



BOŠKO NIKOLIĆ

PROGRAMIRANJE GRAFIČKIH APLIKACIJA

Doc. dr Boško Nikolić

PROGRAMIRANJE GRAFIČKIH APLIKACIJA

- 2. dopunjeno izdanje -

Beograd, 2008.

PROGRAMIRANJE GRAFIČKIH APLIKACIJA

- 2. dopunjeno izdanje -

Autor:

Doc. dr Boško Nikolić

Recenzenti:

Prof. dr Milan Milosavljević

Prof. dr Mladen Veinović

Izdavač:

UNIVERZITET SINGIDUNUM

Beograd, Danijelova 32

Za izdavača:

Prof. dr Milovan Stanišić

Tehnička obrada:

Njeguš Novak

Dizajn korica:

Aleksandar Mihajlović

Godina izdanja:

2008.

Tiraž:

150 primeraka

Štampa:

Čugura print, Beograd

www.cugura.co.yu

ISBN:

Predgovor

Sadržinom i konceptom udžbenik Programiranje grafičkih aplikacija prilagođen je nastavnom planu i programu istoimenog predmeta, koji se izučava na IV godini Fakulteta za Informatiku i Menadžmenta, smjera za računarsku grafiku i dizajn.

Drugi predmeti sa ovog smjera predstavljaju studentima osnove korišćenja multimedijalnih elemenata i principe dizajna softverskih modula. Zato je cilj ovog predmeta da se studenti upoznaju sa tehnikama programiranja koje će im omogućiti korišćenje dosadašnjeg znanja radi realizacije savremene grafičke aplikacije.

Grafičke aplikacije imaju široku primenu, pa je odabrana oblast Interneta, kao trenutno najaktuelnija i najtraženija, za demonstriranje načina programiranja ovakve vrste aplikacija. Nakon položenog ispita studenti će biti u mogućnosti da reše i najsloženije zahteve programiranja klijentskih strana mrežnih aplikacija.

U prvom delu udžbenika detaljno se razmatraju osnovne mogućnosti programskog jezika HTML, jezika koji predstavlja standard za realizaciju Internet dokumenata. Na kraju ovog dela razmatraju se napredne i savremene tehnike HTML programiranja, kao što je CSS (Cascading Style Sheets).

Drugi deo udžbenika obuhvata programski jezik JavaScript, trenutno najpopularniji i najkorišćeniji skript jezik klijentskog dela Internet aplikacija. Ovaj jezik omogućava realizaciju i dinamičkih HTML strana.

Pored teorijskih razmatranja navedenih programskih jezika udžbenik sadrži i veliki broj praktičnih primera.

U okviru drugog izdanja knjige ispravljene su uočene greške iz prvog izdanja i dodat je novi Prilog sa karakterističnim ispitnim zadacima iz prethodnog perioda.

Autor



SADRŽAJ

1. HTML - Uvod	1
Primeri za rad u laboratoriji	9
2. FORMATIRANJE TEKSTA	11
Primeri za rad u laboratoriji	19
3. LISTE	23
Primeri za rad u laboratoriji	28
4. BOJE I SLIKE.....	33
Primeri za rad u laboratoriji	42
5. LINKOVI	45
Primeri za rad u laboratoriji	51
6. TABELE	55
Primeri za rad u laboratoriji	66
7. FREJMOVI.....	71
Primeri za rad u laboratoriji	77
8. ZVUK I FILM.....	85
9. FORME I ELEMINTI FORME	89
Primeri za rad u laboratoriji	96
Zadaci za samostalni rad	102
10. CSS (Cascading Style Sheets).....	105
Primeri za rad u laboratoriji	117
11. Java Script - UVOD.....	125
12. OSNOVE JavaScript JEZIKA.....	129
13. OPERATORI.....	133
14. KONTROLE TOKA.....	141
15. OBJEKTI Date i String.....	149
Primeri za rad u laboratoriji	154
16. RAD SA UZORCIMA (Pattern Matching).....	161
17. JavaScript I FORME.....	169
Primeri za rad u laboratoriji	183
18. RAD SA VISE PROZORA.....	187
19. PISANJE I ČITANJE COOKIE-IJA.....	194
Primeri za rad u laboratoriji	196
 LITERATURA.....	 199
PRILOG 1.....	200
PRILOG 2.....	203
PRILOG 3.....	208
PRILOG 4.....	210
PRILOG 5.....	211



1. HTML - UVOD

Kada se postavi pitanje kako definisati pojam Interneta i ljudi koji su najzaslužniji za njegov razvoj daju različite odgovore. Jedan od tih odgovora je da je to mreža svih mreža, koja povezuje univerzitetske, državne i privatne računare zajedno, i predstavlja infrastrukturu za korišćenje elektronske pošte, fajl arhiva, hipertekst dokumenata, baza podataka i ostalih resursa. Veliki skup računarskih mreža koje postoje i deluju zajedno za prenos podataka i poruka na daljinu i računare, koji mogu biti u susednoj kancelariji ili na bilo kom mestu u svetu.

Drugi odgovor je da je to najveća mreža na svetu koja koristi TCP/IP protokole i paketnu razmenu i izvršava se pomoću bilo koje komunikacione podloge.

Bez obzira na definiciju koja se koristi može se zaključiti da je Internet promenio način života, razmišljanja i poslovanja. O razvoju i mogućnostima Interneta mogu poslužiti i sledeći podaci koji govore o rastu broja servera (host računara) tokom godina:

- 1977: 111 host računara
- 1981: 213 host računara
- 1983: 562 host računara
- 1984: 1,000 host računara
- 1986: 5,000 host računara
- 1987: 10,000 host računara
- 1989: 100,000 host računara
- 1992: 1,000,000 host računara
- 2001: 150 – 175 miliona host računara
- 2002: preko 200 miliona host računara
- Do 2010, po procenama stručnjaka preko 80% planete će biti povezana preko Interneta

U septembru 2002. god. Internet je premašio dve važne brojke: 200 miliona host računara i 840 miliona korisnika.

Ovakav nagli razvoj jedne tehnologije do sada nije zabeležen. Takav zaključak nameće i poređenje sa sličnim tehnologijama koje se masovno upotrebljavaju, kao što su radio i televizija. Tržište od 50 miliona korisnika radio je dostigao posle 38 godina upotrebe, televiziji je trebalo 13 godina, a Internet je, od trenutka kada je postao javan servis, tu cifru dostigao za samo 4 godine.

Masovnu popularnost Internet duguje mnogim prednostima izvršavanja aplikacija preko ovakvog medijuma. Osnovne prednosti su jednostavnost korišćenja, globalan pristup, sve je dostupno na klik miša, aplikacije su platformski neutralne i dobija se isti izgled za većinu korisnika, jednostavna je upotreba interaktivnosti sa korisnikom, kao i nadogradnja postojećih verzija i jeftina distribucija.

Takođe, Internet još uvek nije savršen i postoje određeni nedostaci pri izvršavanja aplikacija na ovaj način, kao što su limit veza (posebno ako je u pitanju modemska pristup), sporost mreža, HTML je primitivan jezik, jer je u suštini jezik za opis stranica, interaktivnost je još uvek limitirana, posebno ako se uporedi sa drugim medijima, poput CD ili DVD medijuma, postoji još uvek nerešen problem intelektualne svojine i copyright-a.

Trenutno, podaci se putem Interneta prenose pomoću TCP/IP protokola, podeljeni u male pakete (chunks) veličine do 1.5K. Svaki paket ima svoju putanju, u zavisnosti od trenutnog saobraćaja. Poruka se sastavlja u krajnjoj tački putanje, pri čemu se eventualno izgubljeni paketi mogu ponovo tražiti i ponovo poslati. Treba naglasiti da Internet i TCP/IP protokol nisu projektovani sa multimedijalnim elementima kao zahtevima dizajna. Multimedijalni podaci se šalju u vidu kontinualnog niza i generišu veliki saobraćaj. Takođe, multimedijalni podaci bi trebali da stignu u redosledu kako su i poslati, a sam TCP/IP ne garantuje takav poredak. Rad sa multimedijalnim podacima ne trpi čekanje na pakete koji nedostaju, pa je gubitak sinhronizacije kritičan momenat (zvuk i slika do 80ms).

Put do savremenog Interneta nije bio jednostavan i bilo je potrebno rešiti mnoge izazove, kao što su: pronalazak digitalnih mreža, u današnjem obliku, razvoj infrastrukture za slanje i primanje elektronskih poruka zadovoljavajućom brzinom i mogućnosti razmenjivanja poruka između računara.

Pored TCP/IP protokola popularnosti Interneta je veoma doprineo i HTML (HyperText Markup Language), jednostavan jezik koji služi za izvršavanje programa na daljinu. Danas, ovaj jezik predstavlja standard za Internet dokumente. Njegove osnovne osobine su da je to platformski neutralan jezik, pomoću koga se specificira određeni sadržaj, koristeći logičke oznake (markup), izvršava se i prikazuje pomoću Web čitača.

HTML stranice imaju ekstenziju .html ili .htm, a nalaze se u određenom direktorijumu servera vezanog na ovu mrežu, što ih čini dostupnim putem Interneta.

Sa razvojem Interneta došlo je do pojave nove vrste dokumenata - hiperteksta. To je tekst koji sadrži veze ili linkove ka drugim dokumentima ili na samog sebe. Preciznije, hipertekst je skup stranica u obliku datoteka, međusobno povezanih linkovima koje su umetnute u stranice. Na ove linkove se može kliknuti. Za razliku od običnog teksta, koji se čita linearno (sleva na desno, odozgo naniže), hipertekst se čita prateći hiper-veze u tekstu, dakle, ne nužno na linearan način.

Pored samog HTMLa, postoji veći broj jezika koji omogućavaju da se precizno opiše izgled i sadržaj jednog teksta. Od posebnog su značaja:

- SGML (skr. od *Standard General Markup Language*),
- TeX i LaTeX (za matematičke tekstove),
- PostScript (jezik koji se koristi pri korišćenju laserskih štampača),
- RTF (skr. od *Rich Text Format*), ...

Takođe, postoji i više jezika koji se bave samim opisom hiperteksta, Među njima naznačajniji su:

- SGML,
- XHTML (skr. od *Expandable HTML*) i
- XML (skr. od *Extensible Markup Language*, prelaz između preterane složenosti SGML-a i jednostavnosti HTML-a; njegova standardizacija je u toku)

Za generisanje HTML stranice potreban je najobičniji tekst editor, na primer Notepad. Ovaj program se u okviru Windows operativnog sistema, nalazi u Start meniju pod Programs->Accessories. Osnovno korišćenje ovog programa je opisano u Prilogu 1.

Za pisanje HTML programa mogu se koristiti i specijalni alati, na primer Front Page ili Dreamviewer. Stranica se može videti u okviru Web browser-a, kao što su Netscape Navigator, Internet Explorer, Opera ili Mozilla.

Tag

HTML komande se pišu u vidu tagova. Jedan tag je u suštini komanda koja govori čitaču kako i na koji način da prikaže sadržaj opisane stranice.

Tagovi se pišu unutar oznaka "<" i ">" (bez znakova navoda) npr:

```
<html>
```

Ovaj tag se nalazi na početku svakog HTML dokumenta

<html> govori browseru da je fajl koji je upravo počeo da učitava HTML dokument i da kao takvog treba i da ga prikaže. Većina tagova ima i početni i završni tag. Završni tag se dobija dodavanjem znaka "/" i označava mesto na kom prestaje dejstvo početnog taga. Tako se na kraj HTML dokumenta piše završni HTML tag:</html> i ovaj tag govori browseru da je to kraj HTML dokumenta.

HTML tagovi su "case insensitive" tj. svejedno je da li se pišu malim ili velikim slovima, pa je isto značenje prethodnog taga, kao i sledećeg

<HTML>

U opštem slučaju tagove možemo podeliti u sledeće grupe:

- prosti tagovi se koriste za opisivanje jednostavnih elemenata logičke strukture. Oblika su: <X>
- složeni tagovi su zgrade oblika <X> *tekst* </X> kojima je opisan izgled dela teksta *tekst*.
- atributi u okviru taga: <X A1=a A2=b ...> *tekst* </X> koji pružaju dodatne informacije dela teksta *tekst*

Naredna tabela prezentuje listu svih tagova koji se mogu koristiti u okviru HTML stranice. U okviru ovog teksta se objašnjavati sintaksa i primena onih tagova koji se najčešće koriste u praktičnim realizacijama.

Ime taga	Opis taga
A	Link
ACRONYM	
ADDRESS	Informacija o autoru
APPLET	Java aplet
AREA	Na klijentskoj strani prostor za sliku u obliku mape
B	Boldovan tekst
BASE	Osnovni URI za dokument
BASEFONT	Osnovni veličina fonta
BIG	Tekst stil
BLOCKQUOTE	long quotation
BODY	Telo dokumenta
BR	Nova linija
BUTTON	Komandno dugme
CAPTION	Naslov tabele

CENTER	Centriran tekst
CITE	Navod
CODE	Fragment koda
COL	Kolona u tabeli
COLGROUP	Grupa kolona utabeli
DD	Opis definicije
DEL	Obrisan tekst
DFN	Definicija instance
DIR	Lista direktorijuma
DIV	Odeljak
DL	Definiciona lista
DT	Definicioni pojam
EM	Emfaza
FIELDSET	Grupa u okviru forme
FONT	Definicija fonta
FORM	Forma
FRAME	Pojedinačni frejm
FRAMESET	Skup frejmova
H1	Naslov
H2	Naslov
H3	Naslov
H4	Naslov
H5	Naslov
H6	Naslov
HEAD	Zaglavlje stranice
HR	Razdvojna linija
HTML	Osnovni tag
I	Italik
IFRAME	Inline frejm
IMG	Slika
INPUT	Element forme
INS	Insertovan tekst
KBD	Tekst koji se definiše od strane korisnika
LABEL	Labela
LEGEND	Opis fieldset-a
LI	Stavka liste
LINK	Link nezavisan od medijuma
MAP	Na klijentskoj strani slika u obliku mape

MENU	Lista menija
META	Osnovne meta informacije
NOFRAMES	Definisanje ponašanja u slučaju da čitač ne podržava forme
NOSCRIPT	Definisanje ponašanja u slučaju da čitač ne podržava script jezike
OBJECT	Objekat
OL	Numerisana lista
OPTGROUP	Grupa opcija
OPTION	OPcija u okviru SELECT elementa
P	Paragraf
PARAM	Osobina elementa
PRE	Preformatiran tekst
Q	Kratak citat
S	Precrtan tekst
SAMP	Primer izlaza programa, skriptovi, itd.
SCRIPT	Definicija script jezika
SELECT	Selekcija opcija
SMALL	Tekst stil
SPAN	Osnovni jezik
STRIKE	Precrtan tekst
STRONG	Tejkst stil
STYLE	Informacije o stilu
SUB	Subscript
SUP	Superscript
TABLE	Tabela
TBODY	Telo tabele
TD	Ćelija tabele
TEXTAREA	Tekst proctor
TFOOT	Footer tabele
TH	Ćelija za header tabele
THEAD	Zaglavlje tabele
TITLE	Naslov dokumenta
TR	Red u tabeli
TT	Tekst stil
U	Podvučen tekst
UL	Nenumerisana lista
VAR	Instanca promenljive ili argumenta

Struktura HTML stranice

Minimalna struktura HTML-dokumenta obuhvata tagove:

`<HTML>`, `</HTML>` - tagovi koji definišu HTML stranu;

`<HEAD>`, `</HEAD>` - tag zaglavlja, sadrži meta-definicije HTML-dokumenta i informacije koje ne pojavljuju eksplicitno u okviru dokumenta;

`<TITLE>`, `</TITLE>` - tag za naziv HTML-dokumenta i

`<BODY>`, `</BODY>` - tag koji definiše telo dokumenta.

Opšti izgled stranice je sledeći:

```
<html>
```

```
<head>
```

```
<title>
```

ovde je naziv stranice

```
</title>
```

```
</head>
```

```
<body>
```

ovde se unosi sve ono sto se prikazuje u okviru stranice

```
</body>
```

```
</html>
```

U opštem slučaju HTML stranica se može podeliti na dva osnovna dela. To su zaglavlje stranice (engl. head) i telo stranice (engl. body). Zaglavlje se odvaja tagovima `<head>` i `</head>` a telo dokumenta tagovima: `<body>` i `</body>`.

U okviru zaglavlja dokumenta definišu se osnovne informacije o stranici kao što su korišćeni alat za realizaciju stranice, korišćeni kod za prikazivanje stranice u okviru čitača, stil stranice, itd.

Sve ono što je napisano između tagova `<body>` i `</body>` predstavlja telo dokumenta i pojaviće se kao sadržaj prezentacije u prozoru browsera.

Posle svakog atributa se stavlja znak jednakosti koji služi za dodelu vrednosti datom atributu, a vrednost atributa se obavezno stavlja između znakova navoda.

Komentar

Kao i većina programskih jezika, HTML omogućava i pisanje komentara u okviru stranice. Tag koji se koristi je `<!--` za početak komentara i `-->` za kraj komentara.

Na primer:

```
<!-- ovaj tekst  
           je komentar  
koji se neće prikazati na stranici  
-->
```



Primeri za rad u laboratoriji

Primer 1. Prvi program u HTML-u.

```
<HTML>  
  <HEAD><TITLE>Ovo je naslov</TITLE></HEAD>  
  <BODY>  
    TEKST KOJI SE PRIKAZUJE  
  </BODY>  
</HTML>
```





2. FORMATIRANJE TEKSTA

U prethodnom poglavlju navedeni su razlozi nastanka i ciljevi ovog programskog jezika. Istaknuto je da je njegova prvobitna namena bila što brža i jednostavnija razmena i pregled dokumenata. Zato i HTML poseduje čitav niz tagova koji omogućavaju željeno formatiranje određenog teksta. U okviru ovog poglavlja se opisuju najvažniji tagovi iz ove grupe sa primerima korišćenja.

Naslov

Uvek je u okviru nekog teksta potrebno istaći naslove koje postoje. Takođe, naslovi se mogu razlikovati po važnosti i nivou teksta kome prepadaju. Zato se u HTML-u naslovi (engl. *headers*) prikazuju prema relativnom nivou ciframa od 1 do 6. Opšti oblik taga za naslov je:

`<Hn> Tekst za naslov nivoa n </Hn>`

gde n dobija vrednosti od 1 do 6.

Primer:

`<H1> Naslov H1 </H1>`

`<H2> Naslov H2 </H2>`

`<H3> Naslov H3 </H3>`

Navedeni kod se na HTML stranici prikazuje na sledeći način:

Naslov H1

Naslov H2

Naslov H3

Tekst

U uvodnom poglavlju je navedeno da HTML nije case sensitive jezik. Treba naglasiti da postoji razlika između naredbi, odnosno tagova, koji nisu case sensitive, i teksta koji se prikazuje u okviru stranice, koji jeste case sensitive. To znači da onako kako je napisan neki tekst, tako će se i prikazati (ako se ne koriste tagovi za formatiranje) na stranici. Takođe, ako se drugačije ne naglasi tekst unutar HTML stranice je poravnat uz levu stranu. Centriranje teksta ili bilo kog drugog elementa stranice se dobija pomoću tagova `<center>` i `</center>`. Tako se dobija sledeći rezultat:

OVO JE CENTRIRAN TEKST

izvršavanjem sledećeg koda

<center> OVO JE CENTRIRAN TEKST </center>

Paragraf i odeljak

Tekst stranice se može podeliti na određeni broj logičkih celina. Osnovne celine u HTML jeziku su paragraf i odeljak

Prelazak u novi paragraf se postiže <p> tagom. Ovaj tag odvoji tekst sa razmakom između redova. Na kraju paragrafa se preporučuje da se umetne i njegov završni tag, ali većina browsera ga ne zahteva, tako da se slobodno može i izostaviti. Ukoliko u ravnom tekstu sledi paragraf za paragrafom, tag </P> se može izostaviti. Tekst se, ako se ništa ne navede, poravnava uz levu marginu. Poravnanje na drugačiji način se postiže **align** atributom u okviru ovog taga. **Align** atribut može imati tri vrednosti: left, center i right i koristi se na sledeći način:

<p align="left">

<p align="center">

<p align="right">

Primer:

levo

centrirano

desno

Kod koji prikazuje gornji primer je:

<p align="left">levo <p align="center">centrirano <p align="right">desno

Odeljak (engl. *division*) se dobija upotrebom tagova <div> i </div>. Tekst koji predstavlja odeljak se od ostalog teksta odvaja novim redom bez razmaka. Ovaj tag može imati atribut za pozicioniranje ALIGN sa istim vrednostima kao tag <p>.

Font

Ukoliko se posebno ne naglasi kojim fontom se želi da bude ispisan tekst na stranici, browser će koristiti default font, uobičajeno Times New Roman. HTML omogućava nekoliko promena vezanih za postojeći font pomoću taga. Ovaj tag može imati sledeće attribute: **face**, **size** i **color**.

Atributom **face** se definiše tip fonta kojim se želi da tekst bude ispisan. Na jednoj stranici se može koristiti i više različitih fontova. Mora se voditi računa da korisnik koji učitava stranicu mora imati instaliran font koji je naveden na svom računaru. U suprotnom njegov browser će prikazati tekst u default fontu. Zato ne treba koristiti neke egzotične fontove, već treba upotrebljavati samo široko rasprostranjene fontove.

Ako se želi da stranica bude ispisana na primer "Comic Sans MS" fontom onda treba navesti sledeći kod:

```
<font face="Comic Sans MS"> Ovde dodje vas tekst. </font>
```

i dobija se rezultat

Ovde dodje vas tekst.

Moguće je navesti i nekoliko fontova alternativni, jedan za drugim, odvojenih zarezom. Ako posetilac stranice na svom računaru nema instaliran prvi font sa liste, on će ga prikazati u sledećem sa liste, i tako dalje.

```
<font face="Comic Sans MS,Arial,Courier">
```

Evo primera nekoliko popularnih fontova koji se mogu koristiti na svojim stranicama:

Verdana
Arial
Courier
Bedrock
Times New Roman
Comic Sans MS

Kod za prethodni primer je sledeći:

```
<center>
```

```
<font face="Verdana" size="3">Verdana</font><br>
```

```
<font face="Arial" size="3">Arial</font><br>
```

```
<font face="Courier" size="3">Courier</font><br>
```

```
<font face="Bedrock" size="3">Bedrock</font><br>
```

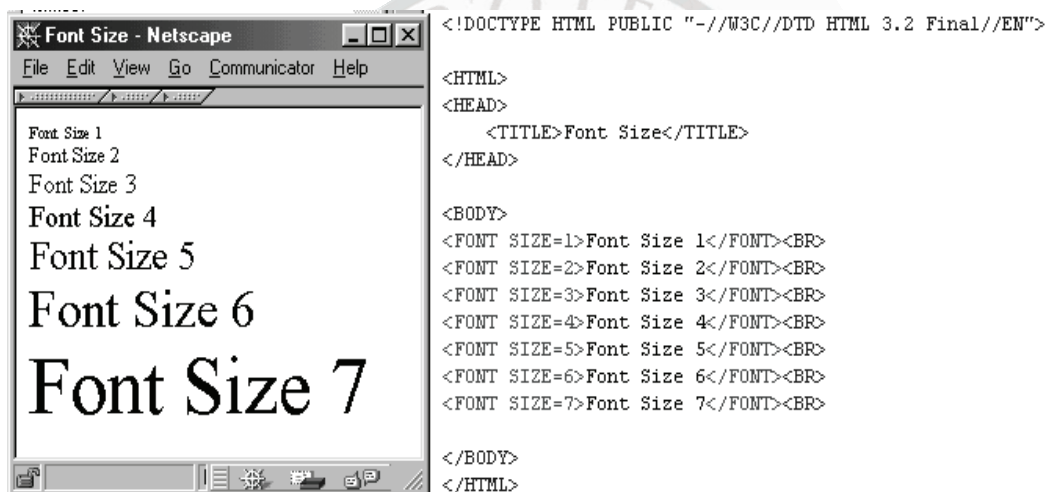
```
<font face="Times New Roman" size="3">Times New Roman</font><br>
```

```
<font face="Comic Sans MS" size="3">Comic Sans MS</font><p>
</center>
```

Pored načina ispisivanja teksta, u okviru stranice može se definisati i veličina slova. Za ove potrebe koristi se **size** atribut. HTML razlikuje 7 veličina slova koje nose vrednosti od 1 do 7. Podrazumevana veličina je 3. Razmera slova je data:

size 1, size 2, size 3, size 4, size 5, size 6, size 7

Primer HTML stranice i koda prikazane stranice:



Poslednja osobina koja se može menjati je boja slova u okviru HTML stranice. Za te potrebe koristi se atribut **color**. Korišćenje boja u okviru HTML stranice se razmatra u posebnom poglavlju, pa i za ovaj atribut važe ista pravila kao i kod definisanja boja <**body**> tagom. Ako se želi dobiti sledeći tekst ispisan crvenom bojom:

Ovde ide vas tekst.

izvršava se sledeći deo koda:

```
<font color="#ff0000"> Ovde ide vas tekst. </font>
```

Pored **** taga HTML poseduje još mogućnosti za obradu teksta. U HTML dokumentima postoji mogućnost pisanja podebljanim (bold ili strong) slovima i kurzivom (italic), kao i mogućnost podvlačenja (underline) i precrtavanja (strike) teksta. Sledeća tabela prikazuje odgovarajuće tagove kao i rezultate njihove primene:

Bold	<code>Bold</code>
Strong	<code>Strong</code>
<i>Italic</i>	<code><i>Italic</i></code>
<u>Underline</u>	<code><u>Underline</u></code>
Strike	<code><strike>Strike</strike></code>

Osim toga mogu se pisati i slova u indeksu ili eksponentu. Indeks se dodaje pomoću taga **<sub>** a eksponent pomoću taga **<sup>**. Sledeća tabela prikazuje primer njihove primene:

H ₂ O	<code>H<sub>2</sub>O</code>
2 ³ =8	<code>2<sup>3</sup>=8</code>

Novi red

Napomenuto je da se kod HTML stranice može pisati u bilo kom tekst editoru. Tekst ispisan u editoru neće se uvek preslikati na HTML stranu na način kako se pojavljuje u editoru. Primeri za ove specijalne slučajeve su novi red, tab i blanko znaci. Znači, novi red (engl. *break*) dobijen u tekst editoru neće se prikazati na HTML stranici, već će tekst biti ispisan u jednom redu. Ako se želi novi red mora se koristiti tag **
. Ovo je prost tag, ne postoji kraj taga **</BR>.

Tekst se u browseru automatski lomi na kraju reda. Da bi se postiglo da neki deo teksta bude u jednom redu koriste se tagovi **<nobr>** i **</nobr>** i umetanjem dela teksta koji se ne želi prelomiti između ovih tagova.

Primer:

ovde je ispisan tekst koji se nece prelomiti na kraju reda

dobija se pomoću sledećeg dela koda

`<nobr>` ovde je ispisan tekst koji se nece prelomiti na kraju reda `</nobr>`

Ukoliko je tekst prevelik da stane u jedan red u okviru ekrana, na dnu prozora će se pojaviti horizontalni "scrollbar" i korisnik će morati da skroluje da bi ga pročitao. Zato treba biti obazriv sa korišćenjem ovog taga. Ukoliko se želi da se red prelomi, ali na tačno određenom mestu, to se može postići korišćenjem `<wbr>` taga. Ovaj tag ne zahteva završni tag. Za razliku od `<br>` taga koji će obavezno prelomiti red, ovaj tag će prelomiti red samo ako je to neophodno tj. ako je tekst pre njega predugačak da bi stao u jedan red. Ovaj tag takođe omogućava da se prelomi neka dugačka reč na kraju reda.

Specijalni znaci

U tekst editorima je moguće koristiti i simbole TAB (služi za prikazivanje određenog broja praznih mesta) i blanko rezmake, ali se na HTML stranici neće videti njihov efekat. Ako se želi blanko razmak u HTML stranici koristi se naredbe ** **. Pomoću ove naredbe definiše se jedno blanko mesto. Ako se želi 8 blanko razmaka jednostavno se napiše osam ovakvih naredbi jedna za drugom odvojenih tačkom-zarezom: `       `

Odgovarajući HTML kod izgleda ovako:

Primer:

```
XXXX XXXX XXXX XXXX XXXX XXXX XXXXX XXXX XXXXX XXXX XXXXX X
XXXXX X XXXXXX XXX XXXX XX XXXXX XX X XXXXX XXXXXX XX XXXXXXX X X
XXXXXXXXX XXXX XXXXXX XXXXX XXXX XXXXXX XXXXXXXXXXXXXXX XXX XXXX XX
XXXXXXXXX X X XXXXXXXXXXXXXXX XXXXX.
```

se dobija pomoću sledećeg koda:

`<p>`

Tag <HR> crta tanku sivu liniju preko cele stranice. I ovaj tag može sadržati odgovarajuće atribute kojima možete regulisati izgled linije. Ako se ne želi da linija bude zasenčena, već potpuno crna potrebno je koristiti atribut **noshade**. Ako se želi promeniti dimenzije linije, koriste se sledeća dva atributa: **size** i **width**. Atributom **size** određuje se debljina linije u pikselima, a atributom **width** određuje se dužina linije ili u pikselima ili u procentima širine stranice. Ovaj tag takođe može sadržati **align** atribut kome se može dodeliti ista vrednosti kao i za pasus. Boja linije se može regulisati pomoću **color** atributa na već opisani način.

Na primer, ako se želi da se nacrti nezasenčena crvena linija koja će biti centrirana i čija će debljina iznositi 6 piksela a protezaće se preko 50% širine stranice, potrebno je izvršiti sledeći kod:

```
<hr noshade align="center" size="6" width="50%" color="#ff0000">
```

Tako će se dobiti sledeća linija:



Treba naglasiti da Netscape Navigator ne podržava atribut **color** za horizontalne linije, tako da će u Navigatoru ova linija biti siva.

Primeri za rad u laboratoriji

Primer 1. Ispisivanje naslova različitih veličina.

```
<HTML>
  <HEAD><TITLE>Naslovi</TITLE></HEAD>
  <BODY>
    <H1> Naslov H1 </H1>
    <H2> Naslov H2 </H2>
    <H3> Naslov H3 </H3>
    <H4> Naslov H4 </H4>
    <H5> Naslov H5 </H5>
    <H6> Naslov H6 </H6>
  </BODY>
</HTML>
```

Primer 2. Primeri osnovnih tagova unutar HTML stranice.

```
<HTML>
  <HEAD><TITLE>Primeri osnovnih tagova</TITLE></HEAD>
  <BODY>
    <H2><P ALIGN=CENTER> HTML</P> </H2>
    <HR>
    <P ALIGN=LEFT>HTML (HyperText Markup Language) je
veoma jednostavan jezik cije osnove svako može da se brzo savlada. U
okviru ovog kursa proučavaju se osnove HTML-a koje su dovoljne da se
naprave jednostavne ali sasvim zadovoljavajuće prezentacije. </P>
    <HR>
    <P ALIGN=RIGHT>Svaki put kada krstareći internetom naletite
na neku zanimljivu stranicu , a ne znate kako je napravljena, njen izvorni
HTML kod možete pogledati ako u liniji menija odaberete: View -> Source
(ili View -> Page Source u Netscape Navigatoru).</P>
    <HR>
    <P ALIGN=CENTER>HTML se koristi i u okviru<BR>
ASP stranica<BR>
JSP stranica, <BR> Zajedno
sa JavaScriptom <BR> ili PHP jezikom <BR>
ili nekim drugim script jezikom</P>
```

<HR>

<P ALIGN=CENTER>HTMLje

<DIV ALIGN=LEFT>hiper-</DIV>

<DIV ALIGN=CENTER>tekst-</DIV>

<DIV ALIGN=RIGHT>Markup jezik.</DIV>

Kad savladate osnove HTML-a mozete preci na upoznavanje
JavaScripta </P>

</BODY>

</HTML>

Primer 3. Primer paragrafa.

<html>

<head>

<TITLE> HTML Primer</TITLE>

</head>

<body>

<H1>HTML Uvod</H1>

<P>Ovo je primer prvog paragrafa.

Koji se radi na prvom casu.

</P>

<P>Drugi paragraf.</P>

<P ALIGN=CENTER>

Ovo je centrirani paragraf

</P>

</body>

</html>

Primer 4. Rad sa fontovima.

<html>

<head><title> Rad sa fontovima</title>

</head>

<body>

<P>

The library

has an arrangement with the

```
<FONT FACE="Arial, Helvetica, Geneva" SIZE="+1"><A  
HREF="cafeteria.html">cafeteria</A></FONT>
```

to provide refreshments during the weekly

```
<FONT FACE="Arial, Helvetica, Geneva" SIZE="+1"><A  
HREF="coffeehouse.html">coffee
```

house lectures.

```
</P>
```

```
</body>
```

```
</html>
```

Primer 5. Rad sa prelomom linije.

```
<html>
```

```
<head><title> Rad sa prelomom linije </title>
```

```
</head>
```

```
<body>
```

```
<nobr>
```

```
<h1>Come Scroll with me, away to the right, as I list  
out all of the long answers to the  
meaning of life accessible only to those that can scroll,  
scroll, scroll...</h1>
```

```
</nobr>
```

```
</body>
```

```
</html>
```



3. LISTE

Često je okviru teksta potrebno navesti nekoliko stavka po određenom redosledu. HTML omogućava definisanje više vrsta lista, kojima se može postići željeni efekat.

U opštem slučaju liste možemo podeliti u tri grupe:

- nenumerisane,
- numerisane i
- definicione liste.

Za sve tri grupe lista zajedničko je da se moraju definisati stavke liste i način prikazivanja pojedinačnih stavki.

Numerisane i nenumerisane liste

Osnovna razlika između ove dve vrste listi je ta što se kod numerisanih lista ispred pojedinačnih stavki liste nalaze redni brojevi ili slova, a kod nenumerisanih lista pojavljuju se specijalni simboli.

Za obe liste važi da se pojedine stavke liste definišu pomoću `<li>` taga (ovaj tag ne zahteva završni tag). Numerisane liste koriste `<ol>` taga, a nenumerisane liste `<ul>` taga.

Nenumerisane liste su liste sa kojima se nabrajaju stavke, bez navođenja rednog broja.

Opšta struktura za ovu vrstu lista je (UL – unordered list):

```
<UL>
  <LI> prva stavka ...
  <LI> druga stavka ...
  ...
  <LI> poslednja stavka ...
</UL>
```

Na primer:

- jabuke
- banane
- limun

Kod za prethodni primer je:

```
<UL>
```

```
<LI> jabuke
<LI> banane
<LI> limun
</UL>
```

Simbol ispred stavke liste može se definisati i na drugi oblik. Koristi se atribut **type** koji može uzimati vrednosti:

- circle - krug
- disc – pun krug
- square - kvadrat

Primer sa kvadratima:

- jabuke
- banane
- limun

a odgovarajući HTML kod je:

```
<ul type="square">
  <LI> jabuke
  <LI> banane
  <LI> limun
</ul>
```

Numerisane liste su liste sa kojima se nabrajaju stavke sa navođenjem rednog broja ili slova.

Opšta struktura ove vrste liste je (OL - ordered list):

```
<OL>
  <LI> prva stavka ...
  <LI> druga stavka ...
  ...
  <LI> poslednja stavka ...
</OL>
```

Primer numerisanih lista:

1. jabuke
2. banane
3. limun

koja se dobija sledećim HTML kodom:

```
<ol>
  <LI> jabuke
  <LI> banane
  <LI> limun
</ol>
```

I u ovom slučaju se mogu promeniti default arapski brojevi koji se pojavljuju ispred stavki liste. Opet se koristi atribut **type** i u ovom slučaju može imati sledeće vrednosti:

- A - velika slova
- a - mala slova
- I - rimski brojevi
- i - mali rimski brojevi

Prethodni primer sa velikim rimskim brojevima:

- I. jabuke
- II. banane
- III. limun

a odgovarajući HTML kod:

```
<ol type="I">
  <LI> jabuke
  <LI> banane
  <LI> limun
</ol>
```

Definicione liste

Ova vrsta listi ima donekle drugačiji izgled od prethodnih. Sada se pored same stavke liste može definisati i pojam koji se navodi ispred stavke. Ime definicione liste (engl. *definition list*) je dobijeno jer se svaki član liste sastoji iz dva dela: pojma koji se definiše i njegove definicije. Pojmovi se poravnavaju uz levu marginu, a njihove definicije se pojavljuju u novom redu i uvučene su za određen broj mesta.

Za formiranje ove vrste lista koristi se tag `<dl>`, dok se unutar člana liste pojmovi definišu pomoću `<dt>` taga, a njihove definicije `<dd>` tagom.

Opšta stuktura ove vrste liste je:

```
<DL>
    <DT> prvi pojam <DD> definicija prvog pojma
    <DT> drugi pojam <DD> definicija drugog pojma
    . . . . .
    <DT> poslednji pojam <DD> definicija poslednjeg pojma
</DL>
```

Primer za ovu vrstu listi je:

Prolece:

traje od 21. marta do 21. juna.

Leto:

traje od 21. juna do 21. septembra.

Jesen:

traje od 21. septembra do 21. decembra.

Zima:

traje od 21. decembra do 21. marta.

HTML kod za prethodni primer je:

```
<dl>

<dt>Prolece: <dd>traje od 21. marta do 21. juna.
<dt>Leto: <dd>traje od 21. juna do 21. septembra.
<dt>Jesen:<dd>traje od 21. septembra do 21. decembra.
<dt>Zima:<dd>traje od 21. decembra do 21. marta.

</dl>
```

Još jedan primer:

```
<DL>
    <DT> -numerisane liste
        <DD> su liste sa kojima se nabrajaju stavke sa navođenjem
rednog broja ili slova.
    <DT> -nenumersane liste
        <DD> su liste sa kojima se nabrajaju stavke bez navođenja
rednog broja.
```

</DL>

Rezultat koji se dobija izvršavanjem prethodnog primera:

- numerisane liste

su liste sa kojima se nabrajaju stavke sa navođenjem rednog broja ili slova.

- nenumerisane liste

su liste sa kojima se nabrajaju stavke bez navođenja rednog broja.

Gnežđenje lista

Svi primeri koji su do sada razmatrani su podrazumevali daje stvka liste običan tekst, što u opštem slučaju ne mora da bude. Stavka liste može da bude bilo koji HTML element, pa i neka druga lista. Ako je to slučaj dolazi do gnežđenja lista, t.j. u okviru jedne liste pojavljuje se druga nova lista. U sledećem primer definisana je nenumerisana lista, u okviru koje prva stavka je neka nova nenumerisana lista, a druga stavka je numerisana liste:

 Neke vrste voca su:

 jabuke

 banane

 limun

 Neke vrste povrca su:

 sargarepa

 kupus

Rezultat prethodnog primera:

- Neke vrste voca su:

- o jabuke

- o banane

- o limun

- Neke vrste povrca su:

- 1. sargarepa

- 2. kupus

Primeri za rad u laboratoriji

Primer 1. Primer opisne liste

```
<HTML>
<HEAD>
  <TITLE>Opisne liste</TITLE>
</HEAD>
<BODY>
<DL>
  <DT>piko-
  <DD> prefiks nekoj jedinici koji pokazuje milioniti deo
    milionitog dela te jedinice;
  <DT>nano-
  <DD> prefiks nekoj jedinici koji pokazuje milijarditi
    deo te jedinice;
  <DT>mikro-
  <DD>prefiks nekoj jedinici koji pokazuje milioniti deo
    te jedinice;
</DL>
</BODY>
</HTML>
```

Primer 2. Primer opisne liste sa formatiranom odrednicom

```
<HTML>
<HEAD>
  <TITLE>Formatirane opisne liste</TITLE>
</HEAD>
<BODY>
<DL>
  <DT><B>piko-</B>
  <DD> <I>prefiks nekoj jedinici koji pokazuje milioniti deo
    milionitog dela te jedinice</I>;
  <DT><B>nano-</B>
  <DD> <I>prefiks nekoj jedinici koji pokazuje milijarditi
    deo te jedinice</I>;
  <DT><B>mikro-</B>
```

```
<DD><I>prefiks nekoj jedinici koji pokazuje milioniti deo  
te jedinice</I>;  
</DL>  
</BODY>  
</HTML>
```

Primer 3. Primer numerisane liste

```
<HTML>  
<HEAD>  
  <TITLE>Numerisane liste</TITLE>  
</HEAD>  
<BODY>  
<OL>Neki predmeti na 1. godini FPI  
  <LI>Poslovna informatika  
  <LI>Sociologija  
  <LI>Osnovi ekonomije  
  <LI>Osnovi prava  
</OL>  
</BODY>  
</HTML>
```

Primer 4. Primer nenumerisane liste

```
<HTML>  
<HEAD>  
  <TITLE>Nenumerisane liste</TITLE>  
</HEAD>  
<BODY>  
  <UL>Tokom zime ima  
    <LI>jabuka  
    <LI>limuna  
    <LI>banana.  
  </UL>  
</BODY>  
</HTML>
```

Primer 5. Primer numerisane liste

```
<html>  
<body>
```

```
<h4>Numerisane liste:</h4>
```

```
<ol>  
  <LI>jabuke  
  <LI>limun  
  <LI>banane  
  <LI>pomorandze  
</ol>
```

```
<h4>Obelezavanje slovima:</h4>
```

```
<ol type="A">  
  <LI>jabuke  
  <LI>limun  
  <LI>banane  
  <LI>pomorandze  
</ol>
```

```
<h4>Obelezavanje malim slovima:</h4>
```

```
<ol type="a">  
  <LI>jabuke  
  <LI>limun  
  <LI>banane  
  <LI>pomorandze  
</ol>
```

```
<h4>Obelezavanje rimskim brojevima:</h4>
```

```
<ol type="I">  
  <LI>jabuke  
  <LI>limun  
  <LI>banane  
  <LI>pomorandze  
</ol>
```

```
<h4>Obelezavanje malim rimskim brojevima:</h4>
```

```
<ol type="i">
```

```
<LI>jabuke  
<LI>limun  
<LI>banane  
<LI>pomorandze  
</ol>  
</body>  
</html>
```

Primer 6. Primer nenumerisane liste

```
<html>  
<body>  
  
<h4>Obelezavanje pomocu diska:</h4>  
<ul type="disc">  
  <LI>jabuke  
  <LI>limun  
  <LI>banane  
  <LI>pomorandze  
</ul>  
  
<h4>Obelezavanje krugom:</h4>  
<ul type="circle">  
  <LI>jabuke  
  <LI>limun  
  <LI>banane  
  <LI>pomorandze  
</ul>  
  
<h4>Obelezavanje kvadratom:</h4>  
<ul type="square">  
  <LI>jabuke  
  <LI>limun  
  <LI>banane  
  <LI>pomorandze  
</ul>  
  
</body>  
</html>
```

Primer 7. Primer liste unutar liste

```
<html>
<body>

<h4>Lista unutar liste:</h4>
<ul>
  <li>Kafa</li>
  <li>Caj
    <ul>
      <li>Crni caj</li>
      <li>Zeleni caj</li>
    </ul>
  </li>
  <li>Mleko</li>
</ul>

</body>
</html>
```

Primer 8. Primer definicione liste

```
<html>
<body>

<h4> Definiciona lista:</h4>
<dl>
  <dt>Kafa</dt>
  <dd>Crno vruce pice</dd>
  <dt>Mleko</dt>
  <dd>Belo hladno pice</dd>
</dl>

</body>
</html>
```

4. BOJE I SLIKE

Boje su sastavni deo svake dobro dizajnirane stranice. Zato HTML podržava brz i jednostavan rad sa paletom boja, pa je moguće promeniti boju pozadine, slova, linkova, ... Svi elementi stranice rad sa bojama organizuju identično, i to na dva načina ili preko prreredefinisanoeg engleskog naziva boje (npr. "yellow" za žutu boju), ili preko heksadecimalne RGB vrednosti ispred koje obavezno treba staviti simbol # (npr. #ffff00). Ako se koristi RGB paleta boja vrednost određenog atributa za boje ima opšti oblik "#cczzpp" gde su cc, zz i pp heksadecimalne vrednosti za intenzitet crvene, zelene i plave boje. Svaka od ovih vrednosti može biti broj između 0 i 255 (u dekadnom brojnom sistemu) ili između 0 i FF (u heksadecimalnom brojnom sistemu). Boje se mogu odrediti navođenjem naziva boje na engleskom umesto kombinacijom heksadecimalnih cifara prema tabeli koju je data od strane SGIR (Service Général Informatique et réseaux, Université René Descartes, Paris). U prilogu je data paleta boja, gde su tabelarno prikazane najvažnije boje, njihov engleski naziv i RGB kod.

BODY tag

U prethodnim poglavljima bilo je reči o BODY tagu i rečeno je da ovaj tag definiše samo telo stranice. Globalne promene boja elemenata stranice se postižu upravo pomoću ovog taga, pa se pozadina i tekst HTML stranice mogu pojaviti u čitavom spektru boja.

Boja pozadine se određuje atributom *bgcolor* u okviru <BODY> taga. U slučaju da se prilikom navođenja taga BODY izostave opisani atributi, čitač im dodeljuje predefinisane vrednosti. Na primer, promena boje pozadine u žutu dobija se na sledeći način:

```
<body bgcolor="#ffff00">
```

ili:

```
<body bgcolor="yellow">
```

Oba primera koda će dati isti rezultat.

Pored pozadine, može se menjati i boja teksta, pomoću atributa *text*. Na primer, ako je potrebno na prethodnoj stranici definisati zelena slova treba napisati sledeći kod:

```
<body bgcolor="#ffff00" text="#008000">
```

ili

```
<body bgcolor="#ffff00" text="green">
```

ili

```
<body bgcolor="yellow" text="#008000">
```

ili

```
<body bgcolor="yellow" text="green">
```

Na ovaj način definiše se osnovna boja slova stranice, a ako se želi da samo jedan deo stranice sadrži slova u zelenoj boji mora se koristiti, ranije objašnjeni, atribut *color* taga .

Pored promene boje pozadine ili slova stranice efektna promena se može dobiti i definisanjem određene slike kao pozadine. Da bi se postigao ovaj efekat, prvo treba odabrati sliku u nekom od podesnih elektronskih formata. Treba voditi računa o veličini odabranog fajla, jer se radi o Internet aplikaciji čije ograničenje je pristup i izvršavanje na udaljenom računaru. Kao najpodesniji formati smatraju se JPEG i GIF format, zbog stepena kompresije originalnog fajla. Pored veličine slike, sledeća stva o kojoj treba voditi računa je lokacija fajla sa slikom u okviru aplikacije. Najjednostavnije rešenje je da se nalazi u istom direktorijumu gde se nalazi i aplikacija. Druge mogućnosti razmatraće se u narednim odeljcima kada se bude diskutovalo o apsolutnoj i relativnoj putanji. Kao i promena boje, pozadina u obliku slike se definiše pomoću odgovarajućeg atributa u BODY tagu. Atribut je u ovom slučaju *background*.

Na primer, ako se želi da se kao pozadina stavi slika "pozadinal.jpg" tada će odgovarajući tag izgledati:

```
<body background="pozadinal.jpg">
```

Pri definisanju pozadine u obliku slike, dobra praksa je da se definiše i *bgcolor* atribut. Razlog tome je mogućnost da korisnik u okviru svog Web čitača isključi automatsko učitavanje slike. Tada se definisana slika neće pojaviti u okviru stranice. Može se desiti i slučaj da slika predstavlja neku tamniju površinu, pa se i boja slova menja u neku svetliju, pa čak i belu. U tom slučaju kada se stranica učitava bez definisane slike, dobijaju se bela slova na beloj pozadini, pa je tekst stranice za korisnika nevidljiv. Da bi se izbegli ovakvi slučajevi BODY dozvoljava definisanje i *background* i *bgcolor* atributa u okviru istog taga. Naravno, upotreba oba atributa ne znači da će se u isto vreme prikazati i definisana slika i boja pozadine, već atribut *bgcolor*

definiše boju pozadine u slučaju da se slika ne učita. Znači čitač u ovakvim slučajevima BODY taga uvek prvo pokušava da prikaže sliku kao pozadinu, pa tek kada ne uspe prikazuje pozadinu u traženoj boji. Prethodni primer sa primenjenom napomenom bi izgledao:

```
<body background="pozadina1.jpg" bgcolor="#0000ff">
```

gde se u slučaju da se ne učita u okviru stranice slika pozadina1.jpg pozadina stranice prikazuje u plavoj boji.

U narednom poglavlju se razmatraju linkovi, odnosno hiperveze, kao najbitni elementi HTML stranice. Linkovi u HTML stranici omogućavaju da se jednim klikom miša promeni navigacija i otvori neka druga stranica. Pri radu sa linkovima, postoje tri default boje koje definišu stanje linka, pa se boje linkova mogu definisati pomoću tri atributa: *link*, *vlink* i *alink*, gde *link* definiše početnu boju linka u okviru stranice, *vlink* definiše boju linka u okviru stranice koji je posećen, a *alink* definiše boju aktivnog linka u okviru stranice. Ako navedeni atributi nisu korišćeni stranica prikazuje default vrednosti. U Internet Exploreru pozadina je po defaultu bela, tekst crn, linkovi plavi, posećeni linkovi maslinasto-zeleni, a aktivni linkovi crveni.

Na primer, definisanje boje linkova u plavo, posećenih linkova u ljubičasto, a aktivnih linkova u crveno se obavlja na sledeći način:

```
<body link="#0000ff" vlink="#ee82ee" alink="#ff0000">
```

IMG Tag

Pored pozadine slika se može u okviru HTML stranice prikazati na proizvoljnom mestu. U JPEG formatu se čuvaju kvalitetnije slike, jer ovaj format podržava 16 miliona boja, a fajlovi su efikasno komprimovani, tako da se slike relativno brzo učitavaju. Sa druge strane GIF format ima samo 256 boja, ali on ima tu prednost da mu se može zadati transparentnost i da se može animirati.

U okviru HTML stranice slika se prikazuje pomoću taga. Ovaj tag mora imati bar jedan atribut. To je *src* atribut koji definiše naziv, i eventualno lokaciju, grafičkog fajla koji se želi prikazati u okviru stranice. Pri opisu lokacije dokumenta najjednostavniji slučaj je da se dokument slike nalazi u istom direktorijumu gde i HTML stranica. Tada je dovoljno da kao vrednost *src* atributa postaviti samo naziv odgovarajućeg dokumenta.

Na primer da bi se prikazala slika sa imenom "osam.gif" na HTML stranici tada će odgovarajući tag glasiti:

na stranici se prikazuje:



Tag IMG može imati različite atribute koji omogućavaju da se precizno opišu položaj, dimenzije i odnos slike prema drugim delovima stranice. Ti atributi su:

- atribut za poravnavanje slike u odnosu na margine ALIGN ... može imati vrednosti
 - o za vertikalno poravnavanje: TOP, MIDDLE, BOTTOM
 - o za horizontalno poravnavanje: LEFT, CENTER, RIGHT
- atributi za dimenzionisanje slike su WIDTH (*širina*) i HEIGHT (*visina*).
- atributi koji opisuju položaj slike su HSPACE i VSPACE
- atribut koji opisuje širinu okvira slike BORDER
- atribut koji na mestu neučitane slike prikazuje komentar ALT

Ako se nijedan od atributa ne navede slike su poravnate sa tekstom na donju ivicu.

ALIGN atribut

Položaj slike u odnosu na tekst stranice se može definisati ubacivanjem align atributa u tagu. Vrednosti ovog atributa i odgovarajuće efekte mogu se videti iz sledećih primera:

left - postavlja sliku uz levu marginu:



XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX X XXXX XXXXX
XXXXXXXX XXXXXX XXXXXX XXXXXX XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX
XXXXXXXX XXXXXXXX XXX XXXXXXXX XXXXXX XXXXXXXX XXXXXXXX XXX XX XXXX
XXXXXXXX XXXXXXXX XXX XXXXXXXX X XXXXXXX XXXXXXX XXXXXX XXXXXXXX XXXXXXXX
XXXXXXXX XXXXXXXX XXXX XXXXXXXX X XXXXXXX XXXX XXXXX XXXX XXXXXXX
XXXXXXXX XXXX XXXXXXX XXXXXXXX XXXXXXX XXXXXXX XXXX XXXXXXXX XXXXX
XXXXXXXX X XXXXX XXXXXXX X XXXXX

Odgovarajući tag glasi:

```

```

right - postavlja sliku uz desnu marginu:

```
XXXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX X XXXX XXXXX XXXXXXXX
XXXXXX XXXXX  XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX
XXXXXXXXX XXX XXXXXXXX XXXXX XXXXXXXX XXXXXXXX XXX XX XXXX XXXXXXXX
XXXXXXXX XXX XXXXXXXX X XXXXXXX XXXXXXX XXXXX XXXXXXXXXX XXXXXXXX XXXXXXXX
XXXXXXXX XXXX XXXXXXXX X XXXXXXXXXX XXXX XXXXX XXXX XXXXXXX XXXXXXX XXXX
XXXXXXXX XXXXXXXXXX XXXXXXXX XXXXXXXX XXXX XXXXXXXXXX XXXXX XXXXXXX X
XXXXXXXX XXXXXXX X XXXXX
```

Odgovarajući tag glasi:

```

```

top - poravnava sliku sa vrhom slova u tekućoj liniji:

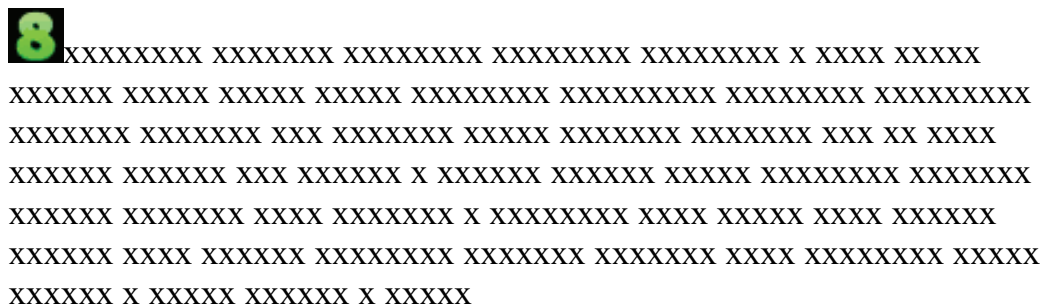
```
 XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX X XXXX XXXXX
XXXXXXXX XXXXX XXXXX XXXXX XXXXXXX XXXXXXXXXX XXXXXXXX
XXXXXXXXXXX XXXXXXX XXXXXXXXXX XXX XXXXXXXXXX XXXXX XXXXXXX XXXXXXX XXX
XX XXXX XXXXXXX XXXXXXX XXX XXXXXXX X XXXXXXX XXXXXXX XXXXX XXXXXXXXXX
XXXXXXXX XXXXXXX XXXXXXXXXX XXXX XXXXXXXXXX X XXXXXXXXXX XXXX XXXXX XXXX
XXXXXXXX XXXXXXX XXXX XXXXXXX XXXXXXXXXX XXXXXXX XXXXXXX XXXX XXXXXXXXXX
XXXXXXXX XXXXXXX X XXXXX XXXXXXX X XXXXX
```

Odgovarajući tag glasi:

```

```

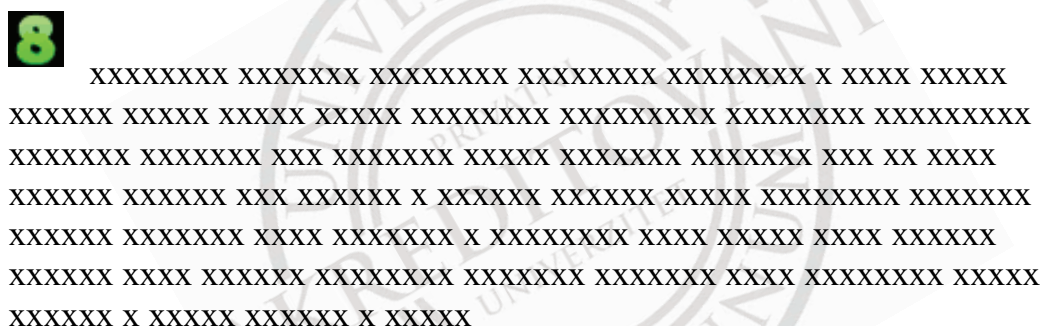
bottom - poravnava sliku sa donjom ivicom slova:



Odgovarajući tag glasi:



middle - postavlja sliku tako je donja ivica slova na sredini slike:

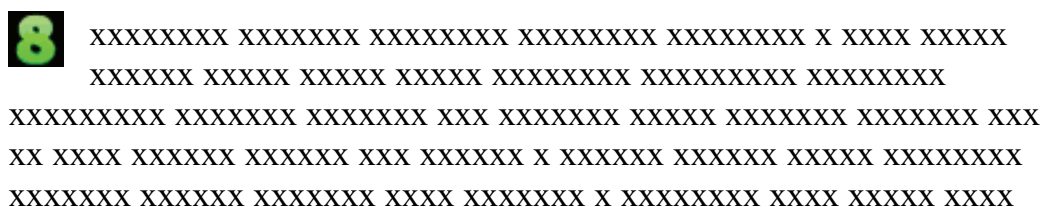


Odgovarajući tag glasi:

```

```

absmiddle - postavlja sliku tako da se sredina slike i sredina slova poklapaju:



```

XXXXXXXX XXXXXX XXXX XXXXXX XXXXXXXXXXX XXXXXXX XXXXXXX XXXX XXXXXXXX
XXXXXXXX XXXXXX X XXXXX XXXXXXX X XXXXX

```

Odgovarajući tag glasi:

```

```

CENTER

Da bi se slika prikazala na sredini stranice potrebno je koristiti tag center.

Na primer:

```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXX X XXXXXXXX
XXXX XXXXX XXXX XXXXXXX XXXXXXX XXXX XXXXXXX XXXXXXXXXXX XXXXXXX
XXXXXXXX XXXX XXXXXXXXXXX XXXXX XXXXXXX XXXXXXX XXXXXXXXXXX XXXXXXXX

```



```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXX X XXXXXXXX
XXXX XXXXX XXXX XXXXXXX XXXXXXX XXXX XXXXXXX XXXXXXXXXXX XXXXXXX
XXXXXXXX XXXX XXXXXXXXXXX XXXXX XXXXXXX XX XXXXXXX XXXX XXXXX XXXX X

```

Gornji primer se postiže sledećim HTML kodom:

```
<center></center>
```

HEIGHT i WIDTH atributi

Da bi se promenila originalna veličina slike, željena veličina se može definisati pomoću dva atributa: *height* i *width*.

Vrednost širine i visine stranice se može zadati ili u pikselima ili u procentima. Da bi slika zauzimala 50% širine prozora i 30% visine prozora HTML stranice, potrebno je napisati sledeći kod:

```

```

Slika koja će se prikazati na stranici je:



Ako se definiše samo *height* ili samo *width* atribut, druga dimenzija će biti uvećana srazmerno sa definisanom karakteristikom tako da će proporcije slika ostati iste.

HSPACE i VSPACE atributi

Prazan prostor između slike i okolnog teksta, ili nekih drugih elemenata stranice, može se definisati pomoću dva atributa: *hspace* i *vspace*. Vrednost ova dva atributa se zadaje u pikselima.

Izgled dela stranice:

8

```
X XXXX XXXXX XXXXXX XXXXX XXXXX XXXXX XXXXXXXX XXXXXXXXX
XXXXXXXX XXXXXXXXX XXXXXXX XXXXXXX XXX XXXXXXX XXXXX
XXXXXXXX XXXXXXX XXX XX XXXX XXXXXXX XXXXXXX XXX XXXXXXX X
XXXXXXXX XXXXXXX XXXXX XXXXXXXXX XXXXXXX XXXXXXX XXXXXXX
XXXX XXXXXXX X XXXXXXXX XXXX XXXXX XXXX XXXXXXX XXXXXXX
XXXX XXXXXXX XXXXXXXX XXXXXXX XXXXXXX XXXX XXXXXXX XXXXX XXXXXXX X
XXXXXXXX XXXXXXX X XXXXX
```

se može dobiti pomoću koda

```

```

Na ovaj način je širina margine oko slike 50 piksela.

BORDER atribut

U okviru stranice može se definisati i debljina granice pomoću atributa *border* čija se vrednost zadaje u pikselima.

Da bi se definisala debljina okvira 5 piksela potrebno je napisati sledeći tag:

```

```

ALT

Postoji još jedan koristan atribut `<IMG>` taga. To je *alt* atribut. Ovaj atribut će u slučaju da čitač korisnika stranice iz bilo kog razloga ne učitava sliku, na onom mestu gde bi trebalo da bude slika prikazati tekst koji predstavlja vrednost ovog atributa. Ovo može biti korisno naročito ako je slika postavljena kao link, jer će onda i u slučaju neučitavanja slike korisnik znati gde vodi taj link.

Primer korišćenja ovog atributa je:

```

```

Ukoliko browser korisnika ne učitava sliku `osam.gif` korisniku će se na definisanom mestu pojaviti tekst "Broj osam".

Primeri za rad u laboratoriji

Primer 1. Boje pozadine i slova

```
<HTML>
<HEAD>
  <TITLE> Boje pozadine i slova</TITLE>
</HEAD>
<BODY BGCOLOR="#FFFF00" TEXT="#FF0000">
<H1>Ovaj naslov je crven zbog atributa TEXT</H1>
<BR><BR>
<H1><FONT COLOR="#0000FF">
  a ovaj naslov je plav zbog etikete FONT
</FONT>
</H1>
</BODY>
</HTML>
```

Primer 2. Boje atributa u tagu BODY

```
<HEAD>
  <TITLE> Atributi u etiketi BODY</TITLE>
</HEAD>
<BODY TEXT="#000000" BGCOLOR="#FFFFFF" LINK="#990000"
VLINK="#660000" ALINK="#ff0000">
<H1>Iz teksta na beloј pozadini sa crnim slovima<BR>
  poziva se <a href="ex0901.html">ovde</a> tekst<BR>
  iz prethodnog primera. <BR><BR>

<UL>
<LI><Font COLOR="#990000"> Boja za atribut LINK</FONT>
<LI><Font COLOR="#660000"> Boja za atribut VLINK</FONT>
<LI><Font COLOR="#FF0000"> Boja za atribut ALINK</FONT>
</UL>
</H1>
</BODY>
</HTML>
```

Primer 3. Definisanje pozadine slikom.

```
<HTML>
<HEAD>
  <TITLE> Slika kao pozadina</TITLE>
</HEAD>
<BODY BGCOLOR="#FFFF00" TEXT="#FF0000"
  BACKGROUND="pozadina2.gif">
<H1>Ovaj naslov je crven zbog atributa TEXT</H1>
<BR><BR>
<H1><FONT COLOR="#0000FF">
  a ovaj naslov je plav zbog etikete FONT
</FONT>
</H1>
</BODY>
</HTML>
```

Primer 4. Primer upotrebe slike u okviru HTML stranice.

```
<html>
<head>
<title>Slika u okviru HTMLa</title>
</head>

<body>
<h1>Pronadjite sliku na vasem racunaru i njeno ime upisite u donji
tag</h1>


</body>
</html>
```



5. LINKOVI

HTML je potreban da bi se na jednostavniji način realizovali kako sami hiperdokumenti, tako i veze između njih. Veze između različitih stranica se nazivaju linkovi ili hiperveze. One omogućavaju jednostavan prelazak sa jednog mesta na stranici na drugo mesto unutar same stranice ili na sasvim novoj stranici. U daljem tekstu će se koristiti termini polazna pozicija, za mesto odakle se prelazi, i krajnja pozicija, za mesto gde se dolazi.

Linkovi koji se mogu definisati u okviru jedne HTML stranice mogu se podeliti na tri grupe:

- krajnja pozicija se nalazi na stranici koja je u okviru iste aplikacije i nalazi se na istom serveru gde i stranica sa polaznom pozicijom,
- krajnja pozicija se nalazi na stranici koja ne pripada istoj aplikaciji, već se nalazi na drugom serveru,
- krajnja pozicija pripada istoj stranici kao i polazna pozicija

Za sve tri vrste linkova jedinstvena je definicija početne pozicije korišćenjem taga <A>. Opšti izgled ovog taga je:

```
<A atribut> ... </A>.
```

Sintaksa ovog taga podrazumeva da se u okviru polazne pozicije opišu:

- fizička pozicija polazne pozicije i
- fizička lokacija krajnje pozicije.

Polazna pozicija označava na stranici mesto sa koga se prelazi na neki drugi deo stranice i definiše se pomoću atributa *href*:

```
<A HREF= adresa krajnje pozicije> polazna pozicija sa koje se prelazi na krajnju </A>
```

Čitač obično prikazuje polaznu poziciju HTML stranice u obliku elementa na koji se može kliknuti mišem, grafički istaknut podvlačenjem i drugom bojom slova od boje slova samog teksta unutar stranice.

KRAJNJA POZICIJA JE NA STRANICI U OKVIRU ISTE APLIKACIJE

Najjednostavniji slučaj rada sa linkovima je prelazak na stranicu u okviru iste aplikacije. Ovaj slučaj se najčešće koristi, jer je to upravo i svrha

upotrebe HTML jezika, na primer linkovi za kretanje napred i nazad kroz aplikaciju.

Aktiviranjem ove vrste linkova napušta se tekuća stranica i dalje izvršavanje se nastavlja na nekoj drugoj stranici tekuće aplikacije. Kod ove vrste linkova koristi se taga <A> sa atributom *href* i nazivom stranice do koje se želi da se napravi veza.

Na primer ako se želi da se napravi veza do stranice Prva.html treba napisati:

```
<a href="Prva.html"> Veza do stranice Prva.html </a>
```

Kada se mišem klikne na tekst "Veza do stranice Prva.html" u čitaču će se prikazati stranica Prosla.html.

Prelazak se izvršava na vrh stranice čija se adresa nalazi u okviru *href* atributa. Ako se želi da se dalji rad nastavi od nekog određenog mesta u okviru druge stranice tada treba kombinovati prvu i treću vrstu linkova, na način koji će u kasnijem tekstu biti razmatran.

Stranica sa krajnjom pozicijom osim što pripada istoj aplikaciji, odnosno serveru, u opštem slučaju ne mora da se i fizički nalazi u istom direktorijumu gde i stranica sa polaznom pozicijom. Na primer želi se da se aplikacija podeli u više logičkih celina i stranice svake celine da se nalaze u različitim direktorijumima. Postoje dva rešenja ovog problema, a to je rad sa apsolutnim i relativnim putanjama.

Korišćenje apsolutnih putanja podrazumeva u okviru *href* atributa navođenje pune putanje do stranice sa krajnjom pozicijom, na primer *href="c:\Aplikacije\Poslovanje\Klijent\Prva.html"*). Ovaj način nije podesan, jer u opštem slučaju u trenutku pisanja koda ne zna se tačno mesto na serveru gde će se aplikacija nalaziti, pa samim tim i putanja do neke stranice.

Korišćenje relativnih putanjaje mnogo podesnije, jer se zasniva na kretanju u odnosu na direktorijum stranice sa polaznom pozicijom. Neka je data sledeća struktura aplikacije koja se nalazi na serveru www.singidunum.ac.yu. Takođe neka direktorijum X sadrži stranice 1 i 2, Y stranicu 3, a Z stranicu 4.



Ako se koriste apsolutne putanje, da bi se došlo do stranice c potrebno je navesti adresu `http://www.singidunum.ac.yu/X/Y/3.html`. Apsolutna adresa je određena jednoznačno pomoću uniformnog lokatora resursa (skr. URL, od engl. Uniform Resource Locator) stranice `3.html`.

Korišćenje relativne putanje je umnogome jednostavnije i razmotriće se nekoliko slučajeva. Na primer:

- o Za stranicu 1,
 - relativna adresa stranice 2 je „2.html”;
 - relativna adresa stranice 3 je „Y/3.html”;
 - relativna adresa stranice 4 je „Y/Z/4.html”;
- o Za stranicu 3,
 - relativna adresa stranice 2 je „../2.html” (simbol .. označava direktorijum iznad direktorijuma gde se nalazi stranica sa polaznom pozicijom);
 - relativna adresa stranice 4 je „Z/4.html”;
- o Za stranicu 4,
 - relativna adresa stranice 2 je „../../2.html”;
 - relativna adresa stranice 3 je „../3.html”.

Datim primerima je ilustrovano kako se postupa, ako se stranica sa krajnjom pozicijom nalazi u direktorijumu ispod ili iznad direktorijuma sa stranicom sa polaznom pozicijom.

KRAJNJA POZICIJA JE NA STRANICI IZVAN ISTE APLIKACIJE

Druga vrstu su linkovi do stranice u nekoj sasvim drugoj prezentaciji koja može biti postavljena na računaru koji se nalazi na nekom sasvim drugom serveru. Da bi se pristupilo toj prezentaciji potrebno je navesti celu web adresu te prezentacije. Opšti oblik adrese koja se pojavljuje kao vrednost *href* atributa je

scheme://server.domen [:port]/putanja/ imeDokumenta

Na primer ako se u okviru prezentacije želi uspostaviti veza sa Univerzitetom Singidunum u Beogradu, čija adresa je `www.singidunum.ac.yu`, tada *href* atribut izgleda:

```
<a href="http://www. singidunum.ac.yu/"> Link ka Univerzitetu Singidunum </a>
```

Na stranici će se prikazati tekst " Link ka Univerzitetu Singidunum " i ako se mišem klikne na njega u čitaču će se pojaviti početna stranica prezentacije Univerziteta Singidunum.

Ovakav način adresiranje se temelji na pojmu URL, koji je omogućio da se precizno navede adresa stranice sa krajnjom pozicijom, ma gde on bio fizički lociran. U opisivanju ove adrese može se desiti više slučajeva u zavisnosti od toga koliki je deo URL-a poznat u tom trenutku.

KRAJNJA POZICIJA JE NA ISTOJ STRANICI

U ovom slučaju je potrebno u okviru iste stranice tačno definisati i krajnju poziciju. Primer može biti da se na kraju stranice napravi link koji korisnika koji ga aktivira vraća na vrh stranice. Za ovu vrstu linkova prvo se napravi oznaka na mestu na stranici gde se nalazi krajnja pozicija. Oznaka se dobija pomoću atributa *name* u `<A>` tagu:

`<A NAME= ime_krajnje_pozicije>` tekst krajnje pozicije sa koje se prelazi iz polazne pozicije `</A>`

Atribut NAME nije obavezan. Ukoliko se on izostavi, čitač se pozicionira na početak stranice sa krajnjom pozicijom, a inače na naznačenu poziciju. Znači da bi se definisala veza između dva dela iste stranice potrebno je uraditi sledeće:

- na polaznoj poziciji navodi se tag sa imenom krajnje pozicije:

`<A HREF="#ime_krajnje_pozicije">` tekst polazne pozicije `</A>`

- na krajnjoj poziciji se navodi tag koji definiše ime krajnje pozicije:

`<A NAME= " ime_krajnje_pozicije ">` tekst krajnje pozicije `</A>`

Između početnog `<A NAME= " ime_krajnje_pozicije ">` i završnog `</a>` taga može stajati bilo koji element stranice (tekst, slika, ...), a ne mora stajati nijedan element, kao što je slučaj u prethodnom primeru. Ovaj tag ne proizvodi nikakav vidljiv efekat u HTML stranici - on deluje u pozadini stranice i služi čitaču da se lakše orijentiše.

Ako se želi da se uspostavi veza sa nekom drugom stranicom, ali ne sa početkom te stranice, već nekim drugim delom, potrebno je koristiti i prvu i drugu vrstu linkova. Opet se definiše mesto odakle se želi nastaviti sa pregledom pomoću atribut *href* u `<A>` tagu sa navođenjem imena stranice,

ali se dodaje deo `#ime_krajnje_pozicije`, da bi se pozicioniralo na krajnju poziciju.

Neka se krajnja pozicija nalazi u okviru stranice `PrimerLinka.html`, tada treba navesti:

```
<a href="PrimerLinka.html#ime_krajnje_pozicije "> tekst polazne pozicije
</a>
```

Sada kada korisnik klikne mišem na link "tekst polazne pozicije " odlazi se na novu stranicu `PrimerLinka.html` i to na njen deo označen sa stranice koji je definisan imenom `ime_krajnje_pozicije`.

SLIKA KAO LINK

Slika u HTML dokumentu može predstavljati i polazni čvor u hipervezi. Dolazna adresa se tada navodi kao kod hiperveze, a umesto teksta koji se može aktivirati navodi se tag `IMG`. Opšti oblik je:

```
<A HREF="URL dolazne datoteke"> "tag IMG za željenu sliku"</A>
```

Primer:

```
<a href="Primer.html"></a>
```

Sada slika `osam.gif` predstavlja link ka stranici `Primer.html`. Kada se klikne mišem na sliku u čitaču se umesto trenutno stranice prikazuje stranica `Primer.html`.

Takođe čitač automatski prikazuje okvir oko slike koja predstavlja link sa bojom koja je definisana pomoću *link* i *vlink* atributa u okviru `<BODY>` taga. Da bi se dobila slika bez takvog okvira treba definisati `border="0"` u okviru `<IMG>` taga.

LINK ZA ELEKTRONSKU POŠTU

Pored odlaska na neku drugu stranicu linkovi se mogu iskoristiti i za pisanje nove mail poruke pomoću default programa za elektronsku poštu (u korisničkim sistemima najviše korišćeni su Outlook Express ili Netscape Messenger). Pri generisanju nove poruke, već će biti upisana željena adresa, a korisniku preostaje samo da otkuca tekst poruke i da je pošalje.

Da bi se poslala poruka na mail adresu nbosko@etf.bg.ac.yu potrebno je u okviru stranice navesti sledeći tekst:

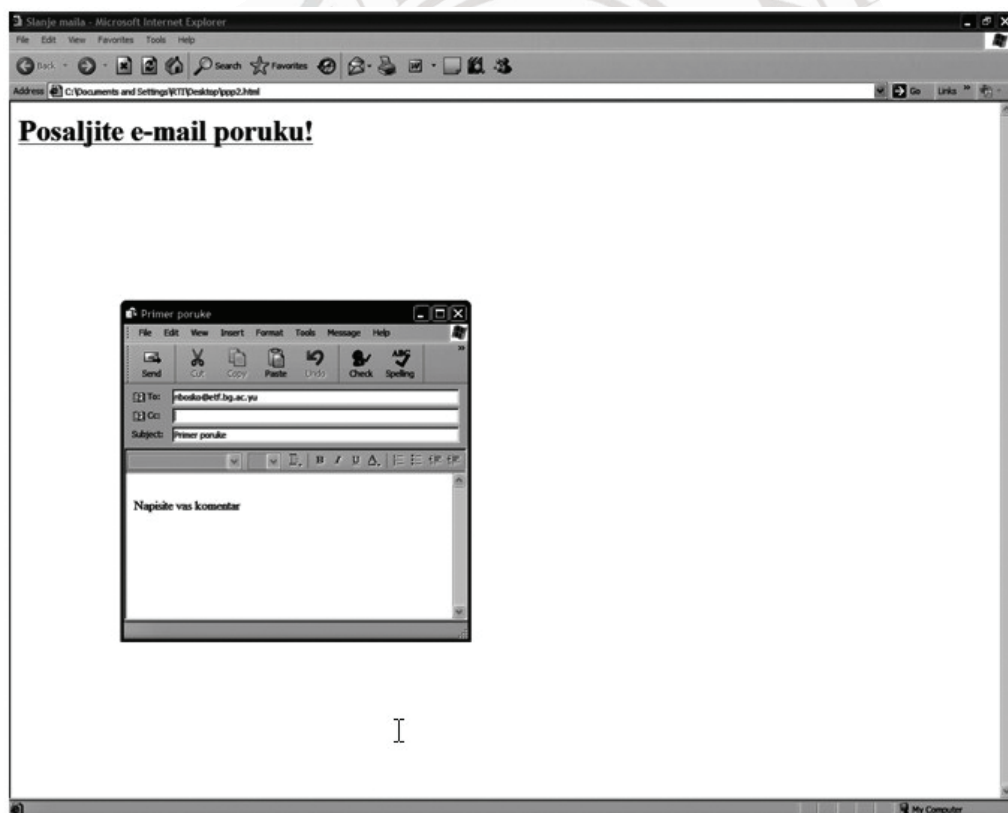
`<a href="mailto: nbosko@etf.bg.ac.yu"> Posaljite e-mail poruku! </a>`

Pored adrese na koju se želi poslati poruka mogu se podesiti još neki parametri, kao što su naslov ili tekst poruke. Sledeći kod

`<a href="mailto:nbosko@etf.bg.ac.yu?subject=Primer poruke&body=Napisite vas komentar"> Posaljite e-mail poruku! </a>`

Definiše poruku koja ima naslov “Primer poruke” i u okviru tela poruke je postavljen tekst “Napisite vas komentar”.

Nakon aktiviranja linka “Posaljite e-mail poruku” dobija se sledeći izgled ekrana



Primeri za rad u laboratoriji

Primer 1. Linkovi u okviru iste stranice

```

<HTML>
<HEAD><TITLE> Link u okviru iste stranice</TITLE></HEAD>
<H2>Programiranja </H2>
<OL>
  <LI> <A HREF="#html"> Web i jezik HTML </A> ;
  <LI> <A HREF="#algo"> Programski jezik Java </A>
  <LI> <A HREF="#uvodc"> Uvod u C</A>
</OL>
<H3> <A name="html"> Web i jezik HTML</A> </H3>
  <p>.....<BR>
  .....</p>
  <p>.....<BR>
  .....</p>
  <p>.....<BR>
  .....</p>
<H3> <A name="java"> Programski jezik Java </A> </H3>
  <p>.....<BR>
  .....</p>
  <p>.....<BR>
  .....</p>
  <p>.....<BR>
  .....</p>
<H3> <A name="uvodc"> Uvod u C </A> </H3>
  <p>.....<BR>
  .....</p>
  <p>.....<BR>
  .....</p>
  <p>.....<BR>
  .....</p>
</UL>
</BODY>
</HTML>

```

Primer 2. Veza ka stranici u okviru iste aplikacije

```

<HTML>
<HEAD>
  <TITLE>Dokument A</TITLE>
</HEAD>
<BODY>
  <p>Ovo je stranica a.html iz koje se poziva
  Stranica b.html u istom direktorijumu. </p>
  <!-- iz stranice a u tekucem direktorijumu ---->
  <!-- poziva se stranica b u istom direktorijumu --->
  <A HREF=" b.html">
    Poziv stranice b.html
  </A>
</BODY>
</HTML>

```

```

<HTML>
<HEAD>
  <TITLE>Dokument B</TITLE>
</HEAD>
<BODY>
  <p><p>Ovo je stranica b.html iz koje se poziva
  Stranica a.html u istom direktorijumu. </p>
  <!-- iz stranice b u tekucem direktorijumu ---->
  <!-- poziva se stranica a u istom direktorijumu --->
  <A HREF=" a.html">
    Poziv stranice a.html
  </A>
</BODY>
</HTML>

```

Primer 3. Veza ka stranici na drugom serveru. U ovom primeru, adresa servera je www.w3.org, a putanja do stranice Addressing.html je [hypertext/WWW/Addressing/](http://www.w3.org/hypertext/WWW/Addressing/):

```

<HTML>
<HEAD>
  <TITLE> Veza ka stranici na drugom serveru </TITLE>

```

```
</HEAD>
<BODY>
<p>Neke stranice o HTML-adresiranju:</p>
<a
href="http://www.w3.org/hypertext/WWW/Addressing/Addressing.html">
Adresiranje</a>
</BODY>
</HTML>
```

Primer 4. Primer slike i linka.

```
<html>
<head>
<title>Hvala!</title>
</head>
<body>
<h1 align="center">
Hvala!
</h1>
<p>Hvala sto ste poslali vas komentar! Mozete posetiti i stranicu
<a href="Primer1.html">.

</a>
</p>
</body> </html>
```



6. TABELE

Tabela je i u HTML-u dvodimenziona matrica čiji se elementi nazivaju ćelije (engl. *cell*). Ćelija može sadržavati raznovrsne informacije: brojeve, tekst, boje, liste, hiper-veze, slike, itd. Tabela se sastavlja tako što se opisuju redom njene vrste (redovi, engl. *row*) i sadržaj svake ćelije u redu.

Tabela se opisuje uz pomoć složenog taga `TABLE` koji može sadržavati više atributa:

- `BORDER` koji opisuje širinu spoljašnjeg okvira tabele;
- `CELLSPACING` koji opisuje širinu linije koja razdvaja dve ćelije;
- `CELLPADDING` koji opisuje prostor oko sadržaja ćelije;
- `WIDTH` koji opisuje ukupnu širinu tabele.

Nadnaslov tabele se može zadati tagom `CAPTION` koja se ispisuje iznad tabele i može imati atribut `ALIGN`:

- za vertikalno poravnavanje: `TOP`, `MIDDLE`, `BOTTOM`
- za horizontalno poravnavanje: `LEFT`, `CENTER`, `RIGHT`

Svaki red u tabeli se opisuje između tagova `<TR>` i `</TR>` (engl. *table row*). I tag `TR` može imati attribute:

- za horizontalno poravnavanje, atribut `ALIGN` sa vrednostima: `LEFT`, `CENTER`, `RIGHT`
- za vertikalno poravnavanje, atribut `VALIGN` sa vrednostima: `TOP`, `MIDDLE`, `BOTTOM`

Pojedinačna ćelija se opisuje između tagova `<TD>` i `</TD>`. Tag `TD`, pored atributa `ALIGN` i `VALIGN`, može imati i attribute:

- za horizontalno spajanje ćelija: `ROWSPAN` (spaja ćelije iste vrste) i
- za vertikalno spajanje ćelija: `COLSPAN` (spaja ćelije iste kolone)

Tag `<TH>` ima ista svojstva kao tag `<TD>` s tom razlikom što obezbeđuje da sadržaj ćelije bude automatski centriran i boldovan.

Tabela ne mora da sadrži <TH> tag, ali mora da sadrži bar jedan <TD> tag, u koji se smešta sadržaj tabele.

Opšta struktura jedne tabele je sledeća:

<TABLE>

<!-- pocetak definicije tabele -->

<CAPTION> sadržaj naslova tabele </CAPTION>

<!-- definicija naslova-->

<TR>

<!-- start definicije headera -->

<TH> sadržaj prve ćelije headera </TH>

<TH> sadržaj poslednje ćelije headera </TH>

</TR>

<!-- kraj definicije headera -->

<TR>

<!-- start prvog reda -->

<TD> sadržaj prve ćelije prvog reda </TD>

<TD> sadržaj poslednje ćelije prvog reda </TD>

</TR>

<!-- kraj prvog reda -->

<TR>

<!-- start poslednjeg reda -->

<TD> sadržaj prve ćelije poslednjeg reda </TD>

<TD> sadržaj poslednje ćelije poslednjeg reda </TD>

</TR>

<!-- kraj poslednjeg reda -->

</TABLE>

<!-- kraj definicije tabele-->

Treba naglasiti da ako se tekstu van tabele upotrebom taga dodeli neki font različit od default fonta, u Internet Exploreru će se ovaj tag odnositi i na tekst u ćelijama tabele. U Netscape Navigatoru će tekst u tabeli biti prikazan u default fontu. Da bi i Netscape Navigator tekst u tabeli

prikazao u željenom fontu treba u svaku ćeliju posebno da se ubaci `<font>` tag.

Ako se u kodu tabele ne navede određeni atribut tabele nemaju graničnu liniju, border. Da bi se dodelila tabeli granična linija odgovarajuće debljine, u `<table>` tag se stavlja atribut *border*, a vrednost debljine linije se zadaje u pikselima.

Sledeći primer pokazuje upotrebu atributa *border*:

Ime:	Prezime:	Zvanje:
Bosko	Nikolic	Predavac

Kod kojim se dobija prethodni primer:

```
<table border="2">
<tr>
  <th><font face="Verdana" size="2">Ime:</font></th>
  <th><font face="Verdana" size="2">Prezime:</font></th>
  <th><font face="Verdana" size="2">Zvanje:</font></th>
</tr>
<tr>
  <td><font face="Verdana" size="2">Bosko</font></td>
  <td><font face="Verdana" size="2">Nikolic</font></td>
  <td><font face="Verdana" size="2">Predavac</font></td>
</tr>
</table>
```

Osnovna razlika između teksta koji se nalazi između `<th>` i `<td>` tagova je u tome što je tekst u prvom slučaju boldovan i centriran unutar ćelije, a u drugom slučaju tekst nije podebljan i poravnat je uz levu ivicu ćelije. Ivice tabele se priljubljuju uz sadržaj ćelija maksimalno koliko je moguće. Širina kolona je određena prvom ćelijom u svakoj koloni.

Naravno, ovim se ne iscrpljuje mogućnost formatiranja tabela. U tu svrhu se koriste razni atributi. Treba napomenuti da tabela uopšte ne mora da bude uniformno formatirana, tj. da se svakoj ćeliji može dodeliti neko drugo svojstvo pomoću atributa koji se definišu unutar `<TD>` tagova.

Da bi tabela imala odgovarajuće dimenzije koriste se atributi *height* i *width*. Vrednost ovih atributa se kao i kod `<img>` taga može zadati ili u pikselima

ili u procentima veličine prozora HTML browsera. Preporučuje se drugi način definicije, jer onda izgled stranice ne zavisi od rezolucije ekrana.

Sledeći primer predstavlja tabelu sa širinom od 80% i visinom od 30%:

Ime :	Prezime :	Zvanje :
Bosko	Nikolic	Predavac

Odgovarajući HTML kod je:

```
<table border="2" height="30%" width="80%">
<tr>
  <th><font face="Verdana" size="2">Ime :</font></th>
  <th><font face="Verdana" size="2">Prezime :</font></th>
  <th><font face="Verdana" size="2">Zvanje :</font></th>
</tr>
<tr>
  <td><font face="Verdana" size="2">Bosko</font></td>
  <td><font face="Verdana" size="2">Nikolic</font></td>
  <td><font face="Verdana" size="2">Predavac</font></td>
</tr>
</table>
```

Da bi kolone bile jednake širine trebalo bi da u odgovarajuće <th> ili <td> tagove ubaciti atribut width sa željenom širinom. U slučaju tabele iz prethodnih primera to znači da treba u svaki <th> tag definisati atribut width sa vrednošću 33%. Tako, na primer, se može realizovati tabela koja će se protezati preko cele širine stranice, a svaka kolona će zauzimati tačno trećinu širine tabele:

Ime :	Prezime :	Zvanje :
Bosko	Nikolic	Predavac

HTML kod za prethodni primer je:

```
<table border="2" width="100%">
<tr>
```

```

<th width="33%"><font face="Verdana" size="2">Ime :</font></th>
<th width="33%"><font face="Verdana" size="2">Prezime
:</font></th>
<th width="33%"><font face="Verdana" size="2">Zvanje :</font></th>
</tr>
<tr>
<td><font face="Verdana" size="2">Bosko</font></td>
<td><font face="Verdana" size="2">Nikolic</font></td>
<td><font face="Verdana" size="2">Predavac</font></td>
</tr>
</table>

```

Dva atributa koja imaju veliku primenu su i *cellpadding* i *cellspacing*. Pomoću *cellpadding* atributa definiše se rastojanje između sadržaja ćelije i njene granične linije. Vrednost ovog atributa se zadaje u pikselima, ako se ne navede nijedna vrednost podrazumeva se da 1. Pomoću *cellspacing* atributa se može odrediti rastojanje između pojedinih ćelija tabele, tj. debljina linije između ćelija. Vrednost ovog atributa se zadaje, takođe, u pikselima ako se ne navede nijedna vrednost podrazumeva se da 1. Ako se u prethodnoj tabeli definiše vrednost *cellpadding* atributa od 30 piksela, a vrednost *cellspacing* atributa od 10 piksela dobija se sledeća tabelu:

Ime :	Prezime:	Zvanje :
Bosko	Nikolic	Predavac

HTML kod ove tabele glasi:

```

<table border="2" cellpadding="30" cellspacing="10">
<tr>

```

```

<th width="33%"><font face="Verdana" size="2">Ime :</font></th>
<th width="33%"><font face="Verdana" size="2">Prezime
:</font></th>
<th width="33%"><font face="Verdana" size="2">Zvanje :</font></th>
</tr>
<tr>
<td><font face="Verdana" size="2">Bosko</font></td>
<td><font face="Verdana" size="2">Nikolic</font></td>
<td><font face="Verdana" size="2">Predavac</font></td>
</tr>
</table>

```

HTML dopušta mogućnost da se pojedine ćelije tabele protežu duž više redova ili kolona tabele. Ovakav efekat se može postići pomoću atributa *colspan* i *rowspan*, koji se ubacuju u `<td>` ili `<th>` tag one ćelije koja se želi posebno da formatira. Vrednost ovih atributa se zadaje brojem kolona ili redova tabele duž kojih treba da se prostire data ćelija. U sledećem primeru je prikazana upotreba ovih atributa:

		Dani u nedelji :				
		ponedeljak	utorak	sreda	cetvrtak	petak
c a s o v i	1.	srpski	istorija	fizicko	hemija	srpski
	2.	matematika	srpski	fizicko	hemija	srpski
	3.	fizicko	matematika	istorija	biologija	engleski
	4.	fizicko	matematika	fizika	informatika	engleski
	5.	fizika	razredni	geografija	informatika	matematika

HTML kod tabele iz prethodne tabele je:

```
<table border="2" width="100%">
<tr>
  <th rowspan="2" colspan="2"></th>
  <th colspan="5"> Dani u nedelji :</th>
</tr>
<tr>
  <th width="20%">ponedeljak</th>
  <th width="20%">utorak</th>
  <th width="20%">sreda</th>
  <th width="20%">cetvrtak</th>
  <th width="20%">petak</th>
</tr>
<tr>
  <th rowspan="5">c<p>a<p>s<p>o<p>v<p>i</th>
  <th>1.</th>
  <td>srpski</td>
  <td>istorija</td>
  <td>fizicko</td>
  <td>hemija</td>
  <td>srpski</td>
</tr>
<tr>
  <th>2.</th>
  <td>matematika</td>
  <td>srpski</td>
  <td>fizicko</td>
  <td>hemija</td>
  <td>srpski</td>
</tr>
<tr>
  <th>3.</th>
  <td>fizicko</td>
  <td>matematika</td>
  <td>istorija</td>
  <td>biologija</td>
  <td>engleski</td>
</tr>
```

```
<tr>
  <th>4.</th>
  <td>fizicko</td>
  <td>matematika</td>
  <td>fizika</td>
  <td>informatika</td>
  <td>engleski</td>
</tr>
<tr>
  <th>5.</th>
  <td>fizika</td>
  <td>razredni</td>
  <td>geografija</td>
  <td>informatika</td>
  <td>matematika</td>
</tr>
</table>
```

U prvu ćeliju tabele je postavljena slika tipa gif pod nazivom space.gif. Ovo je neophodno stoga što svaka ćelija tabele mora imati neki sadržaj, inace je browser neće prikazati.

HTML dozvoljava da cela tabela kao i svaka ćelija posebno može da ima različitu boju pozadine od ostatka stranice. Ovakav efekat se postiže umetanjem atributa *bgcolor* u sledeće tagove: `<table>`, `<td>` ili `<th>`. Vrednost se zadaje heksadecimalnim kodom boje ili njenim imenom, isto kao i kod boje pozadine stranice u `<body>` tagu.

U sledećem primeru je demonstriran upotrebe ovih atributa:

Dani u nedelji :					
	ponedeljak	utorak	sreda	cetvrtak	petak
c a s o v i	1. srpski	istorija	fizicko	hemija	srpski
	2. matematika	fizicko	hemija	srpski	srpski
	3. fizicko	matematika	istorija	biologija	engleski
	4. fizicko	matematika	fizika	informatika	engleski
	5. fizika	razredni	geografija	informatika	matematika

HTML kod ovog primera glasi:

```
<table border="2" width="100%" bgcolor="#ffffff">
<tr>
  <th rowspan="2" colspan="2" bgcolor="#b0e0e6"></th>
  <th colspan="5" bgcolor="#ffff00"> Dani u nedelji :</th>
</tr>
<tr>
  <th width="20%" bgcolor="#9acd32">ponedeljak</th>
  <th width="20%" bgcolor="#9acd32">utorak</th>
  <th width="20%" bgcolor="#9acd32">sreda</th>
  <th width="20%" bgcolor="#9acd32">cetvrtak</th>
  <th width="20%" bgcolor="#9acd32">petak</th>
</tr>
<tr>
  <th rowspan="5" bgcolor="#ffff00">c<p>a<p>s<p>o<p>v<p>i</th>
  <th bgcolor="#9acd32">1.</th>
  <td bgcolor="#b0e0e6">srpski</td>
  <td bgcolor="#b0e0e6">istorija</td>
```

	fizicko
	hemija
	srpski
2.	matematika
	srpski
	fizicko
	hemija
	srpski
3.	fizicko
	matematika
	istorija
	biologija
	engleski
4.	fizicko
	matematika
	fizika
	informatika
	engleski
5.	fizika
	razredni
	geografija
	informatika
	matematika

U prethodnom primeru se može primetiti da atribut *bgcolor* koji je zadat u <th> ili <td> tagu će se prikazati umesto one boje koja se nalazi u <table> tagu. U primeru je u <table> tagu definisana bela boja pozadine koja se vidi samo na ivicama tabele (ako se koristi Internet Explorer), jer je u svakoj pojedinačnoj ćeliji definisana posebna boja pozadine *bgcolor* atributom.

Takođe boja okvira tabele se može promeniti pomoću *bordercolor* atributa u <table> tagu. U sledećem primeru je prethodnoj tabeli dfinisan crni okvir:

		Dani u nedelji :				
		ponedeljak	utorak	sreda	cetvrtak	petak
c a s o v i	1.	srpski	istorija	fizicko	hemija	srpski
	2.	matematika	fizicko	hemija	srpski	srpski
	3.	fizicko	matematika	istorija	biologija	engleski
	4.	fizicko	matematika	fizika	informatika	engleski
	5.	fizika	razredni	geografija	informatika	matematika

HTML kod koji treba promeniti u prethodnoj tabeli je:

```
<table border="2" width="100%" bgcolor="#ffffff"
bordercolor="#000000">
```

Primeri za rad u laboratoriji

Primer 1. Tabela kodova.

```

<HTML>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html;
charset=iso-8859-2">
<HEAD>
  <TITLE>Tabela kodova </TITLE>
</HEAD>
<BODY>
<BR><BR>
<DIV ALIGN="CENTER">
<TABLE CELLPADDING=10 CELLSPACING=0.5 BORDER="5">
<CAPTION><B>YU-karakter i u ISO-8859-2</B></CAPTION>
<TR VALIGN="TOP">
  <TH ALIGN="CENTER">Veiko slovo</TH>
  <TH ALIGN="CENTER"><B>kod 8859-2</B></TH>
  <TH ALIGN="CENTER"><B>Malo slovo</B></TH>
  <TH ALIGN="CENTER"><B>kod 8859-2</B></TH>
</TR>
<TR VALIGN="TOP">
  <TD ALIGN="CENTER"><B>Č</B></TD>
  <TD ALIGN="CENTER">200</TD>
  <TD ALIGN="CENTER"><B>č</B></TD>
  <TD ALIGN="CENTER">232</TD>
</TR>
<TR VALIGN="TOP">
  <TD ALIGN="CENTER"><B>Ć</B></TD>
  <TD ALIGN="CENTER">198</TD>
  <TD ALIGN="CENTER"><B>ć</B></TD>
  <TD ALIGN="CENTER">230</TD>
</TR>
<TR VALIGN="TOP">
  <TD ALIGN="CENTER"><B>Đ</B></TD>
  <TD ALIGN="CENTER">208</TD>
  <TD ALIGN="CENTER"><B>đ</B></TD>
  <TD ALIGN="CENTER">248</TD>

```

```

</TR>
<TR VALIGN="TOP">
  <TD ALIGN="CENTER"><B>Š</B></TD>
  <TD ALIGN="CENTER">169</TD>
  <TD ALIGN="CENTER"><B>š</B></TD>
  <TD ALIGN="CENTER">185</TD>
</TR>
<TR VALIGN="TOP">
  <TD ALIGN="CENTER"><B>Ž</B></TD>
  <TD ALIGN="CENTER">174</TD>
  <TD ALIGN="CENTER"><B>ž</B></TD>
  <TD ALIGN="CENTER">190</TD>
</TR>
</TABLE>
</DIV>
</BODY>
</HTML>

```

Primer 2. Boje u tabeli.

```

<HTML>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html;
charset=iso-8859-2">
<HEAD>
  <TITLE>Tabela boja</TITLE>
</HEAD>
<BODY BGCOLOR="#000000">
<DIV ALIGN="CENTER">
  <TABLE CELLPADDING=80 WITH="100%">
    <TR>
      <!-- COLSPAN spaja tri kolone u jednu ---->
      <TD BGCOLOR="#FF0000" COLSPAN=3><BR></TD>
      <TD BGCOLOR="#00FF00" COLSPAN=3><BR></TD>
    </TR>
      <TD BGCOLOR="#FFFF00" COLSPAN=3><BR></TD>
      <TD BGCOLOR="#0000FF" COLSPAN=3><BR></TD>
    </TR>
  </TABLE>
</DIV>

```

</BODY>

</HTML>

Primer 3. Primer jedne HTML stranice sa tabelom.

```

<html><!-- Ovako se pise komentar. -->
<head>
<title>HTML programiranje</title>
<meta http-equiv="Content-Type" content="text/html; charset=windows-
1250">
</head>
<body bgcolor="#FFFFFF" link="#CE0421" vlink="#000066">
<h1 align=center> Viša zeleynicka škola Beograd &copy;2004<br>
  HTML programiranje </h1>
<hr>
<h3 align=left>H3 tag za naslov!</h3>
<p><b>Boldovan tekst u paragraf tagu!</b><br>
<u>Ovako izgleda podvučeno</u></p>
<div align="right">Div-tag za podešavanje sadržaja (left, right,
center)</div><br>
  <ul><b>Linkovi mogu biti:</b>
    <li><b>Tekst <a href="http://www.vets.edu.yu/"> Viša
elektroethnicka škola</a></b>
    <br>
    <li><b>Slike</b> <a href="http://www.etf.bg.ac.yu/"> </a>
    </ul>
    <dl>
    <dt>Postoje tabele sa: </dt>
    <dd><font color="#0000CC">Borderima-vidljivim ivicama
</font>
    <dd><font color="#990000"> Tabela bez bordera
</font>
    </dd>
    </dl>
    <table border=2 width=100%>
    <tr>
    <td>Ova</td>
    <td>tabela ima</td>

```

```

<td>border(ivicu) 2 pixela</td>
</tr>
<tr>
<td>Ova</td>
<td>tabela ima</td>
<td>3 reda i 3 kolone</td>
</tr>
<tr>
<td>Ova</td>
<td>tabela je</td>
<td>sirine 100% ekrana</td>
</tr>
</table>
<table width=700 cellpadding=2 cellspacing=3>
<tr>
<td>Ovo</td><td>je tabela</td><td>bez bordera</td>
</tr>
<tr>
<td align="center" valign="middle"><font color="Red" size="5"
  face="Verdana, Arial, Helvetica, sans-serif"> Atributi</font></td>
<td>tabele su</td>
<td><font size=+2 color="blue">cellpadding(rastojanje od ivica
celije)</font></td>
</tr>
<tr>
<td><font color="Red" face="Arial">Atributi</font></td>
<td>tabele su</td>
<td><font size=+2 color="blue">cellspacing(rastojanje izmedju
celija)</font></td>
</tr>
<tr>
<td><font color="Red" face="Arial">Atribut</font></td>
<td>tabela je</td>
<td><font size=+2 color="blue">sirine 700 pixela</font></td>
</tr>
</table>
</body>
</html>

```



7. FREJMOVI

Frejmovi omogućavaju da se formira dokument koji će se sastojati od adresa bar dva različita dokumenta. Browser, ukoliko je sposoban za to, interpretira frejmove kao podelu tekućeg prozora na više nezavisnih podprozora od koji svaki sadrži adresirani dokumenat. Osnovni tag je složeni tag `<FRAMESET>`. Ovaj tag zamenjuje tag `BODY` u HTML-dokumentu.

Tekst dokumenta koji se rastavlja na frejmove sadrži isključivo informacije namenjene browseru: browser poziva adresirana dokumenta i postavlja ih u odgovarajuće frejmove.

Tag `<FRAMESET>` ima attribute:

- `COLS` za vertikalnu podelu prozora i
- `ROWS` za horizontalnu podelu prozora navigatora.

Adresa dokumenta se navodi u okviru taga `FRAME` čiji su atributi

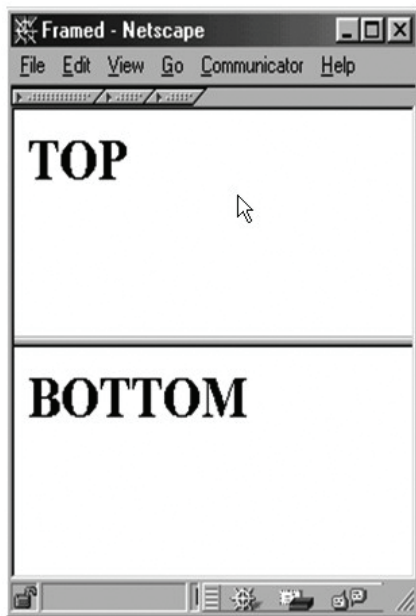
- `SRC`, preko koje se zadaje adresa dokumenta koji će biti prikazan u zoni tog taga i
- `MARGINWIDTH` i `MARGINHEIGHT`

Tag `NOFRAMES` sadrži poruku za browser koji nije u stanju da interpretira frejmove.

Opšta struktura HTML stranice sa frejmovima je:

```
<HTML>
<HEAD>
</HEAD>
<FRAMESET>
...
</FRAMESET>
</HTML>
```

Primer HTML koda stranice koja koristi frejmove i njen izgled je dat na sledećoj slici:



```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">

<HTML>
<HEAD>
  <TITLE>Framed</TITLE>
</HEAD>

<FRAMESET ROWS="50%,50%">

  <FRAME SRC="top.html">
  <FRAME SRC="bottom.html">

</frameset>

<NOFRAMES>
Text for browsers that
don't support frames
</NOFRAMES>

</HTML>
```

Atributi <FRAMESET> taga

- **BORDER:** Vrednost atributa je debljina granice između frejmova izrađena u pikselima.
 - **BORDERCOLOR:** Definiše boju granice između frejmova. Ovaj atribut se primenjuje jedino pod uslovom da je atribut FRAMEBORDER postavljen na aktivnu vrednost.
 - **FRAMEBORDER:** Vrednost '1' ili 'yes' ovog atributa prikazuje granicu između dva frejma, dok vrednost '0' ili 'no' ne prikazuje.
 - **FRAMESPACING:** Definiše razmak između granica frejma i njegovog sadržaja izraženog u pikselima.
- Primer korišćenja navedenih atributa je:

```
<HTML>
<HEAD><TITLE>Nova stranica sa frejmovima</TITLE>
</HEAD>
<FRAMESET ROWS="20%, 80%" FRAMEBORDER="NO"
BORDER="3">
  <FRAME SRC="menu.html">
  <FRAME SRC="main.html">
</FRAMESET>
<NOFRAMES>
```

U okviru vaseg citaca nije moguće prikazati ovu stranicu.

</NOFRAMES>

</HTML>

Atributi <FRAME> taga

- **SRC**: definiše adresu dokumenta koji će se učitati u okviru frejma.
- **BORDERCOLOR**: Definiše boju za ivicu frejma.
- **FRAMEBORDER**: Vrednost '1' ili 'yes' ovog atributa prikazuje granicu frejma, dok vrednost '0' ili 'no' ne prikazuje.
- **MARGINWIDTH**: Definiše veličinu praznog prostor između leve i desne strane frejma i njegovog sadržaja izražen u pikselima.
- **MARGINHEIGHT**: Definiše veličinu praznog prostor između vrha i dna frejma i njegovog sadržaja izražen u pikselima.
- **NAME**: Specificira ime frejma koje se može koristiti u okviru drugih programskih modula, kao što su JavaScript ili VBScript.
- **NORESIZE**: Aktivnom vrednošću ovog atributa se sprečava korisnik da menja zadatu veličinu frejma. Ovaj atribut se samo navodi bez ikakve vrednosti.
- **SCROLLING**: Vrednosti ovog atributa su 'YES', 'NO' i 'AUTO'. Ako je vrednost 'YES' scroll bar će se pojaviti bez obzira da li je potreban iline, 'NO' će sprečiti njegovu pojavu, a sa 'AUTO' će se pojaviti samo kada je potreban

Kada stranica sadrži određeni meni, uvek se upotrebljavaju frejmovi. Korišćenjem frejmova i linkova može se omogućiti promena jednog dela stranice, dok bi preostali deo stranice ostao nepromenjen. Da bi se postigao ovakav efekat pri definiciji frejma, čiji se sadržaj želi menjati treba mu dodeliti i ime. Na primer:

```
<FRAME SRC="Primer.html" name="MenjaSe">
```

Na ovaj način se definiše frejm sa imenom MenjaSe u koji će se učitati stranica Primer.html. Sledeće što treba uraditi je da kod definicije linka (koji se nalazi na drugom frejmu) se navede atribut *target* sa vrednošću koja predstavlja ime frejma čiji se sadržaj menja. Znači ako se želi napraviti link koji će sadržaj prethodno definisanog frejma promeniti i u njega učitati stranicu www.google.com tada treba definisati sledeći kod

```
<A href="http://www.google.com" target="MenjaSe"> Promena frejma  
</A>
```

Kao primer mogu poslužiti sledeće tri strane: Glavna.html, meni.html i PocetniSadrzaj.html. Njihov kod je:

Stranica Glavna.html

```
<html>  
<frameset cols="25%,75%">  
  <frame src="meni.html">  
  < frame src ="PocetniSadrzaj.html" name="MenjaSe">  
</frameset>  
</html>
```

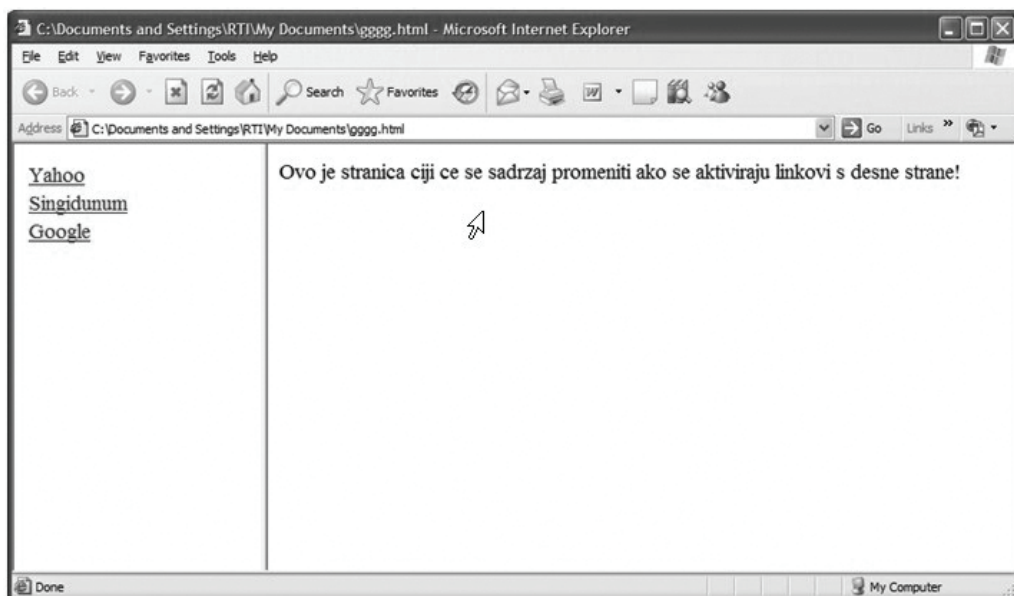
Stranica meni.html

```
<html>  
<body>  
<table>  
<tr>  
<td><a href ="http://www.yahoo.com" target =" MenjaSe">Yahoo</a>  
</td></tr>  
<td>  
<a href ="http://www.singidunum.ac.yu" target="MenjaSe">Singidunum  
</a> </td></tr>  
<td><a href ="http://www.google.com" target ="MenjaSe">Google</a>  
</td></tr>  
</table>  
</body>  
</html>
```

Stranica PocetniSadrzaj.html

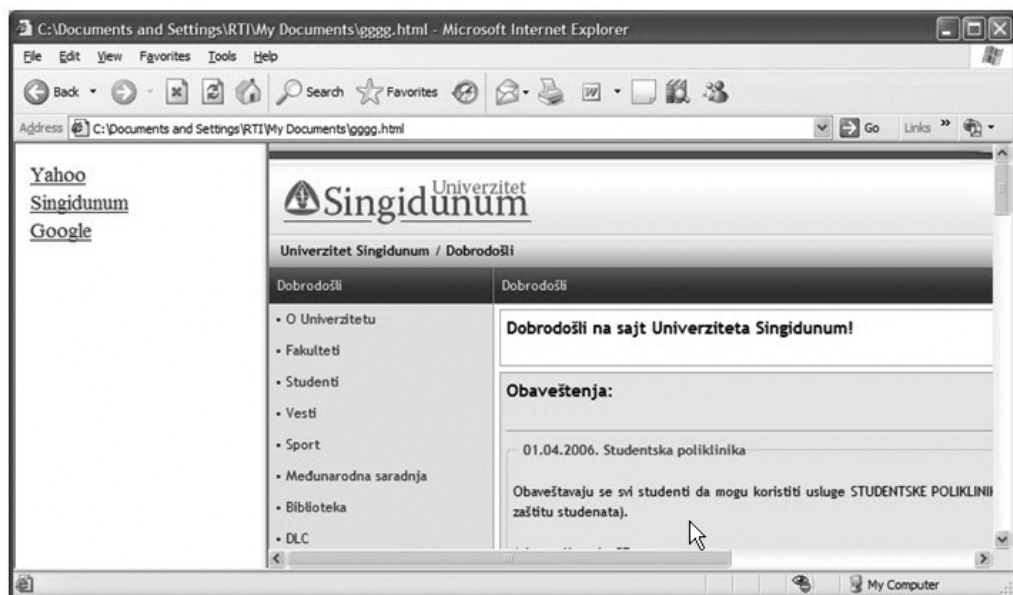
```
<html>  
<body>  
Ovo je stranica ciji ce se sadrzaj promeniti ako se aktiviraju linkovi s desne  
strane!  
</body>  
</html>
```

Nakon pokretanja stranice Glavna.html dobija se sledeći izgled u čitaču:



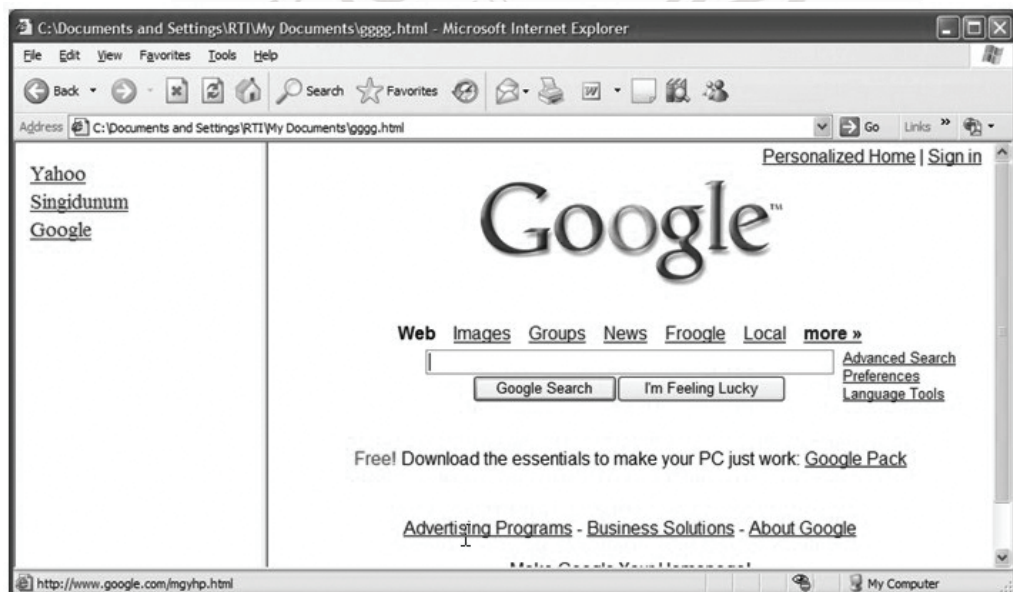
Stranica Glavna.html je pomoću frejmova vertikalno podeljena na dva dela, u prvom frejmu, koji zauzima 25% stranice prikazuje se stranica meni.html, a u drugom, koji zauzima 75% stranice prikazuje se stranica PocetniSadrzaj.html. drugi frejm je nazvan menja se.

Kada se aktivira link na levoj strani *Singidunum* dobija se sledeći izgled u okviru čitača:



Ovakav rezultat je dobijen, tako što je u okviru linka definisano da se menja sadržaj frejma pod nazivom MenjaSe, i da se u njega učita sadržaj stranice www.singidunum.ac.yu.

Ako se aktivira link *Google*, tada se dobija sledeći izgled stranice u okviru čitača:



Primeri za rad u laboratoriji

Primer 1. Rad sa frejmovima.

```
<html>  
  
<frameset cols="25%,50%,25%">  
  
  <frame src="tryhtml_frame_a.htm">  
  <frame src="tryhtml_frame_b.htm">  
  <frame src="tryhtml_frame_c.htm">  
  
</frameset>  
  
</html>
```

Primer 2. Rad sa frejmovima.

```
<html>  
  
<frameset rows="25%,50%,25%">  
  
  <frame src="tryhtml_frame_a.htm">  
  <frame src="tryhtml_frame_b.htm">  
  <frame src="tryhtml_frame_c.htm">  
  
</frameset>  
  
</html>
```

Primer 3. Rad sa frejmovima.

```
<html>  
  
<frameset rows="50%,50%">
```

```
<frame src="tryhtml_frame_a.htm">  
  
<frameset cols="25%,75%">  
<frame src="tryhtml_frame_b.htm">  
<frame src="tryhtml_frame_c.htm">  
</frameset>  
  
</frameset>  
  
</html>
```

Primer 4. Rad sa frejmovima.

```
<html>  
  
<frameset rows="50%,50%">  
  
<frame noresize="noresize" src="tryhtml_frame_a.htm">  
  
<frameset cols="25%,75%">  
<frame noresize="noresize" src="tryhtml_frame_b.htm">  
<frame noresize="noresize" src="tryhtml_frame_c.htm">  
</frameset>  
  
</frameset>  
  
</html>
```

Primer 5. Rad sa frejmovima.

```
<html>  
  
<frameset cols="120,*">  
  
<frame src="tryhtml_contents.htm">  
<frame src="tryhtml_frame_a.htm"  
name="showframe">
```

```
</frameset>
```

```
</html>
```

Primer 6. Rad sa frejmovima.

```
<html>
```

```
<frameset cols="180,*">
```

```
<frame src="content.htm">
```

```
<frame src="link.htm" name="showframe">
```

```
</frameset>
```

```
</html>
```

```
<html>
```

```
<body>
```

```
<iframe src="default.asp"></iframe>
```

```
<p>Vas citac ne podrzava frejmove</p>
```

```
<p>pa se ne mogu prikazati.</p>
```

```
</body>
```

```
</html>
```

Primer 7. Primer promene sadržaja frejma.

Kod stranice main.html

```
<html>
```

```
<frameset rows="5%,95%">
```

```
<frame src="menu.html">
```

```
<frame src="page1.html" name="content">
```

```
</frameset>
```

```
</html>
```

Kod stranice page1.html

```
<html>
```

```
<frameset rows="50%,50%">
```

```
<frame src="page1a.html">
```

```
<frameset cols="25%,75%">
```

```
<frame src="page1b.html">
<frame src="page1c.html">
</frameset>
</frameset>
</html>
```

page2.html

```
<html>
<frameset cols="180,*">
<frame src="page2a.html">
<frame src="page2b.html" name="page2b">
</frameset>
</html>
```

Kod stranice menu.html

```
<html>
<body>
<table width="90%" align="center">
<tr>
<td><a href="http://www.b92.net" target="content">Frame 1</a> </td>
<td><a href="page1.html" target="content">Frame 2</a></td>
<td><a href="page2.html" target="content">Frame 3</a></td>
</tr>
</table>
</body>
</html>
```

Kod stranice page1a.html

```
<html>
<body bgcolor="#AAFFAA">
<h3>Sub frame a</h3>
</body>
</html>
```

Kod stranice page1b.html

```
<html>
<body bgcolor="#AAAAFF">
```

```
<h3>Sub frame b</h3>
</body>
</html>
```

Kod stranice page1c.html

```
<html>
<body bgcolor="#FFAAAA">
<h3>Sub frame c</h3>
</body>
</html>
```

Kod stranice page2a.html

```
<html>
<body>
<ol>
<li><a href="page2b.html" target="page2b">Link without
Anchor</a><br>
<li><a href="page2b.html#1" target="page2b">Section 1</a>
<li><a href="page2b.html#2" target="page2b">Section 2</a>
<li><a href="page2b.html#3" target="page2b">Section 3</a>
</ol>
</body>
</html>
```

Kod stranice page2b.html

```
<html>
<body>
<a name="1"><h2>Section 1</h2>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce
interdum, arcu quis gravida mollis, nunc lectus commodo enim, ac tristique
sem enim eget urna. Vestibulum adipiscing leo nec libero. Phasellus
vestibulum egestas sem. Phasellus id odio in orci pellentesque consequat.
Sed ac pede. Pellentesque tincidunt. Nullam at elit. Aenean nibh.
Suspendisse potenti. Etiam non mi. Nulla commodo adipiscing leo. Quisque
dignissim
</p>
</a>
<a name="2"><h2>Section 2</h2>
```

<p>Curabitur dui sem, ultricies id, cursus at, blandit eu, quam. Nam felis. Mauris condimentum consequat risus. Quisque a nibh sit amet magna sagittis posuere. Duis imperdiet. Fusce congue nisi eu nibh. Etiam sit amet turpis at arcu faucibus ultricies.

</p>

<h2>Section 3</h2>

<p>Sed in sem a nisl varius convallis. Curabitur est. Vestibulum tempor massa sed lorem. Maecenas tortor. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Pellentesque erat. Etiam consectetuer felis non leo. Vivamus

</p>

<h2>Section 4</h2>

<p>Nunc ac nisi in sem elementum semper. Duis nec tortor ut orci facilisis porttitor. Nunc faucibus vestibulum felis. Sed nibh. Phasellus ligula. Praesent nisl. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Nunc diam, sed consequat nibh mi id ligula. Aliquam blandit fermentum nulla. Morbi elementum, diam at fermentum elementum, massa magna tincidunt justo, quis pretium enim massa sit amet felis

</p>

</body>

</html>

Nakon pokretanja stranice main.html dobij ase sledeći izgled stranice u okviru čitača



A nakon aktiviranja linka *Frame3* dobija sledeći izgled stranice:

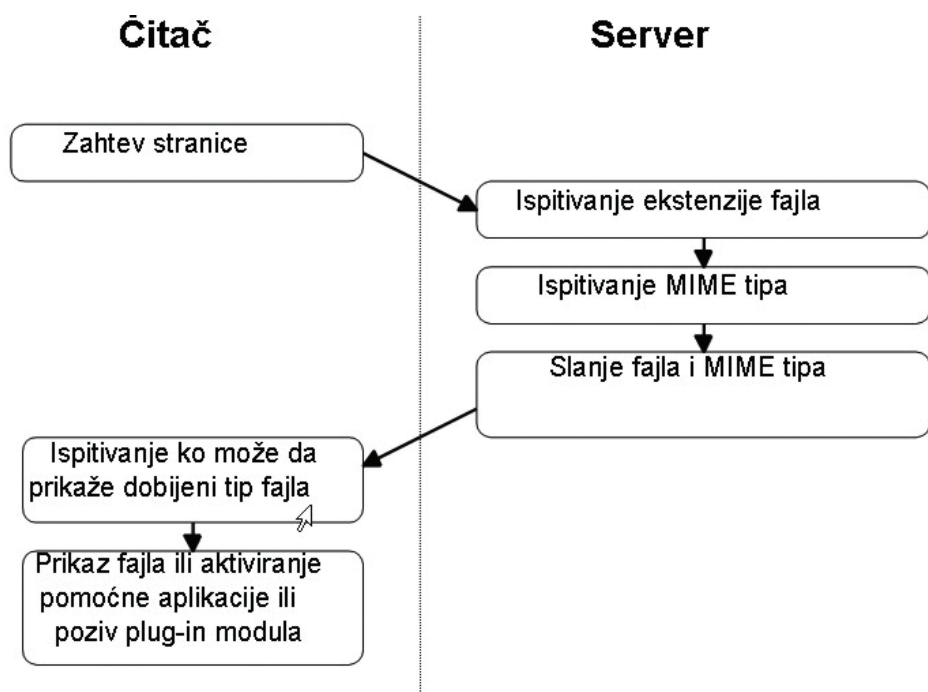
Frame 1	Frame 2	Frame 3
1. Link without Anchor 2. Section 1 3. Section 2 4. Section 3	Section 2 Curabitur dui sem, ultricies id, cursus at, blandit eu, quam. Nam felis. Mauris condimentum consequat risus. Quisque a nibh sit amet magna sagittis posuere. Duis imperdiet. Fusce congue nisi eu nibh. Etiam sit amet turpis at arcu faucibus ultricies. Curabitur porttitor, neque vulputate vehicula scelerisque, elit ante vulputate enim, vel egestas massa enim id felis. Curabitur sem sapien, iaculis ac, dapibus sit amet, pellentesque mollis, neque. Duis sed lorem ac lectus congue porta. Mauris scelerisque, ligula vehicula viverra dignissim, eros metus fringilla nibh, id pellentesque odio libero quis arcu. Etiam vel tortor. Quisque ac enim nec sem tincidunt ultrices. Maecenas quam turpis, hendrerit sed, mollis nec, pulvinar vel, nulla. Pellentesque lectus lacus, ultricies vitae, molestie a, molestie eu, felis. In nibh. Donec mattis. Nulla vel eros. Nunc ac nisi in sem elementum semper. Duis nec tortor ut orci facilisis porttitor. Nunc faucibus vestibulum felis. Sed nibh. Phasellus ligula. Praesent nisl. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Nunc mollis dapibus magna. Donec dignissim commodo orci. Etiam elementum rhoncus pede. Mauris eu ipsum. Ut ullamcorper. Donec commodo, velit eu blandit euismod, ligula odio fringilla diam, sed consequat nibh mi id ligula. Aliquam blandit fermentum nulla. Morbi elementum, diam at fermentum elementum, massa magna tincidunt justo, quis pretium enim massa sit amet felis. Nunc ac nisi in sem elementum semper. Duis nec tortor ut orci facilisis porttitor. Nunc faucibus vestibulum felis. Sed nibh. Phasellus ligula. Praesent nisl. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Nunc mollis dapibus magna. Donec dignissim commodo orci. Etiam elementum rhoncus pede. Mauris eu ipsum. Ut ullamcorper. Donec commodo, velit eu blandit euismod, ligula odio fringilla diam, sed consequat nibh mi id ligula. Aliquam blandit fermentum nulla. Morbi elementum, diam at fermentum elementum, massa magna tincidunt justo, quis pretium enim massa sit amet felis. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce interdum, arcu quis gravida mollis, nunc lectus commodo enim, ac tristique sem enim eget urna. Vestibulum adipiscing leo nec libero. Phasellus vestibulum egestas sem. Phasellus id odio in orci pellentesque consequat. Sed ac pede. Pellentesque tincidunt. Nullam at elit. Aenean nibh. Suspendisse potenti. Etiam non mi. Nulla commodo adipiscing leo. Quisque dignissim leo quis justo. Morbi viverra scelerisque quam. Donec ligula velit, porta eu, interdum id, lacinia porta, turpis. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Praesent lorem. Section 3 Sed in sem a nisl varius convallis. Curabitur est. Vestibulum tempor massa sed lorem. Maecenas tortor. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Pellentesque erat. Etiam consectetur felis non leo. Vivamus commodo rhoncus tortor. Vivamus vulputate, metus nec pharetra condimentum, felis turpis consectetur ligula, eget placerat turpis sapien ac magna. Nulla ac neque sed sem vehicula tincidunt. Praesent viverra ante vitae erat. Mauris ac elit. Donec lectus. Donec at massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Nam ante. In sit amet orci nec dui pretium tincidunt. Nunc ac nisi in sem elementum	

8. ZVUK I FILM

U okviru HTML stranice je moguće izvršavati različite multimedijalne dokumente. U okviru ovog poglavlja se razmatra dodavanje zvučnih i filmskih dokumenata.

Prikazivanje multimedijalnih objekata u okviru čitača

Sledeća slika opisuje način prikazivanja multimedijalnih objekata u okviru čitača.



Kada korisnik traži određenu stranicu, čitač na osnovu unete adrese kontaktira server na kome se stranica nalazi. Server ispituje MIME tip i ekstenziju traženog dokumenta. MIME tip predstavlja standardan način da se identifikuje sadržaj nekog dokumenta. On je integrisan u okviru HTTP protokola i sadrži informacije o tipu i podtipu sadržaja, na primer da je dokument tipa text/plain ili text/html ili image/gif ili application/x-director. Pomoću MIME tipa je omogućeno čitaču da prepozna kako će prikazati dobijene podatke.

Kada definiše MIME tip i ekstenziju dokumenta, server šalje klijentu dobijene podatke. Čitač na strani klijenta prepoznaje MIME tip dokumenta i ispituje da li se dobijeni tip može prikazati samostalno u okviru čitača ili je potreban poziv pomoćne aplikacije ili ugrađenog plug-in modula.

Pomoćne aplikacije su spoljašnje aplikacije u odnosu na čitač i služe za prikaz određene vrste podataka. One su odvojene od čitača. Njihova prednost je što sam čitač može biti manji i time se brže izvršava, a mane su što se ove aplikacije moraju odvojeno instalirati i podaci se prikazuju u odvojenom prozoru, pa je mala mogućnost integracije sa čitačem.

Plug-in moduli omogućavaju prikaz podataka u okviru samog čitača. Prednost korišćenja ovih modula je dobra integracija i povezanost sa čitačem, oni su obično besplatni, ali se s druge strane moraju podesiti od strane korisnika. Poželjno je da se omogući automatska instalacija.

Dodavanje zvuka

Koristi se <EMBED> tag. Ovaj tag podržava uobičajene formate kao što su .wav, .mid, .mp3, i .au. Osnovna struktura taga je sledeća:

```
<embed src="filename.ext" width="x" height="x" autoplay="x" hidden="x" loop="x" volume="x"></embed>
```

Mogući atributi u okviru ovog taga su

- SRC: njegova vrednost je URL (relativna ili apsolutna putanja) do audio dokumenta.
- CONTROLS: moguće vrednosti ovog atributa su CONSOLE i SMALLCONSOLE. Vrednost CONSOLE definiše 144 piksela za širinu i 60 za visinu. Vrednost SMALLCONSOLE definiše 144 i 15 piksela respektivno.
- HEIGHT and WIDTH: Za neke čitače ovo su obavezni atributi <EMBED> taga. Na primer njihovo izostavljanje u Netscape Communicator čitaču prouzrokuje nepredviđeb rad. Ako se za ove attribute definišu veće vrednosti od dimenzija konzole, sivi prostor se dodaje u dodatom prostoru.
- LOOP: Default vrednost ovog atributa je FALSE, što prouzrokuje da se audio dokument izvršava samo jednom. Ako se vrednost postavi na TRUE, dokument će se kontinuirano izvršavati. Takođe, moguće je specificirati i određeni broj koliko će se puta izvršiti dokument.

- **AUTOSTART:** Postavljena vrednost TRUE ovog atributa doprineće da se audio dokument izvršava kada se stranica učitava. Vrednost FALSE označava da korisnik mora aktivirati određeno dugme u okviru konzole da bi se dokument izvršio.
- **HIDDEN:** Postavljena vrednost TRUE ovog atributa doprineće da se sakrije konzola. Default vrednost je FALSE sa kojom je konzola vidljiva korisniku. Ako su vrednosti atributa HIDDEN i AUTOSTART postavljene na TRUE dobiće se isti efekat kao i korišćenjem taga BG SOUND da datim audiom dokumentom.
- **VOLUME:** Vrednost ovog atributa je broj između 0 i 100. Manji broj definiše tiši zvuk.
- **ALIGN:** Moguće vrednosti ovog atributa su LEFT i RIGHT.
- **HSPACE i VSPACE:** Slično kao kod taga ovi atributi se koriste da bi se definisao u pikselima horizontalni odnosno vertikalni prostor oko konzole.

Može se koristiti više <EMBED> tagova u okviru iste stranice, ali se preporučuje da nikada više od jednog ne bude u autoplay modu. Ne preporučuje se ni korišćenje .wav ili .au dokumenata zbog njihove veličine. Video dokumenti se prikazuju na sličan način, takođe korišćenjem <EMBED> taga i njegovih opisanih atributa- Primer korišćenja:

```
<embed src="http://www.davesite.com/webstation/html/davesite.wav"
loop="true" autoplay="false" width="145" height="60"></embed>
```

Kod video dokumenata ovaj tag ima i atribut *CONTROLLER*, sa mogućim vrednostima TRUE i FALSE, i definisanjem prikaza ili ne kontrolera sa funkcijama.



9. FORME I ELEMENTI FORME

Tagovi koji su do sada razmatrani omogućavaju samo da se oformi hipertekstualni dokument koji će se prikazati u okviru čitača na strani klijenta. Pored jednostavnog prikaza HTML omogućava i prenos podataka ka serveru odakle je stranica i učitana. Dobijeni podaci se mogu prihvatiti, zapamtiti ili editovati. Tehnologije koje se mogu primeniti na serverskoj strani aplikacije, kao što su ASP (Active Server Pages), PHP, JSP (Java Server Pages) i servleti, ... nisu tema ovog teksta, pa se neće dalje obrađivati.

Podaci se na klijentskoj strani unose pomoću koncepta formi (engl. form). Forma dopušta da se uspostavi komunikacija između korisnika i servera: korisnik popunjava formu i šalje je ka serveru. Ona omogućava da se u okviru servera pozove odreženi programski modul, koji će nakon osnovne obrade generisati novu HTML klijentsku stranicu. Forme se mogu koristiti u širokom spektru veom arazličitih aplikacija kao što su:

- upiti nad bazama podataka ili na pretraživačima;
- identifikacija pristupa određenom servisu servera;
- elektronska trgovina (prijem narudžbine i regulisanje načina plaćanja);
- obaveštavanje o promenljivim podacima (npr. vreme polaska aviona određenog datuma, i sl.), ...

Za implementaciju formi koristi se osnovni tag <FORM>. Opšti oblik ovog taga je:

```
<FORM>
<!-- definicija elemenata forme -->
...
</FORM>
```

Najvažnija dva atributa taga <FORM> su:

- atribut ACTION koji sadrži adresu (URL) programa na serveru;
- atribut METHOD kojim je opisana metoda prenosa argumenata programa. Ovaj atribut može imati vrednosti GET ili POST. Korišćenjem metoda GET vrednosti parametara koje se šalju ka

serveru navode se u okviru adrese programa koji se poziva. Na primer <http://server.com/ServerskiProgramskiModul?parametar1=88> gde je ServerskiProgramski Modul modul koji se poziva na serveru i kome se šalje vrednost promenljive parametar1 kao 88. Metoda GET ima ograničen broj parametara: mora važiti da je *dužina URL + dužina parametara* < 1KB. Metod POST vrednosti elemenata forme šalje u okviru zaglavlja poruke koju šalje na server, tako da nisu vidljivi korisniku i nema pgraničenja u količini podataka. Programski modul na serveru trebalo bi da zna kojim metodom prima podatke od strane klijenta. U praksi češće se koristi metod POST, posebno za podatke koji ne bi trebalo da budu javni, kao što su korisničko ime i šifra, matični broj, broj kreditne kartice,...

Primer HTML koda pomoću koga se definiše forma je:

```
<form action="Primer.html" method="post">
```

```
...
```

```
<!----- definicija elemenata forme ----->
```

```
</form>
```

Kada se izvrše akcija sa forme iz prethodnog primera na serveru će se pozvati stranica *Primer.html*, i to parametri forme će proslediti svoje vrednosti pomoću POST metoda.

U okviru same forme definišu se različiti elementi forme. Oni obuhvataju tekstualna polja za unos podataka, tekst prostore za komentare, polja za potvrdu, opadajuće liste, različite vrste dugmadi. Treba napomenuti da se jedino u okviru jedne forme ne može definisati druga forma. Već je napomenuto da elementi forme služe za prenos podataka na serversku stranu. Zato je potrebno da svaki element forme, pored tipa, ima i definisane dve karakteristike. To je ime elementa forme i njegova vrednost. Ove dve karakteristike se definišu pomoću atributa *name* i *value*, koji su zajednički za skoro sve tagove koji definišu elemente forme.

U okviru forme moguće je definisati:

- komandna dugmad

- tekst polja
- tekst prostore
- polja za potvrdu
- radio dugmad
- opadajuću listu
- skrivene promenljive

Komandno dugme

U okviru forme je moguće definisati više vrste dugmadi za različitim funkcijama. Sve vrste imaju zajednički tag `<INPUT>`, u okviru koga se pomoću atributa *type* definiše funkcija. Najčešće korišćene vrednosti atributa *type* su `button`, `reset` i `submit`. Funkcije i upotrebe ovih vrednosti su opisane u daljem tekstu:

- `button` – definiše dugme opšteg tipa. Pritiskom na dugme koje je definisano pomoću ove vrednosti korisnik izvršava akciju koja se mora definisati u okviru samog taga. Najčešće je to Java Script funkcija koja izvršava određene provere pre samog slanja podataka na server. Primer koda za ovu vrstu dugmeta je

```
<input type="button" name="Proveri" onClick="funkcijaZaProveru()">
```

Na formi se pojavljuje dugme koje ima ispisan naziv “Proveri” i kada korisnik klikne na ovo dugme izvršava se Java Script funkcija `funkcijaZaProveru()` koja mora biti definisana u okviru iste HTML stranice.

- `submit` - definiše dugme pomoću koga se podaci šalju na server. Podaci se šalju pomoću metoda koji je definisan u okviru `<FORM>` taga

```
<INPUT TYPE="SUBMIT" NAME="Posalji">
```

Sada nije potrebno definisati nijednu funkciju koja će se izvršiti pritiskom na dugme, jer se podrazumevana akcija poziv serverskog programskog modula.

- `reset` – definiše dugme koje postavlja sve vrednosti elemenata forme na predefinisane vrednosti

```
<INPUT TYPE="RESET" NAME="Ponisti">
```

I u ovom slučaju nije potrebno definisati novu funkciju, jer je podrazumevana akcija poništavanje novounetih vrednosti u okviru forme

Tekst polja i polje za šifru

Ovo je jedan od osnovnih i najčešće korišćenih elemenata forme. Pomoću njega korisnik unosi tekstualne informacije koje se kasnije šalju na server, kao što su ime, prezime, mesto stanovanja, ... Takođe se koristi pomoću taga `<INPUT>`, s tim što se za razliku od komandnih dugmadi koristi njegov atribut *type* sa definisanom vrednošću `text`. Osnovni oblik ovog elementa je:

```
<INPUT TYPE="text" NAME="ime_prez" VALUE="Unesite ime i prezime" SIZE="30">
```

U primeru su pored atributa *type* korišćeni i atributi *value* i *size*. Pomoću atributa *value* definiše se početni tekst koji će se pojaviti u tekst polju, a pomoću atributa *size* se zadaje maksimalan broj karaktera koji se mogu uneti u okviru ovog elementa od strane korisnika. Vrednost atributa *name* definiše samo ime elementa pomoću koga drugi elementi aplikacije (JavaScript, serverski deo aplikacije) mogu čitati ili menjati unetu vrednost u okviru tekst polja. Treba naglasiti da su sva tri navedena atributa opcionalna i da se mogu i izostaviti.

Ako se želi da se zamaskira tekst koji korisnik unosi u tekst polje umesto vrednosti `text`, treba koristiti vrednost `password`. Izgled polja će biti isti kao kod običnog tekst polja, ali će svaki uneti znak biti prikazan u obliku `*`. Sam čitač pamti unete karaktere i ostatku aplikacije se predaje tekst koji je korisnik uneo a ne niz simbola `*`.

Polje za potvrdu

Ovaj element se koristi kada je potrebno da korisnik potvrdi određeno tvrđenje. Grafički prikaz predstavlja kvadrat koji korisnik pomoću miša ili tastature potvrđuje ili skida potvrdu. Polje za potvrdu se definiše korišćenjem taga `<INPUT type="checkbox">` i njegovog atributa *type* sa vrednošću `checkbox`. Primer korišćenja ovog elementa je:

Koje programske jezike poznajete?

- ☐ Paskal
- ☐ Java

Kod primera je:

```
<I>Koje programske jezike poznajete?</I><BR>
<UL type="square">
  <LI> <INPUT type="checkbox" name="PPpaskal"> &nbsp;Paskal</LI>
  <LI> <INPUT type="checkbox" name="PPJava"> &nbsp;Java</LI>
</UL>
```

U primeru je korišćen i atribut *name* zbog pristupa ostalih delova aplikacije ovom elementu, provere, odnosno, definisanja da li je polje za potvrdu potvrđeno ili ne. Ako postoji grupisano više polja za potvrdu, nijedan, jedan ili više polja mogu u istom trenutku biti potvrđeni.

Radio dugmad

Ponekad je potrebno da se korisniku ponudi više opcija za potvrdu, a da on ima mogućnost da potvrdi nijednu ili samo jednu mogućnost od ponuđenih. Ovakav element jezika se naziva radio dugmad i grafički su predstavljeni krugom koji korisnik može da potvrdi ili skine potvrdu pomoću miša ili tastature. Radio dugmad se definišu korišćenjem taga `<INPUT>` i njegovog atributa *type* sa vrednošću `radio`. Primer korišćenja ovog elementa je:

Imate li vlastiti racunar?

- ☒ DA
- ☐ NE

Kod primera je:

```
<I>Imate li vlastiti racunar?</I><BR>
<UL type="square">
```

```

<LI> <INPUT type="radio" name="racunar"> &nbsp; Da</LI>
<LI> <INPUT type="radio" name="racunar"> &nbsp; Ne</LI>
</UL>

```

U primeru je korišćen i atribut *name* zbog pristupa ostalih delova aplikacije ovom elementu, provere, odnosno, definisanja koje je radio dugme potvrđeno. Može se specificirati potvrda određenog radio dugmeta pomoću atributa *checked*.

Opadajuća lista

Opadajuća lista je HTML element koji korisniku omogućava izbor jedneod više mogućnosti. U jednom trenutku mu je prikazana samo izabrana mogućnost, ali klikom miša na strelicu koja je u okviru ovog elementa, korisnik dobija listu svih mogućnosti. Opadajuća lista se definiše pomoću taga `<SELECT>` i opcija `<OPTION>`. Polje `<SELECT>` opisuje izbor između mogućih vrednosti navedenih u okviru taga `<OPTION>`. Tag `<OPTION>` sa atributom `SELECTED` se uzima kao izabrana vrednost. Tag `<SELECT>` se realizuje kao padajući meni sa vrednostima zadatim u okviru taga `<OPTION>`. Atributi uz tag `<SELECT>` su:

- *size* kojim je opisan broj elementa u opcija koje će biti prikazane,
- *multiple* kojim se omogućava izbor više opcija u istom trenutku.

Primer za opisani tag je:

Moj omiljeni predmet na 1. godini je:



Kod primera je

```

<B><I>Moj omiljeni predmet na 1. godini je: </I></B>
<select name="predmeti">
  <option>Analiza 1</option>
  <option>Analiza 2</option>
  <option>ORT</option>
</select>

```

Tekst polje

Pored maksimalno jedne linije teksta koju korisnik može da unese u tekst polje, potrebno je i da postoji mogućnost da korisnik upiše više linija svog teksta, na primer svoje komentare. Tag `<TEXTAREA>` prikazuje tekst u

odvojenom tekst prozoru sa scrollbarovima. Dimenzije prozora su određene atributima *rows* (broj redova teksta) i *cols* (broj kolona). Atribut *name* dodeljuje simboličko ime području u kome se nalazi tekst. Opšti oblik ovog taga je

```
<TEXTAREA name="ime" rows="n" cols="m"> ... neki tekst ...  
</TEXTAREA>
```

Skrivena promenljiva

Često je potrebo u okviru stranice zapamtiti određene vrednosti koje ne bi trebalo prikazivati korisniku, na primer vrednost koja se proverava na serverskoj strani radi zaštite aplikacije. Ova mogućnost se izvršava pomoću taga `<INPUT>` i njegovog atributa *type* sa vrednošću `hidden`. Na primer:

```
<input type="hidden" name="ident" value="08100-OEM-38069">
```

Ovako definisanoj promenljivoj ostatak aplikacije prilazi pomoću imena kao i bilo kom drugom elementu forme.

Primeri za rad u laboratoriji

Primer 1. Primer osnovne upotrebe formi kod HTML jezika.

```
<html>
<head>
<title>Ime i prezime</title>
</head>
<body>
<form action="akcija" method=POST>
Unesite svoje ime: <input type=text name="name"><p>
Unesite svoje prezime: <input type=text name="prezime"><p>
<input type=submit>
</form>
</body>
</html>
```

Primer 2. Primer jedne HTML stranice sa osnovnim elementima forme.

```
<html>
<head>
<title>Forma za komentare</title>
</head>

<body>

<h1>Posaljite svoje komentare</h1>

<form action="comments.cgi" method=POST>

<p>Ime i prezime: <input name="name">

<p>Email adresa: <input type=text name="email" size=50>

<p>Komentar:

<textarea name="comments" rows=15 cols=70>
</textarea>
```

```
<input type=submit value="Posaljite komentar">  
<input type=reset value="Obrisite unete podatke">
```

```
</form>
```

```
</body>
```

```
</html>
```

Primer 3. Primer HTML stranice sa formom za narudžbine

```
<html>
```

```
<head>
```

```
<title>Restoran italijanske hrane</title>
```

```
</head>
```

```
<body>
```

```
<h1> Restoran italijanske hrane </h1>
```

```
<form action="order.cgi" method=POST>
```

```
<h2>Pice</h2>
```

```
<p>Koliko pica želite? <input name="numpizzas" value="0" size=3  
maxlength=3>
```

```
<p>
```

```
<input type=radio name="size" value="large" checked>Velika<br>
```

```
<input type=radio name="size" value="medium">Srednja<br>
```

```
<input type=radio name="size" value="small">Mala
```

```
<h3>Dodaci</h3>
```

```
<p>
```

```
<input type=checkbox name="topping" value="pepperoni">Ljuta  
paprika<br>
```

```
<input type=checkbox name="topping" value="sausage">Kobasice<br>
```

```
<input type=checkbox name="topping" value="mushroom">Pecurke<br>
```

```
<input type=checkbox name="topping" value="peppers">Kecap<br>
<input type=checkbox name="topping" value="onion">Jaje<br>
<input type=checkbox name="topping" value="olives">Masline<br>
```

```
<p>Ime: <input type=text name="name">
<p>Broj telefona: <input type=text name="phone">
<p>Adresa:
<textarea name="address" rows=6 cols=50>
</textarea>
```

```
<p>Broj kreditne kartice: <input type=password name="creditcard"
size=20></p>
```

```
<input type=submit value="Posaljite narudzbini">
```

```
</form>
```

```
</body>
```

```
</html>
```

Primer 4. Primer jedne HTML stranice sa složenijom formom.

```
<html>
<head>
<title>Primer upotrebe formi</title>
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1250">
</head>

<body bgcolor="#FFFFFF" text="#000000">
<p><b>Primer upotrebe formi </b></p>
<hr>
<div align="left"><b>Lične informacije:</b> </div>
<form name="info" method="post" action="">

<table width="600" border="0" cellspacing="1" cellpadding="1">
<tr>
<td>Ime:</td>
```

```

        <td><input type="text" name="ime"></td>
        <td>Prezime:</td>
        <td><input type="text" name="prezime"></td>
    </tr>
    <tr>
        <td>Ulica i broj:</td>
        <td><input type="text" name="ulica" size="30"></td>
        <td>&nbsp;</td>
        <td>&nbsp;</td>
    </tr>
    <tr>
        <td>Grad:</td>
        <td><input type="text" name="grad" value="Beograd"></td>
        <td>Država:</td>
        <td>
            <select name="drzava">
                <option>Srbija i Crna Gora</option>
                <option>Turska</option>
                <option>Egipat</option>
                <option>Hrvatska</option>
                <option>Tunis</option>
                <option>Slovenija</option>
            </select>
        </td>
    </tr>
    <tr>
        <td>Poštanski broj:</td>
        <td><input type="text" name="postBroj" size="5"
maxlength="5"></td>
        <td>&nbsp;</td>
        <td>&nbsp;</td>
    </tr>
    <tr>
        <td rowspan="2">Pol:</td>
        <td>Muški <input type="radio" name="pol" value="pol"> </td>
        <td rowspan="2">&nbsp;</td>
        <td rowspan="2">&nbsp;</td>
    </tr>
    <tr>

```

```

        <td>Ženski      <input      type="radio"      name="pol"      value="pol"
checked></td>
    </tr>
    <tr>
        <td>Interesovanja:</td>
        <td colspan="5">
            <table width="100%" border="0" cellspacing="1" cellpadding="1">
                <tr>
                    <td width="10%">Sport</td>
                    <td width="10%"><input type="checkbox" name="sport"
value="sport"></td>
                    <td width="10%">Muzika</td>
                    <td width="10%"><input type="checkbox" name="muzika"
value="muzika" checked></td>
                    <td width="10%">Putovanja</td>
                    <td width="10%"><input type="checkbox" name="putovanja"
value="putovanja"></td>
                </tr>
            </table>
        </td>
    </tr>
    <tr>
        <td>Lozinka:</td>
        <td><input type="password" name="lozinka"></td>
        <td>&nbsp;</td>
        <td>&nbsp;</td>
    </tr>
    <tr>
        <td>Komentar:</td>
        <td><textarea      cols="30"      rows="5">Ovde      ispisati
komentar</textarea></td>
        <td><input      type="hidden"      name="skriveno"      value="OVO      JE
SKRIVENO POLJE"></td>
        <td>&nbsp;</td>
    </tr>
    <tr>
        <td colspan="6">
            <div align="center">
                <input type="submit" name="potvrđi" value="POTVRDI">
            </div>
        </td>
    </tr>

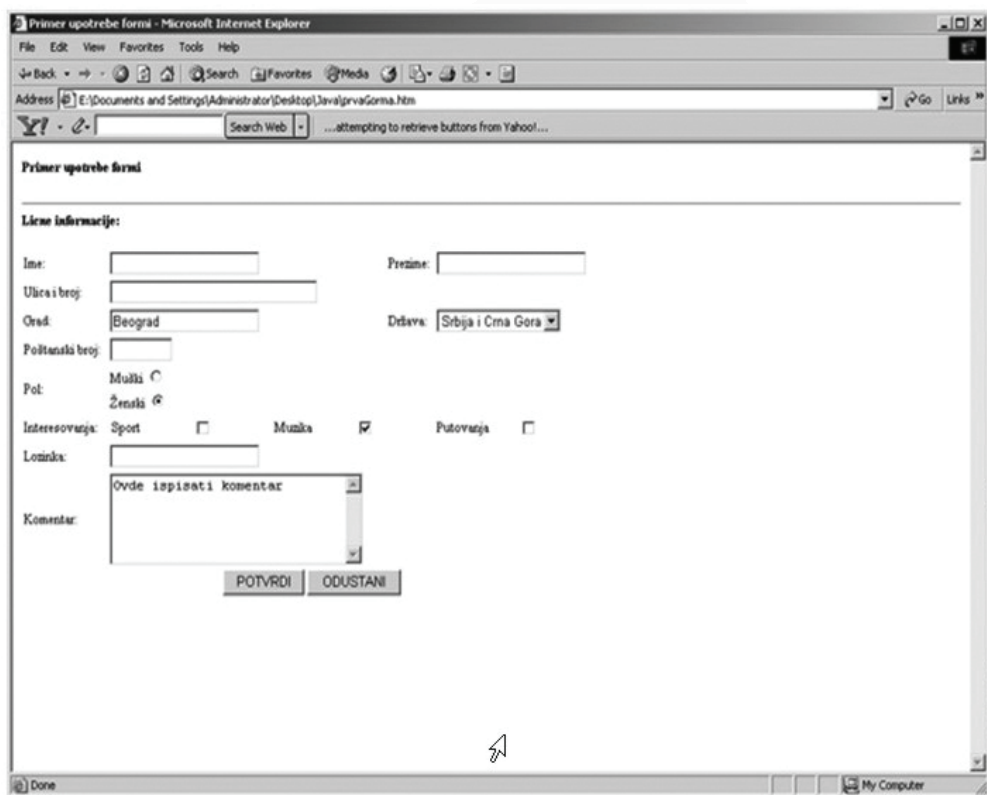
```

```

        <input type="reset" name="odustani" value="ODUSTANI">
    </div>
</td>
</tr>
</table>
</form>
</body>
</html>

```

Nakon pokretanja napisanog prograna, dobija se prozor koji predstavlja izgled HTML stranice u Internet Exploreru (slika 1).



Slika 1. Izgled HTML stranice u Internet Exploreru

Zadaci za samostalan rad

Zadatak 1. Napisati HTML kod koji opisuje stranicu kao što je na slici 2.

The screenshot shows a Microsoft Internet Explorer window titled "HTML FORME - Microsoft Internet Explorer". The address bar displays "C:\Documents and Settings\Administrator\Desktop\zadatak.htm". The form content is as follows:

OVA FORMA ŠALJE MAIL NA ADRESU: web@vets.edu.yu

Korisničko ime:

Lozinka:

Način plaćanja:

Naručujem: Knjigu ☒ CD ☐

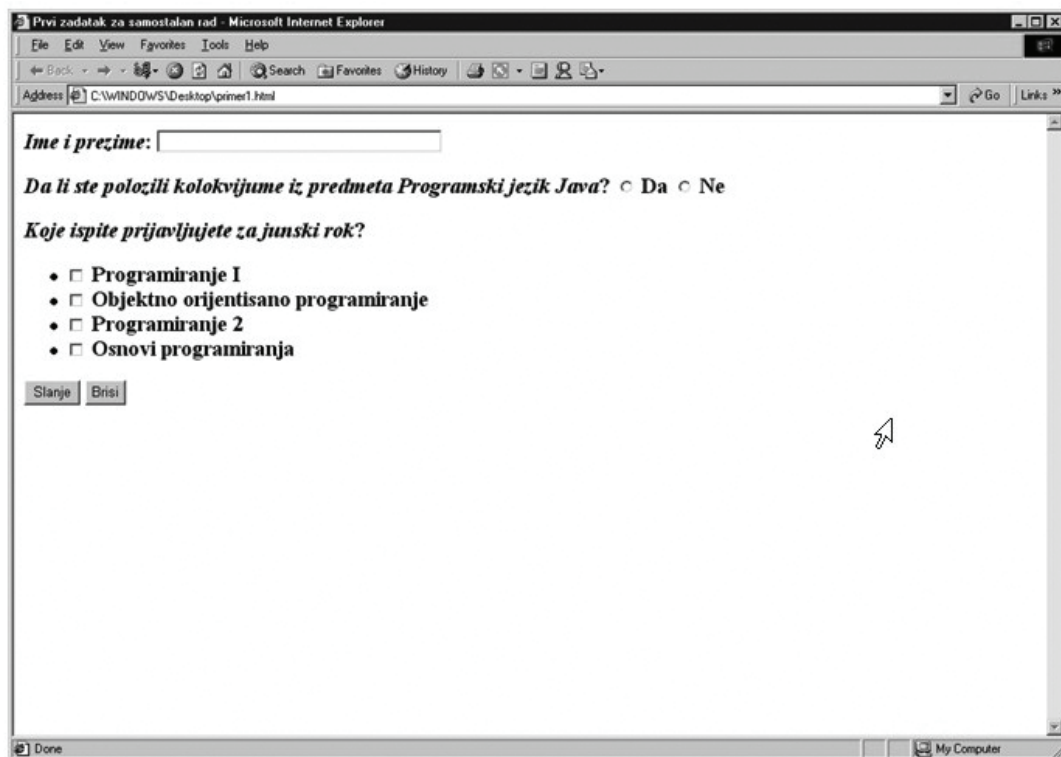
Komentar:

POTVRDI PONIŠTI

Napomena: akcija koju izvodi HTML forma je

<mailto:info@singidunum.ac.yu>

Zadatak 2. Napisati HTML kod koji opisuje stranicu kao što je na slici 3.



The screenshot shows a Microsoft Internet Explorer window titled "Prvi zadatak za samostalan rad - Microsoft Internet Explorer". The address bar shows the file path "C:\WINDOWS\Desktop\primer1.html". The form contains the following elements:

- A text input field for "Ime i prezime:".
- A radio button question: "Da li ste položili kolokvijume iz predmeta Programski jezik Java?" with options "Da" and "Ne".
- A text input field for "Koje ispite prijavljujete za junski rok?".
- A list of checkboxes for exam topics:
 - ☐ Programiranje I
 - ☐ Objektno orijentisano programiranje
 - ☐ Programiranje 2
 - ☐ Osnovi programiranja
- Two buttons at the bottom: "Slanje" and "Brisi".

A large, faint watermark of the "AKR SINGIDUN" logo is visible in the background of the page.



10. CSS (Cascading Style Sheets)

CSS (Cascading Style Sheets), se koriste u okviru HTML stranica da bi se smanjilo ispisivanje istog koda. Omogućavaju isti način formatiranja svih stranica u okviru nekog sajta, kao i jednostavnu promenu pojedinih parametara kao što su boja slova, vrsta slova, veličina slova, izgled tabele, pozadine... Sve ove osobine stranica se mogu čuvati u posebnim dokumentima i to na jednom mestu za sve stranice, pa se promenom jednog parametra na jednom mestu menja izgled svih stranica koje referišu promenjeni parametar.

CSS element

U opštem slučaju CSS element se sastoji od tri glavna dela

objekat {property : vrednost [;property : vrednost] }

gde je:

objekat - HTML element koji se želi promeniti

property - atribut objekta koji se želi promeniti

vrednost - vrednost atributa

CSS elementi se mogu definisati na dva načina, u okviru stranice, ili u okviru zasebnog dokumenta. Da bi se definisali u okviru stranice koristi se tag `<style>` i njegov atribut *type* sa vrednošću "text/css". U okviru ovog taga se definišu pojedinačni CSS elementi.

Primer osnovne upotrebe CSS elemenata u okviru HTML stranice je

```
<html>
<head>
<style type="text/css">
h1 {background-color: #00ff00}
</style>
</head>
<body>
<h1>Prvi naslov</h1>
<h1>Drugi naslov</h1>
<p>Tekst paragrafa</p>
</body>
</html>
```

Izgled stranice je



U primeru je korišćen CSS element

```
h1 {background-color: #00ff00}
```

gde h1 označava naslov nivoa 1, background-color pozadinu naslova nivoa 1 i vrednost #00ff00 zelenu boju. Korišćenjem ovog elementa svi naslovi nivoa 1 u okviru posmatrane stranice se prikazuju u pozadini sa zelenom bojom. Na ovom primeru se mogu uočiti prednosti korišćenja CSS elemenata. Ako bi u okviru stranice postojao veći broj naslova nivoa 1, i ako se želi da svi naslovi ovog nivoa budu u zelenoj pozadini, nije potrebno ništa više dodati datoj stranici, osim tagova samih naslova. Održavanje ovakvog koda je takođe mnogo jednostavnije i efikasnije. Ako je potrebno u nekom trenutku pozadinu naslova nivoa 1 umesto u zelenoj boji prikazati u crvenoj boji, tada treba samo promeniti vrednost u okviru CSS elementa na #ff0000 i svi naslovi će biti prikazani sa pozadinom u crvenoj boji, znači

```
h1 {background-color: #ff0000}
```

Ako se ne koriste mogućnosti CSS bilo bi potrebno promeniti boju pozadine u okviru svakog naslova koji se nalazi na stranici.

Pri definiciji CSS elementa naglašeno je da u okviru jednog objekta može da se navede više parova atribut-vrednost. U tom slučaju parovi se odvajaju pomoću znaka ; . Primer:

```
p {text-align:center;color:red}
```

gde je za objekat paragraf definisano poravnanje teksta po sredini i prikaz teksta u crvenoj boji.

Sličan deo koda se može napisati i na sledeći način

```
p
{
text-align: center;
color: black;
font-family: arial;
}
```

gde je pored atributa poravnanja teksta i boje slova, korišćen i atribut za vrstu fonta u okviru paragrafa.

Analogno, moguće je i da više objekata dele iste attribute sa istim vrednostima. Na primer:

```
h1,h2,h3,h4,h5,h6
{
color: blue
}
```

Na stranici gde se izvršava prethodni naslovi bilo kog nivoa biće ispisani plavom bojom.

Poziv CSS objekata iz posebnog dokumenta

Pored definisanja CSS elemenata u okviru html stranice, ovi elementi se mogu definisati i kasnije, pa koristiti i iz posebnih dokumenata. Na ovaj način se dobija još organizovaniji i efikasniji kod, jer sada više stranica, u principu i sve stranice u okviru aplikacije, deli više istih elemenata. Kao posledica ove osobine uniformnost aplikacije se dobija na veoma efikasan način, a sve osobine stranica se čuvaju na jednom mestu.

Da bi se iskoristila ova karakteristika CSS-a potrebno je uraditi dve stvari. Prvo treba napraviti novi dokument koji će sadržavati niz željenih CSS elemenata. Dokument treba snimiti sa ekstenzijom css. Nakon toga u okviru

html stranice umesto dosadašnjih CSS elemenata treba definisati referencu na novi dokument. To se postiže pomoću taga `<link>` sa definisanom putanjom i imenom CSS dokumenta. Na primer ako se CSS dokument zove `primer.css` inalazi se u istom direktorijumu sa html stranicom, tada head deo html stranice treba da izgleda:

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

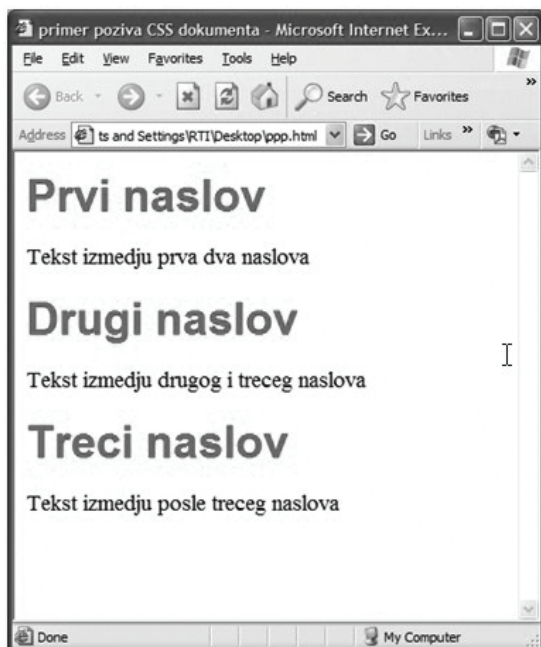
Neka je data html stranica

```
<HTML>
<head>
<TITLE> primer poziva CSS dokumenta </TITLE>
<head>
<link rel="stylesheet" type="text/css" href="test.css" />
</head>
<BODY>
  <H1>Prvi naslov</H1>
  <P>Tekst izmedju prva dva naslova</P>
  <H1>Drugi naslov</H1>
  <P>Tekst izmedju drugog i treceg naslova</P>
  <H1>Treci naslov</H1>
  <P>Tekst izmedju posle treceg naslova</P>
</BODY>
</HTML>
```

I neka se u istom direktorijumu nalazi dokument pod imenom `test.css` koji ima sledeći kod:

```
H1 {
  font-family: sans-serif;
  color: #ff0000;
}
```

Tada se u okviru čitača dobija sledeći izgled stranice



Klase objekata

Nekada je potrebno da određeni CSS objekat može da ima više vrednosti za isti atribut i da su te različite vrednosti potrebne na različitim mestima u html stranici. Na primer stranica se sastoji od dvadeset paragrafa, i potrebno je da kod osamnaest paragrafa tekst bude poravnat uz desnu marginu, a kod dva da bude centriran. Da bi i ovakvi slučajevi mogli da se reše moguće je u okviru jednog objekta definisati više klasa koje definišu različite osobine istog objekta. U okviru CSS objekta klasa se definiše navođenjem simbola . i imena klase. Primer za dve klase koje poravnavaju tekst uz desnu marginu i centrirano je

```
p.desno {text-align: right}
p.centrirano {text-align: center}
```

Obe klase pripadaju objektu p, koji definiše paragraf. Prva je klasa *desno* i definiše da tekst u paragrafu bude poravnat uz desnu marginu, a druga je klasa *centrirano* i definiše da tekst u paragrafu bude poravnat po sredini stranice.

Klasu određenog objekta pozivamo tako što u željenom tagu navodimo kod *class=imeKlase*. Tako će na našoj primer stranici paragrafi koje želimo da poravnamo uz desnu marginu imati strukturu:

```
<p class="desno">
```

Tekst koji treba da bude uz desnu marginu

```
</p>
```

a paragrafi čiji tekst treba da bude centriran imati strukturu

```
<p class="centrirano">
```

Tekst koji treba da bude centriran

```
</p>
```

Neka su definisane četiri klase za formatiranje tabele, koje se razlikuju po tome gde se nalazi granica. Kod tako defnisanih klasa je

```
.grIznad{
```

```
border-bottom: #999999 0px solid;
```

```
border-left: #999999 0px solid;
```

```
border-right: #999999 0px solid;
```

```
border-top: #999999 1px solid;
```

```
}
```

```
.grLevo{
```

```
border-bottom: #999999 0px solid;
```

```
border-left: #999999 1px solid;
```

```
border-right: #999999 0px solid;
```

```
border-top: #999999 0px solid;
```

```
}
```

```
.grIspod{
```

```
border-bottom: #999999 1px solid;
```

```
border-left: #999999 0px solid;
```

```
border-right: #999999 0px solid;
```

```
border-top: #999999 0px solid;
```

```
}
```

```
.grDesno{
```

```
border-bottom: #999999 0px solid;
```

```
border-left: #999999 0px solid;
```

```
border-right: #999999 1px solid;
```

```
border-top: #999999 0px solid;
```

```
}
```

Ako se za određenu tabelu želi granica samo na gornjoj ivici tada će struktura ćelije tabele biti

```
<td class=" grIznad ">
```

Ćelija sa definisanom granicom na gornjoj ivici

```
</td>
```

Kao primer upotrebe klasa može da posluži i sledeći primer dve klase *crnilink* i *zutilink*. Ove dve klase definišu različito prikazivanje linka. U jednom primeru lin će biti prikazan crnom bojom, a au drugom žutom bojom, uz prikaz teksta sa određenim fontom i veličinom slova od 10 piksela. Primer:

```
a.crnilink{
font-family: Verdana, Arial, Helvetica, sans-serif;
font-size: 10px;
color: #000000
}
a.zutilink{
font-family: Verdana, Arial, Helvetica, sans-serif;
font-size: 10px;
color: #ffff00
}
```

Pozivi ove dve definisane klase se realizuju na sledeći način:

```
<a href="Primer.html" class=" crnilink "> Link prikazan crnom bojom</a>
<a href="Primer.html" class=" zutilink "> Link prikazan zutom bojom</a>
```

Atributi CSS elemenata

U dosadašnjim primerima korišćeno je nekoliko CSS atributa kao što su boja, vrsta slova, veličina slova, granica tabela, poravnanje, ..., ali spisak atributa nije naveden deklarativno. Zato se u okviru ovog odeljka navode najvažniji atributi i njihova upotreba. Teorijski, navedene attribute je moguće koristiti u okviru čitača Netscape 4 ili kasnije verzije i Internet Explorer 4 ili kasnije verzije.

Do sada je navedeno da je svrha korišćenja CSS elemenata realizacija efekata koji se ne mogu dobiti korišćenjem uobičajenog HTML koda. Treba napomenuti da su u ovom odeljku naveden samo jedan deo atributa, dok se

lista svih atributa može naći na adresi <http://www.w3.org/TR/1998/REC-CSS2-19980512>.

border-color

Koristi se da bi se definisala boja ivice oko selektovanog elementa. Ivice, kao u prethodnom primeru, za svaku stranu se mogu postaviti posebno korišćenjem atributa `border-top-color`, `border-right-color`, `border-bottom-color` i `border-left-color`. Primer:

```
.errorMessage { border-color: #FF0000; }
```

border-style

Koristi se da bi se definisao stil ivica oko selektovanog elementa. Podržani stilovi su: `double`, `groove`, `inset`, `none`, `outset`, `ridge`, `solid`. Stil ivice, za svaku stranu se može postaviti posebno korišćenjem atributa `border-top-style`, `border-right-style`, `border-bottom-style`, and `border-left-style`. Primer:

```
.framed { border-style: ridge; }
```

border-width

Koristi se da bi se definisala širina ivica oko selektovanog elementa. Širine ivice za svaku stranu se mogu postaviti posebno korišćenjem atributa `border-top-width`, `border-right-width`, `border-bottom-width`, and `border-left-width`. Primer:

```
.thickborder { border-width: 3px; }
```

background-color

Koristi se da bi se definisala boja pozadine određenog elementa. Vrednost ovog atributa se definiše na standardni način i može biti i `transparent`.

Primer:

```
highlighted { background-color: yellow; }
```

background-image

Koristi se da bi se definisala slika kao pozadina određenog elementa.

Upobičajeno je da vrednost atributa bude URL do određene slike. Primer:

```
.flowery { background-image: url(flowers.gif); }
```

background-repeat

Koristi se da bi se definisao metod ponavljanja slika u pozadini određenog elementa. Moguće vrednosti su za ponavljanje horizontalno (`repeat-x`), vertikalno (`repeat-y`), u oba pravca (`repeat`), ili bez ponavljanja (`no-repeat`).

Primer:

```
TD.sidebar {  
  background-image: url(bar.gif);  
  background-repeat: repeat-y;  
}
```

color

Koristi se da bi se definisala boja određenog elementa, obično boja teksta unutar elementa. Primer:

```
P.inverse {  
  color: white;  
  background-color: black;  
}
```

hover

Koristi se da bi se definisale karakteristike linka u trenutku kada se kursor miša nalazi na njemu. Primer:

```
A:hover { color: #00CCFF; }
```

list-style-type

Koristi se da bi se definisao način prikaza stavke nenumerisane ili numerisane liste. Vrednosti atributa su tipovi ove dve vrste liste. Primer:

```
UL { list-style-type: square; }
```

list-style-image

Koristi se da bi se definisala slika kao oznaka stavke liste za sve tri vrste liste. Primer: listItem1 { list-style-image: url(fileicon.gif); }

font-family

Koristi se da bi se definisao tip fonta za izabrani element. Funkcija ovog atributa odgovara funkciji atributa *face* u okviru taga . Vrednost atributa je lista imena fontova, sortiranih po prioritetu i odvojenih zarezima. Poslednja stavka u listi trebalo bi da bude generičko ime fonta: serif, sans-serif, cursive, fantasy, ili monospace. Primer::

```
P { font-family: "Times New Roman", Times, serif; }
```

font-size

Koristi se da bi se definisala veličina fonta za izabrani element. Moguće vrednosti atributa su sledeće: xx-small, x-small, small, medium, large, x-large, xx-large. Takođe mogu se koristiti vrednosti larger i smaller da bi se uspostavio relativan odnos prema veličini slova kod prethodnih elemenata. Takođe, može se specificirati veličina fonta i u pikselima (px), tačkama (pt), ili kao procenat od veličine slova kod prethodnih elemenata (%). Primer:

```
.navbar { font-size: 8px; }
```

font-style

Koristi se da bi se definisao način prikaza teksta za izabrani element, i to normalan (vrednost atributa normal), ili italik (italic). Primer:

```
EM { font-style: italic }
```

font-weight

Koristi se da bi se definisala debljina slova u izabranom elementu. Većina čitača podržava jedino normal i bold vrednosti, ali CSS specifikacija definiše sledeće vrednosti: bold, bolder, lighter, normal, 100, 200, 300, 400, 500, 600, 700, 800, 900. Vrednost normal odgovara vrednosti 400, dok vrednost bold odgovara vrednosti 700. Primer:

```
B { font-weight: bolder; }
```

letter-spacing

Koristi se da bi se definisao prostor između karaktera kod izabranog elementa. Početna vrednost je default razdaljina, a pomoću ovog atributa se zadaju pozitivna ili negativna odstupanja od početne vrednosti. Odstupanje se definiše u tačkama ili pikselima (pt ili px) da bi se slova prikazivala bliže ili udaljenije. Primer:

```
.tight { letter-spacing: 2.0 px }
```

link

Koristi se da bi se definisale karakteristike linka. Primer:

```
A:link { color: #0000FF; }
```

text-align

Koristi se da bi se definisalo horizontalno poravnanje teksta selektovanog elementa i to levo, udesno, centrirano (left, right, center). Primer

```
.newscolumn { text-align: right; }
```

text-decoration

Koristi se da bi se definisali dodatni efekti teksta u izabranom elementu. Moguće vrednosti ovog atributa su podvučen tekst (underline), nadvučen tekst (overline), precrtan tekst (line-through), ili nijedan (none). Primer:

```
A:link { text-decoration: overline; }
```

line-height

Koristi se da bi se definisao razmak između linija teksta u okviru izabranog elementa. Moguće vrednosti ovog atributa su normal (default vrednost čitača), broj (gde je 1 vrednost single-spaced, 2 je double-spaced, i 1.5 je vrednost između prve dve), broj u merama (px ili pt), ili procenat (gde 100% odgovara vrednosti single-spaced).Primer:

```
.doublepace { line-height: 2; }
```

text-indent

Koristi se da bi se definisala razlika između prvog reda i ostatka teksta u izabranom elementu. Ako se atribut ima pozitivnu vrednost, prvi red će biti uvučen u odnosu na ostatak teksta, a ako ima negativnu vrednost ostatak teksta će biti uvučen u odnosu na prvi red. Vrednosti se izražavaju u tačkama ili pikselima (pt ili px), ili u procentima od širine stranice. Primer:

```
P.story { text-indent: 30 px; }
```

text-transform

Koristi se da bi se definisao način prikaza teksta u formi naslova, sa velikim početnim slovom svake reči u okviru izabranog dokumenta. Ako je vrednost atributa none tekst će se prikazati u obliku kakav je definisan u kodu stranice, a ako se koristi vrednost prvoslovo svake reči će biti prikazano velikim slovom, bez obzira kako je napisano u izvornom kodu stranice.

Primer:

```
H1 { text-transform: capitalize; }
```

vertical-align

Koristi se da bi se definisalo vertikalno poravnanje teksta izabranog elementa sa ostalim elementima. Moguće vrednosti ovog atributa su bottom ili top, ili u zavisnosti od okolnog teksta baseline, middle, sub, super, text-bottom, ili text-top. Primer:

```
.superscript { vertical-align: super; text-size: smaller }
```

visited

Koristi se da bi se definisale karakteristike linka koji je posećen. Primer:

```
A:visited { color: #FF00FF; }
```

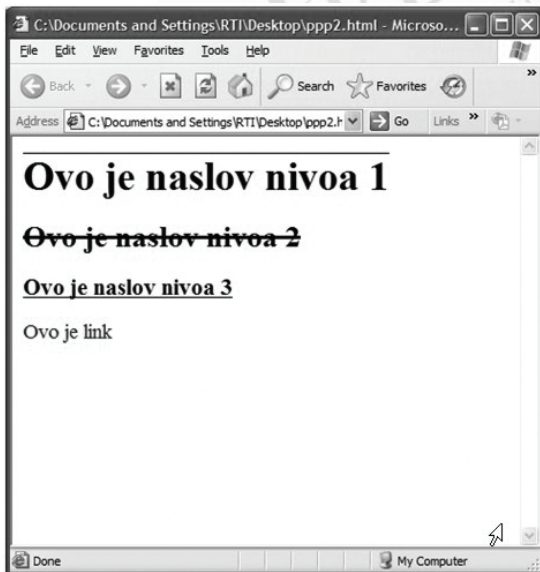


Primeri za rad u laboratoriji

Primer 1

```
<html>
<head>
<style type="text/css">
h1 {text-decoration: overline}
h2 {text-decoration: line-through}
h3 {text-decoration: underline}
a {text-decoration: none}
</style>
</head>
<body>
<h1>Ovo je naslov nivoa 1</h1>
<h2> Ovo je naslov nivoa 2</h2>
<h3> Ovo je naslov nivoa 3</h3>
<p><a href="http://www.w3schools.com/default.asp">Ovo je link</a></p>
</body>
</html>
```

Nakon izvršavanja ovog primera dobija se sledeći izgled stranice



Primer 2

```
<html>
<head>

<style type="text/css">
body
{
background-image:
url('bgdesert.jpg');
background-repeat: repeat-y
}
</style>

</head>

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

Primer 3

```
<html>
<head>
<style type="text/css">
p
{
font: italic small-caps 900 12px arial
}
</style>
</head>
<body>
<p>Primer paragrafa</p>
</body>
</html>
```

Primer 4

```
<html>
<head>
<style type="text/css">
p.leftmargin {margin-left: 2cm}
</style>
</head>
<body>
<p>Primer paragrafa bez specifikacije velicine margina </p>
<p class="leftmargin"> Primer paragrafa sa specifikacijom velicine leve
margine </p>
</body>
</html>
```

Primer 5

```
<html>
<head>
<style type="text/css">
ol.decimal {list-style-type: decimal}
ol.lroman {list-style-type: lower-roman}
ol.uroman {list-style-type: upper-roman}
ol.lalpha {list-style-type: lower-alpha}
ol.ualpha {list-style-type: upper-alpha}
</style>
</head>
<body>
<ol class="decimal">
<li>Coffee</li>
<li>Tea</li>
<li>Coca Cola</li>
</ol>
<ol class="lroman">
<li>Coffee</li>
<li>Tea</li>
<li>Coca Cola</li>
</ol>
<ol class="uroman">
```

```
<li>Coffee</li>
<li>Tea</li>
<li>Coca Cola</li>
</ol>
<ol class="lalpha">
<li>Coffee</li>
<li>Tea</li>
<li>Coca Cola</li>
</ol>
<ol class="ualpha">
<li>Coffee</li>
<li>Tea</li>
<li>Coca Cola</li>
</ol>
</body>
</html>
```

Primer 6

```
<html>
<body>
<p>Predjite misem preko delova teksta da bi ste uocili promenu oblika
kurzora:</p>
<span style="cursor:auto">
Auto</span><br />
<span style="cursor:crosshair">
Crosshair</span><br />
<span style="cursor:default">
Default</span><br />
<span style="cursor:pointer">
Pointer</span><br />
<span style="cursor:move">
Move</span><br />
<span style="cursor:e-resize">
e-resize</span><br />
<span style="cursor:ne-resize">
ne-resize</span><br />
<span style="cursor:nw-resize">
nw-resize</span><br />
```

```

<span style="cursor:n-resize">
n-resize</span><br />
<span style="cursor:se-resize">
se-resize</span><br />
<span style="cursor:sw-resize">
sw-resize</span><br />
<span style="cursor:s-resize">
s-resize</span><br />
<span style="cursor:w-resize">
w-resize</span><br />
<span style="cursor:text">
text</span><br />
<span style="cursor:wait">
wait</span><br />
<span style="cursor:help">
help</span>
</body>
</html>

```

Primer 7

```

<HTML>
<HEAD>
<TITLE> Jednostavan primer </TITLE>
<STYLE TYPE="text/css">
<!--
H1 {
font-family: sans-serif;
color: #3366CC;
}
-->
</STYLE>
</HEAD>
<BODY>
<H1>Prvi naslov</H1>
<P>...</P>
<H1>Drugi naslov</H1>
<P>...</P>
<H1>Treci naslov</H1>

```

```
<P>...</P>
</BODY>
</HTML>
```

Primer 8

```
<HTML>
<HEAD>
<TITLE> Jednostavan primer kao primer 7 uz upotrebu spoljnog dokumenta
</TITLE>
<LINK REL="STYLESHEET" TYPE="text/css" HREF="primer8.css">
</HEAD>
<BODY>
<H1>Prvi naslov</H1>
<P>...</P>
<H1>Drugi naslov</H1>
<P>...</P>
<H1>Treci naslov</H1>
<P>...</P>
</BODY>
</HTML>
```

Dokument primer8.css je:

```
H1 {
  font-family: sans-serif;
  color: #3366CC;
}
```

Primer 9

Dokument primer9.css je:

```
ul
{
  list-style-type: none;
  padding: 0;
  margin: 0;
}
```

```
li
{
  background-image: url(arrow.gif);
```

```
background-repeat: no-repeat;
background-position: 0 .4em;
padding-left: .6em;
}
```

Kod HTML stranice je:

```
<HEAD>
<TITLE> Formatirane liste </TITLE>
<LINK REL="STYLESHEET" TYPE="text/css" HREF="primer9.css">
</HEAD>
<BODY>
<ul>
<li><a href="#">Mleko</a></li>
<li><a href="#">Jaja</a></li>
<li><a href="#">Sir</a></li>
<li><a href="#">Povrce</a></li>
<li><a href="#">Voce</a></li>
</ul>
</BODY>
</HTML>
```

Primer 10

Kod dokumenta primer10.css je:

```
#navcontainer ul
{
margin: 0;
padding: 0;
list-style-type: none;
}
```

```
#navcontainer li { margin: 0 0 .2em 0; }
```

```
#navcontainer a
{
display: block;
color: #FFF;
background-color: #036;
```

```
width: 9em;
padding: .2em .8em;
text-decoration: none;
}
```

```
#navcontainer a:hover
{
background-color: #369;
color: #FFF;
}
```

Kod HTML stranice je

```
<HEAD>
<TITLE> Formatirane liste </TITLE>
<LINK REL="STYLESHEET" TYPE="text/css" HREF="primer8.css">
</HEAD>
<BODY>
<div id="navcontainer">
<ul>
<li><a href="#">Mleko</a></li>
<li><a href="#">Jaja</a></li>
<li><a href="#">Sir</a></li>
<li><a href="#">Povrce</a></li>
<li><a href="#">Voce</a></li>
</ul>
</div>
</BODY>
</HTML>
```

11. JavaScript - UVOD

U dosadašnjem delu tekstu razmatrane su osnove programskog jezika HTML. Pomoću njega je moguće realizovati efektne strane koje se mogu izvršavati na klijentskoj mašini. Nedostatak ovakvih strana je nemogućnost dinamičke obrade unetih podataka od strane korisnika. Zato se sa razvojem Interneta došlo do zaključka da HTML postaje ograničavajući faktor i da je potrebna nova tehnologija za realizaciju dinamičkih delova aplikacije.

Prvi pokušaj je bio pomoću serverskih komponenti, od kojih je najpopularnija bila CGI (Common Gateway Interface). I danas je moguće pretraživajući Internet pronaći stranicu koja se završava sa ekstenzijom cgi. Ova vrsta tehnologije je u početku dala zadovoljavajuće rezultati, jer su Web programeri dobili mogućnost obrade unetih korisnikovih podataka i različite odgovore u zavisnosti od tih podataka. Ipak, problem je predstavljala česta klijent-server komunikacija. Primer takve komunikacije može da bude forma u kojoj korisnik treba da unese datum svog rođenja. Korisnik može da pogreši i da u tekst polje predviđeno za mesec unese dan rođenja, na primer 28. Ovaj podatak se šalje na server, tamo serverska komponenta pre unosa u bazu proverava podatke i utvrđuje da je došlo do greške. Korisniku se šalje obaveštenje da je pogrešio i da mora ponovo uneti vrednost za mesec. Znači, korisnik je pogrešio, čekao je da se podaci prenesu na server, čekao je da se tamo obrade, da se vrati odgovor i, konačno dobio je obaveštenje da je pogrešio.

Tako se došlo do zaključka da je potrebna tehnologija koja će doneti mogućnost obrade određenih podataka na klijentskoj strani, pa su decembra 1995.god, Netscape i Sun predstavili jezik JavaScript 1.0, originalno nazvan LiveScript. Kod pisan na tom jeziku je mogao da se izvršava u okviru Netscape Navigator 2 čitača. Već u početku je ovaj jezik omogućio ne samo formatiranje podataka na klijentskoj strani, već i obradu i dinamičko izvršavanje stranica. Treba napomenuti da je implementiran deo jezika koji se izvršavao na serverskoj strani, čime je omogućeno da se ista tehnologija koristi na obe strane aplikacije, ali ovaj deo JavaScript jezika nije dostigao veću popularnost i neće se razmatrati u okviru ovog teksta.

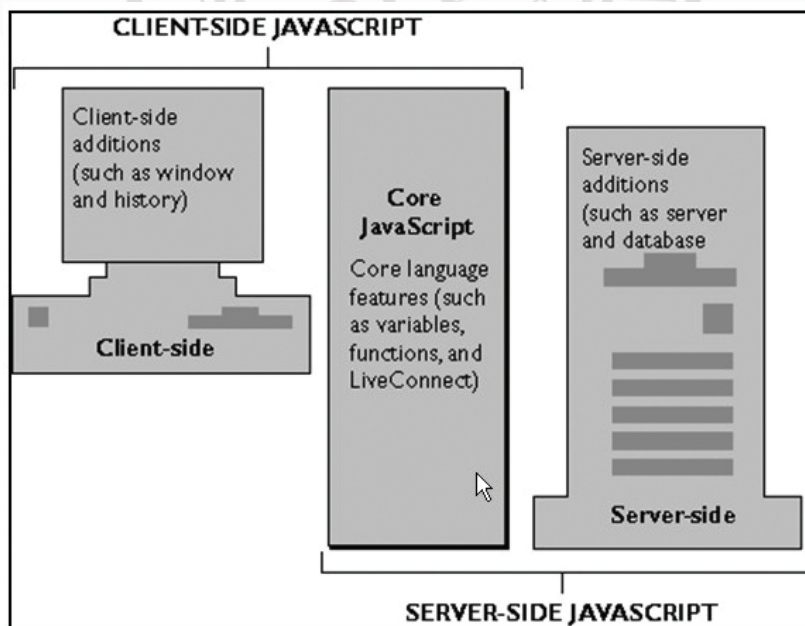
Sledeći korak u popularnosti JavaScript jezika je bila Microsoft-ova implementacija u okviru Internet Explorer 3 čitača, pri čemu je ova verzija od strane Microsoft-a nazvana JScript. JScript je bio baziran na javnoj dokumentaciji Netscape-a bio je skoro identičan JavaScript jeziku.

Konačno telo ECMA (European Computer Manufacturers Association) je

usvojilo standard za ovaj jezik nazvan ECMAScript 1.0, kasnije ISO/IEC 16262, pa je JavaScript postao Netscape-ova implementacija ovog standarda, a JScript Microsoft-ova. I danas Netscape-ova i Microsoft-ova verzija standarda su identične u preko 95% slučajeva.

Danas, JavaScript je objektno bazirani, platformski neutralan, višekorisnički jezik koji programeru omogućava mnogo veću funkcionalnost na klijentskoj strani. Primer naveden kao ilustracija nedostataka serverskih skriptova, sada se rešava pomoću JavaScripta koji na klijentskoj strani, onog trenutka kada korisnik unese pogrešnu vrednost za mesec, tu grešku odmah ustanovi i odmah prikaže obaveštenje korisniku. Korisnik je opet pogrešio, ali je sada korekciju dobio odmah i, samo može da bude iznenađen brzinom izvršavanja aplikacije.

Pojam objektno bazirani, znači da nisu svi koncepti objektno orijentisanih jezika realizovani u ovom jeziku, da je veoma limitiran rad sa nasleđivanjem, važenjem i funkcionalnošću samih objekata. Sa druge strane postoje hijerarhija ugrađenih objekata i oni se mogu koristiti, sa već definisanim metodama i osobinama (property). Ovakvim pristupom dobijeno je na jednostavnosti samog jezika, a pomoću ugrađenih objekata nije izgubljena potrebna funkcionalnost.



Opis JavaScript jezika

Navedeno je da je JavaScript platformski neutralan jezik (kao i HTML). Što znači da bi njegov kod (ako je pisan po standardu) trebalo da se izvršava u okviru čitača klijenta, bez obzira koja je hardverska mašina ili softversko okruženje u pitanju. Zato je i veličina programa pisanih u ovom jeziku dovoljno mala da može da se izvršava i na mašinama sa lošijim performansama.

Radi efikasnijeg i jednostavnijeg programiranja i održavanja samog koda JavaScript omogućava modularno programiranje, jer je moguće kreirati svoje sopstvene objekte, definisati opšte funkcije koje će realizovati uobičajene zadatke i čuvati i izvršavati kod pomoću posebnih dokumenata (sa .js ekstenzijom). Primer efikasne upotrebe ovakvog načina programiranja je funkcija koja proverava da li je adresa elektronske pošte korisnika u dozvoljenom formatu. Verovatno je da je takva funkcija potrebna više puta tokom jedne aplikacije. Ako se realizuje spoljašnji dokument koji će sadržavati ovu funkciju, koja kao argument prihvata uneti tekst, na različitim mestima upotrebe je dovoljno samo pozvati realizovanu funkciju. Ako se želi promeniti format adrese, sve promene se izvršavaju samo na jednom mestu, u JavaScript dokumentu.

Još jedna od prednosti JavaScript jezika je njegova integrisanost sa HTML-om. U okviru jedne stranice je moguće je na proizvoljan način kombinovati JavaScript i HTML kod. Takođe iz JavaScript-a moguće je generisati sam HTML kod, u zavisnosti od određene akcije korisnika.

Česta greška koja se pravi je izjednačavanje programskih jezika Jave i JavaScripta. U suštini radi se o dva potpuno različita koncepta. Dok je JavaScript jednostavan klijentski skript jezik, Java je složen, apsolutno objektno orijentisan, koji se trenutno može uspešno koristiti u skoro svim vrstama aplikacija. Tabela prikaz razlika između ova dva programska jezika je dat u Prilogu.



12. OSNOVE JavaScript JEZIKA

U ovom odeljku se razmatraju osnove samog JavaScript jezika.

Programski kod ovog jezika se može uključiti u okviru HTML stranice na dva načina. Prvi je direktnim pisanjem koda u okviru stranice. Tada se koristi HTML `<SCRIPT>` tag. Između ovog početnog i završnog taga nalazi se JavaScript kod. Uobičajeno je da se u početnom tagu definiše atribut *language* sa vrednošću JavaScript, iako trenutno ako se i ne navede, čitači podrazumevaju da je korišćeni skript jezik JavaScript. Znači struktura koda u okviru HTML stranice je:

```
<SCRIPT LANGUAGE="JavaScript">
...JavaScript kod...
</SCRIPT>
```

Drugi način korišćenja JavaScript koda u okviru HTML stranice je poziv js dokumenta. Opet se koristi `<SCRIPT>` tag, samo što se u okviru taga definiše i spoljašnji dokument u okviru atributa *src*. Struktura ove vrste koda je:

```
<SCRIPT LANGUAGE="JavaScript" SRC="JSkod.js">
</SCRIPT>
```

gde je JSkod.js dokument koji sadrži željene JavaScript funkcije.

Za razliku od mnogih viših jezika (kao što su C, C++, Java) u JavaScript jeziku nije potrebno na kraju svake naredbe pisati simbol “;”, mada neće dolaziti do greške ako se i upotrebljava. Jedini izuzetak, kada se obavezno mora koristiti tačka-zarez je ako se navodi više naredbi u istom redu. Tada se svaka pojedinačna naredba mora odvojiti sa tačkom-zarez.

Za komentar jedne linije koda se koristi oznaka `//`, na primer:

```
// komentar u jednoj liniji ...
```

Za komentarisanje više redova koriste se oznake `/*` za početak bloka pod komentarom i oznake `*/` za kraj bloka pod komentarom. Sve što se nalazi između ovih oznaka biće pod komentarom. Struktura ove vrste komentara:

```
/*
komentar u više redova...
*/
```

HTML tekst se prikazuje pomoću JavaScript koda na stranici korišćenjem metoda `document.write("neki tekst koji se prikazuje na stranici")`. Argument ovog metoda je string koji može biti proizvoljan HTML kod. Na primer:

```
<SCRIPT LANGUAGE="JavaScript">
document.write("<B>Prvi red</B><BR><I>Drugi red</I>")
</SCRIPT>
```

Dobija se sledeći tekst u okviru stranice:

Prvi red

Drugi red

Imena promenljivih mogu da sadrže brojeve i slova engleske abecede, ali prvi znak mora da bude slovo engleske abecede ili simbol “_”. JavaScript je case sensitive jezik, što znači da se velika i mala slova razlikuju, pa je promenljiva *Aaa* različita promenljiva od promenljive *AAA*. Takođe se ključne reči (`for`, `if`, `else`, `class`, `int...`) ne mogu da koriste u imenu promenljivih.

Tipovi podataka koji su podržani su celobrojni brojevi, racionalni brojevi, stringovi i logički tip.

Celobrojni brojevi se mogu koristiti sa brojem osnovom 10, sa osnovom 8 i osnovom 16. Uobičajena je predstava pomoću osnove 10. Ovakvi brojevi imaju cifre od 0 – 9, s tim da početna cifra ne sme biti 0. Brojevi prikazani u oktalanom brojnom sistemu sa osnovom 8 moraju počinjati sa cifrom 0, a ostale cifre su od 0 -7. Brojevi prikazani u heksadecimalnom brojnom sistemu sa osnovom 16 moraju počinjati sa 0x ili 0X, a ostale cifre su od 0 - 15, s tim da se cifre 10 - 15 prikazuju slovima A - F. Znači ako je broj prikazan sa 0716, u decimalnom brojnom sistemu to je broj $7 \cdot 8^2 + 1 \cdot 8 + 6 = 462$, a broj prikazan sa 0x716 u decimalnom brojnom sistemu je $7 \cdot 16^2 + 1 \cdot 16 + 6 = 1814$.

Racionalni brojevi se mogu prikazati na dva načina. Jedan je pomoću decimalne tačke, na primer 3.14, a drugi je eksponencijalna prezentacija, na primer 314E-2 ili 314e-2.

String predstavlja proizvoljan niz karaktera između navodnika (“neki tekst”) ili apostrofa (‘neki tekst’). U stringovima se mogu koristiti i specijalni karakteri kao što su:

`\b` = jedno mesto levo (backspace)

`\f` = jedan red nadole (form feed)

`\n` = početak novog reda (new line character)

`\r` = return (carriage return)

`\t` = tabulator (tab)

Logički tip podataka obuhvata dve vrednosti *true* (tačno) i *false* (netačno). Prilikom rada ako je potrebno može se izvršiti konverzija logičke vrednosti *true* u broj 1, odnosno vrednosti *false* u broj 0.

Takođe, nasuprot jezicima C, C++, Java, kod kojih je konverzija podataka iz jednog tipa podataka u drugi veoma važna i komplikovana, u JavaScript-u je to mnogo jednostavniji proces. Ako je potrebno, sam jezik automatski izvršava promenu jednog tipa u drugi, jer se dozvoljava da promenljiva ima različite tipove podataka u različito vreme izvršavanja programa. Na primer, sledeći deo kod:

```
a = 5;
```

```
b = 8;
```

```
b = “broj ” + a;
```

Tokom izvršavanja ovog programskog koda promenljiva *a* predstavlja broj 5, promenljiva *b* predstavlja broj 8, a u poslednjoj naredbi potrebno je sabrati vrednost tipa String i vrednost tipa broj, što je nemoguće izvršiti bez konverzije obe vrednosti u isti tip. Kako tip String ima prednost broj 5 se konvertuje u String “5” i izvršava sabiranje (konkatenacija) dva Stringa, tako da se na kraju u promenljivoj *b* nalazi String “broj 5”.



13. OPERATORI

Operatori su specijalni karakteri, koji definišu operaciju koja treba da se izvrši nad operandima, koji mogu biti promenljive, izrazi ili konstante. Znači da je željena operacija definisana operatorom. U teoriji postoji više podela operatora. Prema jednoj od njih operatori se klasifikuju prema broju operanada. Ako je u pitanju jedan operand operator se naziva unarni, binarni operator je sa dva operanda, sa tri ternarni. Druga podela je na prefiksne i postfiksne operatore, po tome da li dolaze pre operanda, odnosno posle njega. Najčešći slučaj je da se operator nalazi između dva operanda, i da predstavlja infiksni binarni operator.

Sa druge strane operatori se mogu podeliti i prema tipu podataka koji se koriste u okviru operacija. Takva podela obuhvata aritmetičke, na nivou bita, relacionalne i logičke operatore. U daljem tekstu će se detaljnije razmotriti svaka od navedenih grupa.

Aritmetički

Ovi operatori se koriste za matematičke operacije. Ukoliko je jedan od operanada tipa String za sv operatore, osim za sabiranje, pokušaće se da se izvede konverzija Stringa u broj i da se tako izvrši definisana operacija. Ako se ne uspe kao rezultat se dobija specijalna vrednost NaN (Not A Number). Kod sabiranja podatak koji nije tipa String konvertuje se u String i izvršava se sabiranje dva Stringa. Pregled svih aritmetičkih operatora je dat u sledećoj tabeli.

Operator	Opis	Operator	Opis
+	sabiranje	+=	sabiranje dodela
-	oduzimanje	-=	oduzimanje dodela
*	množenje	*=	množenje dodela
/	deljenje	/=	deljenje dodela
%	moduo	%=	Moduo dodela
++	inkrement	--	dekrement

Sabiranje, oduzimanje, množenje i deljenje obavljaju naznačene operacije nad numeričkim operandima. Unarni minus negira operand kojem prethodi. Sledi primer.

```
a = 1 + 1;  
b = a * 3;  
c = b / 4;  
d = b - a;  
e = -d;  
document.write("a, b, c, d, e, su ",a," ",b," ",c," ",d," ",e);
```

Nakon izvršavanja prethodnog primera dobija se sledeći tekst na ekranu:

a, b, c, d, e, su 2 6 1.5 4 -4

Operator moduo kao rezultat vraća ostatak pri deljenju. Na primer, ako je x jednako 12, dobijamo da je rezultat izvršavanja operacije

$x \% 10$

jednak 2.

Svi definisani binarni operatori koji se definišu u sledećoj formi:

var = var op expression;

mogu se prikazati i pomoću aritmetičkih operacija dodele na sledeći način:

var op= expression;

Tako se isti rezultat dobija izvršavanjem sledećeg izraza

a = a + 4;

kao i izraza

a += 4;

Operatori inkrement i dekrement su sažete forme notacije za operacije oduzimanja i dodavanja 1 operandu. Tako se operacija

x = x + 1;

može iskazati i u formi

x +=1;

a i pomoću novih operatora na sledeći način:

```
++x;
```

Analogno

```
x = x - 1;
```

ekvivalentno je sa

```
--x;
```

Važno je napomenuti da postoji velika razlika ako se operatori ++ i -- koriste u prefiksnoj (kada prethode operandu) ili postfiksnoj formi (kada slede posle operanda). U prvom slučaju operand se menja pre nego što se njegova vrednost koristi. Obrnuto, u postfiksnoj formi prvo se koristi vrednost operanda, pa se tek onda operand inkrementira ili dekrementira. Na primer:

```
x = 42;
y = ++x;
```

Kao rezultat izvršavanja prethodnog koda promenljivoj y je dodeljena vrednost 43, i promenljiva x ima situ vrednost. Drugi slučaj

```
x = 42;
y = x++;
```

Vrednost promenljive x je prvo dodeljena promenljivoj y, pa je nakon dodele vrednost x inkrementirana, pa je sada y jednako 42, a x je 43.

Operatori na nivou bita

Operatori iz ove grupe obavljaju operacije nad celobrojnim brojevima, i to dužine 32 bita. Ukoliko neki od operandi nije celobrojni broj dužine 32 bita, pokušaće se izvršiti konverzija u traženi tip, pa tek onda primeniti operacija. Ovi operatori celobrojne brojeve razmatraju na nivou bita i obavljaju operacije nad bitovima. Kako ovi operatori nemaju široku primenu prikazaće se tabelarno sa kratkim opisom funkcije.

Operator	Upotreba	Opis
Logičko I (AND)	$a \& b$	Rezultat se dobija 1, jedino ako su oba bita 1, u ostalim slučajevima rezultat je 0.
Logičko ILI (OR)	$a b$	Rezultat se dobija 0, jedino ako su oba bita 0, u ostalim slučajevima rezultat je 1.
Logičko ekskluzivno ILI (XOR)	$a \wedge b$	Rezultat se dobija 1, ako biti imaju različite vrednosti, u slučaju da imaju iste vrednosti, rezultat je 0.
Logičko NE (NOT)	$\sim a$	Komplementira bitove operanda a.
Pomeranje ulevo	$a \ll b$	Pomera binarni sadržaj operanda a za b mesta ulevo. Prazna mesta popunjava sa vrednošću 0.
Pomeranje udesno sa znakom	$a \gg b$	Pomera binarni sadržaj operanda a za b mesta udesno. Prazna mesta popunjavju sa vrednošću najstarijeg bita.
Pomeranje udesno sa nulama	$a \ggg b$	Pomera binarni sadržaj operanda a za b mesta udesno. Prazna mesta popunjavju sa vrednošću 0.

Primeri upotrebe operatora na nivou bita

15 & 9 rezultat 9 ($1111 \& 1001 = 1001$)

15 | 9 rezultat 15 ($1111 | 1001 = 1111$)

15 ^ 9 rezultat 6 ($1111 \wedge 1001 = 0110$)

Logički operatori

Operatori iz ove grupe deluju na operande logičkog tipa, koji mogu imati samo vrednosti true i false. Svi oni kombinuju dva operanda logičkog tipa i kao rezultat vraćaju vrednost logičkog tipa. Ovi operatori imaju veliku primenu u okviru kontrolama toka.

Operator	Upotreba	Opis
I (&&)	<code>expr1 &&</code>	Rezultat se dobija true, jedino ako su oba operanda

	expr2	true, u ostalim slučajevima rezultat je false.
ILI ()	expr1 expr2	Rezultat se dobija false, jedino ako su oba operanda false, u ostalim slučajevima rezultat je true.
NE (!)	!expr	Rezultat se dobija komplement od vrednosti operanada. Ako je operand true, rezultat je false, ako je operand false, rezultat je true

Primer upotrebe navedenih operatora je:

```

a = true;
b = false;
c = a || b;
d = a && b;
f = (!a && b) || (a && !b);
g = !a;
document.write( " a = " + a + "<BR>" );
document.write ( " b = " + b + "<BR> " );
document.write ( " c = " + c + "<BR> " );
document.write ( " d = " + d + "<BR> " );
document.write ( " f = " + f + "<BR> " );
document.write ( " g = " + g );

```

Rezultat izvršavanja prethodnog primera je:

```

a = true
b = false
c = true
d = false
f = true
g = false

```

Operatori poređenja

Operatori iz ove grupe obavljaju poređenje dve vrednosti i kao rezultat vraćaju vrednost logičkog tipa, tačno ili netačno, odnosno true ili false. Svaki dozvoljeni tip podataka, celobrojan, racionalni, karakter, String i logičkitip mogu se upoređivati koristeći operatore == i !=. Samo numerički tipovi koriste ostale operatore.

Operator	Upotreba	Opis
Jednakost (==)	Rezultat je true ako su operandi jednaki	$x == y$ rezultat je true ako su x i y jednaki.
Nejednakost (!=)	Rezultat je true ako su operandi različiti.	$x != y$ rezultat je true ako su x i y različiti.
Veće (>)	Rezultat je true ako je levi operand veći od desnog operanda.	$x > y$ rezultat je true ako je x veće od y .
Veće ili jednako (>=)	Rezultat je true ako je levi operand veći ili jednak desnom operandu	$x >= y$ rezultat je true ako je x veće ili jednako y .
Manje (<)	Rezultat je true ako je levi operand manji od desnog operanda	$x < y$ rezultat je true ako je x manje od y .
Manje ili jednako (<=)	Rezultat je true ako je levi operand manji ili jednak desnom operandu	$x <= y$ rezultat je true ako je x manje ili jednako y .
Jednako bez konverzije tipova (===)	Rezultat je true ako su operandi jednaki bez konverzije podataka	$x === y$ rezultat je true ako su x i y jednaki bez konverzije podataka
Različito bez konverzije tipova (!==)	Rezultat je true ako su operandi različiti bez konverzije podataka	$x !== y$ rezultat je true ako su x i y različiti bez konverzije podataka

Operatori `==` i `!=` obavljaju potrebnu konverziju podataka pre poređenja, ukoliko su operandi različitog tipa. Znači za ove operatore vrednosti `5` i `"5"` su iste, pa će posle njihovog poređenja rezultat sa operatorom `==` biti `true`, a sa operatorom `!=` `false`.

S druge strane operatori `===` i `!==` **ne** obavljaju potrebnu konverziju podataka pre poređenja, ukoliko su operandi različitog tipa. Znači za ove operatore vrednosti `5` i `"5"` su različite, pa će posle njihovog poređenja rezultat sa operatorom `===` biti `false`, a sa operatorom `!==` `true`.

Primer korišćenja ovih operatora

```
a = 4;  
b = 1;  
c = a < b;  
d = a == b;  
document.write( " c = " + c + "BR" );  
document.write ( " d = " + d );
```

Rezultat izvršavanja prethodnog primera je

```
c = false  
d = false
```

Ternarni if-then-else operator

Generalna forma ovog operatora je:

`expression ? statement1 : statement2`

gde je izraz `expression` bilo koji izraz čiji rezultat je vrednost logičkog tipa. Ako je rezultat izraza `true`, onda se izvršava `statement1`, u suprotnom `statement2`. Primer:

```
ratio = denom == 0 ? 0 : num / denom;
```

U prethodnom primeru `ratio` dobija vrednost 0 ili `num/denom` u zavisnosti da li je `denom` jednak 0 ili nije.



14. KONTROLE TOKA

Kontrola toka omogućava tok programa željenom putanjom u zavisnosti od određenih uslova. Osnovne izjave kontrole toka koje se koriste u JavaScriptu su veoma slične osnovnim kontrolama toka u višim programskim jezicima.

if-else

if-else konstrukcija omogućava izvršenje odredjenog bloka instrukcija ako je uslov konstrukcije ispunjen. Opšti oblik konstrukcije je:

```
if (boolean_izraz) blok1;  
[else blok2;]
```

gde je else klauzula opciona a boolean_izraz može biti bilo koji izraz čiji rezultat je vrednost logičkog tipa. blok1 i blok2 su proizvoljni blokovi instrukcija, koji mogu da budu i samo jedna instrukcija. Takođe, svaki od blokova, bilo u if ili u else delu može biti nova if-else konstrukcija. Primer upotrebe ove konstrukcije je

```
if (x == 8) {  
    y=x;  
} else {  
    z=x;  
    y=y*x  
}
```

U datom primeru ako promenljiva x ima vrednost 8 promenljiva y će dobiti vrednost x, u suprotnom slučaju promenljiva z će dobiti vrednost x, a promenljiva y vrednost y puta x.

switch

Kod if-else kontrole toka else deo može biti bilo koja JavaScript naredba, pa i nova if-else konstrukcija. Primer gde je potrebno primeniti više ugnježđenih if-else konstrukcija može biti sledeći. Promenljiva mesec sadrži redni broj meseca u godini, i potrebno je na osnovu njega u promenljivu ime_meseca upisati ime definisanog meseca. Rešenje ovog problema pomoću if-else konstrukcije je:

```

if (mesec == 1)
    ime_meseca = "Januar"
else if (mesec == 2)
    ime_meseca = "Februar"
else if (mesec == 3)
    ime_meseca = "Mart"
else if (mesec == 4)
    ime_meseca = "Maj"
else
    ....
else if (mesec == 12)
    ime_meseca = "Decembar"

```

Može se primetiti da je dobijeni kod dosta složen i nepregledan. Za rešavanje ovog i sličnih problema može se koristiti switch konstrukcija. Opšti oblik ove naredbe je:

```

switch (izraz) {
    case vr1: blok1; [break];
    ...
    case vrN: blokN; [break];
    [default: blok_def]
}

```

Pri tome izraz mora biti bilo kog prostog tipa, a vrednosti specificirane u case klauzulama (vr1, vr2, ... , vrN) moraju biti kompatibilnog tipa i ne smeju se duplirati. Svi navedeni blokovi instrukcija kao i break klauzule su opcioni.

Na početku switch-a se izračunava vrednost izraza izraz i dobijena vrednost se poredi sa vrednostima vr1, vr2 ... vrN. Tamo gde dodje do uparivanja izvršava se blok instrukcija u odgovarajućem case delu naredbe. break dovodi do skoka na kraj tekućeg bloka što u ovom slučaju znači na kraj switch konstrukcije. Ukoliko se vrednost izraza izraz ne nalazi medju vrednostima vr1,..., vrN tada se izvršava blok naredbi blok_def;

Treba napomenuti da ukoliko se izostavi neka od break klauzula dolazi do propagiranja kontrole u sledeći case.

Navedeni primer sa imenom meseca sada izgleda ovako:

```

switch(mesec) {
    case 1: ime_meseca = "Januar"; break;
    case 3: ime_meseca = " Mart"; break;
    case 5: ime_meseca = "Maj"; break;
    case 7: ime_meseca = "Jul"; break;
    case 8: ime_meseca = "Avgust"; break;
    case 10: ime_meseca = "Oktobar"; break;
    case 12: ime_meseca = "Decembar"; break;
    case 4: ime_meseca = " April "; break;
    case 6: ime_meseca = "Jun"; break;
    case 9: ime_meseca = "Septembar"; break;
    case 11: ime_meseca = "Novembar"; break;
    case 2: ime_meseca = " Februar ";
}

```

while petlja

while petlja funkcioniše na taj način što se blok instrukcija unutar nje ponovljeno izvršava sve dok je uslov za ostanak u petlji, koji se nalazi na ulasku u petlju, ispunjen. Opšti oblik petlje izgleda ovako:

```

[inicijalizacija;]
while(uslov_ostanka){
    telo_petlje;
}

```

Pre ulaska u petlju vrši se inicijalizacija promenljivih koje će biti korišćene bilo u uslovu, bilo u telu petlje, dok se u telu petlje mora nalaziti kod koji će menjati vrednost varijabli koje formiraju uslov za ostanak u petlji.

Jednostavan primer:

```

i=1
while(i<=10){
    document.writeln(i);
    i=i+1;
}

```

Nakon izvršavanja ovog primera dobiće se prikazani brojevi od 1 do 10.

Treba napomenuti da će se u slučaju da uslov petlje nije ispunjen kada se prvi put pristupi petlji telo petlje neće izvršiti nijednom. Znači ovo je petlja koja se izvršava nijednom, jednom ili više puta.

do-while petlja

Za razliku od prethodne petlje koja je imala uslov na svom početku, do-while petlja ima uslov na kraju. Prema tome, telo petlje će se sigurno izvršiti bar jednom. Opšti oblik petlje izgleda ovako:

```
[inicijalizacija]
do {
    telo_petlje
    [iteracija]
} while (uslov);
```

Prethodni primer realizovan pomoću do-while petlje izgleda ovako:

```
i=1
do {
    document.writeln(i);
    i=i+1;
} while(i<=10)
```

Ovo je primer petlje koja se izvršava jednom ili više puta.

for petlja

Opšti oblik for petlje izgleda ovako:

```
for( inicijalizacija; uslov; iteracija){
    telo_petlje;
}
```

For petlja je najkompaktnija i najčešće korišćena petlja. Unutar zaglavlja petlje se nalazi kod za inicijalizaciju, uslov terminacije petlje i kod za iteraciju. Ukoliko uslov na početku nije ispunjen telo petlje se neće izvršiti nijednom. Evo jednog jednostavnog primera:

```
for(i=0; i<10; i++){
    document.writeln(i);
}
```

I u ovom primeru dobiće se prikazani brojevi od 1 do 10.

Takođe, u vredna pažnje je i deklaracija promenljive `i` u zaglavlju petlje. Ta promenljiva je privremena promenljiva a blok u kome je definisana je blok u kome se nalazi `for` petlja.

break

`break` klauzula je svojevrsna zamena za `goto` klauzulu. Ona se koristi za skok na kraj bloka koji je označen labelom uz `break` ili na kraj bloka u kome se i `break` nalazi ako `break` stoji bez labele. Labele, pomoću kojih se označavaju blokovi, se formiraju kao i svi ostali identifikatori s tim što iza njih mora stajati dvotačka (:). Na primer, sledeći kod:

```
a: {
b:   {
c:   {
    document.writeln("pre break-a");
    break b;
    document.writeln("ovo nece biti prikazano");
    }
  }
  document.writeln("posle break-a");
}
```

Nakon izvršavanja navedenog primer dobija se:

```
pre break-a
posle break-a
```

return

`return` se koristi za povratak iz funkcije na mesto poziva. Ukoliko funkcija vraća neku vrednost tada `return` mora slediti izraz čiji je tip kompatibilan sa povratnim tipom funkcije. U suprotnom `return` izjava može stajati sama.

```
function kvadratBroja( x ){  
    return x * x  
}  
  
x = kvadratBroja(5)  
/* poziv funkcije */  
document.write("Kvadrat od 5 je " + x)
```

Kao rezultat poziva funkcije dobija se:

Kvadrat od 5 je 25

continue

Baš kao što je ponekad potrebno izaći iz petlje prerano, tako je ponekad potrebno ići na sledeću iteraciju petlje a da se deo koda pre njenog kraja ne izvrši. Za takve situacije se koristi continue. Sledeći primer ilustruje upotrebu continue naredbe:

```
for( int i=0; i<10; i++){  
    document.write(i+ " ");  
    if (i%2 ==0)  
        continue;  
    document.writeln(" <br>");  
}
```

Zahvaljujući continue naredbi nakon izvršavanja ovog primera dobija se:

0 1
2 3
4 5
6 7
8 9

Specijalne naredbe

Do sada navedene kontrole toka i petlje se mogu pronaći kod većine viših programskih jezika, U okviru JavaScript jezika postoje i neke toka koje su psecifične za ovaj programski jezik:

- *for...in:*

Izvršava iteraciju po specifičnoj promenljivoj za svaku osobinu (property) u okviru određenog objekta. Znači za svaku definisanu osobinu u okviru nekog objekta izvršava se niz naredbi definisan u okviru tela ove petlje. Primer:

```
function dump_props(obj, objName) {
    var result = "";
    for (var i in obj) {
        result += objName + "." + i + " = " + obj[i] + "<BR>";
    }
    result += "<HR>";
    return result;
}
```

- *function:*

Deklariše JavaScript funkciju sa specificiranim parametrima. Tipovi podataka mogućih parametara obuhvataju stringove, brojevi i objekte.

```
function ime([param1] [, param2] [... , paramN]) {
    izrazi
}
```

- *var:*

Deklariše promenljivu, opciono moguće je izvršiti i njenu inicijalizaciju.

```
var imePromenljive [= vrednost] [... , imePromenljive2 [= vrednost2]
]
```

- *with:*

Definiše tip objekta za niz izraza. U okviru izraza dodeljuje specifične vrednosti za određene osobine objekta. Na primer, matematičkim funkcijama mora prethoditi objekat Math. Sledeći primer podrazumeva Math ispred PI, COS() i SIN():

```
var a, x, y;
var r=10;
```

```
with (Math) {  
  a = PI * r * r;  
  x = r * cos(PI);  
  y = r * sin(PI/2);  
}
```



15. OBJEKTI Date i String

U okviru JavaScript jezika postoje ugrađeni određeni objekti koji umnogome olakšavaju pojedine akcije. U okviru ovog odeljka razmatraće se takvi objekti, njihova upotreba, njihovi metodi i korišćenje metoda.

Date objekat

Ovaj objekat se koristi kada je potrebno primeniti određene operacije u kojima se koriste vremenske promenljive. Svaki datum koji se pojavi u okviru nekog JavaScript programa se pamti kao broj koji predstavlja broj milisekundi između dobijenog datuma i ponoći 1. Januara 1970. god. po UTC vremenu. Na primer argument 5000 će kreirati datum koji predstavlja 5 sekundi posle ponoći 1/1/1970.

U programu kreiranje promenljive od ovog objekta se postiže na jedan od sledećih načina:

```
dateObjectIme = new Date()  
dateObjectIme = new Date("month day, year hours:minutes:seconds")  
dateObjectIme = new Date(year, month, day)  
dateObjectIme = new Date(year, month, day, hours, minutes, seconds)
```

Ako se koristi bez argumenata konstruktor `Date( )` kreira objekat koji predstavlja trenutno vreme i datum. Ako se prosleđuje jedan numerički argument, on se prihvata kao datum u milisekundama po ranije navedenoj konvenciji. Ako je string argument on predstavlja datum u formatu koji je prihvatljiv za računar na kome se izvršava aplikacija (tačan format se može dobiti izvršavanjem metoda `Date.parse( )`). U okviru konstruktora se može definisati između 2 i 7 numeričkih argumenata, koji onda specificiraju pojedinačna polja datuma i vremena. U tom slučaju, svi argumenti, osim prva dva koji predstavljaju godinu i mesec, su opcioni. Treba naglasiti da se vreme definiše na osnovu lokalne mašine, a ne UTC ili GMT vremena.

Konstruktor `Date( )` se može pozivati i u obliku funkcije, bez navođenja operatora `new`. U tom slučaju se ignorišu svi eventualni argumenti i kao rezultat se vraća string prezentacija trenutnog vremena i datuma

Primeri korišćenja navedenih konstruktora:

```
today = new Date()  
birthday = new Date("December 17, 1995 03:24:00")  
birthday = new Date(95,12,17)  
birthday = new Date(95,12,17,3,24,0)
```

U okviru ovog predefinisanog objekta postoje ugrađene metode koje se mogu koristiti. U daljem delu teksta će se razmatrati najvažnije metode, njihova primena i primeri.

Date.parse(datum)

Ovaj metod vraća broj milisekundi do navedenog datuma po lokalnom vremenu (od 1.1.1970 00:00:00). Primer:
datum.setTime(Date.parse("Aug 9, 2005"))

Date.UTC(gg,mm,dd [,hh][,mh][,sec])

Ovaj metod vraća broj milisekundi od 1.1.1970 00:00:00 do datuma, prema Universal Coordinate Time (GMT). Primer:
gmtDatum = new Date(Date.UTC(96, 11, 1, 0, 0, 0))

datum.getDate()

Ovaj metod vraća dan u mesecu (1-31) za navedeni datum. Primer:
datum = new Date("December 25, 2001 23:15:00");
dan = datum.getDate()
Nakon izvršavanja primera promenljiva dan dobija vrednost 25.

datum.getDay()

Ovaj metod vraća dan u nedelji (0-ned, 1-pon ... 6-sub) za navedeni datum. Primer:
datum = new Date("December 25, 2001 23:15:00");
dan = datum.getDay()
Nakon izvršavanja primera promenljiva dan dobija vrednost 2, jer je 25.12.2005.god., bio utorak.

datum.getHours()

Ovaj metod vraća sat za navedeni datum, moguće vrednosti su brojevi u opsegu od 0 do 23. Primer:
datum = new Date("December 25, 2001 23:15:00");
sati = datum.getHours()

Nakon izvršavanja primera promenljiva sati dobija vrednost 23.

datum.getMinutes()

Ovaj metod vraća minute za navedeni datum, moguće vrednosti su brojevi u opsegu od 0 do 59. Primer:

```
datum = new Date("December 25, 2001 23:15:00");  
minuti = datum.getMinutes()
```

Nakon izvršavanja primera promenljiva minuti dobija vrednost 15.

datum.getMonth()

Ovaj metod vraća mesec za navedeni datum (0-januar, 2-februar, ... 11-decembar). Primer:

```
datum = new Date("December 25, 2001 23:15:00");  
mesec = datum.getMonth()
```

Nakon izvršavanja primera promenljiva mesec dobija vrednost 11.

datum.getSeconds()

Ovaj metod vraća sekunde za navedeni datum, moguće vrednosti su brojevi u opsegu od 0 do 59. Primer:

```
datum = new Date("December 25, 2001 23:15:08");  
sekunde = datum.getSeconds()
```

Nakon izvršavanja primera promenljiva sekunde dobija vrednost 8.

datum.getTime()

Ovaj metod vraća vreme do navedenog datuma u milisekundama (od 1.1.1970 00:00:00). Primer:

```
datum = new Date("December 25, 2001 23:15:00");  
proteklo = datum.getTime()
```

Nakon izvršavanja primera promenljiva proteklo dobija vrednost 1009318500000, koja odgovara broju milisekundi od 1.1.1970 00:00:00 do 25.12.2001. 23:15:00.

datum.getTimezoneOffset()

Ovaj metod vraća razliku lokalnog vremena i GMT u minutama Primer:

```
datum = new Date();  
razlikaSati = datum.getTimezoneOffset()/60
```

Nakon izvršavanja primera promenljiva razlikaSati dobija vrednost -1.

datum.getYear()

Ovaj metod vraća godinu iz navedenog datuma (dvocifreno, od 1900 do 1999, u ostalim slučajevima 4 cifre). Primer:

```
datum = new Date();
```

```
godina = datum.getYear()
```

Nakon izvršavanja primera promenljiva godina dobija vrednost 2006.

datum.setDate(brojDana)

Ovaj metod postavlja dan u mesecu za navedeni datum. Argument metoda je broj u opsegu od 1 do 31. Primer:

```
datum = new Date("July 27, 1960 23:30:00");
```

```
datum.setDate(24)
```

Nakon izvršavanja primera promenljiva datum dobija vrednost 24.7.1960 23:30:00.

datum.setHours(brojSata)

Ovaj metod postavlja broj sati za navedeni datum. Argument metoda je broj u opsegu od 0 do 23. Primer

```
datum = new Date("July 27, 1960 23:30:00");
```

```
datum.setHours(7)
```

Nakon izvršavanja primera promenljiva datum dobija vrednost 27.7.1960 07:30:00.

datum.setMinutes(brojMinuta)

Ovaj metod postavlja broj minuta za navedeni datum. Argument metoda je broj u opsegu od 0 do 59. Primer:

```
datum = new Date("July 27, 1960 23:30:00");
```

```
datum.setMinutes(35)
```

Nakon izvršavanja primera promenljiva datum dobija vrednost 27.7.1960 23:35:00.

datum.setMonth(brojMeseca)

Ovaj metod postavlja broj meseci za navedeni datum. Argument metoda je broj u opsegu od 0 do 11. Primer:

```
datum = new Date("July 27, 1960 23:30:00");
```

```
datum.setMonth(11)
```

Nakon izvršavanja primera promenljiva datum dobija vrednost 27.12.1960 23:30:00.

datum.setSeconds(brojSekundi)

Ovaj metod postavlja dan u mesecu za navedeni datum. Argument metoda je broj u opsegu od 0 do 59. Primer:

```
datum = new Date("July 27, 1960 23:30:00");
```

```
datum.setSeconds(35)
```

Nakon izvršavanja primera promenljiva datum dobija vrednost 27.7.1960 23:30:35.

datum.setTime(vreme)

Ovaj metod definiše novi datum. Argument metoda je broj milisekundi od 1.1.1970 00:00:00 do željenog datuma.

```
datum.setTime(1009318500000)
```

Nakon izvršavanja primera promenljiva datum dobija vrednost 25.12.2001. 23:15:00.

datum.setYear(brojGodine)

Ovaj metod postavlja godinu za navedeni datum. Argument metoda je broj u opsegu od 0 do 99 za godine koje počinju sa 19, za ostale je 4 cifre.

```
datum = new Date("July 27, 1960 23:30:00");
```

```
datum.setYear(2010)
```

Nakon izvršavanja primera promenljiva datum dobija vrednost 27.7.2010 23:30:00.

datum.toGMTString()

Ovaj metod vrši konverziju datuma u GMT string iz lokalne vremenske zone. Primer:

```
datum = new Date("December 25, 2001 23:15:00");
```

```
datum.toGMTString()
```

Nakon izvršavanja primera promenljiva datum dobija vrednost "Tue, 25 Dec 2001 22:15:00 UTC"

datum.toLocaleString()

Ovaj metod vrši konverziju datuma u lokalni datum string iz GMT. Primer:

```
datum.toLocaleString()
```

Primeri za rad u laboratoriji

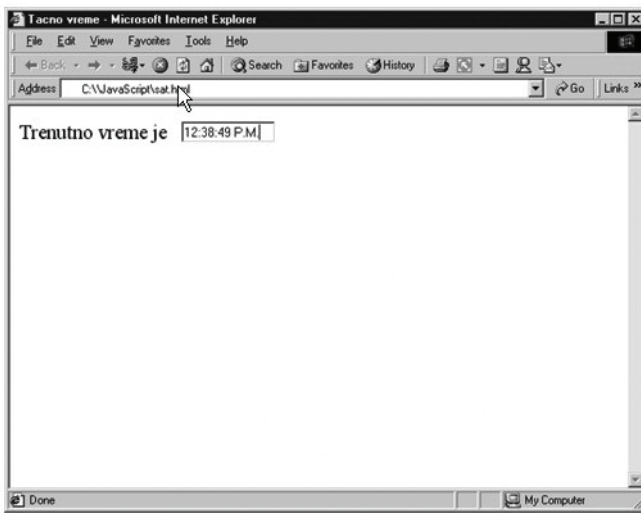
Primer 1

```
<HTML>
<HEAD>
<TITLE> Tacno vreme </TITLE>
<SCRIPT language="JavaScript">
    function JavaScriptVreme(){
        time = new Date()
        cas=time.getHours()
        minuti=time.getMinutes()
        sekunde=time.getSeconds()
        temp = "" + ((cas>12)?cas-12:cas)
        temp += ((minuti<10)?"0":"")+minuti
        temp += ((sekunde<10)?"0":"")+sekunde
        temp += ((cas>=12)?" P.M.":" A.M.")
        document.vremeForma.cifre.value=temp
        setTimeout("JavaScriptVreme()",1000)
    }

</SCRIPT>
</HEAD>

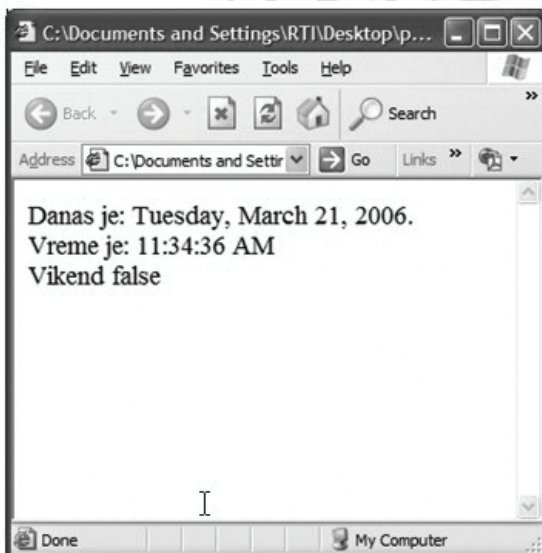
<BODY BGCOLOR="#FFFFFF" onLoad="JavaScriptVreme()">
<FORM name="vremeForma">
    Trenutno vreme je &nbsp; <INPUT TYPE="text" NAME="cifre"
SIZE=12>
</FORM>
</BODY>
</HTML>
```

Nakon izvršavanja prethodnog programa, ako je sve pravilno urađeno, dobija se prozor koji predstavlja izgled HTML stranice u Internet Exploreru (slika 2).



Primer 2

```
<script language="JavaScript">
d = new Date( );
document.writeln("Danas je: " + d.toLocaleDateString( ) + ' ');
document.writeln('Vreme je: ' + d.toLocaleTimeString( ));
var dayOfWeek = d.getDay( );
var weekend = (dayOfWeek == 0) || (dayOfWeek == 6);
document.writeln("Vikend " + weekend)
</script>
```

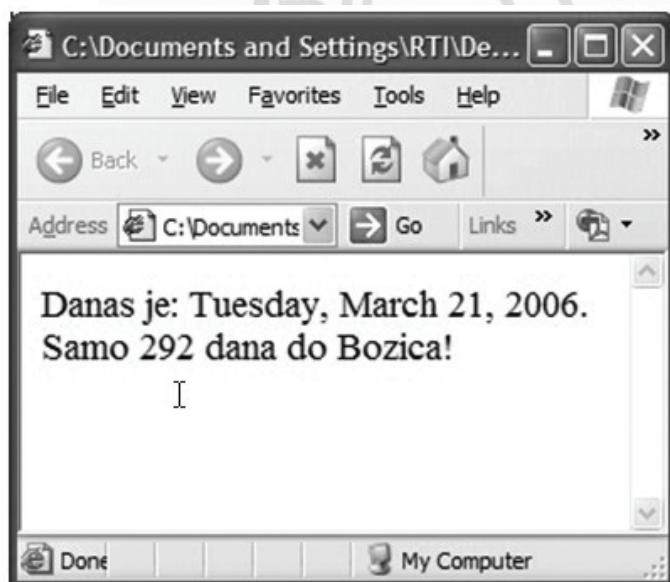


Nakon izvršavanja ovog primera dobija se ovakav izgled stranice.

Primer 3

```
<script language="JavaScript">
today = new Date( );
christmas = new Date( );
christmas.setMonth(0);
christmas.setDate(7);
christmas.setYear(2007);
if (today.getTime( ) < christmas.getTime( )) {
    difference = christmas.getTime( ) - today.getTime( );
    difference = Math.floor(difference / (1000 * 60 * 60 * 24));
    document.write("samo " + difference + ' dana do Bozica!<p>');
}
</script>
```

Nakon izvršavanja ovog primera dobija se sledeća stranica



String objekat

Ovaj objekat se koristi da bi se efikasnije obradio niz karaktera, što objekat tipa String u suštini i jeste. U okviru JavaScript jezika String se definiše kao niz karaktera između apostrofa ili između dvostrukih navodnika: „neki String“ ili 'neki String'. I u okviru ovog objekta postoje dostupni metodi koji se mogu koristiti. U daljem tekstu sledi njihov opis, sintaksa i primeri upotrebe.

escape("string")

Ova funkcija kao rezultat vraća ASCII kôdove karaktera u okviru argumenta. Primer:

```
y = escape("!#")
```

Nakon izvršavanja primera promenljiva y dobija vrednost „%21%23“, jer su ASCII kôdovi za simbole ! i # 21 i 23.

eval(izraz)

Ova funkcija izračunava vrednost izraza koji je definisan kao argument funkcije. Primer

```
var x = eval("4+5-8")
```

Nakon izvršavanja primera promenljiva x dobija vrednost 1.

linkTekst.link(linkURL)

Ovaj metod kreira tekst linkTekst koji predstavlja HTML link na neku drugu stranicu, čiji je adresa definisana sa argumentom linkURL (desjtvokao i HTML taga <A HREF...>), Primer

```
var naziv = „link ka prezentaciji Univerziteta Singidunum“;
```

```
var URL = "http://www.singidunum.ac.yu";
```

```
document.write("Ovo je " + naziv.link(URL))
```

Nakon izvršavanja primera na stranici će se pojaviti tekst [Ovo je link ka prezentaciji Univerziteta Singidunum](http://www.singidunum.ac.yu), koji će predstavljati vezu ka stranici www.singidunum.ac.yu.

parseInt(stringBroj [,osnova])

Ova funkcija kao rezultat vraća ceo broj dobijen konverzijom argumenta stringBroj koji je tipa String u brojno sistem sa osnovom koju definiše argument osnova. Ovaj argument je opcioni i ako se ne navede podrazumeva se osnova 10, t.j. dekadni brojni sistem. Primer:

```
x = parseInt("17", 8);
```

```
y = parseInt("15", 10)
```

Nakon izvršavanja primera i promenljiva x i promenljiva y dobija vrednost 15.

string.big()

Ovaj metod prikazuje string sa uvećanim slovima (isto dejstvo kao HTML tag <BIG>). Primer:

```
"Dobar dan!".big()
```

string.blink()

Ovaj metod prikazuje string sa blinkovanjem (isto dejstvo kao HTML tag <BLINK>). Primer:

```
"Dobar dan!".blink()
```

string.bold()

Ovaj metod prikazuje string boldovano (isto dejstvo kao HTML tag). Primer:

```
"Dobar dan!".bold()
```

string.charAt(broj)

Ovaj metod kao rezultat vraća znak na navedenoj poziciji. Pozicije unutar stringa se računaju sa leve na desnu stranu i prva pozicija ima indeks 0. U okviru svakog objekta tipa String postoji i osobina (property) length koja je jednaka broju karaktera u posmatranom stringu. Korišćenjem ovog podatka može se odrediti i indeks poslednjeg karaktera u stringu, a to je vrednost `string.length - 1`. Primer:

```
x= "Dobar dan!".charAt(4)
```

```
y= "Dobar dan!".charAt(6)
```

Nakon izvršavanja primera promenljiva x dobija vrednost 'r', a promenljiva y je 'd'.

string.fontcolor("boja")

Ovaj metod prikazuje string u zadanoj boji (isto dejstvo kao HTML tag). Primer:

```
"Dobar dan!".fontcolor("blue")
```

string.fontSize(broj)

Ovaj metod prikazuje string u zadatoj veličini (isto dejstvo kao HTML tag)

```
"Dobar dan!".fontSize(7)
```

string.indexOf(traziString, [odPozicije])

Ovaj metod vraća broj pozicije na kojoj je prvi put pronađen argument tipa String traziString. U slučaju da se traženi string ne nalazi u početnom stringu kao rezultat se vraća vrednost -1. Ako postoji i drugi argument odPozicije, tada će se pretraga izvršavati od zadate pozicije. Primer:

```
x="Dobar dan!".indexOf("r")
```

```
y="Dobar dan!".indexOf("a",4)
```

Nakon izvršavanja primera promenljiva x dobija vrednost 4, a promenljiva y je 7.

string.italics()

Ovaj metod prikazuje string sa italik stilom (isto dejstvo kao HTML tag <I>). Primer:

```
"Dobar dan!".italics()
```

string.lastIndexOf(traziString, [doPozicije])

Ovaj metod vraća broj pozicije na kojoj se poslednji put pojavljuje argument tipa String traziString. U slučaju da se traženi string ne nalazi u početnom stringu kao rezultat se vraća vrednost -1. Ako postoji i drugi argument doPozicije, tada će se pretraga izvršavati do zadate pozicije. Primer:

```
x="Dobar dan!".lastIndexOf("a")
```

```
y="Dobar dan!".lastIndexOf("a", 6)
```

Nakon izvršavanja primera promenljiva x dobija vrednost 7, jer je to poslednje pojavljivanje stringa "a", a promenljiva y je 3, jer je to poslednje pojavljivanje stringa "a" do poziciji 6.

string.strike()

Ovaj metod prikazuje precrtani string (isto dejstvo kao HTML tag <STRIKE>). Primer:

```
"Dobar dan!".strike()
```

string.sub()

Ovaj metod prikazuje string kao subscript (isto dejstvo kao HTML tag <SUB>). Primer:

"Hej!".sub()

string.substring(prvi, poslednji)

Ovaj metod vraća deo stringa počev od pozicije prvi do pozicije poslednji, t.j. uzima redom karaktere na pozicijama prvi, prvi + 1, prvi + 2, ..., poslednji - 2, poslednji - 1.

x = "Dobar dan!".substring(6,9)

Nakon izvršavanja primera promenljiva x dobija vrednost "dan", jer su to karakteri na pozicijama 6, 7 i 8.

string.sup()

Ovaj metod prikazuje string kao superscript (isto dejstvo kao HTML tag <SUP>). Primer:

"Hej!".sub()

string.toLowerCase()

Ovaj metod izvrši konverzija svih karaktera u okviru stringa u mala slova. Primer:

x = "Dobar dan!".toLowerCase()

Nakon izvršavanja primera promenljiva x dobija vrednost "dobar dan", jer je izvršena konverzija svih karaktera u mala slova.

string.toUpperCase()

Ovaj metod izvrši konverzija svih karaktera u okviru stringa u velika slova. Primer:

x = "Dobar dan!".toUpperCase()

Nakon izvršavanja primera promenljiva x dobija vrednost "DOBAR DAN", jer je izvršena konverzija svih karaktera u velika slova.

unescape("kodovi")

Ova funkcija kao rezultat vraća ASCII znakove navedenih kodova u okviru argumenta funkcije. Primer:

x = unescape("%21%23")

Nakon izvršavanja primera promenljiva x dobija vrednost "!#", jer su simboli ! i # kodovani sa ASCII kodovima 21 i 23.

16. RAD SA UZORCIMA (Pattern Matching)

Definisanje uzorka

Česta upotreba JavaScript funkcija je provera unetih podataka od strane klijenta. Razlog tome je što JavaScript ima razvijenu podršku za razne vrste provera i one se obavljaju na klijentskoj strani, tako da korisnik veoma brzo dobija obaveštenje ako neki podatak nije unet u propisanom formatu. Primer takvog korišćenja ovog jezika može da bude polje za poštanski broj, gde korisnik treba da unese petocifreni broj, koji ne sme počinjati nulom. Ako nisu ispunjeni ovi uslovi treba korisnika obavestiti da je došlo do greške.

Ovaj i slični zadaci se rešavaju upotrebom uzoraka (pattern) i pozivanjem određenih metoda koji uneti tekst upoređuju sa definisanim uzorkom. Uzorak se još naziva i regularni izraz (regular expression) i može se definisati na dva načina

```
var uPrimer = new RegExp(„HTML“)
```

ili

```
var uPrimer = /HTML/
```

Na oba načina se formira objekat uzorka koji se naziva uPrimer i kome odgovara svaki string koji u sebi sadrži podstring HTML. Kao primer složenijeg uzorka mogu se navesti sledeće deklaracije

```
var uPrimer = new RegExp(„s$“)
```

ili

```
var uPrimer = /s$/
```

Sada je promenljiva uPrimer uzorak koji odgovara bilo kom stringu koji se završava sa s. Ovakva vrednost je dobijena, jer u okviru uzorka simbol s predstavlja samog sebe, a simbol \$ predstavlja metakarakter koji označava kraj stringa.

U sledećoj tabeli su dati mogući karakteri koji se mogu koristiti u okviru uzorka.

Karakter	Predstavlja
Slovo ili broj	Istu vrednost
\0	Specijalni NUL karakter
\t	Tab znak
\n	Nova linija
\v	Vertikalni tab znak
\f	Form feed
\r	Carriage return
\uxxxx	Unicode karakter definisan pomoću heksadecimalnog boja xxxx; na primer, \u0009 ima isti efekat kao i \t

U okviru uzorka se mogu koristiti specijalni simboli sa posebnim značenjem. U okviru sledeće tabele su definisani ovi simboli.

Karakter	Predstavlja pojavljivanje
[...]	Bilo kog karaktera od onih koji su navedeni između [i].
[^...]	Bilo kog karaktera koji nije naveden između [i].
.	Bilo kog karaktera osim nove linije
\w	Bilo kog ASCII definisanog slova.
\W	Bilo kog karaktera koji nije ASCII definisano slovo.
\d	Bilo koje ASCII definisane cifre
\D	Bilo kog karaktera koji nije ASCII definisana cifra
[b]	Blanko znak

Na osnovu ove tabele može se zaključiti da uzorak

/[abc]/

predstavlja jedno pojavljivanje simbola a ili jedno pojavljivanje simbola b ili jedno pojavljivanje simbola c. Tako da string "c" ispunjava uslove definisane uzorkom, a string "s" ne ispunjava definisane uslove. Na sličan način uzorak

/[^abc]/

predstavlja karakter koji nije simbol a ili simbol b ili simbol c.

Primer za poštanski broj koji je naveden na početku teksta bi izgledao kao sledeći uzorak

`^d\d\d\d\d/`

Pomoću ovog uzorka se definiše broj koji se sastoji od 5 cifara. Uzorak bi bio još nepregledniji da je potrebno definisati broj koji se sastoji od 18 ili 28 cifara. Za rad sa ovakvom vrstom uzoraka postoje i specijalne oznake koje su date u sledećoj tabeli.

Oznaka	Značenje
$\{n,m\}$	Ponavljjanje prethodne grupe najmanje n puta, ali najviše m puta.
$\{n,\}$	Ponavljjanje prethodne grupe n ili više puta.
$\{n\}$	Ponavljjanje prethodne grupe tačno n puta.
$?$	Ponavljjanje prethodne grupe jednom ili nijednom. Isto dejstvo kao i $\{0,1\}$.
$+$	Ponavljjanje prethodne grupe jednom ili više puta. Isto dejstvo kao i $\{1,\}$.
$*$	Ponavljjanje prethodne grupe nijednom ili više puta. Isto dejstvo kao i $\{0,\}$.
$ $	Alternative. Pojavljivanje dela izraza sa desne ili pojavljivanje izraza sa leve strane.
$(...)$	Grupisanje simbola u jedan objekat nad kojim se mogu koristiti oznake $*$, $+$, $?$, $ $, itd.
$^$	Pretraga uzorka se obavlja na početku stringa
$\$$	Pretraga uzorka se obavlja na kraju stringa

Uzorak `^d\d\d\d\d/` se sada može definisati kao `^d{5}/`. Neki od primera korišćenja ovih oznaka su:

`^d{2,4}/` // uzorak koji označava 2, 3 ili 4 pojavljivanje cifara

`^w{3}\d?/` // uzorak koji označava tačno tri pojavljivanja slova i opciono jedne cifre, Na primer string koji odgovara ovom uzorku je „abc8” ili „qqq”.

`^s+java\s+/` // uzorak koji označava string "java" sa jednim ili više prostora pre ili posle stringa

`/[^\s]*/` // uzorak koji označava nula ili više pojavljivanje karaktera navoda

`/ab|cd|ef/` // uzorak koji označava pojavljivanje ab ili pojavljivanje cd ili pojavljivanje ef

`\d{3}[a-z]{4}/` // uzorak koji označava pojavljivanje tri cifre ili 4 mala slova

`/java(script)?/` // uzorak koji označava pojavljivanje stringa „java” ili stringa „javascript”

`/(ab|cd)+|ef)/` // uzorak koji označava pojavljivanje stringa „ef” ili pojavljivanje jednom ili više puta stringa „ab” ili pojavljivanje jednom ili više puta stringa „cd”

Ispitivanje da li određeni string odgovara stringu može se dodatno definisati i sa određenim atributima. U sledećoj tabeli su dati mogući atributi.

Atribut	Značenje
i	Izvršavanje case-insensitive ispitivanja.
g	Izvršava globalno ispitivanje, znači pronaći će se sva pojavljivanja definisanog uzorka, a neće se ispitivanje zaustaviti posle prvog pronalaska uzorka.
M	Rad sa više linija. ^ označava početak linije ili stringa, a \$ predstavlja kraj linije ili stringa.

U dosadašnjem delu poglavlja razmatrani su načini definisanja uzorka. Primena uzorka je u okviru ispitivanja da li određeni string odgovara uzorku ili ne. Takva ispitivanja se mogu izvršavati na dva načina: pomoću metoda koje poziva string koji se ispituje ili pomoću metoda koje poziva definisani uzorak.

Ispitivanje uzoraka pomoću metoda objekta tipa String

Ako se koriste metode stringa za ispitivanje, tada se uzorak mora pojavljivati kao argument metoda. Ovi metodi se koriste za ispitivanje da li string odgovara uzorku, kao i mogućnost zamene određenog dela stringa sa drugim.

Najjednostavniji metod je `search()`. Ovaj metod ispituje da li u okviru stringa postoji definisani uzorak, i kao rezultat vraća poziciju njegovog prvog pojavljivanja, ili -1 ako ne pronađe uzorak. Primer:

```
x = /script/
```

```
y = "JavaScript".search(x,i);
```

Kao rezultat izvršavanja ovog primera promenljiva `y` će dobiti vrednost 4. Ovaj metod ne podržava globalnu pretragu, znači on ignoriše upotrebu atributa `g` u okviru definicije uzorka.

Sledeći metod koji se koristi je metod `replace()`. Ovaj metod obavlja ispitivanje dali u stringu postoji uzorak i ako postoji zamenu uzorka unutar stringa sa nekom drugom vrednošću. `Replace()` ima dva argumenta, prvi je uzorka, a drugi je string koji treba da zameni uzorak. Ako se u okviru uzorka navede i `g` atribut, ovaj metod će izvršiti zamenu svakog uzorka koji pronade u okviru stringa. Ako se ne navede `g` atribut izvršiće se zamena samo prvog pojavljivanja uzorka. U sledećem primeru bez obzira kako je napisan string „javascript” što se tiče velikih i malih slova, zameniće se u string "JavaScript":

```
text = "JAVascriPT"  
text.replace(/javascript/gi, "JavaScript");
```

Nakon izvršavanja ovog primera promenljiva `text` će imati vrednost "JavaScript". Još jedan primer:

```
str = "Lana ima 5 pomorandzi i 135 limuna"  
promena3u5 = new RegExp("[3-5]", "g")  
str.replace(promena3u5, "9")
```

Nakon izvršavanja ovog primera promenljiva `str` će imati vrednost " Lana ima 9 pomorandzi i 199 limuna ", jer je uzorkom definisano da se svako pojavljivanje cifri 3, 4 i 5 zameni sa cifrom 9.

Sledeći metod `match( )` je najopštiji metod od mogućih metoda objekta tipa string. Ovaj metod ima samo jedan argument, i to argument tipa uzorka. Rezultat njegovog izvršavanja je niz koji sadrži rezultate ispitivanja. Ako je u okviru uzorka definisan atribut `g`, rezultat je niz sa svim pojavljivanjem definisanog uzorka. Na primer:

```
"1 plus 2 equals 3".match(/d+/g)
```

Rezultat izvršavanja metoda `match()` u primeru je niz ["1", "2", "3"], jer je uzorak definisan kao pojavljivanje 1 ili više puta cifre, i to u celom stringu. U okviru navedenog stringa prvo se pronalazi cifra 1, pa cifra 2, pa cifra 3 i kao rezultat se dobija niz koji kao članove sadrži ove cifre.

U slučaju da nije naveden g atribut, pomoću `match( )` metoda se ne izvršava globalno ispitivanje, već se ispituje prvo pojavljivanje definisanog uzorka. Ali, i u ovom slučaju rezultat je niz iako se ne sprovodi globalno ispitivanje, i to niz čiji je prvi element pronađeni deo stringa, a ostali članovi niza su delovi uzorka. Znači, ako je rezultat niz `a`, `a[0]` je kompletno pronađeni uzorka, `a[1]` je do string akoji odgovara prvom delu uzorka, , itd. Sldeći primer definiše uzorak koji odgovara definiciji URL putanje:

```
var url = /(w+):\\(\\w.+)\\(S*)/;  
var text = "Posetite strnicu http://www.isp.com/~david";  
var result = text.match(url);  
if (result != null) {  
    var fullurl = result[0];  
    var protocol = result[1];  
    var host = result[2];  
    var path = result[3];  
}
```

Nakon izvršavanja navedenog primera promenljiva `fullurl` će imati vrednost "<http://www.isp.com/~david>", jer je ona jednaka prvom članu niza koji predstavlja rezultat izvršavanja metoda `match()`. Promenljiva `protocol` će imati vrednost "`http`", promenljiva `host` vrednost "`www.isp.com`", a promenljiva `path` vrednost "`~david`".

Poslednji metod koji postoji kod objekta `String` a koristi uzorke je metod `split( )`. Ovaj metod ima jedan argument i to tipa uzorka. Rezultat izvršavanja ovog metoda je niz koji se dobija kada se string podeli argumentum uzorkom kao separatorom. Na primer:

```
"123,456,789".split(",");
```

Rezultat izvršavanja ovog primera je niz `["123","456","789"]`, jer je string podeljen pomoću separatora `","`.

U ovom primeru uzorak je bio jednostavan i predstavljao je samo jedan karakter. Treba naglasiti da argument metoda `split( )` može biti bilokoji regularno defnisani uzorka. Na primer:

```
"1,2, 3 , 4 ,5".split(/\s*,\s*/);
```

Rezultat ovog primera je niz ["1","2","3","4","5"], jer je uzorak definisan sa određenim brojem blanko znakova pre i posle zareza, uključujući i zarez.

Metodi objekta RegExp

U prethodnom delu poglavlja korišćen je konstruktor RegExp() da bi se napravila promenljiva tipa regexp. Pored ovog konstruktora ovaj objekat ima tri metoda koji se mogu koristiti za ispitivanje odnosa stringova i uzoraka.

Definicija poštanskog koda koa petocifrenog boja pomoću regexp konstruktora je

```
var zipcode = new RegExp("\\d{5}", "g");
```

Najvažniji metod ovog objekta koji se koristi u ispitivanju uzorka je exec(). Ovaj metod je veoma sličan String metodu match() koji je opisan ru prethodnom delu teksta. Razlike je u tome što je kod ovog metoda argument string, a primenjuje se na uzorka, dok je kod metoda match() argument bio uzorak, a primenjivao se na String. Znači rezultat izvršavanje metoda exec() je niz koji sadrži rezultate ispitivanja, definisane na isti način kao i metod match().

Za razliku od metoda match() metod exec() vraća isti rezultat ako postoji atribut g i ako ne postoji, i to uvek prvo poklapanje i sve relevantne informacije o njemu. Primer:

```
var pattern = /Java/g;
var text = "JavaScript jemnogo zabavniji nego Java!";
var result;
while((result = pattern.exec(text)) != null) {
    alert("Pronadjen `" + result[0] + "`" +
        " na poziciji " + result.index + "; sledeca pretraga pocinje od " +
        pattern.lastIndex);
}
```

Rezultat izvršavanja navedenog primera su dva alerta. Prvi sa tekstom "Pronadjen `Java` na poziciji 0; sledeca pretraga počinje od 4", a drugi je sa tekstom "Pronadjen `Java` na poziciji 34; sledeca pretraga počinje od 38".

Sledeći metod iz ove grupe je metod test(). Ovaj metod je mnogo jednostavniji nego prethodni. Njegov argument je string, a rezultat tru ako string odgovra uzorku. Na primer:

```
var pattern = /java/i;  
pattern.test("JavaScript");
```

Rezultat se dobija true, jer je uzorak pronađen u okviru stringa.



17. JAVASCRIPT I FORME

Programski jezik JavaScript je svoju popularnost stekao mogućnošću da pristupa elementima forme, čita njihove vrednosti, obrađuje ih i definiše nove vrednosti elemenata. Takođe iskorišćena je i osobina HTML jezika da prepozna korisnikovu akciju i reaguje na nju. U ovom poglavlju će se razmatrati ove osobine JavaScripta, njihova primena i primeri upotrebe.

Čitač može da prepozna svaku akciju korisnika, bilo da ona potiče od miša ili tastature. Svaki HTML objekat, kao što je dugme, tekst polje, polje za potvrdu, slika, itd., ima listu događaja koji su povezani sa datim objektom. Na primer tekst polje prepoznaje kada korisnik promeni tekst koji je trenutno u okviru polja, slika prepoznaje kada se kursor miša nalazi na njenoj površini, dugme prepoznaje kada korisnik mišem klikne na njega, itd. U sledećoj tabeli su dati najvažniji događaji koji se mogu desiti, na koje objekte mogu da se primene i kako se pozivaju u okviru taga koji definiše određeni objekat.

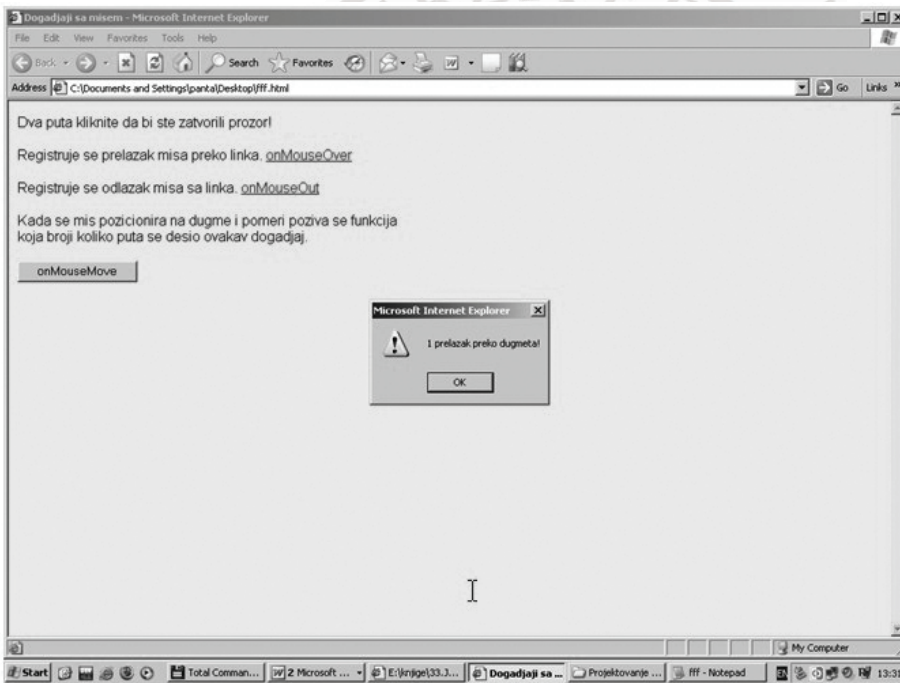
Događaj	Nastaje kada korisnik...	Kod
blur	izađe iz fokusa elementa forme	onBlur
click	klikne na element forme ili link	onClick
change	promeni vrednost izabranog elementa forme	onChange
focus	uđe u fokus nekog elementa forme	onFocus
load	učita stranicu u browser	onLoad
mouseover	pređe pokazivačem miša preko linka i sl.	onMouseOver
mouseout	izađe kursorom miša sa određene površine ili linka	onMouseOut
select	izabere polje elementa forme	onSelect
submit	izvrši slanje forme	onSubmit
unload	napusti stranicu	onUnload
reset	resetuje sadržaj forme	onReset
error	dobije grešku prilikom učitavanja slike ili stranice	onError
abort	prekine učitavanja slike ili stranice	onAbort


```
<a href="#" onMouseOut="alert('Event: onMouseOut');">onMouseOut
</a><p>
```

Kada se mis pozicionira na dugme i pomeri poziva se funkcija
koja broji koliko puta se desio ovakav događaj.

```
<form>
  <input type="button"
    value="onMouseMove"
    onMouseMove="track_Moves();">
</form>
</body>
</html>
```

Izgled stranice nakon prvog prelaska miša preko dugmeta:



Navedeno je da JavaScript može i da pročita vrednost proizvoljnog elementa forma. Vrednosti elementa forme se prilazi u opštem slučaju na sledeći način:

`document.imeForme.imeElementa.value`

gde je `document` službena reč, `imeForme` ime forme u okviru koje se nalazi element čijoj se vrednosti pristupa, `imeElementa` ime elementa i `value` službena reč.

Znači ako je u okviru HTML stranice definisana sledeća forma sa jednim tekst poljem:

```
<FORM NAME="PrimerForme">
<INPUT TYPE="text" NAME="PrimerTekstPolja">
</FORM>
```

tada se definisanom tekst polju u okviru JavaScript koda prilazi na sledeći način

`document.PrimerForme.PrimerTekstPolja.value`

Treba naglasiti još jednom da je JavaScript case sensitive jezik, pa kako je definisano ime u okviru elementa forme, tako se mora koristiti i u okviru JavaScript koda. U prethodnom primeru greska bi bila ako bi se ime tekst polja navelo kao `PRIMERTEKSTPOLJA` ili `Primertekstpolja`, jer JavaScript ne bi prepoznao da se radi o definisanom elementu.

Ako se želi pročitati vrednost navedenog tekst polja u promenljivu `x`, to se rešava sa:

`x = document.PrimerForme.PrimerTekstPolja.value`

i analogno, ako se želi u tekst polje upisati vrednost promenljive `x`:

`document.PrimerForme.PrimerTekstPolja.value = x`

U sledećem primeru se definiše forma koja sadrži:

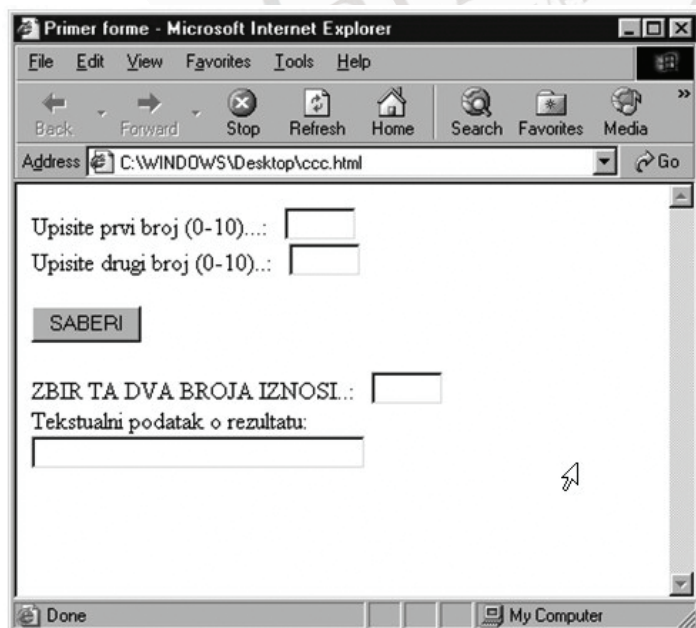
- `broj1` i `broj2` - tekstualna polja predviđena za upis dva broja od strane korisnika
- dugme `SABERI` sa definisanom obradom događaja `onClick="Saberi()"` – kada korisnik klikne na dugme poziva se funkcija `Saberi()`

- zbir je tekstualno polje u koje će se upisati povratna informacija o rezultatu sabiranja
- tekst je tekstualno polje u koje će se upisati određeno tekstualno obrazloženje dobijenog rezultata

HTML kod ove forme je:

```
<FORM METHOD="post" NAME="mojaforma">
  Upišite prvi broj (0-10)...: &nbsp;
  <INPUT TYPE="text" NAME="broj1" SIZE=5>   <br>
  Upišite drugi broj (0-10)...: &nbsp;
  <INPUT TYPE="text" NAME="broj2" SIZE=5>   <br><br>
  <INPUT TYPE="button" VALUE="SABERI" NAME="dugme"
  onClick="Saberi()">   <br><br>
  ZBIR TA DVA BROJA IZNOSI.: &nbsp;
  <INPUT TYPE="text" NAME="zbir" SIZE=5>   <br>
  Tekstualni podatak o rezultatu: &nbsp;
  <INPUT TYPE="text" NAME="tekst" SIZE=30>
</FORM>
```

Izgled forme je



U okviru JavaScript koda ove stranice definiše se funkcija Saberi() koja sabira vrednosti unete je sledeći:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
function Saberi() {
    var br1 = document.mojaforma.broj1.value - 0;
    //prihvatanje prvog broja sa forme u varijablu br1 i konverzija u broj

    var br2 = document.mojaforma.broj2.value - 0;
    //prihvatanje drugog broja sa forme u varijablu br2 i konverzija u broj

    var ukupno = br1 + br2; //sabiranje br1 + br2 i smeštanje rezultata u
    promenljivu ukupno
    var poruka = ""; // početna vrednost promenljive poruka
    if (ukupno <= 0) //definisanje tekstualne poruke
        poruka = "nula ili negativan!";
    else if (ukupno > 10)
        poruka = "veći od deset!";
    else
        poruka = "između 1 i 10!";

    document.mojaforma.zbir.value = ukupno;
    //upis rezultata u tekstualno polje zbir
    document.mojaforma.tekst.value = "Zbir je " + poruka;
    //upis teksta poruke u tekstualno polje "tekst"
}
</SCRIPT>
```

Ako se u tekst polja unesu vrednosti 8 i 18 i aktivira dugme SABERI dobija se sledeći izgled



Sledeći primer definiše HTML formu sa skoro svim elementima forme i JavaScript funkciju koja prepoznaje koji se događaj dogodio u okviru kog elementa i prikazuje to u okviru tekst prozora.

```
<form name="everything">
<table border="border" cellpadding="5">
  <tr>
    <td>Username:<br>[1]<input          type="text"          name="username"
size="15"></td>
    <td>Password:<br>[2]<input      type="password"      name="password"
size="15"></td>
    <td rowspan="4">Input Events[3]<br>
      <textarea name="textarea" rows="20" cols="28"></textarea></td>
    <td rowspan="4" align="center" valign="center">
      [9]<input type="button" value="Clear" name="clearbutton"><br>
      [10]<input type="submit" name="submitbutton" value="Submit"><br>
      [11]<input type="reset" name="resetbutton" value="Reset"></td></tr>
  <tr>
    <td colspan="2">
      Filename: [4]<input type="file" name="file" size="15"></td></tr>
  <tr>
    <td colspan="2">
      <td>My Computer Peripherals:<br>
```

```

[5]<input type="checkbox" name="peripherals" value="modem">56K
Modem<br>
[5]<input type="checkbox" name="peripherals"
value="printer">Printer<br>
[5]<input type="checkbox" name="peripherals" value="tape">Tape
Backup</td>
<td>My Web Browser:<br>
[6]<input type="radio" name="browser" value="nn">Netscape<br>
[6]<input type="radio" name="browser" value="ie">Internet
Explorer<br>
[6]<input type="radio" name="browser"
value="other">Other</td></tr>
<tr>
<td>My Hobbies:[7]<br>
<select multiple="multiple" name="hobbies" size="4">
<option value="programming">Hacking JavaScript
<option value="surfing">Surfing the Web
<option value="caffeine">Drinking Coffee
<option value="annoying">Annoying my Friends
</select></td>
<td align="center" valign="center">My Favorite Color:<br>[8]
<select name="color">
<option value="red">Red <option value="green">Green
<option value="blue">Blue <option value="white">White
<option value="violet">Violet <option value="peach">Peach
</select></td></tr>
</table>
</form>
<div align="center">
<table border="4" bgcolor="pink" cellspacing="1" cellpadding="4">
<tr>
<td align="center"><b>Form Elements</b></td>
<td>[1] Text</td> <td>[2] Password</td> <td>[3] Textarea</td>
<td>[4] FileUpload</td> <td>[5] Checkbox</td></tr>
<tr>
<td>[6] Radio</td> <td>[7] Select (list)</td>
<td>[8] Select (menu)</td> <td>[9] Button</td>
<td>[10] Submit</td> <td>[11] Reset</td></tr>

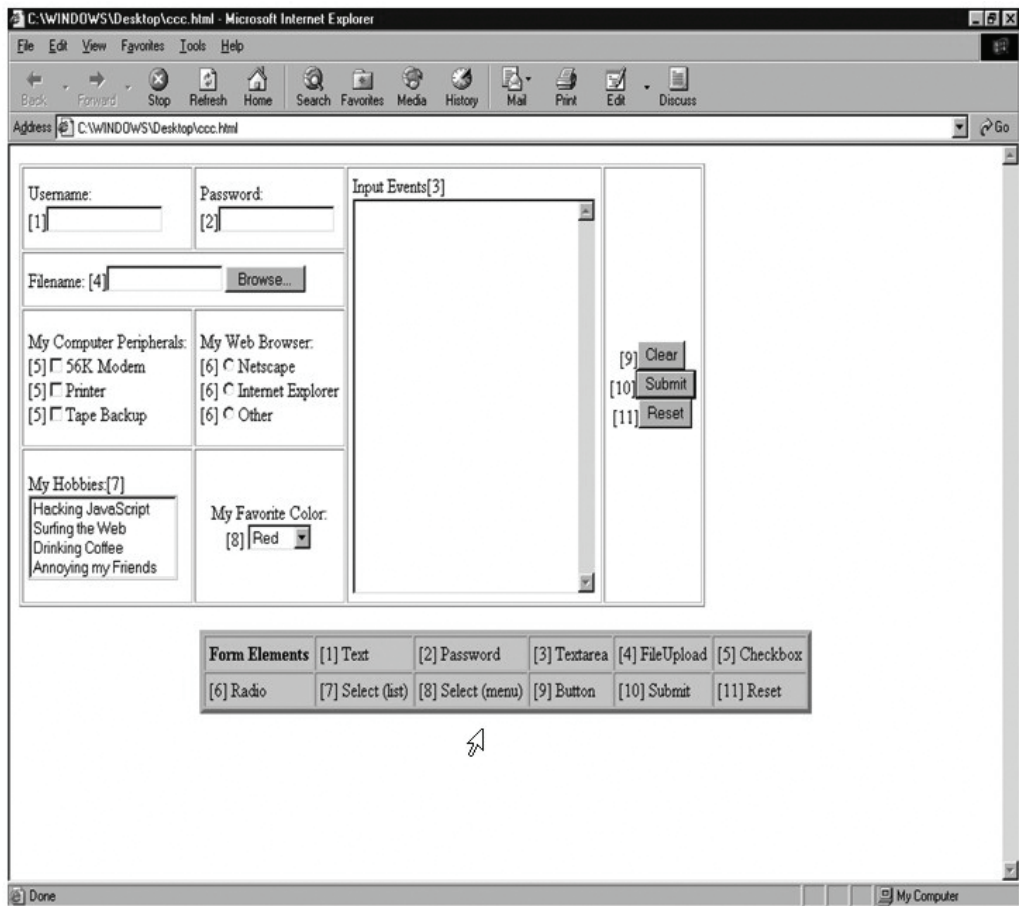
```

```

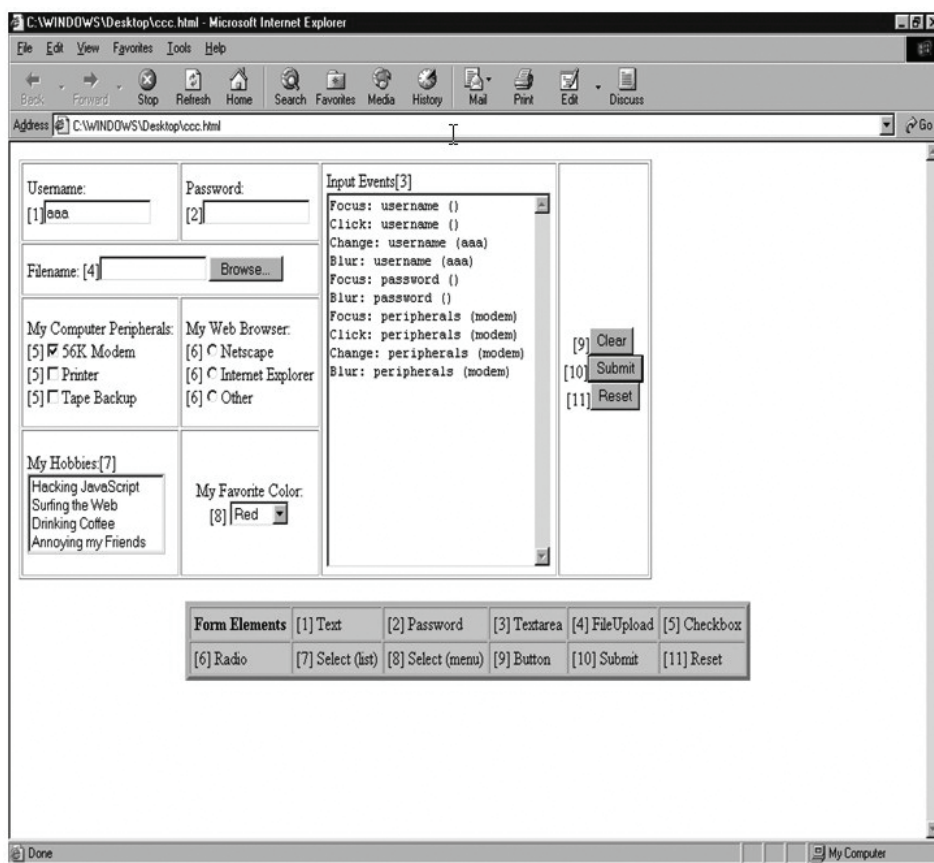
</table>
</div>
<script>
function report(element, event) {
    var elmtname = element.name;
    if ((element.type == "select-one") || (element.type == "select-multiple")){
        value = " ";
        for(var i = 0; i < element.options.length; i++)
            if (element.options[i].selected)
                value += element.options[i].value + " ";
    }
    else if (element.type == "textarea") value = "...";
    else value = element.value;
    var msg = event + ": " + elmtname + ' (' + value + ')\n';
    var t = element.form.textarea;
    t.value = t.value + msg;
}
function addhandlers(f) {
    for(var i = 0; i < f.elements.length; i++) {
        var e = f.elements[i];
        e.onclick = function( ) { report(this, 'Click'); }
        e.onchange = function( ) { report(this, 'Change'); }
        e.onfocus = function( ) { report(this, 'Focus'); }
        e.onblur = function( ) { report(this, 'Blur'); }
        e.onselect = function( ) { report(this, 'Select'); }
    }
    f.clearbutton.onclick = function( ) {
        this.form.textarea.value=""; report(this,'Click');
    }
    f.submitbutton.onclick = function ( ) {
        report(this, 'Click'); return false;
    }
    f.resetbutton.onclick = function( ) {
        this.form.reset( ); report(this, 'Click'); return false;
    }
}
addhandlers(document.everything);
</script>

```

Izgled stranice nakon učitavanja je



a nakon upisa teksta u tekst polje i potvrde polja za potvrdu



```
<SCRIPT LANGUAGE="JavaScript">
```

```
//FUNKCIJA KOJA PROVERAVA DA LI SU INFORMACIJE VALIDNE
```

```
function validacijaUnosa(){
```

```
//DEKLARISANJE VARIJABLI KOJE ČUVAJU UNETE VREDNOSTI U FORMU
```

```
var ime=info.ime.value;
```

```
var prezime=info.prezime.value;
```

```
var ulica=info.ulica.value;
```

```
var grad=info.grad.value;
```

```
var postBroj=info.postBroj.value;
```

```
var telefon=info.telefon.value;
```

```
var lozinka=info.lozinka.value;
```

```
var potvrdaLozinke=info.potvrdaLozinke.value;
```

```
//PROVERA DA LI SU SVA POLJA POPUNJENA
if (ime.toString()=="") {alert("Niste uneli ime!!!");}
if (prezime.toString()=="") {alert("Niste uneli prezime!!!");}
if (ulica.toString()=="") {alert("Niste uneli ulicu!!!");}
if (grad.toString()=="") {alert("Niste uneli grad!!!");}
if (telefon.toString()=="") {alert("Niste uneli broj telefona!!!");}
if (lozinka.toString()=="") {alert("Niste uneli lozinku!!!");}
if (potvrdaLozinke.toString()=="") {alert("Niste potvrdili lozinku!!!");}
var pb=proveriPostBroj(5);
    if (pb==true){
        validanPostBroj=true;
        alert("Postanski broj O.K.");
    }
    else{
        alert("postanski broj nije unet u ispravnom formatu");
    }
    if(!proveriTelefon(telefon)){
        alert("Telefon nije unet u odgovarajucem formatu");
    }
    else{
        alert("Telefon je O.K.");
    }

    if(lozinka!="" && potvrdaLozinke!=""){
        if (lozinka!=potvrdaLozinke){
            alert("Lozinka nije ispravno potvrđjena");
        }
        else{
            alert("Lozinka je O.K.");
        }
    }
    else{
        alert("Morate uneti lozinku i potvrdu lozinke!!!");
    }
}
```

```
function proveriTelefon(tel){
var validanTel=/^(\d{10}|\d{3}-\d{3}-\d{4})|(\d{3}\)\d{3}-\d{4})$/;
```

```
return validanTel.test(tel);  
}
```

```
function proverPostBroj(broj_cifara){  
var unetPostBrojStr=document.info.postBroj.value;  
var unetPostBroj=parseInt(unetPostBrojStr,10);  
  
if (unetPostBrojStr.length==broj_cifara){  
if(unetPostBroj!=0 && isNaN(unetPostBroj)==false){  
return true;  
}  
else {  
return false;  
}  
}  
else{  
return false;  
}  
}  
</SCRIPT>
```



Primeri za rad u laboratoriji

Primer 1

Primer realizacije slide show-a pomoću niza od četiri slike:

```
<html>
  <head><title>The Four Seasons</title>
  <script language="JavaScript">
    var season = new Array();
    var indx = 0;
    var timeDelay=2000;
    if(document.images){
      season[0]=new Image();
      season[0].src="winter.jpg";
      season[1]=new Image();
      season[1].src="summer.jpg";
      season[2]=new Image();
      season[2].src="fall.jpg";
      season[3]=new Image();
      season[3].src="spring.jpg";
    }
    function changeSeason(){
      var size= season.length - 1;
      if( indx < size ) {
        indx++;
      }
      else {
        indx = 0;
      }
      document.images.src= season[indx].src;
      timeout=setTimeout('changeSeason()', timeDelay);
    }
    function stopShow(){
      clearTimeout(timeout);
    }
  </script>
</head>
```

```

<body bgcolor="cornflowerblue"><center><font face="arial">
<h2>The 4 Seasons</h2><b>
Da bi ste videli slide show, neka kurzor misa bude na slici.<br>
Pomerite kurzor sa slike da bi ste zaustavili show.
<a href="javascript:void(null);"
  onMouseOver="return changeSeason();"
  onMouseOut="return stopShow()">

</a>
<br>
</body>
</html>

```

Primer 2

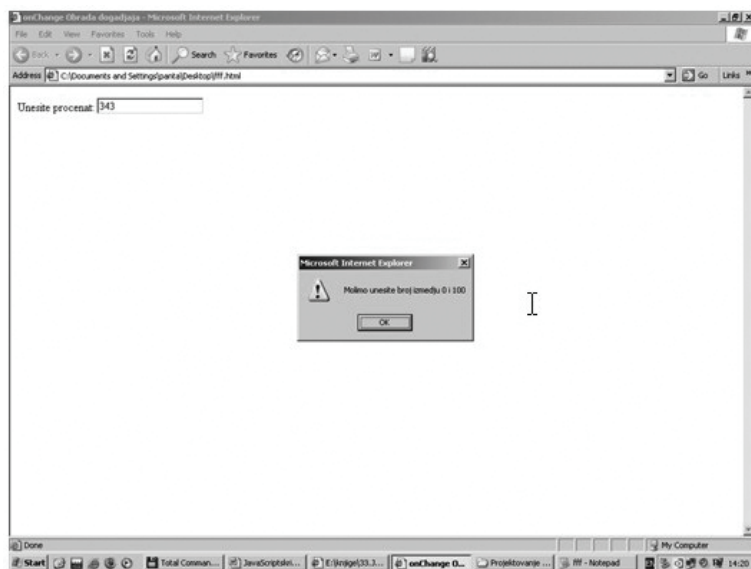
U ovom primeru se reaguje na promenu teksta u okviru tekst polja

```

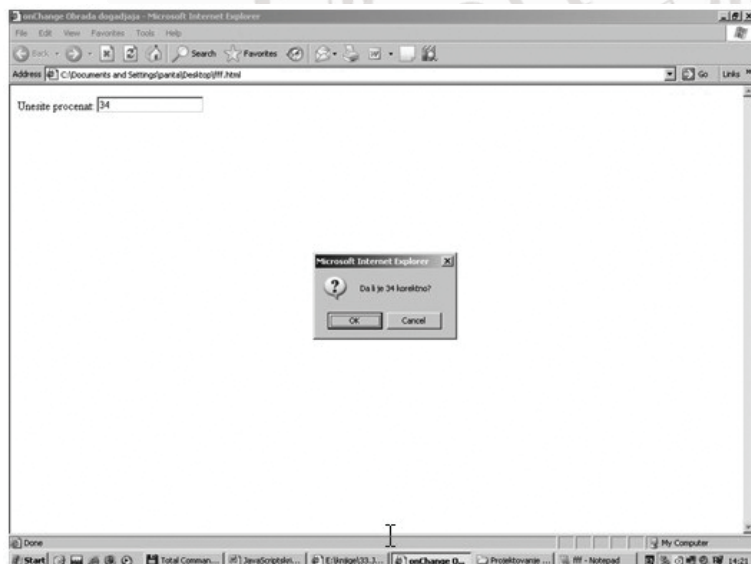
<html>
<head><title>onChange Obrada dogadjaja</title>
</head>
<body>
<form>
  Unesite procenat:
  <input type="text" onChange="
    grade=parseInt(this.value); //konverzija u broj
    if(grade < 0 || grade > 100){
      alert('Molimo unesite broj izmedju 0 i 100');
    }
    else{
      confirm('Da li je '+ grade + ' korektno?');
    }
  ">
</form>
</body>
</html>

```

Nakon unosa broja 343 dobija sledeći izgled



a nakon unosa broja 34 dobija sledeći izgled





18. RAD SA VIŠE PROZORA

JavaScript omogućava da se iz jednog prozora formira, kontroliše ili menja sadržaj u okviru drugog prozora. U ovom poglavlju se razmatraju mogućnosti takvog rada. Prvo se razmatraju specijalne vrste prozora, kao što su alerti, a kasnije se razmatraju mogućnosti rada sa uobilajenim prozorima.

ALERTI

Alerti se koriste unutar HTML stranice kada se želi prikazati određeno obaveštenje. Obaveštenje se prikazuje unutar novog manjeg prozora koji se pojavljuje ispred HTML stranice. U primeru alert se pojavljuje kada se prepozna da je korisnik pritisnuo dugme i poziva se funkcija za prikazivanje alerta.

```
<form action="">
<input type="button" value="Pritisni me" onClick="alert()" />
</form>

<script type="text/javascript">
<!--//
function alert()
{
alert ("Prvi red "+
"i ove je prvi red - \nDrugi red!");
}
//-->
</script>
```

U okviru alerta korišćena je oznaka za prelazak u novi red : "\n"

Primer ALERT poruke koja uzima vrednost koja se upiše u text polje i prikazuje je:

```
<script type="text/javascript">
<!--//
function alert()
{
Ime=document.Form1.VaseIme.value;
alert('Vase ime je '+Ime);
```

```
}  
//-->  
</script>
```

Ime je promenljiva koja čuva ono što je upisano u tekst polje.

Rad sa prozorima

JavaScript omogućava da se iz jednog prozora otvara drugi prozor i da se uspostavlja međusobna komunikacija između njih.

Novi prozor otvara se pomoću naredbe:

```
deteProz = open("noviProzor.html", "deteProz")
```

Ovom naredbom se otvara novi prozor, u okviru koga se prikazuje stranica *noviProzor.html*. Novi prozor je definisan u okviru objekta *deteProz*. Nekom objektu na novom prozoru se može pristupiti pomoću naredbe:

```
deteProz.deteForma.deteObjekat.value
```

gde je *deteForma* - forma definisana u okviru novog prozora, a *deteObjekat* - objekat definisan u okviru forme *deteForma* na novom prozoru.

Pored pristupa objektima na novom prozoru iz starog, može se uspostaviti komunikacija i u drugom smeru. Znači može se objektima na starom prozoru pristupiti iz novog prozora. Primer je:

```
window.opener.document.otacForma.otacObjekat.value
```

gde je *otacForma* - forma definisana u okviru starog prozora, a *otacObjekat* - objekat definisan u okviru forme *otacForma* na starom prozoru

Primer koriscenja:

```
<html>  
<head>  
<title>PROZORI</title>  
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">  
<script language="javascript">  
function prozor( page, width, height, top, left ) {  
    var yes      = 1;
```

```

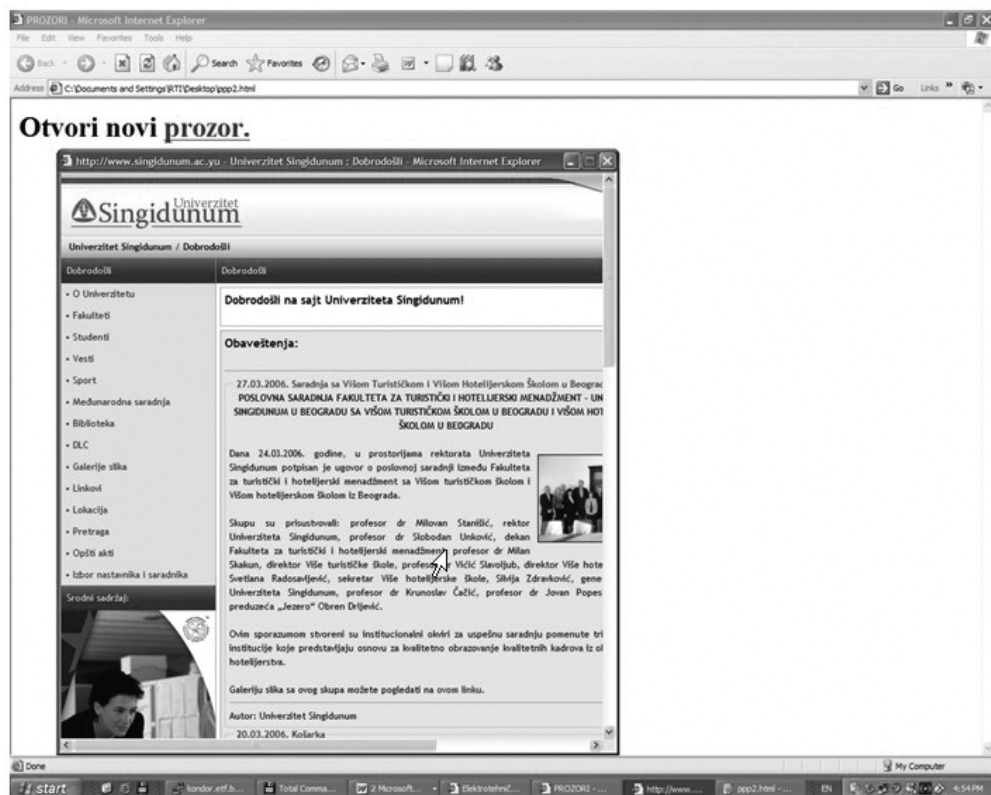
var no          = 0;
var menubar     = no;
var scrollbars   = yes;
var locationbar = no;
var directories = no;
var resizable   = no;
var statusbar   = no;
var toolbar     = no;

features = "" +
    "width=" + width + "," +
    "height=" + height + "," +
    "top=" + top + "," +
    "left=" + left + "";
features += "" +
    (menubar ? ",menubar" : "") +
    (scrollbars ? ",scrollbars" : "") +
    (locationbar ? ",location" : "") +
    (directories ? ",directories" : "") +
    (resizable ? ",resizable" : "") +
    (statusbar ? ",status" : "") +
    (toolbar ? ",toolbar" : "");

window.open( page, 'fullPopup', features );
}
</script>
</head>
<body>
<h1>Otvori novi <a
href="javascript:prozor('http://www.singidunum.ac.yu',720,750,0,0)">
prozor.</a></h1>
</body>
</html>

```

Nakon izvršavanja prethodnog programa, ako je sve pravilno urađeno, dobija se prozor koji predstavlja izgled HTML stranice u Internet Exploreru (slika 2).



ZADACI ZA SAMOSTALAN RAD

Zadatak 1.

U Notepadu napisati HTML kôd koji koristeći JavaScript na klijentskoj mašini upisuje cookie pod imenom Zad1. U okviru HTML stranice postoje dva tekst polja za unos korisnikovog imena i prezimena. Korisnik treba da unese vrednosti u ova dva tekst polja i zatim da se na pritisak dugmeta unete vrednost zapamte u okviru kukija Zad1.

Zadatak 2.

U Notepadu napisati HTML kôd koji obavlja sledeće:

Data je html stranica A.html sa formom FormaA i unutar forme tekst polje TekstPoljeA. Iz ove stranice može se otvoriti novi prozor windowB, koji sadrži formu FormaB i unutar forme tekst polje TekstPoljeB i dugme DugmeB. Na pritisak dugmeta DugmeA poziva se JavaScript funkcija prenesiProzoruB(). Ova funkcija otvara prozor B i sadržaj tekst polja TekstPoljeA prenosi u tekst polje TekstPoljeB.





19. PISANJE I ČITANJE COOKIE-IJA

Cookie su tekstualni fajlovi koji se mogu zapamtiti na mašini klijenta. Internet aplikacije, zbog poboljšane bezbednosti imaju niz ograničenja, kao što je zabranjeno da se čita, piše ili brišu datoteke na sistemu na kome se aplikacija izvršava, tj. zabranjen je svaki pristup lokalnom file sistemu. Jedini izuzetak od ovog pravila predstavljaju upravo cookie fajlovi. Da se i dalje ne bi ugrozila bezbednost sistema, ovi fajlovi moraju da imaju tačno definisan format.

Format koji cookie fajl mora da zadovolji je:

```
ime=vrednost[;EXPIRES=datum][;DOMAIN=imeDomena][;PATH=putanja]
a[;SECURE]
```

gde je:

ime - ime koje definiše upisani cookie. Pomoću njega vrši se čitanje i upis vrednosti u ovaj fajl.

vrednost - je upravo informacija koja se želi zapamtiti na klijentskoj mašini. Ovo polje se čita i definiše u okviru cookie.

datum - je datum koji definiše do kada cookie ostaje upisan na klijentskoj mašini.

imeDomena - definiše jedini domena sa kog cookie može da se čita i da mu se menja vrednost.

putanja - definiše jedinu putanju sa koje cookie može da se čita i da mu se menja vrednost.

SECURE je službena reč koja definiše da se upis i čitanje kukija izvršava preko posebnih, bezbednijih linija.

Opcije EXPIRES, DOMAIN, PATH, SECURE su opcione i nije bitan redosled u kom se pojavljuju. Tekst koji definiše cookie ne sme sadržavati razmake.

Objektu koji predstavlja cookie fajl pristupa se kao objektu u okviru document objekta. Primer čitanja cookie je:

```
var citamCookie=document.cookie
```

U prethodnom primeru su u promenljivoj citamCookie upisani svi kukiji koji postoje na datoj klijentskoj mašini. Sada su ove informacije zapamćene u obliku tekst promenljive. Da bi se pročitao baš određen cookie ovaj tekst mora se parsirati i pronaći deo koji počinje sa imenom kukija koji se traži.

Primer upisa vrednosti u cookie:

```
document.cookie = "primerCookie="+vrednostKojuPamtim+";secure"
```

U prethodnom primeru definisan je cookie pod imenom *primerCookie*. Njegova vrednost je sadržana u promenljivoj *vrednostKojuPamtim*. Nakon izvršavanja ove naredbe na klijentskoj mašini je upisan cookie sa imenom *primerCookie* i vrednošću *vrednostKojuPamtim*.



Primeri za rad u laboratoriji

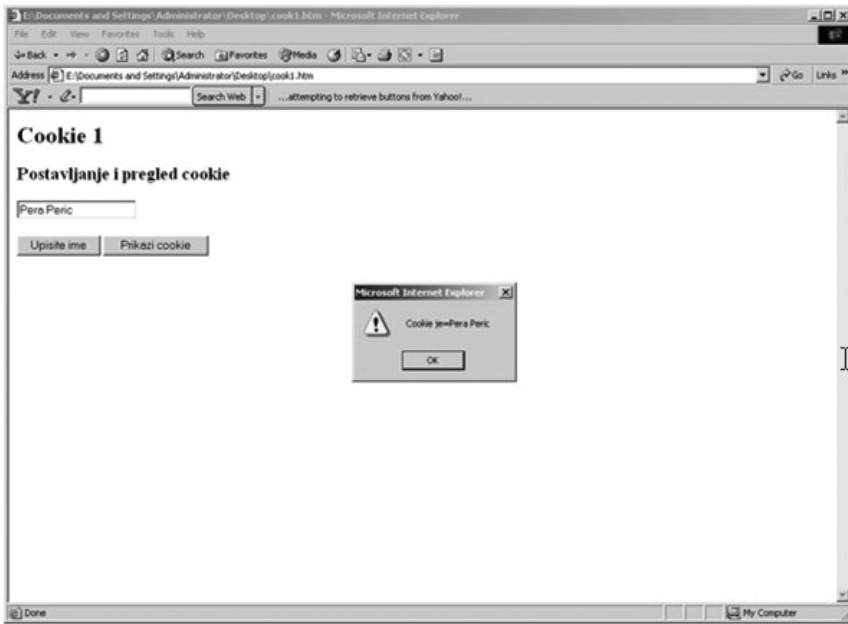
Primer 1

```

<html>
<head>
<script language="javascript">
function postavljanjeCookie(){
document.cookie = 'Cookie je'+document.formal.imeCookie.value;
}
function prikazCookie(){
alert(document.cookie);
}
</script>
</head>
<body>
<h1>Cookie 1</h1>
<h2>Postavljanje i pregled cookie</h2>
<form name="formal">
  <p><input name="imeCookie" type="text" id="imeCookie" size="20">
  </p>
  <p><input type="button" value="Upisite ime" name="B1"
onClick="postavljanjeCookie()">
  <input type="button" value="Prikazi cookie" name="B2"
onClick="prikazCookie()"></p>
</form>
</body>
</html>

```

Nakon izvršavanja prethodnog programa, ako je sve pravilno urađeno, dobija se prozor koji predstavlja izgled HTML stranice u Internet Exploreru (slika 1).



Primer 2

```
<html>
```

```
<head>
```

```
<script>
```

```
go_to = "http://www.singidunum.ac.yu";
```

```
// stranica na koju ide korisnik ako postoji cookies
```

```
broj=1; // broj dana ili minuta zivota cookie
```

```
//funkcija koja racuna duzinu zivota cookie
```

```
function ZivotCookie(duzinaZivota){
```

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

```
    var kraj = new Date(danas.getTime() + duzinaZivota*60*1000);
```

```
//ako stavimo duzinaZivota*24*60*60*1000 onda će varijabla broj da  
predstavlja broj dana
```

```
return kraj.toGMTString();
```

```
}
```

```
function citajCookie(cookieIme){
```

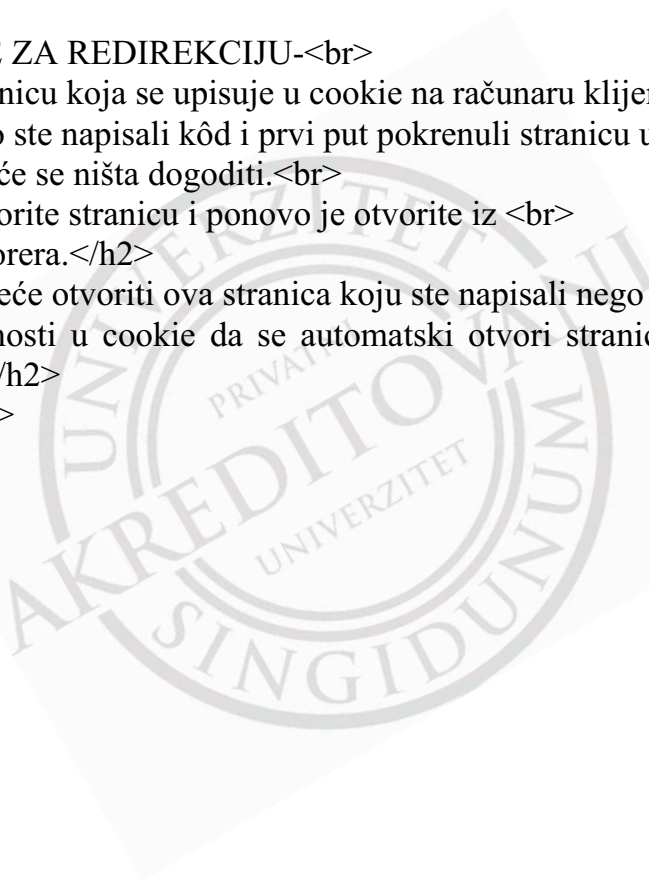
```
    var start = document.cookie.indexOf(cookieIme);
```

```
    if (start == -1){
```

```
        document.cookie = "seenit=da; expires=" + ZivotCookie(broj);
```

```
} else {  
    window.location = go_to;  
}  
}  
citajCookie("seenit");  
</script>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=windows-1250"></head>  
<body>  
<h1> COOKIE ZA REDIREKCIJU-<br>  
odlazak na stranicu koja se upisuje u cookie na računaru klijenta </h1>  
<h2> Nakon što ste napisali kôd i prvi put pokrenuli stranicu u Internet<br>  
Exploreru, neće se ništa dogoditi.<br>  
Treba da zatvorite stranicu i ponovo je otvorite iz <br>  
Internet Explorera.</h2>  
<h2> Tada se neće otvoriti ova stranica koju ste napisali nego će zbog<br>  
upisane vrednosti u cookie da se automatski otvori stranica Univerziteta  
Singidunum. </h2>  
<p>&nbsp;</p>  
</body>  
</html>
```





LITERATURA

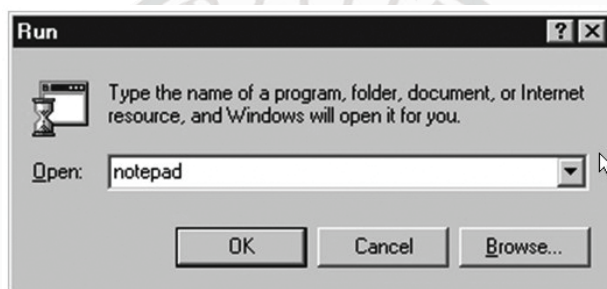
- [1] <http://www.w3.org/MarkUp/>
- [2] D. Flanagan, "JavaScript: The Definitive Guide", O'Reilly, USA, 2001
- [3] D. Raggett, A. Le Hors, I. Jacobs , "HTML 4.0 Specification", <http://www.w3.org/TR/1998/REC-html40-19980424>.
- [4] S. Pemberton et al, "XHTML[tm] 1.0: The Extensible HyperText Markup Language",. <http://www.w3.org/TR/xhtml1>.
- [5] E. van Herwijnen, "Practical SGML", Kluwer Academic Publishers Group, Norwell and Dordrecht, 1990
- [6] A. Navarro, "Effective Web Design", SYBEX, USA, 2001
- [7] <http://validator.w3.org>
- [8] E.A. Mayer, "Cascading Style Sheets: The Definitive Guide", O'Reilly, USA, 2003
- [9] C. Musciano, B. Kennedy, "HTML & XHTML: The Definitive Guide", O'Reilly, USA, 2002
- [10] <http://www.ietf.org/rfc/>
- [11] R. A. Wyke, J. Gilliam, C. Ting, “ Pure JavaScript” , SAMS Publishing, USA, 2001

PRILOZI

PRILOG 1

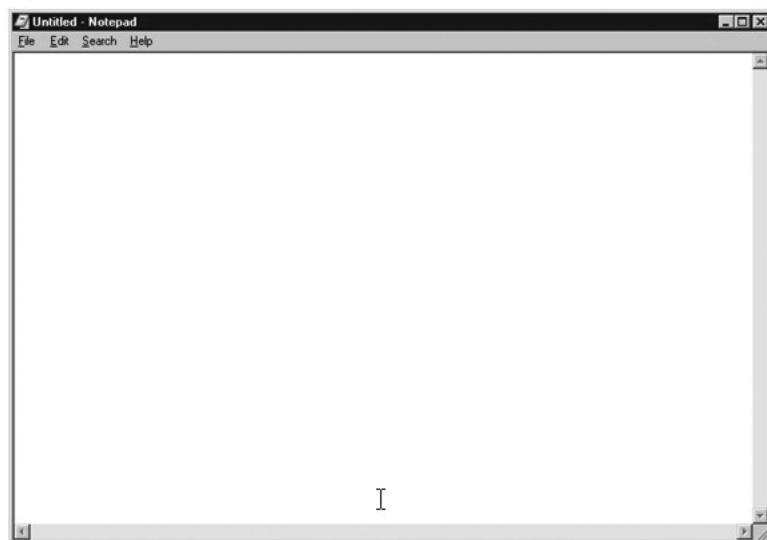
Pisanje programa pomoću tekst editora Notepad

Prvo je potrebno startovati *Notepad* program. U okviru *Windows* operativnog sistema, na statusnoj liniji treba pronaći dugme *Start* i pritisnuti ga. Od ponuđenih opcija treba pronaći opciju *Run* i treba aktivirati ovu opciju. Na prozoru koji se pojavio postoji tekst polje. U njega treba upisati *notepad* (slika 1).



Slika 1 Aktivirana opcija *Run*

Nakon aktiviranja dugmeta *OK*, dobija se prozor tekst editora Notepad (slika 2).



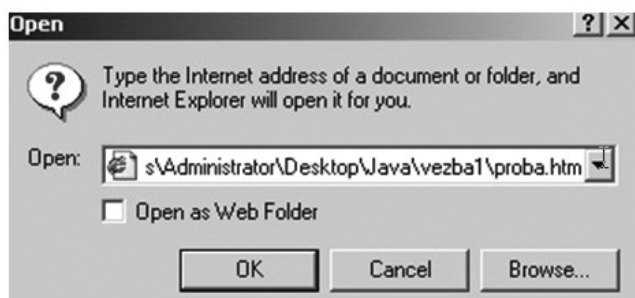
Slika 2 Aktiviran prozor tekst editora Notepad

Na beloj površini piše se kod za HTML stranicu. Stranica se u svakom trenutku može snimiti aktiviranjem na meniju opcija *File*, pa opcije *Save*. Sačuvati stranicu pod nazivom *proba.htm*. Kod snimanja stranice treba obratiti pažnju da dokumenat koji u ovom slučaju treba da predstavlja HTML stranicu mora imati ekstenziju *html* ili *htm*.

Testiranje HTML stranice u Internet browseru

Prvo je potrebno startovati *Internet Explorer* program. U okviru *Windows* operativnog sistema, na statusnoj liniji treba pronaći dugme *Start* i pritisnuti ga. Od ponuđenih opcija treba pronaći opciju *Programs>>Internet Explorer* i treba aktivirati ovu opciju.

U meniju izabrati opciju *File>>Open* pri čemu se otvara dijalog prozor *Open* kao na slici 3. i pritiskom na dugme *Browse* pronaći html stranicu *proba.htm*.



Slika3. Dijalog prozor *Open*

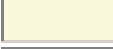
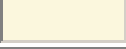
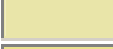
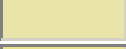
Nakon aktiviranja dugmeta *OK*, dobija se prozor koji predstavlja izgled HTML stranice u Internet Exploreru.



PRILOG 2

Tabela slaganja kodova i boja			
rezultat	kodiranje rečju	heksadecimalno kodiranje	rezultat
	white	FFFFFF	
	ghostwhite	F8F8FF	
	whitesmoke	F5F5F5	
	snow	FFFAFA	
	azure	F0FFFF	
	honeydew	F0FFF0	
	mintcream	F5FFFA	
	lightyellow	FFFFE0	
	floralwhite	FFFAF0	
	ivory	FFFFF0	
	linen	FAF0E6	
	oldlace	FDF5E6	
	seashell	FFF5EE	
	lavenderblush	FFF0F5	
	lavender	E6E6FA	
	gainsboro	DCDCDC	
	lightgrey	D3D3D3	
	silver	C0C0C0	
	darkgray	A9A9A9	
	lightslategray	778899	
	slategray	708090	
	gray	808080	
	dimgray	696969	
	black	000000	
	bisque	FFE4C4	

	wheat	F5DEB3	
	mocasin	FFE4B5	
	peachpuff	FFDAB9	
	mistyrose	FFE4E1	
	pink	FFC0CB	
	lightpink	FFB6C1	
	rosybrown	BC8F8F	
	coral	FF7F50	
	tomato	FF6347	
	orangered	FF3300	
	red	FF0000	
	crimson	DC143C	
	indianred	CD5C5C	
	firebrick	B22222	
	darkred	8B0000	
	maroon	800000	
	brown	A52A2A	
	saddlebrown	8B4513	
	sienna	A0522D	
	chocolate	D2691E	
	darkorange	FF8C00	
	orange	FFA500	
	lightsalmon	FFA07A	
	darksalmon	E9967A	
	salmon	FA8072	
	sandybrown	F4A460	
	navajowhite	FFDEAD	
	blanchedalmond	FFEBCD	
	papayawhip	FFEFD5	

	antiquewhite	FAEBD7	
	beige	F5F5DC	
	cornsilk	FFF8DC	
	palegoldenrod	EEE8AA	
	khaki	F0E68C	
	yellow	FFFF00	
	gold	FFD700	
	darkgoldenrod	B8860B	
	peru	CD853F	
	goldenrod	DAA520	
	tan	D2B48C	
	burlywood	DEB887	
	darkkhaki	BDB76B	
	chartreuse	7FFF00	
	greenyellow	ADFF2F	
	lawngreen	7CFC00	
	lime	00FF00	
	springgreen	00FF7F	
	palegreen	98FB98	
	mediumspringgreen	00FA9A	
	limegreen	32CD32	
	mediumseagreen	3CB371	
	yellowgreen	9ACD32	
	aliceblue	9ACD32	
	mediumaquamarine	66CDAA	
	darkseagreen	8FBC8F	
	forestgreen	228B22	
	green	008000	
	seagreen	2E8B57	

	olive	808000	
	olivedrab	6B8E23	
	darkgreen	006400	
	darkslategray	2F4F4F	
	darkolivegreen	556B2F	
	aquamarine	7FFFD4	
	aqua	00FFFF	
	cyan	00FFFF	
	paleturquoise	AFEEEE	
	turquoise	40E0D0	
	mediumturquoise	48D1CC	
	darkturquoise	00CED1	
	lightseagreen	20B2AA	
	cadetblue	5F9EA0	
	teal	008080	
	darkcyan	008B8B	
	steelblue	4682B4	
	deepskyblue	00BFFF	
	lightskyblue	87CEFA	
	lightsteelblue	B0C4DE	
	powderblue	B0E0E6	
	skyblue	87CEEB	
	cornflowerblue	6495ED	
	dodgerblue	1E90FF	
	mediumslateblue	7B68EE	
	slateblue	6A5ACD	
	royalblue	4169E1	
	blue	0000FF	
	mediumblue	0000CD	

	darkslateblue	483D8B	
	navy	000080	
	darkblue	00008B	
	midnightblue	191970	
	indigo	4B0082	
	blueviolet	8A2BE2	
	darkorchid	9932CC	
	darkviolet	9400D3	
	mediumpurple	9370DB	
	orchid	DA70D6	
	mediumorchid	BA55D3	
	fuchsia	FF00FF	
	magenta	FF00FF	
	hotpink	FF69B4	
	violet	EE82EE	
	thistle	D8BFD8	
	plum	DDA0DD	
	palevioletred	DB7093	
	deeppink	FF1493	
	mediumvioletred	C71585	
	purple	800080	
	darkmagenta	8B008B	

PRILOG 3

TABELA NAJČEŠĆE KORIŠĆENIH KARAKTERA

¢ ¢	¢	Ò Ò	Ò
£ £	£	Ó Ó	Ó
¤ ¤	¤	Ô Ô	Ô
¥ ¥	¥	Õ Õ	Õ
¦ ¦	¦	Ö Ö	Ö
§ §	§	× ×	×
¨ ¨	¨	Ø Ø	Ø
© ©	©	Ù Ù	Ù
ª ª	ª	Ú Ú	Ú
« «	«	Û Û	Û
¬ ¬	¬	Ü Ü	Ü
 ­	­	Ý Ý	Ý
® ®	®	Þ Þ	Þ
¯ ¯	¯	ß ß	ß
° °	°	à à	à
± ±	±	á á	á
² ²	²	â â	â
³ ³	³	ã ã	ã
´ ´	´	ä ä	ä
µ µ	µ	å å	å
¶ ¶	¶	æ æ	æ
· ·	·	ç ç	ç
, ¸	¸	è è	è
¹ ¹	¹	é é	é

°	º	º	ê	ê	ê
»	»	»	ë	ë	ë
¼	¼	¼	ì	ì	ì
½	½	½	í	í	í
¾	¾	¾	î	î	î
¿	¿	¿	ï	ï	ï
À	À	À	ð	ð	ð
Á	Á	Á	ñ	ñ	ñ
Â	Â	Â	ò	ò	ò
Ã	Ã	Ã	ó	ó	ó
Ä	Ä	Ä	ô	ô	ô
Å	Å	Å	õ	õ	õ
Æ	Æ	Æ	ö	ö	ö
Ç	Ç	Ç	÷	÷	÷
È	È	È	ø	ø	ø
É	É	É	ù	ù	ù
Ê	Ê	Ê	ú	ú	ú
Ë	Ë	Ë	û	û	û
Ì	Ì	Ì	ü	ü	ü
Í	Í	Í	ý	ý	ý
Î	Î	Î	þ	þ	þ
Ï	Ï	Ï	ÿ	ÿ	ÿ

PRILOG 4

Uporedne karakteristike programskih jezika JavaScripta i Jave:

JavaScript	Java
Ne kompajlira se, već samo interpretira kod klijenta.	Kompajlira se na serveru pre izvršenja kod klijenta.
Objektno baziran. Koristi postojeće objekte, bez klasa ili nasleđa.	Objektno orjentisan. Appleti sadrže objekte sastavljene od klasa sa nasleđem.
Kôd je integrisan u HTML.	Applet je izdvojen iz HTML-a.
Tip varijable se ne deklarise.	Tip varijable mora da bude deklarisan.
Dinamička povezanost. Objektne reference proveravaju se run-time.	Statička povezanost. Objektne reference moraju postojati u vreme kompajliranja.
Zaštićen. Ne može da zapisuje na hard disk.	Zaštićen. Ne može da zapisuje na hard disk.

PRILOG 5 - Ispitni zadaci

PROGRAMIRANJE GRAFIČKIH APLIKACIJA – Jun 2007

I deo

1. Napisati HTML deo koda kojim se definiše sledeći izgled stranice:

<i>Italik tekst</i>	Boldovan tekst	<u>Prečrtan tekst</u>
I. jedan II. dva III. tri		• lista1: ◦ jedan ◦ dva ◦ tri ◦ četiri • lista2: a. jedan b. dva c. tri
Definisana lista pojam1: definicija 1 pojam2: definicija 2 pojam3: definicija 3		
Slika konj.jpg		

2. Napisati HTML kod stranice PrimerA.html: postoje tri linka - na sajt www.yahoo.com, na stranicu PrimerB.html (koja se nalazi u poddirektorijumu primeri) i na deo stranice PrimerA.html koji je označen sa "drugi".

II deo

1. Napisati kod CSS fajla pomoću koga se definiše da u okviru HTML stranice tekst unutar paragrafa bude ispisivan fontom Arial 12 px, a da leva margina udaljena 2 cm i da stavke liste budu numerisane malim Rimskim ciframa.

2. Pomoću JavaScripta prikazati sve proste brojeve između 1 i 100.

3. U okviru JavaScript programa definisan je broj. Napisati zbir cifara definisanog broja.

III deo

1. Napisati kod sledeće stranice: data su dva tekst polja (TekstMesec i TekstIme) i jedno dugme. Kada se pritisne dugme, potrebno je u TekstIme upisati ime meseca čiji je redni broj u nedelji definisan u TekstMesec.
2. Napisati JavaScript funkciju citam() koja sa forme FormaA prihvata ime cookie-ija koje je upisano u tekst polje TekstA i vrednost definisanog cookie-ija upisuje u tekst polje TekstB.



PROGRAMIRANJE GRAFIČKIH APLIKACIJA – Novembar 2006

I deo

1. Napisati HTML deo koda kojim se slika primerSlike.gif smešta u prvu ćeliju tabele sa 2 reda i kolone HTML stranice i predstavlja link ka stranici Druga.html.

2. Napisati HTML kod stranice sa formom koja sadrži 2 polja za potvrdu (check box), dva tekst polja, jednu listu sa tri opcije jedno submit dugme i jedno obično dugme.

II deo

1. Napisati kod CSS fajla pomoću koga se definiše da u okviru HTML stranice tekst unutar paragrafa bude ispisan fontom Vedrana 20 px,, da bude italik i u plavj boji i da stavke liste budu numerisane malim Rimskim ciframa.

2. Pomoću JavaScripta izračunati zbir svih neparnih brojeva između 10 i 100.

3. U okviru JavaScript programa prikazati sve brojeve između 100 i 200 koji su deljivi sa 5, a nisu deljivi sa 6 (kao na primer 100).

III deo

1. Napisati JavaScript stranicu koja prikazuje tekst polje i jedno dugme. Pritiskom na dugme tekst koji predstavlja broj i koji se unosi unutar tekst polja se prihvata i zbir cifara broja se prikazuje korisniku u novom prozoru.

2. Pomoću JavaScript upisati cookie koji se zove ispit i čija vrednost je trenutno vreme izvršavanja programa.

PROGRAMIRANJE GRAFIČKIH APLIKACIJA – Jun 2006

I deo

1. Napisati HTML kod stranice:

Aaa		▪ Bbb ▪ Hhh ▪ Kkk
Ccc	Mmm	www.google.com
Nnn	Qqq	

2. Napisati HTML kod stranice koja je po horizontali podeljena na tri dela (koja su u odnosu 40%:20%:40). U prvom delu stranice od vrha prikazana je stranica PrimerA.html. Drugi deo stranice od vrha je podeljen vertikalno na dva jednaka dela, i kojima se izvršavaju stranice PrimerB.html i PrimerC.html. U trećem delu od vrha prikazana je stranica PrimerD.html.

II deo

1. Koristeći CSS formatirati HTML stranice tako da se u okviru naslova nivoa 1 tekst bude prikazan žutom bojom, a pozadina plavom bojom, nenumerisane liste sa kvadratima brojevima, a u okviru paragrafa tekst poravnat udesno.

2. Pomoću JavaScripta prikazati sve brojeve između 1 i 100 čiji je zbir cifara jednak 8.

3. U okviru JavaScript programa dat je broj koji predstavlja dan u mesecu i broj koji predstavlja redni broj meseca. Napisati koji je redni broj dana u godini definisani tim datumom.

III deo

1. Napisati JavaScript stranicu koja prikazuje tekst polje i jedno dugme. Pritiskom na dugme tekst koji predstavlja ime meseca i koji se unosi unutar tekst polja se prihvata i redni broj meseca se prikazuje korisniku u novom prozoru.

2. Pomoću JavaScript koda izračunati koliko ima dana do sledeće Nove godine (do 1.1. sledeće godine).

PROGRAMIRANJE GRAFIČKIH APLIKACIJA – Oktobar 2006

I deo

1. Napisati HTML deo koda kojim se slika primerSlike.gif smešta u prvu ćeliju tabele sa 2 reda i kolone HTML stranice i predstavlja link ka stranici Druga.html.
2. Napisati HTML kod stranice:

Ime	Prezime :	Zanimanje :
• jedan • dva • tri	1. pet 2. sest 3. sedam	Osam

II deo

1. Napisati kod CSS fajla pomoću koga se definiše da u okviru HTML stranice tekst unutar paragrafa bude ispisian fontom Arial 18 px., da bude boldovan i u zelenoj boji i da stavke liste budu numerisane malim Rimskim ciframa.
2. Pomoću JavaScripta izračunati zbir svih neparnih brojeva između 1 i 100.
3. U okviru JavaScript programa prikazati sve brojeve između 100 i 200 koji su deljivi sa 4, a nisu deljivi sa 3 (kao na primer 104).

III deo

1. Napisati kod HTML stranice koja prikazuje dva tekst polja Tekst1 i Tekst2, grupu od dva radio dugmeta Jedan i Dva i tekst prostor. Kada se potvrdi radio dugme Dva treba preneti sadržaj polja Tekst1 u tekst prostor, a kada se potvrdi radio dugme Jedan treba preneti sadržaj polja Tekst2 u tekst prostor.
2. Pomoću JavaScript upisati cookie koji se zove NG i čija vrednost je broj dana od prethodne Nove godine do trenutka izvršavanja programa.

PROGRAMIRANJE GRAFIČKIH APLIKACIJA – Jun 2007

I deo

1. Napisati HTML deo koda kojim se definiše sledeći izgled stranice:

Forma za ispit:

Ime i prezime :

Grad: Država:

Koji put polažete : ☐ Prvi ☐ Drugi ☐ Treći

Omiljena oblast: ☐ Javascript ☐ HTML

Korisničko ime: lozinka:

Ovde ispisati komentar:

Komentar:

2. Napisati HTML kod stranice koja je po horizontali podeljena na tri dela (koja su u odnosu 40%:20%:40). U prvom delu stranice od vrha prikazana je stranica PrimerA.html. Drugi deo stranice od vrha je podeljen vertikalno na dva jednaka dela, i kojima se izvršavaju stranice PrimerB.html i PrimerC.html. U trećem delu od vrha prikazana je stranica PrimerD.html.

II deo

1. Napisati kod CSS fajla pomoću koga se definiše da u okviru HTML stranice tekst unutar paragrafa bude ispisivan fontom Arial 18 px., da bude boldovan i u zelenoj boji i da stavke liste budu numerisane malim Rimskim ciframa.
2. Pomoću JavaScripta izračunati zbir svih parnih brojeva između 1 i 100.
3. U okviru JavaScript programa prikazati sve brojeve između 100 i 200 koji su deljivi sa 3, a nisu deljivi sa 4 (kao na primer 102)..

III deo

1. Za stranicu realizovanu u prvom zadatku izvršiti proveru da li su sva tekstualna polja uneta, da li se tekst unet za ime poklapa sa tekstom unetim za lozinku i ako se potvrđena oba polja za potvrdu prikazati alert sa pitanjem „Da li ste sigurni?“.
2. Napisati JavaScript stranicu koja prikazuje tekst polje i jedno dugme. Pritiskom na dugme tekst koji predstavlja ime meseca i koji se unosi unutar tekst polja se prihvata i redni broj meseca se prikazuje korisniku u novom prozoru.