve računare i operativne sisteme, kao što je *Arial*, kako bi se povećala verovatnoća da će računar prikazati jedan od izabranih fontova (slika 10.10).



Slika 10.10. Promena fontova po paragrafima¹¹

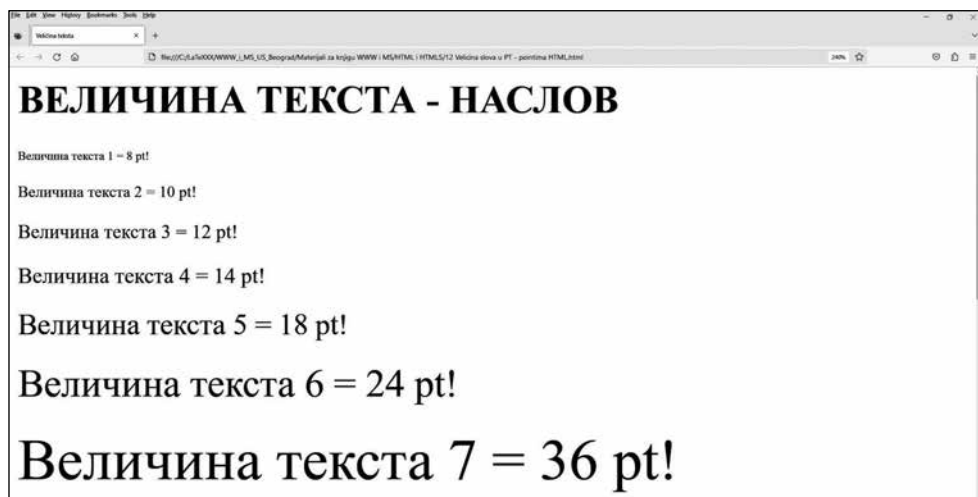
Iako ne postoji ograničenje po pitanju broja fontova koliko veb stranica može da podrži, korišćenje više fontova može da odvuče pažnju korisnika.

Atribut <SIZE> omogućava da se promeni veličina teksta na veb stranici. Povećanje veličine teksta čini tekst lakšim za čitanje, a smanjenje veličine teksta omogućava da stane više informacija na ekranu (slika 10.11).

Upotreba atributa <SIZE> sa oznakom (tagom) <BASEFONT> omogućava da se promeni veličina celokupnog teksta na veb stranici. Korišćenjem atributa <SIZE> sa oznakom omogućava da se promeni veličina jednog dela teksta. Postoji 7 veličina slova koje mogu da se koriste. Najmanja veličina fonta je 1, a najveća veličina je 7. Trebalo bi imati na umu da korišćena veličina slova mogu da se ne pojave onako kako korisnik

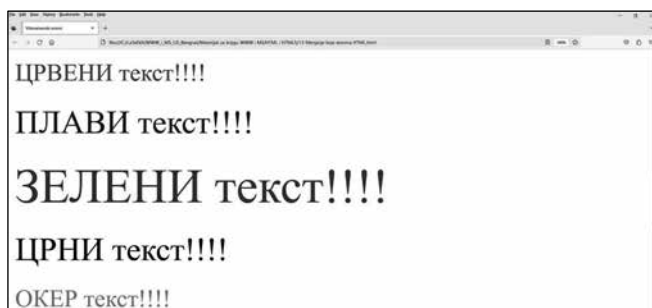
¹¹ 11 Osnovni dokument sa promenom fonta HTML.html

očekuje na nekim računarima, budući da neki autori mogu podese svoje veb pretraživače da prikazuju veličinu fonta kako im odgovara.



Slika 10.11. *Promena veličine slova u tačkama ili pointima*¹²

Menjanje boje celokupnog teksta ili nekog dela unutar veb stranice može da doprinese vizuelnom interesu raznih korisnika za veb stranicu. Korišćenjem atributa <TEXT> sa tagom <BODY> omogućeno je da se promeni boja kompletnog teksta na veb stranici. Korišćenjem atributa <COLOR> sa tagom omogućeno je da se promeni boja dela teksta, kao što je prikazano na slici 10.12.



Slika 10.12. *Promena boje slovima*¹³

Kada se menja boja teksta, mora da se navede ime ili heksadecimalna vrednost boje koju korisnik želi da koristi. Heksadecimalna vrednost je kôd koji govori veb pretraživaču koju boju bi trebalo prikazati. Kôd se sastoji od znaka (#), a iza nje slede RGB (*red*, *green*, *blue*) komponente boje. Postoji samo 16 boja koje mogu da se izaberu po imenu: *aqua* (#00FFFF) tirkizno plava; *black* (#000000) crna; *blue* (#0000FF) plava;

¹² 12 Velicina slova u PT - pointima HTML.html

¹³ 13 Menjanje boje slovima HTML.html

fuchsia (#FF00FF) tirkizno roze; *gray* (#808080) siva; *green* (#008000) tamno zelena; *lime* (#00FF00) tirkizno zelena; *maroon* (#800000) bordo/braon; *navy* (#000080) marin plava; *olive* (#808000) maslinasto zelena; *purple* (#800080) ljubičasta; *red* (#FF0000) crvena; *silver* (#C0C0C0) srebrna; *teal* (#008080) plavo/zelena; *white* (#FFFFFF) bela; *yellow* (#FFFF00) žuta.

Atribut <BGCOLOR> omogućava da se promeni boja pozadine veb stranice. Kada se menja boja pozadine, mora da se navede ime ili heksadecimalna vrednost boje koju autor želi da koristi (slika 10.13).



Slika 10.13. *Promena boje pozadine veb stranice*¹⁴

Trebalo bi odabrati boju pozadine koja "radi" dobro sa bojom teksta. Na primer, crveni tekst na plavoj pozadini može biti težak za čitanje.

Promena margina unutar veb stranice omogućava autoru da podesi veličinu prostora koji se pojavljuje između sadržaja stranice i ivica prozora veb pretraživača (slika 10.14). Ovo može da poboljša izgled veb stranice.



Slika 10.14. *Podešavanje i promena margina*¹⁵

¹⁴ 14 Promena boje pozadine veb stranice HTML.html

¹⁵ 15 Promena margina HTML.html

Po pravilu, većina veb pretraživača prikazuje veb stranicu sa marginama čija je vrednost oko 10 piksela. Atributi LEFTMARGIN, RIGHTMARGIN, TOPMARGIN i BOTTOMMARGIN omogućavaju autoru da promeni margine koje se pojavljuju kada se veb stranica. Ovi atributi nisu deo HTML standarda, ali su podržani od strane većine veb pretraživača.

Korišćenjem oznaka (tagova) <TT>, <CODE>, <KBD> ili <SAMP> omogućen je prikaz teksta na veb stranici u *monospace* fontu. Reč je o vrsti fonta, kao što je *Courier*, gde **svaki karakter zauzima istu veličinu prostora** (slika 10.15).



Slika 10.15. Fontovi gde svaki karakter zauzima istu veličinu prostora¹⁶

Oznaka <TT> je najčešće korišćena oznaka za prikazivanje teksta u monospace fontu. Ova oznaka je korisna za vizuelno odvajanje reči ili fraza koje korisnik želi da naglasi, ali koje nemaju određenu svrhu, kao što su uputstva.

Ako tekst ima specifičnu namenu, onda bi trebalo koristiti tagove <CODE>, <KBD> ili <SAMP> kako bi se prikazao tekst u *monospace* fontu. Oznaka <CODE> se obično koristi za tekst koji predstavlja računarski kôd. Oznaka <KBD> je korisna za isticanje teksta koje bi trebalo da korisnici kucaju. Oznaku <SAMP> bi trebalo koristiti za prikazivanje teksta primera ili rezultata, kao što je izlaz iz računarskog programa. Uobičajen *monospace* font je *Courier New*.

Kod nekih veb pretraživača moguće je primeniti dodatno formatiranje teksta koji koristi <CODE>, <KBD> ili <SAMP> tagove. Na primer, veb pretraživači mogu da podebljaju (*bold*) tekst koji koristi <KBD> oznaku. Oznaka <BLOCKQUOTE> omogućava da se kreira **tekstualni blok** na veb stranici. Tekstualni blok je deo teksta koji je odvojen od ostatka teksta na veb stranici (slika 10.16).

Tekstualni blokovi se, obično, pojavljuju uvučeni sa obe strane veb stranice i često se koriste za prikazivanje dugih ili važnih citata. Ne postoji ograničenje u iznosu teksta koji može biti uključen u tekstualni blok. Ako je tekstualni blok veoma dug, onda bi trebalo

¹⁶ 16 Monospace fontovi HTML.html

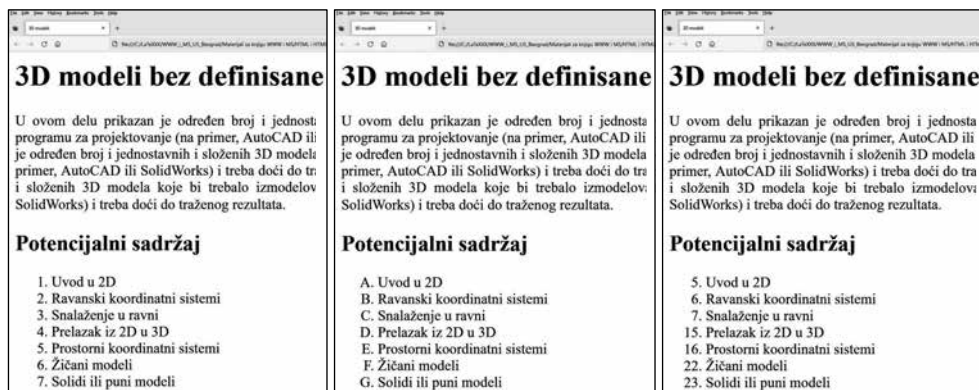
razmisлити o korišćenju taga <P> kako bi se "razbio" tekst u tekstualnom bloku na paragrafe ili pasuse.



Slika 10.16. *Formiranje i naglašavanje tekstualnog bloka*¹⁷

Numerisana lista je korisna za prikazivanje stavki koje se nalaze u određenom redosledu, kao što je set instrukcija ili sadržaj.

Kada se pravi numerisana ili uređena lista, postoje dve glavne oznake koje moraju da se koriste. Oznaka označava početak uređene liste i oznaka označava početak svake stavke u listi. Iako ne postoji ograničenje po pitanju veličine teksta stavki u listi, preporuka je da se svaka stavka ograniči na jedan ili dva reda teksta. Ovo će poboljšati čitljivost spiska.



Slika 10.17. *Numerisane liste*^{18 19 20}

¹⁷ 17 Formiranje i naglašavanje tekstualnog bloka HTML.html

¹⁸ 18 Brojane numerisane liste HTML.html

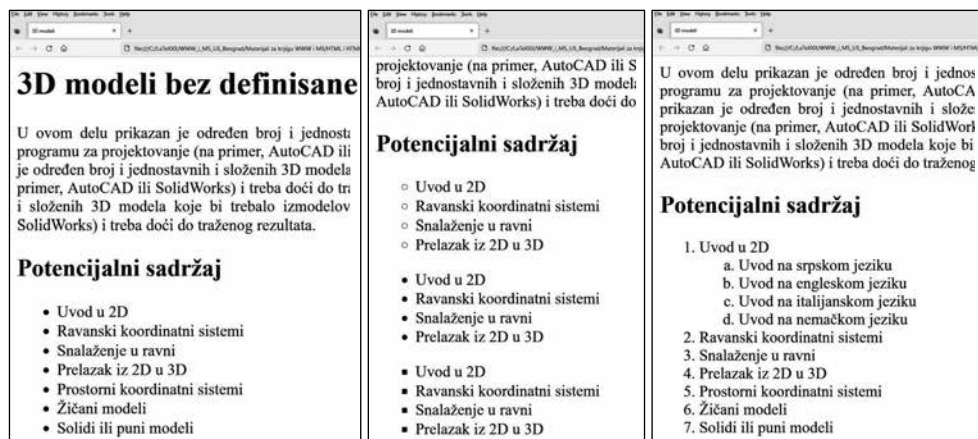
¹⁹ 19 Složno označene liste HTML.html

²⁰ 20 Proizvoljno brojane označene liste HTML.html

Uobičajeno je da se brojevi (1, 2, 3, ...) pojavljuju pored svake stavke u numerisanoj listi. Atribut <TYPE> omogućava da se promeni način obeležavanja stavki u listi. Nakon kreiranja uređene liste, korisnik može da doda novu stavku u listu u bilo kom trenutku. veb pretraživač će automatski da prenumerise stavke unutar liste.

Atributi START i VALUE omogućavaju da se prilagode "naručene" liste. Atribut START omogućava da se započne numerisanje liste brojem različitim od broja 1. Atribut VALUE omogućava da se "preskoče" dodeljeni brojevi i da se dodeli novi broj stavki u listi.

Nenumerisane ("neuređene") liste su korisne kada je potrebno prikazati stavke bez nekog posebnog reda, kao što su liste proizvoda ili veb sajtova. Kada se pravi neuređena lista, postoje dve glavne oznake koje moraju da se koriste. Oznaka označava početak jedne neuređene liste i oznaka označava početak svake stavke unutar liste.



Slika 10.18. Nenumerisane liste^{21 22 23}

Da bi se poboljšala čitljivost neuređene liste, trebalo bi pokušati da se svaka stavka ograniči na jedan ili dva reda teksta. veb pretraživači, obično, počinju da "ređaju" stavke u neuređenim listama od leve ivice prozora veb pretraživača.

Uobičajeno je da veb pretraživač obeležava svaku stavku u nenumerisanoj listi ispunjenom kružnicom. Atribut TYPE omogućava da se promeni stil obeležavanja stavki. Na raspolaganju su tri stila – ispunjena kružnica (*disc*), prazna ili neispunjena kružnica (*circle*) i ispunjen kvadrat (*square*), kao što je prikazano na slici 10.18, u sredini.

Ugnežđena lista je lista unutar liste. Ova vrsta liste omogućava autoru da pruži dodatne informacije o određenoj stavci unutar listi. Na ovaj način mogu da se kreiraju liste sa nekoliko nivoa stavki (slika 10.18, desno).

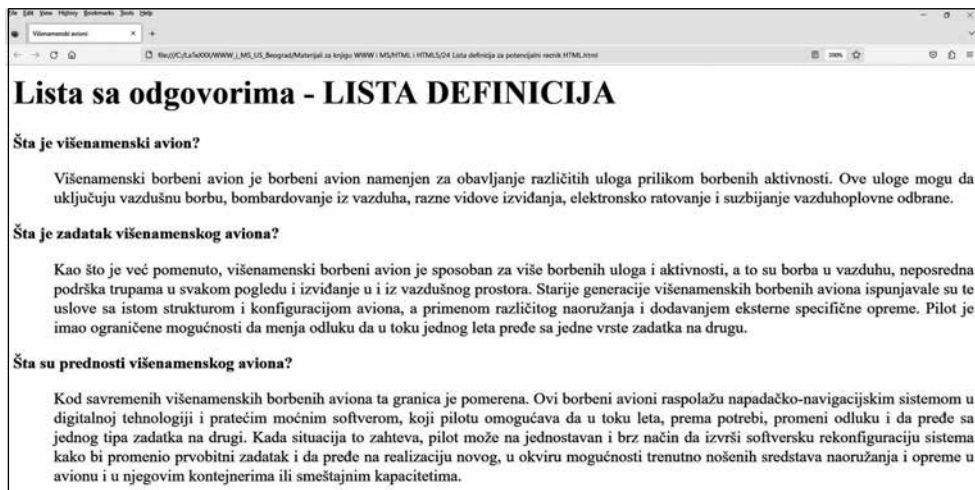
Da bi se kreirala ugnežđena lista, trebalo bi dodati novu uređenu ili neuređenu listu unutar postojeće liste na veb stranici. Ovo je korisno kada bi trebalo samo neke stavke numerisati ili ih složiti po određenom redosledu.

²¹ 21 Nenumerisane liste HTML.html

²² 22 Nenumerisane liste sa tri razlicite oznake HTML.html

²³ 23 Ugnezdjene liste unutar postojece liste HTML.html

Lista definicija (lista za rečnike) prikazuje uslove i njihove definicije. Ovaj tip liste je idealan za rečnike. Postoje tri glavne oznake koje se koriste za kreiranje liste definicije. Oznaka <DL> označava početak liste definicije, oznaka <DT> označava početak svakog termina u listi i oznaka <DD> označava početak svake definicije u listi. veb pretraživači automatski poravnava termine u listi definicije po levoj margini. Definicije se pojavljuju ispod termina i uvučene su u odnosu na termine (slika 10.19).



Slika 10.19. Lista definicija za rečnik²⁴

Unutar definicione liste ne moraju da se navode oznake <DT> i <DD> naizmenično. Ako je potrebno navesti nekoliko termina za jednu definiciju, onda može da se koristi oznaka <DT> nekoliko puta u jednom redu. Slično tome, ako je potrebno specificirati nekoliko definicija za jedan termin, onda može da se koristi oznaka <DD> nekoliko puta u redu.

10.3. HTML i slike

Sledi kratka priča o slikama ili fotografijama, njihovom smeštanju unutar veb stranice, njihovom međusobnom pozicioniranju i pozicioniranju u odnosu na druge elemente, odgovarajućim formatima i podešavanjima veličine.

Oznaka omogućava da se **doda slika** u veb stranicu. Slike predstavljaju linijske elemente, što znači da će veb pretraživači automatski da prikažu slike u okviru teksta, a ne u novoj liniji.

Ima mnogo mesta koja "nude" slike koje mogu da se koriste na veb stranicama, kao što su veb sajtovi i "skladišni prostor" na eksternim računarima. Takođe, može da koristi skener za skeniranje slike i smeštanje u računar ili može da se koristi program za uređivanje slika u cilju kreiranja sopstvenih slika. Trebalo bi voditi računa oko dozvola za korišćenje bilo koje slike koju autor nije kreirao sâm. Slike povećavaju potrebno vreme da se prikaže sadržaj

²⁴ 24 Lista definicija za potencijalni recnik HTML.html

veb stranice na monitoru. Kad god je moguće, trebalo bi koristiti slike manje rezolucije, jer se ove slike učitavaju mnogo brže. Slike koje se dodaju u veb stranice bi trebalo da budu u GIF ili JPEG formatu.



Slika 10.20. Smeštanje slika ili fotografija u dokument²⁵

Neki korisnici imaju veb pretraživače koji ne mogu da prikazuju slike, dok drugi isključuju prikaz slika radi bržeg pregledanja veb stranica. Trebalo bi koristiti atribut <ALT> kako bi se prikazao propratni tekst, ako se određena slika ne pojavi na veb stranici (slika 10.21, desno). To će dati korisnicima informacije koje slike nisu prikazane.



Slika 10.21. Ubacivanje slika i propratni tekst ako se slika ne pojavi^{26 27}

²⁵ 25 Ubacivanje fotografija u dokument HTML.html

²⁶ 26 Ubacivanje više fotografija u jedan red dokumenta HTML.html

²⁷ 27 Propratni tekst za fotografije u dokumentu HTML.html

Korišćenjem atributa `BACKGROUND` unutar oznake (taga) `<BODY>` omogućeno je da se **doda pozadinska slika** za veb stranicu. Pozadinska slika je mala slika koja se ponavlja kako bi se popunila celokupna veb stranica (slika 10.22).



Slika 10.22. *Slika ili fotografija kao pozadinski element dokumenta*²⁸

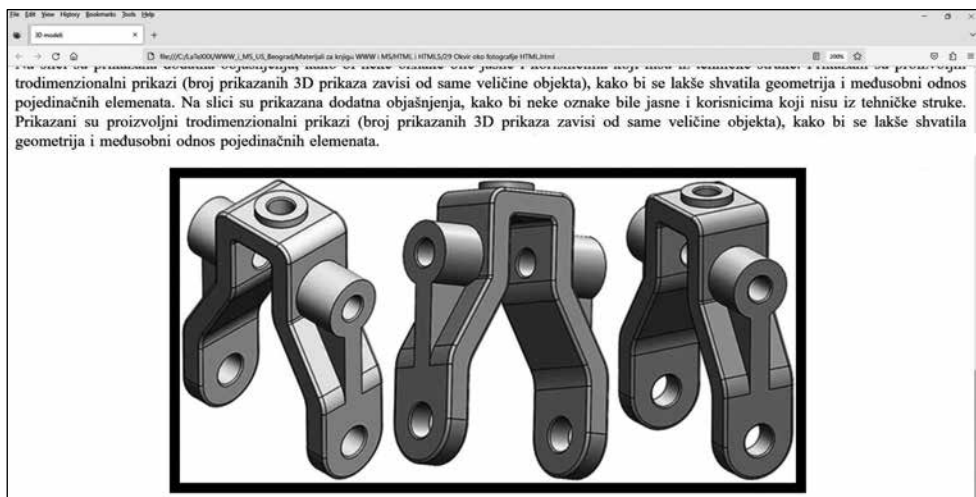
Prilikom dodavanja pozadinske slike, trebalo bi koristiti sliku koja stvara zanimljive pozadinske "efekte", bez "gušenja" sadržaja veb stranice. Dobra pozadinska slika bi trebalo da ima nevidljive ivice, tako da kada se slika ponavlja prilikom ispunjavanja veb stranice, formiraće besprekornu pozadinu.

Budući da slike u pozadini (pozadinske slike) povećavaju vreme koje je potrebno da se veb stranica pojavi na ekranu (monitoru), trebalo bi koristiti pozadinske slike male veličine. Trebalo bi se uveriti i da pozadinska slika ne utiče na čitljivost veb stranice. Može da se desi da će autor morati da promeni boju teksta kako bi se lakše čitalo unutar veb stranice sa postavljenom pozadinskom slikom.

Atribut `BORDER` omogućava da se doda **granični okvir slici** na veb stranici (slika 10.23). Granični okvir ili okvir može sliku da ističe i može da poboljša izgled veb stranice. Kada se dodaje granični okvir, mora da se navede "debljina" (širina linije) okvira u pikselima. Okvir koji se dodaje bi trebalo da bude dovoljno veliki da bude vidljiv, ali i dovoljno mali da ne skreće pažnju sa same slike.

Atribut `BORDER` je, takođe, koristan za uklanjanje graničnog okvira koji se automatski pojavljuje oko linka slike (*image link*). Da bi se uklonio okvir sa linka slike, trebalo bi odrediti širinu ili odmaćen naziv "debljinu" graničnog okvira na 0 piksela. Nakon uklanjanja okvira, trebalo bi se uveriti da li sama slika još uvek predstavlja link (vezu), jer mnogi korisnici očekuju da "slikoviti" linkovi prikazuju okvire, što nije uvek slučaj i zna da se ponaša različito u određenim okolnostima.

²⁸ 28 Fotografija kao pozadina dokumenta HTML.html



Slika 10.23. Granični okvir oko slike ili fotografije²⁹

Atribut **ALIGN** omogućava da se **tekst "prelomi"** oko slike. Ovo može pomoći da veb stranica dobije profesionalni izgled. Ako je autor već koristio atribut **ALIGN** da uskladi sliku sa tekстом, onda ne može da koristi ovaj atribut i za prelom teksta (slika 10.24).

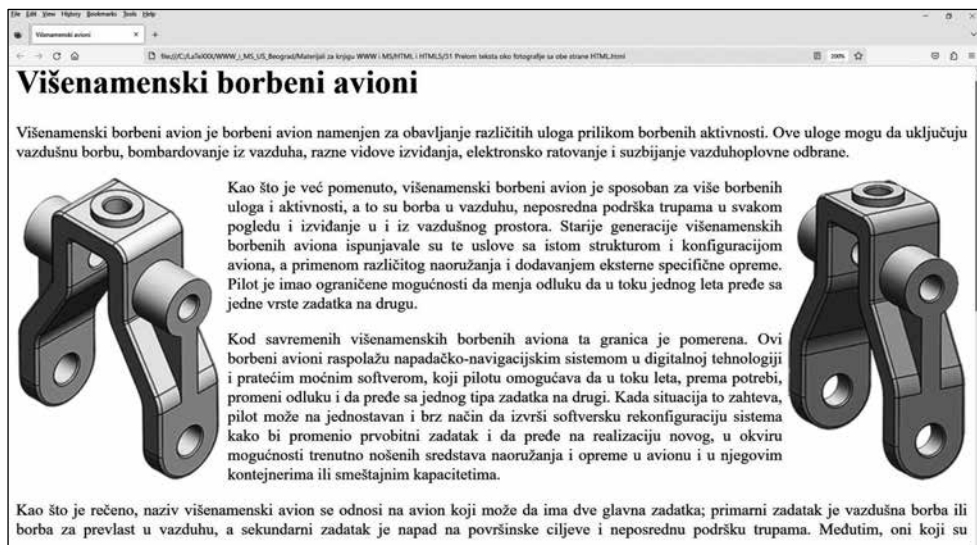


Slika 10.24. Prelom teksta oko fotografije sa jedne strane³⁰

Kada se vodi računa o prelomu teksta oko slike, trebalo bi koristiti vrednosti *left* i *right*, kako bi se definisalo kako će se tekst urediti u odnosu na sliku. Vrednost *left* ostavlja sliku na levoj strani teksta, a vrednost *right* ostavlja sliku na desnoj strani teksta.

²⁹ 29 Okvir oko fotografije HTML.html

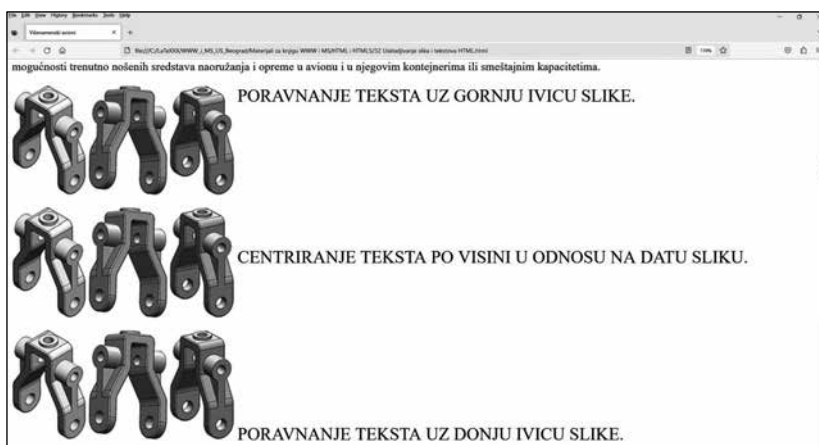
³⁰ 30 Prelom teksta oko fotografije sa jedne strane HTML.html



Slika 10.25. Prelom teksta oko fotografije sa obe strane³¹

Ako autor želi da se samo neki tekst "prelomli" oko slike, onda bi trebalo da koristi atribut `CLEAR` sa oznakom `<BR>` za obeležavanje gde bi trebalo zaustaviti prelom teksta oko slike. Trebalo bi koristiti vrednosti `left`, `right` ili `all` kako bi se definisalo kako će tekst da nastavi da se pruža, kada leva, desna ili obe margine budu "oslobođene" od prisustva slike.

Trebalo bi koristiti atribut `ALIGN` sa vrednostima `top`, `middle` ili `bottom` kako bi se po vertikalni **uskladili slika i tekst**. Uobičajeno je da veb pretraživači poravnavaju donji deo slike sa donjom linijom teksta.



Slika 10.26. Uskladjivanje slika i tekstova po vertikali³²

³¹ 31 Prelom teksta oko fotografije sa obe strane HTML.html

³² 32 Uskladjivanje slika i tekstova HTML.html

Ako postoji više od jedne slike na istoj liniji, onda može da se desi da određene usklađivanje ne ispadne onako kako je autor očekivao. Kada se ovako nešto radi, onda bi veb stranicu trebalo pogledati u par različitih veb pretraživača, kako bi se videlo na šta to liči. Ako je podešen prelom teksta slike, onda pada u vodu poravnavanje slike i teksta. Pored pomenutih vrednosti (top, middle i bottom) postoje i sledeće vrednosti texttop, absmiddle i absbottom, koje imaju isto dejstvo kao pomenute vrednosti, ali ove novopomenute vrednosti nisu sastavni deo HTML standarda. Trebalo bi pomenuti da vrednosti texttop, absmiddle i absbottom nisu podržane od strane svih veb pretraživača.

Uobičajeno je da većina veb pretraživača prikazuje samo nekoliko piksela praznog prostora (beline) između slika i drugih elemenata veb stranice. Povećanje **veličine praznog prostora oko slike** može da poboljša ukupni izgled veb stranice (slika 10.27).



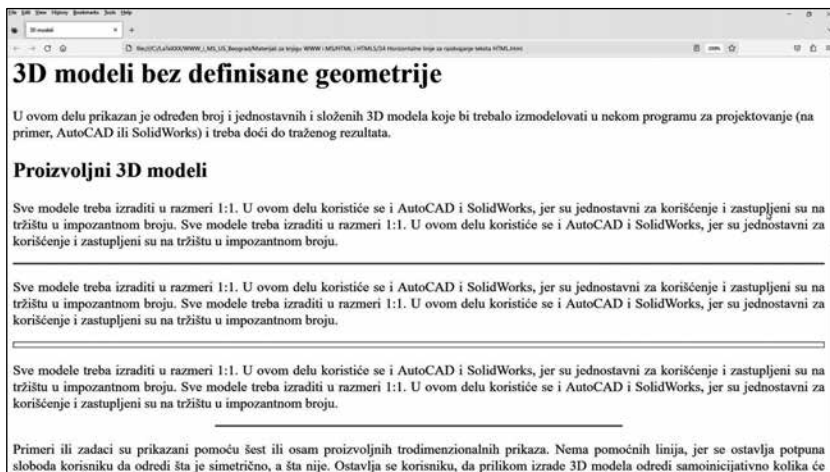
Slika 10.27. Povećanje i usaglašavanje belog prostora oko slike³³

Dodavanje praznog prostora (beline) između slike i okolnog teksta će omogućiti lakše čitanje teksta. Povećanje količine praznog prostora između dve slike, koje se pojavljuju "rame uz rame", će sprečiti slike da se pojavljuju kao jedna velika slika.

Atribut HSPACE omogućava da se doda prazan prostor (belina) sa leve i desne strane slike. Atribut VSPACE omogućava da se doda prazan prostor iznad i ispod slike. Ne može da se dodaje prazan prostor sa samo jedne strane slike koristeći attribute HSPACE i VSPACE. Da bi se dodao prazan prostor samo na jednoj strani slike, onda bi trebalo promeniti samu sliku unutar programa za obradu slika.

Oznaka <HR> omogućava da se doda **horizontalna linija za razdvajanje** unutar veb stranice i reč je o liniji koja vizuelno razdvaja delove veb stranice. Na primer, možda će autor da poželi da koristi horizontalnu liniju da razdvoji delove teksta ili pasuse od zaglavlja na veb stranici.

³³ 33 Definisane velicine praznog prostora oko slike HTML.html



Slika 10.28. Horizontalne "linije" za razdvajanje teksta³⁴

Trebalo bi izbegavati "preteranu" upotrebu horizontalnih linija za razdvajanje unutar veb stranice, jer to može da bude zbunjujuće i može da proizvede da veb stranica bude teško čitljiva. Preporuka je da se ne stavlja više od jedne horizontalne linije za razdvajanje po svakom ekranu.

Uobičajeno je da "debljina" (širina) dodate horizontalne linije za razdvajanje bude 2 piksela. Atribut `SIZE` omogućava da se promeni vrednost "debljine" (širine) horizontalne linije.

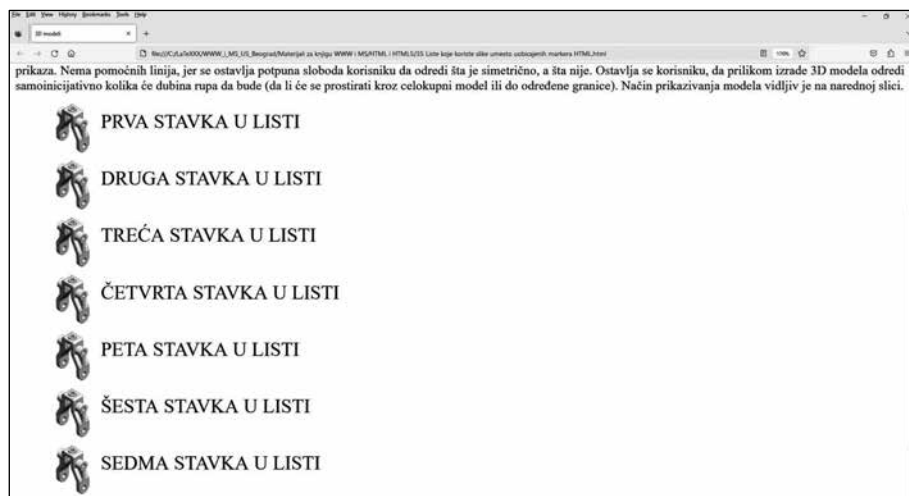
Ako autor ne želi da se horizontalna linije prostire, po širini, preko cele veb stranice, onda bi trebalo koristiti atribut `WIDTH`, koji omogućava da se definiše dužina horizontalne linije za razdvajanje. Trebalo bi navesti novu dužinu horizontalne linije za razdvajanje kao procentualnu vrednost celokupne širine veb stranice unutar prozora veb pretraživača.

Kreiranje liste koja koristi slike umesto uobičajenih markera (*bullets*) omogućava da liste postanu dopadljive unutar veb stranice. Na primer, autor će možda poželeti da doda manju verziju logoa firme ispred svake stavke na listi proizvoda.

Slika koja bi se koristila za listu trebalo bi da bude mala slika, koja će se uklopiti uz svaku stavku na listi. Da bi se kreirale ovakve slike, trebalo bi koristiti program za obradu slike, kao što su *Adobe Photoshop* ili *GIMP*. Kada se dodaje slika u listu umesto markera, trebalo bi koristiti atribut `ALIGN` sa vrednostima `top`, `middle` ili `bottom` koje definišu na koji način se slike usklađuju sa tekстом unutar liste. Vrednost `top` poravnava vrh slike sa tekстом, vrednost `middle` usaglašava sredinu slike po visini sa tekстом i vrednost `bottom` poravnava donji deo slike sa tekстом.

Atributi `HSPACE` i `VSPACE` omogućavaju da se poveća veličina praznog prostora koji se pojavljuje između slika i teksta (predmeta) u listi. Ovo može da poboljša izgled i čitljivost liste. Kada se koriste ovi atributi, onda bi trebalo definisati veličinu dodatnog praznog prostora u pikselima.

³⁴ 34 Horizontalne linije za razdvajanje teksta HTML.html



Slika 10.29. Slike kao markeri unutar liste nabrojanja³⁵

Ako slika koju bi trebalo dodati u veb stranicu nije u GIF ili JPEG formatu, onda bi je trebalo pretvoriti u pomenute formate. Formati GIF i JPEG su najpopularniji formati za slike na internetu. Slike u GIF (*Graphics Interchange Format*) formatu su ograničene na 256 boja i često se koriste za prikazivanje logoa, banera i računarski generisanih "umetničkih dela". GIF slike podržavaju funkcije kao što su *animacija* (*animation*) i *transparentnost* (*transparency*). GIF slike imaju ekstenziju .gif; na primer, logo.gif.

Slike u JPEG (*Joint Photographic Experts Group*) slike mogu da sadrže oko 16,7 miliona boja i često se koriste za fotografije i veoma velikih slika. JPEG slika može da sadrži milione boja, ali i dalje da zadrži malu veličinu, omogućavajući da se slika brzo prenese preko Interneta. JPEG slike obično imaju ekstenziju .jpg; na primer, univerzitet.jpg.

Da bi se pretvorila slika GIF ili JPEG format, autoru je potreban program za obradu slike, kao što su GIMP ili *Adobe Photoshop*.

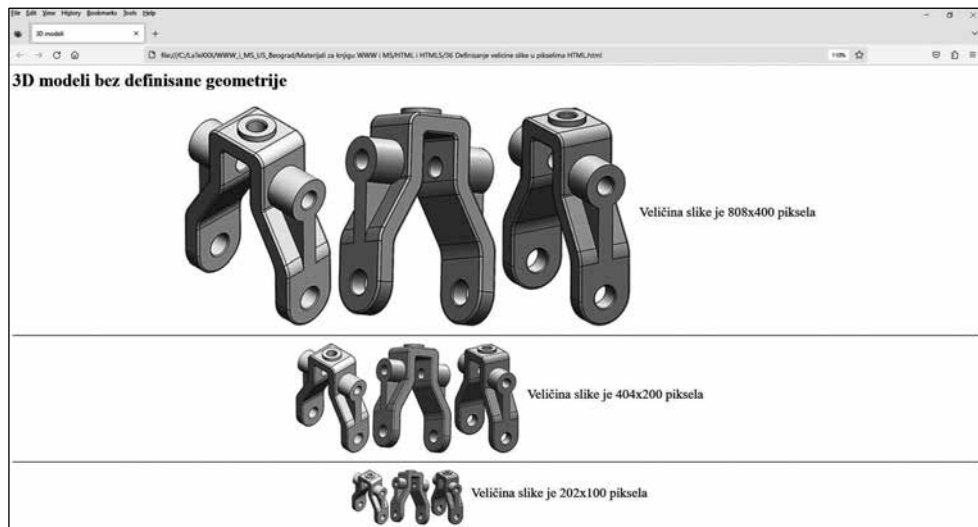
Definisanje veličine slike na veb stranici može da pomogne da se veb stranica pojavljuje na ekranu brže unutar aktivnog veb pretraživača. Da bi se definisala veličina slike, najpre mora da se odredi veličina slike pomoću programa za obradu slika. Jednom kada se saznaju dimenzije slike u pikselima, onda upotrebom atributa WIDTH i HEIGHT može da se definiše veličina slike (slika 10.30).

Kada se definiše veličina slika na veb stranici, veb pretraživači mogu da odrede koliko će prostora da zauzimaju na pripadajućoj veb stranici pre nego što se slike prebace na računar. Ovo sprečava veb pretraživače da promene izgled veb stranice svaki put kada se slika prenosi na računar. Kao rezultat toga, korisnici mogu lako da pročitaju tekst na veb stranici, dok se vrši transfer slike.

Atributi WIDTH i HEIGHT omogućavaju da se poveća veličina slike na veb stranici bez povećanja veličine datoteke slike. Ovo omogućava da se uključuje velika slika na veb

³⁵ 35 Liste koje koriste slike umesto uobicajenih markera HTML.html

stranici bez povećanja potrebnog vremena za transfer veb stranice na računar. Povećanjem veličine slike, određuje se nova veličina slike u pikselima ili u procentima prozora veb pretraživača. Trebalo bi izbegavati da slika bude prevelika, jer slika može da izgleda "zrnasto".



Slika 10.30. Definisanje apsolutne veličine slika³⁶

Kada se navodi nova veličina slike u pikselima, trebalo bi koristiti atribute WIDTH i HEIGHT. Da bi se izbeglo deformisanje ili iskrivljenje slike, odnos širine i visine slike bi trebalo da bude proporcionalan. Kada se navodi novu veličina slike kao procenat prozora veb pretraživača, onda ne mora da se navode vrednosti za širinu i visinu. veb pretraživač će automatski da izračuna drugu procentualnu vrednost kako bi se osiguralo da slika ne bude iskrivljena ili deformisana.



Slika 10.31. Smanjenje veličine slike u programu Adobe Photoshop

³⁶ 36 Definisanje velicine slike u pikselima HTML.html

Možda će autor želiti da **smanji veličinu velike slike**, kako bi zauzimala manje prostora na veb stranici. Smanjenje dimenzija slike smanjuje veličinu slike. Ovo omogućava da se slika prenese brže i da se brže pojavi na monitoru korisnikovog računara. Za ovo bi trebalo koristiti program za uređivanje slika, kao što je GIMP ili *Adobe Photoshop*, kako bi se smanjila veličina slike.



Slika 10.32. *Opsecanje slike u programu Adobe Photoshop*

Većina programa za uređivanje slika omogućava da se nova veličina slike navede u pikselima. Kada se odredi nova širina za sliku, program će automatski da izračuna visinu kako bi se zadržala proporcija celokupne slike. Nakon smanjenja veličine slike, trebalo bi novu sliku sačuvati u novom fajlu, kako bi na računaru ostala i originalna slika. Ovo omogućava da zadržana originalna slika na računaru može i kasnije da se iskoristi, pa čak i ako su zahtevi drugačiji. Na primer, autor može da poželi da koristi originalnu sliku za stvaranje umanjenih fotografija (*thumbnail images*).

Opsecanje ili odsecanje slike omogućava autoru da ukloni delove slike koji mu ne trebaju. Ovo je korisno kada se želi da korisnik obrati pažnju na važan deo slike. Kada se izreže slika ili njen deo, veličina datoteke slike se smanjuje. Ovo omogućava da se slika brže prenese i pojavi na ekranu. Mnogi programi za uređivanje slika, kao što su GIMP i *Adobe Photoshop*, imaju alat koji može da se koristi za odsecanje ili isecanje slike. Nakon odsecanja ili isecanja slike, trebalo bi izmene snimiti u novom fajlu, da bi se sačuvala i originalna slika. Ovo omogućava da se zadrži originalna slika na računaru u slučaju da će originalna slika biti potrebna i u nekom kasnijem trenutku. Čuvanje originalne slike, takođe, omogućava da se ponovo opseca kasnije, ako je autor nezadovoljan rezultatima.

Autor će možda želiti da **pozadina GIF slike bude transparentna**, tako da se pozadina uklopi u veb stranicu. Potreban je program za obradu slike, kao što je GIMP ili *Adobe Photoshop*, da bi se napravila pozadina GIF slike transparentnom.



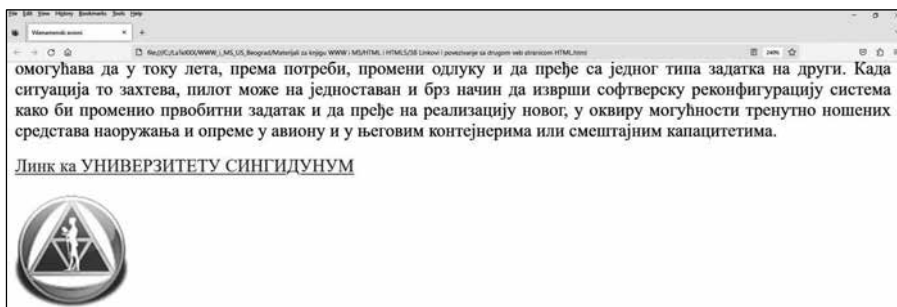
Slika 10.33. Fotografija u PNG formatu na obojenoj pozadini veb dokumenta³⁷

Da bi pozadina GIF slike bila transparentna, cela pozadina mora da sadrži samo jednu boju. Program za obradu slike će svaki piksel koji sadrži tu boju pretvoriti u transparentan ili providan. Ako je pozadina slike sa više boja (raznobojna), samo pikseli koji sadrže definisanu boju će postati transparentni, što će učiniti da pozadina slike izgleda čudno.

10.4. HTML i linkovi i/ili veze

U ovom delu biće reči o linkovima ka drugim veb stranicama, ka elementima unutar tekuće veb stranice, ka slikama (fotografijama), fajlovima, serverima, kao i njihovom podešavanju.

Upotrebom oznake <A> može da se **poveže** tekst ili slika na veb stranici **sa drugom veb stranicom** na internetu. Kada korisnik izabere vezu ili link, sadržaj druge veb stranice će se pojaviti na ekranu. Trebalo bi se uveriti da tekst ili slika jasno pokazuje gde će veza ili link odvesti³⁸ korisnika.



Slika 10.34. Povezivanje sa drugom veb stranicom³⁸

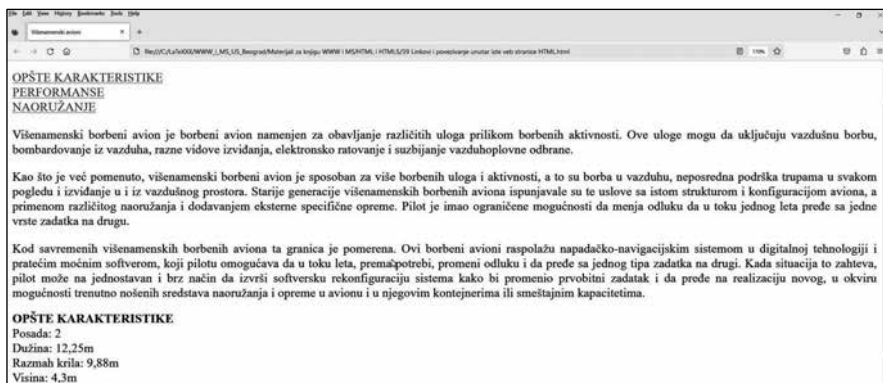
³⁷ 37 Fotografija PNG na obojenoj pozadini dokumenta HTML.html

³⁸ 38 Linkovi i povezivanje sa drugom veb stranicom HTML.html

Link može da odvede korisnike na veb stranicu unutar autorovog sajta ili na bilo koju stranicu na internetu. Kada se kreira veza ka veb stranici koju nije autor napravio, određuje se apsolutna adresa URL (*Uniform Resource Locator*) veb stranice. Apsolutna URL adresa je puna adresa veb stranice, kao što je `http://www.singidunum.ac.rs`. Kada se kreira vezu ka veb stranici na sopstvenom sajtu, onda bi trebalo koristiti relativnu URL adresu.

Trebalo bi izbegavati stavljanje dva teksta linkova rame uz rame na veb stranici, jer korisnici mogu teško da raspoznaju da se radi o dve odvojene veze. Kada se napravi link u obliku slike, treba imati na umu da neki korisnici ne vide slike unutar svog veb pretraživača. Treba uključiti i dodatne odgovarajuće tekstualne linkove za ovakve korisnike.

Možda će autor poželeti da **uključi vezu (link) unutar svoje veb stranice**, koja će korisnike "odvesti" do druge oblasti u istoj veb stranici. Ovo omogućava korisnicima da brzo prikažu informacije od interesa. Na primer, možda će želeći da naprave sadržaj koji sadrži linkove ka različitim delovima dugih veb stranica (slika 10.35).



Slika 10.35. Povezivanje unutar iste veb stranice³⁹

Da bi se napravila veza na veb stranici, najpre bi trebalo imenovati oblast veb stranice koju korisnici žele da brzo prikažu. Ova oblast se često naziva i sidro. Korišćenjem oznake `<A>` sa atributom `NAME` imenuje se oblast veb stranice. Ime se zadaje i može da sadrži samo slova i brojeve.

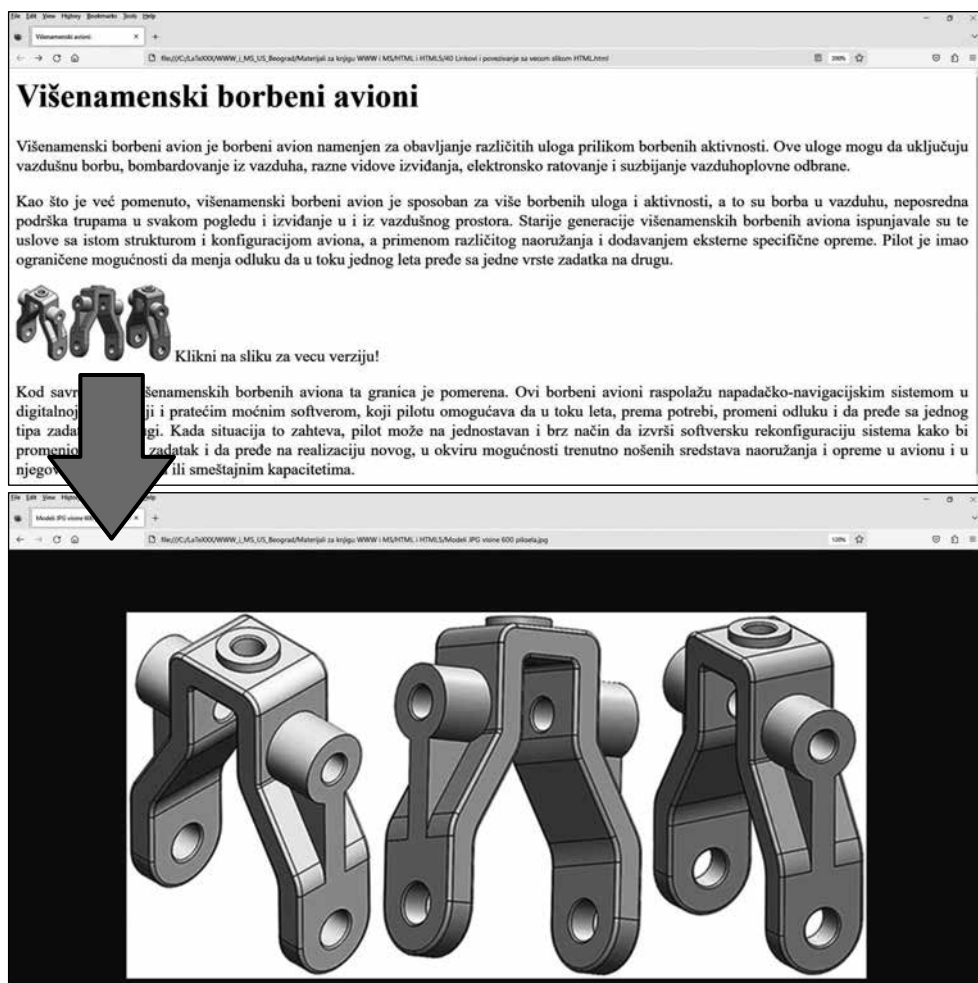
Nakon što je imenovana oblast veb stranice, trebalo bi formirati link ili vezu ka toj oblasti. Oznaka `<A>` sa atributom `HREF` omogućava da se kreira veza ka specifičnoj oblasti tekuće veb stranice. Kada korisnik odabere link, naznačena oblast veb stranice će se pojaviti na ekranu. Kada se kreira više veze u okviru iste veb stranici, trebalo bi proveriti da li je svakoj oblasti dodeljen drugi naziv ili drugačije ime.

Ako autor želi da uključi veliku sliku na svoju veb stranicu, trebalo bi da razmisli o kreiranju **veze koja će korisnike "odvesti" do slike**. To će dati korisnicima pristup slikama bez povećanja potrebnog vremena za učitivanje veb stranice.

Kreiranje linka ka slici je korisno kada je autor uključio umanjene fotografije (*thumbnail images*) u svoju veb stranicu. Umanjene fotografije (*thumbnails*) predstavljaju male ver-

³⁹ 39 Linkovi i povezivanje unutar iste veb stranice HTML.html

zije slika koje korisnici mogu da odaberu kako bi se prikazale veće slike. Ovo omogućava korisnicima da odluče da li žele da čekaju veću sliku da se prenese do njihovog računara.



Slika 10.36. Povezivanje manje i veće slike unutar iste veb stranice⁴⁰

Stvaranje linka na sliku je, takođe, korisno kada autor želi da uključi sliku koju neki veb pretraživači ne mogu da prikažu, kao što je BMP (*Windows Bitmap*) slika. Kada korisnik odabere link, slika će se pojaviti u programu na korisnikovom računaru.

Kada se kreira veza ili link ka slici, trebalo bi obezbediti dodatni opis slike, uključujući i veličinu "povezane" slike u kilobajtima (Kb). Ovo pomaže korisnicima da procene koliko dugo će trebati da se povezana slika učita u njihov računar. Kreiranje linka, kojim korisnici mogu da odaberu određenu datoteku za prenos na svoje računare, omogućava da fajlovi, kao što su dokumenti i programi, budu na raspolaganju korisnicima.

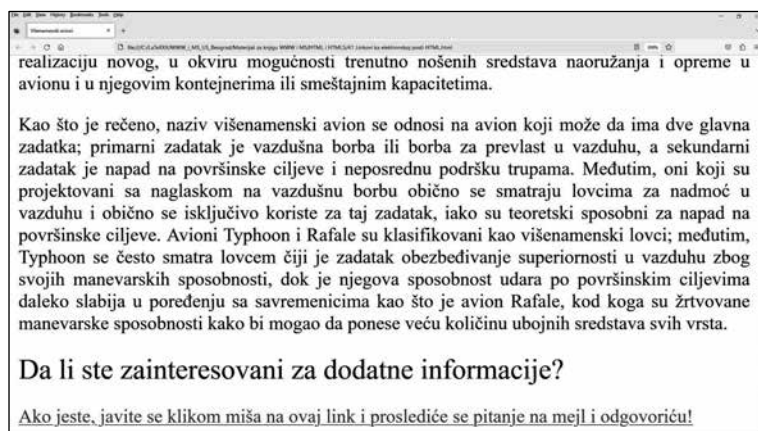
⁴⁰ 40 Linkovi i povezivanje sa vecom slikom HTML.html

Kada se kreira **veza ka datoteci ili fajlu**, mora da se navede lokacija datoteke na računaru. Ako je datoteka sačuvana u istom direktorijumu kao i veb stranica, onda bi trebalo samo navesti ime datoteke (na primer, proba.exe). Ako je datoteka sačuvana u poddirektorijumu, onda bi trebalo navesti i ime poddirektorijuma i naziv datoteke (na primer, programi/proba.exe).

Kada korisnik izabere link ka datoteci, korisnikov veb pretraživač će pokušati da prikaže sadržinu izabrane datoteke (fajla). Ako pretraživač ne može da prikaže datoteku, onda će pokušati da otvori datoteku u odgovarajućem programu na korisnikovom računaru. Ako korisnik nema program koji može da prikaže ili pokrene datoteku, veb pretraživač će dati korisniku mogućnost da sačuva datoteku na svom računaru.

Uključivanje opisa, pored linka ka fajlu, pomaže korisnicima odlučiti da li žele da učitaju ili prenesu datoteku. Opis treba da sadrži vrstu i veličinu datoteke. Takođe bi trebalo obezbediti i vezu (link) koja će "odvesti" korisnike na veb lokaciju na kojoj mogu da se pronađu i dobiju programi koji će prikazati ili pokrenuti datoteku.

Određena veb stranica može da sadrži link koji omogućava korisnicima da pošalju *e-mail* poruku. Kreiranje **linka ka elektronskoj pošti** je korisno kada autor želi da korisnici mogu da pošalju pitanja i da dobiju povratne informacije, kojima mogu da poboljšaju izgled svoje veb stranice.



Slika 10.37. Povezivanje sa elektronskom poštom⁴¹

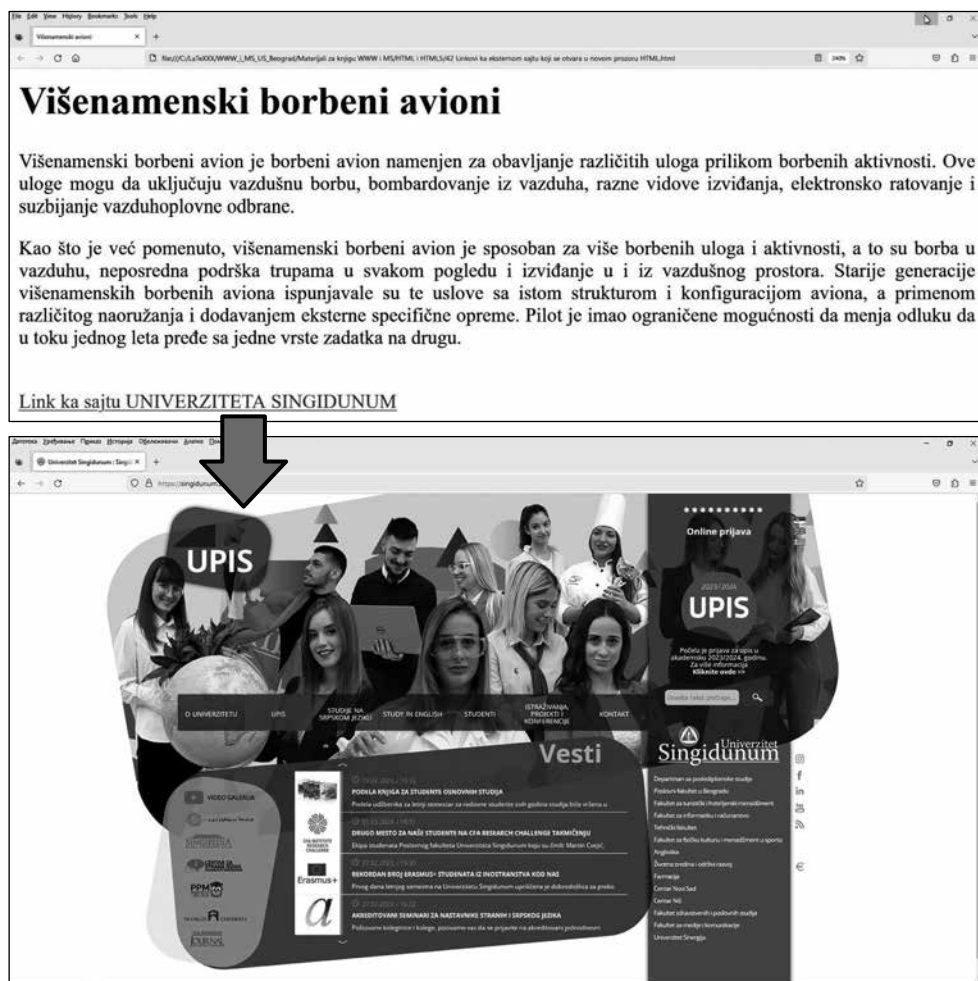
Kad se kreira *e-mail* link, mora da se navede *e-mail* adresa osobe koja bi trebalo da primi poruke od strane korisnika. Na primer, autor će, možda, želeći da napravi *e-mail* link koji omogućava korisnicima da kontaktiraju njega ili nekog od zaposlenih.

Kada se koristi tekst za *e-mail* link, trebalo bi proveriti da li tekst jasno ukazuje na to ko će da primi *e-mail* poruke. Ovo pomaže korisnicima da utvrde da li su kontaktirali pravu osobu. Ako se koristi slika za *e-mail* link, trebalo bi postaviti i odgovarajući tekstualni link teksta za korisnike koji ne mogu da vide slike. Neki korisnici isključuju prikaz slika radi bržeg pregledanja veba, dok drugi koriste veb pretraživače koji ne mogu da prikažu slike.

⁴¹ 41 Linkovi ka elektronskoj pošti HTML.html

Određena veb stranica može da sadrži vezu koja će korisnike "odvesti" do FTP (*File Transfer Protocol*) sajtova. FTP sajtovi su namenjeni za čuvanje datoteka koje korisnici mogu da preuzmu. Kada se kreira veza sa FTP sajtom, mora da se navede puna adresa sajta, počevši sa ftp://. Na primer, da bi se stvorila veza sa ftp.dvd.com sajtom, moralo bi da se ukuca ftp://ftp.dvd.com.

Kada korisnik izabere link ka **eksternom (FTP) sajtu**, glavni direktorijum sajta se, obično, pojavljuje na ekranu. Korisnik može, zatim, da se kreće kroz sajt kako bi pronašao fajlove koji ga zanimaju. U većini slučajeva, korisnici mogu da anonimno posete FTP lokacije, što znači da im ne treba korisničko ime i lozinka kako bi posetili ove sajtove. Ako FTP sajt zahteva korisničko ime i lozinku, onda bi trebalo ove informacije uključiti u link.



Slika 10.38. Otvaranje sajta u novom prozoru⁴²

⁴² 42 Linkovi ka eksternom sajtu koji se otvara u novom prozoru HTML.html

Atribut **TARGET** omogućava da se otvori neki link u novom prozoru. Prozor koji sadrži link će ostati otvoren, omogućavajući korisnicima da se brzo vrate na veb stranicu, kada završe pregled povezane informacije.

Prilikom otvaranja linka u novom prozoru, trebalo bi navesti ime za prozor pomoću atributa **TARGET**. Ime koje se odredi će identifikovati prozor veb pretraživačima, ali se ime neće pojaviti u prozoru. Da bi se nekoliko linkova otvorilo u istom prozoru, trebalo bi navesti isto ime za svaki link. Imena prozora vode računa o velikim i malim slovima (razlikuju ih), tako da bi trebalo koristiti ista velika i mala slova za svako ime. Da bi se veza (link) otvorila u novom, neimenovanom prozoru, trebalo bi koristiti vrednost `_blank` u atributu **TARGET**. Svaki link koji koristi `_blank` vrednost će se otvoriti u drugom prozoru.

Atributi **LINK**, **VLINK** i **ALINK** omogućavaju da se **promeni boja linkova** na veb stranici. Trebalo bi koristiti atribut **LINK** da se promeni boja linkova koje korisnici još nisu izabrali, a atribut **VLINK** za promenu boje linkova koje su korisnici prethodno već izabrali. Atribut **ALINK** omogućava da se promeni boja aktivnih linkova. Aktivan link je link koji korisnik trenutno bira.

Kada se menja boja linkova, trebalo bi voditi računa o tome da se biraju različite boje za "neaktivirane", "aktivirane" i aktivne linkove. Takođe bi trebalo voditi računa da se izabrane boje dobro vide u odnosu na boju pozadine veb stranice.



Slika 10.39. Promena podrazumevanih boja za linkove i veze⁴³

Trebalo bi voditi računa i o tome da izabrane boje mogu da se prikažu na neodgovarajući način na nekim računarima, jer neki korisnici mogu da setuju svoje veb pretraživače da "premoste" izabrane boje.

Korišćenjem atributa **ACCESSKEY** unutar oznake ili taga `<A>` mogu da se stvaraju **prečice na tastaturi za linkove**. Prečice na tastaturi omogućavaju korisnicima da izaberu veze bez upotrebe miša.

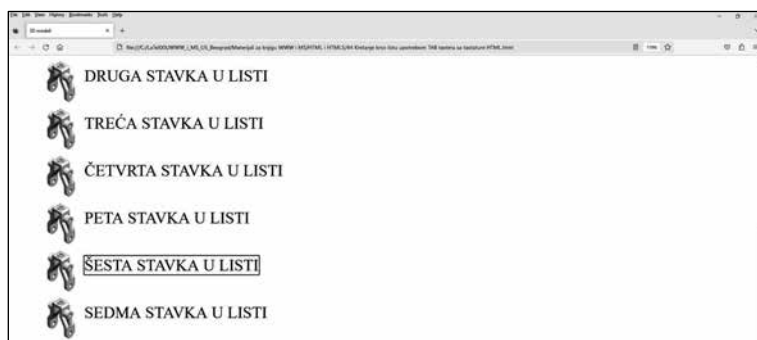
Kada se kreira prečica za vezu, trebalo bi navesti slovo ili broj koji bi trebalo koristiti za prečicu. Takođe bi trebalo da sadrži opis koji će omogućiti korisnicima da znaju kako da koriste prečicu.

⁴³ 43 Promena podrazumevanih boja linkova HTML.html

Da bi se izabrala veza pomoću prečice na tastaturi, korisnik će držati pritisnut taster Alt dok pritiska navedeno slovo ili broj. Korisnik može, zatim, da pritisne taster Enter kako bi posetio link. Kreirane prečice na tastaturi za veze još uvek ne rade na *Macintosh* računarima, jer će *Macintosh* korisnici, verovatno, koristiti taster Command pre nego taster Alt, kada su podržane prečice.

Prečice sa tastature, koje su kreirane za linkove, će "nadjačati" prečice na tastaturi za veb pretraživače. Na primer, ako se napravi prečica Alt + F na tastaturi, korisnici više neće moći da koriste Alt + F da bi prikazali meni *File* u *Internet Explorer*-u.

Korisnici mogu da se **kreću kroz linkove** na veb stranici korišćenjem Tab tastera. Uobičajeno je da se, na ovaj način, korisnici kreću po linkovima po redosledu koji je kreiran u HTML kodu. Promenom redosleda kretanja po linkovima pomoću Tab tastera može da pomogne korisnicima da izaberu linkove po nekom logičkom redu.



Slika 10.40. Kretanje kroz listu TAB tasterom⁴⁴

Korišćenjem atributa TABINDEX može da se promeni redosled biranja linkova pomoću tastera Tab unutar veb stranice. Da bi se uključila veza (link) u redosled biranja pomoću Tab tastera, trebalo bi linku dodeliti TABINDEX vrednost između 1 i 32.767. Da bi se isključila veza iz Tab naloga, trebalo bi dodeliti negativnu TABINDEX vrednost (na primer, TABINDEX="-1").

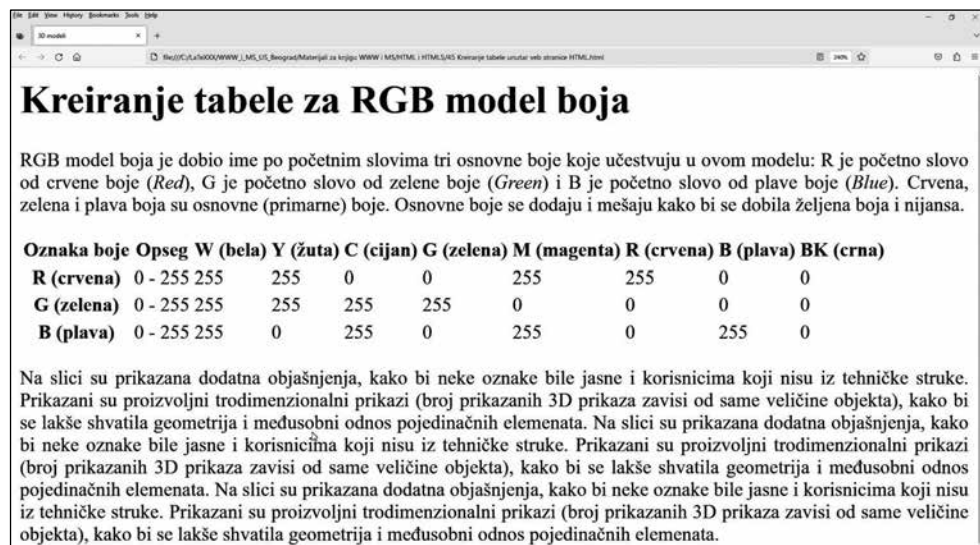
Kada korisnici koriste taster Tab za biranje linkova, redosled počinje sa linkom koji ima najnižu vrednost, a završava sa linkom koji ima dodeljenu najvišu vrednost. Ako dva linka imaju istu vrednost, onda će se aktivirati veza koja se prva pojavljuje u HTML kodu. Atribut TABINDEX može da se koristi i sa drugim HTML elemenatima, kao što su slike i forme ili sekcije.

10.5. HTML i tabelle

Tabele su sastavni deo skoro svake veb stranice i u ovom delu biće reči o njima – kako se kreiraju, podešavaju, povezuju, kako im se spajaju "čelije", kako im se menjaju boje, kako se podešava vidljivost graničnih linija ili okvira i slično.

⁴⁴ 44 Kretanje kroz listu upotrebom TAB tastera sa tastature HTML.html

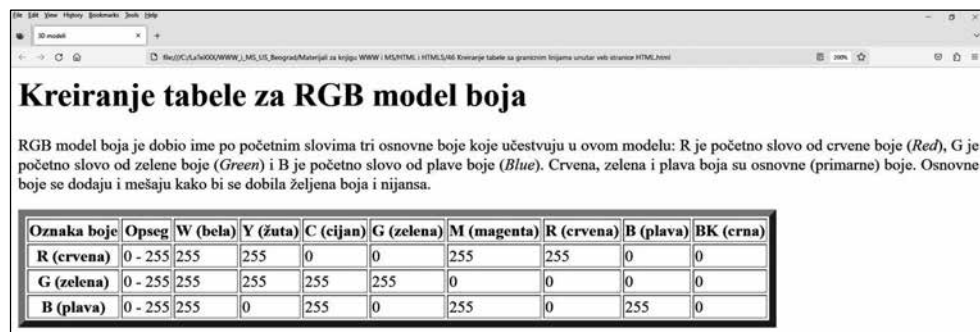
Kreiranjem tabele omogućava se uredno prikazivanje informacija na veb stranici, kao što su finansijski podaci ili cenovnik. Kada se pravi tabela, moraju da se koriste oznake ili tagovi za kreiranje redova i ćelija u tabeli. Oznaka <TR> omogućava kreiranje redova i oznake <TH> i <TD> omogućavaju kreiranje ćelija.



Slika 10.41. *Kreiranje tabele unutar veb stranice*⁴⁵

Postoje dve vrste ćelija koje mogu da se kreiraju u tabeli. Oznaka <TH> omogućava da se kreiraju ćelije zaglavlja i oznaka <TD> omogućava da se kreiraju ćelije podataka. Ćelije zaglavlja obično sadrže tekst koji opisuje podatke u redu ili koloni, dok ćelije podataka, obično, sadrže glavne informacije u tabeli. Tekst u ćelijama zaglavlja je obično podebljan (**bold**) i centriran unutar ćelija.

Atribut BORDER omogućava da se dodaju **granične linije u tabeli**. Granična linija će razdvojiti svaku ćeliju u tabeli, što dovodi do lakšeg čitanja podataka u tabeli.



Slika 10.42. *Kreiranje tabele unutar veb stranice sa graničnim linijama*⁴⁶

⁴⁵ 45 Kreiranje tabele unutar veb stranice HTML.html

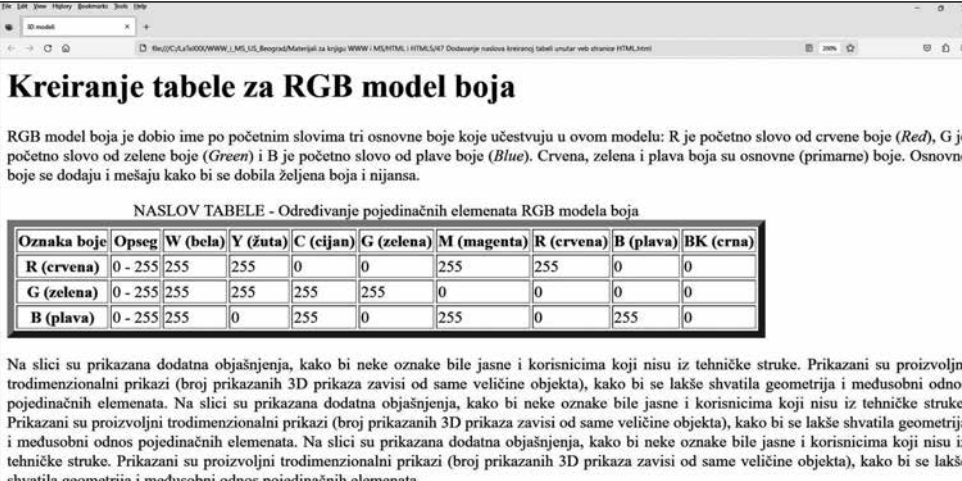
⁴⁶ 46 Kreiranje tabele sa granicnim linijama unutar veb stranice HTML.html

Kada se dodaje granična linija u tabelu, trebalo bi definisati "debljinu" ili širinu linije u pikselima. Navedena "debljina" linije će uticati samo na spoljašnju graničnu liniju tabele. Da bi se promenila "debljina" granične linije između ćelija u tabeli, trebalo bi koristiti atribut `CELLSPACING`.

Granične linije tabele se obično pojavljuju kao sive linije u veb stranici. Ako je promenjena boja pozadine veb stranice, može da se desi da veb pretraživači prikazuju graničnu liniju tabele u istoj boji kao i pozadina. Trebalo bi koristiti atribut `BORDERCOLOR` za definisanje drugačije boje granične linije tabele. Iako je atribut `BORDERCOLOR` podržan od strane većine veb pretraživača, nije deo HTML standarda.

Ako se koristi tabela za organizovanje izgleda veb stranice, onda je poželjno da se granične linije privremeno vide, kako bi moglo da se vidi gde su smešteni tekst, slike i drugi elementi. Iako mogu da se uklone granične linije kada je rad na veb stranici završen. Oznaka `<CAPTION>` omogućava da se **doda naslov tabeli**. Naslovi su korisni za sumiranje informacija u tabeli.

Po pravilu, većina veb pretraživača će prikazati centrirane naslove iznad tabele. Ovo je korisno za prikazivanje naslova za tabelu. Korišćenje atributa `ALIGN` sa vrednošću `bottom` omogućava da se naslov pojavljuje ispod tabele. Prikazivanje naslova ispod tabele je korisno kada autor želi da dostavi dodatne informacije ili da sumira podatke u tabeli.



Kreiranje tabele za RGB model boja

RGB model boja je dobio ime po početnim slovima tri osnovne boje koje učestvuju u ovom modelu: R je početno slovo od crvene boje (*Red*), G je početno slovo od zelene boje (*Green*) i B je početno slovo od plave boje (*Blue*). Crvena, zelena i plava boja su osnovne (primarne) boje. Osnovne boje se dodaju i mešaju kako bi se dobila željena boja i nijansa.

NASLOV TABELE - Određivanje pojedinačnih elemenata RGB modela boja

Oznaka boje	Opseg	W (bela)	Y (žuta)	C (cijan)	G (zeleni)	M (magenta)	R (crvena)	B (plava)	BK (crna)
R (crvena)	0 - 255	255	255	0	0	255	255	0	0
G (zeleni)	0 - 255	255	255	255	255	0	0	0	0
B (plava)	0 - 255	255	0	255	0	255	0	255	0

Na slici su prikazana dodatna objašnjenja, kako bi neke oznake bile jasne i korisnicima koji nisu iz tehničke struke. Prikazani su proizvoljni trodimenzionalni prikazi (broj prikazanih 3D prikaza zavisi od same veličine objekta), kako bi se lakše shvatila geometrija i međusobni odnos pojedinačnih elemenata. Na slici su prikazana dodatna objašnjenja, kako bi neke oznake bile jasne i korisnicima koji nisu iz tehničke struke. Prikazani su proizvoljni trodimenzionalni prikazi (broj prikazanih 3D prikaza zavisi od same veličine objekta), kako bi se lakše shvatila geometrija i međusobni odnos pojedinačnih elemenata. Na slici su prikazana dodatna objašnjenja, kako bi neke oznake bile jasne i korisnicima koji nisu iz tehničke struke. Prikazani su proizvoljni trodimenzionalni prikazi (broj prikazanih 3D prikaza zavisi od same veličine objekta), kako bi se lakše shvatila geometrija i međusobni odnos pojedinačnih elemenata.

Slika 10.43. Dodavanje naslova kreiranoj tabeli⁴⁷

HTML standard određuje da ne bi trebalo dodavati više od jednog naslova tabeli na veb stranici. Dok neki veb pretraživači podržavaju korišćenje više tagova `<CAPTION>` sa jednim tagom `<TABLE>`, rezultati su različiti u različitim veb pretraživačima.

Oznake `<COLGROUP>` i `<COL>` omogućavaju da se **grupišu kolone u tabeli**. Grupisanje kolona deli tabelu u vertikalne sekcije, omogućavajući da se formatira jedna ili više kolona ćelija u isto vreme.

⁴⁷ 47 Dodavanje naslova kreiranoj tabeli unutar veb stranice HTML.html

Upotrebom oznake <COLGROUP> se vrši kreiranje strukturnih grupa kolona koje razdvajaju tabelu u logične celine. Na primer, može da se koristi jedna grupisana kolona ćelija za naslove, dok druga strukturna kolona grupu služi za smeštaj ostalih kolona u tabeli.

Oznaka <COL> omogućava da se kreiraju nestrukturne grupe kolona, koje razdvajaju tabelu u sekcije bez definisanje strukture unutar tabele.

Kada se naprave grupe kolona u tabeli, onda može da se pristupi njihovom formatiranju i oblikovanju. Na primer, može da se iskoristi atribut BGCOLOR unutar tagova <COLGROUP> i <COL> kako bi se boja svim ćelijama u grupisanoj koloni.

Korišćenjem tagova <THEAD>, <TBODY> i <TFOOT> je omogućeno **kreiranje grupisanih redova u tabeli**. Grupisani redovi dele tabelu u horizontalne sekcije, što omogućava brzo formatiranje više redova ćelija u isto vreme.

Oznaka <THEAD> omogućava da se kreira grupisani red zaglavlja (*heading*) u tabeli. Ovo je korisno za zaglavlja ili podnaslove koje bi trebalo oblikovati drugačije u odnosu na osnovne podatke u tabeli. Oznaka <THEAD> može samo jednom da se pojavi u tabeli.

Korišćenjem oznake <TBODY> je omogućeno kreiranje jedne ili više "glavnih" grupa u tabeli. Ove grupe obično sadrže redove sa podacima u tabeli.

Oznaka <TFOOT> omogućava da se kreirate grupisani red podnožja (*footer*) u tabeli. Ovo je korisno za prikazivanje zbirnih podataka ili iznosa koji se pojavljuju na dnu tabele. Oznaka <TFOOT> može samo jednom da se pojavi u tabeli.

Atribut BGCOLOR omogućava da se **doda boja u tabeli** na veb stranici. Ovo može da pomogne da se ukaže na važne informacije u tabeli.

Kreiranje tabele za RGB model boja

RGB model boja je dobio ime po početnim slovima tri osnovne boje koje učestvuju u ovom modelu: R je početno slovo od crvene boje (*Red*), G je početno slovo od zelene boje (*Green*) i B je početno slovo od plave boje (*Blue*). Crvena, zelena i plava boja su osnovne (primarne) boje. Osnovne boje se dodaju i mešaju kako bi se dobila željena boja i nijansa.

NASLOV TABELE - Određivanje pojedinačnih elemenata RGB modela boja

Oznaka boje	Opseg	W (bela)	Y (žuta)	C (cijan)	G (zelena)	M (magenta)	R (crvena)	B (plava)	BK (crna)
R (crvena)	0 - 255	255	255	0	0	255	255	0	0
G (zelena)	0 - 255	255	255	255	255	0	0	0	0
B (plava)	0 - 255	255	0	255	0	255	0	255	0

Slika 10.44. Dodavanje pozadinske boje određenim ćelijama⁴⁸

Boja može da se doda ćeliji (<TH> ili <TD>), grupisanoj koloni (<COLGROUP> ili <COL>), grupisanom redu (<THEAD>, <TBODY> ili <TFOOT>) ili celoj tabeli (<TABLE>).

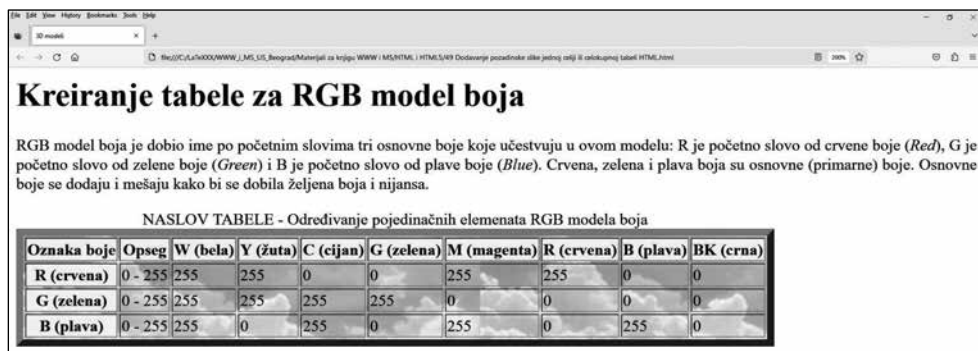
Kada se dodaje boja, trebalo bi da se navede ime ili heksadecimalna vrednost boje koju autor želi da koristi. Heksadecimalna vrednost je kôd koji govori veb pretraživaču koju boju treba da prikaže. Kôd se sastoji od znaka (#), koga slede crvena, zelena i plava (RGB) komponenta boje.

⁴⁸ 48 Dodavanje pozadinske boje određenim poljima HTML.html

Trebalo bi proveriti da boja, koja se koristi, ne utiče na čitljivost podataka unutar tabele. Možda će morati da se promeni boja teksta da bi se podaci u tabeli lakše pročitali.

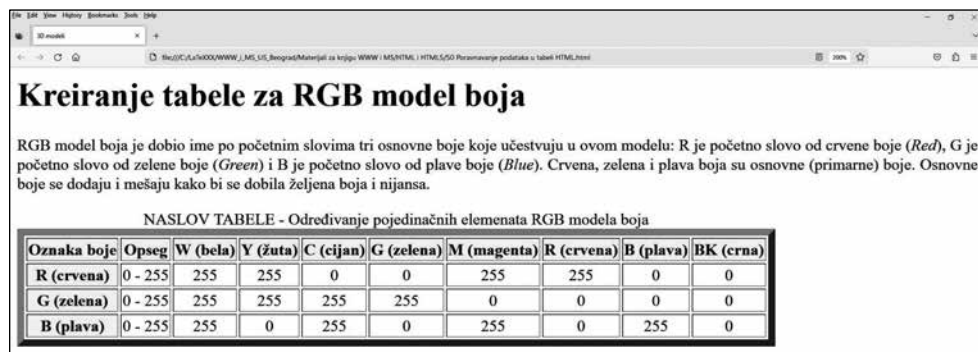
Korišćenjem atributa BACKGROUND omogućeno je **dodavanje pozadinske slike jednoj ćeliji ili celoj tabeli**. Dodavanje pozadinske slike jednoj ćeliji može da pomogne da se informacije u ćeliji ističu. Dodavanje pozadinske slike celokupnoj tabeli može da doprinese zanimljivom dizajnu tabeli.

Trebalo bi se uveriti da izabrana pozadinska slika ne utiče na čitljivost podataka u tabeli. Takođe bi trebalo proveriti da li je pozadinska slika odgovarajuće veličine. Ako se koristi pozadinska slika koja je veća od ćelije ili tabele, veb pretraživač će da "odseče" sliku kako bi stala u tabelu.



Slika 10.45. Dodavanje pozadinske slike ćeliji ili tabeli⁴⁹

Prilikom dodavanja pozadinske slike celokupnoj tabeli, trebalo bi obratiti pažnju na to da će različiti veb pretraživači prikazivati pozadinsku sliku na različite načine. Atributi ALIGN i VALIGN omogućavaju da se promene horizontalno i vertikalno **poravnanje podataka u tabeli**. Za promenu horizontalnog poravnanja podataka trebalo bi koristiti atribut ALIGN. Uobičajeno je da su podaci u ćelijama zaglavlja (<TH>) centrirani, a podaci u ćelijama podataka (<TD>) poravnati po levoj margini.



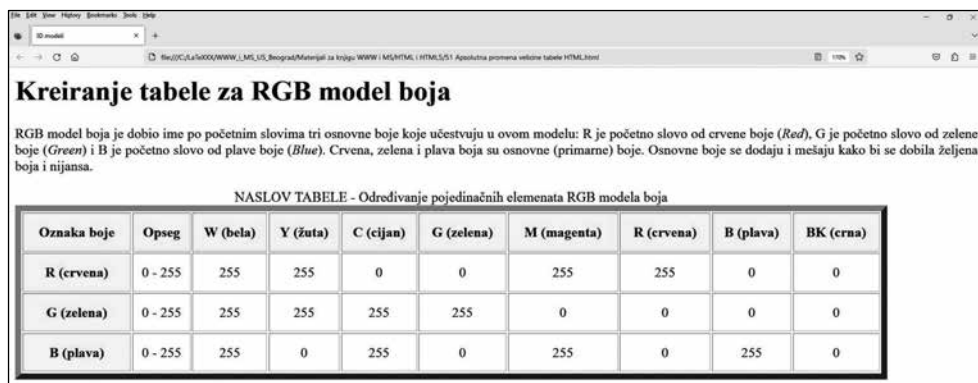
Slika 10.46. Poravnavanje podataka u tabeli⁵⁰

⁴⁹ 49 Dodavanje pozadinske slike jednoj ćeliji ili celokupnoj tabeli HTML.html

⁵⁰ 50 Poravnavanje podataka u tabeli HTML.html

Atribut **VALIGN** omogućava da se promeni vertikalno poravnanje podataka. Uobičajeno je da se podaci pojavljuje u sredini svake ćelije u tabeli. Korisniku je dozvoljeno da promeni poravnanje podataka u ćeliji (<TH> ili <TD>), redu (<TR>), grupisanoj koloni (<COLGROUP> ili <COL>) ili grupisanom redu (<THEAD>, <TBODY> ili <TFOOT>).

Promena veličine tabele je korisna kada autor želi da tabelom zauzme određeni prostor na veb stranici.



Kreiranje tabele za RGB model boja

RGB model boja je dobio ime po početnim slovima tri osnovne boje koje učestvuju u ovom modelu: R je početno slovo od crvene boje (*Red*), G je početno slovo od zelene boje (*Green*) i B je početno slovo od plave boje (*Blue*). Crvena, zelena i plava boja su osnovne (primarne) boje. Osnovne boje se dodaju i mešaju kako bi se dobila željena boja i nijansa.

NASLOV TABELE - Određivanje pojedinačnih elemenata RGB modela boja

Oznaka boje	Opseg	W (bela)	Y (žuta)	C (cijan)	G (zeleni)	M (magenta)	R (crvena)	B (plava)	BK (crna)
R (crvena)	0 - 255	255	255	0	0	255	255	0	0
G (zeleni)	0 - 255	255	255	255	255	0	0	0	0
B (plava)	0 - 255	255	0	255	0	255	0	255	0

Slika 10.47. *Apsolutna promena veličine tabele*⁵¹

Atribut **WIDTH** omogućava da se promeni širina tabele navođenjem nove širine u pikselima ili u procentima prozora veb pretraživača. Kada se navodi širina u pikselima, trebalo bi koristiti širinu od 600 piksela ili manje, kako bi se cela tabela videla na monitoru korisnika. Ako autor želi da širina tabele varira u zavisnosti od veličine prozora veb pretraživača, onda bi trebalo odrediti širinu kao procenat prozora veb pretraživača.

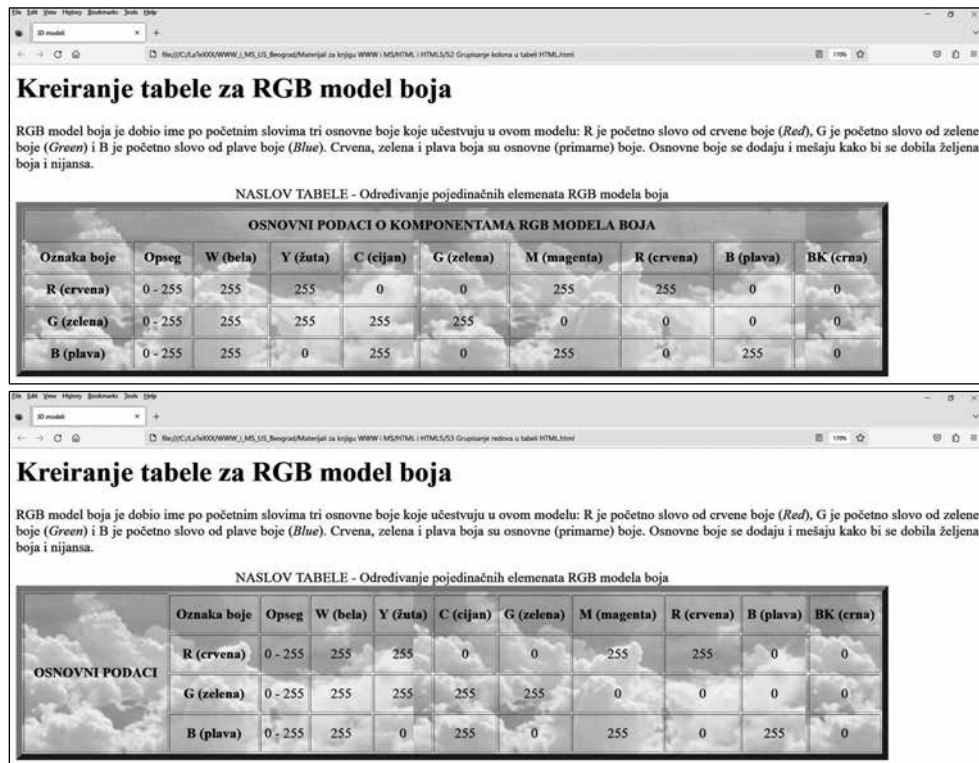
Korišćenjem atributa **HEIGHT** omogućena je promena visine tabele navođenjem nove visine u pikselima ili u procentima prozora veb pretraživača. Atribut **HEIGHT** nije deo HTML standarda za tabele i može da se desi da neki veb pretraživači u potpunosti ne podržavaju ovaj atribut.

Ako se navedu širina ili visina koje su manje od sadržaja tabele, veb pretraživač će pokazati tabelu kako bi se sadržaj tabele prikazao kako treba. **Promena veličine ćelija u tabeli** može da poboljša izgled tabele. Atribut **WIDTH** omogućava da se navede nova širina ćelije u pikselima ili u procentima širine tabeli. Atribut **HEIGHT** omogućava da se odredi nova visina ćelije u pikselima. Ne mora da se menja veličina svake pojedinačne ćelije u tabeli. Kada se promeni širina ćelije, sve ćelije u istoj koloni će prikazati novu širinu. Kada se promeni visina ćelije, sve ćelije u istom redu će prikazati novu visinu.

Ako se navedu širina ili visina koji su manji od sadržaja ćelije u koloni ili redu, veb pretraživač će prikazati kolonu ili red toliko malim koliko to sadržaj ćelije omogućava.

"Spajanje" ćelija omogućava da se kombinuju dva ili više ćelija u redu ili koloni u jednu veliku ćeliju. Ovo je korisno kada autor želi da prikaže naslov na vrhu ili u dnu tabele. "Spajanje" ćelija je korisna kada autor želi da prikaže naslov u više redova ili kolona.

⁵¹ 51 Apsolutna promena velicine tabele HTML.html



Slika 10.48. "Spajanje" ćelija unutar tabele^{52 53}

Upotreba atributa COLSPAN unutar tagova <TH> ili <TD> omogućava "spajanje" ćelija unutar kolone. Korišćenjem atributa ROWSPAN unutar tagova <TH> ili <TD> omogućeno je "spajanje" ćelija unutar reda. Kada se vrši "spajanje" ćelija duž kolone ili reda, autor mora da navede broj ćelija koje se spajaju u jednu.

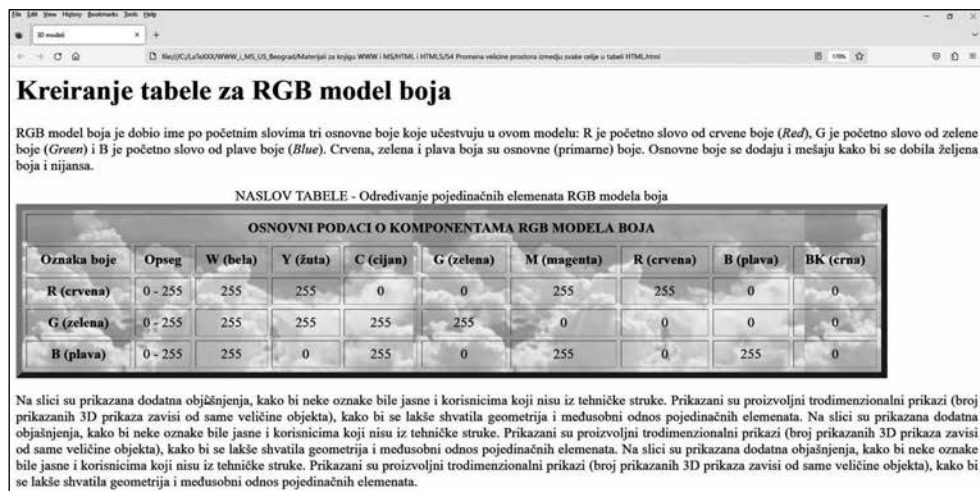
Atributi CELLSPACING i CELLPADDING se često koriste za poboljšanje izgleda i čitljivosti tabele. Atribut CELLSPACING omogućava da se **promeni veličina prostora između svake ćelije u tabeli**. Promenom razmaka ćelija promeniće se veličina granične linije između ćelija. Veličina dodatnog prostora se navodi u pikselima. Uobičajeno je da tabele prikazuju ćelije sa "slobodnim" prostorom od 2 piksela.

Atribut CELLPADDING omogućava da se **promeni veličina prostora oko sadržaja svake ćelije u tabeli**. Ovo pospešuje bolji izgled tabele, gde tabela ne izgleda da je pretrpana. Uobičajeno je da se ostavlja slobodan prostor od 1 piksela oko sadržaja ćelija.

Promena praznog prostora oko sadržaja ćelija u tabeli će uticati na usklađivanje i poravnanje podataka u tabeli. Na primer, ako su podaci poravnati po gornjoj (top) margini ćelije ili u odnosu na vrh ćelije i ako se odredi vrednost atributa CELLPADDING da bude 5 piksela, onda će se podaci spustiti 5 piksela ispod vrha ćelije.

⁵² 52 Grupisanje kolona u tabeli HTML.html

⁵³ 53 Grupisanje redova u tabeli HTML.html



Slika 10.49. Promena veličine prostora u i oko ćelija^{54 55}

Kada se doda granična linija tabeli upotrebom BORDER atributa, onda se granična linija (okvir) automatski pojavljuje oko spoljne strane tabele i između svake ćelije. Upotrebom atributa FRAME i RULES omogućeno je da se **prikažu samo neke granične linije u tabeli**, što će definisati korisnik.

Atribut FRAME omogućava da se odredi koje spoljna granične linije treba prikazati. Atribut RULES omogućava da se odredi koje unutrašnje granične linije treba prikazati. Korišćenje atributa FRAME i RULES unutar istog taga <TABLE> omogućava autoru da kreira interesantne prikaze graničnih linija u tabeli.

Atribut FRAME može da ima različite vrednosti u cilju prikazivanja spoljašnjih ili spoljnjih graničnih linija. Te vrednosti su prikazane u tabeli 10.1.

Tabela 10.1 Atribut FRAME i prikaz spoljašnjih graničnih linija

Vrednost	Šta se prikazuje?
void	Nema eksternih graničnih linija
above	Granična linija iznad tabele
below	Granična linija ispod tabele
rhs	Granična linija sa desne strane tabele
lhs	Granična linija sa leve strane tabele
hsides	Granična linija i iznad i ispod tabele (slika 10.50, iznad)
vsides	Granična linija i sa leve i sa desne strane tabele
border	Sve spoljašnje granične linije (<i>default</i>)

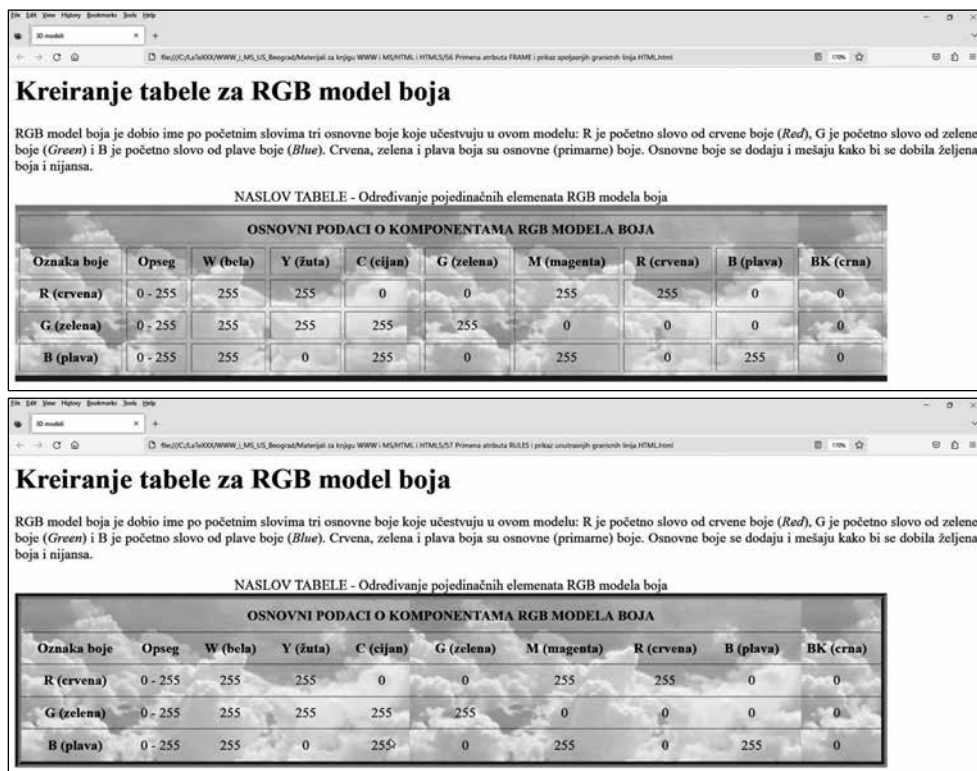
⁵⁴ 54 Promena velicine prostora izmedju svake celije u tabeli HTML.html

⁵⁵ 55 Promena velicine prostora oko sadrzaja svake celije u tabeli HTML.html

Atribut **RULES** može da ima različite vrednosti u cilju prikazivanja unutrašnjih graničnih linija. Te vrednosti su prikazane u tabeli 10.2.

Tabela 10.2 Atribut *RULES* i prikaz unutrašnjih graničnih linija

Vrednost	Šta se prikazuje?
none	Nema internih graničnih linija
cols	Granična linija između kolona
rows	Granična linija između redova (slika 10.50, ispod)
groups	Granična linija između grupisanih kolona i redova
all	Sve unutrašnje granične linije (<i>default</i>)



Slika 10.50. Prikazivanje pojedinih graničnih linija u tabeli^{56 57}

Veb pretraživač će, obično, automatski da prenese tekst u naredni red u ćeliji u zavisnosti od veličine tabele i veličine veb prozora. Upotrebom atributa **NOWRAP** onemogućava se bilo kakav **prelom teksta u ćeliji**. Ovo je korisno kada se želi da se sav tekst u ćeliji pojavi u

⁵⁶ 56 Primena atributa **FRAME** i prikaz spoljasnih graničnih linija HTML.html

⁵⁷ 57 Primena atributa **RULES** i prikaz unutrašnjih graničnih linija HTML.html

jednoj liniji.

Trebalo bi koristiti atribut NOWRAP samo u ćelijama koje sadrže malu količinu teksta, kao što su ćelije koje sadrže naslov ili ime proizvoda. Korišćenje atributa NOWRAP u ćeliji koja sadrži veliku količinu teksta će stvoriti široku (dugačku) ćeliju koja može da izazove da tabela "izađe" van granica prozora veb pretraživača. Kada se koristi atribut NOWRAP u ćeliji, trebalo bi imati na umu da se tekst u ćeliji neće pojaviti u jednoj liniji, ako se tagovi <P> ili
 pojavljuju u ćeliji.

Smeštanje (ugnežđivanje) tabele unutar druge tabele omogućava da se kreiraju složeniji rasporedi tabela.

Da bi se "ugnezдила" tabela, trebalo bi prvo napraviti glavnu tabelu, ostavljajući praznu ćeliju u koju autor želi da smesti drugu tabelu. Onda može da se "ugnezdi" tabela, kreiranjem nove tabele u praznoj ćeliji.

Prilikom kreiranja tabele, trebalo bi proveriti da li su uključene sve krajnje oznake ili tagovi (</TH>, </TD>, </TR> i </TABLE>) i za glavnu i za ugneždenu tabelu. Zaboravljanje ovih krajnjih tagova može negativno da utiče na izgled tabele.

DOSTUPNI BORBENI AVIONI ZA PRODAJU		
Naziv aviona	Cena po komadu	Dostupan broj aviona
F-16A F-16B F-16C	20.000.000\$	200
F-15	40.000.000\$	150
F/A-18	25.000.000\$	300
F-14	55.000.000\$	85

Slika 10.51. Smeštanje tabele unutar tabele⁵⁸

Atribut ALIGN omogućava da se **prelomi tekst oko tabele**. Ovo može da pomogne da veb stranica dobije profesionalniji izgled.

Kada se vrši prelom teksta oko tabele, trebalo bi koristiti vrednost left i right, kako bi se odredilo kako će se tekst rasporediti oko tabele. Vrednost left ostavlja tabelu sa leve strane teksta, a vrednost right ostavlja tabelu sa desne strane teksta.

Ako autor želi da se samo deo teksta prelomi oko tabele, trebalo bi da koristi atribut CLEAR sa tagom
 na mestu gde bi trebalo zaustaviti prelamanje teksta oko tabele. Trebalo

⁵⁸ 58 Smestanje tabele unutar postojeće tabele HTML.html

bi koristiti vrednosti `left`, `right` ili `all` kako bi se definisalo da li će se tekst nastaviti tamo gde su leva, desna ili obe margine "oslobođene" prisustva bilo kakve tabele.



Slika 10.52. Prelom pratećeg teksta u dokumentu oko tabele⁵⁹

Preporuka je da se vrši prelamanje teksta samo oko malih tabela unutar veb stranice. Prelom teksta oko velikih tabela može da proizvede neočekivane rezultate, kao što je preklapanje teksta i tabele.

10.6. HTML, zvuk i video

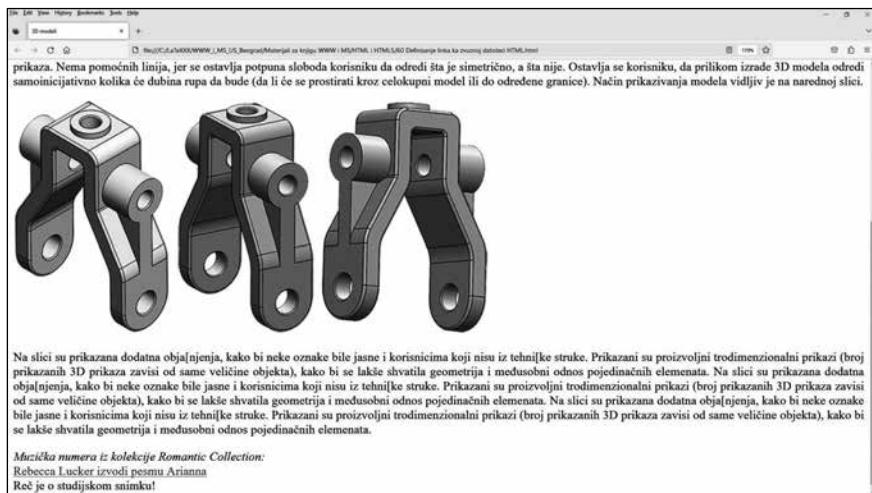
U ovom delu će biti priče oko zvučnog i video materijala (fajlova sa zvukom, kao što su pesme i fajlova sa videom, kao što su filmovi i "klipovi") i kako ih "ugraditi" u veb stranicu, kao i mogućim problemima prilikom "ugradnje" fajlova.

Svaka veb stranica može da sadrži **link ka zvučnoj datoteci** kojim korisnici mogu da izaberu da se čuje zvuk (pesma, zvučni signal i slično). Kada se kreira veza (link) ka zvučnoj datoteci, trebalo bi proveriti da li je format zvučne datoteke prikladan za korišćenje na internetu (slika 10.53). Takođe bi trebalo proveriti da li se ima dozvola za korišćenje zvučne datoteke koju autor nije kreirao sâm.

Kada korisnik izabere link ka zvučnoj datoteci, veb pretraživač će pokušati da reprodukuje zvuk. Ako veb pretraživač ne može da reprodukuje zvuk, pretraživač će pokušati da ga reprodukuje u nekom programu na korisnikovom računaru. Ako korisnik nema odgovarajuć program, veb pretraživač će omogućiti korisniku da sačuva zvučnu datoteku na svom računaru.

Postoji nekoliko tipova zvučnih datoteka koje mogu da se uključe u veb stranicu. Najpopularniji format zvučne datoteke je WAV. Ekstenzija zvučne datoteke ukazuje na vrstu zvuka. Sledeća tabela prikazuje formate zvučnih datoteka koji se najčešće koriste na internetu.

⁵⁹ 59 Prelom tekst oko tabele HTML.html



Slika 10.53. Definisanje linka ka zvučnoj datoteci⁶⁰

Oznaka ili tag <EMBED> omogućava da se **"ugradi"** zvuk u veb stranicu. Ugrađen zvuk će se reprodukovati direktno na veb stranici. Kada se ugradi zvuk, veb stranica će prikazati specifične kontrole koje omogućavaju korisnicima da pokreću i zaustavljaju reprodukciju zvuka. Upotreba atributa WIDTH i HEIGHT omogućava autoru da definiše veličinu zvučnih kontrola.

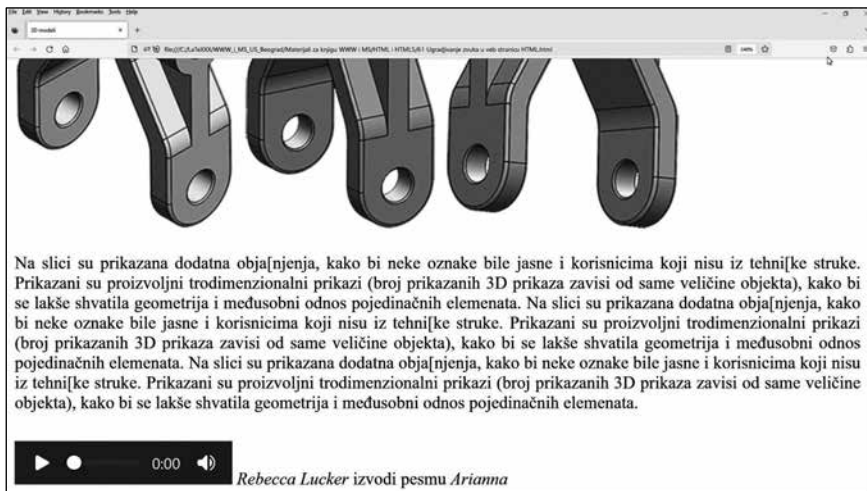
Tabela 10.3 Najčešće korišćeni zvučni formati

Vrsta zvuka	Ekstenzija	Koristi se za
MIDI	.mid	Instrumentalnu muziku
MP3	.mp3	Pesme
RealAudio	.ra	Emitovanje uživo
WAV	.wav	Kratke zvučne snimke

Uobičajeno je da neki veb pretraživači automatski reprodukuju ugrađen zvuk kada korisnik poseti veb stranicu. Trebalo bi iskoristiti atribut AUTOSTART sa vrednošću `false` kako bi se sprečilo automatsko reprodukovanje zvuka. Ako autor želi da se ugrađeni zvuk reprodukuju u kontinuitetu (stalno), trebalo bi da iskoristi atribut LOOP sa vrednošću `true`. Da bi reprodukovali ugrađeni zvuk, veb pretraživači moraju da imaju adekvatan dodatni modul (*plug-in*) instaliran. Dodatni modul je softver koji dodaje i proširuje funkcije veb pretraživaču. Ako korisnik nema ispravan dodatni modul, onda može da se desi da veb pretraživač zatraži od njega da preuzme softver sa interneta i da ga instalira (slika 10.54).

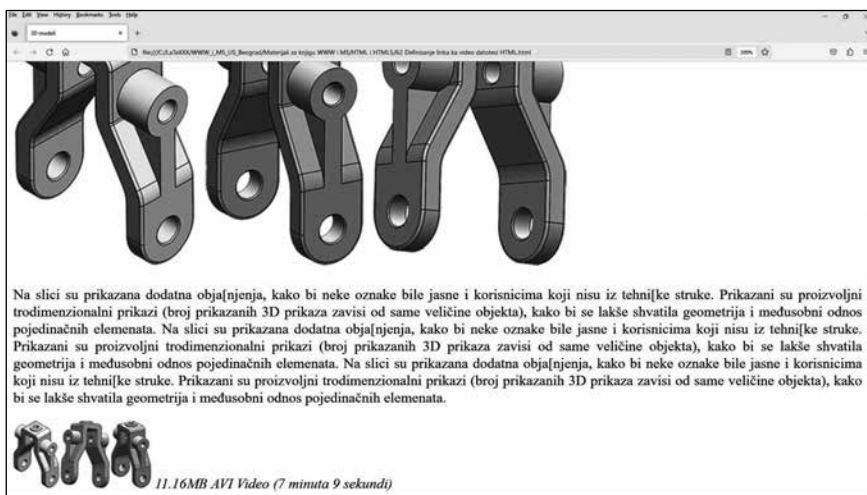
Možda će autor želeći da kreira vezu ili link na veb stranici kako bi korisnici mogli da odoberu **reprodukovanje video materijala**. Video, koji korisnici reprodukuju izborom linka, naziva se *eksterni video*.

⁶⁰ 60 Definisanje linka ka zvučnoj datoteci HTML.html



Slika 10.54. Ugrađivanje zvučne datoteke u fajl⁶¹

Kada se kreira link ka video materijalu trebalo bi obezbediti i kratak opis videa, koji uključuje tip i format videa, veličinu i dužinu vremena koliko reprodukcija materijala traje (slika 10.55). Ovo može da pomogne korisnicima da odluče da li uopšte žele da gledaju video. Takođe bi trebalo proveriti da li je format video materijala odgovarajući za korišćenje na internetu.



Slika 10.55. Definisavanje linka ka video datoteci⁶²

Kada korisnik izabere video link, veb pretraživač će pokušati da reprodukuje video. Ako veb pretraživač ne može da pokrene i reprodukuje video, pretraživač će pokušati da otvori

⁶¹ 61 Ugrađivanje zvuka u veb stranicu HTML.html

⁶² 62 Definisavanje linka ka video datoteci HTML.html

video nekim programom na korisnikovom računar. Ako korisnik nema odgovarajući program, veb pretraživač će omogućiti korisniku da snimi video na svom računar. Postoji nekoliko tipova video datoteka koje mogu da se uključe u veb stranicu. Ekstenzija video datoteke ukazuje na vrstu materijala i prateći format. Sledeća tabela prikazuje formate video datoteka koji se najčešće koriste na internetu.

Tabela 10.4 *Najčešće korišćeni video formati*

Vrsta zvuka	Ekstenzija
AVI	.avi
MPEG	.mpg ili .mpeg
QuickTime	.mov ili .qt

10.7. HTML i forme

Oblici ili forme omogućavaju da se prikupe informacije od korisnika koji posećuju veb stranice. Na primer, forme ili oblici omogućavaju korisnicima da pošalju pitanja i komentare o svojim veb stranicama. Neki oblici (forme) omogućavaju korisnicima da kupuju proizvode i zahtevaju usluge.

Upotrebom taga <FORM> omogućeno je **uspostavljanje obrasca ili forme** na veb stranici. Autor mora da podesi formu pre nego što može da doda informacije u tu istu formu. veb stranica može da sadrži više od jedne forme ili više od jednog obrasca.

Atribut METHOD omogućava da se odredi kako će unete informacija u formu da korisnik prenese preko interneta. Najčešći metod prenosa je post.

Korišćenjem atributa ACTION se precizira gde će veb pretraživač poslati podatke koje je korisnik uneo u polje forme. veb pretraživač može da pošalje informacije CGI scenarijima na svom veb serveru ili na *e-mail* adresu koja je navedena u formi.

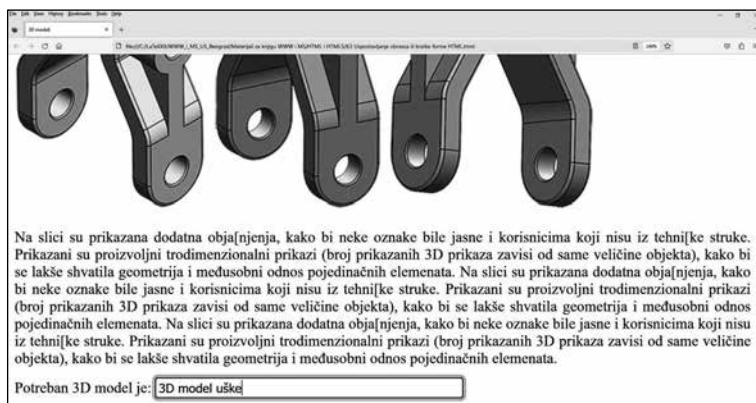
CGI scenario (*Common Gateway Interface Script*, scenario opšteg interfejsa mrežnog prolaza) je program koji veb server staruje u cilju obrade informacije iz forme. Ako je ovo neophodno, onda autor treba da se obrati administratoru veb servera kako bi odredio lokaciju CGI scenarija na svom veb serveru.

Ako autor odluči da veb pretraživač pošalje rezultate unutar forme na *e-mail* adresu, onda bi trebalo da koristi atribut ENCTYPE sa vrednošću *text/plain* kako bi se osigurao da će se informacije korisnika preneti u odgovarajućem formatu. Slanje rezultata forme na *e-mail* adresu je korisno ako veb server ne dozvoljava upotrebu CGI scenarija. Neke veb pretraživači ne podržavaju slanje rezultata forme na *e-mail* adresu.

Okvir za tekst ili polje za unos teksta (*text box*) omogućava korisnicima da unesu kratku liniju teksta. Okvir za tekst se obično koristi za unos imena, adrese i kratkih odgovora, kao što je prikazano na slici 10.56.

Kada se kreira okvir za tekst, trebalo bi iskoristiti atribut NAME da bi se obezbedilo ime za polje za unos teksta. Ime koje se odredi će identifikovati okvir za tekst u rezultatima formulara ili forme. Ime može da sadrži slova i brojeve, ali ne bi trebalo da sadrži razmake

ili znakove interpunkcije. Ako autor želi da uključi prostor u imenu, trebalo bi da koristi donju liniju (_).

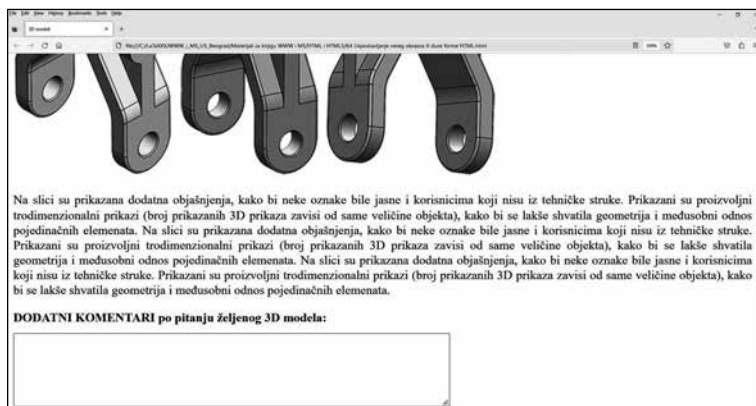


Slika 10.56. Kreiranje obrasca ili kratke forme⁶³

Trebalo bi koristiti atribut `SIZE` da se definiše širina (dužina) okvira za tekst, koji bi trebalo prikazati. Uobičajeno je da je dužina (širina) okvira za tekst oko 20 otkucanih karaktera. Širina ili dužina ne utiče na broj karaktera koje korisnik može da unese u polje za unos teksta.

Da bi se odredio maksimalni broj karaktera koje korisnik može da unese u polje za unos teksta, trebalo bi koristiti atribut `MAXLENGTH`. Ovo može da pomogne da korisnik bude siguran koliko karaktera može da ukuca u polje za unos teksta.

Upotrebom oznake ili taga `<TEXTAREA>` omogućava da se kreira **velika oblast za unos teksta unutar forme**, u kojoj korisnici mogu da unesu nekoliko linija teksta ili nekoliko pasusa. Ovo je korisno za prikupljanje komentara ili pitanja (slika 10.57).



Slika 10.57. Veća oblast za unos teksta⁶⁴

⁶³ 63 Uspostavljanje obrasca ili kratke forme HTML.html

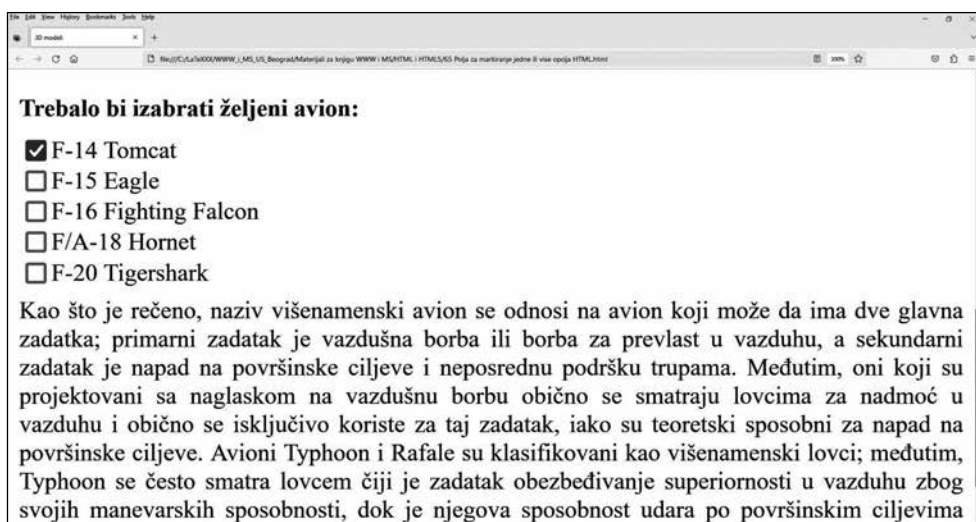
⁶⁴ 64 Uspostavljanje većeg obrasca ili duže forme HTML.html

Atribut NAME omogućava da se obezbedi ime za veliku oblast teksta koja se kreira. Ime će identifikovati tekstalno područje u rezultatima formulara.

Atributi ROWS i COLS omogućavaju da odredi visina i širina velikog prostora za unos teksta. Pošto korisnici neće moći da promene veličinu ovog prostora za unos teksta na svojoj veb stranici, autor bi trebalo da proveri da je napravljeno tekstualno područje dovoljno veliko da se prikaže celokupni korisnikov tekst i da će to područje moći da se prikaže u potpunosti na monitoru.

Korišćenjem WRAP atributa određuje se maksimalna "dužina" linije teksta, pre nego što se tekst prelomi i pređe u novi red ili liniju. Vrednost soft će omogućiti prelamanje velike količine teksta, ali neće da prelomi tekst u rezultatima formulara. Vrednost hard vrši prelamanje i velike količine teksta i teksta unutar formulara. Trebalo bi koristiti vrednost off kada se ne želi da se tekst, koji korisnik ukucava, automatski prenosi u naredni red. Iako je atribut WRAP podržan od strane većine veb pretraživača, nije deo HTML standarda.

Uključivanje **polja za potvrdu** (*check box*) unutar formi ili obrazaca omogućava se korisnicima da mogu da **izaberu ili markiraju jednu ili više opcija** (slika 10.58).



Slika 10.58. Markiranje jedne ili više opcija⁶⁵

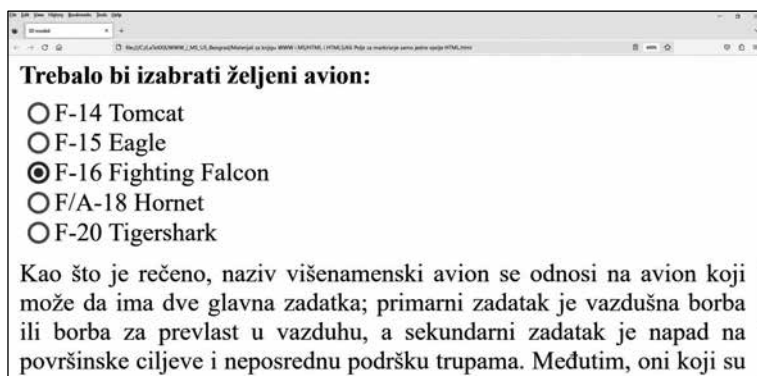
Atribut NAME omogućava da se obezbedi ime za svaku kreiranu grupu polja za potvrdu. Navedena imena će prepoznati svaku grupu polja za potvrdu u rezultatima formulara.

Atribut VALUE omogućava da se navede ime svakog polja za potvrdu unutar grupe. Ime koje se odredi za polje će se pojaviti u rezultatima formulara, ako korisnik izabere to polje za potvrdu. Trebalo bi koristiti imena koja mogu lako da se identifikuju, kao što je tekst koji se pojavljuje pored polja u veb stranici. Ako se ne koristi atribut VALUE, rezultati formulara prikazati ime on za svako izabrano polje za potvrdu, tako da se neće znati koje je polje stvarno markirano.

⁶⁵ 65 Polja za markiranje jedne ili više opcija HTML.html

Imena specificirana upotrebom atributa NAME i VALUE mogu da sadrže slova i brojeve, ali ne bi trebalo da sadrži razmake ili znakove interpunkcije. Ako autor želi da uključi prostor u imenu, trebalo bi da koristi donju liniju (_). Autor će možda želeći da koristi atribut CHECKED, kako bi se pojavilo markirano (izabrano) jedno ili više polja za potvrdu kada se aktivira forma. Ovo je korisno kada očekujete da će većina korisnika izabrati određeno polje u pripadajućem formularu ili obrascu.

Uključivanjem **radio dugmadi** ili **dugmadi za izbor opcije** (*radio buttons*) u obrazac je poželjno kada se želi da korisnici **izaberu samo jednu opciju**, od nekoliko ponuđenih (slika 10.59).



Slika 10.59. Markiranje jedne opcije⁶⁶

Atribut NAME omogućava da se obezbedi ime za svaku kreiranu grupu radio dugmadi. Navedena imena će prepoznati svaku grupu radio dugmadi u rezultatima formulara.

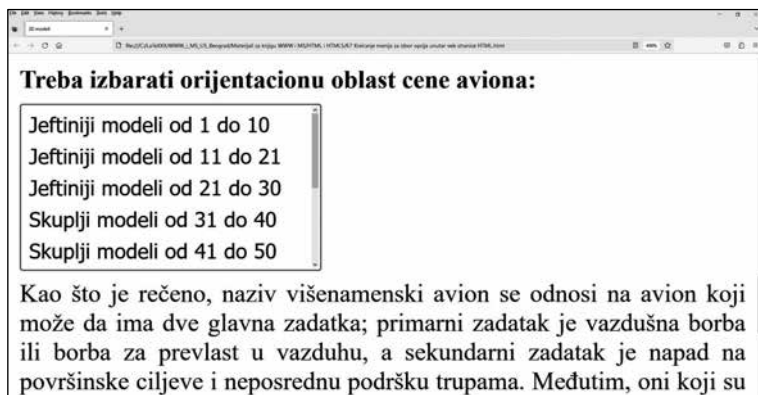
Atribut VALUE omogućava da se navede ime svakog radio dugmeta unutar grupe. Ime koje se odredi za dugme će se pojaviti u rezultatima formulara, ako korisnik izabere to radio dugme polje za potvrdu. Trebalo bi koristiti imena koja mogu lako da se identifikuju, kao što je tekst koji se pojavljuje pored polja u veb stranici. Ako se ne koristi atribut VALUE, rezultati formulara će prikazati naziv on za izabrano radio dugme, tako da se neće znati koje je dugme stvarno markirano.

Imena specificirana upotrebom atributa NAME i VALUE mogu da sadrže slova i brojeve, ali ne bi trebalo da sadrži razmake ili znakove interpunkcije. Ako autor želi da uključi prostor u imenu, trebalo bi da koristi donju liniju (_).

Ako autor želi da obezbedi korisnicima listu opcija za biranje, trebalo bi da **doda meni** u formu ili formular. Meniji se obično koristi za prikazivanje liste starosnih grupa, država i proizvoda (slika 10.60).

Atribut NAME omogućava da se obezbedi ime za svaki kreiran meni. Navedena imena će prepoznati svaki meni u rezultatima formulara. Uobičajeno je da se meni pojavljuje u obrascu kao padajući meni sa vidljivom prvom opcijom. Ako autor želi da se vidi više od jedne opcije, onda bi trebalo da koristi atribut SIZE kako bi definisao broj vidljivih opcija unutar pripadajućeg padajućeg menija.

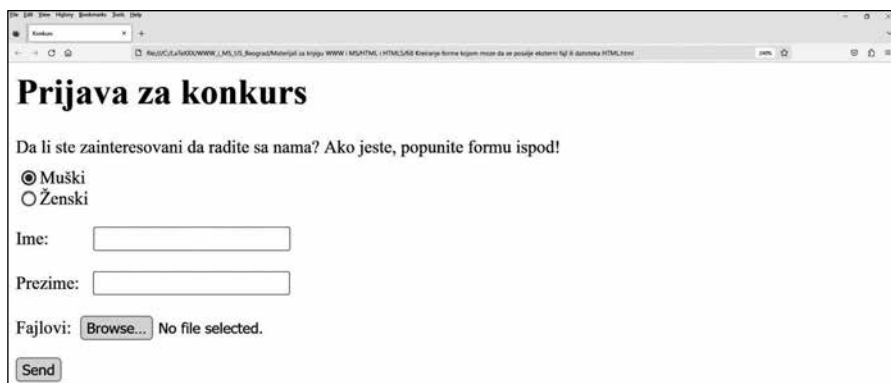
⁶⁶ 66 Polje za markiranje samo jedne opcije HTML.html



Slika 10.60. Ubacivanje menija za izbor opcija⁶⁷

Oznaka ili tag `<OPTION>` omogućava da se odrede informacije za jednu opciju menija. Trebalo bi koristiti atribut `VALUE` za definisanje imena opcije. Ime će se pojaviti u rezultatima formulara ako korisnik izabere tu opciju. Preporuka je da se koriste imena koja mogu lako da se identifikuju, kao što je tekst koji se pojavljuje u meniju na veb stranici.

Forma, obrazac ili formular može da sadrži prostor koji omogućava korisnicima da **pošalju** (*upload*) **fajl**. Omogućavajući korisnicima da pošalju fajlove je korisno za prikupljanje informacija koje su najbolje prikazane u posebnom fajlu, kao što je rezime ili narudžbenica. Fajlovi koje korisnici šalju se obično čuvaju na veb serveru (slika 10.61).



Slika 10.61. Slanje eksternog fajla "kroz" formu⁶⁸

U tagu `<FORM>` trebalo bi koristiti atribut `ENCTYPE` sa vrednošću `multipart/form-data` kako bi se osiguralo da će korisnici svoje fajlove poslati u odgovarajućem formatu.

Korišćenjem atributa `NAME` unutar taga `<INPUT>` omogućava se definisanje imena datoteke koju korisnici šalju. Ime će identifikovati datoteku na veb serveru.

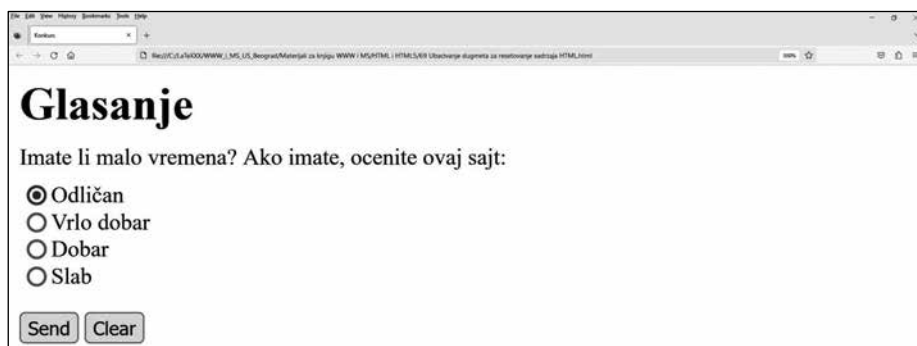
⁶⁷ 67 Kreiranje menija za izbor opcija unutar veb stranice HTML.html

⁶⁸ 68 Kreiranje forme kojom može da se pošalje eksterni fajl ili datoteka HTML.html

Kada je omogućeno korisnicima da šalju fajlove, veb stranica će prikazati polje u kojem korisnici mogu da unesu lokaciju i ime datoteke koju žele da pošalju. Atribut `SIZE` omogućava da se odredi širina ili dužina polja za unos ovih podataka. Povećanje širine ovog polja mogu pomoći da se osigura da će korisnici moći da vide celu putanju do datoteke. veb stranica će prikazati i dugme `Browse` koje će da pomogne korisnicima da pronađu datoteke koje žele da pošalju.

Uključujući dugme za slanje (`Submit`) unutar obrasca, autor omogućava korisnicima da šalju informacije koje su uneli u obrazac ili formular.

Upotrebom atributa `VALUE` definiše se tekst koji bi trebalo da se pojavu na dugmetu za slanje. Ako se ne koristi atribut `VALUE`, onda će se na dugmetu pojaviti podrazumevani tekst kao *Submit Query* ili *Submit*.



Slika 10.62. *Resetovanje trenutnog sadržaja*⁶⁹

Kreiranje **dugmeta za resetovanje** (`Clear` ili `Reset`) unutar obrasca, omogućeno je korisnicima da obrišu informacije koje su uneli u formular ili obrazac. Kada korisnik klikne na dugme za resetovanje, korisnikov veb pretraživač će resetovati sve elemente obrasca na prvobitne postavke. Ovo omogućava korisnicima da počnu ispočetka, ako su napravili grešku prilikom popunjavanja formulara.

Upotrebom atributa `VALUE` definiše se tekst koji bi trebalo da se pojavu na dugmetu za resetovanje. Ako se ne koristi atribut `VALUE`, onda će se na dugmetu pojaviti podrazumevani tekst kao `Reset`.

Kada korisnik klikne na dugme za slanje (`Submit`) ili resetovanje (`Reset`), njegov veb pretraživač će poslati ili obrisati informacije u svim elementima obrasca ili formulara. Korisnik neće biti u stanju da poništi (*undo*) ili preokrene operacije slanja ili resetovanja.

Grafičko dugme za slanje (`Submit`) predstavlja sliku koja omogućava korisnicima da unesu informacije i da dostave rezultate iz formulara u isto vreme.

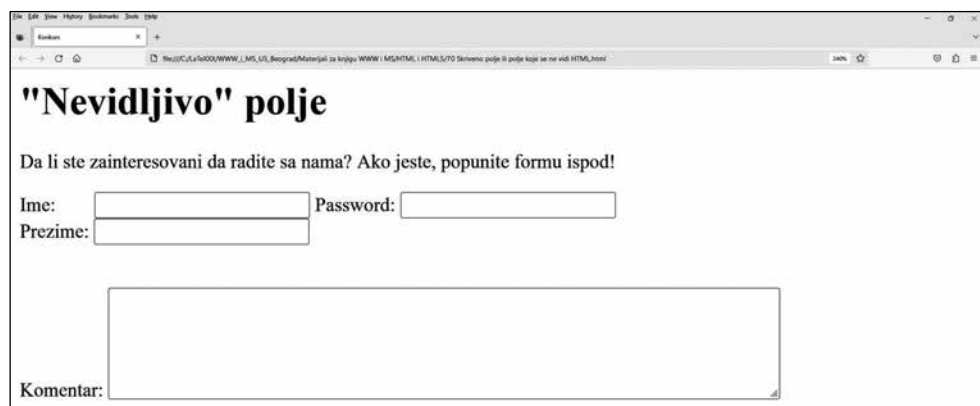
Kada korisnik klikne na grafičko dugme za slanje, horizontalna i vertikalna koordinate miša se šalju zajedno sa ostalim informacijama u formularu. Ovo je korisno kada područja na slici predstavljaju različite opcije, kao što su lokacija na planu karte. Trebalo bi obezbediti i tekst koji upozorava korisnike da će, klikom na sliku, dostaviti sve svoje informacije.

⁶⁹ 69 Ubacivanje dugmeta za resetovanje sadržaja `HTML.html`

Atribut SRC unutar taga <INPUT> omogućava da se navede lokacija i ime GIF ili JPEG slike koju bi trebalo koristiti.

Atribut NAME omogućava da se obezbedi ime za grafičko dugme za slanje. Ime će identifikovati horizontalne (x) i vertikalne (y) koordinate miša u rezultatima formulara. Ove koordinate će se meriti u pikselima od gornjeg levog ugla slike.

Sakriveno polje ili **polje koje se ne vidi** (*hidden field*) element formulara koji se neće pojaviti na veb stranici, ali će se pojaviti u rezultatima formulara. Sakrivena polja su korisna za uključivanje informacija, kao što su datum i vreme kada je kreirana sama forma ili određeni obrazac.



Slika 10.63. "Nevidljivo" polje⁷⁰

Kada se pravi sakriveno polje, trebalo bi koristiti atribut NAME da obezbedi naziv za sakriveno polja i atribut VALUE da odredi informacije koje bi sakriveno polje trebalo da sadrži. Ako u veb stranici postoje višestruki obrasci, sakrivena polja mogu pomoći prilikom identifikacije informacija koje podnose korisnici. Na primer, sakriveno polje može da se iskoristi da se svakom obrascu dodeli drugo ime ili broj. Kada se vide rezultati formulara, ime ili broj će identifikovati formular koji je korisnik popunjavao.

Ako obrazac ili formular sadrži mnogo elemenata, možda će autor želeći da koristi tag <FIELDSET> da **organizuje elemente** u grupe. veb pretraživači, koji podržavaju oznaku <FIELDSET>, vizuelno će odvojiti elemente i da ih uključe u grupu ili grupe. Na primer, pretraživač prikazuje tanku, graničnu jednobojnu liniju oko elemenata (slika 10.64).

Organizovanje elemenata formulara može pomoći korisnicima da razumeju formu, koja je podeljena u logične celine. Na primer, možda bi trebalo stvoriti jednu grupu za tražene informacije i drugu grupu za opcione informacije.

Oznaka ili tag <LEGEND> omogućava da se doda naslov grupisanim elementima. Ako se autor odluči da uključi naslov, onda atribut ALIGN omogućava da se definiše mesto gde će se naslov pojaviti – levo, desno, iznad ili ispod grupe elemenata. Poznato je kako se ovaj atribut upotrebljava.

⁷⁰ 70 Skriveno polje ili polje koje se ne vidi HTML.html

Organizovanje elemenata forme

Da li ste zainteresovani da radite sa nama? Ako jeste, popunite formu ispod!

POTREBNE INFORMACIJE

Ime:

Prezime:

Password:

Komentar:

Trebalo bi izabrati određenu opciju:

☒ Početnik sa završenim fakultetom

☐ Početnik sa završenim masterom

☐ Sa malim radnim iskustvom

☐ Sa većim radnim iskustvom

☐ Sa iskustvom u struci za koju je raspisan konkurs

Slika 10.64. Organizovanje sadržaja ili elemenata forme⁷¹

Korišćenjem oznake ili taga <LABEL> omogućeno je da se u formular ubacuje **tekst kao labela** (oznaka) za određene elemente. Ovo nalaže veb pretraživaču da interpretira tekst kao deo elementa obrasca.

Da bi se "etiketirao" ili "obeležio" element formulara, trebalo bi koristiti atribut ID u cilju identifikacije elementa kojibi trebalo obeležiti. Zatim bi trebalo iskoristiti atribut FOR unutar taga <LABEL> za identifikovanje teksta za labele.

Trebalo bi izabrati željeni avion:

☐ F-14 Tomcat

☐ F-15 Eagle

☒ F-16 Fighting Falcon

☐ F/A-18 Hornet

☒ F-20 Tigershark

Kao što je rečeno, naziv višenamenski avion se odnosi na avion koji može da ima dve glavna zadatka; primarni zadatak je vazдушna borba ili borba za prevlast u vazduhu, a sekundarni zadatak je napad na površinske ciljeve i neposrednu podršku trupama. Međutim, oni koji su projektovani sa naglaskom na vazдушnu borbu obično se smatraju lovcima za nadmoć u vazduhu i obično se isključivo koriste za taj zadatak, iako su teoretski sposobni za napad na površinske ciljeve. Avioni Typhoon i Rafale su klasifikovani kao višenamenski lovci; međutim, Typhoon se često smatra lovcem čiji je zadatak obezbeđivanje superiornosti u vazduhu zbog

Slika 10.65. Identifikacija elemenata u listi⁷²

Obeležavanje elemenata obrasca može pomoći korisnicima da brzo dođu do informacija u formularu. Na primer, kada korisnik klikne na labelu pored polja za potvrdu na veb

⁷¹ 71 Organizovanje elemenata forme HTML.html

⁷² 72 Identifikacija elemenata u listi pomoću ID identifikatora HTML.html

stranici, polje za potvrdu će biti izabrano ili markirano. Slično tome, korisnici mogu da kliknu na labelu za polje za unos teksta kako bi se postavio kursor miša u polje za unos teksta.

Iako je tag <LABEL> deo HTML standarda, trenutno nije podržan od strane mnogih veb pretraživača.

Korisnici mogu da se **kreću kroz elemente** formulara korišćenjem Tab tastera. Uobičajeno je da se pritiskom na taster Tab korisnik kreće po elementima kako se pojavljuju u HTML kodu. Promena ovog redosled može pomoći korisnicima da popune formular u logičkom redosledu.

Da bi se promenio ovaj redosled, trebalo bi koristiti atribut TABINDEX unutar tagova za svaki element formulara. Da bi se elementi uključili u Tab redosled, trebalo bi elementima dodeliti TABINDEX vrednosti da budu između 1 i 32.767. Da bi se element isključio iz Tab reda, trebalo bi elementu dodeliti negativnu TABINDEX vrednost (na primer, TABINDEX="-1").

The screenshot shows a web browser window with the title 'Promena pristupa poljima pomoću tastera TAB'. The form contains the following elements:

- A heading: **Promena pristupa poljima pomoću tastera TAB**
- A subheading: **Da li ste zainteresovani da radite sa nama? Ako jeste, popunite formu ispod!**
- Input fields for 'Ime:' and 'Prezime:'.
- A password field labeled 'Password:'.
- A large text area labeled 'Komentar:'.

The browser address bar shows the URL: `http://www.mli.rs/Blog/Materijali/za_knjigu_WWW/Mli_HTML/HTML5/73_Promena_pristupa_poljima_u_formi_pomocu_TAB_tastera_HTML.html`.

Slika 10.66. *Pristup poljima unutar forme TAB tasterom sa tastature⁷³*

Kada korisnici krene kroz formular pritiskom na taster Tab, najpre će se markirati element sa dodeljenom najnižom vrednošću, a na kraju će biti element sa najvećom vrednošću. Ako dva elementa imaju istu vrednost, onda će element koji se prvi pojavljuje u HTML kodu imati prednost.

Atribut TABINDEX može da se koristi sa elementima izvan formulara, kao što su slike (), linkovi (<A>) i mapirane oblasti slika (<AREA>).

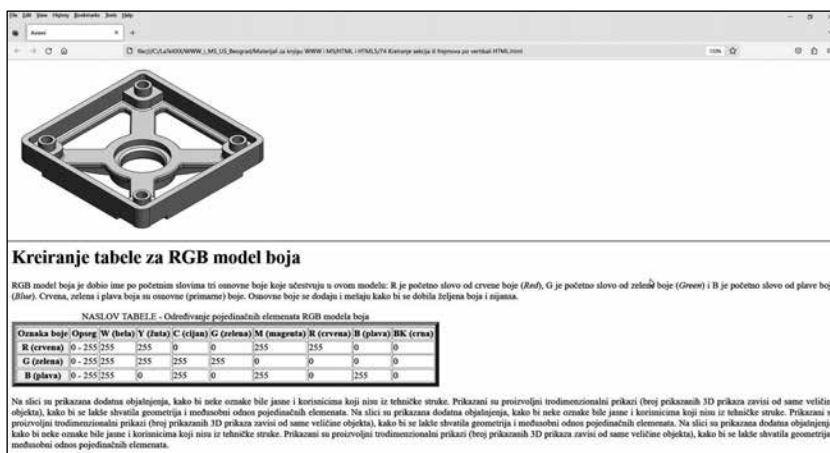
10.8. HTML i sekcije (frejmovi)

U ovom delu biće reči o frejmovima ili sekcijama pomoću kojih je omogućeno da se podeli tekući prozor veb pretraživača u određene oblasti, o njihovom kreiranju i o njihovim pratećim elementima i osobinama. Ako je veb sajt veliki i složen, onda primena frejmova

⁷³ 73 Promena pristupa poljima u formi pomocu TAB tastera HTML.html

ili sekcija nudi mnoge prednosti koje mogu da pomognu korisnicima da bolje razumeju strukturu sajta i kretanje kroz veb stranice. Frejmovi omogućavaju prikazivanje više veb stranica u jednom prozoru veb pretraživača. To omogućava da se odjednom prikažu različite informacije na ekranu. Upotreba frejmova, takođe, omogućava da se zadrži podaci, kao što je logo firme, na ekranu u svakom trenutku, bez potrebe da se dodaju informacije na svaku veb stranicu unutar veb sajta. Mnogi autori veb stranica smatraju frejmove korisnim elementima za prikazivanje navigacije unutar sajta. Ovo omogućava korisnicima da vide strukturu veb sajta u svakom trenutku i omogućava im lak pristup informacijama od interesa.

Kao i sve u životu, moraju da postoje i određeni nedostaci kada su frejmovi u pitanju (slika 10.67). Pošto frejmovi "dele" prozor veb pretraživača u sekcije, onda je prostor za gledanje svake veb stranice smanjen. Veb stranice dizajnirane da budu prikazane u celokupnom prozoru veb pretraživača ne mogu dobro da se uklope u frejmove (sekcije). Prilikom kreiranja frejmova, može da se da bude teško da se odredi najbolja veličina za svaki frejm. Teško je predvideti kako će se frejmovi prikazivati na korisničkim računarima, jer korisnici na različite načine određuju količinu informacija koja se pojavljuje u frejmovima. Ljudi koji koriste manje rezolucije ekrana neće videti toliko informacija u svakom kadru. Takođe, trebalo bi imati u vidu i da svi veb pretraživači ne podržavaju frejmove.



Slika 10.67. Kreiranje sekcija po vertikali⁷⁴

Frejmovi omogućavaju da se podeli prozor veb pretraživača na **sekcije**. Svaka sekcija će prikazati različite veb stranice unutar tekućeg veb sajta.

Frejmovi su korisni kada autor želi da zadrži informacije na ekranu u svakom trenutku. Na primer, postaviti sadržaj ili navigacione veze u frejm, kako bi se pomoglo korisnicima po pitanju kretanja kroz veb stranice i pronalaženja informacija od interesa.

Da bi se kreirali frejmovi, trebalo bi podesiti veb stranicu definisanje strukture frejmova upotrebom tagova ili oznaka <FRAMESET> i <FRAME>. Oznaka ili tag <FRAMESET> omogućava da se kreira skup frejmova (*framest*) koji deli prozor veb pretraživača u redove

⁷⁴ 74 Kreiranje sekcija ili frejmova po vertikali HTML.html

i kolone. Trebalo bi navesti visinu ili širinu za svaki red ili kolonu koristeći procentualnu vrednost prozora veb pretraživača, broj piksela ili zvezdicu (*). Kada se koristi zvezdica, veličina frejma će zavisiti od veličine drugih frejmova. Na primer, ako jedan frejm koristi 70% prozora, frejm sa zvezdicom će koristiti 30% prozora.



Slika 10.68. Kreiranje sekcija po horizontali⁷⁵

Upotrebom taga <FRAME> omogućeno je specificiranje informacija za svaki frejm., kao što su ime frejma i veb stranice koja bi trebalo da se pojavi u tom frejmu.

Upotrebom atributa TARGET sa tagom <A> omogućeno je kreiranje linka koji korisnici mogu da odaberu i da prikažu veb stranicu u drugom frejmu. **Kreiranje veze ka frejmovima ili sekcijama** može da olakša korisnicima kretanje kroz veb sajt (slika 10.69). Na primer, mogu da se kreiraju navigacioni linkovi u jednom frejmu koji će se otvoriti u drugom frejmu. To omogućava da se navigacioni linkovi prikazuju na ekranu, dok korisnici pregledaju stranice.



Slika 10.69. Kreiranje linkova ka sekcijama⁷⁶

⁷⁵ 75 Kreiranje sekcija ili frejmova po horizontali HTML.html

⁷⁶ 76 Kreiranje veze ka frejmovima ili sekcijama HTML.html

Kada se kreira veza sa frejmom, trebalo bi definisati ime pomoću atributa TARGET. Ako se ne koristi atribut TARGET da se navede ime jednog frejma, veza će se otvoriti u frejmu koji sadrži link.

Uobičajeno je da će veb pretraživači prikazati **trake za pomeranje sadržaja prozora** ili **trake za skrolovanje** (*scroll bar*) frejma samo ako veb stranica sadrži previše informacija koje je nemoguće uklopiti u postojeći frejm. Korišćenje atributa SCROLLING sa vrednostima no ili yes omogućava da se trake za skrolovanje ne prikazuju ili prikazuju (slika 10.70).



Slika 10.70. Prikazivanje traka za pomeranje sadržaja prozora⁷⁷

Ako autor ne želi da prikazete traku za skrolovanje frejma, onda bi trebalo definisati vrednost no. Skrivanje ili ne prikazivanje trake za skrolovanje se korisno zbog smanjenja "gužve" u malom frejmu. Ako autor želi da se tačno prikazuje traka za skrolovanje, onda bi trebalo definisati vrednost yes. Ovo je korisno, jer trake za skrolovanje poboljšavaju izgled frejma.

Upotreba atributa NORESIZE **sprečava korisnika da promeni veličine frejmova**. Ovo je korisno kada autor ne želi da menja izgled i raspored frejmova. Ako se ne spreče korisnici da menjaju veličinu frejmova, oni mogu da prevuku granični okvir na novu lokaciju kako bi se promenila veličina frejma ili sekcije. Korisnici, obično, žele da promene veličinu frejma kada hoće da prikažu više informacija u frejmu.

Kada se "odradi" sprečavanje korisnika da promena veličine frejmova, korisnici, takođe, neće moći da promene veličinu i frejmova koji dele iste granične okvire. Ova osobina ukida potrebu da se koristi atribut NORESIZE za sve frejmove.

Ako se već ukida mogućnost korisniku da promeni veličine frejmova, onda bi trebalo da je autor siguran u predložene veličine frejmova, kako bi stale sve relevantne informacije. Trebalo bi imati na umu da će korisnici koji vide veb stranice na nižim rezolucijama videti manje informacija u svakom frejmu.

⁷⁷ 77 Prikazivanje i upotreba traka za pomeranje sadržaja prozora HTML.html

Postoji nekoliko opcija na raspolaganju za promenu izgleda graničnih linija između frejmova ili sekcija.

Korišćenjem atributa BORDER sa tagom <FRAMESET> je omogućeno da se promeni "debljina" (širina) graničnih linija frejmova. Uobičajeno je da većina veb pretraživača prikazuje granične linije frejmova širine ("debljine") oko 6 piksela.



Slika 10.71. Granične linije sekcija^{78 79}

Kad veb pretraživači, obično, prikazuju granične linije frejmova u sivoj boji. Korišćenjem atributa BORDERCOLOR sa tagom <FRAMESET> je omogućeno da se promeni boja graničnih linija frejma. Ovo je korisno kada autor želi da koordinira boju graničnih linija sa bojom pozadine veb stranica u koje su prikazane u frejmovima.

Iako su atributi BORDER i BORDERCOLOR podržani od strane većine veb pretraživača, oni nisu deo HTML standarda za frejmove ili sekcije.

^{78 78} Granične linije sekcija ili frejmova HTML.html

^{79 79} Nema granične linije sekcija ili frejmova HTML.html

Da bi se sakrile sve granične linije između frejmova, potrebno je koristiti `FRAMEBORDER` sa tagom `<FRAMESET>`. Ovo je korisno kada autor želi da se sadržaj frejmova pojavi kao jedna veb stranica. Nakon što se sakriju granične linije frejmova, ostaje prazan prostor između frejmova. Da bi se potpuno uklonio prostor, trebalo bi postaviti atribut `BORDER` na vrednost 0 piksela.

Atribut FRAMEBORDER sa oznakom ili tagom <FRAMESET> je podržan od strane većine veb pretraživača, iako nije deo HTML standarda.

Promena margina frejma omogućava da se podesi veličina prostora između sadržaja frejma i same ivice frejma (slika 10.72 gore). Ovo može da poboljša izgled frejma. Uobičajeno je da većina veb pretraživača prikazuje margine frejmova veličine oko 10 piksela. Upotrebom atributa `MARGINWIDTH` i `MARGINHEIGHT` mogu da se promene veličine margina samog frejma. Atribut `MARGINWIDTH` omogućava da se odredi nova veličina leve i desne margine, u pikselima.



Slika 10.72. *Promena margina sekcija i "plutajuće" sekcije*^{80 81}

⁸⁰ 80 Promena margina sekcije ili frejma HTML.html

⁸¹ 81 Kreiranje plutajućeg frejma ili sekcije HTML.html

Atribut `MARGINHEIGHT` omogućava da se odredi nova veličina gornje i donje margine, u pikselima. Ako autor želi da se sadržaj frejma direktno pojavi uz samu ivicu frejma, onda bi trebalo podesiti margine frejma na vrednost od 0 piksela. Samo neki veb pretraživači mogu da prikažu margine čija je vrednost 0 piksela.

Kada se menjaju margine frejmova, trebalo bi proveriti da li su navedene vrednosti margina i po visini i po dužini (širini). Ako se navede samo jedna veličina margina (ili visina ili širina), veb pretraživač može automatski da promeni i drugu veličinu margine na najmanju moguću veličinu koju pretraživač podržava.

Upotrebom oznake ili taga `<IFRAME>` omogućeno je **kreiranje "plutajućeg" frejma** unutar veb stranice (slika 10.72 dole). "Plutajući" frejm ili "plutajuća" sekcija je frejm koji se pojavljuje u okviru sadržaja veb stranice.

Upotrebom atributa `NAME` se definiše i obezbeđuje ime za "plutajući" frejm. Imenovanje ovakve vrste frejma omogućava da se kasnije stvore veze ili linkovi koji će otvoriti još jednu veb stranicu u okviru "plutajućeg" frejma.

Upotrebom atributa `WIDTH` i `HEIGHT` je omogućeno da se definišu visina i širina "plutajućeg" frejma. Trebalo bi se uveriti da je navedena veličina dovoljno velika da se pravilno prikazuje sadržaj frejma. Korisnici neće moći da promene veličinu "plutajućeg" frejma na veb stranici.

10.9. Šta je doneo HTML5?

HTML5 je najnoviji standard za HTML. Ranije verzije HTML su se pojavile još 1999. godine, a internet se, od tada, značajno promenio. HTML5 je dizajniran da zameni HTML 4, XHTML i HTML DOM Level 2. Trebalo bi napomenuti da je specijalno dizajniran da pruži bogat sadržaj bez potrebe za dodacima (*plugins*). Trenutna verzija "donosi" animaciju u grafiku, muziku u filmove, a može da se koristi i za izgradnju komplikovanih veb aplikacija. HTML5 je dizajniran da radi bez obzira da li se koristi PC, tablet, smart telefon ili smart TV. Za bolje rukovanje unutar interneta, HTML5 je, takođe, uključio nove elemente za crtanje grafike, prikazivanje medijskog sadržaja, bolju strukturu stranice i bolje forme za rukovanje. Pored toga, uključio je i nekoliko novih API-a (*Application Programming Interface*).

Ispod je prikazana sintaksa jednostavnog HTML5 dokument, sa minimumom potrebnih tagova ili oznaka:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Naslov dokumenta ...</title>
  </head>
  <body>
    Sadržaj dokumenta ...
  </body>
</html>
```

10.10. HTML5 i semantika

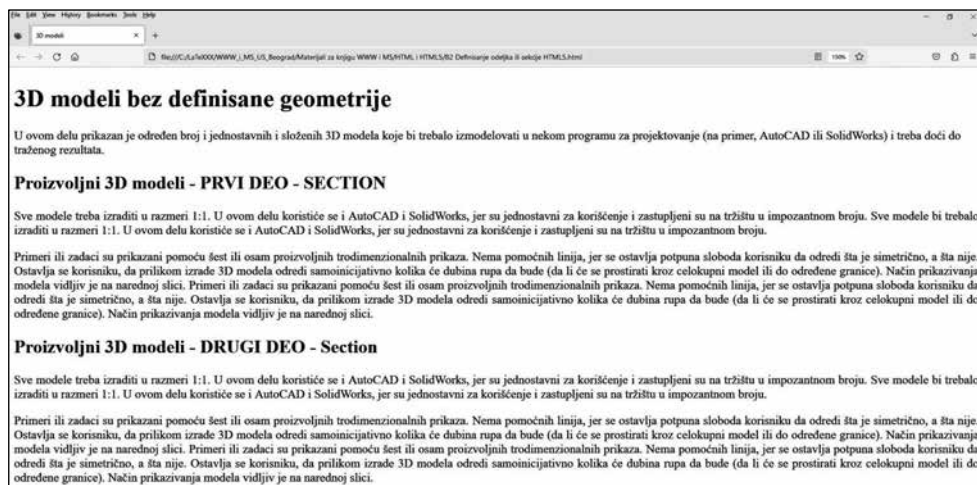
Semantički elemenat jasno opisuje svoje značenje kako pretraživaču, tako i programeru. **Nesemantički elementi**, kao što su `<div>` ili `<span>` ne govore ništa o njihovom sadržaju. **Semantički elementi**, kao što su `<form>`, `<table>` ili `<img>` jasno definišu njihov sadržaj.

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

Slika 10.73. Struktura veb stranice u HTML5

HTML5 je ponudio nove semantičke elemente koji jasno definišu različite delove veb stranice i ovde će biti pomenuti najvažniji elementi.

Element `<section>` definiše odeljak ili sekciju unutar dokumenta (slika 10.73). Kompletna sintaksa je prikazana u fajlu koji je naveden u naslovu slike u fusnoti, a prikaz ekrana na slici 10.74.

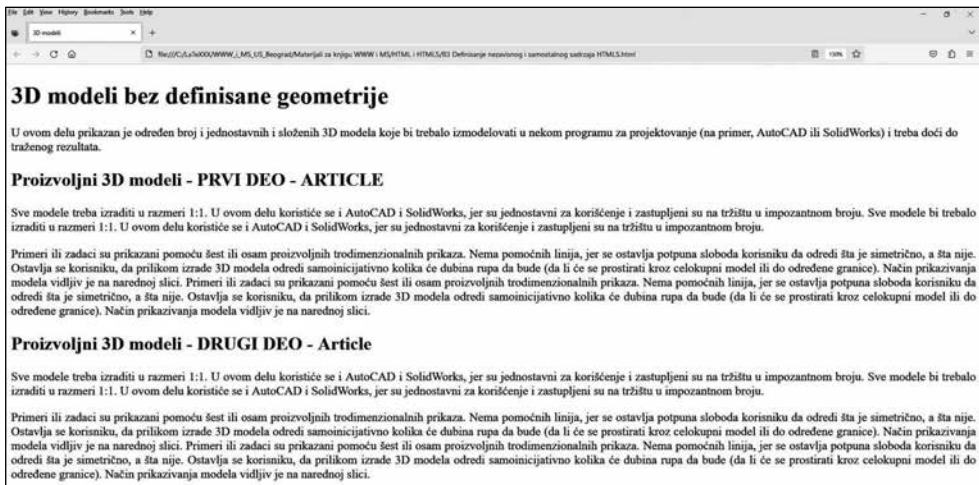


Slika 10.74. Definisanje i kreiranje sekcije⁸²

Element `<article>` određuje nezavisan i samostalan sadržaj. Primena ovog elementa trebalo bi da ima smisla sama po sebi i trebalo bi da omogući distribuiranje njegovog

⁸² 82 Definisanje odeljka ili sekcije HTML5.html

sadržaja nezavisno od ostatka sajta. Element `<article>` može da se koristi za forume, blogove, vesti i komentare.



Slika 10.75. Definisanje nezavisnog sadržaja⁸³

Element `<nav>` definiše **skup navigacionih linkova**. Ovaj element je namenjen za velike blokove navigacionih linkova, mada ne bi trebalo da svi linkovi u dokumentu budu unutar elementa `<nav>`.



Slika 10.76. Definisanje i pozicioniranje navigacionih linkova⁸⁴

Element `<aside>` definiše neki sadržaj pored postavljenog sadržaja (slika 10.77). Reč je o bočnom stupcu (*sidebar*, blok teksta smešten sa strane glavnog teksta dokumenta, često odvojen linijom ili nekim drugim grafičkim elementom). Sadržaj elementa `<aside>` bi trebalo da se odnosi na okolni sadržaj.

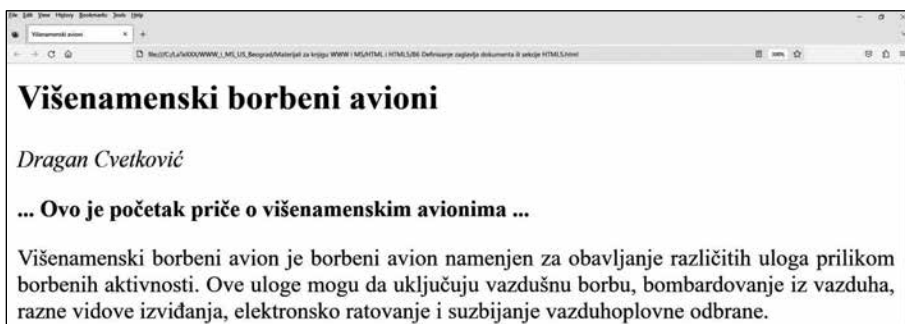
Element `<header>` određuje **zaglavlje dokumenta ili sekcije**. Ovaj element bi trebalo da se koristi kao kontejner za uvodni sadržaj (slika 10.78). Trebalo bi napomenuti da u jednom dokumentu može da se nalazi nekoliko `<header>` elemenata i da u tom smislu nema nikakvih ograničenja.

⁸³ 83 Definisanje nezavisnog i samostalnog sadržaja HTML5.html

⁸⁴ 84 Definisanje skupa navigacionih linkova HTML5.html

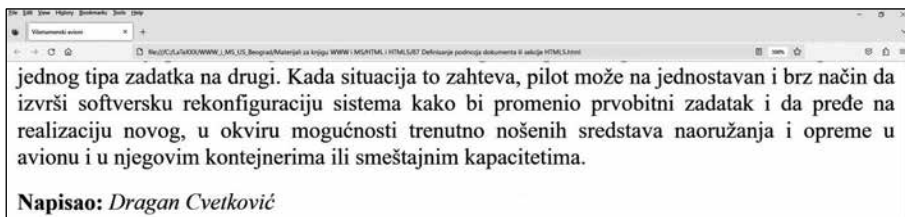


Slika 10.77. Definisanje proizvoljnog sadržaja⁸⁵



Slika 10.78. Definisanje zaglavlja⁸⁶

Element `<footer>` određuje **podnožje dokumenta ili sekcije**. Ovaj element bi trebalo da sadrži informacije o sadržaju elementa. Podnožje, obično, sadrži informacije autora o dokumentu, informacije o autorskim pravima, linkove sa uslovima korišćenja, kontakt informacije itd. U dokumentu može da postoji nekoliko `<footer>` elemenata.

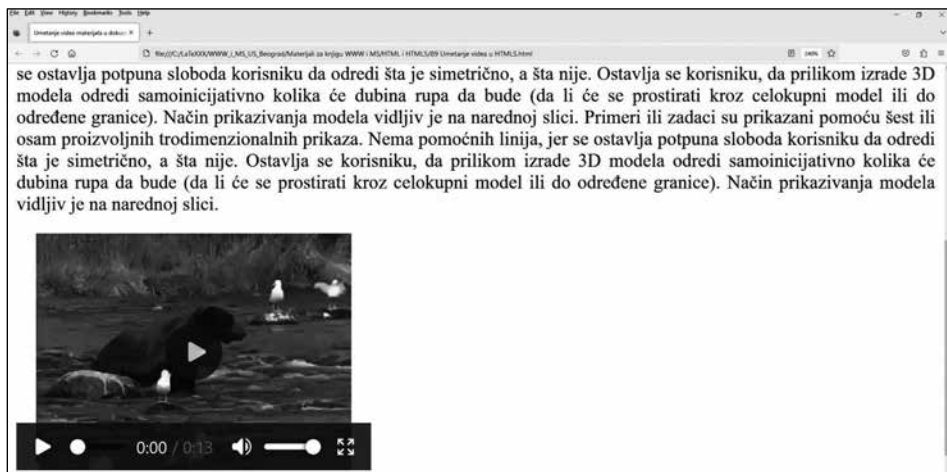


Slika 10.79. Definisanje podnožja⁸⁷

⁸⁵ 85 Definisanje proizvoljnog sadržaja pored postojećeg HTML5.html

⁸⁶ 86 Definisanje zaglavlja dokumenta ili sekcije HTML5.html

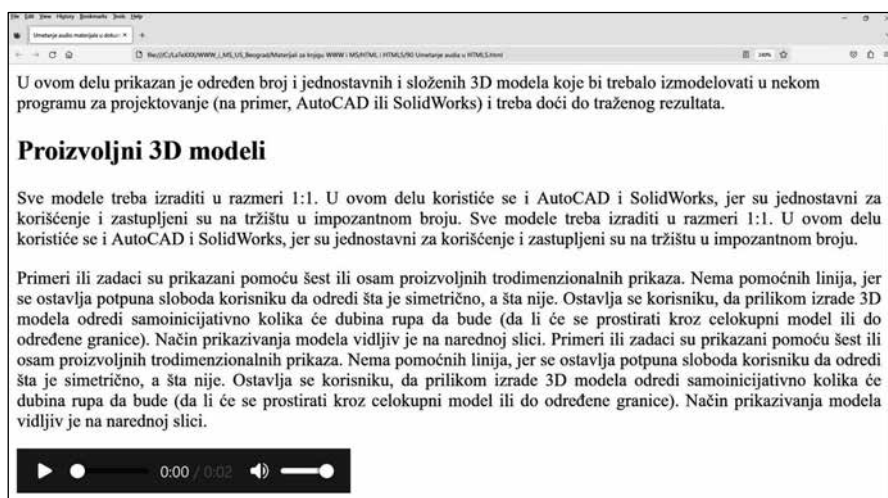
⁸⁷ 87 Definisanje podnožja dokumenta ili sekcije HTML5.html



Slika 10.81. Umetanje videa u veb stranicu⁸⁹

Trenutno, postoje 3 podržana video formata za element `<video>` – MP4 fajlovi sa H264 video kodekom i sa AAC audio kodekom; webM fajlovi sa VP8 video kodekom i *Vorbis* audio kodekom; Ogg fajlovi sa *Theora* video kodekom i *Vorbis* audio kodekom.

HTML5 obezbeđuje standard za slušanje **audio** fajlova. Pre HTML5, nije bilo standard za slušanje audio fajlova unutar veb stranice. Pre HTML5, audio fajlovi su mogli da se slušaju jedino sa ugrađenim dodatkom (*plug-in*, kao što je to *Flash*). Međutim, različiti pretraživači se podržavali različite dodatke. HTML5 definiše novi element koji određuje standardan način da se "ugradi" audio fajl na veb stranicu, a reč je o `<audio>` elementu.



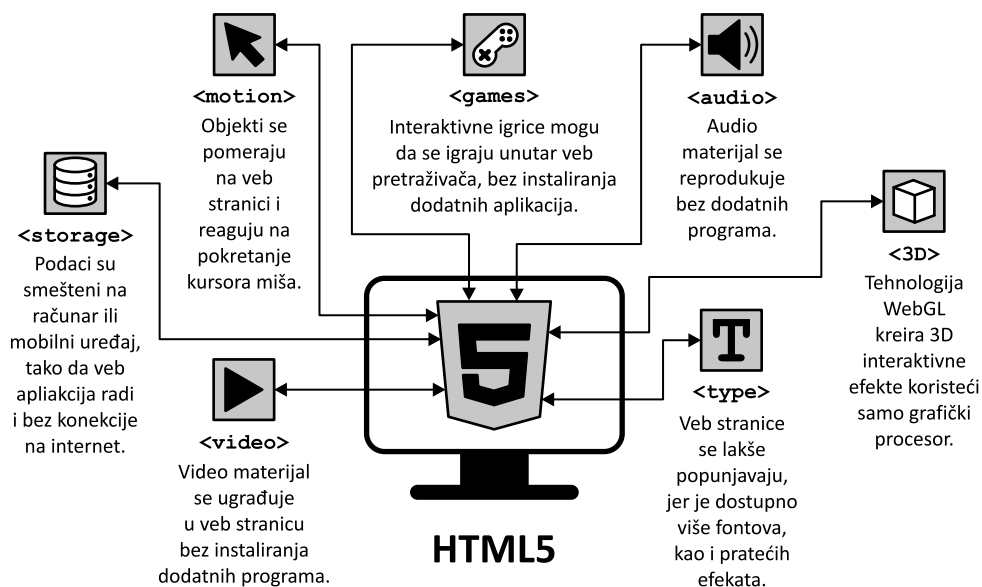
Slika 10.82. Umetanje audio datoteke u veb stranicu⁹⁰

⁸⁹ 89 Umetanje videa u HTML5.html

⁹⁰ 90 Umetanje audia u HTML5.html

Kontrolni atribut dodaje audio kontrole, kao što su play, pause i volume. Takođe bi trebalo da se ubaci određeni tekstualni sadržaj između oznaka `<audio>` i `</audio>`, koji bi "signalizirao" da tekući pretraživač ne podržava `<audio>` element. Element `<audio>` omogućava primenu i upotrebu više `<source>` elemenata. Elementi `<source>` mogu da se povezuju na različite audio fajlove. Pretraživač će koristiti prvi prepoznat format za reprodukovanje audio materijala.

Trenutno, postoje 3 podržana audio formata za `<audio>` element – MP3, najrasprostranjeniji audio format u kome je primenjena kompresija sa gubitkom; Ogg "otvoreni" format (*open source*) sa višim stepenom kompresije, ali i manjom rasprostranjenošću; Wav audio format za audio fajlove na personalnim računarima (*Waveform Audio File Format*, *WAVE* ili odomaćeno *WAV*).



Slika 10.83. Šta HTML5 može da radi?

Glava 11

CSS i CSS3

"Ispod" svih veb stranica nalazi se stari i dobar HTML jezik, koji kontroliše stvari kao što su veličina fonta, boja teksta, gde slika ide, kao i informacije o drugim elementima na stranici. HTML jezik se ponekad naziva običan HTML jezik, da bi se razlikovale veb stranice izgrađene sofisticiranijim tehnikama, kao što su **liste stilova** (*style sheets*). I reč običan je sigurno dobra za HTML.

Bez pomoći, HTML često stvara, zaista, dosadne stranice. Zbrka nastaje kada se pokušava da se napravi veb stranica koršćenjem samo HTML koda. Tu priskaču u pomoć liste stilova (*style sheets*).

Jednostavan način da se razume svrha CSS-a je da se na njega gleda kao na dodatak HTML-u koji pomaže da se olakša i unapredi dizajn veb stranice. U stvari, neki CSS efekti nisu mogući bez pomoći HTML jezika.

Još jedna prednost CSS-a je da omogućava da stil odredi jednom, a pretraživač može taj stil da primeni mnogo puta u dokumentu. Na primer, ako autor želi da neka slika, prikazana na njegovom veb sajtu, da ima tanak, plavi okvir, može da definiše ovaj okvir kao stil u CSS-u. Zatim, umesto da ponavlja HTML definiciju tankog i plavog okvira, svaki put kada želi da se promeni konkretni okvir, može samo da ubaci CSS stil kao atribut za svaki grafički element koji želi da bude uramljen.

Drugačije rečeno, CSS bi trebalo koristiti za definisanje opštih pravila o tome kako se elementi ponašaju unutar veb stranica i kako izgledaju, gde se nalaze, njihove veličine, transparentnost itd. Onda bi samo trebalo viditi računa o imenu pravila koje treba primeniti i koje treba sprovesti u pripadajućoj HTML stranici.

Evo primera CSS pravila koje definiše nekoliko osobina koje bi trebalo primeniti na najveći naslov H1 unutar HTML stranice:

```
<style>
  H1 {font-size:16pt color:blue;}
</style>
```

Sa ovim CSS pravilom na snazi, bilo koji HTML kôd koji sadrži H1 element će ovaj naslov ispisati veličinom 16 pt i plavom bojom:

```
<html>
  <body>
    <h1> Ovaj naslov je plave boje i velicine 16pt </h1>
  </body>
</html>
```

CSS pravila mogu da se definišu u posebnom .css fajlu (datoteci) ili mogu da budu ugrađena u HTML fajl. Sledi primer CSS pravila koje će biti "ugrađeno" u zaglavlje HTML fajla:

```
<html>
  <head>
    <style>
h1 {font-size:16pt color:blue;}
    </style>
  </head>
  <body>
    <h1> Ovaj naslov je plave boje i velicine 16pt </h1>
  </body>
</html>
```

Trebalo bi obratiti pažnju na <style> element, jer mogu unutar ovog elementa da se definišu CSS stilovi. Unutar veb stranice, mogu da postoje višestruki <style> elementi, ako autor to želi.

11.1. CSS od početka

U ovom delu će biti predstavljeno nekoliko, donekle, teorijskih koncepata koji su bitni za razumevanje CSS-a. Autor može da napiše CSS koristeći bilo koji običan tekst editor. Kao i XML i drugi savremeni računarski jezici izvedeni iz HTML-a, i CSS ne pokušava da stvori neki čudni, specijalizovani jezik kodova, koji ne bi mogao lako da se čita i piše.

Lista stilova se često piše i čuva kao .css datoteka na hard disku. Nakon toga, svaka HTML (sa ekstenzijama .htm ili .html) stranica koja želi da koristi stilove definisane u datoteci .css trebalo bi samo da je referencira kao link. Očigledna je veza između HTML-a i CSS-a.

Na primer, trebalo bi pokrenuti program Notepad++. Onda bi trebalo ukucati sledeću sintaksu stila koja govori veb sajtu kako da prikaže tekstualne paragrafe:

```
p {color: blue; text-align: justify; font-family: courier;}
```

Ovaj stil bi trebalo zapamtiti primenom komande *Save As* unutar programa Notepad++, a zatim imenovati fajl kao CSS-1.css. Može da se koristi bilo koje ime, ali to ime mora da se referencira u HTML kodu željene veb stranice. Trebalo bi zatvoriti ovaj fajl i startovati novi dokument unutar programa Notepad++, gde će se otkucati sledeći tekst:

```

<html>
<head>
<title>Visenamenski avioni</title>
<meta charset="UTF-8">
<link type="text/css" rel="stylesheet" href="CSS-1.css">
</head>
<body>
<h1> Visenamenski borbeni avioni </h1>
<p align="justify"> Visenamenski borbeni avion je borbeni ... </p>
</body>
</html>

```

Ova veoma pojednostavljena sintaksa veb stranice ilustruje kako se radi sa eksternom CSS datotekom. U .css datoteci je specificirano da se paragraf teksta <p> ispisuje plavom bojom, da je poravnat tekst po obe margine (i po levoj i po desnoj) i da se koristi font *Courier*. U prikazanom HTML kodu, izvršeno je linkovanje ka listi stilova pod nazivom CSS-1.css. Ova lista stilova se nalazi u istom direktorijumu kao i .htm ili .html fajl.



Slika 11.1. Osnovni dokument sa poravnatim tekstom po obe margine¹

U ovom slučaju nije definisan stil za zaglavlje i naslov H1, tako da se za ovaj naslov primenjuju podrazumevane standardne veličine koje "propagiraju" pretraživači, kao i boje, pozicije, font (a možda i druge osobine). Kod većine pretraživača naslov je poravnat uz levu marginu, crne je boje i ispisuje se serifnim fontom.

Kada je sintaksa u pitanju, svako pravilo CSS-a je podeljeno na 3 (tri) dela – selektor (*selector*), osobina (*property*) i vrednost (*value*); upareni elementi su odvojeni velikim zagradama od ostatka sintakse:

```
{property: value pair}
```

Takođe, osobina je odvojena od vrednosti dvotačkom. Kompletно CSS pravilo izgleda ovako:

¹ 01 Osnovni dokument sa poravnatim paragrafima CSS.html

```
selector {property: value};
```

Evo primera CSS stila iz realnog sveta. Definisan je HTML paragraf <p> i dodeljeno mu je da se tekst ispisuje plavom bojom:

```
p {color: blue};
```

Selektor CSS-a je ono što se u HTML-u zove **tag** ili **element** – stavka u HTML kodu koja je okružena uglastim zagradama (kao što su <h3>, <p> ili <td>).

Prethodna pisanija opisuje najjednostavnije CSS pravilo. Međutim, može da se primeni grupa više selektora na jedan stil (na primer, elementima h1, h2, p mogu u jednom pravilu da budu dodeljene određene osobine). Isto tako, može grupa više osobine da bude u jednom pravilu, kao što je prikazano u narednom primeru:

```
p {  
color: blue;  
text-align: justify;  
font-family: courier;  
}
```

Kao i HTML kôd, CSS kôd može slobodno da se preuredi na bilo koji način – linija koda može da se formatira na bilo koji način. Sledeća sintaksa je identična prethodnoj:

```
p {color: blue;  
text-align: justify; font-family: courier;}
```

Ova dva pravila (načina pisanja sintakse) znače potpuno istu stvar u veb pretraživaču, ali CSS programeri imaju svoje lične želje u vezi formatiranja.

CSS osobina (*property*) je ono što se u HTML-u zove **atribut** – ime osobine ili kvaliteta koje je specificirano (kao što su boja ili veličina fonta). Mnogi računarski jezici, takođe, koriste termin *property* na isti način.

Konačno, CSS vrednost (*value*) pokazuje stvarne podatke o osobini. Drugim rečima, vrednosti crno ili zeleno mogu biti vrednosti osobine za color. Vrednost u CSS ima isti koncept kao vrednost koja je definisana za promenljivu ili konstantu veličinu u programiranju. Računarski programski jezici koriste termin *value* na isti način na koji to i CSS radi.

Neke vrednosti su opisane sa više od jedne reči, kao što je upotreba boje *minty green* umesto *green*. Kada je vrednost definisana sa više reči, onda se naziv vrednosti daje pod navodnicima:

```
p {color: "minty green";}
```

Ako autor želi, može da definiše više tagova (selektora) odjednom. Ovo može da uštedi dosta vremena, ako želi, na primer, da mu svi naslovi budu ispisani *Arial* fontom. HTML ima četiri elementa za naslove H1, H2, H3 i H4. Umesto definisanja svakog pojedinačnog elementa u CSS fajlu, mogu da se navedu i definišu svi selektori zajedno, odvojeni zarezi-ma. Na primer, sledeća sintaksa pokazuje da se svi naslovi (bez obzira na veličinu) ispisuju fontom *Calibri*:

```
h1,h2,h3,h4
{font-family: Calibri;}
```

Na slici 11.2 prikazan je izgled ekrana nakon startovanja odgovarajućeg fajla u aktivnom veb pretraživaču.



Slika 11.2. *Promena podrazumevanog fonta za naslove²*

Evo sledećeg primera. Autor želi da mu se većina paragrafa u tekstu ispisuje crnom bojom, ali želi da naglasi neke paragrafe i ti paragrafi bi trebalo da se ispisuju crvenom bojom. To je lako uraditi ako se doda **klasa** (*class*) selektoru, kao što je prikazano u donjoj sintaksi. Ovde su definisane klase `bodytext` i `uzbuna` za selektor `p`:

```
p.bodytext {color: black}
p.uzbuna {color: red}
```

Autor ne mora uvek da dodeljuje klasu tagu (oznaci). Može da se odredi generička klasa koja se može primeniti na bilo koju vrstu oznake (taga) u HTML kodu. Na ovaj način može da se izbegne dodeljivanje specifične vrednosti svakom tagu ponaosob. Ovakva generička klasa može da se dodeli svakom elementu, kada je to potrebno (na primer, kad god bi trebalo da neki element bude plav ili da ima drugu boju).



Slika 11.3. *Promena boje određenog teksta³*

² 02 Promena fonta za naslove CSS.html

³ 03 Promena atributa - boje - pomocu klasa CSS.html

Na primer, ako autor želi da bude u stanju bilo koji plavi tekst naznači ili istakne (*highlight*), onda bi trebalo izostaviti ime selektora i pratećih elemenata (na primer, izostaviti oznaku `p` u prethodnoj sintaksi), kada se vrši definisanje `highlight` klase u CSS datoteci. Sledeća generička klasa definiše primenu plave boje (slika 11.4):

```
/* Ova klasa se koristi u svakom HTML tagu */
.highlight {color: blue}
```



Slika 11.4. Promena boje proizvoljnog teksta⁴

Ako treba da se doda komentar u okviru CSS fajla, onda se komentar prilaže između `/*` i `*/` simbola. Svrha komentarisanja koda je lakše čitanje i, eventualno, modifikovanje:

```
/* Ovo je KOMENTAR */
```

ID selektori (*ID selectors*) su slični generičkim klasama i oni su nezavisni od konkretnih elemenata dokumenata. Ali ID selektori se razlikuju od klasa u tome što ID selektori rade samo jednom na datoj veb stranici – utiče se samo na jedan element (na jedan sa ID selektorom). Sve naredne reference u HTML kodu ignorišu već iskorišćen ID selektor.

U sledećoj sintaksi, ID selektor sa nazivom `highlight` je definisan u zaglavlju (`<style>`), tako da može da se koristi unutar veb stranice gde god se pominje ID (slika 11.5).

```
<style>
#highlight {color: green; font-style: italic}
</style>
```

```
<h1 id="highlight">
Visenamenski borbeni avioni - ZELENI I ITALIC NASLOV </h1>
```

```
<p id="highlight" align="justify"> <b> ZELENI I ITALIC TEKST. </b>
Kao sto je receno, naziv visenamenski avion ...</p>
```

⁴ 04 Promena boje bilo kog dela teksta CSS.html



Slika 11.5. Promena parametara pomoću ID selektora⁵

Kada se kuca HTML i CSS kôd (obično) nije važno da li se to radi malim ili velikim slovima. Imena tagova, atributa i tako dalje mogu da se napišu na bilo koji način – veb pretraživač će razumeti da se radi o jednom entitetu. Na primer, BODY, body, BoDy, boDY ili BoDy su sve dozvoljeni tagovi "tela teksta" i mogu da se koriste naizmenično.

U praksi, većina ljudi, uglavnom, pokušava da piše svoje programe u HTML, CSS i drugim kodovima u potpunosti malim slovima. Ako ništa drugo, lakše je za kucanje. Kao drugo, neće biti signaliziranja oko grešaka ako je negde, greškom, korišćeno neko veliko slovo. Na primer, XHTML zahteva primenu samo malih slova.

Dodatni izuzeci po pitanju "osetljivosti" na mala i velika slova su izlazne sekvence koje počinju karakterom & i internet adresom (URL, *Uniform Resource Locator*, uniformni lokator resursa). Nisu sve URL adrese "osetljive" na mala i velika slova, ali neke jesu. Ova vrsta konfuzije i neefikasnosti je zaista previše dosadna i nepotrebna.

Nekoliko faktora određuju koji stil "pobeđuje" kada se stilovi "sudare" – nasledstvo, strukturalno drvo u dokumentu, kao i specifičnost (ili bliskost) stila. Verovatno je najlakše razumeti pravila prilikom "sudara" stilova na osnovu toga gde su stilovi definisani i locirani. CSS stilovi mogu da se definišu u četiri glavne lokacije:

- podrazumevani stilovi veb pretraživača;
- eksterna lista stilova (.css fajl koji sadrži definicije stila je označen unutar HTML dokumenta pomoću elementa link);
- ugrađena lista stilova (stilovi su definisani u okviru HTML dokumenta unutar <head> taga i ova vrsta stila se ponekad naziva interna);

⁵ 05 Promena boje unutar samog dokumenta pomocu ID selektora CSS.html

- redni stil (definisan kao atribut u samom HTML elementu, a čije je dejstvo ograničeno samo na taj elemenat).

Na primer, ovo je tipičan redni (*inline*) stil:

```
<p style="color: blue;" align="justify">
```

Ova lista, takođe, ilustruje redosled kojim konfliktne stilovi "pobeđuju" u svakom "sukobu" ili preklapanju.



Slika 11.6. Redosled stilova i njihova primena⁶

Kada se odredi stil pomoću CSS, stilovi se primenjuju i na elementu i njegovim "potomcima". Na primer, ako autor navede da `<p>` element treba da bude prikazan sa zelenom pozadinom, onda većina elemenata "dece" tog `<p>` elementa će imati svetlo zelenu pozadinu. Evo kako to radi:

```
<style type="text/css">
p {background: lightgreen}
</style>
```

Dodatna sintaksa je prikazana u datoteci pod brojem 7, a izgled dokumenta u određenom veb pretraživaču je prikazan na slici 11.7.



Slika 11.7. Promena boje pozadine paragrafa⁷

⁶ 06 Redosled stilova CSS.html

⁷ 07 Definisane i promena boje pozadine poglavlja CSS.html

Bez obzira koliko ko poznaje CSS, većina će nastaviti da piše tekst upozorenja kao HTML kôd na sledeći način:

```
<h3><font color="red">UPOZORENJE</font></h3>
```

To je vrsta linije koja se, verovatno, piše i desetak puta unutar jednog veb sajta. Recimo, na primer, da autor ima niz upozorenja kroz veb sajt i da svakom upozorenju prethodi H3 naslov u crvenoj boji. Za svaki naslov, navodi se da je crven pomoću elementa.

Onda se pojavljuje CSS. Sada može da se napravi jedan selektor H3 na jednoj lokaciji, a ipak će da utiče na sve H3 naslove širom celog veb sajta. Taj stil čini da su H3 naslovi automatski crveni. Ako kasnije autor odluči da promeni H3 naslove iz crvene boje u podebljane (*bold*) naslove, nema problema. Mora tu promenu da obavi samo jednom u CSS stilu, umesto na desetinama mesta gde su H3 elementi "rasuti" širom HTML koda.

Selektor određuje na koji element u HTML kodu se primenjuje pravilo CSS stila. Na primer, trebalo bi koristiti selektor p ako se želi da se pravilo primenjuje na HTML element <p> za paragrafe. Ili koristiti selektor h1 da se definiše CSS pravilo koje se primenjuje na H1 naslove u HTML kodu. Drugim rečima, CSS selektor predstavlja HTML element, bez zagrada (<>). Evo kako izgleda definisanje selektora H1 u okviru CSS pravila:

```
<style> h1 {color: red;} </style>
```

Ovo pravilo znači da će svaki put, kada bude izabran element H1 u HTML kodu, na njega biti primenjena crvena boja. Selektori govore CSS-u na koje se HTML elemente primenjuje CSS pravilo.

Da bi ove sitnice mogle da se primenjuju bez upotrebe univerzalnog selektora, trebalo bi napraviti grupu selektora na sledeći način:

```
body, p, table, h1, h2, h3, h4, {font-weight: bold;}
```

Umesto izgradnje ogromne grupe selektora, može jednostavno da se koristi univerzalni selektor i da promeni tekst celog veb sajta da bude podebljan na sledeći način:

```
* {font-weight: bold;}
```

Zvezdica, sama po sebi, je univerzalni selektor. To znači da se naznačena osobina primenjuje na sve elemente u dokumentu. U prethodnoj sintaksi je obavljeno dodeljivanje osobine font-weight sa vrednošću bold svakom elementu.

Bilo je već reči o grupisanju. Na jednom mestu je bilo specificirano više selektora, a zatim im je dodeljen stil koji bi trebalo da dele, na sledeći način:

```
h1, h2, h3, h4, {font-weight: bold;}
```

Mogu da se koriste i višestruke deklaracije (parovi osobina/vrednost unutar zagrada), na sledeći način:

```
h1 {font: 48pt Calibri; color: blue}
```

Trebalo bi zapamtiti, kada se koristi višestruka deklaracija, da se odvajaju tačkom-zarezom (;). Prostori (blanko prostori) se ovde, jednostavno, ignorišu od strane CSS-a, tako da ne mogu da se računaju kao prostor za razdvajanje deklaracija, jednih od drugih.

Međutim, izostavljanje konačne (završne) tačke-zareza nije dobra ideja. Jednostavno treba da autoru pređe u naviku da svaku deklaraciju završava tačkom-zarezom. Izostavljanje tačke-zareza iz liste deklaracija veoma često izaziva konfuziju unutar veb pretraživača. Na slici 11.8 prikazan je primer gde je naslov H1 prikazan plavom bojom i podebljan, fontom *Calibri* i veličinom 48 pt.

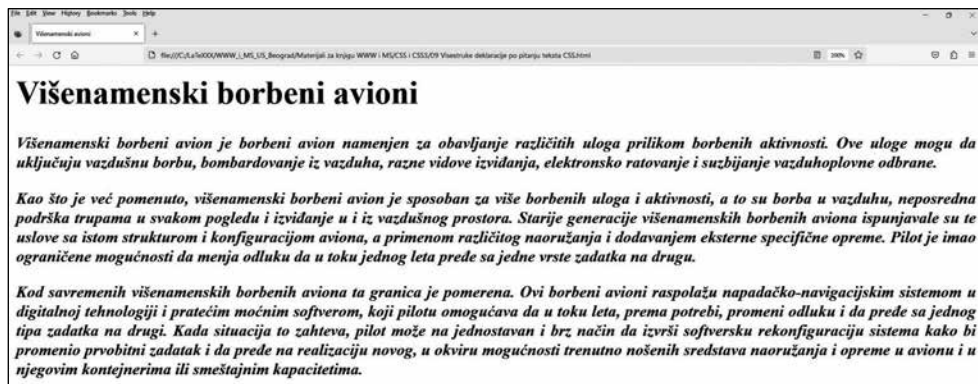


Slika 11.8. Višestruke deklaracije na razne načine⁸

Na osnovu ovoga, sledi i logičan zaključak da je moguće istovremeno grupisati i selektore i deklaracije, na sledeći način:

```
h1, h2, h3 {color: green; background: white; font-family: Calibri;}
```

Selektori atributa mogu da prošire korisnost i fleksibilnost klasa i ID atributa, pružajući veću slobodu da se probaju neki korisni trikovi, što omogućava autoru veb stranice da bude kreativan na više načina.



Slika 11.9. Višestruke deklaracije kada je tekst u pitanju⁹

⁸ 08 Višestruke deklaracije CSS.html

⁹ 09 Višestruke deklaracije po pitanju teksta CSS.html

Do sada je bilo eksperimentisanja sa kratkim primerima – definisanje jednog selektora i dodavanje jednog ili dva elementa u HTML kôd. U fajlu pod brojem 10 kreirana je veb stranica, a unutar CSS-5.css datoteke (koja je uvrštena u sintaksu pomenutog .html fajla) mogu da se menjaju određeni parametri.



Slika 11.10. Višestruke deklaracije u eksternom fajlu¹⁰

Sadržaj pratećeg CSS-5.css, koji se nalazi unutar fajla pod brojem 10, je:

```
strong {color: #400000;}
em {color: #400000;}
p {mso-margin-top-alt: auto; margin-right: 0mm;
  mso-margin-bottom-alt:auto; margin-left: 0mm;
  text-indent: 12.25pt; mso-pagination: widow-orphan;
  font-size: 12.0pt; font-family:"Calibri";
  mso-fareast-font-family: Times;}

h1,h2,h3,h4 {font-family:"Calibri";
  mso-fareast-font-family: Times;}
```

Trebalo bi obratiti pažnju da su oba selektora (strong i em) definisana da imaju boju sa vrednošću #400000. Pošto su ovi stilovi ugrađeni u ovom HTML dokumentu, oni menjaju eksterne liste stilova, koje su povezane sa dokumentom. Ugrađeni stilovi su "preči" od spoljašnjih stilova, koji su namenjeni da menjaju HTML tagove. "Preči" stilovi se, takođe, opisuju kao precizniji ili specifičniji. Dakle, ugrađeni stilovi "pobeđuju" ako dođu u "sukob" sa eksternim stilovima.

Evo nekih pravila o specifičnostima CSS-a:

- Elementi koji se tretiraju kao "deca" (kao što element <p> paragrafa u okviru jednog "roditelja" <body>) nasleđuju stilove roditelja. Ako <body> ima stil u vezi sa tim da se tekst naglašava podvlačenjem, onda će i svaki <p> element unutar roditeljskog

¹⁰ 10 Višestruke deklaracije unutar eksternog CSS fajla CSS.html

elementa <body> naslediti isticanje teksta podvlačenjem. Međutim, ako postoje specifičniji ("važniji") stilovi u veb stranici (kao što je *inline* stil), onda će oni "nadjati" i zameniti stil sa podvlačenjem teksta.

- CSS stilovi sadržani u HTML dokumentu (ugrađeni ili *inline* stilovi) imaju prednost nad spoljnim stilovima (nalazi u posebnom .css fajlu i navedeni su u HTML kodu pomoću taga <link>).
- Stil sa !Important komandom ima prednost, nezavisno od drugih faktora.

11.2. CSS i pozicioniranje

Možda jedan od najosnovnijih elemenata dizajna je pozicioniranje. Na primer, korisnik dobije pet objekata i od njega se traži da ih organizuje (pozicionira) unutar okvira. Tekstura, boja ili oblik tih pet objekata nisu mnogo važni. Ako je korisnik talentovan, onda može da napravi da ti objekti izgledaju dobro unutar okvira uređujući (pozicionirajući) ih na najbolji način. Pozicioniranje elemenata na veb stranici je, takođe, od ključnog značaja za "uspeh" te stranice.

Uobičajeno je da jednostavna veb stranica organizuje njene elemente od vrha do dna, ređajući ih jednu ispod druge, poravnavajući ih uz levu marginu stranice. Sledeća skraćena sintaksa prikazuje dva paragrafa, sa slikom između njih:

```
<html>
<head>
<title> Visenamenski avioni </title>
<meta charset="UTF-8">
<link type="text/css" rel="stylesheet" href="CSS-5.css">
</head>
<body>
<h1 align="center"> Visenamenski borbeni avioni </h1>
<p align="justify">Prvi paragraf ...</p>
<p>
 
</p>
<p align="justify"> Drugi paragraf ...</p>
</body>
</html>
```

Poboljšanje dizajna prikazanog na slici 11.11 je postignuto na jednostavan način – dodat je atribut da centrira naslov:

```
<h1 align="center"> Visenamenski borbeni avioni </h1>
```

HTML nije dizajniran da se efikasno bavi izgledom i razmeštajem elemenata van stranice. CSS je, delom, razvijen da reši ovaj problem. CSS sadrži različite alate za pozicioniranje, uključujući i sofisticirano **relativno pozicioniranje**. Može da se definiše relativna

lokacija jednog objekta u odnosu na drugi, koji se već nalazi unutar veb stranice. Ovo rešava problem korisnika prilikom promene veličine prozora veb pretraživača. Relativno postavljani objekti zadržavaju međusobni odnos, bez obzira šta se dešava sa samim pretraživačem. Relativno pozicioniranje znači da je elemenat lociran (pozicioniran) u odnosu na neki drugi ili postojeći element.



Slika 11.11. Osnovno pozicioniranje elemenata u veb dokumentu¹¹

Relativna lokacija može da znači da je element smešten u sredini drugog elementa, ili 35 piksela sa njegove leve strane ili 25% ispod njegovog vrha. Suština je da pozicija prvog elementa zavisi od lokacije (pozicije) drugog elementa.

Apsolutno pozicioniranje navodi određeni broj (piksela, tačaka, inča, šta god izabrani merni sistem zahteva) kojim element treba da bude pozicioniran i posatvljen na određeni položaj. Apsolutna veličina je opisana brojevima koji definišu širinu i visinu, kao što je opisana veličina slike u ranijem HTML kodu:

```

```

Apsolutni položaj se opisuje sa dva broja – gornji (*top*) i levi (*left*). Ovi brojevi se u tehnici nazivaju koordinate, jer dva broja će "reći" pretraživaču gde tačno da postavi element u prozoru pretraživača. Vrednost *top* 70mm će smestiti element 70 mm ispod gornje ivice okvira prozora pretraživača, a vrednost *left* 50mm će pomeriti element 50 mm od leve ivice okvira prozora pretraživača. Koristeći ove dve koordinate, autor može da postavi bilo šta, bilo gde.

Sledi pregled različitih načina kako može da se promeni ili definiše pozicija u CSS selektoru: *static*, *relative*, *absolute*, *fixed* i *inherit*.

Režim *static* je sličan uobičajenom (*default*) režimu – šta se dešava kada se ne navede nikakva vrednost položaja ili pozicije. **Statički elementi** se prostiru na podrazumevani

¹¹ 11 Osnovno pozicioniranje unutar dokumenta CSS.html

način (slažu se sa leve strane), ali ne mogu biti premešteni; oni ostaju gde jesu i ne mogu da im se dodele `top` i `left` koordinate. Ne može da se podešava položaj statičnog elementa ni korišćenjem skript jezika, kao što su *VBScript* ili *JavaScript*. Ti elementi ostaju tamo gde su – statični (*static*).

Relativno pozicioniranje je slično statičnom pozicioniranju, ali može da koristi `top` i `left` koordinate, kako bi se menjalo uobičajeno "prostiranje" ili pozicioniranje elemenata, relativno (*relative*) u odnosu na druge i postojeće elemente.

Apsolutno pozicioniranje odvaja element od drugih objekata – njegova pozicija nema "veze" sa njima i nije u relativnom odnosu na njih. Reč je o nezavisnom pozicioniranju. Nalazi se u prozoru pretraživača na određenim rastojanjima u odnosu na gornju i levu ivicu prozora. Ako se primene `top` i `left` atributi i ako im se dodele vrednosti `top: 0px; left: 0px`, onda će element da bude smešten u gornjem levom uglu prozora pretraživača.

Da bi se videlo kako apsolutno pozicioniranje utiče na smeštaj slike, trebalo bi probati sintaksu ispod (to je deo sintakse koja se nalazi u fajlu pod brojem 12):

```
<style> .klasa {position: absolute; top: 0px; left: 0px;} </style>
<p>

</p>
```



Slika 11.12. *Apsolutno pozicioniranje elemenata u veb dokumentu*¹²

Ovaj primer pokazuje kako pozicioniranje grafike na koordinatu 0,0 (*absolute*) uzrokuje da se delimično pokrije naslov i potpuno pokrije tekst unutar elementa `<body>`. Šta ako se promeni atribut po pitanju pozicioniranja, od apsolutnog u relativni? Sintaksa je slična prethodnoj, samo je promenjen atribut *absolute* u *relative*:

```
<style> .klasa {position: relative; top: 0px; left: 0px;} </style>
<p>

</p>
```

Promena atributa pozicioniranja sa apsolutne relativnu vrednost dovodi do uobičajenog ponašanja. Grafički element je postavljen na "normalno" mesto – po redosledu elemenata na osnovu njihovog reda u HTML kodu, kao što je prikazano na slici 11.13.

¹² 12 Apsolutno pozicioniranje slike unutar dokumenta CSS.html



Slika 11.13. *Relativno pozicioniranje elemenata u veb dokumentu*¹³

Elementi sa fiksnom vrednošću za položaj ostaju na svojoj lokaciji u pretraživaču, čak i ako korisnik skroluje (pomera sadržaj prozora) kroz pretraživač. Ostali elementi koji nisu fiksni se pomeraju prilikom skrolovanja ali fiksni elementi ne. Ovaj trik može da se koristi kako bi iz dobio "putujući" naslov ili ako treba naslove polja fiksirati za vrh tabele.

Vrednosti `top` i `left` su specifične lokacije u odnosu na prihvatni objekat. Na primer, ako specifični element nije "zatvoren" unutar drugog elementa, vrednosti `top` i `left` su podrazumevane za smeštaj takvih elemenata u odnosu na prozor pretraživača (ili prozor prikaza ili *viewport*).

Ponekad su elementi pozicionirani u okviru drugih elemenata (kao što je paragraf `<p>` unutar bloka `<div>`). U tom slučaju, precizirajući vrednosti `top` i `left` u odnosu na element "kontejner", blok `<div>` će da okruži paragraf `<p>`.

Trebalo bi zamisliti dokument koji je sastavljen od blokova koji su smešteni unutar drugih blokova. Jednostavan dokument sa nizom paragrafa (`<p>`) može da se posmatra kao element `<body>` koji sadrži `<p>` elemente. Onda bi zvanična terminologija mogla da bude da se tekstualni blok sastoji od blokova paragrafa.

11.3. CSS i tekst

CSS nudi mnogo načina da se uredi tekst. Ako se naprave mudri izbori, ukupni dizajn veb stranica može da utiče na posetioce sajta. Sa druge strane, inače elegantna i moćna veb stranica može biti uništena ako tekst nije na odgovarajući način integrisan sa grafičkim efektima.

¹³ 13 Relativno pozicioniranje slike unutar dokumenta CSS.html

Korisnik vidi na odgovarajućem sajtu i grafiku i tekst. Kao dizajner, autor je odgovoran za izbegavanje očiglednih problema, kao što su dovoljno tamne grafičke pozadine da je tekst u prvom planu nečitljiv, "iskačuće" animacije, odvlačenje pažnje bojama, loš opšti dizajn itd.

Ali autor je odgovoran i za izbor karakteristika teksta i grafičkih elemenata, kako bi sveukupni dizajn odražavao ideju koju autor želi da projektuje do korisnika. Čak i kada se izabere vrsta slova, moraju da se razmotre i druga pitanja prilikom izrade CSS stilova za tekst – veličina fonta, da li se ispisuje podebljano (*bold*) ili zakošeno (*italic*), da li je tekst centriran ili poravnat po obe margine, kako se pišu stepeni ili indeksi, da li se piše prvo slovo poglavlja uvećano ili da li je tekst trebalo da bude podvučen.

Prvo, porodice fontova (*font families*) dele dovoljno karakteristika, tako da je to dovoljno u većini slučajeva. Šta više, velika većina veb korisnika koristi operativni sistem *Windows*, koji se isporučuje sa standardnim setom fontova, uključujući *Times New Roman*. Ovo je font koji će koristiti većina dizajnera, a čak neka ispitivanja pokazuju da font *Times New Roman* koristi oko 96% dizajnera.

Porodice fontova su, što nije iznenađujuće, navedene u CSS pravilima korišćenjem atributa ili osobine `font-family`. Ako autor želi da svi paragrafi u dokumentu koriste font *Arial*, onda bi sintaksa trebalo da izgleda ovako:

```
p {font-family: arial;}
```

Neka imena fontova imaju više reči (i međuprostor), kao što su *Times New Roman* ili *Showcard Gothic*. U tim slučajevima (kao i ako naziv sadrži simbole kao što su % ili @), trebalo bi umetnuti ime u pojedinačne navodnike, na sledeći način:

```
p {font-family: 'Showcard Gothic', Arial;}
```

Fontovi spadaju u dve osnovne kategorije, na osnovu toga da li imaju ili nemaju "zavijutke" i različite širine linija. Na slici 11.14 prikazan je primer *SansSerif* fontova i ti fontovi imaju jednostavan stil, fiksnu širinu linija i linije se jasno završavaju.

Microsoft SansSerif **ARIAL** Calibri

Slika 11.14. *SansSerif fontovi*

Kada se pogleda slika 11.15, vidi se očigledna razlika kada je u pitanju *Serif* font. Stil nije jednostavan, postoje linije različite širine, a i postoje završeci, koji nisu baš preporučljivi.

Times New Roman **Q** Cambria

Slika 11.15. *Serif fontovi*

Ne bi trebalo ni trošiti mnogo vremena monospace fontove. Ovakvi fontovi (*Courier*) se koriste poodavno. Oni su se koristili, ne zato što su izgledali dobro ili što su bili laki za čitanje, već zato što su rešili tehnološki problem. Ova vrsta fonta ima istu širinu svakog karaktera, čak i kada je to besmisleno. Drugim rečima, slovo *i* je iste širine kao i *w*. Dakle, postoji "hrpa" belog prostora na obe strane slova *i*, tako da ostala slova imaju ili previše ili premalo prostora između njih.

Kada autor želi da podesi "težinu" (*weight, darkness, boldness, lightness*) fonta, onda može da pribegne relativnim CSS vrednostima – `normal`, `bold`, `bolder`, `lighter`, `100`, `200`, `300`, `400`, `500`, `600`, `700`, `800`, `900`, `inherit`. Što je veća vrednost, to je font "teži" ili "tamniji". Najčešće se jednostavno koristi `bold` vrednost:

```
H2 {font-weight: bold;}
```

Vrednosti od 100 do 900 su samo naznake željene težine, ali malo korisničkih pisama imaju ovoliko raznovrsnu težinu. Iako ova šema nije apsolutna, vrednosti od 100 do 300 mogu da budu ekvivalentne vrednosti `light`, kao što je font *Copperplate Gothic Light*, vrednosti od 400 do 500 kao redovni (CSS vrednost `normal`) i vrednosti od 600 do 900 kao `bold`. Međutim, ako font ima definisanu "težinu" kao `bold`, `ultrabold` i slično, onda brojčane vrednosti dodaju na njegovoj specifičnosti.

Autor može da koristi vrednosti `lighter` ili `bolder` kako bi naveo stepen naglašenosti u odnosu na osobine "roditeljskog" elementa, na sledeći način:

```
body {font-weight: bold;}
p {font-weight: bolder;}
```

U ovom slučaju, `<body>` elementi su podebljani, ali `<p>` elementi su, ako je to moguće, prikazani u još "debljem" izdanju. Font *Times New Roman* nema nikakvu "težinu" izvan vrednosti `bold`, tako da vrednost `bolder`, kod njega, nema nikakvog efekta. Međutim, u sledećoj sintaksi, tekst van `<p>` oznake je redovna (vrednost `regular`), onda je tekst unutar `<p>` tagova podebljan (vrednost `bold`):

```
body {font-weight: regular;}
p {font-weight: bold;}
```

Vrednost `small-caps` je specijalizovana varijanta sa normalnim slovima. To stvara velika slova koja su oko 75% veličine normalnih velikih slova. Ovo se ne koristi često, ali izgleda dobro kada je potrebno. Najčešće se koriste kada treba označiti datum, kao što je na slici 11.16 reč o datumu pre nove ere.

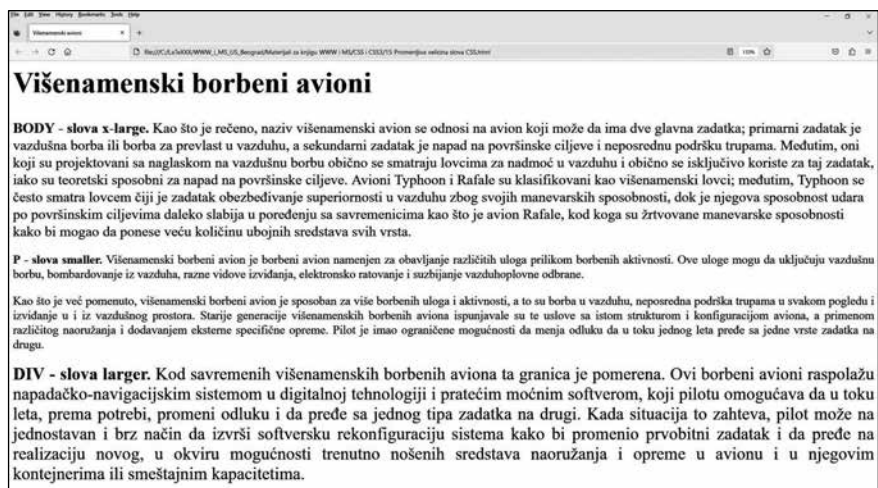


Slika 11.16. Specifični fontovi¹⁴

¹⁴ 14 Specijalizovana varijanta slova CSS.html

Dve stvari bi trebalo zapamtiti prilikom odlučivanja o veličini fonta za veb stranicu. Prvo, trebalo bi se ograničiti se na dve ili tri veličine po strani; previše različitih veličina zbuni čitaoce i stranice izgledaju pretrpano i neorganizovano. Isti efekat može da se dobije i izborom previše različitih fontova. Drugo, veličine fontova bi trebalo da odgovaraju prostoru predviđenom za to na stranici. Korišćenje malih veličina slova u velikom praznom prostoru može da bude dezorijentišuće. Jednako neatraktivno je primena velikih slova u malom prostoru.

Trebalo bi povesti računa o još jednoj stvari kada se razmatra veličina fontova. Reč je o veličini monitora korisnika i njihova rezolucija. Uobičajeno je da se prave veb stranice za veličine monitora od 17 do 19 inča, koristeći 800×600 ili 1024×768 rezolucije, a rezultat će verovatno zadovoljiti blizu 96% ljudi koji posete veb sajt. Oni će koristiti monitore slične veličine, kao i slične veb pretraživače. Ovo nije razlog za brigu. Ako autor pokušava da zadovolji sve oprečne zahteve (različite veličine monitora, različite rezolucije, softver otvorenog koda, prenosni uređaj sa malim ekranima, različiti operativni sistemi), neće zadovoljiti nikoga. Preporuka je da se "prati" većina i da se vidi šta većina korisnika ima na raspolaganju.



Slika 11.17. Promenljiva veličina slova¹⁵

Razlike u veličinama i rezolucijama monitora većine ljudi nisu ogromne, ali trebalo bi koristiti relativne veličine i pozicije, unutar veb stranice, kad god je to moguće. Na ovaj način, dobijaju se najpredvidljivije proporcije na većini pretraživača. Evo CSS vrednosti koje mogu da se dodele atributu `font-size` kako bi se definisala veličina fonta: `xx-small`, `x-small`, `small`, `medium`, `large`, `x-large`, `xx-large`, `smaller`, `larger`, `length`, `percentage` i `inherit`.

Vrednost `xx-large` je najveća vrednost na listi mogućih vrednosti za veličinu CSS teksta. Međutim, ako se zatim primeni `larger` vrednost, onda veb pretraživač neće da prikaže još veći tekst, nego će i da ga smanji.

¹⁵ 15 Promenljiva velicina slova CSS.html

Takođe, mogu da se izbegnu relativističke specifikacije do sada raspravljane i veličina slova može da se definiše u mernim jedinicama, kao što su 0.5 inča ili 12pt. Sledeći primer definiše da se tekst ispisuje fontom čija je veličina dve desetine inča, a da se naslov prvog nivoa (h1) ispisuje fontom veličine 24pt:

```
body {font-size: .2in;}
h1 {font-size: 24pt;}
```

Tokom dosadašnje pisanije videlo se da postoji mnogo načina kako da se definiše kako će da izgleda tekst i koje će osobine da budu aktivne:

```
p {font-family: Calibri; font-size: 24px;
font-weight: bold; font-style: italic;}
```

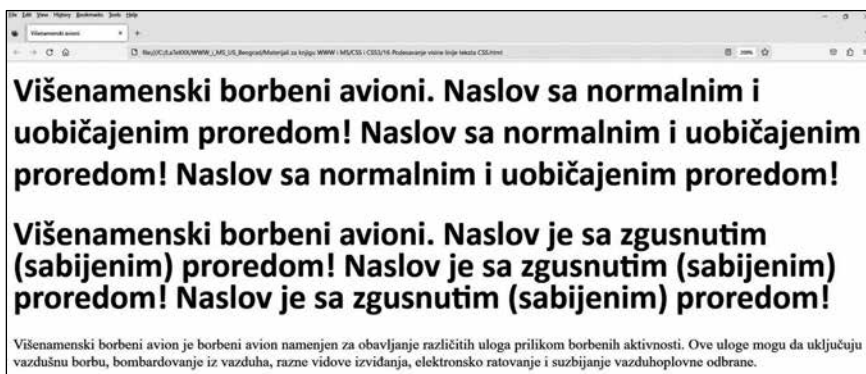
Ali, ako autor voli da koristi skraćenice, to je njegova stvar, ali trebalo bi probati kada su u pitanju font atributi. Prethodna sintaksa za p može da se kondenzuje na sledeći način:

```
p {font: bold italic 24px Calibri;}
```

Oba stila će imati isti rezultat, ali kondenzovano verzija štedi malo vremena ako se autor seća pravila! A pravilo glasi da prve dve vrednosti u listi (style i weight) mogu biti navedene bilo kojim redom. Na primer, prethodna sintaksa može da se napiše i na sledeći način, bez ikakvih posledica na izgled teksta u veb stranici:

```
p {font: italic bold 24px Calibri;}
```

Visine linija teksta su prilagođene iz dva osnovna razloga. Naslovi često izgledaju bolje sa manje praznog prostora između redova; tekst sa fontom manje veličine će se bolje čitati sa više belog prostora između redova.

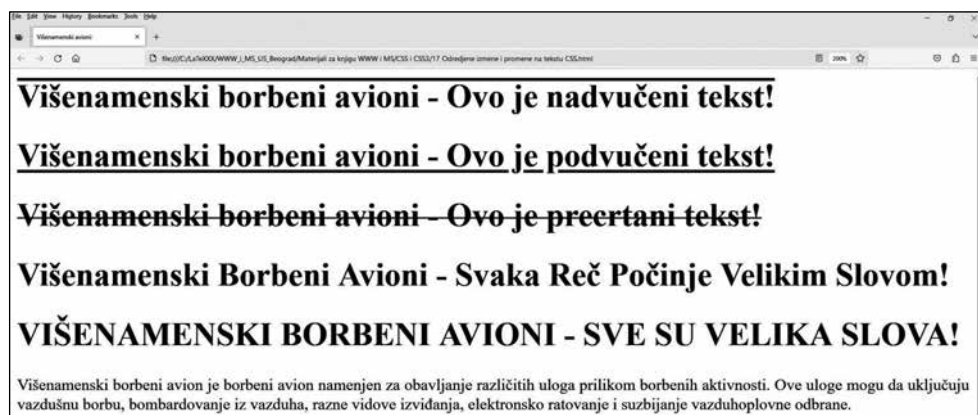


Slika 11.18. *Podlašavanje visine linije teksta*¹⁶

Kao što se može videti na slici 11.18, prilagođena visina linije teksta na dnu pretraživača izgleda bolje nego podrazumevana na vrhu. Novine i časopisi, a da se ne pominju TV, bilbordi i drugi mediji, gotovo uvek smanjuju prored u svojim naslovima.

¹⁶ 16 Podlašavanje visine linije teksta CSS.html

Korisnici programa *Micorsoft Word* su navikli na neke nazive efekata na tekstu, ali CSS malo odstupa od "odomačenih" naziva. Na primer, u *Word*-u precrtavanje slova se naziva *striketrough*, ali u CSS-u ta vrsta efekta "nosi" naziv *line-through*. Autori i korisnici bi trebalo da se naviknu na ovo i da budu spremni da prihvate određene izmene i promene. Primena uobičajenih efekata na otkucani tekst prikazana je na slici 11.19.



Slika 11.19. Određene izmene na tekstu¹⁷

U CSS stilu ili pravilu, može da se opiše pozicija i veličina na mnogo načina, koristeći različite **merne jedinice**. Često mogu da se izaberu proizvoljne jedinice, mada neke specifične vrste mernih jedinica najbolje rade u određenim situacijama.

Na primer, mnogi dizajneri koji rade sa papirom i olovkom koriste tačke (*points*) za određivanje veličine fontova. Tačka je apsolutna dužina i iznosi 1/72 inča. Korišćenje apsolutnih veličina slova prilikom specifikacija za magazin ili knjige je dobro – na kraju krajeva, korisnik ne može da smanji, isteže ili promeni razmeru (odnos visine i širine) stranice u knjizi. To kod pretraživača može da se menja po volji. Ako se "vuče" jedna strana pretraživača, korisnik ga menja i prozor pretraživača postaje tanji ili širi (deblji). Drugim rečima, menjanjem oblika prozora pretraživača menja se i razmera stranice.

Sledi spisak i objašnjenje svih jedinica koje mere dužinu:

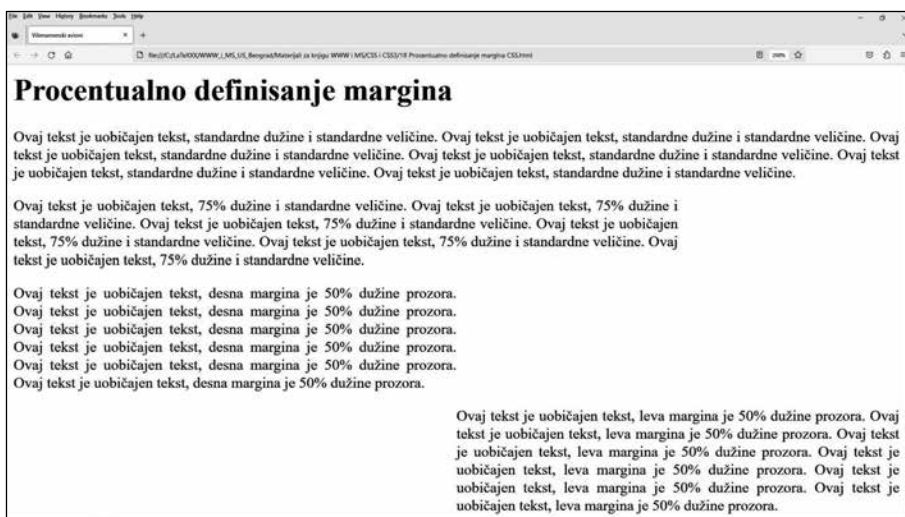
- px (*pixels*, pikseli) – pikseli su najmanja jedinica na displeju – ove tačkice nekad mogu da se vide ako se korisnik približi dovoljno blizu TV-u ili monitoru; na primer, postavljanjem rezolucije monitora na 800 × 600 piksela, znači da je na tom monitoru prikazano 800 piksela po širini i 600 piksela po visini.
- pt (*points*, tačke) – tačka je jednaka 1/72 inča; *points* i *picas* su klasični tip merenja slova, a kod većine pretraživača uobičajena je veličina fonta 12 pt.
- pc (*picas*, pajkovi) – jedan pajk iznosi 12 pt.
- mm (*millimeters*, milimetri) – milimetar je 0,03937 inča, tako da jedan inč sadrži 25,4 mm; jedan centimetar sadrži 10 mm i veći deo sveta koristi ovaj metrički sistem.

¹⁷ 17 Određene izmene i promene na tekstu CSS.html

- **cm** (*centimeters*, centimetri) – centimetar je 0,3937 inča, tako jedan inč sadrži 2,54 cm.
- **in** (*inch*, inč) – inč je jedinica koju koriste SAD, Velika Britanija i još neke zemlje. Jedan inč iznosi 25,4 mm.
- **em** – ovo je jedinica mere koja potiče od približne širine slova *m* tekućeg fonta; ovo se generalno smatra najboljim načinom za definisanje dužina.
- **ex** (*the x-height*, visina slova *x*) – predstavlja visinu malog slova *x* tekućeg fonta; veb pretraživači obično dele vrednost *em* sa 2 kako bi dobili vrednost *ex*.
- **%** (*percent*, procenat) – procenti su odlični za određivanje relativne veličine.

Stručnjaci preporučuju da se *em* koristi prilikom dizajniranja veb stranice, jer je fleksibilna. Jedinice *em* i *ex* su relativne u odnosu na svaku familiju fontova. Ovo je korisno jer to znači da se veličina određena pomoću *em* menja na precizan način, na osnovu rezolucije korisnikovog monitora, prioriternih postavki i drugih faktora. Drugim rečima, koristeći *em* omogućeno je da se odredi šta se dešava u odnosu na familiju fontova. Rezultat je proporcionalan ostalim kvalitetima sloga i okolnog teksta. Takođe, relativne specifikacije primenom *em* omogućavaju ljudima sa hendikepom da uvećavaju slog u svom pretraživaču po potrebi. Fiksne specifikacije kao što su *px* ili *pt* ne nude korisniku ovu opciju.

Ako je autor ili korisnik upoznat sa klasičnim HTML procentualnim specifikacijama (na primer, `<p width='80%'>`), neće imati problema sa razumevanjem kako procenti rade u CSS-u. Definisani HTML kôd u stvari kaže da se formira paragraf širine 80% (četiri petine) od ukupne širine prozora veb pretraživača. Drugim rečima, procenat je u specifičnom odnosu prema pripadajućem "kontejneru" ili "roditelju". CSS procenti rade na isti način.



Slika 11.20. Definisanje i podešavanje margina¹⁸

¹⁸ 18 Procentualno definisanje margina CSS.html

Boja je korisno i za projektovanje i organizovanje veb stranice. Izbor efikasnih, odgovarajućih i harmoničnih boja dosta doprinosi ukupnom uspehu veb stranice. Možda je manje očigledna uloga koju boju igra u privikavanju oka na strukturu i logiku primenjenog rasporeda.

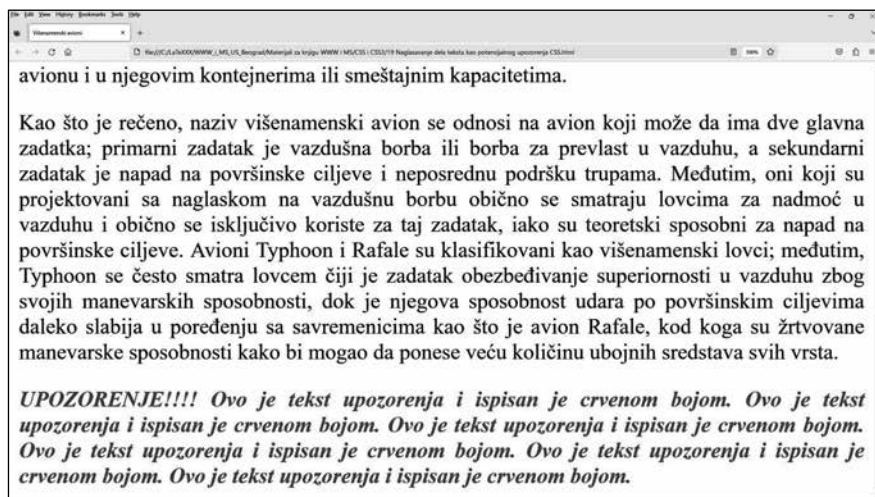
Mogu da se kreiraju logičke zone sa bojom, baš kao što se to radi tabelama ili linijama. Na primer, može da se pomogne korisnicima bojenjem linkova istom bojom ili njihovim grupisanjem u posebnoj oblasti na veb stranici, sa zajedničkom pozadinskom bojom. Sve ovo pomaže da posetilac sajta shvati da svi naslovi u ovoj oblasti, sa primenjenim bojama, imaju nešto zajedničko. A ako autor zaista želi da im privuče pažnju, onda bi trebalo da izabere neku specifičnu i nesvakidašnju boju.

Mnogi od najbolje dizajniranih sajtova imaju učitane pozadinske slike, jer stvaranje takve slike u grafičkom programu kao što je *Photoshop* daje autoru kontrolu nad svakim pikselom pozadine. Na taj način autor može da kreira prijatan, složen dizajn bilo koje vrste, a ne da statička boja "izigrava" pozadinu:

```
<body background="Pogled-na-vodu.jpg">
```

Na primer, autor piše veb stranicu koja sadrži posebne stavove upozorenja, a želi da se ta upozorenja izdvajaju. Pored toga što će takav tekst da zakosi (*italic*), želeće i da ga ispíše crvenom bojom. Da bi se postigao ovaj efekat, trebalo bi definisati stil paragrafa koji se zove upozorenje, a zatim bi trebalo dodati zakošenje i specificirati boju:

```
<style> p.upozorenje {color: #FF0000; font-style: italic;} </style>
```



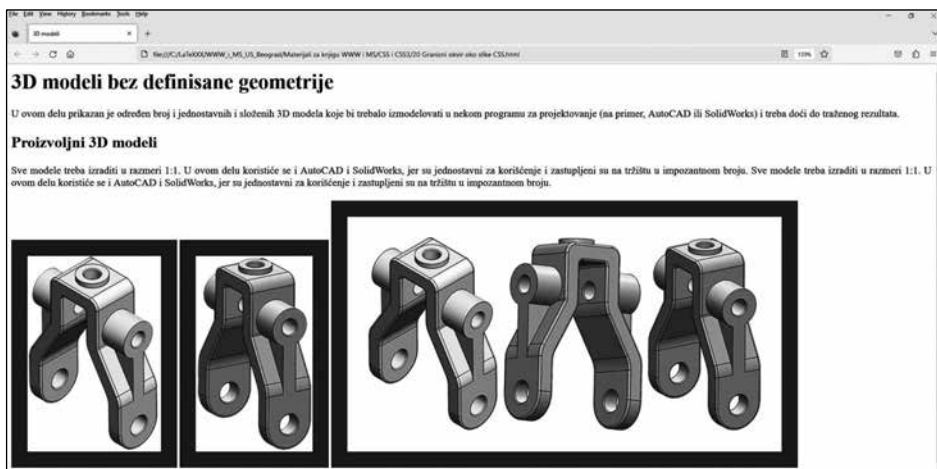
Slika 11.21. Naglašavanje dela teksta¹⁹

Boja pozadine se ne odnosi samo na tekst. Može da se koristi da se odredi **boja grafičnog okvira ili okvira**. U sledećoj sintaksi podešava se `color` atribut "na" sliku *sky-blue*. Alternativno, može da se koristi `border-color` atribut, ako autor to želi. Pored

¹⁹ 19 Naglašavanje dela teksta kao potencijalnog upozorenja CSS.html

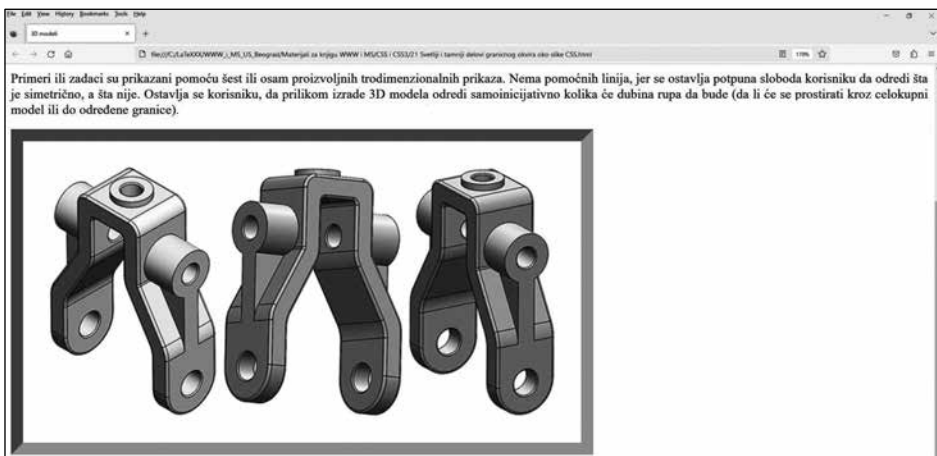
toga, može da se podese border-width kako bi se "proširila" granična linija okvira. Takođe, mora da se navede i atribut border-style:

```
<style>
img {border-style: solid; border-width: 25px; color: blue;}
</style>
```



Slika 11.22. Granični okvir oko slike unutar veb stranice²⁰

Ako autor želi da kreira suptilne, atraktivne granice, može da koristi komandu `inset` sa različitim bojama za okvir oko grafičkog elementa. To stvara zakošen izgled okvira (slika 11.23), koji je svakako korak napred u odnosu ravne, obične granične okvire prikazane na slici 11.22.



Slika 11.23. Svetliji i tamniji tonovi graničnog okvira oko slike²¹

²⁰ 20 Granični okvir oko slike CSS.html

²¹ 21 Svetliji i tamniji delovi graničnog okvira oko slike CSS.html

Kada se koristi *inset* dizajn potamnjuju se boje koje se koriste za levu i gornju graničnu liniju, a ostavlja ravne donju i desnu graničnu liniju. Tasteri, koji nisu aktivirani (primenjen *outset*), imaju svetlije gornju i levu graničnu liniju, u odnosu na tamnije donju i desnu granične linije. Zašto je to tako?

Kod grafičkih elemenata operativnog sistema *Windows* (meniji, dugmad, prozorski okviri i tako dalje), pretpostavlja se da svetlo "pada" iz gornjeg levog ugla ekrana. Onda je logično da neaktivirani elementi budu svetliji u gornjem levom uglu, a da senka pada ka donjem desnom uglu. Međutim, ako se objekat "pritisne" i on "utone" u pozadinu, onda se dešava suprotna situacija.



Slika 11.24. *Obrnuto svetliji i tamniji tonovi graničnog okvira oko slike*²²

U prethodnoj sintaksi trebalo bi promeniti atribut *inset* u *outset* i rezultat ove promene će biti vidljiv na slici 11.24:

```
<style>
img {border-right: silver outset; border-left: silver outset;
      border-top: silver outset; border-bottom: silver outset;
      border-width: 15px;}
</style>
```

Pozadina elementa uključuje i njegove granice i dodatne sadržaje ili proširenja. Opcija *padding* dodaje prostor oko elementa. Evo primera koji ilustruje kako utiču atributi *padding* i *background-color* na izgled dokumenta (slika 11.25).

Atribut *background-color* prihvata iste vrednosti boje kao što ih ima boja prednjeg (*foreground*) plana. Uobičajeno je da je pozadinska boja transparentna, tako da će se texture, slike ili drugi elementi, koji se nalaze ispod pozadine, videti. Takođe, pozadinska boja može da se dodeli skoro svakom elementu – od linijskih em elemenata do celokupnog `<body>` teksta.

Trebalo bi izbegavati "čudne" kombinacije boja. Neki dizajneri veb stranica u iskušenju da

²² Obrnuti svetliji i tamniji delovi granicnog okvira oko slike CSS.html

stvore "specijalne efekte", stvaraju grozomorne, neki bi čak rekli vulgarne, boje zakrpa na svojim stranicama. Osim ako se ima vrlo dobar razlog da se to uradi, trebalo bi izbegavati neisplanirane, detinjaste izgleda stranica, sa različitim zonama boje.

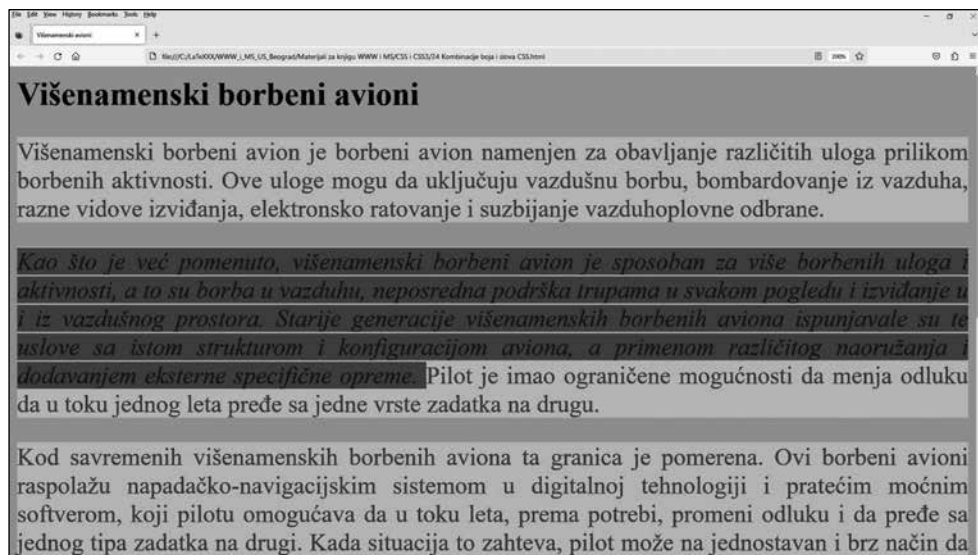


Slika 11.25. *Podešavanje pozadinske boje teksta*²³

Klasičan primer onoga što ne bi trebalo raditi je sledeća sintaksa:

```
<style>
body {background-color: skyblue;}
p {background-color: lightgreen; color: red; font-size: 24px;}
i {background-color: OliveDrab; color: blue;}
</style>
```

Ne samo da se takve stranice teško čitaju, nego su one jednostavno ružne, kao što to može da se vidi na slici 11.26. Ako CSS ne učini ništa drugo nego autoru dodeli kontrolu nad time kako će veb stranica da izgleda, to bi i dalje bilo ogromno poboljšanje u odnosu na HTML.



Slika 11.26. *Kombinacija boja slova i pozadine*²⁴

²³ 23 Pozadinska boja naslova CSS.html

²⁴ 24 Kombinacije boja i slova CSS.html

U ovom delu će biti reči o različitim načinima da se poboljša izgled veb stranica koje koriste posebne alate i tehnike. Može da se počne od opcije *kerning* – **podešavanja razmaka između karaktera unutar teksta**.

Opcija *kerning* – prilagođavanje prostora između karaktera – ponekad se pogrešno koristi za isticanje. Neki dizajneri veb stranica pokušavaju da "razvlačenjem" reči, primenom ove opcije, naglase neke stvari. Trebalo bi primeniti i ostale načine kako bi se neki tekst naglasio – podebljanje, boja, velika slova i slično. Ali praksa pokazuje da bi ove vizuelne trikove trebalo izbegavati kao način naglašavanja reči ili fraza. Za naglašavanje reči ili fraze trebalo bi koristiti *italic*, i samo *italic*.

Opcija *kerning* se koristi da "podesi" razmak između slova u naslovima. Autor može da smanji razmak između slova kako bi reči u naslovu bile zajedno i kako bi naslov bio kompaktniji. Većina fontova (osim *monospace* fontova kao što je *Courier*) imaju varijacije u razmaku između slova kako bi se poboljšala čitljivost familije fontova.



Slika 11.27. Podešavanje rastojanja između karaktera²⁵

Kao što se vidi na slici 11.27, kada se koristi negativna vrednost za letter-spacing (-0.08em u ovom primeru; redefiniše širinu slova m), karakteri u naslovu su zbijeniji. Tradicionalno, blago "sabijanje" prostora između karaktera u naslovima se smatra estetski superiornim u odnosu na nemodifikovanu verziju istog naslova. Ako se autor ili korisnik ne slažu da gornji naslov na slici 11.27 izgleda previše "labavo" sa podrazumevanim razmakom između slova, onda bi u sintaksi trebalo postaviti vrednost 0em , što znači da nema promena u odnosu na uobičajeni razmak između slova.

Na slici 11.28 je još vidljiviji efekat primene opcije *kerning*, jer je očigledno šta se dešava sa međuprostorom između pojedinih karaktera.

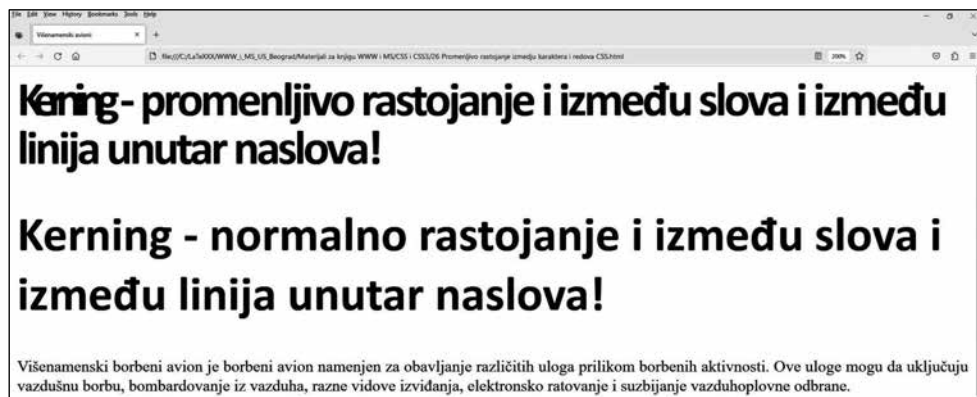
Kerning - pisanje sa ovim podešavanjem

Kerning - pisanje sa ovim podešavanjem

Slika 11.28. Kerning – vidljivi efekti

Ako se naslov sastoji iz više redova, onda bi trebalo da se autor pozabavi i vertikalnim podešavanjem proreda – smanjenje belog prostora između redova teksta, kao što je prikazano na slici 11.29.

²⁵ 25 Prilagodjavanje prostora između karaktera CSS.html



Slika 11.29. *Podlašavanje rastojanja između karaktera i redova*²⁶

Ako vrednost atributa `font-size` "prati" kosa crta i iza nje procentualna vrednost, onda to definiše procentualno prilagođavanje željene visine linije. U donjoj sintaksi, precizirana vrednost 99%, ali autor može da se "igra" sa razmakom dok ne dođe do izgleda koji njemu odgovara:

```
<style>
h1.spaced {font:bold 48px/.99 "Calibri"; letter-spacing:-.06em;}
h1 {font-family:Calibri; font-size: 48px; letter-spacing:-.06em;}
h1.normal {letter-spacing:normal;}
    .kern {letter-spacing:-.16em;}
    .ultrakern {letter-spacing:-.20em;}
</style>
```

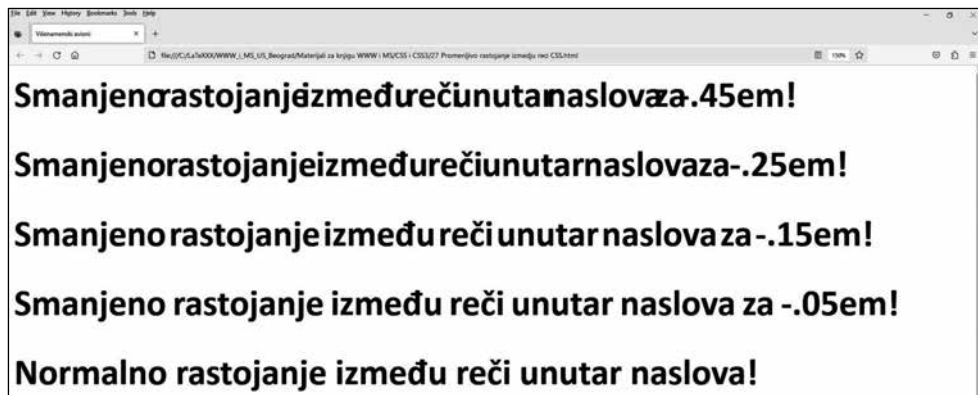
Atribut `word-spacing`, kao i `letter-spacing`, može da se koristi za kreiranje poravnatog (*justified*) teksta po obe margine. Međutim, veb dizajneri mogu dobro da iskoriste opciju *kerning* na naslove, kako bi smanjili nepotrebne bele razmake između reči. Evo primera koji smanjuje prostor između reči za 0,45em:

```
style>
h1.WordSpaced {font-family: Calibri;
                font-size: 48px; word-spacing: -.45em;} </style>
```

Kao i kod određivanja praznog prostora između karaktera i linija, i ovde mogu da se koriste bilo koje CSS jedinice za određivanje razmaka između reči. Međutim, jedinica `em` je obično najpouzdaniji ako, iz nekog razloga, veb pretraživač korisnika zameni izvorni font. Merenje pomoću jedinice `em` je najpreciznije merenje **prosečne** veličine karaktera za većinu fontove.

Ovde će biti malo više priče o različitim mogućim poravnanjima. Atribut `vertical-align` određuje kako se tekst poravnava po vertikali u odnosu na drugi element, kao što su tekst ili slike.

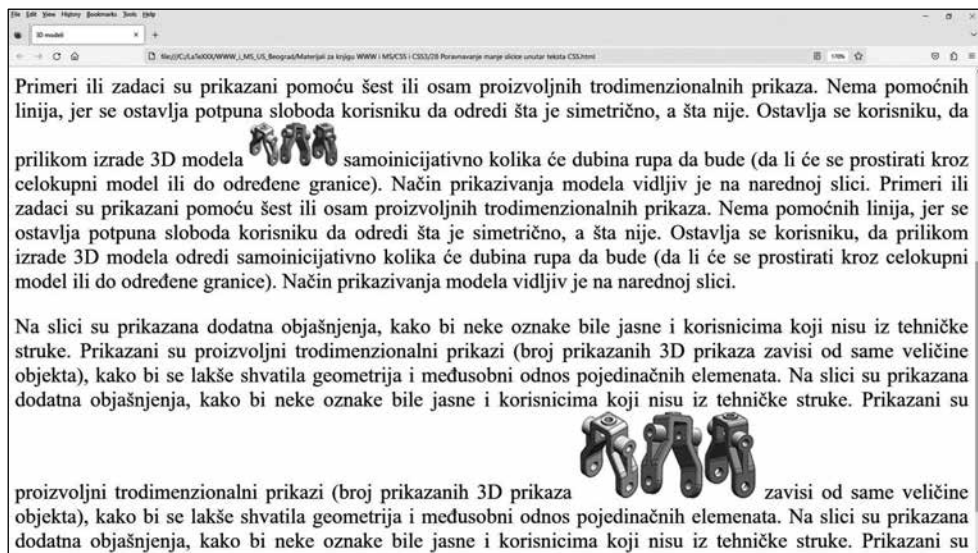
²⁶ 26 Promenljivo rastojanje između karaktera i redova CSS.html



Slika 11.30. Promenljivo rastojanja između reči²⁷

Atributu `vertical-align` može da se dodeli jedna od sledećih osam vrednosti: `bottom`, `baseline`, `middle`, `sub`, `super`, `text-top`, `text-bottom` i `top`. Ovom atributu mogu da se dodele u određene vrednosti, kao što su, na primer, 5 px ili procenti.

U okviru elemenata bloka, atribut `vertical-align` nema efekta u odnosu na blok. Međutim, taj atribut radi na usklađivanju elemente unutar ćelije tabele ili unutar parcijalnog teksta.



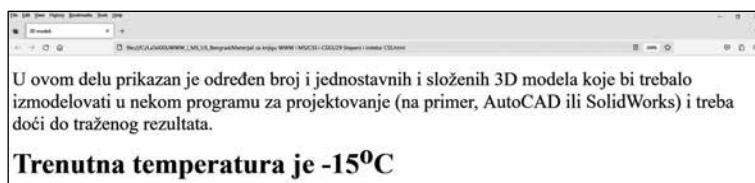
Slika 11.31. Poravnavanje sličica sa tekstom unutar paragrafa²⁸

Ako autor želi da koristi stepene (*superscript*) ili indekse (*subscript*), trebalo bi da ima na umu da se veličina teksta ne smanjuje automatski. Ako se želi taj efekat, onda bi morala

²⁷ 27 Promenljivo rastojanje između reci CSS.html

²⁸ 28 Poravnavanje manje slicice unutar teksta CSS.html

da se navede manja veličina fonta. Da bi se dobio željeni efekat (prikazan na slici 11.32), trebalo bi kombinovati stepene sa procentualnim smanjenjem atributa `font-size`.



Slika 11.32. *Unošenje stepena i indeksa u veb dokument²⁹*

Mogu da se odrede i apsolutna rastojanja kada se koristili poravnavanje po vertikali. Dovoljno je koristiti bilo koju od uobičajenih mernih jedinica, kao što su px, inch itd.

Podešavanje (poravnavanje) teksta po horizontali pomoću atributa `text-align` je slično poravnanju po vertikali, osim što se koriste vrednosti `left`, `center`, `right` i `justify`, a rezultati se odnose na celokupni paragraf, a ne na pojedinačne reči.

Vrednosti, u suštini, propagiraju ista poravnanja koja mogu da se pronađu u bilo kom programu za obradu teksta. Uobičajeno poravnanje je poravnanje po levoj margini (*left*). Atribut `text-align` može da se koristi samo sa elementima na nivou bloka, kao što je element paragrafa `<p>`.

HTML nije imao sposobnost uvlačenja prvog reda paragrafa ili pasusa, tako da su ljudi pribegavali ubacivanju nevidljivih slika i drugim trikovima. Dodavanje prostora ne radi zbog toga što je kod HTML-a isključena opcija za dodavanje razmaka. CSS je odlično reagovao sa svojim atributom `text-align`, po tom pitanju. Kada se ovaj atribut koristi sa elementima na nivou blokova kao što je `<p>`, onda može da se odredi veličina za uvlačenje prvog reda kao, na primer, 6em, 5px ili procentualna vrednost. Na slici 11.33 je prikazano kako se definiše vrednost uvlačenja prvog reda paragrafa ili pasusa.



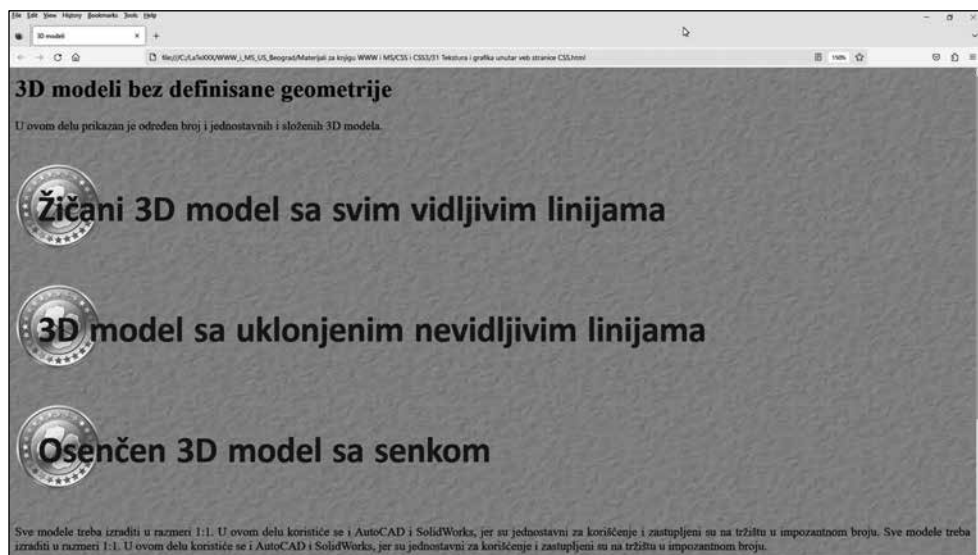
Slika 11.33. *Uvlačenje prvog reda paragrafa³⁰*

Većina veb sajtova izgledaju najbolje sa nekom pozadinskom **teksturom i grafikom**. "Obične" pozadinske boje deluju prilično "grubo", ali lepa, uzdržana, bleđa tekstura iza

²⁹ 29 Stepene i indeksi CSS.html

³⁰ 30 Uvlačenje prvog reda paragrafa ili pasusa CSS.html

teksta unapređuje mnoge veb stranice. Dobro postavljena pozadinska grafika može samo da unapredi veb stranicu. Tekstura, takođe, pomaže da se ujedine stranicu, povezivanjem različitih zona. Na primer, može da se kreira pozadina od male, ponavljajuće teksture. Primer na slici 11.34 pokazuje pozadinu sastavljenu od "pločice" veličine 236×214 piksela, kao deo teksture koja se pruža preko celokupne pozadine pretraživača. Uz operativni sistem "dolaze" i mnoge teksture, ali autorove teksture (koje on napravi sâm) su prilično superiorne u odnosu na one koje su isporučena sa operativnim sistemom *Windows*.



Slika 11.34. *Primena teksture i prateće grafike*³¹

Autor može da podesi pozadinu za celu stranu (u `<body>` elementu), ili za pojedine elemente, kao što je naslov (`h2`) u sintaksi ispod. Pomenuta sintaksa je:

```
body {background: white url("Pozadina.jpg") repeat;}
h2 {background: url("Novcic.png") no-repeat left top; color: blue;
    width: 85%; padding: 30px; font: bold 48px "Calibri";
    letter-spacing: -.01em; word-spacing: 0.06em; line-height: 58px;}
```

Trebalo bi primetiti da je fotografija novčića uključena kao deo definicije `h2` elementa, tako da se pojavljuje u gornjem levom uglu svakog `h2`. Međutim, dodatne osobine kao što su `width` i `padding` definišu razmak naslova, a određena svojstva se koriste za naglašavanje teksta naslova.

To je slično načinu na koji osobine fonta mogu da imaju višestruke vrednosti. Međutim, svojstva fonta mogu da se individualizuju ako autor to želi (na primer, `font-size`), a takođe mogu da se pojedinačno navedu svojstva pozadine, ako autor želi, na sledeći način:

```
body {background: white url("pozadina.jpg") repeat;}
```

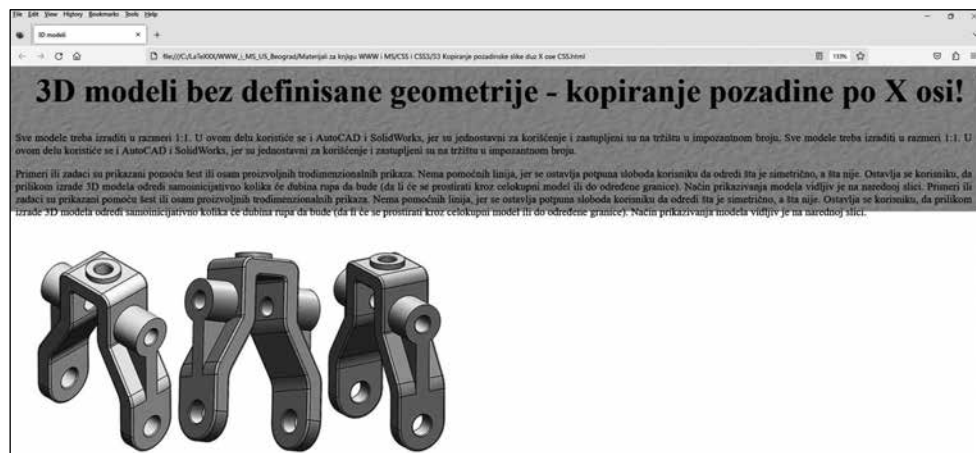
³¹ 31 Tekstura i grafika unutar veb stranice CSS.html

Osim ako nije drugačije određeno, pozadinska slika se **ponavlja i po vertikalni i po horizontali**, dok se ne ispuni prozor. Međutim, ako autor želi da se tekstura prostora po levoj margini, odozgo na dole, onda se to postiže vrednošću `repeat-y`, kao što je to prikazano na slici 11.35.



Slika 11.35. *Kopiranje pozadinske slike po vertikalni*³²

Međutim, ako autor želi da se tekstura prostora od leve margine, sa leve na desnu stranu, onda se to postiže vrednošću `repeat-x`, kao što je to vidljivo na slici 11.36 ispod.



Slika 11.36. *Kopiranje pozadinske slike po horizontali*³³

Atribut `background-position` može da ima sledeće vrednosti: `left`, `middle`, `right`, `top` i `bottom`. Ove vrednosti mogu i da se kombinuju na sledeći način:

`background-position: top right;`

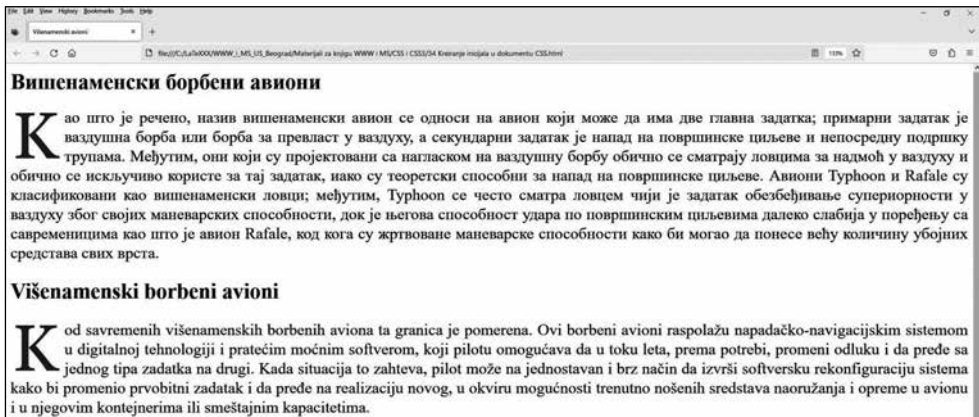
³² 32 Kopiranje pozadinske slike duž Y ose CSS.html

³³ 33 Kopiranje pozadinske slike duž X ose CSS.html

11.4. CSS i vizuelni efekti

Dobro osmišljena veb stranica je "čudo", ali to ne traje zauvek. Trenutno se veb stranice takmiče sa časopisima i drugim statičkim medijima. Međutim, ovo neće uvek biti slučaj. Na kraju, veb stranice mora da se "bori" protiv uzbuđenja koje nudi televizija i drugi aktivni, dinamični, animirani mediji. U stvari, jednog dana mogu da se "stope" veb i TV u jedan medij. Na kraju krajeva, piksel je piksel, a sve što razdvaja internet od televizije su neka hardverska ograničenja, neke stare navike koje se malo teže "probijaju" i, iznad svega, činjenica da su TV produkcija i veb dizajn, za sada, 2 različita radna mesta.

Inicijal (*drop cap*) na slici 11.37 je napravljen korišćenjem sledeće tehnike. Trebalo bi definisati klasu koja će specificirati font koji je veći od svog roditeljskog elementa, koji pluta na levoj strani svog roditelja i, ako autor želi, može da ima drugu boju. U kodu unutar fajla pod brojem 34 je definisan element dropcap koji se ispisuje podebljano (*bold*) i koji je 400% veći od okolnog paragrafa teksta. Onda bi trebalo koristiti element `<span>` da se definiše početno veće slovo.



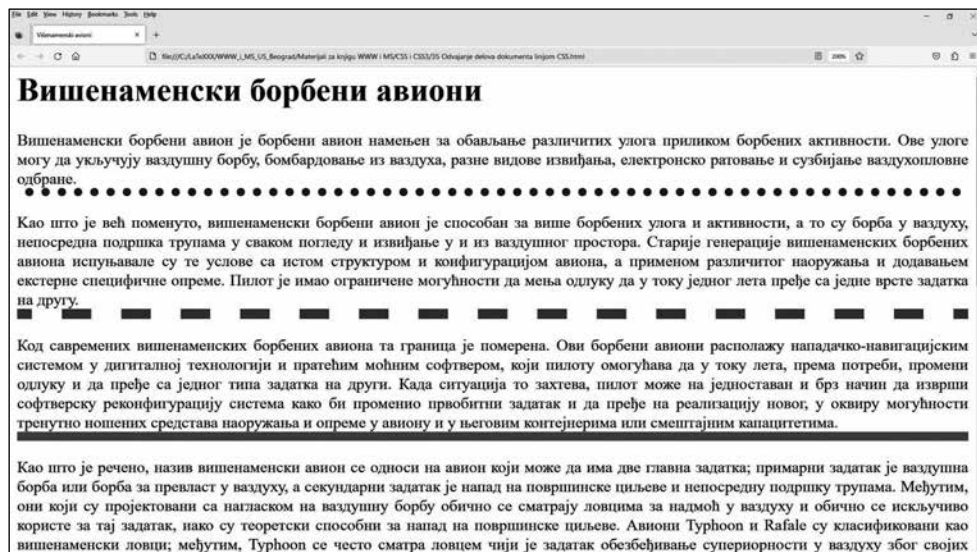
Slika 11.37. Kreiranje inicijala³⁴

Svaki vizuelni element u CSS-u ima imaginarni "**prazan pravougaonik**" oko njega, koji uzrokuje da zauzima više prostora na stranici od njenog sadržaja (tekstualnih ili grafičkih elemenata). Drugim rečima, autor može da opciono doda *padding*, granične okvire i margine elementu, kako bi se proširilo područje koje element koristi i kako bi se poboljšao sveukupni izgled dokumenta.

Ako autor želi da paragraf ili pasus budu podvučeni ili odvojeni jednom linijom koja je sastavljena od tačaka, onda su u igri vrednost `border-bottom` i `border-top`, s tim što naredna sintaksa koristi vrednost `border-bottom`, a primena ove sintakse vidljiva je na slici 11.38:

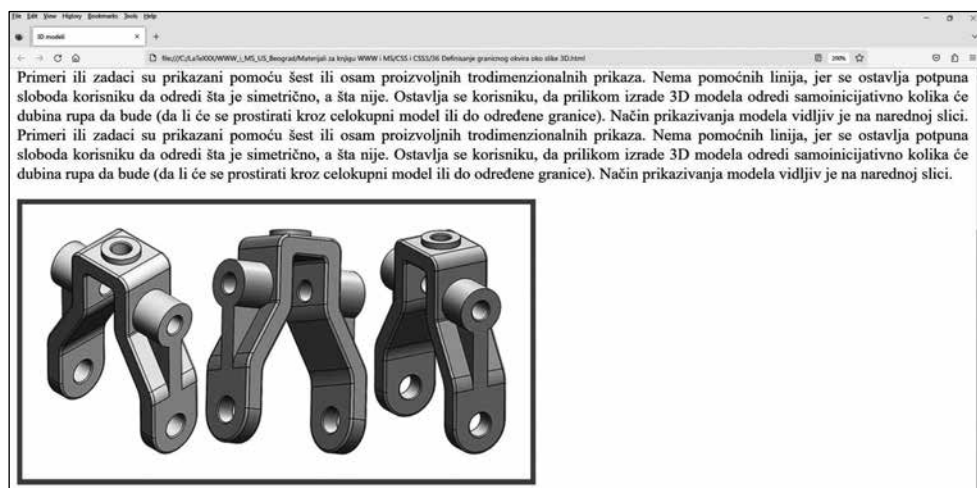
```
<p style="border-bottom: 8px blue dotted" align="justify">
<p style="border-bottom: 10px green dashed;" align="justify">
<p style="border-bottom: 9px red solid;" align="justify">
```

³⁴ 34 Kreiranje inicijala u dokumentu CSS.html



Slika 11.38. *Odvajanje paragrafa linijom*³⁵

Ako se eksplicitno ne navedu elementi pozicioniranja, elementi veb stranica se "slažu" koristeći podrazumevani tok – pretraživač jednostavno gomila elemente duž leve strane ekrana u cilju da ih locira u HTML kodu, kao što je prikazano na slici 11.39.



Slika 11.39. *Granični okvir oko slike*³⁶

Ovo podrazumevano pozicioniranje je stalno aktivno, osim ako se izričito ne koristi komande za pozicioniranje, koje će narušiti podrazumevano ili uobičajeno slaganje. Uočajeno

³⁵ 35 Odvajanje delova dokumenta linijom CSS.html

³⁶ 36 Definisanje granicnog okvira oko slike CSS.html

je da visina elementa, na najosnovnijem nivou, određuje sadržaj elementa. Na primer, fotografija 3D modela na slici 11.39 je ove veličine u pretraživaču jer je tolika u stvarnosti. Veličina odgovarajuće datoteke (iskorišćena u fajlu pod brojem 36) definiše ovu sliku. Slično tome, može da se utiče na visinu paragrafa teksta tako što će se suziti paragraf, a isti će da "poraste" u visinu kako bi se prikazao sav tekst. Precizirajući da bi širina paragrafa trebalo da bude samo 60% (širine prozora pretraživača), može da se desi da sav tekst ne može da se vidi unutar "otvorenog" prozora pretraživača, nego će se dodeliti i traka za pomeranje (*scroll bar*) sa strane. Na slici 11.40 nema trake za pomeranje, jer je sav tekst stao unutar prozora.



Slika 11.40. *Smanjivanje širine pojedinih paragrafa*³⁷

Pored ovoga, može da se razmotri primer koji pokazuje da boja pozadine ilustruje atribut *width* paragrafa, a rezultat je prikazan na slici 11.41.



Slika 11.41. *Smanjivanje paragrafa i dodeljivanje pozadinske boje*³⁸

Ali ako se dodaju prazan prostor (*padding*) i margine paragrafu, onda, na taj način, može da se poveća širina (*width*) predviđenog prostora (*box*) za dati tekst (slika 11.42).

³⁷ 37 Suzavanje pojedinačnih paragrafa CSS.html

³⁸ 38 Suzavanje pojedinačnih paragrafa i promena njihove boje pozadine CSS.html



Слика 11.42. *Dodavanje međuprostora između paragrafa*³⁹

Ako se obrati pažnja i ako se uporedi sadržaj slike 11.41 sa slikom 11.42, očigledno je da paragraf na slici 11.42 ima širu boju pozadine (atribut `padding`) i takođe je postavljen dalje po horizontali od leve ivice prozora pretraživača (atribut `margin`).

Dizajneri veb stranica često pogrešno veruju da kada oni definišu atribut `width`, da oni navode udaljenost između njenih levih i desnih graničnih linija. Ili još gore, neki misle da definisana `width` vrednost označava celokupno vidljivo rastojanje između tekućeg i sledećeg elementa. U stvari, atributi `padding`, širina graničnih linija i margine moraju da se uzmu u obzir.

Horizontalne linije predstavljaju, često, veoma efikasan način odvajanja različitih logičkih oblasti na veb stranici. One organizuju stvari vizuelno za čitaoca na način koji je i nena-metljiv i efikasan. Takođe, može da se poboljša estetski izgled dodavanjem raznih linija.

Čitaoci su veoma zahvalni ako je moguće vizuelno "razbiti" stranicu koja sadrži veliku količinu teksta. Da bi se to uradilo, od velike pomoći su naslovi i podnaslovi, kao i granične linije, pozadinske boje i teksture. Ali jedan od najboljih načina da se ukaže na to da paragrafi pripadaju zajedno nekoj grupi je jednostvano ubacivanje horizontalne linije po potrebi. Verovatno najlakši način da se "uposli" horizontalna linija je da se koristi `<hr>` element, kao što je prikazano na slici 11.43.



Слика 11.43. *Odvajanje teksta horizontalnom linijom*⁴⁰

³⁹ 39 Dodavanje međuprostora između pasusa CSS.html

⁴⁰ 40 Odvajanje teksta horizontalnom linijom CSS.html

Ako autor želi da eksperimentiše, može da proba sa atributom `background-image` i da ga transformiše u liniju razdvajanja sa grafikom. Međutim, ako želi da razdvoji paragrafe slikama, trebalo bi voditi računa o dve stvari:

- Trebalo bi koristiti `<img>` element da "uposli" grafiku, radije nego da "uvrće" horizontalne elemente i da se desi nešto nepredviđeno.
- Trebalo bi pažljivo razmisliti o vizuelnom efektu korišćenja grafike umesto jednostavnih linija za razdvajanje paragrafa.

Ako autor ima neodoljivu želju da isprobava novitete, kompletan kôd se nalazi u fajlu pod brojem 41. Slika 11.44 ilustruje šta se dešava kada entuzijazam trijumfuje nad zdravim razumom.

Granice i granični okviri mogu da budu dobar način da se pomogne korisnicima da shvate izgled veb stranice, da shvate njene različite zone i njihove potrebe. Paragrafi su način se podeli tekst na logičke jedinice – čitalac zna da podela pomoću paragrafa ukazuje na novu ideju ili dalje širenje ideje iz prethodnog pasusa.



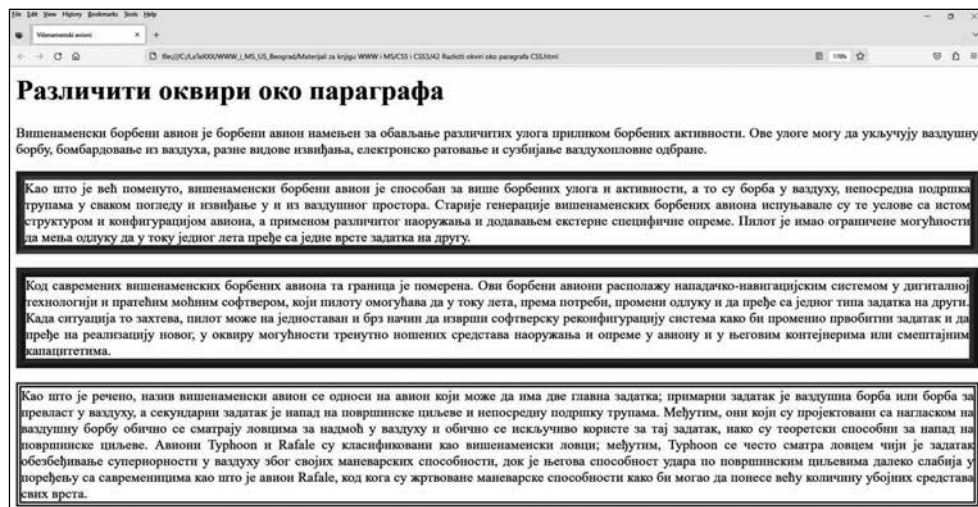
Slika 11.44. *Odvajanje teksta slikom ili fotografijom*⁴¹

Slično tome, sami paragrafi mogu se grupisati u veće logičke jedinice. Na primer, veb stranica može biti podeljena u deo koji opisuje kompaniju, drugi deo koji sadrži linkove ka raznim lokacijama unutar veb sajta i još jedan deo za reklamiranje najnovijeg proizvoda.

Ako je autor odlučio da "zaokruži" elemente veb stranice granicama ili graničnim linijama, onda je imao sreće; CSS nudi dosta načina da se iskoriste i primene granice i granične linije, a, takođe, omogućava da se dodaju granični elementi (granice) na bilo koji element.

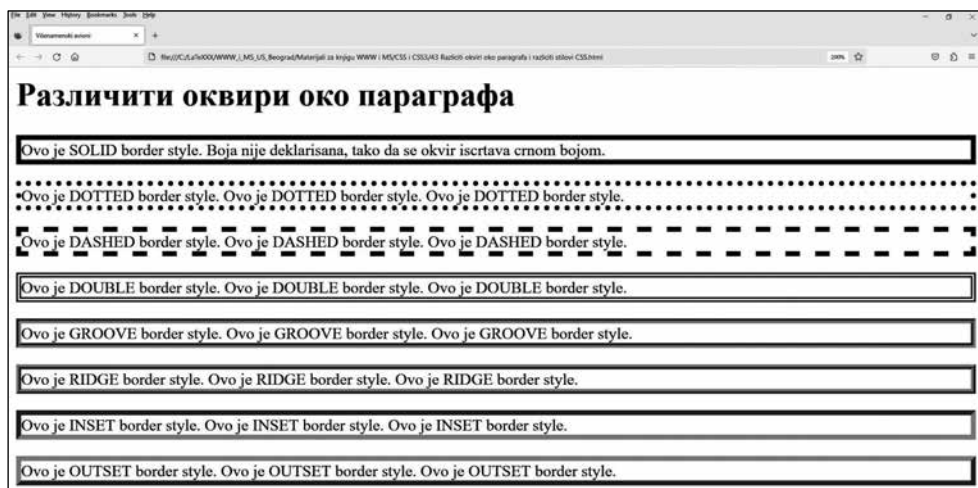
Autoru su na raspolaganju različite granične linije kada je rešio da koristi CSS. Najjednostavnija specifikacija granice (granične linije) samo koristi atribut `border` praćen serijom od tri vrednosti (`width`, `style` i `color`) odvojene praznim prostorom.

⁴¹ 41 Odvajanje teksta stilizovanom slikom CSS.html



Slika 11.45. *Različiti okviri oko delova teksta*⁴²

Može da se odredi osam različitih graničnih stilova – solid, dotted, dashed, double, groove, ridge, inset ili outset. Slika 11.46 ilustruje svaki stil, iako su neki od njih malo širi nego što se obično koriste. Unutar fajla pod brojem 43 svaka granična linija je široka 5 piksela, kako bi se linije lakše videle na slici u ovoj knjizi. Uopšteno govoreći, samo granični stilovi groove, ridge, inset i outset bi trebalo koristiti u "debljoj" formi.



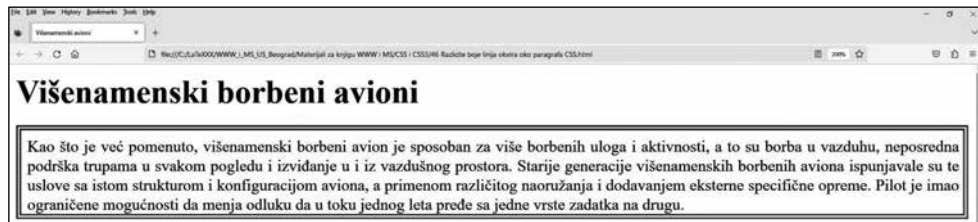
Slika 11.46. *Različiti stilovi okvira oko delova teksta*⁴³

Ako autor želi, može da koristi samo jednostranu granicu (ili dvostranu ili trostranu). Može čak i da meša stilove; na primer, jedna strana slike je ograničena isprekidanom linijom i

⁴² 42 Razliciti okviri oko paraagrafa CSS.html

⁴³ 43 Razliciti okviri oko paraagrafa i razliciti stilovi CSS.html

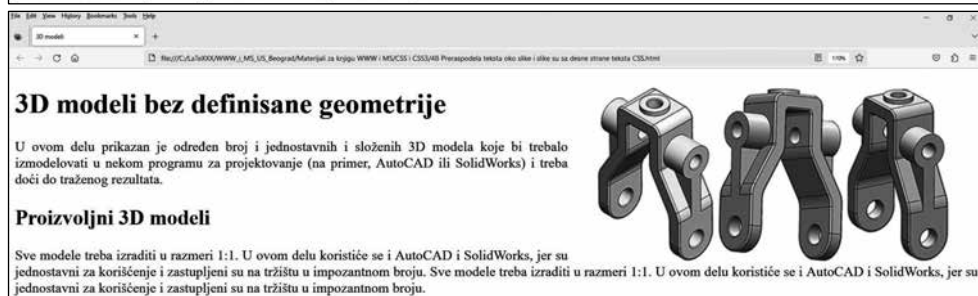
u standardnom HTML-u. Dizajneri su bili u mogućnosti da puste da tekst "teče" (*float*) ili da se "omota" (*wrap*) oko slike ili tabele pomoću koda `align = "right"` u HTML-u, ali sada može da primeni *float* na sve što želi. Očigledno je da CSS daje dizajnerima daleko veću slobodu u dizajniranju, nego što je to ranije bilo moguće.



Slika 11.49. *Različite boje okvira oko paragrafa*⁴⁶

Ali, kada se doda atribut `float`, tekst i drugi elementi okružuju "plutajuće" elemente. U sledećem primeru, slika je okružena i naslovom i tekстом koji ga prati, kao što je prikazano na slici 11.50 iznad. Kada se pogleda kôd u fajlu pod brojem 47 vidi se da slika predstavlja *inline* element, smešten unutar H1 elementa. Kada se element ovako "ugnezdi", to osigurava da većina veb pretraživača poravnava gornju ivicu teksta naslova i slike. Slika 11.50 ispod prikazuje promenu atributa `float`, upotrebom prethodnog koda, ali sa jednom promenom:

`float: right;`



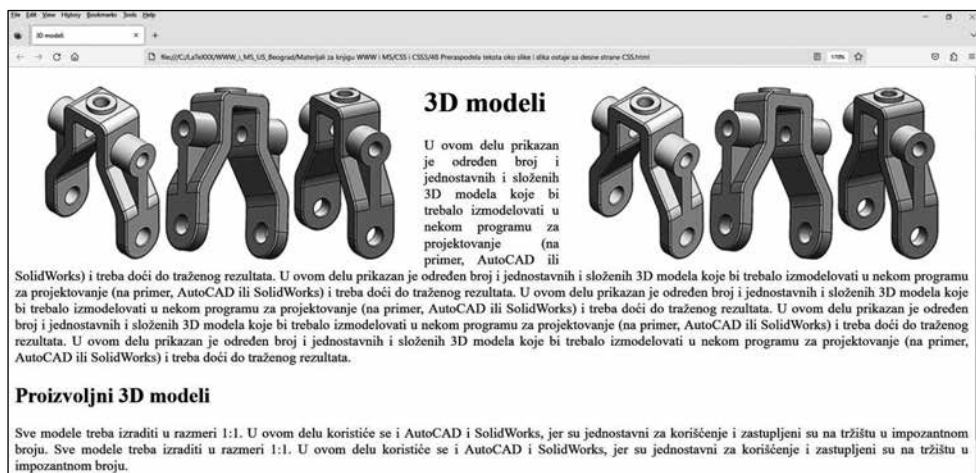
Slika 11.50. *Razmeštaj teksta oko slike i sa desne i sa leve strane*^{47 48}

⁴⁶ 46 Razlicite boje linija okvira oko paragrafa CSS.html

⁴⁷ 47 Preraspodela teksta oko slike i slika ostaje sa leve strane CSS.html

⁴⁸ 48 Preraspodela teksta oko slike i slike su sa desne strane teksta CSS.html

Ako autor želi da tekst "teče" i sa leve i desne strane, onda bi trebalo da definiše dva puta vrednost atributa `float`. Slika 11.51 ilustruje ovu dvostruku primenu atributa `float`.



Slika 11.51. Preraspodela teksta između slika i oko njih⁴⁹

Šta bi autor trebalo da uradi ako poželi da primora neki tekst da se odvoji od slike, kako bi normalno "tekao" okolo? Šta ako želi da tekst pomeri dole, ispod slike? Ponekad ne želi da tekst bude povezan sa određenom slikom. Jedan od problema sa primenom atributa `float` nastaje kada se ima nekoliko slika na stranici. Atribut `clear` može da pomogne u osiguravanju da se svaki element teksta pojavljuje uz sliku sa kojom je povezan ili na koju se odnosi.

CSS verzija "čišćenja" je slična tradicionalnom HTML elementu `<br>`, kada se koristi atribut `clear`. U CSS-u, trebalo bi koristiti atribut `clear` zajedno sa vrednostima `left`, `right` ili `both`. Samo korišćenje elementa `<br>` sa ovim CSS atributom "tera" tekst na silu da sledi `<br>` element i da se pomeri ispod svih postojećih margina; drugim rečima, tekst se pomera na dole do prve prazne oblasti na stranici.

Međutim, dodavanjem CSS atributa `clear` sa dodeljenom vrednošću `left`, pomera se tekst ispod "plutajuće" slike.

CSS element `box` (zamišljeni pravougaonik) je virtuelni, imaginarni oblik sastavljen od sadržaja elementa, kao i njegovog praznog prostora, graničnih linija i margina, ako ih ima. Dimenzije ovog pravougaonika se često ne poklapaju sa vidljivim sadržajem ili vidljivim granicama. Ako se koristi atribut `padding`, kutija raste u njegovu dimenziju. Ako se dalje dodaje granične linije ili margine, to takođe, povećava veličinu zamišljenog pravougaonika. Međutim, atribut `padding` nije vidljiv, jer samo stvara prostor između tekućeg elementa i okolnih elemenata ili okvira pretraživača.

Sledeći primer omogućava pretraživaču automatsko podešavanje veličine desne margine, kako bi se osiguralo da paragraf zadrži svoju fiksnu veličinu i na fiksnoj udaljenosti od

⁴⁹ 49 Preraspodela teksta oko slike i slika ostaje sa obe strane CSS.html

leve strane prozora pretraživača. Autor može individualno da odredi *left*, *right*, *top* i *bottom* margine, kao što je prikazano u donjoj sintaksi:

```
p {width: 750px; margin-right: auto; margin-left: 50px;}
```

Ovaj zanimljiv stil kaže da bi sadržaj trebalo da bude fiksiran na širinu od 750 piksela, kao i da je leva margina fiksna i iznosi 50 piksela. "Roditeljski" element je, u stvari, sâm prozor pretraživača (tehnički, roditelj je <body> elemenat ako nijedan drugi roditelj nije uključen, ali prozor pretraživača svakako izgleda kao roditelj). Drugim rečima, ukupna dužina od leve strane prozora pretraživača do desne strane tekstualnog paragrafa mora da se održava na 300 piksela. Desna margina, međutim, automatski se podešava kako bi održale ove dve druge širine fiksnim. Efekat atributa *auto* je da se zamrzne paragraf na određenju horizontalnoj lokaciji u prozoru pretraživača.



Slika 11.52. Ograničavanje veličine teksta u veb dokumentu⁵⁰

Na slici 11.61 se vidi efekat dodavanja CSS stila koji "angažuje" atribut *auto*. Evo koda koji izaziva zamrzavanje paragrafa u jednom položaju ("stabilna" ili fiksna udaljenost od leve strane prozora pretraživača roditelja):

```
<style>
p {width:750px; margin-right:auto; margin-left:50px;
  border:2px solid;}
</style>
```

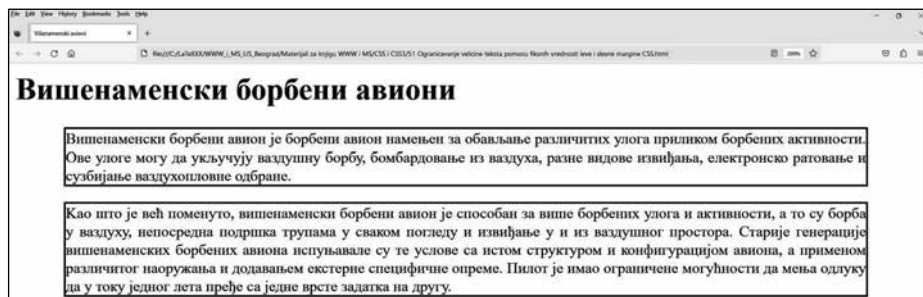
Slika 11.52 pokazuje da istezanje (promena veličine) prozora pretraživača ne razvlači i paragraf. Desna margina na ovoj slici se automatski prilagođava, bez obzira na veličinu neophodnu da se održi veličina paragrafa i pozicija.

Ako se navedu vrednosti leve (*left*) i desne (*right*) margine, ali se ne precizira širina, element će razvlačiti svoju širinu kako bi se održale fiksne tražene margine. Na primer, ako se navede da su i leva i desna margina 50 piksela, onda će atribut *width* paragrafa dobiti vrednost *auto*, ako to bude bilo neophodno.

```
<style>
p {width:auto; margin-right:50px; margin-left:50px; border:2px solid;}
</style>
```

⁵⁰ 50 Ogranicavanje velicine teksta u dokumentu CSS.html

Na slici 11.53 je prikazano dejstvo preciziranja leve i desne margine, ali izostavljanje specifikacije širine (*width*) izaziva da pretraživač pretpostavlja da se širina automatski podešava.



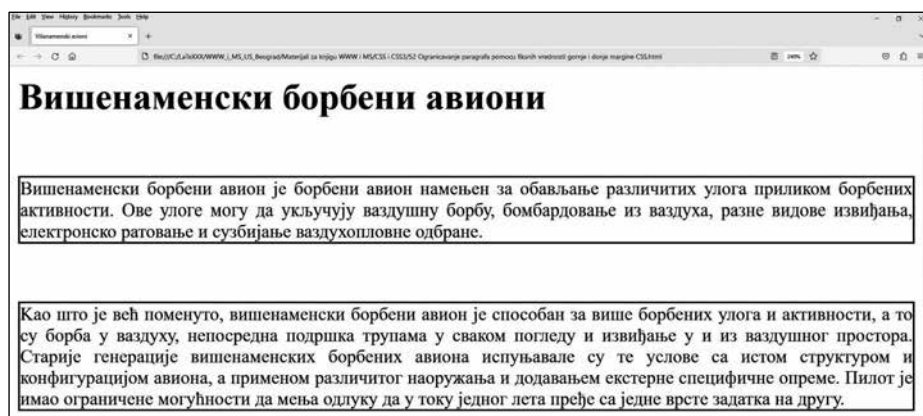
Slika 11.53. *Ograničavanje veličine teksta fiksnim vrednostima margina*⁵¹

Kao što mogu da se podese sva tri svojstva da imaju vrednost auto kada se vrši dimenzionisanje i pozicioniranje po horizontali (*width*, *left* i *right* margina), tako mogu, u teoriji, da se dodele vrednosti auto i ekvivalentnim vertikalnim atributima – *height*, *top* i *bottom* margine.

Autor može da "zamrzne" top poziciju jednog elementa koristeći sledeći kôd. Samo bi trebalo odrediti atribut *margin-top*, ali bi trebalo ignorisati visinu elementa:

```
<style>
p {width:auto; margin-top:50px; margin-bottom:50px; border:2px solid;}
</style>
```

Slika 11.54 pokazuje kako to izgleda kada se elemenat postavi na specifičnu vertikalnu lokaciju, ali njegova visina nije zamrznuta. Promena veličine prozora pretraživača (ili roditeljskog elementa) može da doda ili oduzme od visine elementa.



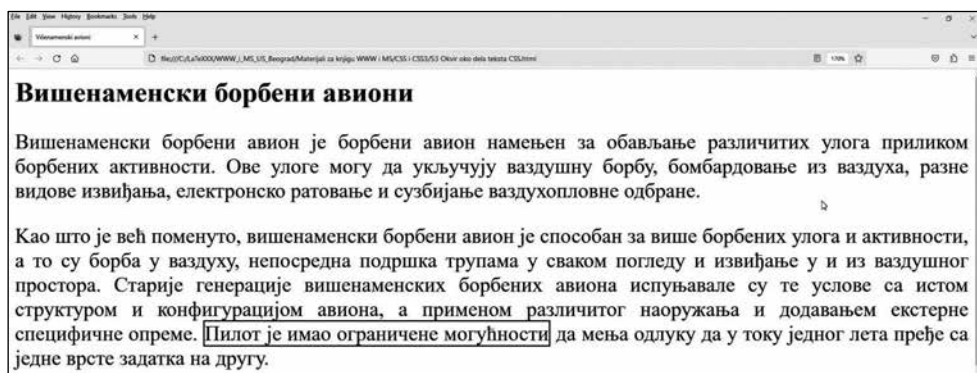
Slika 11.54. *Ograničavanje veličine teksta fiksnim vrednostima margina*⁵²

⁵¹ 51 Ograničavanje veličine teksta fiksnim vrednostima leve i desne margine CSS.html

⁵² 52 Ograničavanje paragrafa fiksnim vrednostima gornje i donje margine CSS.html

U CSS, redni (*inline*) element je mali elemenat koji je "usaglašen", na neki način, sa linijom teksta (kao što su elementi `<em>` ili `<i>`). Autor može da se "igra" sa *inline* elementima u tekstu na mnogo načina – crtanje graničnih linija oko pojedinih reči ili fraza, dodavanje beline, uključivanje margina, ubacivanje dodatnih velikih znakova ili reči, ubacivanje slike ili fotografije u sredinu paragrafa itd.

Manipulisanje tekstom, uopšteno govoreći, je najbolje prepustiti onima koji su i dizajnirali font ili familiju fontova. Dodatno dodavanje ugnežđenih okvira, granica, margina, velikih znakova ili grafike u red teksta može da dovede da je takav tekst neugledan, težak za čitanje i slično. Autor može da ima savršeno podešen paragraf i da ga "unakazi" dodavanjem *inline* elementa, u ovom slučaju `span` elementa, koji definiše granični okvir oko izabranog teksta, kao što je prikazano na slici 11.55.



Slika 11.55. Naglašavanje dela teksta njegovim uokviravanjem⁵³

Umesto ovoga, autor može da iskoristi boje i pomoću njih da istakne i naznači deo teksta. Evo koda koji proizvodi poseban efekat prikazan na slici 11.56:

```
<style> p {font-size: 24px;}
span {background: yellow;} </style>
```



Slika 11.56. Naglašavanje dela teksta pozadinskom bojom⁵⁴

⁵³ 53 Okvir oko dela teksta CSS.html

⁵⁴ 54 Naglasavanje dela teksta pozadinskom bojom CSS.html

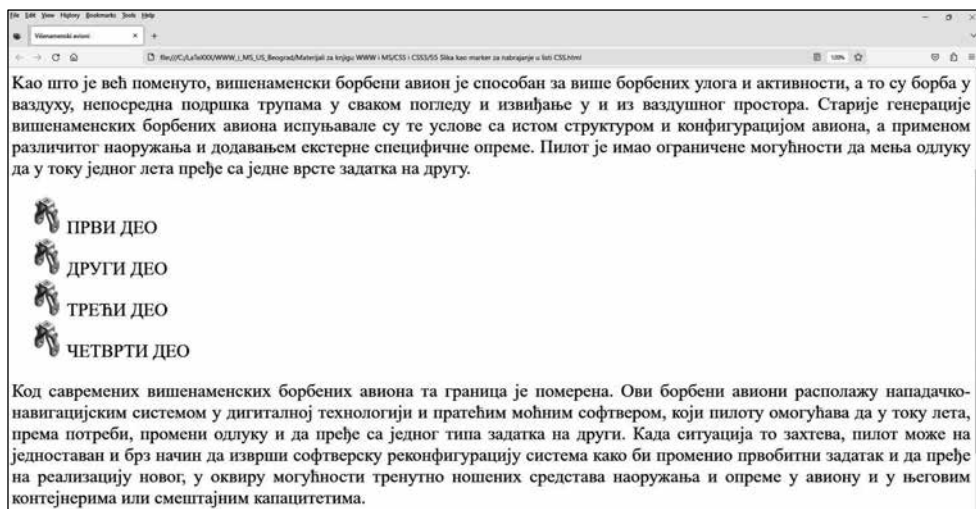
Sada se dolazi do klasike, prepodobne okosnice klasičnog HTML-a. Reč je o **tabelama**. Šta bi veb dizajneri ikada učinili bez tabela? Nalazili bi se unutar tradicionalnog izgleda veb stranice koji bi bio omeđen frejmovima i ne bi bio nešto preterano fleksibilan.

Liste su, naravno, "jednostavniji" rođaci tabela. Umesto stavki koje se šire u dva pravca, i po horizontali i po vertikalni, kao u tabeli, kod lista se horizontalne stavke "izlistavaju" kao da je reč o tabeli sa jednom kolonom. Pošto je reč o jednostavnijem formatu, listama obično "ne treba pomoć" u vidu graničnih linija u cilju razgraničenja pojedinačnih ćelija. Simboli za nabranjanje (*bullets*), dekorativni elementi (*dingbats*) ili brojevi generalno služe za vizuelno odvajanje komponenti unutar jedne liste.

CSS omogućava da se definišu liste na mnogo načina koji nisu moguće u HTML-u. Mogu da se koriste jedan od tri grafička simbola kao simboli za nabranjanje (*bullets*) stavki liste. Očigledno, čak i lista sa simbolima za nabranjanje nije nasumična. Koriste se tri simbola za nabranjanje – disc (najčešći simbol; reč je o crnoj ispunjenoj tački i to je podrazumevani CSS stil za liste); circle (prazan kružić); square (ispunjen kvadrat).

Sledi spisak 6 stilova za numeričko označavanje stavki unutar liste – decimal (1. 2. 3.); lower-roman (i. ii. iii.); upper-roman (I. II. III.); lower-alpha (a. b. c.); upper-alpha (A. B. C.); none (stavke u listi su uvučen kao i svim drugim listama, ali bez grafičke ili numeričkim simbolima).

Ako autor želi da koristite drugi grafički element kao simbol za nabranjanje, nešto uzbudljivije ili egzotičnije od priložene crne ispunjene tačke, praznog kružića ili ispunjenog kvadrata, onda bi trebalo da koristi atribut `list-style-image`, kao što je prikazano na slici 11.57.



Slika 11.57. Slike kao markeri u listi za nabranjanje⁵⁵

Trebalo bi posebno napomenuti da je nemoguće promeniti veličinu simbola za nabranjanje kada se koristi atribut `list-style-image` koristeći, na primer, vrednosti `height`

⁵⁵ Slika kao marker za nabranjanje u listi CSS.html

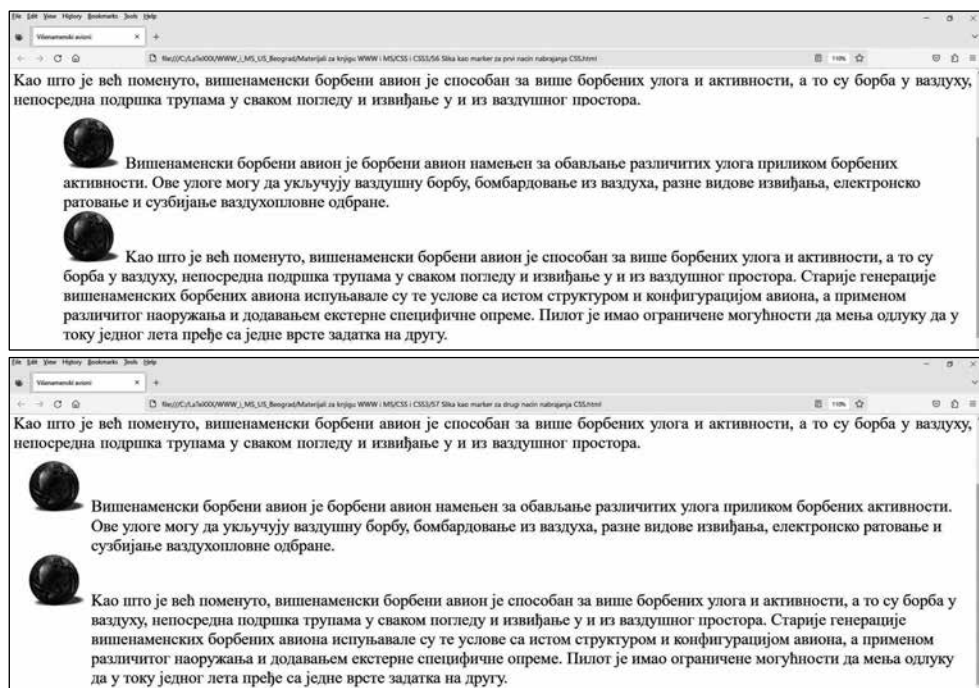
ili `width`. Umesto toga, veličina takve slike ili grafičkog simbola (elementa) mora da se podesi u nekoj grafičkoj aplikaciji, kao što su *Photoshop* ili *GIMP*.

Ako autor želi da prilagodi način na koji će simboli za nabranje biti definisani i pozicionirani u odnosu na tekst stavki (posebno kada stavke imaju više redova teksta), onda bi trebalo da definiše vrednosti `inside` ili `outside` unutar atributa `list-style-position`. Efekat je relativno mali, ali ako se izabere vrednost `inside`, dobiće se sličan efekat kao što su inicijali (*drop caps*). Donji deo koda proizvodi `inside` efekat prikazan na slici 11.58 iznad:

```
ul {list-style-position: inside}
```

Uobičajena vrednost ovog atributa je `outside` i rezultat njegove primene je prikazan na slici 11.58 ispod. Zašto je stil prikazan na slici 11.58 ispod (podrazumevan, uobičajen, *default*) najpopularniji? Izgleda bolje, a pomaže čitaocu na isti način kao što to rade simboli za nabranje ili brojevi – brzo prepoznaju stavke u listi.

Danas se **tabele** koriste u veb stranicama za daleko više stvari, nego što je prvobitno bilo planirano. Plan je bio da se jednostavno koriste za prikaz tabelarnih podataka i da se formiraju obične tabele sa informacijama. Međutim, tabele su služile godinama kao pomoćna mreža linija (*grid*) na kojoj će se graditi cela veb stranica.



Slika 11.58. Razmeštaj slika kao markera u listi za nabranje^{56 57}

^{56 56} Slika kao marker za prvi način nabranja CSS.html

^{57 57} Slika kao marker za drugi način nabranja CSS.html

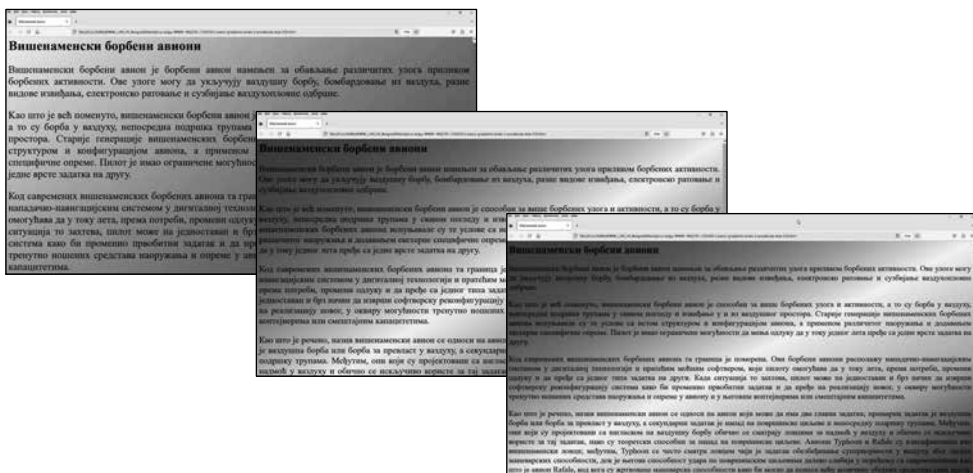
Osnovni problem je u tome što HTML nije pružio održiv način da se organizuju različite komponente ili vizuelne zone unutar stranice. Dizajneri koji rade sa časopisima, u oglašavanju i drugim oblastima očekuju da će moći da pozicioniraju elemente stranice upravo tamo gde žele i da to definišu do najsitnijih detalja.

CSS atribut `table-layout` omogućava se odredi da li ćelija može da promeni svoju veličinu kako bi se prikazao njen celokupni sadržaj ili će veličina ćelije da ostane fiksna. Podrazumevana vrednost je `auto`, što znači da će da se pretraživač bavi izračunavanjem veličine svake ćelije, pre nego što prikaže tabelu i njen sadržaj. Međutim, autor može da koristi fiksnu vrednost za `table-layout` atribut i da ubrza stvar navodeći maksimalnu veličinu za ćelije. Ovo može, međutim, da "odseče" deo fotografije.

Podešavanje rasporeda u više kolona (*multicolumns*) je možda najčešći način organizacije za veb stranice, baš kao što su prikazane kolone sa tekstom u novinama i časopisima. Kolone služe da organizuju sadržaje po horizontali, baš kao što elementi kao što su paragrafi, pravila i naslovi pomažu čitaocu da vide kako su informacije organizovane po vertikalni. Rezultat, u većini stranica, se svodi na pomenute vrste zamišljenih mreža (*grid*), koje, u suštini, predstavljaju tabele, skup "ćelija" koje sadrže paragrafe ili grupe paragrafa. I već godinama, ugnježene tabele (tabele unutar tabela) su bile jedino održivo rešenje za kreiranje različitih veb stranica.

CSS je specijalizovan u opisivanju kako bi stvari trebalo da izgledaju, uključujući i kako bi trebalo da budu postavljene ili pozicionirane. Atributi `top`, `left` i `position` omogućavaju da se kreiraju stilovi koji opisuju različite zone na veb stranici.

U narednim fajlovima prikazan je način kako može da se preraspodeljuje pozadinska boja kroz **različite gradijentne** (linearne, elipsaste i kružne) **prelaze sa primenom 2, 3 ili 4 boje**.

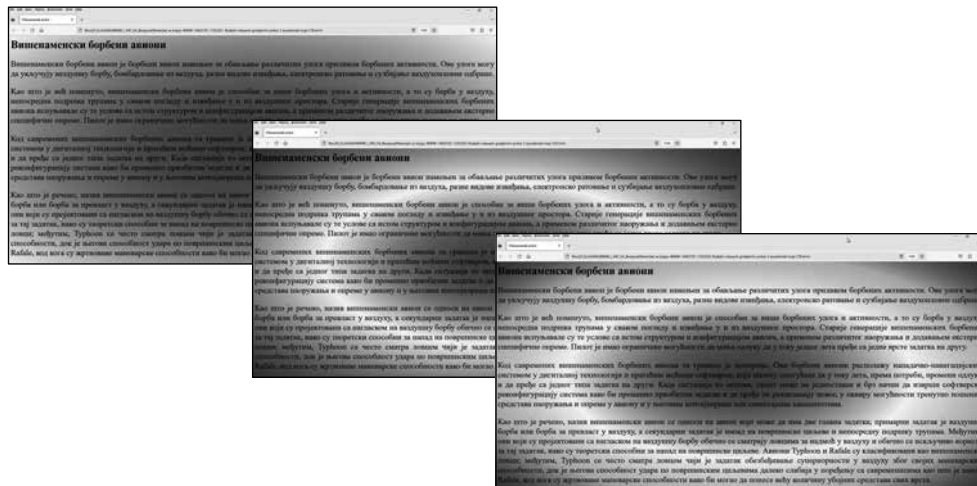


Slika 11.59. Linearni gradijentni prelaz 2, 3 i 4 boje^{58 59 60}

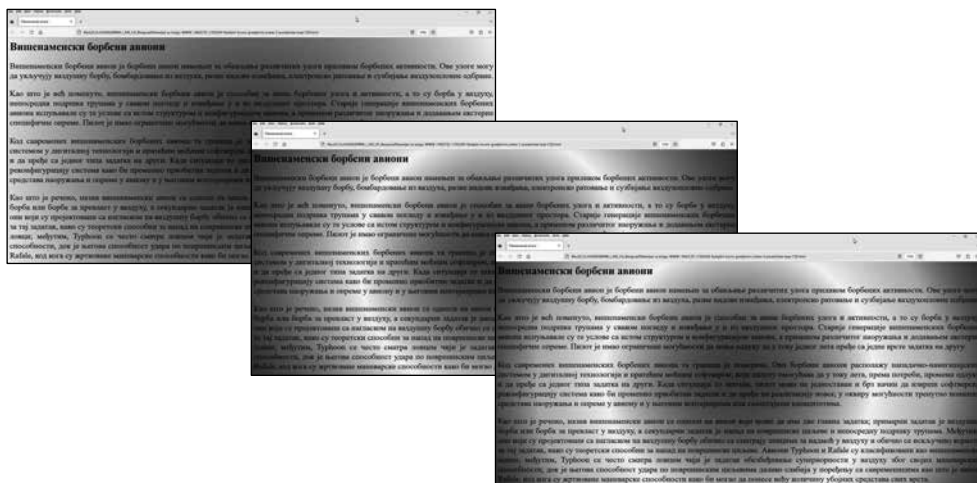
^{58 58} Linearni gradijentni prelaz 2 pozadinske boje CSS.html

^{59 59} Linearni gradijentni prelaz 3 pozadinske boje CSS.html

^{60 60} Linearni gradijentni prelaz 4 pozadinske boje CSS.html



Slika 11.60. Radijalni elipsasti gradijentni prelaz 2, 3 i 4 boje^{61 62 63}



Slika 11.61. Radijalni kružni gradijentni prelaz 2, 3 i 4 boje^{64 65 66}

11.5. CSS3 komande i noviteti

CSS3 je najnoviji standard za CSS i u potpunosti je kompatibilan sa ranijim verzijama CSS-a. CSS3 je podeljen na "module", koji sadrže "stare CSS specifikacije" (koje su pode-

⁶¹ 61 Radijalni elipsasti gradijentni prelaz 2 pozadinske boje CSS.html

⁶² 62 Radijalni elipsasti gradijentni prelaz 3 pozadinske boje CSS.html

⁶³ 63 Radijalni elipsasti gradijentni prelaz 4 pozadinske boje CSS.html

⁶⁴ 64 Radijalni kružni gradijentni prelaz 2 pozadinske boje CSS.html

⁶⁵ 65 Radijalni kružni gradijentni prelaz 3 pozadinske boje CSS.html

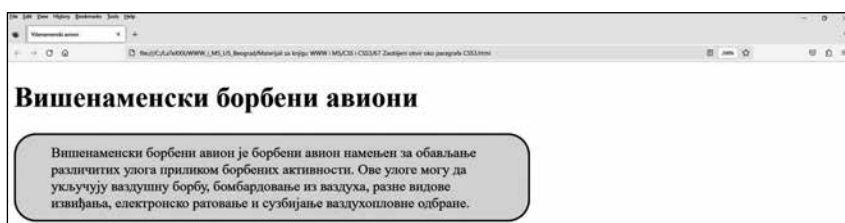
⁶⁶ 66 Radijalni kružni gradijentni prelaz 4 pozadinske boje CSS.html

ljene na manje elemente). Pored toga dodati su novi moduli. Neke od najznačajnijih CSS3 modula su **selektori** (*selectors*), **modeli pravougaonih površi** (*box models*), **pozadine i granični okviri** (*backgrounds and borders*), **vrednosti slike i zamenjen sadržaj** (*image values and replaced content*), **efekti na tekstu** (*text effects*), **2D i 3D transformacije** (*2D/3D transformations*), **animacije** (*animations*) i **višestruke kolone** (*multiple column layout*).

11.5.1. CSS3 i granični okviri

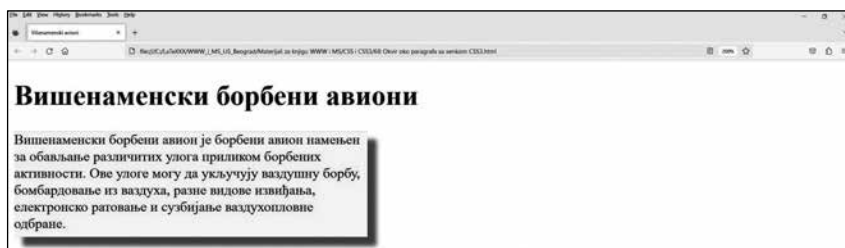
Upotrebom CSS3 autor može da kreira zaobljene granične okvire (granice), da dodaje senku pravougaonim površinama i da koristi sliku kao granicu ili granični okvir, bez upotrebe eksternih programa kao što su GIMP ili *Photoshop*. U ovom delu će biti reči o tri atributa koji su "vezani" za granične okvire ili granice – `border-radius`, `box-shadow` i `border-image`.

Kreiranje zaobljenih čoškova u CSS2 je bilo malo komplikovano. Morale su da se koriste različite slike za svaki ugao. U CSS3 stvaranje zaobljenih čoškova je lako. U CSS3, atribut `border-radius` se koristi za kreiranje zaobljenih čoškova, kao što je prikazano na slici 11.62.



Slika 11.62. Zaobljeni okvir oko paragrafa⁶⁷

U CSS3, atribut `box-shadow` se koristi za dodavanje senke pravougaonim površinama, koje se koriste u najrazličitije svrhe, kao što se vidi na slici 11.63.



Slika 11.63. Okvir oko paragrafa sa pripadajućom senkom⁶⁸

Atribut `border-image` može da koristi sliku u cilju kreiranja graničnog okvira ili granice. Ovaj atribut omogućava i da autor sâm izabere ili definiše sliku koja će "glumiti" granični element.

⁶⁷ 67 Zaobljeni okvir oko paragrafa CSS3.html

⁶⁸ 68 Okvir oko paragrafa sa senkom CSS3.html

11.5.2. CSS3 i pozadine

CSS3 sadrži nekoliko novih atributa po pitanju pozadina, kojai omogućavaju veću kontrolu nad pozadinskim elementima. U ovom delu će se pomenuti atributi `background-size` i `background-origin`. Pored ovih atributa, obratiće se pažnja i na korišćenje višestrukih pozadinskih slika.

Atribut `background-size` određuje veličinu pozadinske slike. Pre CSS3, veličina pozadinske slike je bila određena stvarnom veličinom slike. U CSS3 moguće je odrediti veličinu pozadinske slike, što omogućava da iste slike ponovo koriste u pozadini, u različitim kontekstima. Veličina slike može da se odredi u pikselima ili u procentima. Ako se veličina odredi u procentima, veličina se određuje u odnosu na širinu i visinu "roditeljskog" elementa (slika 11.64, gore levo).



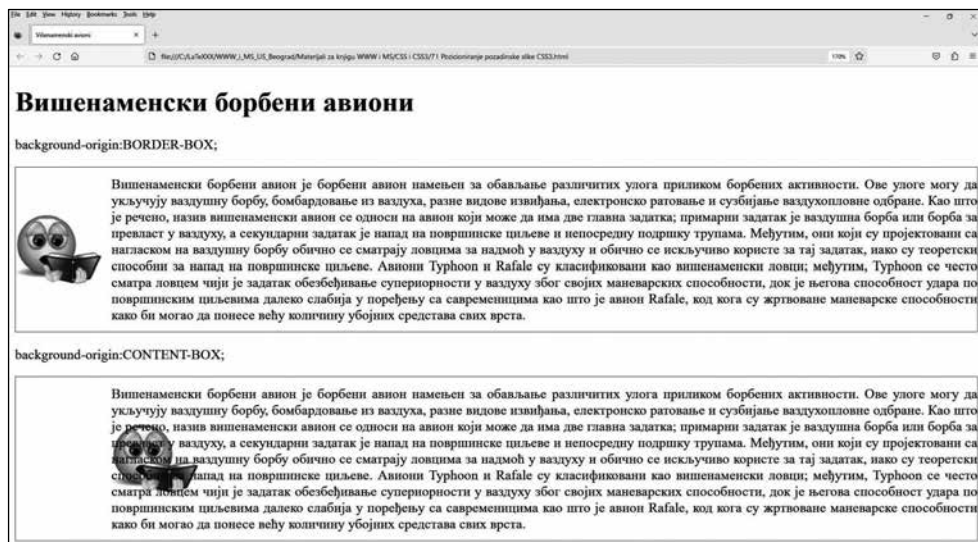
Slika 11.64. *Različite veličine i prikazi pozadinske slike*^{69 70}

Pozadinska slika može i da se "razvuče" (*stretch*) kako bi u potpunosti popunila željeni prostor, kao što se se vidi na slici 11.64, dole desno.

Atribut `background-origin` određuje područje gde će biti pozicionirana pozadinska slika. Pozadinska slika može da se postavi u pravougaonu površinu sa određenim sadržajem (*content box*), u oblast sa dodatnim belim prostorom (*padding box*) ili u graničnu oblast pravougaone površine namenjene za neki sadržaj (*border box*).

⁶⁹ 69 Veličina pozadinske slike CSS3.html

⁷⁰ 70 Razvlačenje pozadinske slike da pokrije tekst CSS3.html



Slika 11.65. Pozicioniranje pozadinske slike u veb dokumentu⁷¹

11.5.3. CSS3 i gradijentni prelazi boja

Gradijentni prelazi u CSS3 dozvoljavaju autoru da prikaže glatke prelaze između dve ili više specificiranih boja. Ranije su morale slike da se koriste za ove efekte. Međutim, koristeći CSS3 gradijentne prelaze može da se smanji vreme preuzimanja i korišćenja odgovarajućeg opsega. Pored toga, elementi sa gradijentnim prelazima izgledaju bolje kada se zumiraju, jer gradijentni prelazi se generišu od strane pretraživača. CSS3 definiše dva tipa gradijentnih prelaza:

- **linearni gradijentni prelazi** (*linear gradients*) koji se pružaju dole, gore, levo, desno ili dijagonalno;
- **radijalni gradijentni prelazi** (*radial gradients*) koji su definisani njihovim centrom.

Da bi se kreirao **linearni gradijentni prelaz** autor mora da definiše najmanje dve zaustavne boje ili dve boje, uopšte. Zaustavne boje (*color-stop*) su boje između kojih autor želi da kreira "gladak" prelaz. Takođe, autor može da podesi početnu tačku i pravac (ili ugao), zajedno sa gradijentnim efektima.

Opšta sintaksa za linearni gradijentni prelaz ima oblik:

```
background: linear-gradient(direction, color-stop1, color-stop2, ...);
```

a njena primena je prikazana na slici 11.66. Na slici 11.66 gore prikazana je podrazumevana vrednost za *direction*, a to je Top to Bottom, a definisana je žuta boja kao početna, a plava kao završna.

⁷¹ 71 Pozicioniranje pozadinske slike CSS3.html



Slika 11.66. *Različiti linearni gradijentni prelazi 2 boje*^{72 73 74}

Primer na slici 11.66 sredina (fajl pod brojem 73) pokazuje linearni gradientni prelaz koji počinje sa leve strane. Počinje sa žutom bojom, a prelazi u plavu.

Autor može da napravi i dijagonalni gradientni prelaz navođenjem i horizontalne i vertikalne startne pozicije (to bootom right). Primer na slici 11.66 dole pokazuje linearni gradientni prelaz koji počinje u gornjem levom uglu i ide ka donjem desnom uglu. Počinje sa primenom žute boje, a završava se plavom bojom.

⁷² 72 Gradientni prelaz 2 boje odozgo nadole CSS3.html

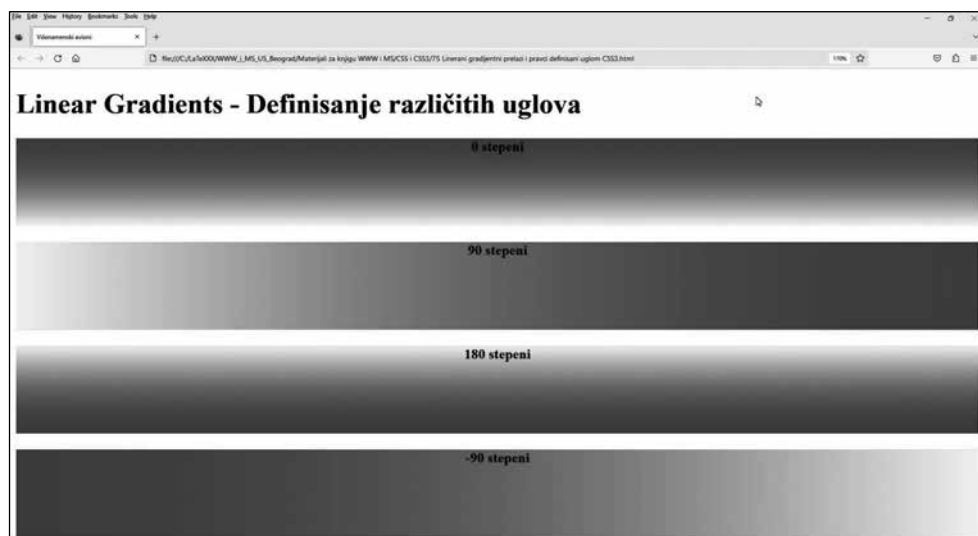
⁷³ 73 Gradientni prelaz 2 boje sa leve na desnu stranu CSS3.html

⁷⁴ 74 Gradientni prelaz 2 boje sa gornje leve na donju desnu stranu CSS3.html

Ako autor želi više kontrole nad pravcem gradijentnog prelaza, to može definisanjem ugla, umesto unapred definisanih pravaca (to bottom, to top, to right, to left, to bottom right itd). Opšta sintaksa za ovakvu vrstu linearnog gradijentnog prelaza ima oblik:

```
background: linear-gradient(angle, color-stop1, color-stop2);
```

Definisani ugao je naveden kao ugao između horizontalne linije i linije gradijenta, koji "ide" u suprotnom smeru od smera kretanja kazaljke na satu. Drugim rečima, vrednost 0deg (0°) stvara gradijentni prelaz od dna ka vrhu, dok vrednost 90deg (90°) generiše gradijentni prelaz sa leve na desnu stranu. U fajlu pod brojem 75 su definisani različiti uglovi za prostiranja linearnog gradijentnog prelaza, a rezultat je prikazan na slici 11.67.



Slika 11.67. *Linearni gradijentni prelazi definisani uglovima*⁷⁵

Fajl pod brojem 76 pokazuje kako se podešava više zaustavnih boja i kako se definiše njihova raspodela (ravnomerna ili neravnomerna), kao što je prikazano na slici 11.68.

Radijalna gradijentna raspodela je definisana centrom ili tačkom centra. Da bi se kreirala radijalna ili kružna gradijentna raspodela, moraju se definisati najmanje dve zaustavne boje. Obično može da se odredi centar gradijentne raspodele, oblik (kružnica ili elipsa), kao i njena veličinu. Uobičajene vrednosti definišu radijalnu gradijentnu raspodelu čiji je centar i tačka centra, oblika elipse, a proteže se do najdalje dostupne tačke.

Opšta sintaksa za radijalni ili kružni gradijentni prelaz ima oblik:

```
background: radial-gradient(center, shape size,
                           start-color, ..., last-color);
```

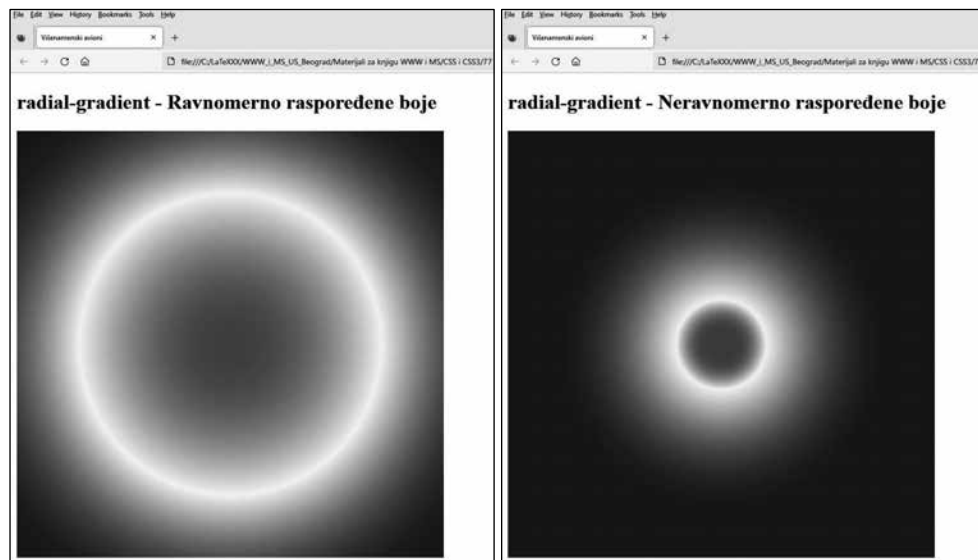
a njena primena prikazana je na slici 11.69.

⁷⁵ 75 Linerani gradijentni prelazi i pravci definisani uglom CSS3.html



Slika 11.68. *Linearni gradijentni prelazi više boja*⁷⁶

Podrazumevana vrednost je da su boje ravnomerno raspoređene, kao što je prikazano na slici 11.69 levo (sintaksa je u fajlu pod brojem 77). Radijalna gradijentna raspodela boja sa različitim raspoređivanjem boja je prikazan na slici 11.69 desno, a sintaksa je u fajlu pod brojem 78.



Slika 11.69. *Radijalna gradijentna preraspodela 3 boje*^{77 78}

⁷⁶ 76 Linerani gradijentni prelazi više boja CSS3.html

⁷⁷ 77 Radijalni gradijentni prelaz 3 boje ravnomerno CSS3.html

⁷⁸ 78 Radijalni gradijentni prelaz 3 boje neravnomerno CSS3.html

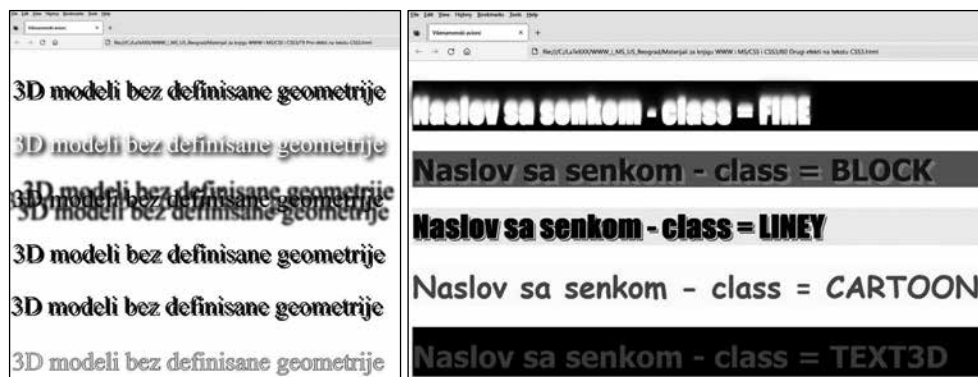
11.5.4. CSS3 i efekti na tekstu

Dodavanje senke tekstu je posledica primene atributa `text-shadow` sa pratećim elementima i njihovim vrednostima. Treba napomenuti da ovaj atribut prihvata da ima u sebi i više definisanih senki, ali moraju da budu odvojene tačkom-zarezom (;). Sintaksa je:

```
text-shadow: h-shadow v-shadow blur-radius color
            |none|initial|inherit;
```

Elementi sa svojim vrednostima su:

- `h-shadow` – obavezan podatak; definiše horizontalni položaj senke. Negativne vrednosti mogu da se primenjuju i kod `h-shadow` i `v-shadow` elemenata.
- `v-shadow` – obavezan podatak; definiše vertikalni položaj senke.
- `blur-radius` – opcioni podatak; definiše poluprečnik "mrljanja".
- `color` – opcioni podatak; definiše boju senke.
- `none` – uobičajena vrednost; nema senke.
- `initial` – podešava sve vrednosti na inicijalne ("fabričke") vrednosti.
- `inherit` – "preuzimaju" se vrednosti objekta za koji se definiše senka.



Slika 11.70. Različiti efekti primenjeni na tekstu^{79 80}

11.5.5. CSS3 i 2D (ravanske) transformacije

Sa primenom CSS3 transformacija autor ili korisnik može da premešta ili pomera elemente (*move*), da skalira (*scale*), rotira (*rotate*), "uvrće" (*spin*) i razvlači (*stretch*). Transformacija je efekat koji omogućava da jedan element menja oblik, veličinu i položaj. Elementi mogu da se transformišu primenom 2D ili 3D transformacija.

⁷⁹ 79 Prvi efekti na tekstu CSS3.html

⁸⁰ 80 Drugi efekti na tekstu CSS3.html

U ovom delu biće reči o sledećim metodama za 2D transformacije – `translate()`, `rotate()`, `scale()`, `skew()` i `matrix()`. Primenom metode `translate()`, element se pomera sa svoje trenutne pozicije, u zavisnosti od zadatih parametara, u levu ili desnu stranu (x osa) i gore ili dole (y osa).

Primenom metode `rotate()`, elemenat se rotira u smeru kretanja kazaljke na satu za ugao koji je definisan u stepenima. Negativne vrednosti su dozvoljene i rotiraju element u suprotnom smeru od smera kretanja kazaljke na satu.

Primenom metode `scale()`, povećava se ili se smanjuje veličina elementa, zavisno od zadatih parametara za širinu (x osa) i visinu (y osa).

Primenom metode `skew()`, vrši se "zakošenje" ili "smicanje" pojedinih elemenata prema zadatom uglu, koji zavisi od zadatih parametara za horizontalne (x osa) i vertikalne (y osa) linije. Vrednost `skew(30deg, 20deg)` smiče element za 30° po x osi i 20° po y osi.

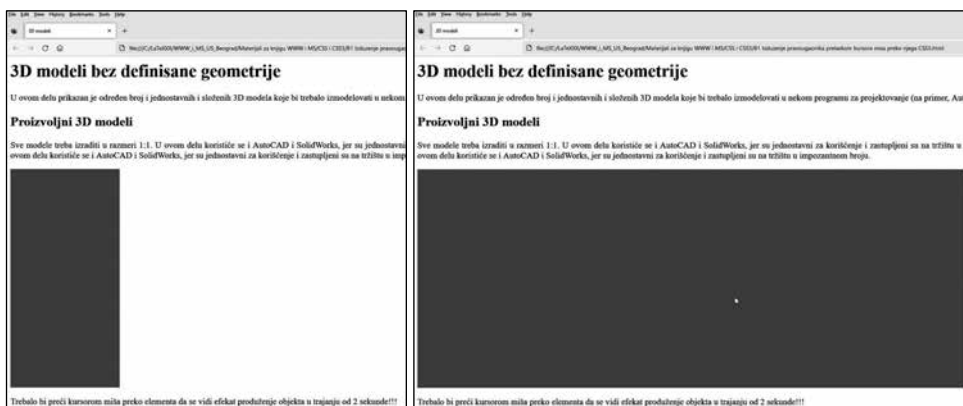
11.5.6. CSS3 i 3D (prostorne) transformacije

CSS3 omogućava oblikovanje elemenata i pomoću 3D transformacija. U ovom delu će biti, Ukratko, reći o dve metode za 3D transformacije – `rotateX()` i `rotateY()`.

Primenom metode `rotateX()`, element se rotira oko njegove x ose za određeni ugao, koji je definisan u stepenima. Primenom metode `rotateY()`, element se rotira oko svoje y ose za određeni ugao i vrednost ugla je definisana u stepenima.

11.5.7. CSS3 i prelazni (tranzicioni) efekti

Primenom CSS3, autor može da doda efekat kada se prelazi iz jednog stila u drugi, bez upotrebe *Flash* animacija ili programskog jezika *JavaScript*. CSS3 tranzicioni efekti omogućavaju da se elemenat postepeno menja iz jednog stila u drugi. Da bi se to uradilo, moraju da se navedu dve stvari – CSS atribut na koji se dodaje efekat i vreme trajanja samog efekta.



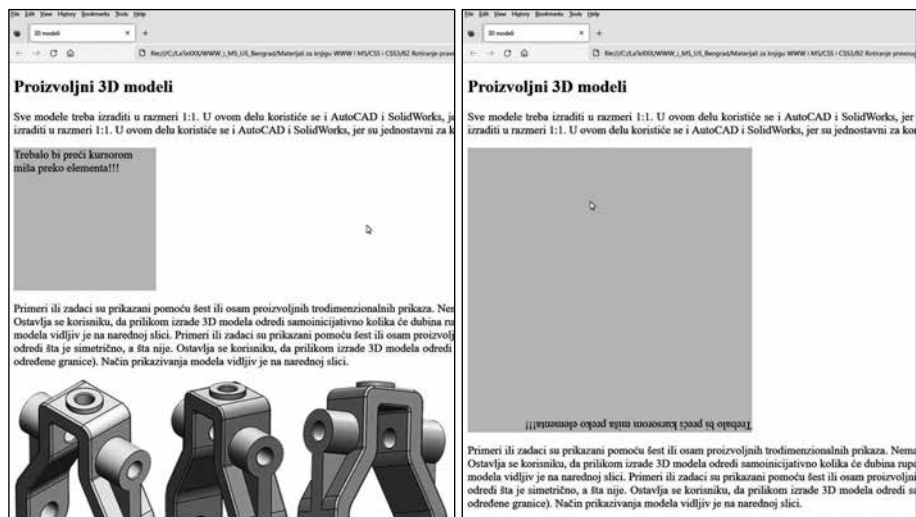
Slika 11.71. Izduženje pravougaonika prelaskom kursora miša preko njega⁸¹

⁸¹ 81 Izduženje pravougaonika prelaskom kursora misa preko njega CSS3.html

Trebalo bi napomenuti da ako vreme trajanja efekta nije navedeno ili definisano, onda "tranzicija" neće imati efekta, jer podrazumevana vrednost je 0. Efekat tranzicija će početi kada navedeni CSS atribut menja vrednost. Tipičan promena CSS atributa se dešava kada korisnik "prelazi" kursorom miša preko elementa.

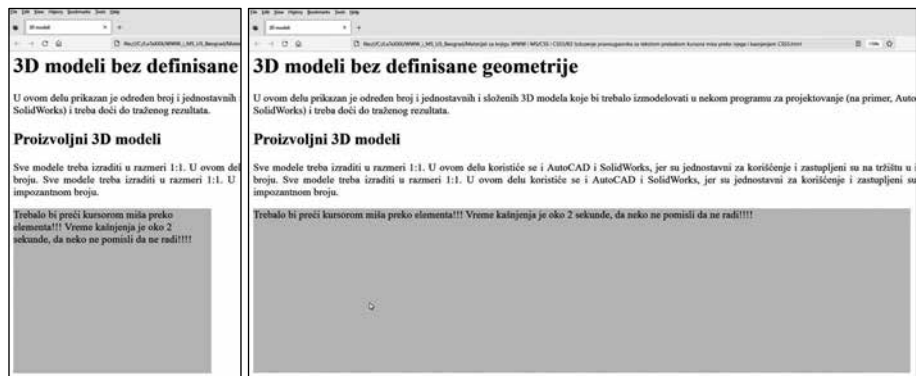
Kada se kursor miša pomeri van određene oblasti elementa ili samog elementa, njegovo stanje se postepeno vraća svom originalnom stilu.

Da bi se dodali prelazni efekti na više od jednog CSS atributa, onda se atributi, u sintaksi, odvajaju zarezom, kao što je prikazano u fajlu pod brojem 82.



Slika 11.72. Rotiranje pravougaonika prelaskom kursora miša preko njega⁸²

Primer u fajlu pod brojem 83 (slika 11.73) koristi sva četiri tranziciona atributa kako bi željeni efekat primenio kako treba.



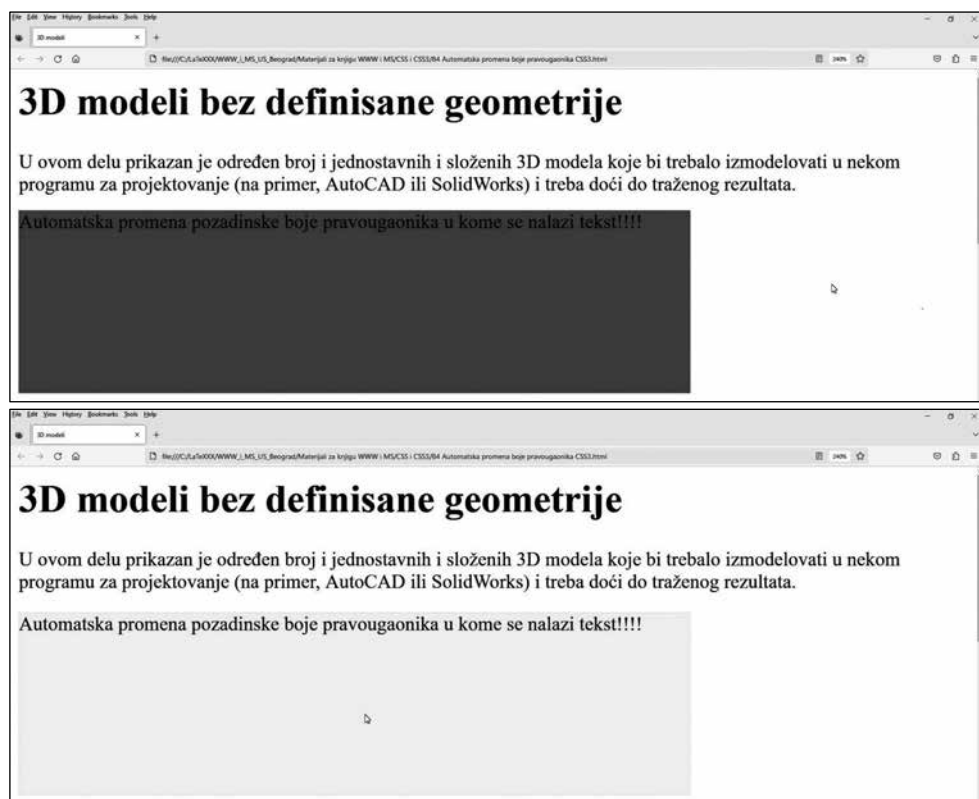
Slika 11.73. Izduženje pravougaonika prelaskom kursora miša i kašnjenje⁸³

⁸² 82 Rotiranje pravougaonika prelaskom kursora miša preko njega CSS3.html

⁸³ 83 Izduženje pravougaonika prelaskom miša preko njega i kašnjenjem CSS3.html

11.5.8. CSS3 i animacija

Adekvatnom primenom CSS3, autor može da stvori animacije koje mogu da zamene *Flash* animacije, animirane slike i *JavaScript* u postojećim veb stranicama. Kada se animacija kreira unutar pravila `@keyframe`, mora da se "veže" za selektore, inače animacija neće imati nikakvog efekta. "Vezivanje" animacije za selektor (element) se vrši navođenjem najmanje dva atributa – ime i vreme trajanja animacije.

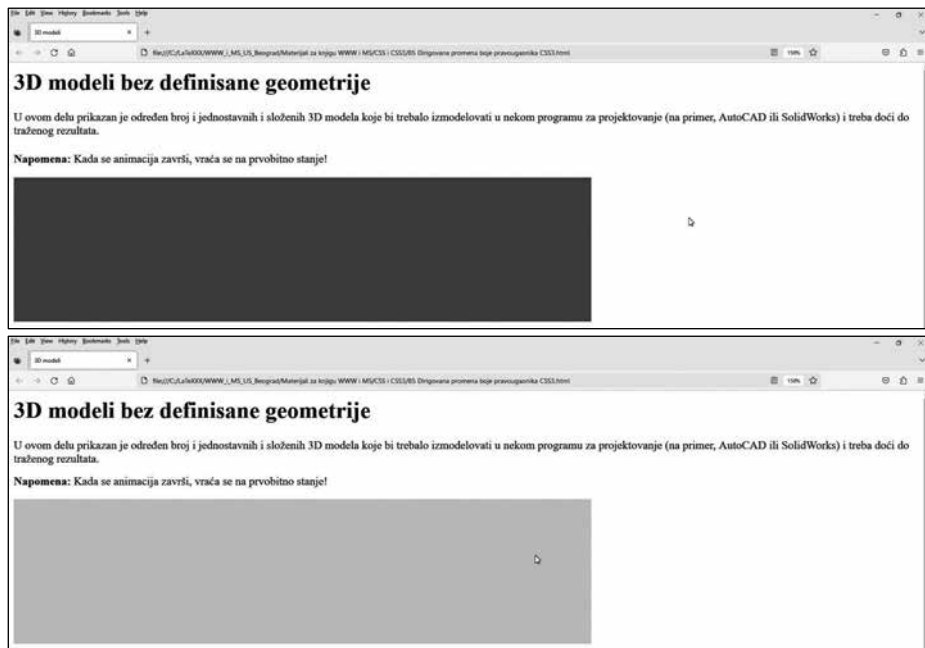


Slika 11.74. Automatska promena boje pravougaonika⁸⁴

Trebalo bi napomenuti da ako vreme trajanja efekta nije navedeno, onda animacija neće imati efekta, jer je podrazumevana (uobičajena) vrednost 0.

Animacija omogućava da se element postepeno menja iz jednog stila u drugi. Može da promeni onoliko atributa koliko treba i koliko puta treba. Prilikom određivanja trenutaka kada će se promene događati, mogu vrednosti da se definišu u procentima ili mogu da se koriste ključne reči *from* i *to* (što predstavlja vrednosti 0% i 100%). Vrednost 0% predstavlja početak animacije, 100% je kada je animacija završena. U primeru u fajlu pod brojem 85 izvršena je promena boja elementa kroz animaciju 25%, a onda 50%, da bi na kraju bila vrednost 100%, kada je animacija završena.

⁸⁴ 84 Automatska promena boje pravougaonika CSS3.html

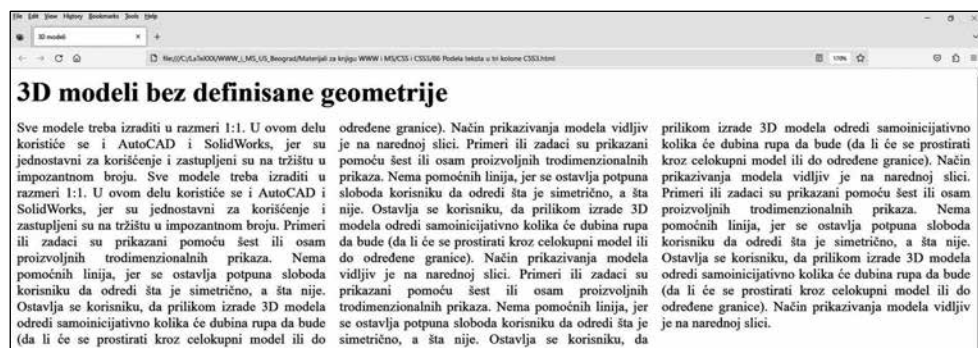


Slika 11.75. *Dirigovana promena boje pravougaonika*⁸⁵

11.5.9. CSS3 i višestruke kolone

Sa CSS3, autor može da kreira više kolona za upisivanje teksta – kao kolone (stupci) u novinama! U ovom delu će, ukratko, biti reči o atributima koji "barataju" sa više kolona, a to su `column-count`, `column-gap` i `column-rule`.

Atribut `column-count` određuje na koliko bi kolona element trebalo da bude podeljen. U primeru u fajlu pod brojem 86 se deli `<div>` element na 3 kolone (slika 11.76).



Slika 11.76. *Podela teksta u 3 kolone*⁸⁶

⁸⁵ 85 Dirigovana promena boje pravougaonika CSS3.html

⁸⁶ 86 Podela teksta u tri kolone CSS3.html

Atribut `column-gap` određuje fiksno rastojanje između formiranih kolona. U primeru prikazanom u fajlu `CSS3-029.html` definisano je fiksno rastojanje od 35 px između kolona.



Slika 11.77. Podela teksta u 3 kolone i fiksno rastojanje između njih⁸⁷

Atribut `column-rule` određuje vrstu linije koja se iscrtava između kolona, kao i njenu širinu i boju. U primeru prikazanom u fajlu `CSS-030.html` je definisana linija širine ("`debljine`") 4 piksela i određene boje (`#ff00ff`). Sintaksa prikazana ispod potvrđuje to:

```
<style>
.visekolona {column-count:3;
              column-gap:40px;
              column-rule:4px outset #ff00ff;}
</style>
```

Kao što se vidi iz gornje sintakse, dokument je podeljen na 3 kolone sa međusobnim fiksnim rastojanjem od 40 piksela i sa linijom između kolona "debljine" 4 piksela i boje `ff00ff`. Rezultat primene pomenute sintakse unutar pretraživača *Firefox* je prikazan na slici 11.78.



Slika 11.78. Podela teksta u 3 kolone i linija između njih⁸⁸

Kada se modifikuje gornja sintaksa (fajl pod brojem 89) mogu da se dobiju različiti rezultati. Na primer, na slici 11.79 dokument podeljen na 4 kolone sa međusobnim fiksnim

⁸⁷ 87 Fiksno rastojanje između kolona `CSS3.html`

⁸⁸ 88 Fiksno rastojanje između kolona i linija između `CSS3.html`

rastojanjem od 30 piksela i sa linijom između kolona "debljine" 3 piksela i boje sa heksadecimalnom vrednošću ff0000. Reč je o liniji crvene boje. Pored pomenutih podešavanja, iskorišćen je i atribut <p> za naknadno odvajanje pojedinačnih paragrafa, a bez dodatnog podešavanja, što znači da je tekst poravnat po levoj margini.



Slika 11.79. Podela teksta u 4 kolone i linija između njih⁸⁹

Na slici 11.80 dokument podeljen na 3 kolone sa međusobnim fiksnim rastojanjem od 60 piksela i sa linijom između kolona "debljine" 10 piksela i boje 0000ff. Reč je o liniji plave boje. Pored pomenutih podešavanja, iskorišćen je i atribut <p align="justify"> za naknadno odvajanje pojedinačnih paragrafa, a insistira se na poravnanju paragrafa po obe margine.



Slika 11.80. Fiksno rastojanje između kolona i šira linija između njih⁹⁰

11.6. CSS modul Flexible Box Layout

CSS modul poznat kao *Flex Box Layout* donosi inovacije u dizajnu korisničkog interfejsa, pružajući spektakularnu prilagodljivost različitim ekranima. Ovaj izvanredan CSS standard

⁸⁹ Fiksno rastojanje između 4 kolone i linija između CSS3.html

⁹⁰ Fiksno rastojanje između kolona i sira linija između CSS3.html

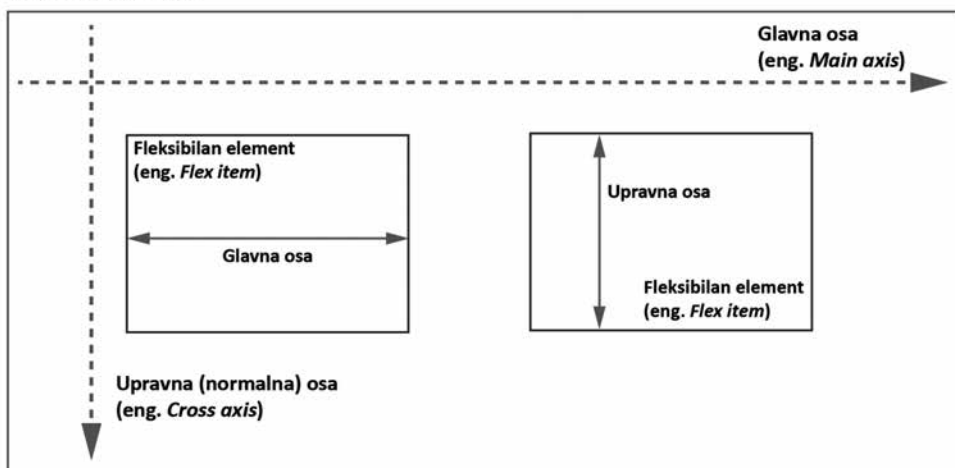
omogućava organizaciju elemenata duž jedne dimenzije, potpuno transformišući način na koji se elementi pozicioniraju na veb stranicama. Fleksibilni raspored definiše kontejnere i hijerarhiju elemenata, koji se fleksibilno poravnavaju duž odabrane ose.

Ovakav napredni raspored omogućava fleksibilnost prilikom prilagođavanja dimenzija elemenata, postizanju impresivnih vizuelnih efekata i optimalnom iskorišćenju raspoloživog prostora na veb stranici. Fleksibilni raspored predstavlja neophodnu tehnologiju koja definiše način na koji se elementi mogu poravnati u redove i kolone, otvarajući put za efikasno iskorišćavanje raspoloživog prostora. Sjajna mogućnost fleksibilnog poravnanja elemenata, uz mogućnost prilagođavanja njihove veličine, dodatno omogućava i postizanje visoko kvalitetnih dizajnerskih rešenja na modernim veb stranicama.

U srcu koncepta *Flexbox*-a leži revolucionarna ideja koja omogućava kontejneru da prilagodi širinu/visinu i redosled svojih elemenata kako bi maksimalno iskoristio raspoloživi prostor. Ovo se posebno odnosi na prilagođavanje različitim uređajima i njihovim veličinama ekrana. Ovakva tehnika omogućava fleksibilnim elementima unutar kontejnera da se šire kako bi ispunili dostupan prostor. Omogućava smanjenje da se izbegne prekoračenje.

Flexbox modul je nastao kao rezultat potrebe programerske zajednice za unapređenjem organizacije rasporeda elemenata na veb stranici i prevazilaženjem nedostataka koje donose tradicionalne CSS tehnike poput *float*, *position*, fiksiranje margina i slično. U sklopu ovog modula, postoji koncept fleksibilnih elemenata koji omogućava poravnavanje elemenata duž primarne i sekundarne ose. Primarna osa, poznata kao glavna osa (eng. *Main axis*), određuje smer rasporeda elemenata, dok se sekundarna osa, poznata kao upravna osa (eng. *Cross axis*), koristi za kontrolu njihove pozicije u odnosu na glavnu osu.

Fleksibilan kontejner
(eng. *Flex container*)



Slika 11.81. Model fleksibilnog kontejnera i fleksibilnih elemenata u kontejneru

U fleksibilnom kontejneru, glavna horizontalna osa se koristi za pozicioniranje elemenata, pri čemu se pretpostavlja da će se oni rasporediti sleva nadesno. Vertikalna osa,

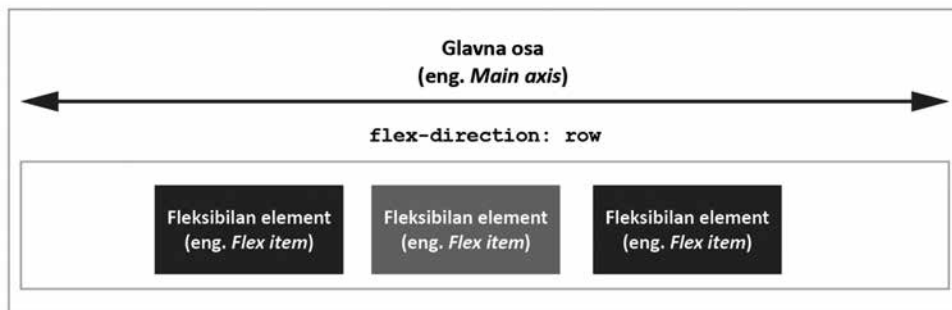
koja je normalna na glavnu osu, se koristi za raspoređivanje elemenata duž te vertikalne osi. Fleksibilni elementi imaju mogućnost da se raspoređuju u oba smera duž obe ose.

Konkretno, po podrazumevanim postavkama, elementi se raspoređuju sleva nadesno duž glavne ose, ali se smer raspoređivanja može promeniti tako da se elementi raspoređuju zdesna nalevo. Isto pravilo važi i za upravnu osu, gde se elementi podrazumevano raspoređuju odozgo nadole, ali se smer raspoređivanja može promeniti tako da se elementi raspoređuju odozdo nagore.

Roditeljski element u fleksibilnom kontejneru naziva se **kontejnerski element**. Ovo se odnosi na element koji je odabran da bude fleksibilan kontejner putem CSS svojstva `display: flex`. Elementi koji se nalaze unutar kontejnerskog elementa nazivaju se **dečji elementi** ili **fleksibilni elementi**.

Kontrola smera raspoređivanja elemenata postiže se pomoću naredbe `flex-direction`. Ovom naredbom se definiše smer i raspoređivanje dečjih elemenata duž odgovarajuće ose. Podrazumevana vrednost je postavljena na "red" (eng. *row*), što znači da se dečji elementi raspoređuju horizontalno, sleva nadesno.

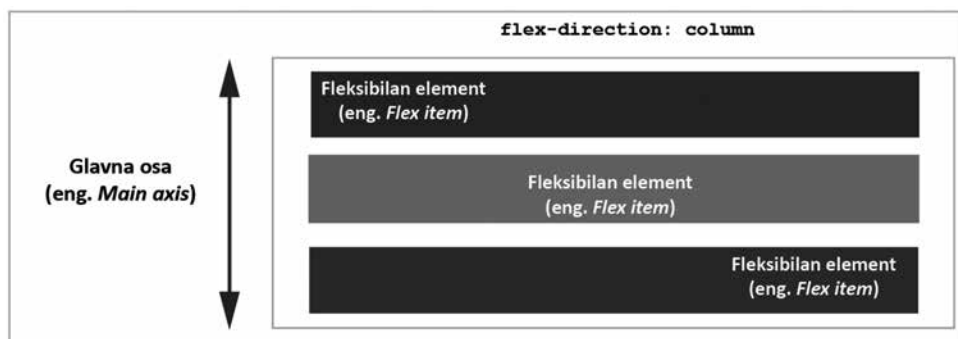
Ovo znači da se dečji elementi podrazumevano raspoređuju sleva nadesno duž glavne ose. Međutim, ukoliko je potrebno da se elementi raspoređuju zdesna nalevo duž glavne ose, može se koristiti naredba `flex-direction: row-reverse`. Na taj način se menja smer raspoređivanja elemenata duž glavne ose, omogućavajući da se elementi prikažu u obrnutom redosledu.



Slika 11.82. Podrazumevana vrednost glavne ose

NAPOMENA: Gore navedeno pravilo važi za pretraživače koji koriste podešavanje govornog jezika gde je podrazumevani smer pisanja sa leve strane prema desnoj strani. Međutim, u slučaju arapskog jezika i takvog podešavanja podrazumevnog pretraživača, elementi u fleksibilnom kontejneru imaju potpuno suprotan scenario. U tom slučaju, elementi se podrazumevano raspoređuju s desna na levo.

Promenom podrazumevane ose, raspoređivanje elemenata se može postaviti u kolonu. Novo pravilo u kontejneru definiše se naredbom: `flex-direction: column`. Elementi unutar kontejnera se podrazumevano raspoređuju odozgo prema dole. Takođe, moguće je pomoću naredbe `flex-direction: column-reverse` promeniti smer raspoređivanja fleksibilnih elemenata u kontejneru, pri čemu se elementi deca raspoređuju odozdo prema gore.

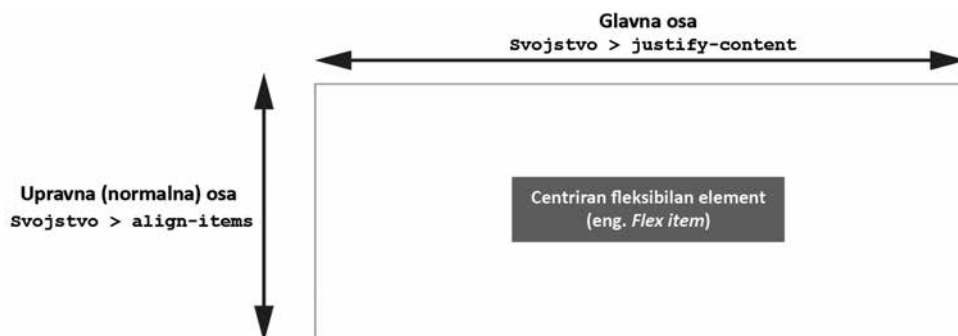


Slika 11.83. Podrazumevani raspored elemenata postavljenih na kolonu

11.6.1. Poravnavanje elemenata unutar fleksibilnog kontejnera

Unutar *Flexbox* sistema, postoje dva svojstva koja se mogu primeniti na fleksibilne elemente kako bi se postiglo centriranje, uzimajući u obzir podrazumevane vrednosti glavne ose (koja je horizontalna) i normalne ose (koja je vertikalna). Implementacijom ovih karakteristika, programer je sposoban da jednostavno izvrši željeno centralno poravnanje elemenata unutar *Flexbox* kontejnera, prilagođavajući ih strukturnim zahtevima veb stranice:

- `Justify-content` – koristi se ukoliko je elemente potrebno centrirati duž horizontalne ose,
- `Align-items` – koristi se ukoliko je elemente potrebno poravnati duž vertikalne ose.



Slika 11.84. Centriranje fleksibilnog elementa duž glavne i upravne ose

Svojstva za kontrolu poravnavanja mogu biti

- `Justify-content` – kontroliše poravnanje fleksibilnih elemenata unutar duž glavne (podrazumevane) ose;
- `Align-items` – kontroliše poravnanje svih fleksibilnih elemenata duž vertikalne ose;

- **Align-self** – kontroliše poravnanje pojedinačnog fleksibilnog elementa duž vertikalne ose;
- **Align-content** – kontroliše prostor između fleksibilnih linija duž vertikalne ose;
- **Gap, column-gap i row-gap** – koriste se za dodavanje praznog prostora ili margina između fleksibilnih elemenata.

Kôd koji ilustruje korišćenje svojstava može se napisati kao što je prikazano na slici 11.85.

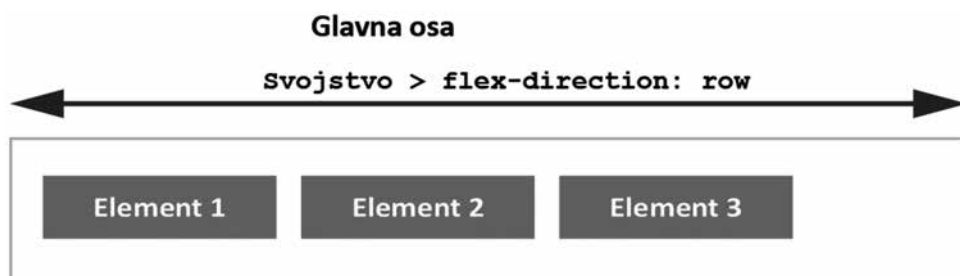
<pre><div class="flexItem"> <div></div> </div></pre> HTML	<pre>.flexItem { display: flex; align-items: center; justify-content: center; } .flexItem div { width: 300px; height: 80px; }</pre> CSS
--	---

Slika 11.85. Sintaksa za korišćenje svojstava

11.6.2. Raspoređivanje elemenata u fleksibilnom kontejneru

Moduli CSS-a *Flexbox* i *Grid* su snažni alati za dizajn i organizaciju sadržaja na veb stranici. Mogućnost kontrole redosleda elemenata pruža dizajnerima veću slobodu i fleksibilnost u dizajniranju stranica. Definišu se novi metodi rasporeda koji donose mogućnost kontrole redosleda sadržaja. Na takve metode rasporeda utiču sama svojstva fleksibilnog kontejnera, koji definišu pravilo na koji način se elementi raspoređuju u okviru datog kontejnera. Važno je istaći da se svojstva fleksibilnog kontejnera definišu strikno nad datim kontejnerom, a ne nad elementima pojedinačno unutar tog HTML bloka.

Jedna od ključnih tehnika za postavljanje prikaza elemenata je korišćenje svojstva pod nazivom *flex-direction*, koje ima četiri moguće vrednosti: *row*, *column*, *row-reverse* i *column-reverse*. Prve dve vrednosti održavaju redosled elemenata istim kao u izvornom kodu i prikazuju ih sekvencijalno od početne linije.



Slika 11.86. Redosled prikazivanja elmenata po horizontalnoj osi



Slika 11.87. Redosled prikazivanja elemenata po vertikalnoj osi

U nastavku se može videti inverzija vrednosti, gde se menjaju početna i krajnja linija.



Slika 11.88. Prikazivanje elemenata inverzno, po horizontalnoj osi



Slika 11.89. Prikazivanje elemenata inverzno, po vertikalnoj osi

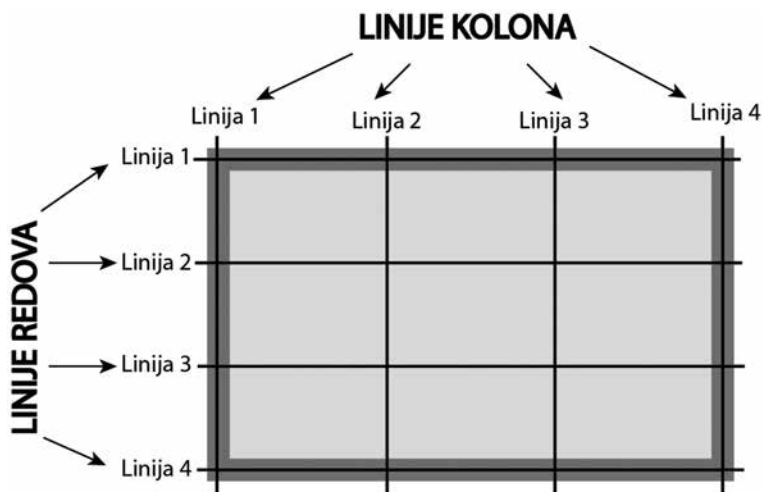
Reorganizacija elemenata može biti izuzetno značajna u prilagođavanju rasporeda sadržaja prema korisničkim zahtevima dizajna veb stranice. Osim što pruža vizuelni efekat, ova tehnika može imati društveno koristan uticaj na pristupačnost, gde se posebna pažnja usmerava na korisnike sa posebnim potrebama. Stoga je od suštinske važnosti pažljivo razmotriti implikacije reorganizacije elemenata kako bi se obezbedilo da za krajnje korisnike sadržaj ostane jasan i pristupačan.

Razumevanje fleksibilnog rasporeda elemenata i mogućnosti promene njihovih redosleda omogućava primenu naprednih tehnika dizajniranja i strukturiranja sadržaja na vebu. Korišćenje *Flexbox*-a pruža programerima i dizajnerima znatno veću fleksibilnost prilikom rasporeda elemenata, omogućavajući prilagođavanje njihovih dimenzija i pozicioniranje u skladu sa dostupnim prostorom. Osim toga, omogućava izrazito precizno upravljanje redosledom prikaza elemenata.

11.7. CSS sistem mreže

Sistem mreže (eng. *CSS Grid Layout*), sa svojom dvodimenzionalnom prirodom, uvodi sistem mreže koji pruža dodatnu kontrolu nad rasporedom elemenata na veb stranici. Ovaj sistem omogućava precizno pozicioniranje elemenata u redovima i kolonama, što je posebno korisno za složene rasporede i prilagođavanje različitim veličinama ekrana. Dizajneri i programeri mogu tačno definisati broj redova i kolona, kao i njihove veličine i raspored, čime se postiže precizan i fleksibilan raspored elemenata na stranici.

Sistem CSS mreže donosi napredne mogućnosti za dizajniranje i optimizaciju veb stranica, omogućavajući programerima i dizajnerima veću kontrolu nad rasporedom elemenata. Dok su tradicionalne CSS tehnike ograničene na jednodimenzionalno raspoređivanje elemenata, sistem mreže omogućava dvodimenzionalno pozicioniranje elemenata unutar mreže kolona i redova. Ova fleksibilnost omogućava precizno i efikasno organizovanje elemenata na stranici, prilagođavanje različitim veličinama ekrana i postizanje željenog vizuelnog izgleda.



Slika 11.90. Grafička ilustracija kolona i redova u CSS sistemu mreže

CSS *Grid* raspored predstavlja skup horizontalnih i vertikalnih linija koje se međusobno seku i definišu kolone i redove u mreži. Elementi se mogu smestiti unutar ove mreže, uzimajući u obzir definisane kolone i redove. Sistem raspoređivanja omogućava kreiranje mreže sa fiksnim dimenzijama, gde se koriste pikseli kao jedinica mere. Takođe, moguće je koristiti fleksibilne veličine, izražene u procentima ili frakcijama (jedinica mere je *fr*). Elementi se mogu precizno pozicionirati na mreži koristeći brojeve linija, imena ili odabirom odgovarajućeg područja u grid sistemu. Bitno je napomenuti da postoji algoritam koji kontrolise pozicioniranje elemenata koji nemaju eksplicitno navedenu poziciju u mreži.

Pomoću rasporeda mreže, moguće je definisati poseban *grid* sistem dodavanjem dodatnih granica za smeštanje sadržaja. Na taj način, po potrebi se mogu dodati dodatni redovi i kolone u mrežu. *Grid* sistem takođe pruža mogućnost kontrolisanja poravnanja elemenata, omogućavajući automatsko poravnanje elemenata unutar ciljanog područja mreže. Više

elemenata se može smestiti u jednu ćeliju ili područje *grid* mreže, a ti elementi mogu se delimično preklapati. Kontrola preklapanja se postiže korišćenjem svojstva *z-index*.

Sistem mreže (CSS *Grid*) predstavlja moćan modul koji, kada se kombinuje sa drugim modulima CSS-a, kao što je *Flexbox*, omogućava precizno i jednostavno organizovanje veb stranica. Kombinacijom ova dva modula, veb stranica postaje potpuno prilagodljiva različitim veličinama ekrana.

Deklaracija CSS sistema mreže mora se smestiti u kontejnerskom nivou; pravi se kontejner koji prihvata deklaraciju `display: grid`, takođe, nad elementom se može postaviti svojstvo `display: inline-grid`. Nakon deklaracije, svi podređeni elementi, odnosno, elementi deca postaju deo mreže. Pozivanjem i postavljanjem ovih svojstava neće se promeniti prikaz u pretraživaču, već će se pozadini postaviti mreža redova i kolona. Od koristi je rad u inspektor režimu podrazumevanog pretraživača, tj. razvojni alat (eng. *Developer Tools*). Svaki element koji na veb stranici ima postavljeno jedno od prethodno definisana dva svojstva, razvojni alat pretraživača može prikazati da kontejnerski element predstavlja sistem mreže, kao što se može videti na slici ispod:



Slika 11.91. Sistem mreže u kontejnerskom elementu na primeru veb sajta⁹¹

Razgraničenje mreže (eng. *Grid track*) označava prostor između dve (bilo koje) susedne linije u mreži. Primarno se definišu redovi i kolone, putem svojstava `grid-template-rows` i `grid-template-columns`; pomenute definicije predstavljaju granice mreže (slika 11.92).

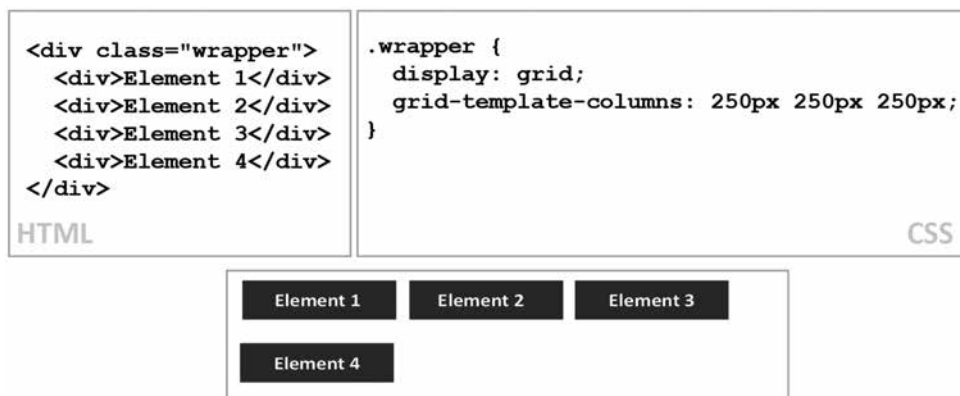


Slika 11.92. Tamniji deo predstavlja element u okviru granica mreže

⁹¹<https://developer.chrome.com/docs/devtools/css/grid/>

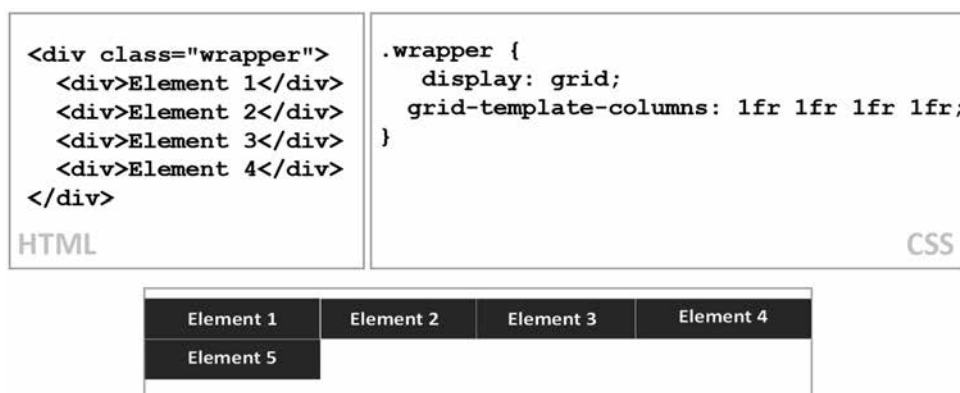
Tamniji deo predstavlja oblast u kojoj se smesti element unutar granica date mreže. Takođe, razgraničenje mreže se može formirati u implicitnoj mreži, kada se elementi pozicioniraju van granica mreže koje su definisane u eksplicitnoj mreži.

Uz dodatak svojstva `grid-template-columns`, može se proširiti prethodni primer. Može se definisati mreža sa četiri kolone, pri čemu je svaka kolona širine 250 piksela. Nakon toga, elementi deca se raspoređuju na mreži, pri čemu se svaki element smesti u jednu od ćelija mreže.



Slika 11.93. *Raspoređivanje elemenata u mreži*

Jedna od korisnih jedinica za definisanje granica mreže je *fr* (frakcija), koja omogućava fleksibilno prilagođavanje granica mreže. Ova jedinica predstavlja deo raspoloživog prostora u okviru kontejnera mreže. U sledećem primeru je prikazana definicija mreže sa četiri jednake širine granica mreže, koje se mogu smanjivati ili proširivati u skladu sa dostupnim prostorom.



Slika 11.94. *Elementi zauzimaju jednake vrednosti u raspoloživom prostoru mreže*

U grid sistemu je moguće definisati nejednake veličine za granice mreže. U sledećem primeru prikazan je raspored elemenata koji zauzimaju četiri dela u tri kolone. Vrednosti

su postavljene na dve frakcije (2 fr) i jednu frakciju (1 fr), što rezultuje različitim širinama elemenata unutar mreže.

```
<div class="wrapper">
  <div>Element 1</div>
  <div>Element 2</div>
  <div>Element 3</div>
  <div>Element 4</div>
</div>
```

HTML

```
.wrapper {
  display: grid;
  grid-template-columns: 2fr 1fr 1fr;
}
```

CSS

Element 1	Element 2	Element 3
Element 4		

Slika 11.95. *Nejednake vrednosti u raspoloživom prostoru sistema mreže*

Moguće je kombinovati fleksibilne i apsolutne jedinice u grid sistemu. Na primer, može se postaviti element mreže sa vrednošću od 400 piksela, koja ostaje fiksna bez obzira na raspoloživi prostor. Preostali prostor se zatim deli na dva jednaka dela, koristeći fleksibilne jedinice, u zavisnosti od raspoloživog prostora. Važno je napomenuti da ova praksa može dovesti do suboptimalne optimizacije veb stranice, jer jedan element zauzima fiksnu vrednost, bez prilagođavanja promenama raspoloživog prostora.

```
<div class="wrapper">
  <div>Element 1</div>
  <div>Element 2</div>
  <div>Element 3</div>
  <div>Element 4</div>
</div>
```

HTML

```
.wrapper {
  display: grid;
  grid-template-columns: 400px 1fr 1fr;
}
```

CSS

Element 1	Element 2	Element 3
Element 4		

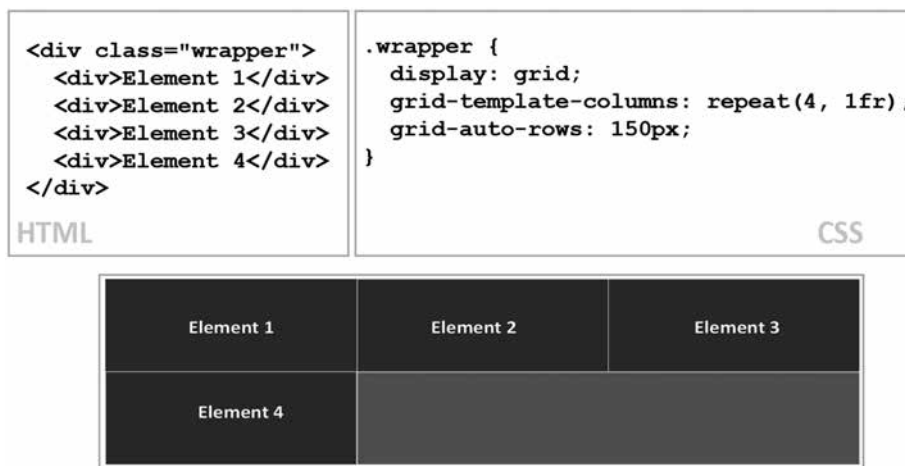
Slika 11.96. *Kombinovanje fleksibilnih i apsolutnih vrednosti*

Koristeći notaciju `repeat()` u CSS-u, sistem mreže može ponavljati određeni broj elemenata prema definisanim vrednostima.

<pre>.wrapper { display: grid; grid-template-columns: 1fr 1fr 1fr 1fr; }</pre> CSS	<pre>.wrapper { display: grid; grid-template-columns: repeat(3, 1fr); }</pre> CSS
---	--

Slika 11.97. *Korišćenje notacije `repeat()`*

Definicija kolona u CSS sistemu mreže ostvaruje se putem `grid-template-columns` svojstva, pri čemu sistem mreže automatski generiše odgovarajuće redove. Ti generisani redovi čine deo implicitne mreže. Sa druge strane, eksplicitna mreža se sastoji od definisanih redova i kolona, koje se specificiraju putem svojstva `grid-template-columns` ili `grid-template-rows`. Kada se elementi postave izvan definisane mreže, sistem mreže automatski generiše dodatne redove i kolone unutar implicitne mreže. U okviru implicitne mreže mogu se definisati i fiksne veličine, a u tu svrhu se koriste svojstva `grid-auto-rows` i `grid-auto-columns`. Sledeći primer ilustruje upotrebu `grid-auto-rows` kako bi se visina elemenata u implicitnoj mreži postavila na 150 piksela.



Slika 11.98. Podešavanje visine u implicitnoj mreži

U sistemu mreže, funkcija `minmax()` omogućava definisanje minimalne i maksimalne veličine elemenata kako bi se prilagodili raspoloživom prostoru. Ova funkcionalnost je korisna kada programer želi da osigura da element ne bude manji od određene vrednosti (npr. ne manji od 150 piksela), ali da se istovremeno može proširiti do svoje maksimalne vrednosti (npr. do 400 piksela visine). Za postizanje ovog efekta, može se koristiti `grid-auto-rows` zajedno sa funkcijom `minmax()`. U tom slučaju, redovi se automatski generišu sa definisanom minimalnom vrednošću, dok se maksimalna vrednost prilagođava automatski. U narednom primeru se demonstrira korišćenje ove tehnike kako bi se omogućilo proširivanje elemenata i prilagođavanje njihove visine sadržaju, kako bi se osiguralo dovoljno prostora za najviši element u redu (slika 11.99).

Prilikom definisanja sistema mreže, ne definišu se tačno linije, već se koristi koncept granica. U narednom primeru (slika 11.100) prikazana je mreža sa tri kolone i tri reda, pri čemu svaka kolona ima četiri granice označene brojevima od 1 do 4 (tamnija boja). U ovom primeru je prikazana samo središnja kolona radi jednostavnosti, ali isto pravilo važi i za sve ostale kolone. Slično tome, redovi imaju svoje granice označene brojevima od 1 do 4, prikazane svetlijom bojom, s leva na desno.

Prilikom postavljanja područja u sistemu mreže, može se koristiti koncept templejt područja mreže (eng. *grid template areas*), koji omogućava jednostavnije pozicioniranje eleme-

nata na mreži. Ova tehnika se može koristiti samostalno ili u kombinaciji sa definisanjem linija. Za postavljanje područja, koristi se svojstvo `grid-area`.

```

<div class="wrapper">
  <div>Element 1</div>
  <div>
    Element 2
    <p>Sadržaj elementa.</p>
    <p>Više od 150 piksela.</p>
  </div>
  <div>Element 3</div>
  <div>Element 4</div>
  <div>Element 5</div>
</div>

```

HTML

```

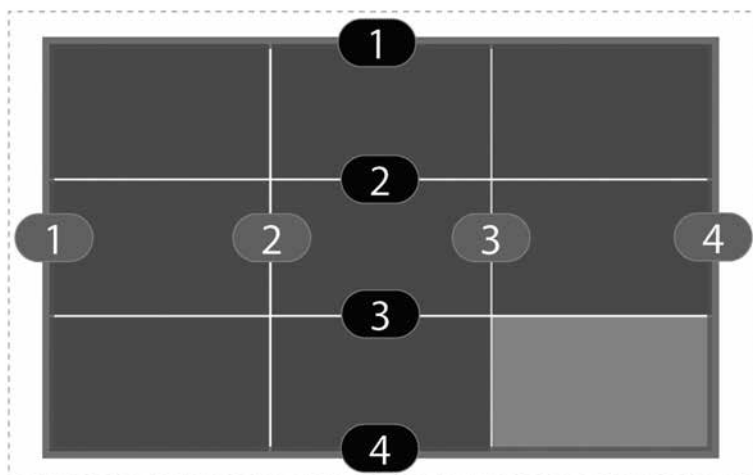
.wrapper {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  grid-auto-rows: minmax(150px, auto);
}

```

CSS

Element 1	Element 2 Sadržaj elementa. Više od 150 piksela.	Element 3
Element 4		

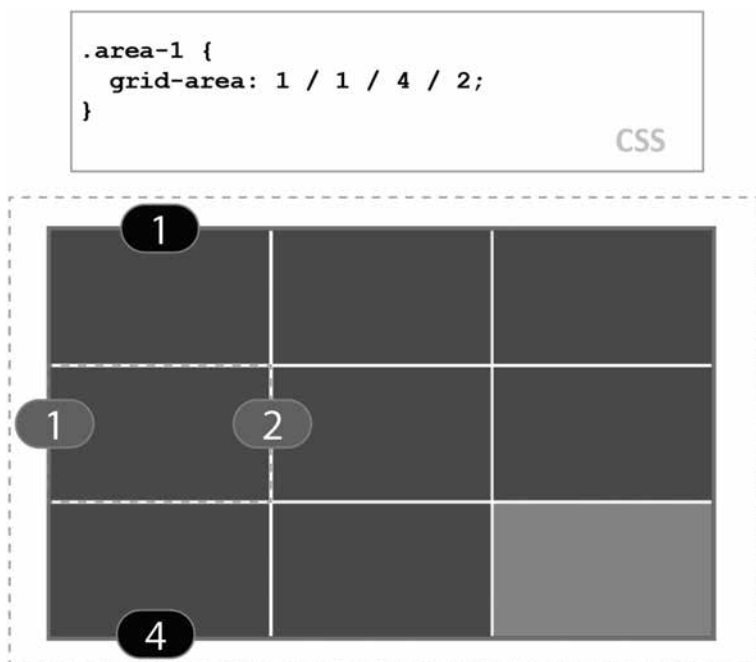
Slika 11.99. Korišćenje `minmax()` funkcije



Slika 11.100. Definisanje granica u okviru sistema mreže

Kada se koristi za celokupno područje mreže, ovo svojstvo se postavlja kao vrednost za sve četiri linije koje okružuju dati element. Kombinovanjem definicija za sve četiri linije, može se precizno odrediti područje. U sledećem primeru (slika 11.101) je prikazano ogra-

ničavanje područja klase `area-1` od prve linije reda (1) i prve linije kolone (1), do četvrte linije reda (4) i druge linije kolone (2).



Slika 11.101. Definisanje templejt područja sistema mreže

Može se primetiti da CSS sistem mreže predstavlja izuzetno efikasan instrument pri organizaciji elemenata na veb stranici. Kroz iskorišćavanje adekvatnih unapred definisanih svojstava, moguće je regulisati raspored redova i kolona, što omogućava visok stepen prilagodljivosti u pogledu zauzimanja dostupnog prostora. Takođe, unutar tih redova i kolona, moguće je precizno pozicionirati elemente. U skladu sa navedenim funkcionalnostima, postiže se visok nivo odziva veb stranice na različite dimenzije ekrana. Integracijom CSS sistema mreže sa *Flexbox* modulom, moguće je dodatno unaprediti organizaciju elemenata na veb stranici, obezbeđujući znatno veću preciznost. Osim toga, preporuka je izbegavanje kompleksnih idejnih i strukturnih rešenja CSS sistema mreže kako bi se održale visoke performanse i izbegao negativan uticaj na odziv i prilagodljivost veb stranice.

Glava 12

HTML5 i osnove računarske grafike

Element <canvas> jezika HTML5 omogućava korisniku da na lagan način definiše i koristi grafičke elemente korišćenjem jezika *JavaScript*. Mogu da se iscrtavaju grafikoni, da se prave kompozicije fotografija ili da se prave jednostavne (a i složenije) animacije.

Element <canvas> koristi dva specifična atributa `width` i `height`, kao i ostale HTML5 attribute, kao što su `id`, `name`, `class` itd. Sledeća sintaksa obuhvata prva dva atributa:

```
<canvas id="moj_canvas" width="200" height="200"></canvas>
```

Sledi jednostavniji primer sintakse za upotrebu <canvas> elementa u HTML5 dokumentu (rezultat je prikazan na slici 12.1 gore):

```
<!DOCTYPE html>
<html>
  <head>
    <title> Podesavanje CANVASA </title>
    <meta charset="UTF-8">
    <style> canvas {border: 4px solid black;} </style> </head>
  <body>
    <canvas id="canvas-element" width="800" height="600"></canvas>
  </body>
</html>
```

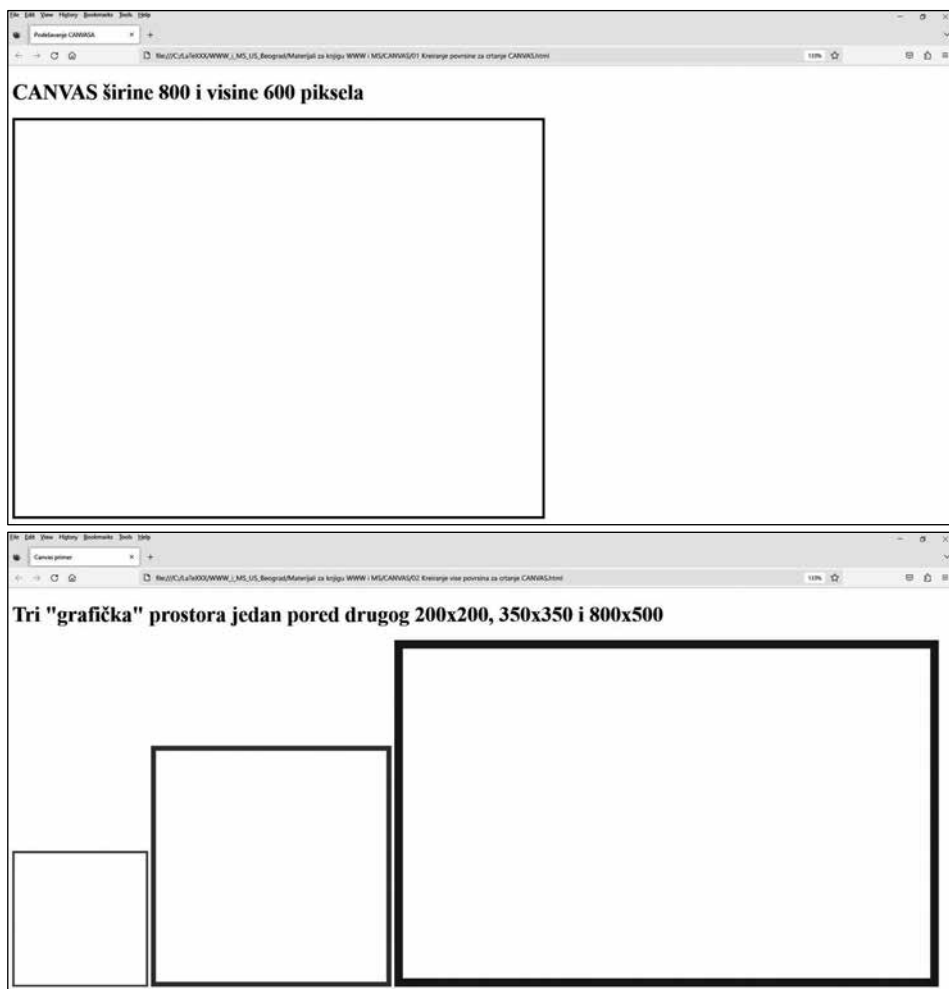
Sledi malo složeniji primer sintakse za upotrebu <canvas> elementa u HTML5 dokumentu (rezultat je prikazan na slici 12.1 dole):

```
<!DOCTYPE html>
<html>
  <head>
    <title>Canvas primer</title>
    <meta charset="UTF-8">
    <style>
      #moj_canvas{border:3px solid red;}
      #moj_canvas_01{border:8px solid green;}
    </style>
  </head>
  <body>
    <canvas id="moj_canvas"></canvas>
    <canvas id="moj_canvas_01"></canvas>
  </body>
</html>
```

```

        #moj_canvas_02{border:13px solid blue;}
    </style>
</head>
<body>
<h1> Tri graficka prostora 200x200, 350x350 i 800x500 </h1>
<canvas id="moj_canvas" width="200" height="200"></canvas>
<canvas id="moj_canvas_01" width="350" height="350"></canvas>
<canvas id="moj_canvas_02" width="800" height="500"></canvas>
</body>
</html>

```

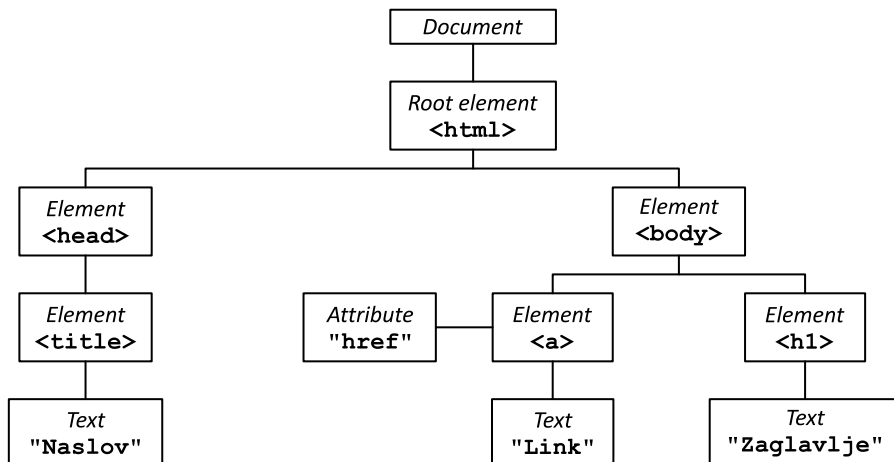


Slika 12.1. Kreiranje osnovne grafičke oblasti¹²

¹01 Kreiranje površine za crtanje CANVAS.html

²02 Kreiranje više površina za crtanje CANVAS.html

Pomoću HTML DOM (*Document Object Model*) programski jezik *JavaScript* može da pristupi svim elementima HTML dokumenta i da ih menja. Kada se učita veb stranica, pretraživač kreira DOM na toj stranici. Model HTML DOM se konstruiše ili kreira kao stablo objekata, kao što je prikazano na slici 12.2.



Slika 12.2. HTML DOM stablo objekata

Sa primenom ovog modela, *JavaScript* ima sve što mu treba da kreira dinamički HTML:

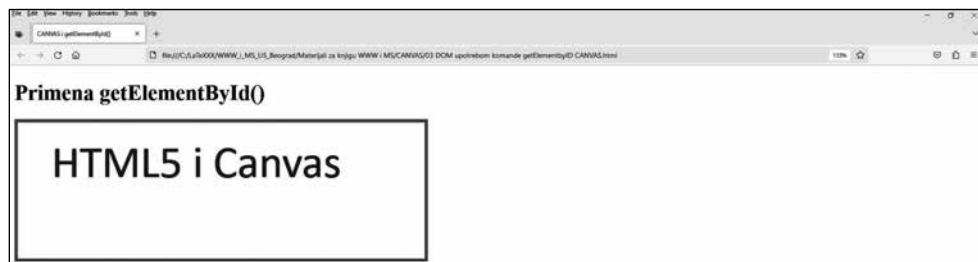
- *JavaScript* može da menja sve HTML elemente na stranici;
- *JavaScript* može da menja sve HTML atribute na stranici;
- *JavaScript* može da menja sve CSS stilove unutar stranice;
- *JavaScript* može da premesti ili ukloni postojeće HTML elemente i atribute;
- *JavaScript* može da dodaje nove HTML elemente i atribute;
- *JavaScript* može da odreaguje na sve postojeće HTML događaje unutar stranice;
- *JavaScript* može da kreira nove HTML događaje unutar stranice.

Korisnik može da koristi `<canvas>` kao DOM korišćenjem metode `getElementById()` kao što je prikazano sledećom sintaksom:

```
var canvas = document.getElementById('moj_canvas');
```

Sledi primer sintakse za upotrebu DOM upotrebom metode `getElementById()` u fajlu pod brojem 3, a rezultat je prikazan na slici 12.3.

Element `<canvas>` je inicijalno prazan, a da bi se prikazao neki sadržaj unutar pomenutog elementa *JavaScript* mora, najpre, da pristupi renderovanom kontekstu i da sadržaj "nacrti". Ovaj element ima DOM metod pod nazivom `getContext` koji se koristi za pristupanje renderovanom kontekstu i funkcijama za crtanje.



Slika 12.3. *Tekst unutar grafičke oblasti³*

Sintaksa za ovu metodu je sledeća:

```
canvas.getContext(contextType, contextAttributes);
```

Identifikator contextType može da ima sledeće vrednosti:

- 2d – kreira dvodimenzionalni renderovan kontekst;
- webgl ili experimental-webgl – kreira trodimenzionalni renderovan kontekst i ovaj kontekst je dostupan samo kod pretraživača (*browsers*) koji su implementirali WebGL verziju 1 (OpenGL ES 2.0);
- webgl2 – kreira trodimenzionalni renderovan kontekst i ovaj kontekst je dostupan samo kod pretraživača koji su implementirali WebGL verziju 2 (OpenGL ES 3.0);
- bitmaprenderer – omogućava zamenu <canvas> sadržaja definisanom bitmapom.

Inicijalna sintaksa za nekog ko nije baš vičan programskom jeziku *JavaScript* prikazana je ispod:

```
<!DOCTYPE HTML>
<head><title> TITLE </title>
  <meta charset="UTF-8">
  <h1> NASLOV </h1>
</head>
<html>
  <body onload="draw();">
    <canvas id="canvas" width="700" height="700" style="border:2px
      solid #000000;"></canvas>
  </body>
  <script type="text/javascript">
function draw() {
var canvas = document.getElementById('canvas');
if (canvas.getContext) {
var ctx = canvas.getContext('2d');
```

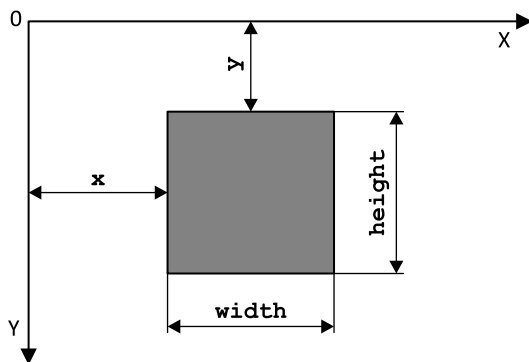
³ 03 DOM upotrebom komande getElementbyID CANVAS.html

```
// OVDE SE UNOSI SINTAKSA ZA CANVAS!!!!!!
    } }
</script>
</html>
```

12.1. Pravougaoni elementi i površi

Postoje tri metode ili načina da se iscrtaju pravougaone površine u HTML5 primenom `<canvas>` elementa:

- `fillRect(x, y, width, height)` – iscrtava se ispunjena pravougaona površina (pravougaonik ili kvadrat);
- `strokeRect(x, y, width, height)` – iscrtava se konturna linija pravougaone površine;
- `clearRect(x, y, width, height)` – briše ili čisti definisanu površinu i kreira je da bude kompletno transparentna ili providna.

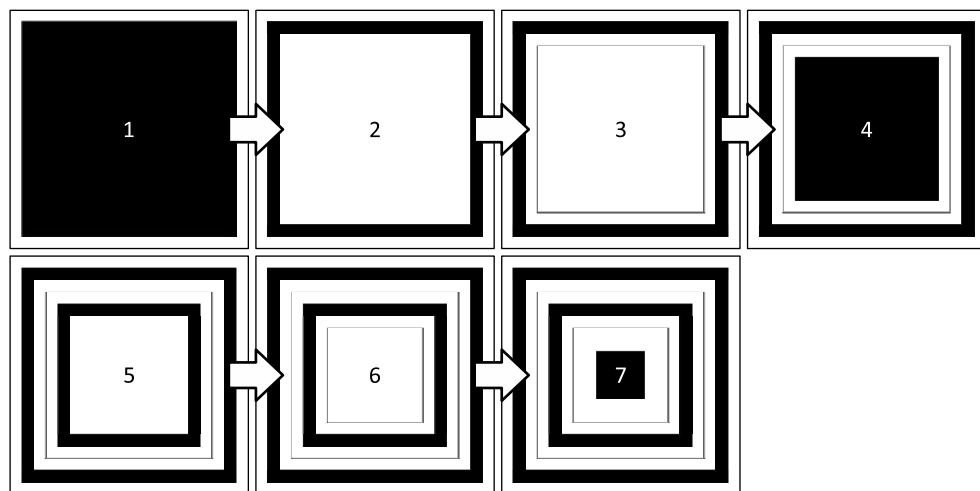


Slika 12.4. *Ekranjski koordinatni sistem*

U gornjim sintaksama `x` i `y` definišu položaj gornjeg levog ugla elementa `<canvas>` (relativno u odnosu na koordinatni početak), a `width` i `height` definišu širinu i visinu pravougaone površine ili konture.

Sledi primer iscrtavanja pravougaonih, tj. kvadratnih površina, a sintaksa je definisana u fajlu pod brojem 4 (slika 12.5). Deo bitne sintakse po pitanju geometrije je prikazan ispod:

```
ctx.fillRect(25, 25, 450, 450); -----> 1
ctx.clearRect(50, 50, 400, 400); -----> 2
ctx.strokeRect(75, 75, 350, 350); -----> 3
ctx.fillRect(100, 100, 300, 300); -----> 4
ctx.clearRect(125, 125, 250, 250); -----> 5
ctx.strokeRect(150, 150, 200, 200); -----> 6
ctx.fillRect(200, 200, 100, 100); -----> 7
```

Slika 12.5. Iscrtavanje kvadratnih površi⁴

12.2. Linije, završeci linija i zatvorene konture

Sledi spisak metoda da bi se iscrtale odgovarajuće linije unutar <canvas> elementa:

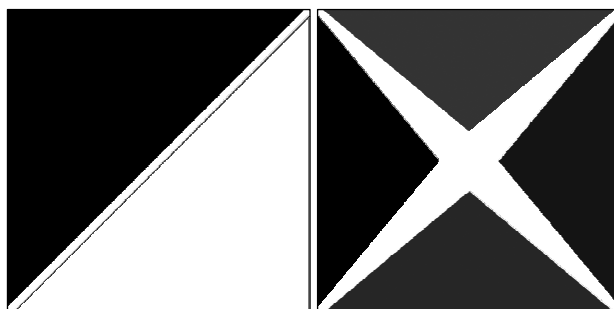
- `beginPath()` – resetuje tekuću putanju za liniju;
- `moveTo(x, y)` – kreira novu putanju u definisanoj tački;
- `lineTo(x, y)` – iscrtava liniju do definisane koordinate, iscrtava novu liniju, a njena početna tačka je završna tačka prethodne linije;
- `closePath()` – markira tekuću putanju kao zatvorenu;
- `fill()` – ispunjava tekuću putanju odgovarajućom ispunom;
- `stroke()` – iscrtava konturu po putanji.

Postoji i nekoliko osobina koje omogućavaju da se podese stilovi linija:

- `lineWidth [ = value ]` – ova osobina prikazuje tekuću širinu linije i može da promeni širinu linije na novu vrednost;
- `lineCap [ = value ]` – ova osobina prikazuje tekući završetak linije i može da podesi novi završetak (mogući stilovi za završetak linije su `butt`, `round` i `square`);
- `lineJoin [ = value ]` – ova osobina prikazuje tekući način "spajanja" ili povezivanja dve linije i može da podesi novi način povezivanja (mogući načini povezivanja su `bevel`, `round` i `miter`);

⁴ 04 Pravougaoni elementi i površi CANVAS.html

- `miterLimit [ = value ]` – ova osobina prikazuje tekuća ograničenja miter načina povezivanja linija i može da promeni taj odnos ili ograničenja.



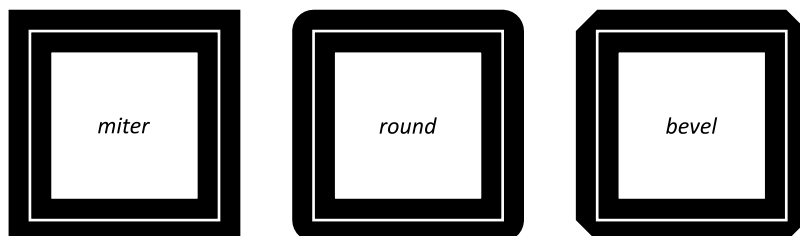
Slika 12.6. Iscrtavanje proizvoljnih zatvorenih kontura^{5 6}

Sledi primer osobine `lineCap` sa vrednostima "butt", "round" i "square" koja može da prikaže tekući završetak linije i može da podesi novi završetak (slika 12.7). Unutar prikazane sintakse je definisana i širina linije pomoću osobine `lineWidth = 20`.



Slika 12.7. Specifični završeci linija bez spajanja⁷

Što se spajanja linija tiče, postoje tri načina za njihovo spajanje, kao što je to prikazano na slici 12.8. Na pomenutoj slici belom linijom je prikazana putanja zamišljenog pera koji iscrtava liniju, gde polovina širine pera ide sa jedne strane bele linije, a druga polovina sa druge strane.



Slika 12.8. Različiti načini spajanja linija

⁵ 05 Crtanje crnog i belog trougla CANVAS.html

⁶ 06 Crtanje 4 trougla u boji CANVAS.html

⁷ 07 Specificni završeci linija CANVAS.html

Sledi primer osobine `lineJoin` sa vrednostima "miter", "round" i "bevel" koja može da prikaže tekući način "spajanja" ili povezivanja dve linije i može da podesi novi način povezivanja (slika 12.9). Unutar prikazane sintakse je definisana i širina linije pomoću osobine `lineWidth` = 20.



Slika 12.9. *Specifični završeci linija sa spajanjem*⁸

Očigledno je da je najbolja varijanta koristiti osobine:

```
lineCap = "round";
lineJoin = "round";
```

12.3. Kružnice i opšti kružni lukovi

Sledi spisak metoda da bi se iscrtale odgovarajuće linije, kružnice i kružni lukovi unutar `<canvas>` elementa:

- `beginPath()` – resetuje tekuću putanju za liniju;
- `moveTo(x, y)` – kreira novu podputanju u definisanoj tački;
- `lineTo(x, y)` – iscrtava liniju do definisane koordinate;
- `closePath()` – markira tekuću podputanju kao zatvorenu;
- `fill()` – ispunjava tekuću podputanju odgovarajućom ispunom;
- `stroke()` – iscrtava konturu po podputanji;
- `arc(x, y, radius, startAngle, endAngle, anticlockwise)` – iscrtava putanju u obliku kružnog luka i taj luk je opisan koordinatama centra, poluprečnikom, definisanim početnim i završnim uglom luka i kojim putem se iscrtava luk (u smeru kretanja kazaljke na satu ili u suprotnom smeru);
- `fillStyle = "BOJA"` – definiše boju kojom će ispuniti zatvorena kontura;
- `strokeStyle = "BOJA"` – definiše boju kojom će biti iscrtana ("obojena") granična ili konturna linija objekta koji je nacrtan;

⁸ 08 Specifichni zavrseci linija i njihovo spajanje CANVAS.html

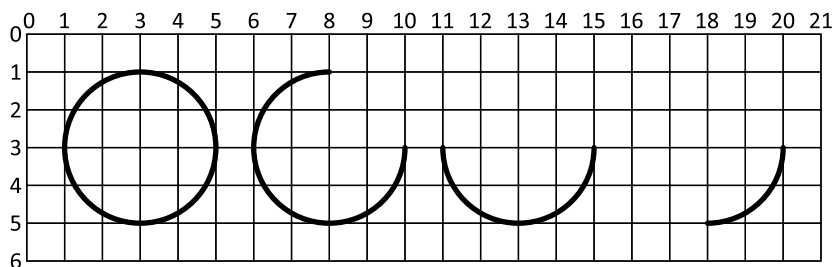
- `lineWidth = VREDNOST` – definiše "debljinu" (širinu) granične ili konturne linije i uobičajeno je da se ova vrednost daje u pikselima.

NAPOMENA: Kada se pozove `fill()`, svaka otvorena putanja se automatski zatvara, tako da nije neophodno da se aktivira `closePath()`. Ovo nije slučaj kada se pozove `stroke()`.

NAPOMENA: Uglovi unutar `arc` funkcije mere se u radijanima, a ne u stepenima. Da bi se stepeni pretvorili u radijane, treba koristiti *JavaScript* izraz

```
radians = (Math.PI/180)*degrees.
```

Na primer, treba nacrtati jednu ispunjenu kružnicu i tri kružna luka, kao što je to prikazano na slici 12.10.



Slika 12.10. Kružnica i tri luka

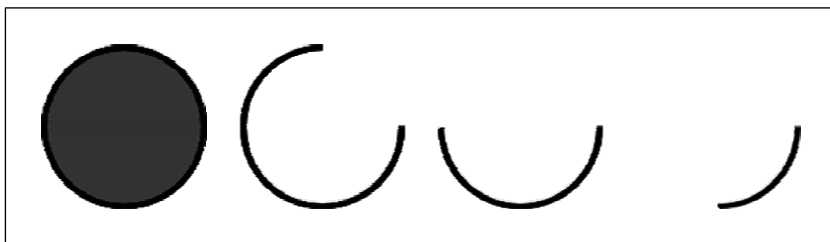
Treba napomenuti da je u primeru pod brojem 9 usvojeno da je reč o kvadratnoj mreži gde je stranica svakog pojedinačnog kvadrata 30 piksela. Širina linije je 4 piksela.

```
// Pun krug
ctx.arc(90, 90, 60, 0, Math.PI*2, true);
ctx.fillStyle = "red";
ctx.fill();
ctx.strokeStyle = "black";
ctx.lineWidth=4;
ctx.stroke();

// 3/4 kruga
ctx.moveTo(300, 90);
ctx.arc(240, 90, 60, 0, Math.PI*1.5, false);

// 1/2 kruga
ctx.moveTo(450, 90);
ctx.arc(390, 90, 60, 0, Math.PI*1, false);

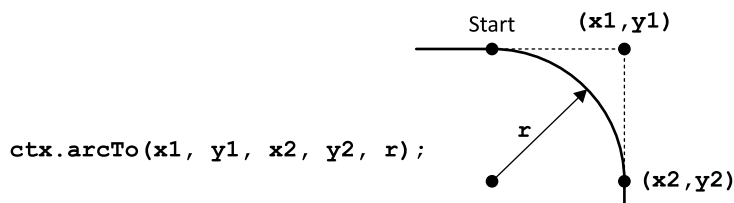
// 1/4 kruga
ctx.moveTo(600, 90);
ctx.arc(540, 90, 60, 0, Math.PI*0.5, false);
```



Slika 12.11. *Iscrtan crveni krug i tri luka⁹*

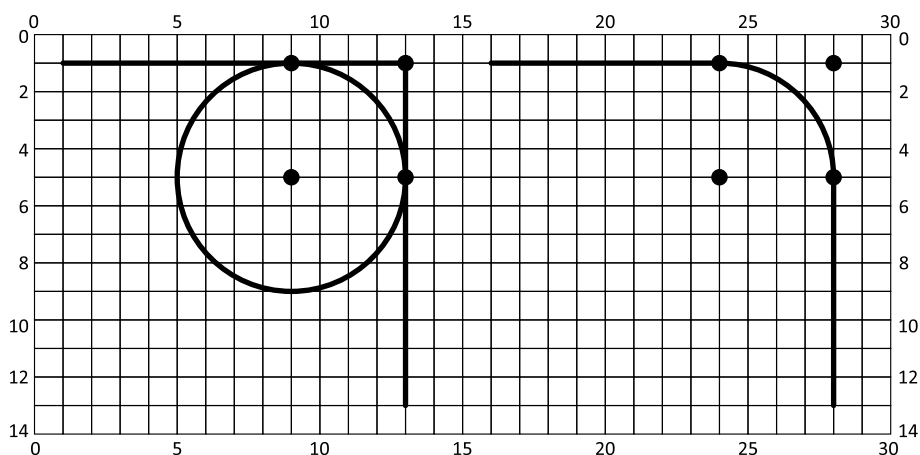
Sledi metoda da bi se iscrtao kružni luk koji tangira dve odgovarajuće linije (te linije su tangente na dati kružni luk), koji je deo kružnice, u dodirnim tačkama:

- `arc(x1, y1, x2, y2, radius)` – iscrtava putanju u obliku kružnog luka i taj luk počinje gde se prvi pravolinijski segment završio, a opisan je koordinatama potencijalne presečne tačke gde bi se linije, koje luk tangira, presekle, koordinatama završne tačke luka (iz te tačke "kreće" druga linija) i poluprečnikom (slika 12.12).



Slika 12.12. *Slikovito objašnjenje luka koji tangira dve prave linije*

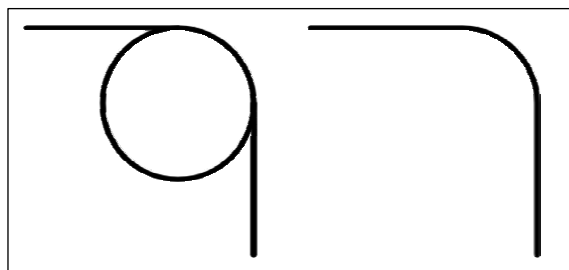
Na primer, treba nacrtati jednu ugaonu "garnituru" od dve linije i kružnice koja tangira te dve linije u levom delu, a u desnom delu treba nacrtati luk koji tangira nacrtane dve linije (desni deo), kao što je to prikazano na slici 12.13.



Slika 12.13. *Početno stanje i luk koji tangira dve prave linije*

⁹ 09 Crtanje kruznica u kruznih lukova CANVAS.html

Treba napomenuti da je u primeru por brojem 10 usvojeno da je reč o kvadratnoj mreži gde je stranica svakog pojedinačnog kvadrata 20 piksela. Širina linije je 5 piksela (slika 12.14).

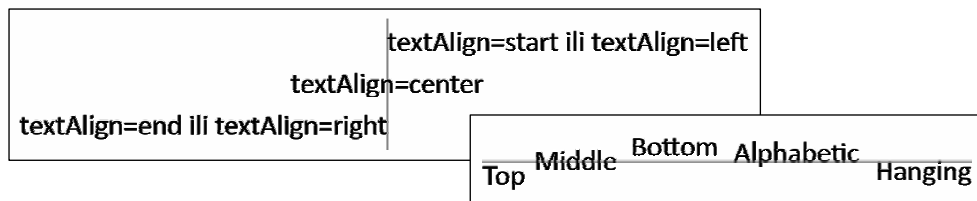


Slika 12.14. Luk koji tangira dve prave linije¹⁰

12.4. Tekst

HTML5 <canvas> omogućava da se kreira željeni tekst primenom različitih fontova i odgovarajućih osobina, koje se nabrojane ispod:

- `font [ = value ]` – osobina koja prikazuje tekuće podešavanje i omogućava da se promeni font;
- `textAlign [ = value ]` – osobina koja prikazuje tekuće podešavanje poravnanja teksta i omogućava da se to setovanje promeni; moguće su sledeće vrednosti `start`, `end`, `left`, `right` i `center`.
- `textBaseline [ = value ]` – osobina koja prikazuje podešeno poravnanje osnovne linije teksta i omogućava da se to setovanje promeni; moguće su sledeće vrednosti `top`, `hanging`, `middle`, `alphabetic`, `ideographic` i `bottom`.
- `fillText(text, x, y)` – osobina koja "ispunjava" dati text na datoj lokaciji koja je definisana koordinatama `x` i `y`;
- `strokeText(text, x, y)` – osobina koja ispisuje konturu text-a na lokaciji koja je definisana koordinatama `x` i `y`.



Slika 12.15. Pozicioniranje teksta i po horizontali i po vertikali^{11 12}

¹⁰ 10 Crtanje lukova koji tangiraju dve linije CANVAS.html

¹¹ 11 Poravnanje teksta po horizontali grupisano CANVAS.html

¹² 13 Poravnanje teksta po vertikali CANVAS.html

Ako tekst treba centrirati unutar grafičke oblasti, onda može da se iskoristi sledeća sintaksa (slika 12.16) koja je sastavno deo fajla pod brojem 14:

```
// Definisanje fonta i njegove velicine
ctx.font="120px Calibri";
ctx.textBaseline = "middle";
ctx.textAlign = "center";
ctx.fillStyle="blue";
ctx.fillText("CENTAR", canvas.width/2, canvas.height/2);
```



Slika 12.16. *Centriranje teksta i po horizontali i po vertikali*¹³

Osobine `fillText` i `strokeText` omogućavaju da se ispisuje "ispunjen" tekst ili samo kontura željenog teksta (slika 12.17).



Slika 12.17. *Raznoliko ispisivanje teksta*^{14 15 16}

12.5. FOR petlja

Mnogo puta se javlja potreba da se unutar programa neke linije koda izvršavaju više puta. Ovo je moguće rešiti pomoću odgovarajućih upravljačkih struktura. Petlje i iteracije su osnovni gradivni element svakog programa. Standardne petlje se javljaju manje više u istom obliku u skoro svim programskim jezicima. Najkorišćenija je **FOR petlja**. Uopštena sintaksa za klasičnu FOR petlju je:

¹³ ¹⁴ Centriranje teksta i po horizontali i po vertikali CANVAS.html

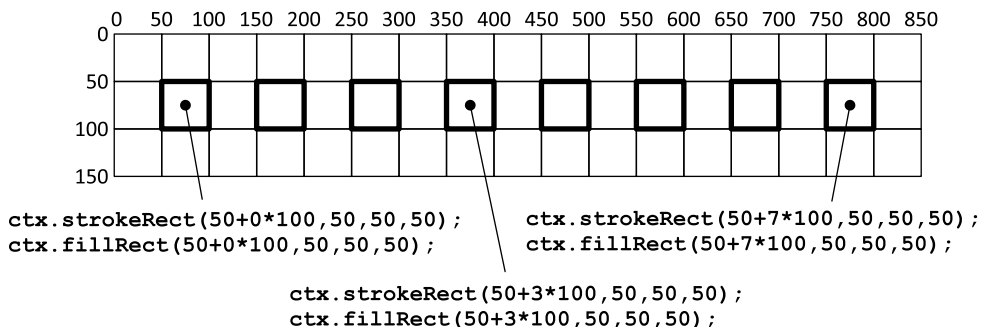
¹⁴ ¹⁵ Ispunjen tekst CANVAS.html

¹⁵ ¹⁶ Tekst kao kontura CANVAS.html

¹⁶ ¹⁷ Ispunjen tekst i konturni tekst CANVAS.html

```
for ([pocetna vrednost]; [uslov]; [povecenje ili smanjenje])
    { izraz za izvršenje }
```

Na primer, treba nacrtati 8 kvadrata i postaviti ih u red, prema pratećoj geometriji, kao što je to prikazano na slici 12.18.



Slika 12.18. Geometrija postavljenog zadatka i pojedinačni koraci

Opšta sintaksa za iscrtavanje 8 svetlo plavih kvadrata prikazana je ispod, a rezultat je vidljiv na slici 12.19 gore:

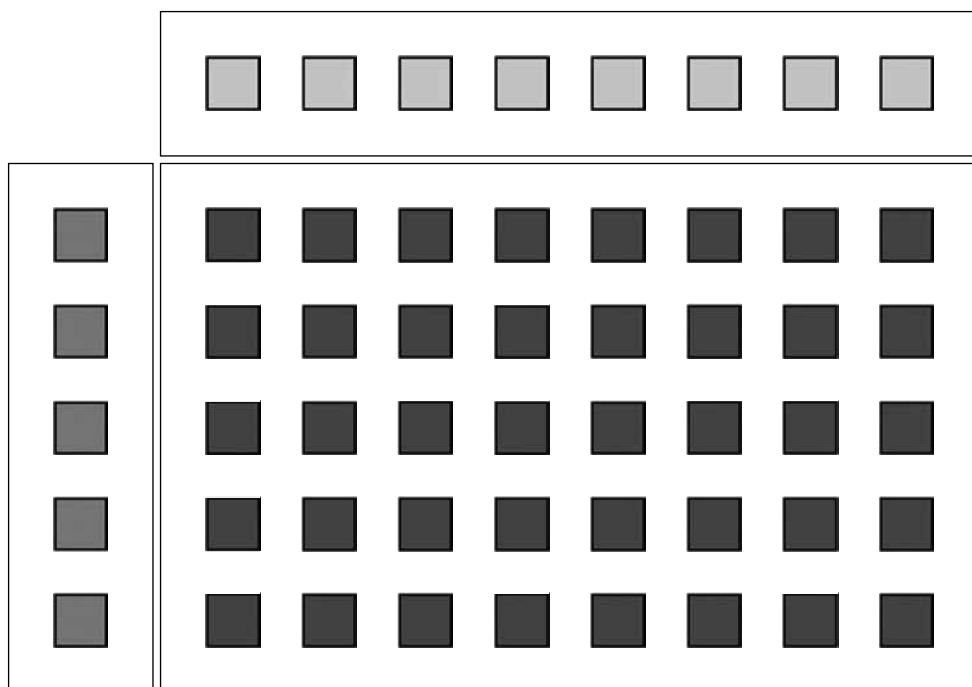
```
// Iscrtavanje 8 svetlo plavih kvadrata u desnu stranu
for(i=0; i<8; i++) {
    ctx.beginPath();
    ctx.fillStyle = "cyan";
    ctx.fill();
    ctx.strokeStyle = "black";
    ctx.lineWidth=5;
    ctx.strokeRect(50+i*100, 50, 50, 50);
    ctx.fillRect(50+i*100, 50, 50, 50);
    ctx.strokeStyle = "black";
    ctx.stroke();
}
```

Opšta sintaksa za iscrtavanje 5 narandžastih kvadrata u jednoj koloni prikazana je ispod, a rezultat je vidljiv na slici 12.19 levo:

```
// Iscrtavanje 5 narandžastih kvadrata u donju stranu
for(j=0; j<5; j++) {
    ctx.beginPath();
    ctx.fillStyle = "orange";
    ctx.fill();
    ctx.strokeStyle = "black";
    ctx.lineWidth=5;
    ctx.strokeRect(50, 50+j*100, 50, 50);
    ctx.fillRect(50, 50+j*100, 50, 50);
    ctx.strokeStyle = "black";
    ctx.stroke();
}
```


Opšta sintaksa za iscrtavanje matrice 5×8 (5 redova i 8 kolona) ljubičastih kvadrata prikazana je ispod (slika 12.19 dole desno):

```
// Iscrtavanje 5x8 ljubicastih kvadrata u desnu i donju stranu
for( i=0; i<8; i++) {
  for( j=0; j<5; j++) {
    ctx.beginPath();
    ctx.fillStyle = "magenta";
    ctx.fill();
    ctx.strokeStyle = "black";
    ctx.lineWidth=5;
    ctx.strokeRect(50+i*100, 50+j*100, 50, 50);
    ctx.fillRect(50+i*100, 50+j*100, 50, 50);
    ctx.stroke();      } }
```



Slika 12.19. *Crtanje 1 reda, 1 kolone i matrice od 5 redova i 8 kolona*^{17 18 19}

12.6. Definisanje boja i providnost

Sintaksa za žutu boju na tri različita načina:

^{17 18} FOR petlja za 1 red CANVAS.html

^{18 19} FOR petlja za 1 kolonu CANVAS.html

^{19 20} FOR petlja za 5 redova i 8 kolona CANVAS.html

- `ctx.fillStyle = "yellow";` – definisanje boje pomoću njenog naziva;
- `ctx.fillStyle = "#FFFF00";` – definisanje boje pomoću njene heksadecimalne oznake;
- `ctx.fillStyle = "rgb(255, 255, 0)";` – definisanje boje pomoću RGB koordinata.
- `ctx.fillStyle = "rgba(255, 255, 0, 0.3)";` – definisanje boje pomoću RGB koordinata i definisanje providnosti.

Prva tri načina su opšte poznata i do sada su nebrojeno puta korišćena. Poslednji (četvrti) način za definisanje boje je:

```
ctx.fillStyle = "rgba(R, G, B, alpha)";
```

U gornjoj sintaksi nalaze se sledeće oznake:

- **R (red)** – definiše intenzitet crvene osnovne boje kao celobrojnu vrednost između 0 i 255, ili kao procentualnu vrednost između 0% i 100%.
- **G (green)** – definiše intenzitet zelene osnovne boje kao celobrojnu vrednost između 0 i 255, ili kao procentualnu vrednost između 0% i 100%.
- **B (blue)** – definiše intenzitet plave osnovne boje kao celobrojnu vrednost između 0 i 255, ili kao procentualnu vrednost između 0% i 100%.
- **alpha** – definiše providnost ili prozirnost kao vrednost između 0.0 (totalno providno ili potpuno prozirno) i 1.0 (nema providnosti).

Jednostavnim prisustvom (100%) ili odsustvom (0%) jedne od tri osnovne boje (crvena, zelena, plava) može se dobiti sledećih osam boja, kao što je to prikazano u tabeli 12.1.

Tabela 12.1. "Granične" kombinacije tri osnovne boje

Srpski naziv	Engleski naziv	Heksadecimalni broj	R	G	B
Crna	<i>Black</i>	#000000	0	0	0
Bela	<i>White</i>	#FFFFFF	255	255	255
Crvena	<i>Red</i>	#FF0000	255	0	0
Zelena	<i>Green</i>	#00FF00	0	255	0
Plava	<i>Blue</i>	#0000FF	0	0	255
Žuta	<i>Yellow</i>	#FFFF00	255	255	0
Ciklama	<i>Magenta</i>	#FF00FF	255	0	255
Cijan	<i>Cyan</i>	#00FFFF	0	255	255

Sledeća funkcija definiše providnost ili prozirnost elementa, a vrednost providnosti se definiše procentualno od 0% (0.0) do 100% (1.0):

- `globalAlpha` – definiše providnost (prozirnost) boje i objekta.

Način korišćenja ove funkcije je prikazan kroz donju sintaksu:

```
ctx.globalAlpha = 0.0 ... 1.0;
```



Slika 12.20. Pozadinska boja i providnost^{20 21}

12.7. Gradijentni prelazi boja

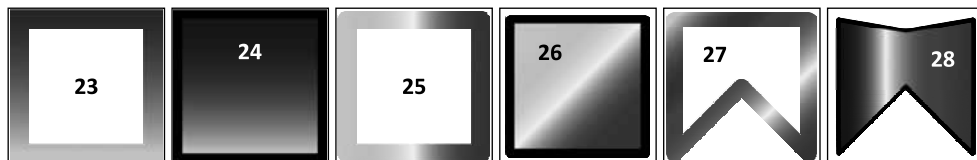
Sledi spisak metoda za kreiranje gradijentnih prelaza – linearnog i radijalnog:

- `addColorStop(offset, color)` – ova metoda definiše zaustavnu boju gradijenta na određenom rastojanju ili pomeraju (*offset*); vrednost 0.0 definiše pomeraj na jednom kraju gradijentnog prelaza, a 1.0 definiše pomeraj na drugom kraju.
- `createLinearGradient(x0, y0, x1, y1)` – ova metoda definiše linearni gradijentni prelaz duž zamišljene linije koja je definisana koordinatama kao argumentima; četiri argumenta definišu početnu (x_0, y_0) i završnu (x_1, y_1) tačku gradijentnog prelaza.
- `createRadialGradient(x0, y0, r0, x1, y1, r1)` – ova metoda definiše radijalni (kružni) prelaz između dve zamišljene kružnice koje su predstavljene argumentima; prva tri argumenta definišu prvu kružnicu tačkom centra (x_0, y_0) i njenim poluprečnikom r_0 , a druga kružnica je definisana njenim centrom (x_1, y_1) i pripadajućim poluprečnikom r_1 .
- `fillStyle` – ovaj atribut definiše boju ili stil koji bi trebalo da se primene unutar konture;
- `strokeStyle` – ovaj atribut definiše boju ili stil koji bi trebalo primeniti na konturnu liniju.

Sledi šest primera gde se vidi kako se vrši kreiranje linearnih gradijentnih prelaza i konture i ispune. Rezultati pomenutih fajlova prikazani su na slici 12.21.

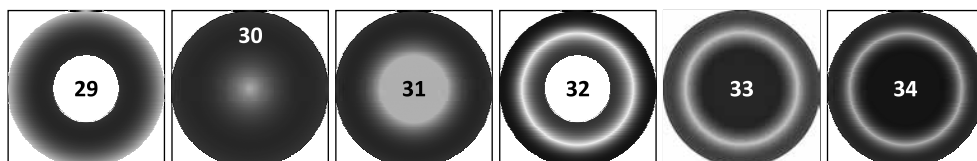
^{20 21} Definisane boje kao pozadine teksta CANVAS.html

^{21 22} Definisane boje i providnosti CANVAS.html



Slika 12.21. Linearni gradijentni prelazi 2, 3 i 4 boje^{22 23 24 25 26 27}

Na slici 12.22 prikazano je šest primera kreiranja radijalnih gradijentnih prelaza i konture i ispune (slika 12.22).



Slika 12.22. Radijalni gradijentni prelazi 2 i 3 boje^{28 29 30 31 32 33}

12.8. Umetanje fotografija u grafičku oblast

Da bi se "ubacila" fotografija unutar grafičke oblasti, najbolje je koristiti PNG fotografije, jer je u ovom grafičkom formatu moguće neutralisati pozadinu, tj. pozadina može da bude transparentna ili providna (**ne postoji**). Vidljivi su samo objekti u prednjem planu i onda nema nikakvih problema sa postavljanjem ovakvih fotografija u <canvas>, bez obzira šta se nalazi u njegovoj pozadini.

Fotografije ili slike mogu da se uvezu u tekući dokument pomoću:

- `drawImage(image, x, y, width, height)` – ubacuje eksternu sliku unutar elementa <canvas>; image predstavlja referencu na sliku ili <canvas> objekat; parametri x i y formiraju koordinatu gde će biti slika biti smeštena unutar određnog elementa <canvas> (njen gornji levi ćošak), a parametri width i height definišu širinu i visinu slike kako će se pojaviti unutar grafičke oblasti.

Unutar fajla pod brojem 35 umeću fotografije u grafičku oblast (u ovom primeru to je fotografija drveta u PNG formatu rezolucije 500 × 500 piksela). Kao rezultat se pojavljuje

²² 23 Linearni gradijentni prelaz 2 boje konture CANVAS.html

²³ 24 Linearni gradijentni prelaz 2 boje unutrasnjosti CANVAS.html

²⁴ 25 Linearni gradijentni prelaz 3 boje konture CANVAS.html

²⁵ 26 Linearni gradijentni prelaz 3 boje unutrasnjosti CANVAS.html

²⁶ 27 Linearni gradijentni prelaz 4 boje konture CANVAS.html

²⁷ 28 Linearni gradijentni prelaz 4 boje unutrasnjosti CANVAS.html

²⁸ 29 Radijalni gradijentni prelaz 2 boje konture CANVAS.html

²⁹ 30 Radijalni gradijentni prelaz 2 boje unutrasnjosti 1 CANVAS.html

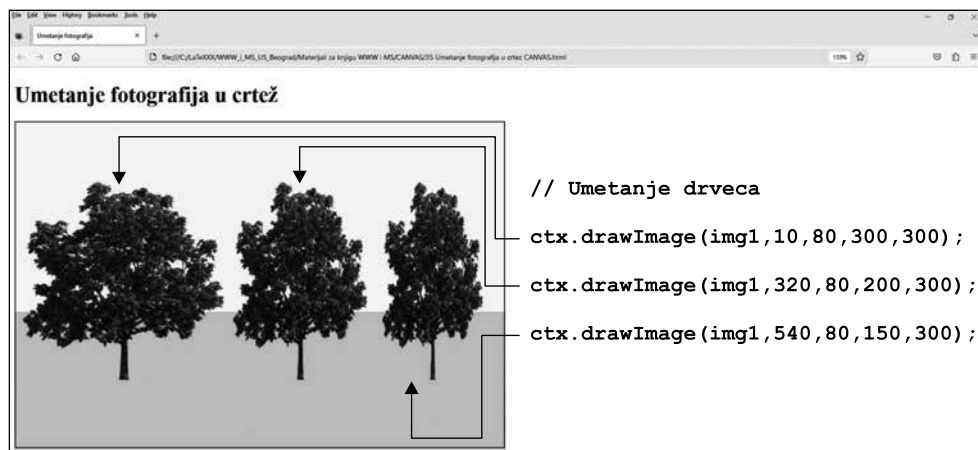
³⁰ 31 Radijalni gradijentni prelaz 2 boje unutrasnjosti 2 CANVAS.html

³¹ 32 Radijalni gradijentni prelaz 3 boje konture CANVAS.html

³² 33 Radijalni gradijentni prelaz 3 boje unutrasnjosti 1 CANVAS.html

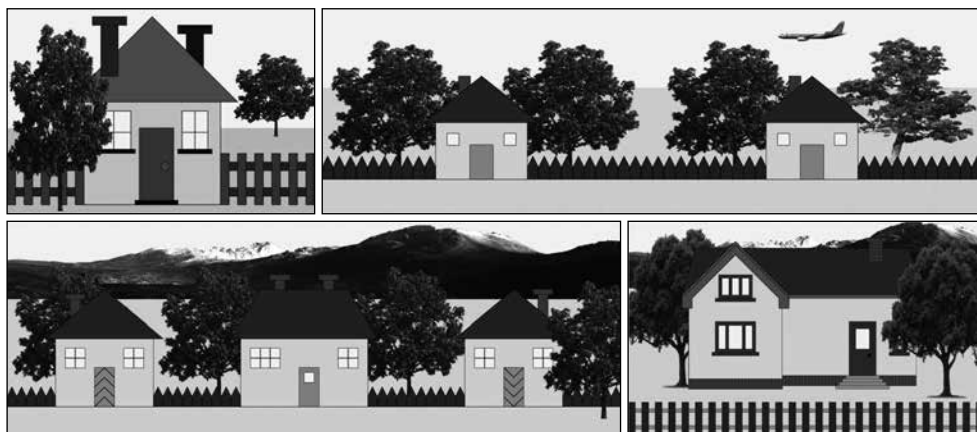
³³ 34 Radijalni gradijentni prelaz 3 boje unutrasnjosti 2 CANVAS.html

slika 12.23 gde je upotrebljena jedna PNG fotografija na tri lokacije (povezane sintakse sa određenim lokacijama) i gde sve tri lokacije imaju istu visinu od 300px, ali širina se respektivno smanjuje od 300 do 150 px.



Slika 12.23. *Umetanje fotografija u grafičku oblast*³⁴

Treba napomenuti da je bitno gde stoji, u programskom kodu, sintaksa za umetanje fotografija, jer to određuje kako će se fotografija prikazati u grafičkoj oblasti; da li će drvo zaklanjati kuću (slika 12.24 gore levo), da li će drveće biti iza ograde i kuća (slika 12.24 gore desno), da li će planina zaista biti u pozadini svih elemenata unutar grafičke oblasti (slika 12.24 dole, levo i desno).



Slika 12.24. *Umetanje fotografija u crteže*³⁵³⁶³⁷³⁸

³⁴ 35 Umetanje fotografija u crtez CANVAS.html

³⁵ 36 Umetanje PNG fotografija u crtez 1 CANVAS.html

³⁶ 37 Umetanje PNG fotografija u crtez 2 CANVAS.html

³⁷ 38 Umetanje PNG fotografija u crtez 3 CANVAS.html

³⁸ 39 Umetanje PNG fotografija u crtez 4 CANVAS.html

12.9. Transformacije objekata

U ovom delu biće reči o tri komande koje mogu da transformišu objekat. To su komande za pomeranje koordinatnog početka iz uobičajene lokacije (gornji levi ugao grafičke oblasti), skaliranje i rotiranje.

HTML5 `<canvas>` omogućava komandom `ctx.translate(x, y)`³⁹ da se pomera element i njegov koordinatni početak na drugu lokaciju. Argument `x` omogućava pomeranje levo ili desno (po X osi), a argument `y` omogućava pomeranje gore ili dole (po Y osi).

HTML5 `<canvas>` omogućava komandom `ctx.scale(x, y)`^{40 41 42 43 44 45 46} da skalira element (povećava ili smanjuje). Kada se upotrebi ova komanda, sve iza nje se skalira i definisani parametri se primenjuju sve dok se fizički ne vrate na stare (početne) vrednosti. I pozicija elementa se, takođe, skalira!

Kod komande `ctx.scale(x, y)` argument `x` skalira širinu (po X osi) tekućeg crteža (1 = 100%, 0.5 = 50%, 2 = 200% itd), a argument `y` skalira širinu (po Y osi) tekućeg crteža (1 = 100%, 0.5 = 50%, 2 = 200% itd).

Ako se primeni komanda `scale(2,2)`, crteži će biti postavljeni dvostruko dalje od leve i gornje ivice grafičke oblasti `canvas` (u odnosu na koordinatni početak u gornjem levom uglu same grafičke oblasti) i biće uvećani duplo po obe ose (proporcionalno uvećanje).

HTML5 `<canvas>` omogućava komandom `ctx.rotate(ugao)`^{47 48 49 50} da se rotira element oko njegovog koordinatnog početka. Argument `ugao` definiše ugao rotacije u **radijanima** i rotiranje se vrši oko trenutno aktivnog koordinatnog početka.

U ovom slučaju **pozitivan ugao rotacije** (+) se poklapa sa smerom kretanja kazaljke na satu, za razliku od većine programskih paketa i programa za crtanje u ravni!

Kao što je pomenuto, kod komande `ctx.rotate(ugao)` argument `ugao` definiše ugao u radijanima. I ovo je opšte poznato. Pretvaranje stepena u radijane se obavlja pomoću formule: $\text{stepeni} \cdot \text{Math.PI} / 180$. Na primer, za rotiranje objekta za 35° primeniće se formula u sledećem obliku $35 \cdot \text{Math.PI} / 180$.

12.10. Kreiranje šablona

HTML5 `<canvas>` omogućava da se kreiraju šabloni unutar grafičke oblasti, što doprinosi boljem izgledu elemenata unutar grafičke oblasti u kojoj se radi.

³⁹ 40 TRANSLATE Pomeranje koordinatnog pocetka CANVAS.html

⁴⁰ 41 SCALE uvecanje iz 00 CANVAS.html

⁴¹ 42 SCALE umanjenje iz 00 CANVAS.html

⁴² 43 SCALE uvecanje iz centra figure CANVAS.html

⁴³ 44 SCALE umanjenje iz centra figure CANVAS.html

⁴⁴ 45 SCALE koncentricne figure CANVAS.html

⁴⁵ 46 SCALE kruznica u X elipsu CANVAS.html

⁴⁶ 47 SCALE kruznica u Y elipsu CANVAS.html

⁴⁷ 48 ROTATE i PLUS ugao iz 00 CANVAS.html

⁴⁸ 49 ROTATE i MINUS ugao iz 00 CANVAS.html

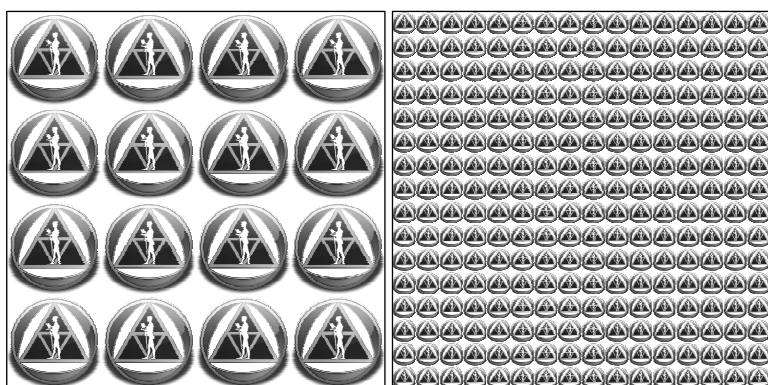
⁴⁹ 50 ROTATE i PLUS ugao i TRANSLATE kontura CANVAS.html

⁵⁰ 51 ROTATE i PLUS ugao i TRANSLATE kruznice CANVAS.html

Sve operacije se obavljaju pomoću komande `createPattern(img, repetition)` koja omogućava kreiranje odgovarajuće preraspodele fotografija, slika ili objekata u obliku pravougaonog šablona (matrica sa određenim brojem redova i sa određenim brojem kolona, prikazan jedan red, prikazana jedna kolona, prikazan jedan jedini element kao A_{11} element unutar zamišljene matrice – nema šablona).

Element `img` definiše fotografiju, sliku, `canvas` objekat ili video element koji će kreirati definisani šablon. Element `repetition` definiše način na koji će se definisati šablon:

- `repeat`⁵¹ – ovo je uobičajena vrednost (*default*); pravi se "puna" matrica.
- `repeat-x`⁵² – izabrani objekat se prostire samo horizontalno, duž X ose.
- `repeat-y`⁵³ – izabrani objekat se prostire samo vertikalno, duž Y ose.
- `no-repeat`⁵⁴ – izabrani objekat će se prikazati jednom i neće biti kreiran šablon.



Slika 12.25. Dva različita šablona^{55 56}

12.11. Kreiranje senki

HTML5 `<canvas>` omogućava da se kreiraju senke oko crteža ili elemenata, što doprinosi boljem izgledu elemenata unutar grafičke oblasti u kojoj se radi. Sve operacije se obavljaju pomoću sledeće četiri komande:

- `shadowColor [ = value ]` – komanda ili atribut prikazuje trenutno setovanu boju senke i omogućava promenu iste; boja može da se definiše na već pomenute načine.
- `shadowBlur [ = value ]` – komanda ili atribut prikazuje tekući nivo zamućenja (*blur*) senke i omogućava promenu istog.

⁵¹ 52 Šablon REPEAT CANVAS.html

⁵² 53 Šablon REPEAT-X CANVAS.html

⁵³ 54 Šablon REPEAT-Y CANVAS.html

⁵⁴ 55 Šablon NO-REPEAT CANVAS.html

⁵⁵ 56 Šablon - PATTERN - 250x250 CANVAS.html

⁵⁶ 57 Šablon - PATTERN - 50x50 CANVAS.html

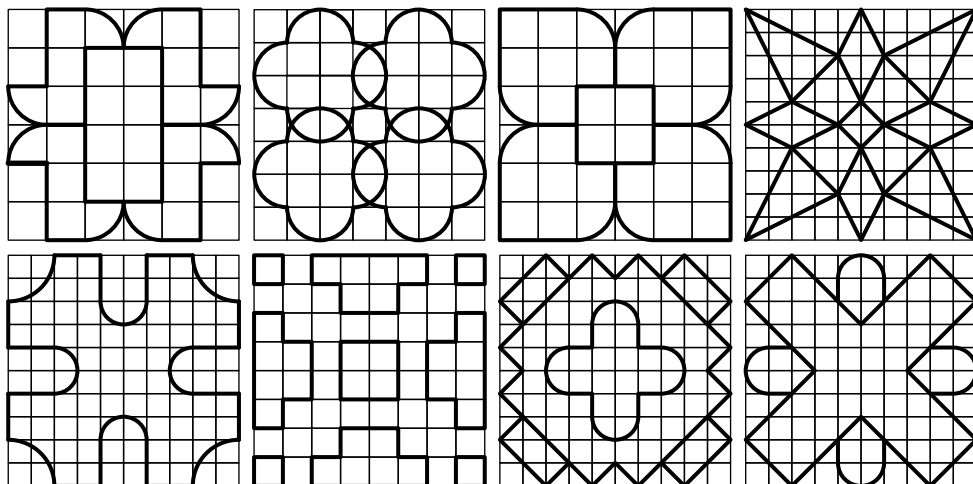
- `shadowOffsetX [ = value ]` – osobina prikazuje tekući pomeraj (*offset*) senke po X osi i omogućava promenu istog; pomeranje senke duž apscise.
- `shadowOffsetY [ = value ]` – osobina prikazuje tekući pomeraj (*offset*) senke po Y osi i omogućava promenu istog; pomeranje senke duž ordinate.



Slika 12.26. Senke bez zamućenja i sa zamućenjem^{57 58 59 60}

12.12. Zadaci za vežbu

Na slici 12.27 prikazani su pojedinačni potencijalni šabloni za kreiranje odgovarajućih šara.



Slika 12.27. Potencijalni šabloni za šare (od Sara-01.html do Sara-08.html)

⁵⁷ 57 SHADOW OffsetX i OffsetY CANVAS.html

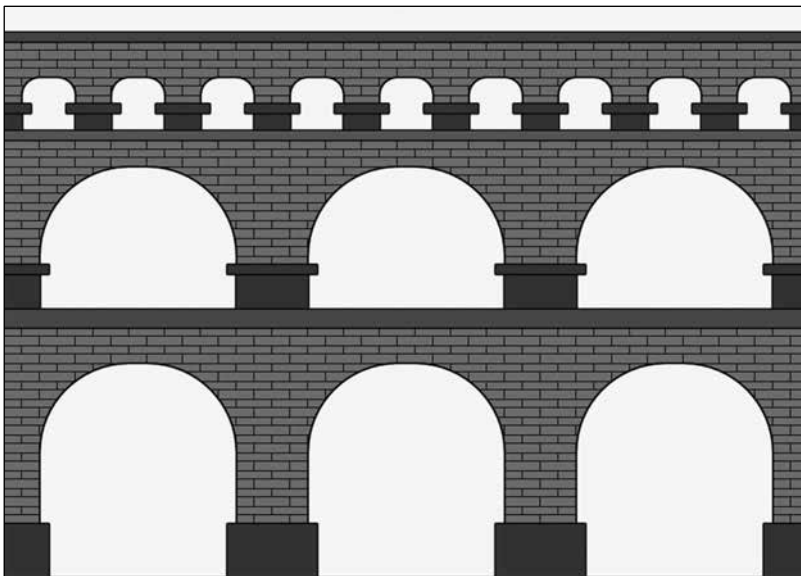
⁵⁸ 58 SHADOW OffsetX i OffsetY TEKST CANVAS.html

⁵⁹ 59 SHADOW OffsetX, OffsetY i Blur TEKST CANVAS.html

⁶⁰ 60 SHADOW OffsetX, OffsetY, Blur PRAVOUGAONIK CANVAS.html



Slika 12.28. *Prikaz ekrana na osnovu dokumenta Velika kuca.html*



Slika 12.29. *Prikaz ekrana na osnovu dokumenta Vijadukt.html*

Glava 13

JavaScript

JavaScript predstavlja skript jezik, koji podržava i funkcionalno programiranje i objektno-orijentisanu paradigmu. Omogućava dodavanje različitih dinamičkih korisničkih interaktivnosti na veb stranicama i predstavlja jezik veba koji veb sajt čini **živim**. Ovim skript jezikom moguće je implementirati mehanizme varijacije formi prilikom popunjavanja različitih formula, prema zadatim kriterijumima. Takvi formulari na veb aplikacijama mogu biti registracija, prijava, ostavljanje podataka u odgovarajućem formularu i slično. Programi koji se pišu ovom tehnologijom nazivaju se **skripte**. One se mogu pisati direktno u HTML kodu, te se automatski mogu izvršavati prilikom učitavanja veb stranice. Takođe, mogu se pisati i u zasebnom (eksternom) fajlu, namenjenim u te svrhe. Generalno govoreći, skripte se izvršavaju kao otvoren tekst (eng. *plain text*). Nije neophodno specifično kompajliranje koda kako bi se određeni deo koda izvršio i prikazao u pretraživaču.

JavaScript se može izvršavati, pored bilo kog pretraživača i na serverima. Takođe, može se izvršiti i na bilo kom uređaju koji u sebi sadrži *JavaScript* podršku (eng. *engine*). Svaki pretraživač u sebi ima ugrađenu podršku (*engine*), te različiti pretraživači nose i drugačije nazive za njih. Pretraživači *Google Chrome* i *Opera* nose kodni naziv V8 za podršku *JavaScript* tehnologiji. U *Firefox* pretraživaču relevantna podrška (*engine*) naziva se *SpiderMonkey*. Takođe, mogu se susresti i drugi nazivi za druge pretraživače; *Internet Explorer* – *Chakra*, *Microsoft Edge* nosi naziv *Nitro* itd. Preporuka je poznavanje navedenih termina, radi lakšeg snalaženja u dokumentaciji zvanične tehnologije i radi lakšeg snalaženja u zajednici programera.

13.1. Istorija *JavaScript* tehnologije

Prvobitan naziv ove tehnologije bilo je *Mocha*, zatim, *LiveScript*, u periodu od 1995. godine od meseca maja do decembra te godine. Zatim je došlo do promena među velikim kompanijama tog perioda, te su *Netscape* i *Sun* postigli sporazum u pogledu licenci. Iz te ideje iznikao je naziv *JavaScript*. U vreme pojavljivanja skript jezika, veliku popularnost među programerima širom sveta imao je programski jezik *Java*, te je osnovna ideja bila da *JavaScript* bude komplementarni jezik. Ubrzo je jezik u potpunosti predat međunarodnoj organizaciji ECMA, a u to vreme kompanija *Netscape* nije dozvoljavala

korišćenje imena *JavaScript*. *JavaScript* kao takav postaje potpuno nezavisan jezik, sa svojim standardom i to pod nazivom ECMAScript (*European Computer Manufacturers Association*). Još uvek *JavaScript* nije slobodan, tj. otvoren naziv, a u današnjici predstavlja zaštićeni žig poznate kompanije *Oracle*, koja je, zapravo, nekadašnja kompanija *Sun*. Zvaničan sajt Međunarodne organizacije za ECMAScript je dostupan je putem linka: <https://www.ecma-international.org/>; na istom se mogu videti različita pravila, odluke, standardi i drugo.

Danas se često može susreti terminološka zabuna u zajednici programera, gde se smatra da ECMAScript, *JavaScript* i *JScript* predstavljaju tri različita standarda, odnosno, tri različite tehnologije. Stoga je važno napomenuti da je reč istoj tehnologiji i da je jedan jedinstveni standard u potpunosti isti za sve, a on glasi ECMAScript.

13.2. Podrška *JavaScript* tehnologiji u pretraživačima

Principi i osnove su jednostavne, ali *JavaScript engine* predstavlja izuzetno složenu arhitekturu. Zapravo, *JavaScript engine* je ugrađen u pretraživač i može da čita, tj. da izvršava parsiranje skripte. Zatim, *JavaScript engine* skriptu kompajlira u mašinski jezik, gde se mašinski kôd izvršava i obrađuje brzo i efikasno. U okviru tih procesa, primenjuju se algoritmi optimizacije kako bi se odgovarajuće funkcije izvršavale optimalnije.

13.3. *JavaScript* u pretraživačima

Moderan *JavaScript* je programski jezik koji se može opisati kao izuzetno siguran i pouzdan. Nastao je kao jezik koji ne zahteva pristup memoriji ili samom procesoru na fizičkom nivou. *JavaScript* ima sposobnost da manipuliše veb stranicama putem različitih dinamičkih interakcija sa korisnicima, kao i sa povezanim servisima na serverskoj strani.

Naime, *JavaScript* može dinamički ugraditi novi HTML u odgovarajući deo veb stranice, zatim, može vršiti odgovarajuće izmene nakon relevantnih upita ili zahteva. Samim tim, može uticati na vizuelni izgled same stranice, te joj može menjati stilove, boje, fontove i drugo. Takođe, može reagovati na korisničke akcije, pokretati odgovarajuće događaje prilikom klikova mišem, pritiskom na tastere tastature i slično.

JavaScript može slati različite zahteve putem računarske mreže ka udaljenim veb serverima. Može preuzimati ili otpremati različita dokumenta (video datoteke, slike, različite tekstualne sadržaje i slično). Prilikom autentikacije korisnika, može prihvatati i postavljati kolačiće (eng. *cookie*), te korisniku može prosledivati relevantne odgovore od strane servera. Dodatno, može čuvati podatke na klijentskoj strani, u lokalnoj memoriji (eng. *local storage*) pretraživača.

JavaScript na veb stranicama ima i svoje ograničene sposobnosti radi bezbednosti korisnika, a sve to u cilju smanjenja zlonamernih aktivnosti koje treće lice može izvršiti nad posetiocima veb sajta. Stoga, *JavaScript* ne sme imati mogućnost čitanja ili pisanja nesusumnjivih datoteka na hard disku korisnika, niti kopirati takav sadržaj na drugu lokaciju. Ta-

kođe, *JavaScript* ne sme pokretati niti izvršavati nijedan program na nivou operativnog sistema. Sve ove akcije odnose se isključivo na to da ovakve aktivnosti nisu dozvoljene osim ukoliko krajnji korisnik to nije eksplicitno dozvolio.

Takođe, *JavaScript* jezik može pristupati i ostvarivati interakciju sa korisničkim mikrofonom, kamerom i drugim eksternim uređajima, te je saglasnost korisnika računara u tom slučaju obavezna. Korisnik u pretraživaču može imati više otvorenih kartica (tabova); izvršni kôd koji se pokreće u okviru jednog veb sajta (ili veb aplikacije) smeo bi da ima uvid samo u svoju karticu. Preciznije, ne bi smeo da ostvaruje pristup drugim otvorenim prozorima i da prikuplja informacije (čak i ukoliko veb sajтови pripadaju istom domenu).

Naime, ova politika se naziva *Same Origin Policy*, odnosno, **politika istog porekla**. Ukoliko je, pak, neophodno omogućiti međusobni pristup između otvorenih tabova, kako bi se problem prevazišao, relevantne stranice moraju sadržati u sebi specifičan *JavaScript* kôd koji može dozvoljavati takvu interakciju, a koji za cilj ima razmenu podataka sa istog domena.

13.4. Razvojna okruženja

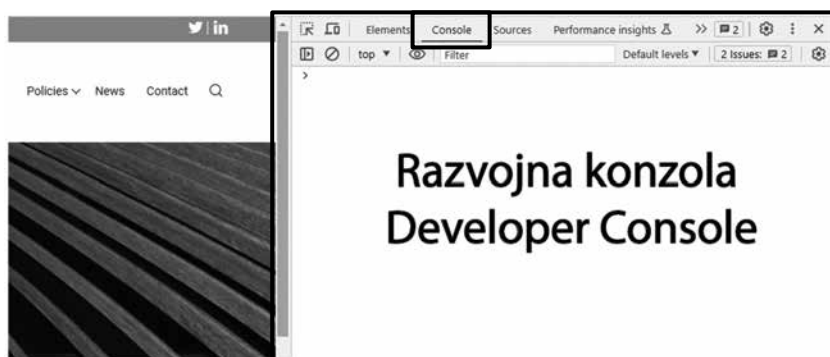
Programeri prilikom pisanja *JavaScript* koda mogu koristiti različita razvojna softverska okruženja. To mogu biti jednostavni uređivači koda (eng. *code editor*), a mogu biti i složenija razvojna okruženja, tj. IDE (eng. *Integrated Development Environment*).

Suštinski IDE predstavlja snažan i moćan alat sa ugrađenim složenim funkcionalnostima, koje, zapravo, mogu raditi sa celim projektom; time se smatra da može raditi sa svakim fajlom koji se nalazi u projektu. Samim tim, IDE predstavlja, ne samo običan uređivač koda, već sveobuhvatno razvojno okruženje. Poznata IDE rešenja jesu *Visual Studio*, *NetBeans*, *WebStorm* i drugi, a koji su ujedno i predstavnici premijum rešenja.

Takođe, postoje i jednostavnija razvojna okruženja (eng. *lightweight code editor*), koja su izuzetno brza i jednostavna za rukovanje i uz dodatne alate (ekstenzije) predstavljaju takođe izuzetno moćan alat za rad sa kodom. Popularna razvojna okruženja jesu besplatna rešenja, kao što su *Visual Studio Code*, *Atom*, *Notepad++* i drugi.

13.5. Razvojna konzola veb pretraživača

Jedna od ključnih komponenti prilikom rada sa *JavaScript* kodom predstavlja rad sa ugrađenom razvojnom konzolom (eng. *Developer Console*). Kako bi se otvorila razvojna konzola, u velikoj većini pretraživača može se u bilo kom otvorenom prozoru veb sajta pritisnuti kombinacija tastera Ctrl + Shift + J, u operativnom sistemu *Windows*. Za *Mac* naredba je Cmd + Option + J. Razvojna konzola omogućava prikazivanje različitih poruka, upozorenja ili grešaka nastale usled izvršavanja *JavaScript* koda. Takođe, olakšava mehanizam ispravke koda (eng. *debug*), zatim, olakšava proces inspekcije koda, unesenih vrednost i drugo. Za sveobuhvatne performanse koda, za programere razvojna konzola predstavlja moćan alat koji se koristi kontinualno, i preporuka je da se ovakvi alati koriste kad god je to moguće.



Slika 13.1. Primer razvojne konzole u okviru pretraživača Google Chrome

13.6. Osnovni jezici na vebu

HTML (eng. *Hyper Text Markup Language*) predstavlja opisni jezik sa međusobno povezanim hipertekstualnim linkovima. Važno je napomenuti da HTML ne predstavlja programski jezik. Koristi se za pravljenje i struktuiranje veb dokumenata, tačnije, veb stranica i veb aplikacija. Ovaj jezik se koristi u svrhe označavanja, tj. opisivanja stranice koja inicijalno predstavlja strukturu sadržaja dokumenta. Jednostavnije rečeno, HTML predstavlja tekstualni dokument na vebu.

CSS (eng. *Cascading Style Sheets*) predstavlja jezik za stilizovanje veb stranica, odnosno, HTML dokumenata. On je zadužen za izgled veb stranice u kontekstu boja, rasporeda elemenata, dizajna stranica, formatiranja i drugo.

JavaScript jezik predstavlja skript jezik za veb stranice koje omogućava različite korisničke interaktivnosti, dinamička ažuriranja sadržaja veb dokumenata i drugo. Ovaj jezik predstavlja logički deo veb sajta koji veb stranici dodeljuje dinamičko ponašanje reagovanjem na odgovarajuće događaje, kao što su klikovi tastera miša, korisnički unos sa tastature i drugo. Predstavlja široko rasprostranjen jezik koji se danas koristi, gotovo, u svim tehnologijama. U osnovi, predstavlja interpreterski (prevedeni) jezik, sa mogućnostima objektno-orijentisanog programiranja. Ono što je u prethodnoj deceniji bilo nezamislivo danas se uz pomoć JavaScript jezika na serverskoj strani, a uz pomoć NodeJS jezika, koristi za razvoj API-ja (eng. *Application Programming Interface*).



Slika 13.2. Trendovi korišćenja JavaScript jezika; globalno (levo) i Srbija (desno)

Zanimljivo je komparativno poređenje u sferi interesovanja za *JavaScript* jezik koje je globalno, u poređenju sa Republikom Srbijom, gde se može videti da indeks pretrage u Republici Srbiji u zadatom opsegu ne zaostaje za globalnim trendovima. Izvor pretrage je preuzet sa *Google Trends* (trends.google.com).

13.7. Upotreba *JavaScript* jezika i izlaz koda

Dugo vremena *JavaScript* jezik je služio kao jezik za razvijanje klijentske strane. Zahvaljujući velikoj podršci zajednice programera, *JavaScript* postaje izuzetno moćan i višenamenski jezik, koji se koristi za razvoj poslovnih veb aplikacija, mobilnih aplikacija, tako i za veb aplikacije koje imaju međusobno povezivanje u realnom vremenu, zatim, za razvoj mnogobrojnih video igara itd.

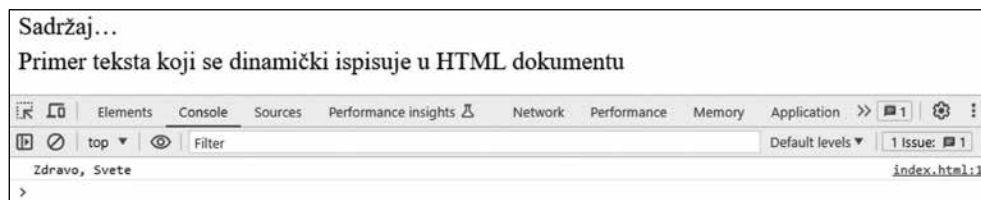
JavaScript se može ugrađivati u HTML dokument, te se tagovi `<script> ... </script>` koriste za umetanje *JavaScript* koda u dokument. Naime, preporuka je da se skript kôd ugrađuje na dnu tela stranice, neposredno pre zatvaranja tela stranice, odnosno, neposredno pre zatvaranja `</body>` taga, kako bi se usled potencijalnih grešaka u *JavaScript* kodu omogućilo prikazivanje veb stranice. *JavaScript* kôd se čita odozgo prema dole, te se tako i izvršava. Svaka greška *JavaScript* koda reflektuje se kao blokada u kontekstu renderovanja veb stranice, sve dok kôd u potpunosti nije ispravan. Svakako, postoje izuzeci i slučajevi kada se može odstupati od ovog pravila, a u zavisnosti od potreba veb aplikacije.

Naredni kôd ilustruje korišćenje `<script> ... </script>` tagova u HTML-u, gde se rezultat koda prikazuje u razvojnoj konzoli pretraživača.

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Uvod u JavaScript</title>
</head>
<body>
  <p>Sadržaj...</p>

  <!-- Interni JavaScript - body section -->
  <script>
    console.log('Zdravo, Svete');
    document.write('Primer teksta koji se dinamički ispisuje u HTML dokumentu');
  </script>
</body>
```

Rezultat gore napisanog koda se ogleda u pretraživaču kao:



Slika 13.3. Prikazivanje izvršenog skript koda u pretraživaču i razvojnoj konzoli

13.8. Izlaz

JavaScript kôd kao takav nema sopstvene funkcije u sebi za ispisivanje podataka. Kako bi se prikazao odgovarajući kôd u pretraživaču ili razvojnoj konzoli, mora se pozvati odgovarajuća funkcija za prikazivanje. Od krucijalne važnosti poznavanje ugrađenih funkcionalnosti, odnosno, metoda za prikazivanje podataka na ekranu. Postoje različiti načini za prikazivanje koda u pretraživaču, a oni mogu biti napisani na različite načine.

Naime, mogu se koristiti **iskaćući prozori u vidu dijaloga** (eng. *popup window*); metod koji se koristi jeste `alert()`, a može biti i `window.alert()`. Prikazivanje vrednosti u razvojnoj konzoli piše se pozivom metode `console.log()`. Pisanje u telu HTML dokumenta vrši se pomoću `document.write()`.

```
alert('Zdravo, svete!');  
window.alert('Učimo JavaScript');  
console.log('Učimo JavaScript');
```

13.9. Sintaksa

Sintaksa *JavaScript* jezika je, suštinski gledano, skup precizno definisanih pravila koja određuju i definišu strukturu *JavaScript* programa, te je od fundamentalne važnosti razumeti osnove i principe *JavaScript* sintakse.

U *JavaScript* jeziku jedna linija koda predstavlja *JavaScript* naredbu ili instrukciju. Naredba daje smernice pretraživaču kako izvršiti odgovarajuću akciju i na koji način prikazati rezultat naredbe. Takođe, naredba može biti deo koda, sa odgovarajućim funkcionalnostima i akcijama. Stoga se može izvesti zaključak da *JavaScript* kôd predstavlja niz naredbi pretraživaču koje je potrebno izvršiti.

13.10. Instrukcije

Naredbe, odnosno, instrukcije pišu se u HTML-u, u okviru skript tagova, tj. `<script> ... </script>`. Svaka naredba se završava tačkom-zapetom, odnosno, `;`. Ujedno, sve naredbe jezika *JavaScript* sastavljene su od vrednosti, *JavaScript* operatora, odgovarajućih ključnih reči i komentara u samom kodu. *JavaScript*, kao jezik, ignorise beline u vidu razmaka ili dodatne prelome koda u vidu praznih redova. Prazan red ili eventualni razmak između karaktera u *JavaScript* sintaksi dodaje se kako bi programer jednostavnije čitao napisani kôd. *JavaScript* brine o karakteristikama, te veoma razlikuje velika i mala slova. Takođe, *JavaScript* jezik jeste interpreterski jezik, pa se tako izvodi zaključak da on ne predstavlja kompajlerski jezik.

13.11. Komentari

JavaScript komentari igraju vrlo važnu ulogu programerima prilikom pisanja koda. Komentari predstavljaju opis koda. Koriste se kako bi se programer prisetio odgovarajućih

funkcionalnosti koje je implementirao u *JavaScript* program. Takođe, izuzetno su bitni i značajni za razvojni tim, omogućavajući ostalim članovima tima bržu i lakšu razumevanje suštine tog dela koda. Takođe, prilikom razvoja *JavaScript* programa, komentari mogu biti korisni jer omogućavaju pisanje relevantnih sugestija ili upozorenja, zasnovanih na prethodnim iskustvima, kako bi se olakšalo razumevanje koda.

JavaScript program prilikom svog izvršavanja u pretraživaču ignoriše komentare i preskače ih. Takođe, pretraživači ignorišu komentare u kodu i ne prikazuju ih krajnjem korisniku. Komentari se mogu pisati bilo gde u kodu, ali dobra praksa je napraviti uniformno pravilo za ceo kôd, kako bi i komentari bili razumni onoj osobi koja ih čita.

U *JavaScript* jeziku postoje dve varijante komentare; mogu biti komentari napisani u jednom redu (jednolinijski) ili se mogu pisati u više redova, tj. linija (višelinijski).

Sintaksa komentara jednolinijskog komentara predstavlja se sa dve kose crte, tj. `"/"`. Jednolinijski komentar ne zahteva bilo kakvo zatvaranje komentara nakon što se isti napiše. U nastavku se može videti primer:

```
//Primer je primer jednolinijskog komentara
```

Više linijski komentari predstavljaju složeniji opis blok koda. Zapravo, ukoliko programer želi u više rečenica da opiše odgovarajući kôd, u tom slučaju se koristi višelinijski komentar. Za razliku od jednolinijskih komentara, oni zahtevaju zatvaranje komentara. Višelinijski komentari počinju kosom crtom i zvezdicom, tj. `"/*"`, a zatvaraju se obrnutim redosledom, odnosno, zvezdicom i kosom crtom – `"*/"`. Sledi primer višelinijskog komentara.

```
/*  
  Primer višelinijskog komentara.  
  Koristi se za opisivanje koda u više linija.  
  Ne bi trebalo koristiti sintaksu višelinijskog komentara za jednolinijski komentar.  
*/
```

13.12. Operatori i aritmetika

U *JavaScript* jeziku postoje aritmetičke operacije koje se mogu koristiti za izvođenje različitih matematičkih operacija nad operandima. Naime, postoje standardne aritmetičke operacije poput sabiranja, oduzimanja, množenja, deljenja, te postoje zagrade koje se mogu koristiti – obične `(...)` ili uglaste `[...]`. Operatori predstavljaju odgovarajuće ključne reči koje govore kako da *JavaScript* izvrši relevantnu operaciju. Takođe, u kodu programa *JavaScript* operatori se mogu kombinovati i koristiti zajedno sa prethodno definisanim promenljivama radi izgradnje izraza koji omogućavaju manipulaciju logikom koda ili izvršavanje određenih algoritama.

Struktura operatora i operanada se može videti u narednom primeru:

```
var suma = 1 + 1;
```

Iz prethodnog zapisa može se izvesti zaključak:

- Znak jednakosti i znak sabiranja predstavljaju operatore,
- Znak jednakosti predstavlja operator dodele,
- Znak plus predstavlja aritmetički operator.

JavaScript ima podršku za operatore:

- Operatori dodele,
- Logički operatori,
- Relacioni operatori (koriste se za poređenja),
- Ternarni (uslovni) operatori,
- Operatori za rad sa tekstualnim zapisom, tj. stringovima,
- Bitski operatori.

U *JavaScript* jeziku koriste se i aritmetički operatori; koriste se za izvršavanje svih aritmetičkih operacija. To se odnosi na standardne matematičke operacije koje se izvode nad brojevima. U nastavku se mogu videti aritmetički operatori koji su podržani u *JavaScript* jeziku:

- Sabiranje; primer: $1 + 1 = 2$,
- Oduzimanje; primer: $2 - 1 = 1$,
- Množenje; $10 * 10 = 100$,
- Deljenje; primer: $10 / 2 = 5$,
- Ostatak pri deljenju (*Modulus*); primer $20 \% 10 = 0$,
- Inkrement broja; primer: uvećavanje broja: piše se spojeno u vidu dva znaka za plus, tj. ++
- Dekrement broja; primer: umanjenje broja: piše se spojeno u vidu dva znaka za minus, tj. --
- Stepovanje (eksponent); primer: piše se sintaksom spojeno u vidu dva znaka množenja, tj. $3 ** 3 = 9$

Za dodeljivanje vrednosti, koriste se operatori dodele. Naime, u *JavaScript* jeziku podržani su naredni operatori dodele:

- Dodela vrednosti: =
- Sabiranje i dodeljivanje: +=
- Oduzimanje i dodeljivanje: -=

- Množenje i dodeljivanje: `*`
- Deljenje i dodeljivanje: `/`
- Ostatak pri deljenju i dodeljivanje: `%`

U nastavku se mogu videti *JavaScript* aritmetičke operacije:

```
// Operatori dodeljivanja
var broj1 = 1;
var broj2 = 1

// Saberi i dodeli
broj1 += broj2;
console.log(broj1); // Rezultat: 2

// broj2 -= broj1;
// console.log(broj2); // 0

// broj1 *= broj2;
// console.log(broj1); // 1

// broj2 /= broj1;
// console.log(broj1); // Rezultat: 1
```

Podržani su i logički operatori u *JavaScript* jeziku. Najčešće se koriste za kombinovanje više uslovnih iskaza. U nastavku sledi prikaz raspoloživih i podržanih logičkih operatora u *JavaScript* jeziku:

- `&&` – predstavlja logičko I (AND); primer: *true* ukoliko su oba operanda/uslova tačna,
- `||` – predstavlja logičko ILI (OR); primer: *true* ukoliko je jedan ili drugi operand/uslov tačan,
- `!` – predstavlja logičko NE (NOT); primer: *true* ukoliko operand nije tačan.

```
// && (Logičko AND) - vraća true ukoliko su oba operanda true
console.log('true && true: ', true && true);
console.log('true && false: ', true && false);
console.log('false && true: ', false && true);

// -----

// || (Logičko OR) - vraća true if ukoliko je jedan operand true
console.log('true || true: ', true || true);
console.log('true || false: ', true || false);
console.log('false || true: ', false || true);

// -----

// ! (Logičko NOT) Vraća true ukoliko operand nije true
var jePolozio = true;
var nijePolozio = !jePolozio;
console.log('nijePolozio: ', nijePolozio);
```

Relacioni operatori, odnosno, operatori poređenja mogu porediti dve vrednosti na logički način. Mogu odrediti sličnost i razliku između promenljivih. U nastavku se mogu videti dostupni operatori poređenja:

- Operator eksplicitne (stroge) jednakosti; tri znaka jednakosti, tj. ===
- Nije jednako (različito od); piše se kao uzvičnik i dva znaka jednakosti: !==
- Manje od; piše se znakom manje: <
- Veće od; piše se znakom za veće: >
- Manje od ili jednako; piše se kao: <=
- Veće od ili jednako; piše se kao: >=

U nastavku slede primeri korišćenja relacionih operatora:

```
// Operatori poređenja
var broj1 = 5;
var broj2 = 10;
var broj3 = "5";

alert(broj1 == broj3); // true
alert(broj1 === broj3); // false
alert(broj1 != broj2); // true
alert(broj1 !== broj3); // true
alert(broj1 < broj2); // true
alert(broj1 > broj2); // false
alert(broj1 <= broj2); // true
alert(broj1 >= broj2); // false
```

Uslovni operator može vratiti vrednost na osnovu uslova. Ternarni operator predstavlja jedini operator u *JavaScript* jeziku koji koristi tri operanda. Dodeljuje vrednost promenljivoj isključivo na osnovu uslova. Operator se može koristiti kao skraćeni oblik naredbe *if*. Operator se predstavlja znakom pitanja, tj. "?". U nastavku se može videti primer gde se poredi da li korisnik ima manje od definisanog broja godina, a potom se kao rezultat vraća odgovor kojoj grupi korisnika pripada. Pripadajuća grupa je utvrđena ternanim operatorom.

```
var trenutnoGodina = 45;
var kategorija;

kategorija = (trenutnoGodina < 20) ? 'Manje od 20' : 'Više od 20';
alert('Korisnik pripada grupi: ' + kategorija);
```

String operator može dobiti ili imati vrednost u string (tekst) formatu. Naime, aritmetički operator plus (+) se može koristiti za spajanje, tj. concatenaciju stringova (isto važi i za brojeve). U nastavku se mogu videti dva tipa operatora koja se koriste za stringove:

- Concatenacija; predstavlja se spajanjem, korišćenjem aritmetičkog operatora plus (+): stringX + stringY (podrazumevano je da promenljive stringX i stringY imaju prethodno dodeljen string),

- Dodeljivanje konkatencijom; predstavlja se kao `stringX += stringY`, a ogleda se u tome da dodaje `stringY` na `stringX`. Isto kao i u prethodnom primeru, preduslov je da su i `stringX` i `stringY` prethodno deklarirani.

Primeri string operatora mogu se videti u nastavku:

```
var poruka1 = "Zdravo, ";  
var poruka2 = " svete!";  
  
alert(poruka1 + poruka2); // Ispisuje: Zdravo, svete!  
  
poruka1 += poruka2;  
alert(poruka1); // Ispisuje: Zdravo, svete!
```

13.13. Dodeljivanje vrednosti

Promenljive predstavljaju jedan od najvažnijih segmenata *JavaScript* jezika, kao i svih drugih programskih jezika. Razumevanje principa dodeljivanja vrednosti je od ključnog značaja za razumevanje daljih koraka programskih jezika uopšte. Kontejneri (eng. *container*) predstavlja mehanizam za skladištenje podataka. Promenljiva je, zapravo, kontejner i samim tim predstavlja mehanizam za čuvanje odgovarajuće (dodeljene ili dobijene) vrednosti. Vrednost može biti dodeljena u vidu broja, karaktera (stringa), logičke vrednosti, niza podataka itd. Takođe, promenljivoj se može dodeliti prazan string ili se samo definisati promenljiva bez bilo kakve dodeljene vrednosti. Ukoliko se ne dodeli vrednost, promenljiva kasnije u kodu može dobiti svoju vrednost, nakon izvršenih funkcija ili seta naredbi koje se izvršavaju.

Promenljive suštinski čuvaju privremeno podatak u memoriji, a sâm naziv promenljive u *JavaScript* jeziku (a i ostalim programskim jezicima) predstavlja simboličko ime za čuvanje vrednosti. Drugim rečima, može se reći da naziv promenljive predstavlja ime lokacije gde se skladišti vrednost. Svim vrednostima koje su definisane u promenljivama može se pristupiti na odgovarajući način, koristeći predefinisane ili ručno napisane funkcije. Takođe, vrednosti promenljivih se mogu ažurirati.

Kako bi se promenljivoj dodelila vrednost, potrebno je izvršiti deklaraciju. Termin deklaracija odnosi se na odgovarajuće dodeljivanje tipa promenljive ili njene vrednosti. Globalno (javno) deklarisanje promenljive definiše se ključnom rečju `var`. Operator dodele vrednosti predstavlja znak jednakosti, tj. `"="`.

Obrazac za deklaraciju vrednosti glasi `var ime_promenljive = vrednostPromenljive`. Naredni primer ilustruje korišćenje globalne promenljive pod nazivom "poruka", sa dodeljenom vrednošću u vidu teksta "Zdravo, svete!".

```
var poruka = 'Zdravo, svete!'
```

JavaScript kao programski jezik ima svoje podrazumevane vrednosti i svakoj promenljivoj dodeljuje vrednost *undefined*, sve dok ista ne dobije vrednost. U nastavku se mogu videti

tri različita primera dodeljivanja vrednosti i to kao string, zatim, ocena kao numerička vrednost i na kraju – logička vrednost, kao tačno, tj. *true*.

```
var nazivPredmeta = 'Veb dizajn i multimedijalni sistemi';  
var ocena = 10;  
var jeUnesena = true;
```

Prema ES6 standardu (ECMAScript 6), uvode se dve nove ključne reči. One su *let* i *const* za deklaraciju promenljivih. Razlikuju se u odnosu na *var* samo u opsegu. Naime, u starijim verzijama programa *JavaScript* koda korišćena je deklaracija globalne deklaracije, tj. *var*, te se promenljivoj može pristupiti iz bilo kog dela koda i može joj se dodeliti, odnosno, promeniti vrednost. Usled mnogobrojnih zlonamernih aktivnosti to može predstavljati veoma veliki problem. Ključne reči *let* i *const* deklariraju promenljive vrednosti sa datim opsegom blok koda. Opseg bloka predstavlja se vitičastim zagradama (otvorenim i zatvorenim), tj. "{...}". Time je definisan opseg u kome važi deklaracija promenljivih. Ključna reč *let* predstavlja lokalnu promenljivu koja se koristi unutar opsega, dok *const* predstavlja konstantu, odnosno, promenljivu čija se vrednost ne menja tokom izvršavanja koda. Konstantna vrednost predstavlja deklarisanu promenljivu koja služi samo za čitanje.

```
let tehnologija = 'ECMAScript';  
const PI = 3.14;
```

13.14. Tipovi podataka

Promenljiva može imati bilo koji tip podataka. Tipovi podataka predstavljaju i određuju kakva je suštinski vrsta podatka koji se smešta i čuva u promenljivoj. Naime, *JavaScript* ima dve glavne podele na tipove podataka, a to su primitivni i složeni tipovi podataka.

Primitivni tipovi podataka predstavljaju: stringovi, brojevi, logične vrednosti, nedefinisane vrednosti, prazno polje, simbol. Složeni (ne primitivni) tipovi podataka mogu biti: nizovi, objekti i funkcije.

Primitivni tipovi podataka mogu sadržati samo jednu vrednost u datom trenutku. Moguće ih je porediti međusobno prema vrednosti. Dve vrednosti su jednake samo i samo ako imaju istu vrednost. Postoje različiti primitivni tipovi podataka. a oni su:

- *String* – predstavlja tekstualni zapis, odnosno, tekstualni podatak; uobičajeni primer je sledeći: 'Zdravo, svete!',
- *Number* – predstavlja numeričku vrednost; primer: 1, 2, 100, 1000,
- *Boolean* – predstavlja logičku vrednost (tačno ili netačno); primer: *true* ili *false*,
- *Undefined* – predstavlja nedefinisanu vrednost; promenljiva postoji, ali nije prihvatila ili dobila vrednost,
- *Null* – predstavlja *null* vrednost, odnosno, ne postoji vrednost uopšte.

U nastavku se mogu videti primeri za sve primitivne vrednosti, respektivno:

```
var verzija = 'ECMAScript 6';
var godina = 2023;
var uradjeno = true;
var ime;
var prezime = null;
```

Složeni tipovi imaju složenije entitete i mogu sadržati značajno kompleksniju strukturu koda u sebi, u poređenju sa primitivnim tipovima. Složeni tipovi se porede prema referencama, a ne prema vrednostima, kao što to predstavlja slučaj kôd primitivnih tipova. Podaci se ne skladište u privremenoj memoriji, već se čuvaju u hip (eng. *heap*) memoriji. Hip memorija služi za čuvanje složenih tipova, odnosno, objekata i funkcija. Stoga, složeni tipovi se dinamički alociraju kao referenca u hip memoriji. Složeni tipovi jesu pokazivači na određenu lokaciju u memoriji, a kojoj se može pristupiti putem odgovarajuće reference. Složeni tipovi mogu biti:

- **Array** – predstavlja niz (listu) vrednosti; primer: ['Zdravo, svete!', 1, true],
- **Object** – predstavlja instancu putem koje se može pristupiti složenoj strukturi, tj. svojstvu i vrednosti objekta; piše se u JSON formatu,
- **Function** – predstavlja funkciju, tj. set naredbi koje je potrebno izvršiti.

13.15. Funkcije

JavaScript funkcije predstavlja grupu manje ili više kompleksnih naredbi koje mogu obavljati jednostavne ili složene programske operacije. Funkcije u sebi sadrže ponovno iskoristiv kôd, te se isti kôd više puta može koristiti (i/ili pozivati) za više različitih operacija. Takođe, na taj način se smanjuje količina koda, te nije potrebno svaki put definisati nove naredbe koda i pisati novu (istu) logiku. Takav skup složenih operacija programeru značajano olakšava posao prilikom raspodele koda na manje celine, u vidu manjih upravljivih funkcija, tj. logičkih celina. Preporučeni način je podesan, jer se njime postiže jednostavnije održavanje funkcija, a samim tim se promene u kodu izvršavaju samo na jednom mestu. Otklanjanje grešaka samim tim postaje značajno jednostavnije. Funkcije, kao takve, olakšavaju pisanje modularnog koda. U svakom programskom jeziku, funkcije predstavlja jednu od najvažnijih struktura za kontrolu i izvršavanje programa. Prilikom rada sa funkcijama, za njihovo ispravno korišćenje najpre se funkcija mora pravilno definisati, pomoću ključne reči `function`, a zatim se pravilno pozvati.

Funkcije se, pre svega, definiše, a potom se poziva. Redosled definisanja funkcije može se predstaviti u nekoliko koraka:

1. Deklaracija funkcije (koristeći ključnu reč `function`).
2. Definisanje jedinstvenog imena funkcije.

3. Definisanje parametara funkcije u zagradama, odnosno, (...). Takođe, funkcija može biti i bez parametara; u tom slučaju zagrade ostaju prazne.
4. Postavljanje blok naredbi, uokvirenih vitičastim zagradama, odnosno, {...}.

Struktura funkcije prikazuje se kao:

```
function imeFunkcije() {
    ... telo funkcije i naredbe ...
}
```

Prema prethodnoj strukturi funkcije, primer funkcije sa porukama koje se prikazuju u razvojnom prozoru (konzoli) pretraživača i u iskaćućem prozoru (*popup* dijalogu), može se videti u narednom primeru:

```
function prikaziPoruku() {
    console.log('Poruka iz funkcije ispisana u razvojnoj konzoli.');
```

```
    alert('Poruka iz funkcije u iskaćućem (popup) dijalogu.');
```

```
}
```

Funkcija se može pozivati, tj. pokretati iz bilo kog mesta u kodu. Proces pozivanja funkcije je jednostavan; dovoljno je pozvati ime (naziv) funkcije sa zagradama, prema sledećem šablonu: `imeFunkcije()`. Primer takvog poziva može se videti u nastavku gde se poziva imaginarna funkcija za prikazivanje poruke:

```
prikaziPoruku();
```

Postoje različiti tipovi funkcije, kao što su regularne funkcije, funkcije sa parametrima ili povratne funkcije (funkcije koje vraćaju vrednost).

Regularna funkcija predstavlja klasičnu funkciju za obavljanje uobičajenih akcija ili zadataka.

```
var ime = 'Ana';

// 1. Deklaracija funkcije
function pozdraviOsobu () {
    // Telo funkcije
    // Kod koji se izvršava
    console.log('Zdravo, ' + ime);
    alert('Zdravo, ' + ime);
}

// 2. Poziv funkcije
pozdraviOsobu ();
```

Funkcija sa parametrima predstavlja mehanizam gde se određeni podaci mogu proslediti proslediti, koristeći parametre, tj. argumente. Parametri funkcije se navode u običnim zagradama, tj. (...). Takođe, parametri se mogu navoditi prilikom deklaracije funkcije, kako bi se prihvatale ulazne vrednosti prilikom izvršavanja, tj. pokretanja funkcije. Naredni

primer ilustruje prikazivanje funkcije sa parametrima, gde se računa suma. Izvan bloka funkcije funkcija `izracunajSumu()` poziva se dva puta, gde se unutar parametra dodeljuju argumenti. U prvom pozivu to su 5 i 15, a u drugom pozivu to su numeričke vrednosti 150 i 300.

```
var ukupno;

function izracunajSumu (vrednost1, vrednost2) {
    ukupno = vrednost1 + vrednost2;
    console.log(ukupno);
}

izracunajSumu (5, 15);
izracunajSumu (150, 300);
```

Funkcija sa povratnim tipovima može vratiti vrednost skripti koja je zahtevala njen poziv. U te svrhe koristi se naredba `return`. Vraćena vrednost može se koristiti u *JavaScript* programu. Naredba `return` se obično postavlja na kraju funkcije kao poslednja linija, unutar opsega.

```
//1. Deklaracija funkcije
function izracunajSumu (vrednost1, vrednost2) {
    var ukupno = vrednost1 + vrednost2;
    return(ukupno);
}

//2. Poziv funkcije
console.log(izracunajSumu (5, 15));
console.log(izracunajSumu (150, 300));

var ukupno = izracunajSumu (10,10);
alert(ukupno);
```

13.16. Objekti

U realnom okruženju, kao i u stvarnom životu, skoro sve se može predstaviti objektom. Na primer, automobil, avion, brod, računar, monitor, mobilni telefon i drugo. Svaki od navedenih objekata ima svoje sastavne delove. Sastavni delovi imaju svojstva. Svako svojstvo ima odgovarajuću vrstu vrednosti (često i više njih!). Isti princip se može preslikati i na programske jezike. *JavaScript* jezik je zasnovan na objektima. Objekat predstavlja jedan kompleksan tip podatka, a podesan je za skladištenje kolekcije podataka. On predstavlja svojevrsni kontejner u kom se mogu grupisati različiti podaci, gde se nalaze svojstva i njihove vrednosti. Objekat se definiše vitičastim zagradama, tj. `{...}`; svojstvo objekta se piše unutar vitičastih zagrada prema šablonu `imeSvojstva: vrednost`.

Ime svojstva se uvek predstavlja string zapisom. Tehnički gledano, vrednost koju svojstvo sadrži može biti bilo koji tip podatka (stringovi, brojevi, logičke operacije, nizovi, funkcije i drugo). *JavaScript* objektu se može pristupiti putem dve metode:

1. Notacijom tačke, tj referenciranjem; `objekat.nazivSvojstva`
2. Notacijom uglastih zagrada; `objekat['nazivSvojstva']`

Takođe, objekat se može napraviti objektnim literalnim zapisom, korišćenjem ključne reči `new` za pravljenje objekta i korišćenjem konstruktora objekta (takođe, ključnom reči `new`).

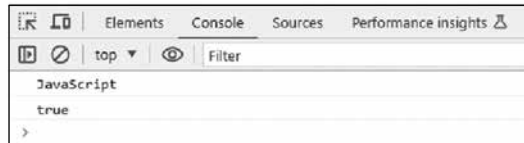
Najjednostavniji način za pravljenje objekta u *JavaScript* jeziku jeste definisanje svojstva i vrednosti, razdvojenih zapeatom. Obrazac za pravljenje objekta može se predstaviti kao:

```
let prviObjekat = {
    svojstvo1: vrednost1,
    svojstvo2: vrednost2,
    ...
};
```

Na osnovu obrazaca i šablona za pravljenje objekta, primer objekta sa svojim svojstvima, kao i prikazivanjem u konzoli pretraživača može se videti u narednom primeru:

```
let skriptJezik = {
    'ime': 'JavaScript',
    'verzija': 6,
    'tip': 'script',
    'stabilan': true
};

console.log(skriptJezik.ime);
console.log(skriptJezik['stabilan']);
```



Može se napraviti instanca objekta direktno, korišćenjem ključne reči `new`. Takav objekat se može napraviti kao:

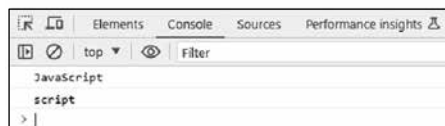
```
let mojObjekat = new Object();
```

Primer koji prati prethodni obrazac može se prikazati kao:

```
let mojObjekat = new Object();

mojObjekat.ime = 'JavaScript';
mojObjekat.verzija = 6;
mojObjekat.tip = 'script';
mojObjekat.stabilan = true

console.log(mojObjekat.ime);
console.log(mojObjekat ['tip']);
```



Može se napraviti objekat ključnom rečju `new`, gde se pravi funkcija sa argumentima. S druge strane, vrednost argumenta se može dodeliti trenutnom objektu korišćenjem ključne reči `this`, a koja predstavlja trenutni objekat.

Naredni primer ilustruje funkciju sa tri parametra funkcije. Unutar opsega funkcije, tj. blok koda postavljaju se vrednosti argumentima, a dve se prikazuju u konzoli kao test. Nakon opsega funkcije, definiše se lokalna promenljiva koja pravi novi objekat sa novim vrednostima koje popunjavaju prethodno definisana polja (kao parametri funkcije; ime, verzija, tip). U konkretnom primeru, parametri dobijaju vrednosti `JavaScript`, `6` i `Skriptni jezik`. Zatim se u konzoli podrazumevanog pretraživača prikazuje jedan od tipova, tj. verzija.

```
function programskiJezik (ime, verzija, tip) {  
    this.ime = ime;  
    this.verzija = verzija;  
    this.tip = tip;  
    console.log(this.ime);  
    console.log(this.tip);  
}  
  
let skriptniJezik = new programskiJezik('JavaScript', 6, 'Skriptni jezik');  
console.log(skriptniJezik.verzija);
```

13.17. Forme, događaji i osluškivači događaja

Događaji se mogu dešavati bilo gde u kodu. Suštinski, događaj daje informaciju programerima da se neki događaj desio. Primer događaja može biti kada korisnik (sa klijentske strane) ostvaruje interakcije sa veb stranicom. Interakcije mogu biti različitog karaktera, kao npr. reagovanje klikom miša, pomeranjem kursora miša, zatim, različiti se događaji mogu desiti prilikom korisničkog klika u odgovarajuće polje.

Takođe, primer dinamičke klijentske interakcije može biti korisnički unos teksta u polje za unos, zatim, potvrđivanje polja forme i drugo. Nakon ostvarivanja takvih događaja, *JavaScript* koristi osluškivač događaja (eng. *event listener*) koji može detektovati izvršenu akciju. Predefinisani nazivi i imena osluškivača događaja obično počinju sufiksom `on`, a primeri takvih osluškivača su

- `onclick`,
- `onload`,
- `onblur`,
- `onkeypress` i drugi.

Događaj aktivnosti mišem izvršava se i pokreće se prilikom korisničke interakcije sa elementom na veb stranici. Naime, korisnik može kliknuti na element (dugmad, linkove i drugo) koji u sebi sadrži implementiranu dinamičku funkcionalnost i koji može reagovati na svaki korisnički poziv.

Primer jednog takvog scenarija iz prakse može se predstaviti kroz interakciju sa navigacionim menijem. Kada korisnik klikne na polje navigacionog menija, meni se dinamički otvara, pružajući mu mogućnost da izabere željenu opciju. Nakon što korisnik završi sa interakcijom i napusti otvoreni navigacioni meni, meni se automatski zatvara kako bi se oslobodio prostor na ekranu. Te aktivnosti se mogu definisati događajima `onmouseover` i `onmouseout`.

U nastavku se može videti primer korišćenja.

```
<!-- HTML korisnička aplikacija -->
<form>
  <li>click (`onclick` osluškivač događaja)
    <span onclick="alert('Kliknuli ste na veb element!')">
      Izvršava se prilikom korisničkog klika na element, dugme i drugo.
    </span>
  </li>

  <li>mouseover / mouseout (`onmouseover` i `onmouseout` osluškivači događaja)
    <span onmouseover="console.log('Prešli ste mišem preko elementa!')"
      onmouseout="console.log('Prešli ste mišem preko elementa!')">
      Izvršava se kada kursor miša pređe ili napusti okvir elementa</span>
    </li>

  <li>mousedown / mouseup (`onclick/onmousedown` i `onmouseup`)
    <span onmousedown="console.log('Kliknuli ste mišem na element!')"
      onmouseup="console.log('Otpustili se taster miša sa elementa!')">
      Izvršava se onda kada je taster miša pritisnut ili otpušten sa elementa</span>
    </li>

  <li>mousemove (`onmousemove` osluškivač događaja) <br/>
    <span onmousemove="console.log('Prešli ste mišem preko elementa!')">
      Izvršava se onda kada je kursor miša pomeren.</span>
    </li>
</form>

// JavaScript kod koji može pratiti događaj
const dugme = document.querySelector('.element-dugme');
const naslov = document.querySelector('h1');

dugme.addEventListener('mousemove', primenjenaFunkcija);

function primenjenaFunkcija (evt) {
  console.log('Detalji događaja: ', evt);
  naslov.innerText = 'Promenjen tekst nakon klika!';

  document.body.style.backgroundColor = `rgb(${evt.offsetX},${evt.offsetY},50)`
}
```

Kao deo korisničke interakcije sa veb aplikacijom, jedan od mogućih događaja može biti interakcija putem tastature. Aktivira se prilikom pritiska (tipkanja) tastera na tastaturi. Postoje predefinisane funkcije koje se mogu implementirati u kodu, a koje osluškuju takve događaje; na primer, to su `onkeydown`, `onkeyup` ili prilikom pritiska odgovarajućeg tastera na tastaturi, tj. `onkeypress` osluškivač događaja.

Postoje sistemski izuzeci koji se ignorišu u nekim predefinisanim funkcijama. Funkcije nemaju mogućnost da generišu događaj `keypress` za tastere poput Esc, Ctrl, Alt, Shift

i drugih. Međutim, moguće je odgovoriti na takve tastere kroz događaje poput `keydown` ili `keyup`, koji se aktiviraju prilikom pritiska ili oslobađanja tastera tastature od strane korisnika.

```
<!-- HTML korisnička aplikacija -->
<label>Korisnička adresa stanovanja</label>

<input type="text" placeholder="Unesite naziv ulice:"
onkeydown="console.log('onkeydown pritisnut taster na tastaturi unutar input teksta!')"
onkeyup="console.log('onkeyup otpušten taster sa tastature unutar input teksta!')"
onkeypress="console.log('onkeypress drugi taster osim sistemskih tastera je pritisnut!')"/>

// JavaScript kod koji može pratiti događaj
const nazivTeksta = document.querySelector('unosNazivaTeksta');
nazivTeksta.addEventListener('keypress', nazivUlice);

function nazivUlice(evt) {
  console.log(nazivUlice funkcija pozvana');
  unesenTekst.innerText = evt.target.value;
}
```

Na modernim veb stranicama često se može primetiti prisustvo određenog tipa obrasca koji je moguće popuniti. Obrazac, odnosno forma na veb stranici, može sadržati različite elemente za interakciju, kao što su polja za unos teksta (jedno ili više), radio dugmad za odabir jedne opcije, padajuće liste za izbor vrednosti iz predefinisane liste i mnoge druge. Kontrola forme se aktivira kada korisnik ostvari interakciju sa formom, kao što je unošenje vrednosti u polja forme, odabir opcija iz padajuće liste ili izvršavanje drugih akcija predviđenih formom. Najčešći primeri predefinisanih događaja jesu:

- `onfocus` – izvršava se prilikom korisničkog reagovanja sa formom, npr. fokus na polje za unos teksta
- `onblur` – izvršava se onda kada korisnik ukloni fokus sa nekog polja, forme ili prozora
- `onchange` – izvršava se onda kada korisnik menja vrednost forme
- `onsubmit` – izvršava se prilikom korisničke potvrde forme (može biti klik na dugme)

U nastavku se može videti primer upotrebe prethodno navedenih funkcija:

```
<form onsubmit="alert('Podaci forme uneseni');">

  <label>Ime:</label>
  <input type="text" name="ime onfocus="oznaciTekst(this)"
    onblur="resetujOznakuTeksta(this)" required>

  <select onchange="console.log('Odabrali ste element iz padajuće liste');">
    <option>Pol</option>
    <option>Muško</option>
    <option>Žensko</option>
  </select>

  <input type="submit" value="Submit">

</form>
```

```
<script>

    // Tekstualno polje - focus
    function onfocus="oznaciTekst(trenutnoTekstPolje) {
        trenutnoTekstPolje.style.background = "red";
    }

    function rresetujOznakuTeksta(trenutnoTekstPolje) {
        trenutnoTekstPolje.style.background = "white";
    }

</script>
```

13.18. Brojevi

JavaScript podržava rad sa brojevima i ima samo jedan numerički tip podataka. Brojevi se mogu pisati u decimalnom formatu ili bez decimalnih mesta. Eksponenta notacija je moguća za male i velike brojeve. Brojevi u *JavaScript*-u se uvek čuvaju kao brojevi sa pokretnim zarezom (eng. *floating-point*). Standard koji se primenjuje nad *JavaScript* brojevima jeste IEEE 754. *JavaScript* interpreter funkcioniše prema principu čitanja koda (brojeva) sleva na desno.

Napomena: Često se mogu susresti i termini **pokretna tačka** ili **pokretni zarez**.

Standard IEEE 754 definiše pravilo gde se brojevi čuvaju u 64 bita. Frakcija, odnosno, broj se čuva u bitovima u rasponu od 0. do 51. bita. Eksponent se čuva u bitovima od 52. do 62. Znak se čuva u 63. bitu. Celobrojni brojevi, tj. brojevi bez tačke ili eksponenta tačni su do maksimalnih 15 cifara. Broj mogućih decimalnih mesta jeste 17.

JavaScript koristi operator + za dve operacije i to za sabiranje i konkatenciju stringova. Brojevi se mogu sabirati, dok se nad stringovima može izvršiti konkatencija, tj. spajanje. Ukoliko se brojevi sabiraju, rezultat se predstavlja u formatu broja.

Preporuka je da se brojevi u *JavaScript* jeziku ne pišu tako da počinju nulom; nije dobra praksa započeti broj nulom, kao npr. 011. *JavaScript* može tumačiti ovakav zapis kao oktalni broj, te se na izlazu dobija u potpunosti drugačiji rezultat.

Podrazumevani *JavaScript* decimalni brojevi imaju bazu, tj. osnovu 10. Postoje predefinisane metode za prikaz baze, odnosno, osnove u rasponu od 2 do 36; u te svrhe koristi se metod `toString()`. Kada je reč o heksadecimalnim brojevima, oni se predstavljaju podrazumevano osnovom 16. Takođe, decimalni brojevi imaju podrazumevanu osnovu 10, oktalni 8, a binarni osnovu 2.

13.19. Matematički objekti

JavaScript matematički objekti podesni su za izvršavanje matematičkih operacija. U *JavaScript* jeziku objekat `Math` je predefinisani i u sebi sadrži određen broj konstanti i metoda koje se mogu koristiti za izvršavanje raznoraznih matematičkih operacija. Primeri matematičkih operacija jesu: `min`, `max`, `sqrt`, `pi`, `random` i drugo.

```
// Vrednost PI
let piVrednost = Math.PI;
console.log(piVrednost);

// Zaokruživanje broja
console.log(Math.round(1.8)); // Zaokružuje broj
console.log(Math.ceil(1.8)); // Vraća manji intedžer
console.log(Math.floor(1.8)); // Zaokružuje na donju celobrojnu vrednost

// Koren broja
console.log(Math.sqrt(25));

// Vraća apsolutnu vrednost broja
console.log(Math.abs(-5));

// Metoda vraća vrednost baze podignutu na neku potenciju
console.log(Math.pow(5, 2)); //Rezultat je 10
console.log(Math.pow(5, 3)); //Rezultat se prikazuje kao: 5 * 5 * 5 = 25

// Vraćanje najmanjeg/najvećeg broja zadatih vrednosti
console.log(Math.min(15, 5, 3, 25, 2)); // Rezultat je 2
console.log(Math.max(25, 15, 3, 15, 1)); // Rezultat je 25

// Nasumični brojevi
console.log(Math.random());
console.log(Math.random() * 5 + 1);

// Random brojevi bez decimala - rezultat se vraća kao ceo broj
console.log(Math.floor(Math.random() * 5 - 1));
```

13.20. Nizovi

Specijalan tip promenljive koji u sebi može sadržati, tj. čuvati različite vrednosti. Predstavljaju kompleksne promenljive koje mogu sadržati u sebi više od jedne vrednosti. Definišu se uglastim zagradama, odnosno, [...]. Svaki član niza naziva se elementom niza. Elementi niza se navode zapetama, jedan za drugim. Svaki element niza ima numeričku vrednost, odnosno, poziciju. Ta pozicija se naziva indeks. Pozicija prvog elementa u nizu se indeksira na vrednost nula. Ukoliko postoji niz sa tri elementa, treći element se nalazi na indeksu 2, pri čemu se elementi broje kao 0, 1, 2.

Nizovi se mogu konstruisati na tri načina:

- Literalnom notacijom (konstante niza)
- Kreiranjem instance niza direktno (ključna reč new)
- Kreiranjem instance klase `Array()` (pomoću konstruktora objekta `Array`)

Najjednostavniji način za kreiranje niza jeste smeštanje vrednosti u uglaste zagrade; elementi niza se navode unutar istih. Primer definicije niza može se videti u nastavku:

```
let nizVrednosti = [vrednost0, vrednost1, ..., vrednostN];
```

U nastavku se može videti primer osnovne definicije niza.

```
// Primer 1
let nizBoja = ['Crvena', 'Zelena', 'Plava'];
console.log(nizBoja); // Prikazuje sve elemente niza

// Primer 2
let nizTehnologija = ['JavaScript', 'NodeJS', 'TypeScript'];
console.log(nizTehnologija[1]); // Prikazuje se drugi element niza

// Primer 3
let nizTehnologija = []; // Deklariše se promenljiva i prazan niz
// Definišu se vrednosti niza sa odgovarajućim indeksima
nizTehnologija[0] = 'JavaScript';
nizTehnologija[1] = 'NodeJS';
nizTehnologija[2] = 'TypeScript';
// Prikazuju se svi elementi niza
console.log(nizTehnologija);
// Izračunavanje dužine niza
console.log(Ukupno elemenata niza: ' + nizTehnologija.length);
```

Niz se može se kreirati direktno i pomoću ključne reči new. Naredni primer ilustruje obrazac za deklaraciju niza na ovaj način.

```
// Način 1
let niz = new Array();

// Način 2
let niz = Array();
```

U nastavku se mogu videti primeri kreiranja niza:

```
// Primer 1
let nizBoja = new Array();

nizBoja[0] = 'Crvena'
nizBoja[1] = 'Zelena'
nizBoja[2] = 'Plava'
console.log(arrColors); // Prikazuje sve elemnte niza

// Čita sve elemente niza
for (let i = 0; i < nizBoja.length; i++) {
  alert(nizBoja[i]);
}

// Primer 2
var nizTehnologija = Array();
nizTehnologija[0] = 'NodeJS';
nizTehnologija[1] = 'TypeScript';
nizTehnologija[2] = 'JavaScript';
nizTehnologija[3] = 'React';
console.log(nizTehnologija); // Prikazuje sve elemente niza
```

```
// Primer 3
var nizTehnologija = new Array();

// Dodavanje novog elementa/člana niza
for (let i = 0; i <= 4; i++) {
    nizTehnologija[i] = 'JavaScript';
}

console.log(arrTechnologies); // Prikazuje sve elemente niza
```

Instanca niza može se kreirati pomoću ključne reči `new` i pomoću instance klase `Array()`. Parametri se mogu proslediti u konstruktor klase, te nije neophodno eksplicitno navođenje vrednosti.

```
// Način 1 - obrazac za kreiranje članova niza
let clanoviNiza = new Array(clan0, clan1, ..., clanN);
```

U nastavku se mogu videti primeri kreiranja instance niza:

```
// Primer 1
let nizBoja = new Array('Crvena', 'Zelena', 'Plava');
console.log(nizBoja); // Prikazuje sve elemente niza

// Primer 2
let nizTehnologija = new Array('JavaScript', 'NodeJS', 'TypeScript');
console.log(nizTehnologija); // Prikazuje sve elemente niza

// Primer 3
let nizFrameworkTehnologija = new Array('Angular', 'React', 'Node', 'Vue');
console.log(nizFrameworkTehnologija); // Prikazuje sve elemente
```

13.21. Uslovni iskazi

Koristeći uslovne iskaze, moguće je izvršiti različite akcije na osnovu zadatih uslova. Uslovni iskazi omogućavaju donošenje ispravnih odluka i izvršavanje tačnih akcija na osnovu zadatih uslova. Mogu se koristiti uslovni operatori za proveru uslova: `>`, `<`, `<=`, `>=`, `==`, `!=`, `===`.

U *JavaScript* jeziku mogu se koristiti sledeći uslovni iskazi:

- If
- If...else
- If...else...if
- Switch...Case

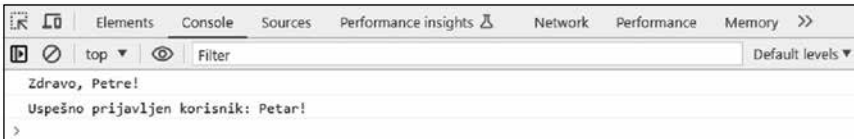
Uslovna izjava `if` predstavlja najosnovniji oblik uslovnog iskaza i služi za donošenje odluka i izvršavanje niza izjava na osnovu zadatih uslova. Koristi se za izvršavanje opsega,

tj. blok koda samo onda ukoliko dati uslov ima vrednost `true`. Takođe, izvršava se samo onda ukoliko je navedeni izraz tačan. U nastavku se može videti primer gde je definisana lokalna promenljiva `korisnik` kojoj je dodeljena string vrednost `Petar`. Proverava se da li je lokalna promenljiva zaista jednaka stringu `Petar`. Ukoliko jeste, ispisuje se u konzoli pozdravna poruka. Nakon uspešne provere uslova, u drugom koraku se prikazuje poruka koja potvrđuje uspešnu prijavu korisnika na imaginarni portal, sa prikazom imena korisnika koje se nalazi u lokalnoj promenljivoj.

```
let korisnik = 'Petar';

if (korisnik == 'Petar') {
  console.log('Zdravo, Petre!');
}

if (korisnik == 'Petar') {
  console.log('Uspešno prijavljen korisnik: ' + korisnik + '!');
}
```



Uslovni iskaz `If...else` koristi se za izvršavanje koda, ali bez obzira na ispravnost uslova, tj. da li je uslov tačan ili nije. Na taj način, programeru je omogućeno pružanje alternativnog izbora prilikom donošenja odluka, što dodaje dodatnu fleksibilnost u logici programa. Na primer, ukoliko je uslov tačan, izvršava se tvrđenje 1, u suprotnom, izvršava se tvrđenje 2.

```
// Struktura if...else uslovnog iskaza

if(uslov/izraz){
  // Sadržaj koji se izvršava ukoliko je uslov true - tvrđenje 1
}
else{
  // Sadržaj koji se izvršava ukoliko je uslov false - tvrđenje 2
}
```

U sledećem primeru se ilustruje dati događaj, gde se definiše lokalna promenljiva i dodeljuje joj vrednost u obliku stringa. Proverava se da li vrednost lokalne promenljive zaista odgovara definisanom stringu. Ukoliko jeste, prikazuje se odgovarajuća poruka; ukoliko izraz nije tačan, izvršava se `else` i prikazuje se druga poruka, tj. `Zdravo, svete!`.

```
let korisnik = 'Marija';

if (korisnik == 'Marija') {
  console.log('Zdravo, Marija!');
} else {
  console.log('Zdravo, svete!');
}
```

Uslovni iskaz `If...else...if` predstavlja posebnu verziju uslovnog iskaza koja može kombinovati više `If...else` uslovnih iskaza. Predstavlja napredniju formu `If...else` uslovnog iskaza. Omogućava da se donese tačna odluka na osnovu nekoliko različitih uslova. Ovaj uslovni iskaz prati naredni obrazac:

```
/*if(uslov/izraz 1) {  
    // Kod koji se izvršava ukoliko je uslov 1 true; izvršava se tvrđenje 1  
} else if(uslov/izraz 2) {  
    // Kod koji se izvršava ukoliko je uslov 1 false i uslov 2 je true;  
    izvršava se tvrđenje 2  
} else {  
    // Kod koji se izvršava ukoliko su uslov 1 i uslov 2 false;  
    izvršava se tvrđenje 3  
}*/
```

Na osnovu prethodnog obrasca, može se napraviti primer gde se ilustruje uslovni iskaz gde se poredi vrednosti lokalnih promenljivih. U tom slučaju, vrši se provera različitih uslova, počevši od provere da li su brojevi dodeljeni lokalnim promenljivama jednaki. Zatim se proverava da li je prvi broj manji od drugog. Ukoliko nijedan od tih uslova nije tačan, izvršava se treći iskaz, tj. `else` blok.

```
let broj1 = 50;  
let broj2 = 10;  
  
if (broj1 == broj2 ) {  
    console.log('Brojevi su jednaki');  
} else if (broj1 < broj2 ) {  
    console.log('Broj2 je veći!');  
} else {  
    console.log('Broj1 ije veći!');  
}
```

Naredni uslovni iskaz jeste `Switch...Case`, koji suštinski predstavlja alternativu prethodnom uslovnom iskazu, tj. `If...else...if`. Oba uslovna iskaza suštinski imaju istu svrhu. Svako tvrđenje se proverava i poredi sa svakim slučajem, tj. `case`. Ukoliko se slučaj poklapa – izvršava se odgovarajući blok koda; ukoliko nijedan slučaj nije ispravan, izvršava se podrazumevani (*default*) slučaj. Navedeni uslovni iskaz prati sledeći format:

```
switch(promenljiva/izraz){  
    case vrednost1:  
        // Kod koji se izvršava ukoliko je: promenljiva/izraz === vrednost1  
        break;  
    case vrednost2:  
        // Kod koji se izvršava ukoliko je: promenljiva/izraz === vrednost2  
        break;  
    default:  
        // Kod koji se izvršava ukoliko je promenljiva/izraz različita  
        od svih drugih vrednosti  
}
```

U nastavku se može videti primer koji se predstavlja gore opisani scenario, gde se, zapravo, proveravaju uslovi, a ukoliko nijedan nije ispravan – izvršava se podrazumevani:

```
let ocena = '10';
switch (ocena) {
  case '10':
    console.log('Svaka čast! - Ocena 10.');
```

break;

```
  case '9':
    console.log('Odlično! - Ocena 9');
```

break;

```
  case '8':
    console.log('Čestitamo! - Ocena 8.');
```

break;

```
  case '7':
    console.log('Uspešno! - Ocena 7.');
```

break;

```
  case '6':
    console.log('Položili ste! - Ocena 6');
```

break;

```
  default:
    console.log('Potrebno je još rada i učenja.');
```

break;

```
}
```

13.22. Petlje

Petlje igraju veoma važnu ulogu u svim programskim jezicima, pa tako i u *JavaScript* kodu. One su zadužene da izvrše isti blok koda jednom ili više puta, koristeći različite vrednosti. Petlja se izvršava sve dok se ne ispuni odgovarajući uslov. Osnovna namena petlji je automatizacija posla, čime se postiže značajna ušteda vremena. Korišćenjem petlji koda ima manje, a samim tim se kôd pravi optimalnijim i kompaktnijim. Najširu primenu ima u radu sa velikim nizovima ili velikim skupovima objekata. Petlje imaju iteracije u njima, što predstavlja ponavljanje instrukcija sve dok se uslov ne ispuni. Postoji nekoliko različitih tipova petlji u *JavaScript* jeziku; te petlje su sledeće:

- for
- while
- do while
- for...in
- for...of
- for...each

Petlja for koristi se prilikom izvršavanja određenog dela koda, određeni broj puta. Naime, petlja prolazi kroz čitav opseg blok koda, sve dok iterator (često nazivan: brojač) ne ispuni

zadati broj ponavljanja. Ponavlja se sve dok petlja ne ispuni odgovarajući uslov. Ova petlja predstavlja najjednostavniji oblik i u sebi sadrži tri naredbe, tj. izjave. Obrazac for petlje može se videti u nastavku:

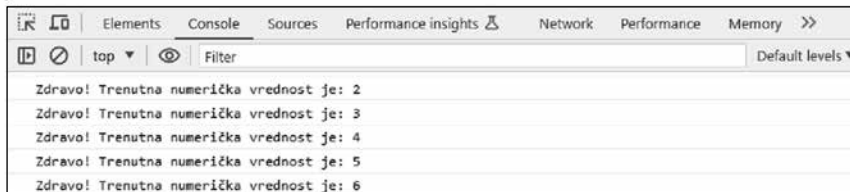
```
for (naredba 1; naredba 2; naredba 3) {  
    // Kod koji se izvršava  
}
```

Takođe, struktura tri naredbe u nastavku se može videti nešto detaljnije:

```
for(definicija promenljive; proveravanje uslova; inkrement/dekrement izraz)  
{  
    // Kod koji se izvršava  
}
```

U nastavku se može videti primer prikazivanja svih tekućih vrednosti, sve dok se uslov ne ispuni. Preciznije, petlja se zaustavlja kada je deklarirana vrednost manja ili jednaka vrednosti 10. Tokom iteracija, poruke će se ispisivati u konzoli, kao i u iskačućem prozoru (alert).

```
for (let i = 1; i <= 10; i++) {  
    alert('Zdravo! Trenutna numerička vrednost je: ' + i);  
    console.log('Zdravo! Trenutna numerička vrednost je: ' + i);  
}
```



Petlja while prolazi kroz kôd sve dok zadati uslov ne postane tačan. Za razliku od for petlje, gde promenljiva predstavlja deo petlje, u while petlji se promenljiva deklarise eksterno. U nastavku se može videti struktura while petlje i format zapisa. Zatim, prikazuje se eksterna deklaracija promenljive koja se koristi u petlji. Petlja se izvršava sve dok se uslov ne ispuni. Prilikom svake iteracije, petlja ispisuje poruku, zatim, proverava da li je uslov ispunjen; ukoliko nije, proces se ponavlja, ukoliko jeste – petlja se zatvara.

```
// 1. Struktura while petlje  
while(uslov) {  
    // Kod koji se izvršava  
}
```

```
// 2. Primer while petlje  
let i = 1;
```

```
while (i <= 10) {  
    console.log(alert('Zdravo! Trenutna numerička vrednost je: ' + i));  
    i++;  
}
```

Petlja `do while` je veoma slična prethodnoj petlji, osim što se provera postavljenog uslova izvršava na samom kraju petlje. Ono što ovu petlju čini drugačijom u odnosu na klasičnu `while` petlju, to je da se `do while` petlja uvek izvrši (najmanje) jednom i to pre provere uslova, čak i u slučaju netačnog uslova.

```
// 1. Struktura do while petlje
do {
    // Kod koji se izvršava
}
while(uslov);

// 2. Primer do while petlje
let i = 1;

do {
    console.log(alert('Zdravo! Trenutna numerička vrednost je: ' + i));
    i++;
}
while (i <= 10);
```

Petlja `for in` predstavlja specifičan tip petlje, koja suštinski prolazi u iteracijama kroz svojstva objekata ili kroz elemente niza.

```
// Struktura i primer for...in petlje
for(promenljiva in nizu/objektu) {
    // Kod koji se izvršava
}

// 1. Definisanje niza sa elementima niza (stringovi)
let nizBoja = ['Crvena', 'Zelena', 'Plava'];

// Petlja kroz koja prolazi kroz sve elemente u nizu
for (let boja in nizBoja) {
    console.log('Color is: ' + arrColors[color]);
}
```

Naredna petlja jeste `for...of`, ona je došla uvođenjem ECMAScript 6 standarda. Omogućava jednostavnu iteraciju kroz nizove ili stringove. Kôd se unutar petlje izvršava za svaki element *iterabla* objekta. Iterabla predstavlja iterativnu strukturu, a koja može implementirati ponašanje iteracije. Format zapisa `for...of` petlje može se videti u nastavku:

```
// Struktura for...of petlje

for(element u nizu/stringu) {
    // Kod koji se izvršava
}
```

Primer korišćenja `for...of` petlje sa iteracijama kroz niz i objekat, može se videti u primeru ispod:

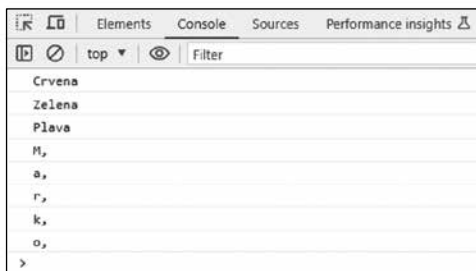
```

/*
  Iteracije kroz niz
  Definisanje niza sa elementima niza (stringovi)
*/
let nizBoja = ['Crvena', 'Zelena', 'Plava'];

for (let boja of nizBoja) {
  console.log(boja);
}

/*
  Iteracije kroz string
  Definisanje niza sa elementima niza (stringovi)
*/
let student = 'Marko';
for (let slovo of student) {
  console.log(slovo + ',');
}

```



Petlja `for...each` koristi se u *JavaScript* jeziku za iteraciju kroz elemente nizova ili objekata. Navedena petlja ima mogućnost izvršavanja koda za svaki element u nizu ili objektu koji implementira ponašanje interakcije. U nastavku je prikazan primer u kojem je definisana lokalna promenljiva `brojevi` koja sadrži niz brojeva. Nakon toga, koristi se referenca na definisani niz `brojevi` i izvršava se petlja koja prolazi kroz svaki element tog niza. Svaki element niza se prikazuje pojedinačno u konzoli.

```

let brojevi = [1, 2, 3, 4, 5];

brojevi.forEach(function(broj) {
  console.log(broj);
});

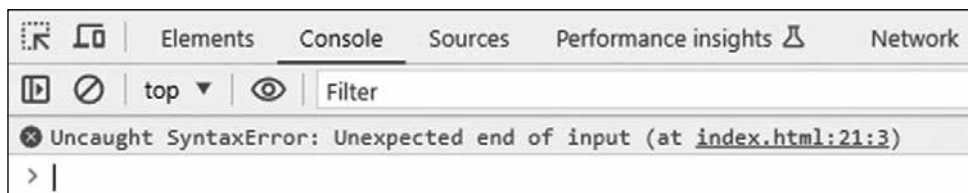
```

13.23. Greške

Prilikom pisanja *JavaScript* koda, mogu se javiti i greške. Razvojni prozor, tj. konzola pretraživača u tom slučaju je od ključnog značaja. Naime, prikazivanje greške (ili više njih) može biti od koristi, kako bi se lakše shvatilo na kojoj liniji koda je *JavaScript* program prestao da radi. Naravno, ne znači da će odgovarajuća poruka biti u potpunosti jasna i da će se jednostavno rešiti problem. Postoji lista *JavaScript* grešaka koja značajno može olakšati postupak otklanjanja istih, a ona je dostupna putem linka:

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Errors>.

Greške mogu biti različitog tipa, mogu biti sintaksne greške, logičke, zatim, greške nastale usled nedostajućeg argumenta (parametra). Primer jedne greške, nastale usled nedostatka vitičaste zagrade u opsegu koda, u konzoli se prikazuje kao:



Slika 13.4. Prikazivanje greške u konzoli pretraživača

13.24. Rezervisane (ključne) reči

JavaScript jezik sadrži u sebi određen broj rezervisanih ključnih reči. One se ne mogu koristiti kao identifikator nekog polja u okviru *JavaScript* koda. Takođe, rezervisane reči se ne mogu koristiti kao nazivi promenljivih ili funkcija. Na kraju, takve reči se ne mogu koristiti kao parametar ili promenljiva vrednost unutar petlje. U nastavku se može videti spisak rezervisanih ključnih reči:

```
arguments, await, break, case, catch, class, const, continue, debugger, default,
delete, do, else, enum, eval, export, extends, false, finally, for, function, if,
implements, import, in, instanceof, interface, let, new, null, package, private,
protected, public, return, static, super, switch, this, throw, true, try, typeof,
var, void, while, with, yield
```

13.25. Asinhroni *JavaScript*

Predstavlja povezivanje nove `async` funkcije za odgovarajuće ime. Koristi se ključna reč `await` unutar bloka funkcije. Podesan je jer se izbegava eksplicitno konfigurisanje lanca promisa. Samim tim, asinhroni *JavaScript* omogućava pisanje čistog asinhronog koda, zasnovanog na `promisima`. `Promis` predstavlja objekat koji je uspešno završen (ili odbijen) na osnovu asinhronog specifičnog zadatka; svaki `promis` nakon izvršenog zadatka dobija vrednost. Prilikom deklaracija asinhronih funkcija kreira se odgovarajući objekat. Takav kreiran objekat naziva se **asinhrona funkcija**. Prilikom poziva navedene funkcije, vraća se novi `promis`. On se razrešava vrednošću dobijenom od asinhrone funkcije.

Asinhrone funkcije u sebi mogu sadržati nijedan, jedan ili više `await` izraza. Mogu zaustaviti, odnosno, suspendovati izvršavanje koda sve dok se ne ispuni (ili pak odbije) vraćeni `promis`. Prilikom korišćenja `async` i `await` omogućava ujedno i korišćenje `try/catch` blokova koda, koji se može izvršavati u vezi sa asinhronim kodom.

U nastavku se može videti primer asinhrone funkcije koja razrešava zahtev nakon 10 sekundi i to prikazuje u konzoli. Vraća se `promis` koji koristi predefinisani metod `setTimeout` za izvršavanje funkcije koja se završava za određeni broj milisekundi. Nakon toga, funkcija pod nazivom `razresiZahtev()` se izvršava prikazuje u asinhronoj funkciji

```
function razresiZahtev() {
  return new Promise(razresi => {
    setTimeout(() => {
      razresi('Uspešno razrešen zahtev!');
    }, 10000);
  });
}

async function asinhronPoziv() {
  console.log('poziv');
  const rezultat = await razresiZahtev();
  console.log(rezultat); // Izvršava se kao: Uspešno razrešen zahtev!
}

asinhronPoziv();
```

13.26. Web API

API (eng. *Application Programming Interface*) predstavlja mehanizam za konstrukciju složenih funkcionalnosti veb aplikacije. Naime, Web API omogućava implementirane mnogobrojnih funkcija i objekata iz različitih kolekcija biblioteka koje su dostupne širom globalne mreže. Dok su neke kolekcije potpuno besplatne za korišćenje, postoje i kompanije koje naplaćuju pružanje usluga korišćenja njihovog API-ja. Poznate i popularne mreže koje pružaju programerima integraciju njihovih API-ja jesu *Google*, *Twitter*, *YouTube*, *Facebook*, *Pinterest* i druge. Programeru omogućavaju korišćenje jednostavnije sintakse uopšte. Svi moderni pretraživači podržavaju rad sa API-jem, što znači da se bilo koji pretraživač može koristiti za prikazivanje i manipulaciju podacima putem API-ja. Neki API-ji treće strane nisu inherentno integrisani u pretraživače i za njihovo korišćenje potrebno je ugraditi odgovarajući kôd u *JavaScript* program.

Generalno, Web API-ji omogućavaju manipulaciju nad dokumentima koji se učitavaju ili prikazuju u veb pretraživaču. Primer takvog API-ja može biti rad sa DOM-om (eng. *Document Object Model*), koji pruža mogućnost rada sa HTML jezikom i jezikom za stilizovanjem veb dokumenata tj. CSS-om. Takođe, rad sa DOM-om može biti mehanizam izvršavanja različitih akcija nad elementima veb strane, kao što je izvršavanja izmena, kreiranja ili uklanjanja HTML fajlova. Mogu se izvršavati različite izmene u pogledu vizuelnih akcija i drugo. API može preuzimati različite podatke sa servera, može raditi sa elementima za crtanje, manipulacijom grafike i audio sadržajem.

JavaScript objektna notacija, tj. JSON može se predstaviti kao oblik za predstavljanje podataka. Ovaj format se često koristi prilikom razmene podataka sa veb serverom u cilju njihovog prikazivanja u okviru veb aplikacije. Struktura i zapis JSON sintakse podseća na standardan *JavaScript* objekat, ali u pogledu izvršavanja *JavaScript* koda koristi se u potpunosti nezavisno. Objekti zapisani u JSON formatu obično se čuvaju u datoteci sa ekstenzijom `.json`.

Primer JSON formata može biti definisanje svojstava (*frontend* i *backend*) u kojem se svakom svojstvu pridružuje niz tehnologija, pri čemu svaki element niza predstavlja jednu od vrednosti.

```
{
  "frontend": [
    "HTML",
    "CSS",
    "JavaScript"
  ],
  "backend": [
    "Java",
    "PHP",
    "NodeJS"
  ],
  ...
}
```


13.27. Kompatibilnost

Pošto *JavaScript* kôd može da se izvršava u različitim okruženjima, neophodno je da se definisani kôd testira i proveriti da li radi na više različitih pretraživača. Treba napomenuti da svaki pretraživač ne izvršava na isti način *JavaScript* kôd.

Da bi se ove razlike neutralisale, *JavaScript* programeri treba da pokušaju da napišu kôd koji će u većini pretraživača da se izvršava na isti ili vrlo sličan način. Ako je to nemoguće, onda programeri treba da definišu procedure koje će proveravati da li pretraživači poseduju određene funkcije kako bi sve bilo u redu. Postoje slučajevi da dva pretraživača izvršavaju istu funkciju, ali se rezultati izvršenja te funkcije drastično razlikuju.

Možda je najjednostavniji način da se ovo reši da programeri naprave osnovne veb stranice bez "uplitanja" *JavaScript* koda i tada je sigurno da će se veb stranica slično ili isto izvršavati na različitim pretraživačima. Nakon toga može da sledi poboljšanje sadržaja namenjen korisnicima koji podržavaju *JavaScript*.

Očigledno je da mora da se povede računa o kompatibilnosti i materijala i njegovog prikazivanja usled primene različitih pretraživača ili platformi.

Glava 14

jQuery

Predstavlja brzu i efikasnu *JavaScript* biblioteku. Sadrži obiman skup različitih *JavaScript* biblioteka i omogućava jednostavnu i efikasnu manipulaciju DOM-om. *jQuery* predstavlja izuzetno interaktivnu biblioteku koja može izvršavati različite manipulacije nad HTML dokumentima, kao npr. može rukovati različitim događajima, zatim, može raditi sa animacijama, dizajnom, vizuelnim prikazima itd. Suštinski, ova biblioteka omogućava implementaciju izuzetno složenog i kompleksnog *JavaScript* koda na jednostavan način, koristeći pojednostavljene korake. Ova biblioteka je dobro optimizovana za skoro sve pretraživače, što znači da programer ne mora brinuti o tome da li će *jQuery* skripta ispravno funkcionisati na svim pretraživačima. Celokupna biblioteka dostupna je putem sledećeg linka: <https://jquery.com/>. Često se danas može čuti sve veći trend da ova biblioteka nije zastupljena u savremenim veb aplikacijama, zasigurno da nije u potpunosti isključena iz korišćenja. Mnogi popularni veb sajtovi poput *Google*, *Microsoft*, *IBM* i dalje koriste ovu biblioteku na svojim veb sajtovima.

14.1. Korišćenje

jQuery poseduje mnogobrojne besplatne dodatke (eng. *plugins*) koji mogu značajno unaprediti ukupno korisničko iskustvo. Pruža podršku za gotovo sve moderne pretraživače. Integracija *jQuery* biblioteke u pretraživače je jednostavna i može se postići na nekoliko načina. Može se preuzeti kôd biblioteke sa zvanične stranice i uključiti u HTML dokument, ili se može povezati sa HTML-om putem CDN-a (eng. *Content Delivery Network*). To omogućava jednostavno povezivanje sa raspoloživim bibliotekama. CDN predstavlja grupu međusobno povezanih servera, koji mogu isporučivati sadržaj korisnicima na osnovu geo lokacije. U konkretnom slučaju, korisniku se isporučuje *jQuery* biblioteka.

14.2. Verzije

Na raspolaganju su dve verzije *jQuery* fajla koji se mogu preuzeti. Naime, postoje produkcijska i razvojna verzija. Produkcijska verzija se može koristiti za žive veb stranice, dok se

druga, tj. razvojna koristi za testiranje. Razvojna verzija je čitljivija i nije komprimovana, dok produkcijska predstavlja suštu suprotnost u oba segmenta.

14.3. Ugradnja *jQuery* koda u dokument

Nakon što je verzija preuzeta sa zvaničnog sajta, ona se ugrađuje u HTML dokument između head tagova, a navodi se kao atribut izvora script taga.

```
<head>
  <script src="jquery-3.5.1.min.js">
</script>
</head>
```

Ukoliko se koristi CDN za *jQuery*, specifičan CDN kôd se ugrađuje jednostavno; navodi se kao atribut izvora unutar script tagova. Preporuka da se navede neposredno pred zatvaranje body taga HTML dokumenta.

```
<!doctype html>
<html lang="en">
<head>
  <title>Zdravo, svete!</title>
  <link rel="stylesheet" href="css/style.css">
</head>
<body>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js">
</script>
<script src="js/scripts.js"></script>
</body>
</html>
```

14.4. Selektori

Mogu se koristiti za označavanje, tj. selektovanje i manipulaciju odgovarajućih HTML elemenata koristeći specifičnu funkciju koja se piše kao `$(...)`. *JavaScript* jezik ima slične funkcije, poput `document.getElementById`, `document.getElementsByClassName` i drugo. Treba napomenuti da *jQuery* funkcija menja sve *JavaScript* predefinisane funkcije tog tipa. Obrazac koji prati korišćenje selektora može se definisati kao:

```
$(selektor).akcija( )
```

14.5. *jQuery* događaji

Isto kao i u *JavaScript* jeziku, događaji u *jQuery* implementaciji predstavljaju raznolike dinamičke akcije. Događaji mogu biti različita reagovanja mišem preko elemenata na veb stranici, odabir nekog polja, popunjavanja polja forme i drugo. Postoji nekoliko najčešćih predefinisanih *jQuery* događaja, a oni su `click()`, `keypressed()`, kao i `on()` koji može dodati naredni događaj nakon što se prethodni izvrši. Pozivanje metoda prate prethodno definisani obrazac selektora, te se svi navedeni događaji mogu definisati kao:

```
$(selector).click(function)
$(selector).keypress(function)
```

Za dodeljivanje događaja klika na paragrafima u veb dokumentu, koristi se funkcija koja se primenjuje na element paragrafa, a zatim se koristi metoda `click()` koja kao parametar ima funkciju sa pripadajućim kodom.

```
$("#p").click(function() {
    // Izvršavanje koda nakon klika mišem na paragraf
});
```

14.6. Implementacija koda

Kroz *jQuery* biblioteku, moguće je programirati različite događaje na veb stranici, uključujući klikove na dugmad, prelazak mišem preko elemenata (*hover*) i postavljanje fokusa na polja. U sledećem primeru se demonstrira reakcija na klik dugmeta putem prikazivanja odgovarajuće poruke. Dugme ima atribut `id`, putem čega se suštinski prikazuje generisani događaj (funkcija) unutar `script` tagova.

```
<button id="alertDugme">Pritisnuti dugme za generisanje prozora
</button>

<script type="text/javascript" src="JS/jquery.min.js"></script>
<script type="text/javascript">
$(document).ready(function () {
    $("#alertDugme").click(function (e) {
        alert("Inicirali ste ovu poruku klikom na dugme.")
    });
});
</script>
```

14.7. Manipulacija DOM-a

Manipulacija nad *Document Object Model* (DOM) i rad s njim može biti izazovan i zahtevan koristeći samo *JavaScript* jezik. *jQuery* takav mehanizam može značajno ubrzati, te ga čini izuzetno podesnim za operacije takvog tipa. Na primer, definisana je lista od pet gradova u obliku radio dugmadi. Prilikom korisničkog odabira polje koje predstavlja prestonicu Republike Srbije, na ekranu se prikazuje poruka "Tačno". U suprotnom, ispisuje greška "Netačno".

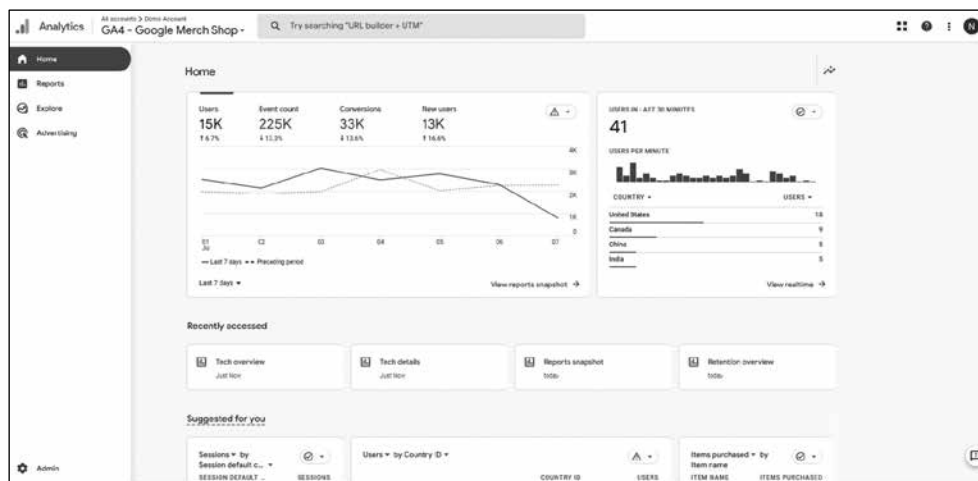
```
<ul>
<li><input type="radio" value="Beograd" name="radiopolje">Beograd</li>
<li><input type="radio" value="Kragujevac" name="radiopolje">Kragujevac</li>
<li><input type="radio" value="Niš" name="radiopolje">Niš</li>
<li><input type="radio" value="Novi Sad" name="radiopolje">Novi Sad</li>
<li><input type="radio" value="Užice" name="radiopolje">Užice</li>
</div>
```

```
<script type="text/javascript" src="JS/jquery.min.js"></script>
<script type="text/javascript">
$(document).ready(function () {
    $("input[type='radio']").click(function (e) {
        if ($(this).val()=="Beograd")
            $(this).parent().append
                ('<p style="margin:0;padding-left:10px;color:red;">Tačno</p>');
        else
            $(this).parent().append
                ('<p style="margin:0;padding-left:10px;color:red;">Netačno</p>');
    });
});
</script>
```

Glava 15

Google analitika

Neizostavni deo predstavlja analitika veb sajtova. Prvog jula 2023. godine Google je predstavio i lansirao najnoviju verziju svoje analitike pod nazivom *Google Analytics 4* (skraćeno GA4). Predstavlja značajno unapređenje u odnosu na svog prethodnika. U sebi ima implementirana nova svojstva za merenje analitike na veb sajtu. Naime, nova verzija donosi poboljšanja u pogledu prikupljanja podataka sa veb sajtova i aplikacija. Prethodnih godina akcenat analitike je stavljen na korisničku sesiju; GA4 sada prikuplja podatke koji se zasnivaju na korisničkim događajima. Kontrola merenja se zasniva na merenju bez kolačića (eng. *cookie*), što do sada nije bila praksa.

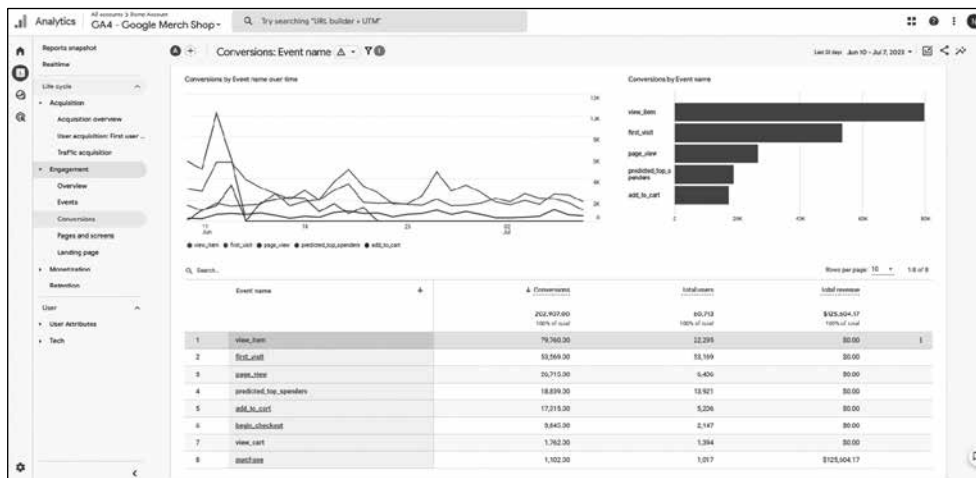


Slika 15.1. Ilustracija administratorskog panela verzije GA4 Google Merch Shop

15.1. Značaj analitike

Važnost analitike veb sajtova ogleda se primarno u marketingu i od značaja je marketinškom timu organizacije. Fokus analitike je usmeren na sve korisničke procese i inter-

akcije koje se odvijaju na veb sajtu. S tim u vezi, postoje korisnički procesi (najčešće prilikom kupovine nekog proizvoda ili usluge) koji imaju svoje odgovarajuće faze. Tri faze koje predstavljaju ove događaje jesu akvizicija, ponašanje i konverzija. Akvizicija predstavlja sticanje korisnika i stvaranje interesovanja kod korisnika za željenim uslugama ili proizvodima. Ponašanje predstavlja fazu gde korisnici mogu pristupati različitim interakcijama sa servisima dostupnim na veb sajtu. Proces konverzije (ili samo skraćeno: konverzija) jeste proces u kom korisnik postaje kupac, te može obaviti različite transakcije (kupovinu, dodavanje proizvoda u korpu, plaćanje putem interneta) i drugo.



Slika 15.2. Ilustracija konverzije demo verzije GA4 Google Merch Shop; prikazuju se nazivi događaja, konverzije, broj korisnika i drugo

Proces ovog tipa gotovo da je nemoguće prilagoditi životu van globalne mreže, tj. van digitalnog sveta – interneta. Digitalna analitika može uključivati različite aspekte korisničkih događaja i procesa na veb sajtu. Marketinškom timu jedne organizacije je od ključalnog značaja praćenje svih faza korisničkih procesa. Tokom korisničkog procesa, zapravo se mogu pratiti svi koraci koji jednog korisnika vode ka kupovini. Takođe, može se ustanoviti šta je to što je najpozitivnije uticalo na različite korisnike koji su odabrali isti put (grafike, vizuali, interaktivan sadržaj, naslovi itd.) do kupovine usluge ili proizvoda. Takvi podaci su izuzetno važni za unapređenje sadržaja veb sajta.

Prethodno opisani scenario može se preneti i na primer i scenario iz prakse. Hosting provajder nudi usluge hostinga. Cilj hosting organizacije je prodaja određenih vrsta usluga, tj. paketa, sa određenim brzinama protoka, brojem baza podataka, brojem domena (i poddomena) i određenim veličinama prostora na veb serveru. Marketinškim analizama korišćenjem Google analitike mogu se dobiti odgovori iz kojih gradova ili regiona u državi se najviše prodaju ponuđeni paketi usluga. Kroz ovaj pristup, marketinški tim ima mogućnost da precizno identifikuje ciljnu grupu korisnika. Dobijena analitika može se koristiti kako bi se pojačala marketinška kampanja za popularnu uslugu u određenom regionu ili gradu. Na osnovu analitičkih podataka, marketinški tim može identifikovati oblasti s visokim potencijalom i fokusirati marketinške napore na te delove kako bi ostvario efikasnije rezultate.

15.2. GA4

Google analitika (eng. *Google Analytics 4*) predstavlja platformu za prikupljanje podataka sa veb sajtova. Omogućava generisanje izveštaja i omogućava uvid u korisničke akcije koje su izvršene prilikom različitih interakcija sa veb sajtom. Analitika ima sposobnost merenja veb sajta. U svaku veb stranicu ugrađuje se *JavaScript* kôd, koji suštinski prati i prikuplja različite podatke i informacije o korisničkim aktivnostima na veb sajtu. Primer i struktura *Google* taga, koji se koristi za prikupljanje podataka i njihovo čuvanje, može se videti u narednom kodu:

```
<!-- Google tag (gtag.js) -->
<script async src="https://www.googletagmanager.com/gtag/js?id=TAG_ID">
</script>
<script>
  window.dataLayer = window.dataLayer || [];
  function gtag(){dataLayer.push(arguments);}
  gtag('js', new Date());
  gtag('config', 'TAG_ID');
</script>
```

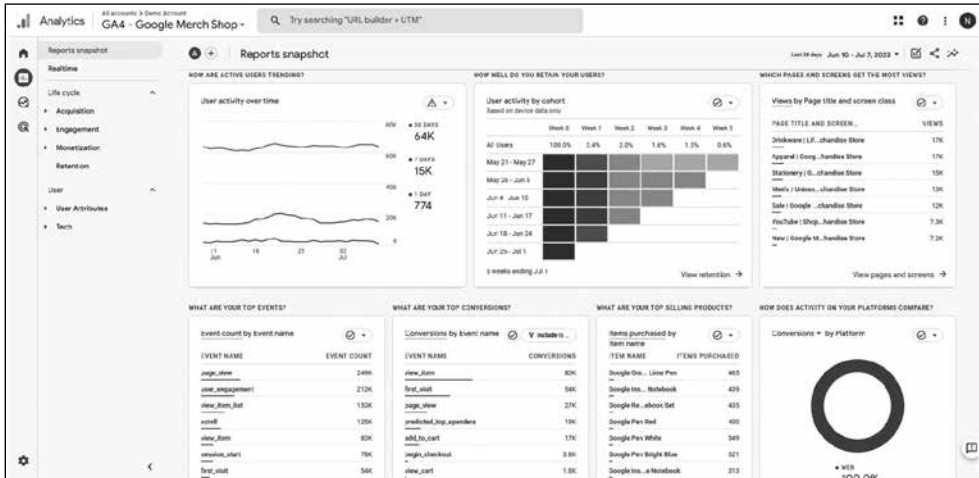
Veb sajt koji pruža usluge prodaje usluga ili proizvoda, ugrađenim kodom može se doći do informacija o komparativnom broju poseta različitih stranica. Na primer, može se pratiti komparativna analiza dve jeftinije i skuplje hosting usluge u vidu broja poseta. Takođe, može se pratiti da li je usluga stigla do digitalne korpe za kupovinu ili je korisnik odustao od iste.



Slika 15.3. Primer demo verzije GA4 Google Merch Shop, gde se može videti raznoliko izveštavanje po više kriterijuma

Prilikom prikupljanja podataka mogu se ustanoviti koji su pretraživač korisnici koristili u najvećem broju slučajeva, a koji najmanje. Ova informacija može biti izuzetno od koristi

kako bi se kompatibilnost veb sajta sa pretraživačem (npr. *Google Chrome*, *Firefox*, *Safari* itd.) usmerila na sveukupno poboljšanje korisničkog interfejsa. Osim informacija o pretraživačima, takođe se prikupljaju informacije o vrsti operativnog sistema koji se najčešće koristi prilikom pristupa veb sajtu. Takođe, analitika može otkriti kroz koje operativne sisteme je usluga zaista dovela do procesa kupovine. Takve informacije mogu motivisati dizajnere i programere da unaprede i razvijaju nove funkcionalnosti specifično za te vrste uređaja.

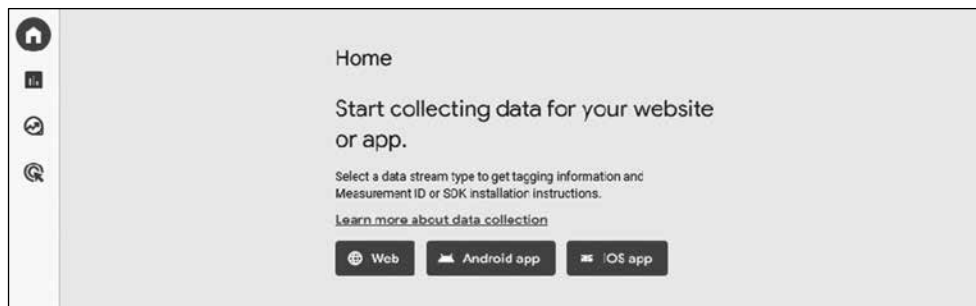


Slika 15.4. Prikazuje se korisnička aktivnost, najbolji događaji, konverzije i drugo, za demo verziju GA4 Google Merch Shop

GA4 (*Google Analytics 4*) omogućava kreiranje svojstava koja predstavljaju veb sajtove ili druge digitalne entitete. Ova svojstva koriste koncept razdvojenih podataka kako bi omogućila praćenje relevantnih tačaka kontakta sa korisnicima usluge. Te tačke šalju podatke svojstvu koje zatim može biti analizirano i pružiti uvide o ponašanju korisnika i performansama sajta ili entiteta.

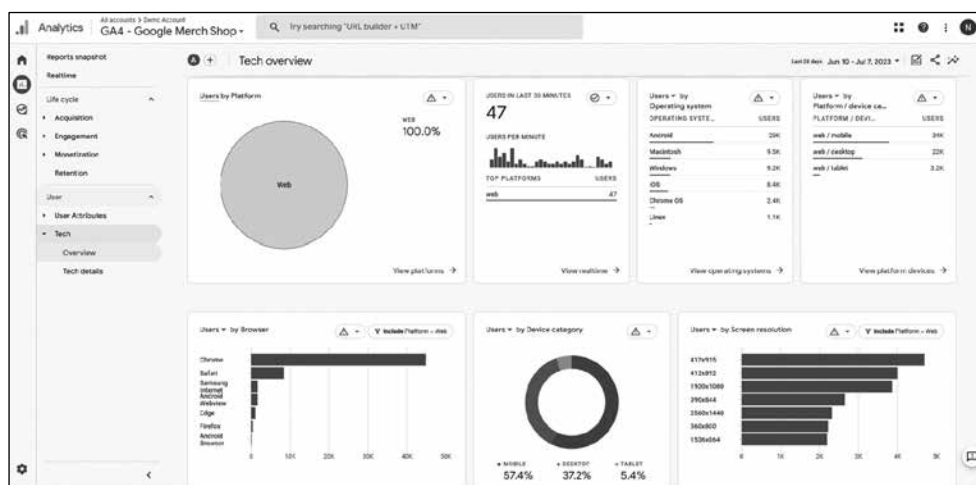
Pored standardnih GA4 funkcionalnosti u pogledu izveštaja i analitike, mogu se koristiti i *Google Analytics 360*. Ona predstavljaju podršku za velike organizacije i timove koji koriste naprednije alate kao što su *Unsampled Reports* ili rad sa *BigQuery*. Generalno, *Google Analytics 360* ima dodatna dva tipa svojstava, tj. podsvojstvo i *roll-up* svojstvo. Podsvojstvo može dobiti podatke iz drugog (npr. izvornog) svojstva i predstavlja podskup podataka. Drugo svojstvo, *roll-up* može dobiti podatke iz nekoliko drugih svojstava, te ih može kombinovati.

Kada se radi o analitici, važno je napomenuti da prikupljeni podaci o logičkoj veb aplikaciji mogu biti pregledani na različitim platformama i operativnim sistemima. Na primer, praćenje toka podataka može se vršiti na *Android* OS-u i *iOS*-u, pri čemu se za svaku aplikaciju kreira zaseban tok podataka (slika 15.5). GA4 čuva prethodnu istoriju tokova podataka i neće nastaviti s daljim procesiranjem i prikupljanjem podataka nakon što je tok u potpunosti uklonjen.



Slika 15.5. Odabir prikupljanja podataka za odgovarajuću platformu

U nastavku se može videti primer koji ilustruje administratorski panel gde se može videti izveštaj kojim se tehnologijama pristupalo (slika 15.6).



Slika 15.6. Primer pregleda korisničkih tehnologija koje su korišćenje za pristupanje platformi (demo verzija GA4 Google Merch Shop)

Korišćenjem funkcionalnosti praćenja uređaja putem kojih su korisnici ostvarili pristup ili interakciju sa veb sajtom (ili njenim delom), omogućava se značajan parametar u marketinškom pogledu. Pored pribavljanja informacija o lokaciji, podatak o tipu operativnog sistema je marketinškom timu od izuzetne koristi, s obzirom na to da se takva prikupljena analitika može iskoristiti u celokupnom unapređenju korisničkog iskustva. Pored tipova uređaja i operativnih sistema, posebno se može posvetiti pažnja rezoluciji ekrana koju su korisnici imali prilikom ostvarenih poseta. Optimizacija veb sajta, tj. veb aplikacije u tom segmentu može imati ključan faktor prilikom zadržavanja korisnika i ostvarivanjem jednostavnije pristupačnosti traženim resursima.

Google analitika predstavlja *on-line* softversko rešenje, koje korisnicima širom sveta omogućava praćenje analitike veb sajta, kako u realnom vremenu, tako i u zadatom opsegu. U današnjoj eri digitalnog sveta gotovo da je nemoguće lansirati veb sajt, bez ugrađene

podrške za analitiku. Kompanija *Google* svojim korisnicima nudi svoje rešenje analitike besplatno, ali na taj način kompanija od svojih direktnih korisnika ostvaruje, skoro, neiscrpan izvor podataka. Pored korisnih podataka koji se mogu prikupiti od korisnika prilikom unošenja različitih kriterijuma pretrage, kompaniji *Google* su značajni i loši rezultati pretrage. Takvi rezultati *Google* analitici predstavljaju značajan parametar, kako bi implementirani algoritmi procesuirali i obradili loš rezultat, a u cilju razumevanja sleda takvih događaja. Takođe, prethodnih godina, tj. u prethodnim verzijama *Google* analitika je prikupljala značajne informacije o korisnicima. Time se u široj javnosti stvorila slika u pogledu kršenja i zabrinutosti u vezi sa zaštitom ličnih podataka korisnika (GDPR). Nova verzija GA4 donosi značajna poboljšanja u tom pogledu, te je politika privatnosti usklađena sa usvojenim svetskim regulativama.

Glava 16

Saveti za dobar UI/UX dizajn

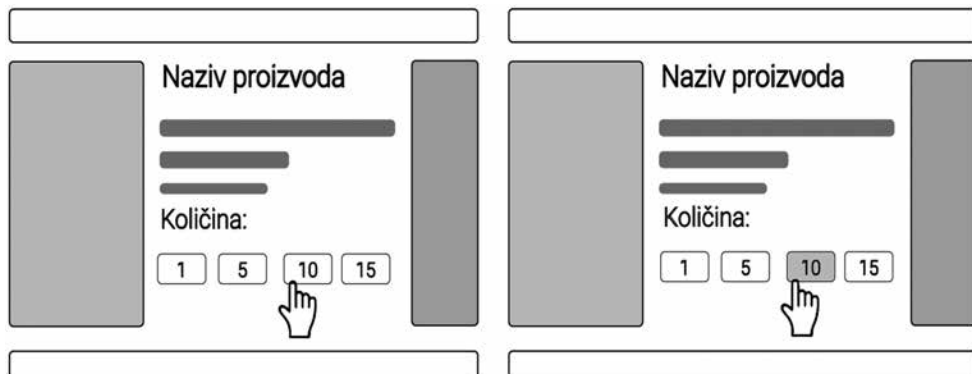
Dizajniranje korisničkog interfejsa i korisničkog iskustva, poznato pod terminom UI/UX (eng. *User Interface / User Experience*), igra veoma važnu, pa čak i ključnu ulogu prilikom ostvarivanja uspeha u pogledu upotrebljivosti veb ili mobilne aplikacije. U digitalnom svetu, gde korisnici virtualnim putem mogu ostvariti interakcije sa kompanijama i korporacijama širom sveta, jednostavnost u pogledu korišćenja veb sajta ili mobilne aplikacije je od krucijalnog značaja. U skladu s tim, ključno je da se obezbedi izuzetno kvalitetno dizajniran i intuitivan korisnički interfejs, kako bi krajnji korisnik lako razumeo kako da pristupi željenom proizvodu ili informaciji.

Različiti vizuelni i interaktivni aspekti jesu deo korisničkog interfejsa. Dizajn korisničkog iskustva se suštinski primarno oslanja na sveobuhvatno iskustvo korisnika. Pronalaženje balansa između korisničkog interfejsa i iskustva može kompaniji ostvariti pozitivne rezultate i uticati na sveukupni imidž brenda. Takođe, kupci postaju lojalni, a eventualna prodaja usluga koje nudi kompanija postaje traženija.

Važno je stvoriti snažan i pozitivan prvi utisak kod korisnika prilikom posete početnoj stranici veb sajta. Početna stranica treba biti kreativna i odražavati identitet i poslovanje agencije, organizacije, kompanije ili bilo kojeg drugog subjekta. Takođe, stranica ne bi se smelo odstupati od standarda koja jasno propisuje konzorcijum za donošenje pravila i standarda na vebu, tj. W3C; dostupan je pitam linka: <https://www.w3.org/>. Svako odstupanje od standarda ili uobičajenih pozicija na vebu može dovesti do konfuzije kod korisnika, jer mogu imati poteškoće pri pronalaženju željenih informacija. Sklad između tima dizajnera i programera je od suštinske važnosti za postizanje sveukupnog poboljšanja korisničkog iskustva. Prvenstveno, veb sajt bi trebalo da bude čist, jasan i dobro organizovan. Naslovi blogova, informacija o proizvodu i drugo trebalo da budu intuitivni, kako bi privukli korisnika da odabere traženi proizvod ili uslugu. Takođe, potrebno je voditi računa o tome da se stekne poverenje korisnika; relevantnost i tačnost podataka je od krucijalne važnosti. Slike i grafički vizuali suštinski utiču na sveukupni korisnički doživljaj, te bi oni trebalo da budu u potpunosti upečatljivi, tako da privuku korisničko interesovanje.

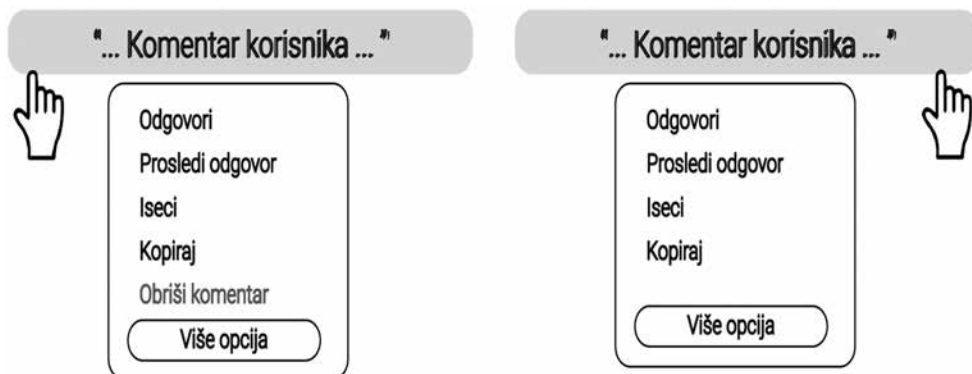
Heuristika je jako važan detalj veb sajta. Za izvođenje heurističkih testova nije potrebna složena računarska oprema, već se ona može izvršavati i na papiru. Za njeno sprovođenje potrebno je ljudstvo kako bi se uradila što bolja procena i evaluacija.

Primeri evaluacije mogu biti sledeći.



Slika 16.1. Poređenje dve verzije veb stranice pri odabiru količine nekog proizvoda

Naredni primer ilustruje poređenje dve varijante uređivanja korisničkih komentara, što može da bude prilično značajno u mnogima situacijama.



Slika 16.2. Primer poređenja uređivanja komentara



Slika 16.3. Primer poređenja polja pretrage, uz automatske sugestije

Primer pretrage u okviru veb sajta može uticati na korišćenje ovog polja kao takvog. U situaciji kada korisnik nije uneo ispravan kriterijum, preporučuje se implementacija mehanizma koji može prepoznati ključne reči. Na osnovu unetih karaktera, ova funkcionalnost može intuitivno sugerisati korisniku potencijalne rezultate pretrage, pružajući mu bolje smernice prilikom pronalaženju relevantnih informacija (slika 16.3).

16.1. Ciljna publika

Razumevanje ciljne grupe korisnika koja potražuje odgovarajući digitalni proizvod ključno je za korisničke potrebe i njihovu lojalnost brendu. Naime, sprovođenjem evaluacije, detaljnijim istraživanjem prema korisničkim potrebama, može značajno uticati i olakšati posao prilikom dizajniranja digitalnog proizvoda.

Pružanje mogućnosti korisnicima da imaju sopstveni korisnički profil, gde mogu uneti podatke od koristi kao što su njihove potrebe, omiljeni proizvodi i slično, može biti izuzetno korisno prilikom prikaza digitalnih proizvoda tokom korisničke prijave na veb aplikaciju. Ovo omogućava da se prijavljenim korisnicima pruži personalizovana i relevantna usluga, uzimajući u obzir njihove favorite u grupi ponuđenih usluga koje organizacija nudi.

16.2. Korisničko iskustvo i unapređenje stope konverzije

Dizajniranje korisničkog interfejsa može pozitivno uticati na ishod unapređenja korisničkog iskustva. Važnu ulogu igra pojednostavljenje nekog dela veb sajta (npr. navigacionog menija, polja za ostavljanje komentara, polje za pretragu itd.), kako bi se brže i lakše dolazilo do tražene informacije ili usluge. Povratne informacije koje dobijaju od korisnika mogu biti od velike koristi razvojnog timu kako bi unapredili personalizovano iskustvo za svoje kupce. Kroz te povratne informacije, tim može efikasnije identifikovati željene usluge i bolje zadovoljiti potrebe korisnika.

Ukoliko veb sajt nudi mogućnost kupovine putem digitalne korpe, pojednostavljene procesa plaćanja korisnicima može značajno doprineti povećanju prodaje. Dobra praksa jeste smanjenje svih komplikovanih procesa za krajnje korisnike. Komplikovani procesi sa korisničke strane se ogledaju u velikom broju nepotrebnih koraka.

Dugme CTA (eng. *Call to Action*), kao marketinški termin za poziv na akciju (npr. naredni korak koji bi trebalo da usmeri korisnika na odgovarajuću aktivnost, prilikom klika na dugme. Poboljšanje takvog dugmeta u pogledu vidljivosti, unapređenog dizajna, ubedljivih termina na poziv na akciju, može se povećati broj stope klikova, kao i konverzija. U osnovi, stopa konverzije predstavlja meru izvršenih akcija na veb sajtu. Preciznije, prati posetioce i korisnike koji završavaju započetu (ili željenu) akciju. Takođe, stopa konverzije meri i sâm cilj u odnosu na ukupan broj posetioaca, odnosno, korisnika.

Stopa konverzije se može predstaviti, kao što je prethodno navedeno, metrika koja prati i meri procenat završenih akcija, kao što je npr. završetak započete kupovine, zatim, popunjavanje različitih formulara, ostavljanje podataka u odgovarajuća polja forme i

drugo. Takođe, važna je za razumevanje želja i potrebe korisnika, a u cilju pretvaranja korisničke posete u stvarnu kupovinu usluge ili proizvoda. Dobra, tj. visoka stopa konverzije može ukazivati da je reč o dobroj optimizaciji sajta, te da je korisničko iskustvo pruža jasan cilj. U suprotnom, niska stopa generalno može ukazivati na to da je potrebna bolja optimizacija sajta, te je možda neophodno poraditi na boljem dizajnerskom rešenju, unaprediti postojeće poruke usmerene ka potencijalnim kupcima itd. Stoga, stopa konverzije je od izuzetnog značaja u cilju marketinške analize i celokupne analitike.

Proračun stope konverzije jednostavan, ali može biti izuzetno složen na velikim brojevima, tj. posetama i završenim procesima. Na jednostavnom primeru mehanizam proračuna se može prikazati kao: ukoliko veb sajt ima 100 poseta, a pet korisnika izvršava kupovinu nekog proizvoda ili usluge, stopa konverzije jeste 5%. Suštinski, ne postoji jednako i uniformno rešenje za svaki veb sajt, ali postoje servisi i alati poput *Google Analytics* koji mogu biti od koristi.



Slika 16.4. *Ilustracija banera koji poziva na akciju*

Važno je istaći da korisničko iskustvo ne predstavlja konačan proizvod prilikom procesa dizajniranja veb stranice. Suština ovog procesa je da istraživački tim sprovede studiju, analizira postojeća rešenja i uporedi ih sa prethodno dobijenim evaluacijama vizuala i drugih faktora. Moguće je na osnovu rezultata evaluacije dobijenih prilikom ispitivanja korisničkog iskustva dostaviti rezultate svim stranama, tj. dizajnerima korisničkog interfejsima i programerima.

Prema evaluaciji, dizajneri korisničkog iskustva mogu pripremati vizualizaciju veb sajta ili njegovih delova. Nakon izvršenih eventualnih izmena i korekcija, programeri mogu otpočeti proces implementacije funkcionalnosti i grafičke vizuale sa svim evaluacijama mogu pretočiti u kôd.

Kako bi korisnik došao najlakše do cilja, preporuka je praviti vizualizacije priče u stilu stripa. Tme se stvara jednostavnije vizualizacija korisničkog intereagovanja sa veb sajtom. Priča u stilu stripa može biti i šaljivog karaktera. Važno je izdvojiti tri komponente o kojim je potrebno povesti računa, a to su scenario, vizuali i komentari. Scenario predstavlja određeni deo aplikacije nad kojim se vrši ispitivanje iskustva. Vizuali predstavljaju

skup slika, skica i drugih grafičkih elemenata koji predstavljaju suštinski svaku iteraciju, od početka pa do kraja priče, odnosno, fragmenta koji se ispituje. Komentari prate, zapravo, svaki vizual uz različita stanja i radnje korisnika; prikazuju se okruženje, uređaji, signal interneta, domet, emocionalno stanje korisnika, takođe, može se prikazati scenario ukoliko veb sajt nema podršku za različite dimenzije ekrana i drugo.

Takođe, neophodno je izvršiti i evaluaciju upotrebljivosti. Ovaj proces je potrebno izvršiti pre nego što dizajnerski tim pripremi dizajn veb sajta ili određenog dela na samom veb sajtu. Upotrebljivost i njena evaluacija predstavljaju izveštaj koji obuhvata različite informacije, kao što su podaci o učesnicima, zadaci koje je dobio svaki učesnik da izvrši (ili koje je zadao aplikaciji), različiti korišćeni alati i drugo. Ovaj segment, zapravo, predstavlja konačne preporuke koje se mogu izvršiti na veb sajtu, za dati scenario.

16.3. SEO

Optimalan izgled veb stranice, kako za desktop računare, tako i za mobilne telefone može poboljšati opseg vidljivosti na vebu. Takođe, ukoliko veb sajt ima stranice prilagodljive različitim dimenzijama ekrana, postiže ujedno i bolje rangiranje na svetskim pretraživačima. Prilagodljiv dizajn veb stranice suštinski može uticati pozitivno na SEO (eng. *Search Engine Optimization*). Ova optimizacija odnosi se na optimizaciju za pretraživače.

U cilju postizanja dobrih indeksa pretrage, dizajn veb sajta koji je prilagodljiv svim veličinama ekrana od ključnog je značaja. Time se postiže da svaki pretraživač jasnije i preciznije razume sadržaj veb sajta, kao i njegov kontekst. Sveukupno može poboljšati rangiranje sajta u rezultatima pretrage. Takođe, može uticati pozitivno na vidljivost sajta određenog grupi korisnika.

Osim gore navedenih važnih segmenata, može se popraviti celokupno korisničko iskustvo. To značajno doprinosi i važan je faktor rangiranja. Prilagodljiv dizajn veb sajta omogućava krajnjim korisnicima da na svakom uređaju mogu jednako pristupati i koristiti usluge veb sajta; time se postiže zadržavanje korisnika, a može pozitivno uticati na njihovom aktivnom uzimanju učešća u korišćenju. Ovaj faktor je samo jedan u nizu koji značajno utiče na SEO.

Važno je istaći da pored ovakvih veb sajtova, od ogromne je važnosti imati kvalitetan sadržaj, kompletan, lako vidljiv i pristupačan, gde je korisničko iskustvo visoko. Takođe, pored navedenog, ključne reči na veb sajtu igraju važnu ulogu, kako bi se prilikom korisničke pretrage lakše poklapale reči i time se korisnici usmeravali ka veb sajtu. Takođe, važno je imati mnogo teksta, u mnogo različitih fraza, a korišćenje sinonima je takođe veliki plus.

Dobra praksa je pokazala da je korisno da svaki veb sajt koji je informativnog karaktera za agenciju, organizaciju ili kompaniju sadrži blog sa autorskim vestima vezanim za delatnost te organizacije. Time se postiže veći broj karaktera, a samim tim globalni pretraživači mogu potencijalno umeti da ponude blog sa vestima korisniku, kao rezultat korisničke pretrage.

Suvišni kodovi i bespotrebno usporavanje stranica istim značajno doprinosi optimizaciji brzine veb stranice. Upotreba relevantnih ključnih reči na veb stranicama može sveukupno doprineti celokupnoj optimizaciji i povećanju vidljivosti na vebu.

Literatura

1. Ben Frain: **Responsive Web Design with HTML5 and CSS: Build future-proof responsive websites using the latest HTML5 and CSS techniques**, *4th Edition*, Packt Publishing, 2022.
2. Jong-Moon Chung: **Emerging Metaverse XR and Video Multimedia Technologies: Modern Streaming and Multimedia Systems and Applications**, Apress, 2022.
3. Laurence Lars Svekis, Maaïke van Putten, Rob Percival: **JavaScript from Beginner to Professional: Learn JavaScript quickly by building fun, interactive, and dynamic web apps, games, and pages**, Packt Publishing, 2021.
4. David Flanagan: **JavaScript: The Definitive Guide: Master the World's Most-Used Programming Language**, *7th Edition*, O'Reilly Media, 2020.
5. Jason Beaird, Alex Walker, James George: **The Principles of Beautiful Web Design**, SitePoint, 2020.
6. Dragan Cvetković: **Multimedija**, UNIVERZITET SINGIDUNUM, Beograd, 2019.
7. Jennifer Robbins: **Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics**, *5th Edition*, O'Reilly Media, 2018.
8. Charles Marsh, David W. Guth, Bonnie Short: **Strategic Writing: Multimedia Writing for Public Relations, Advertising and More**, *Fourth Edition*, Routledge, 2017.
9. Ethan Brown: **Learning JavaScript: JavaScript Essentials for Modern Application Development**, *3rd Edition*, O'Reilly Media, 2016.
10. Dragan Cvetković, Dragan Marković, Nikola Savanović: **Multimedija**, UNIVERZITET SINGIDUNUM, Beograd, 2015.
11. Andy Harris: **HTML5 and CSS3 All-in-One For Dummies**, *3rd Edition*, For Dummies, 2014.
12. Tay Vaughan: **Multimedia: Making it Work**, *Ninth Edition*, McGraw-Hill Education, 2014.

13. Geary David: **Core HTML5 Canvas: Graphics, Animation, and Game Development**, Pearson Education, 2013.
14. Steve Fulton, Jeff Fulton: **HTML5 Canvas: Native Interactivity and Animation for the Web**, *Second Edition*, O'Reilly Media, 2013.
15. T. M. Savage, K. E. Vogel: **An Introduction to Digital Multimedia**, *2nd Edition*, Jones & Bartlett Learning, 2013.
16. Ian Devlin: **HTML5 Multimedia: Develop and Design**, Peachpit Press, 2012.
17. Craig Grannell, Victor Sumner, Dionysios Synodinos: **The Essential Guide to HTML5 and CSS3 Web Design**, Apress and friends of ED books, 2012.
18. Don Cowan: **HTML5 Canvas For Dummies**, *1st Edition*, For Dummies, 2012.
19. Jonathan Lane, Tom Barker, Joseph R. Lewis, Meitar Moscovitz: **Foundation Website Creation with HTML5, CSS3, and JavaScript**, Apress and friends of ED books, 2012.
20. Tay Vaughan: **Multimedia: Making it Work**, *Eighth Edition*, McGraw-Hill Osborne Media, 2011.
21. Jon Duckett: **HTML and CSS: Design and Build Websites**, *First Edition*, John Wiley & Sons, 2011.
22. Ranko Popović, Dragan Cvetković, Dragan Marković: **Multimedija**, UNIVERZITET SINGIDUNUM, Beograd, 2010.
23. Boriana Koleva: **Web Programming and Scripting**, *Teaching slides*, University of Nottingham, 2010.
24. Lisa DaNae Dayley, Brad Dayley: **Photoshop CS5 Bible**, Wiley Publishing, Inc., 2010,
25. Karen S. Ivers: **Multimedia Projects in Education: Designing, Producing, And Assessing**, *Fourth Edition*, Libraries Unlimited, 2010.
26. Parag Havaladar, Gerard Medioni: **Multimedia Systems: Algorithms, Standards, and Industry Practices (Advanced Topics)**, *1st Edition*, Cengage Learning, 2009.
27. Nigel Chapman, Jenny Chapman: **Digital Multimedia**, Wiley, *3rd edition*, 2009.
28. Simon Collison, Andy Budd, Cameron Moll: **CSS Mastery: Advanced Web Standards Solutions**, *Second Edition*, friends of ED, 2009.
29. David Sawyer McFarland: **CSS: The Missing Manual**, O'Reilly Media, 2009.
30. Jon Duckett: **Beginning HTML, XHTML, CSS, and JavaScript (Wrox Programmer to Programmer)**, Wrox, 2009.

31. Richard York: **Beginning JavaScript and CSS Development with jQuery**, Wiley Publishing, Inc., 2009.
32. Milenko Heleta, Dragan Cvetković: **Osnove inženjerstva i savremene metode u inženjeringu**, UNIVERZITET SINGIDUNUM, Beograd, 2009.
33. Jenq-Neng Hwang: **Multimedia Networking: From Theory to Practice**, Cambridge University Press, 2009.
34. Mike Wooldridge, Linda Wooldridge: **Teach Yourself VISUALLY HTML and CSS, 1st Edition**, Visual, 2008.
35. Grupa autora: **Adobe Dreamweaver CS4 – Classroom in a Book**, Adobe Press, 2008.
36. John Vince: **Geometric Algebra for Computer Graphics**, Springer-Verlag London Limited, 2008.
37. Rob Huddleston: **HTML, XHTML, and CSS: Your visual blueprint for designing effective Web pages**, Wiley Publishing, Inc., 2008.
38. Grupa autora: **Adobe Flash CS4 Professional Classroom in a Book**, Adobe Press, 2008.
39. Andy Harris, Chris McCulloh: **HTML, XHTML, and CSS – All-in-One Desk Reference For Dummies**, For Dummies, 2008.
40. Ray Villalobos: **Exploring Multimedia for Designers (Design Exploration)**, Delmar Cengage Learning, *1st edition*, 2007.
41. Kiki Eldridge: **The Complete Color Harmony Workbook: A Workbook and Guide to Creative Color Combinations**, Rockport Publishers, 2007.
42. Timothy Garrand: **Writing for Multimedia and the Web, Third Edition: A Practical Guide to Content Development for Interactive Media**, Focal Press, 2006.
43. Tay Vaughan: **Multimedia: Making it Work, Seventh Edition**, McGraw-Hill Osborne Media, 2006.
44. Chuck Musciano, Bill Kennedy: **HTML & XHTML: The Definitive Guide, 6th edition**, O'Reilly Media, 2006.
45. Elizabeth Castro: **HTML, XHTML, and CSS, Sixth Edition**, Peachpit Press, 2006.
46. Dragan Cvetković: **Računarska grafika**, CET i Računarski fakultet, Beograd, 2006.
47. Sanjaya Mishra, Ramesh C. Sharma: **Interactive Multimedia in Education and Training**, Idea Group Publishing, 2005.

48. Duncan Marsh: **Applied geometry for computer graphics and CAD**, *2nd edition*, Springer undergraduate mathematics series, Springer-Verlag London Limited, 2005.
49. J. Vince: **Geometry for Computer Graphics – Formulae, Examples & Proofs**, Springer-Verlag, New York, SAD, 2005.
50. Alan C. Bovik: **Handbook of Image and Video Processing (Communications, Networking and Multimedia)**, *Second Edition*, Academic Press, 2005.
51. Ze-Nian Li, Mark S. Drew: **Fundamentals of Multimedia**, Prentice-Hall, 2004.
52. John Dimarco: **Computer Graphics and Multimedia: Applications, Problems and Solutions**, Idea Group Publishing, 2004.
53. William Lee, Diana Owens: **Multimedia-based Instructional Design: Computer-Based Training; Web-Based Training; Distance Broadcast Training; Performance-Based Solutions**, *Second Edition*, Pfeiffer, 2004.
54. Edward Counts: **Multimedia Design and Production for Students and Teachers**, Allyn & Bacon, 2003.
55. Philip J. Schneider, David H. Eberly: **Geometric Tools for Computer Graphics**, Elsevier Science (USA), 2003.
56. J. Vince (Ed.): **Handbook of Computer Animation**, Springer-Verlag, New York, SAD, 2003.
57. Ken Jordan, Randall Packer: **Multimedia – From Wagner to Virtual Reality**, *Expanded Edition*, W. W. Norton & Company, 2002.
58. Rafael C. Gonzalez, Richard E. Woods: **Digital Image Processing**, *Second Edition*, Prentice Hall, SAD, 2002.
59. Sugata Mitra, Gaurav Bhatnagar: **Introduction to Multimedia Systems (Communications, Networking and Multimedia)**, *1st Edition*, Academic Press, 2001.
60. Grupa autora: **Computer Graphics, Multimedia & Animation**, *6th Edition*, Penrose Pr., 2001.
61. Kevin Jeffay, Hong Jiang Zhang: **Readings in Multimedia Computing and Networking (The Morgan Kaufmann Series in Multimedia Information and Systems)**, Morgan Kaufmann, 2001.
62. Stephen M. Alessi: **Multimedia for Learning: Methods and Development**, *3rd Edition*, Allyn & Bacon, 2000.
63. Fred Halsall: **Multimedia Communications: Applications, Networks, Protocols and Standards**, Addison Wesley, 2000.

64. Jerry D. Gibson: **Multimedia Communications: Directions and Innovations (Communications, Networking and Multimedia)**, Academic Press, 2000.
65. David Salomon: **Computer Graphics and Geometric Modeling**, *First Edition*, Springer, 1999.
66. J. D. Foley, A. van Dam, S. K. Feiner, J. F. Hughes: **Computer Graphics - Principles and Practice**, *Second Edition in C*, Addison-Wesley Publishing Company, Inc., Boston, SAD, 1997.
67. Jakob Nielsen: **Multimedia and Hypertext: The Internet and Beyond (Interactive Technologies)**, Morgan Kaufmann, 1995.

Internet adrese

- <http://www.aonlinetraining.com/>
- <http://www.scholarshipsinindia.com/courses.html>
- <https://www.pearson.com/english/en/professional-development/>
- <https://www.ed2go.com/courses/arts-and-design/>
- <https://www.academiccourses.com/Courses/Multimedia/>
- <http://www.wu.ece.ufl.edu/courses/eel6935f04/index.htm>
- <http://www.csd.uwo.ca/courses/CS1033/labs.html>
- <http://www.csd.uwo.ca/courses/CS1033/lectures.html>
- <http://ocw.mit.edu/courses/>
- <http://www.fanshawec.ca/programs-courses>
- <http://www.princetonreview.com/Majors.aspx?cip=090702&page=1>
- <https://www.maacindia.com/multimedia-design-courses.aspx>
- <https://www.courses.com.au/design-and-multimedia>
- <https://www.arenaerode.com/>
- <http://www.arena-thrissur.com/>
- <https://www.maacdilsukhnagar.com/>
- <https://www.indiemusicacademy.com/blog/music-licensing-overview>
- <https://creativecommons.org/>

- <https://wistia.com/learn/production/what-is-frame-rate>
- <https://flauntdigital.com/blog/video-frame-rates/>
- <https://www.nfi.edu/frame-rate/>
- <https://photographylife.com/best-frame-rate-for-video>
- https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/
- <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>
- <https://www.w3.org/>
- https://developer.mozilla.org/en-US/docs/Glossary/Grid_Tracks
- <https://developer.chrome.com/docs/devtools/css/grid/>
- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_grid_layout/

CIP - Каталогизација у публикацији
Народна библиотека Србије, Београд

004.4'27:004.738.1(075.8)
004.4'27(075.8)

ЦВЕТКОВИЋ, Драган, 1963-

Veb dizajn i multimedijalni sistemi / Dragan Cvetković, Nikola Savanović.
- Beograd : Univerzitet Singidunum, 2025 (Beograd : Birograf). - XII,
356 str. : ilustr. ; 24 cm

Tiraž 1.500. - Napomene i bibliografske reference uz tekst. -
Bibliografija: str. 351-356.

ISBN 978-86-7912-795-2

1. Савановић, Никола, 1988- [autor]

а) Web презентације -- Дизајн б) Мултимедији

COBISS.SR-ID 173406985

COBISS.SR-ID 30982409

© 2025

Sva prava zadržana. Nijedan deo ove publikacije ne može biti reprodukovан u bilo kom vidu i putem bilo kog medija, u delovima ili celini bez prethodne pismene saglasnosti izdavača.



Dragan Cvetković
Nikola Savanović

VEB DIZAJN I MULTIMEDIJALNI SISTEMI

Veb dizajn može da se shvati kao proces prikupljanja ideja i njihovog estetskog uređenja i raspoređivanja radi postizanja određenog cilja, a to je izrada veb sajta. Krajnji korisnici i njihovo što lakše snalaženje na određenom veb sajtu ono je što bi trebalo da veb dizajn najpre uzima u obzir. Veb dizajn podrazumeva upotrebu raznih veština i disciplina koje se koriste za planiranje i izradu veb stranica i sajtova.

Multimedija je zajednički naziv za medije koji kombinuju više tipova pojedinačnih medija, da bi se stvorila jedna celina. U običnom govoru multimedija najčešće znači interaktivni računarski projekat u kome se koristi tekst, zvuk, animacija, video i film.

Ova knjiga spada u grupu stručnih dela iz oblasti računarskih i inženjerskih nauka, koja pokriva značajno područje, koje se izučava na nivou dodiplomskih i poslediplomskih studija danas u svetu i kod nas.