UNIVERZITET SINGIDUNUM
FAKULTET ZA INFORMATIKU I MENADŽMENT

Mirko Milošević

Izrada web prezentacije „Kuća snova“

- Diplomski rad -

Beograd, 2009.

UNIVERZITET SINGIDUNUM
FAKULTET ZA INFORMATIKU I MENADŽMENT

IZRADA WEB PREZENTACIJE „KUĆA SNOVA“

- Diplomski rad -

Mentor:
Prof. dr Mladen Veinović

Student:
Mirko Milošević
Br. indeksa:
III-18/2007

Beograd, 2009.

UNIVERZITET SINGIDUNUM
FAKULTET ZA INFORMATIKU I MENADŽMENT
Beograd, Danijelova 32

Kandidat: Mirko Milošević

Broj indeksa: III-18/2007

Smer: Programiranje i projektovanje

Tema: Izrada web prezentacije „Kuća snova“

Zadatak: Opisati tehnologije i procese izrade web prezentacije „Kuća snova“.

MENTOR

_____
Prof. dr Mladen Veinović

DEKAN

_____
Prof. dr Mladen Veinović

Apstrakt

U ovom radu opisane su neke od najznačajnijih tehnologija koje se koriste u web programiranju i web dizajnu. Od *(X)HTML* i *CSS* tehnologije za izradu statičkih web stranica, preko kombinacije *PHP* i *MySQL* tehnologija za ubacivanje dinamičkih sadržaja do *Flash* tehnologije za dodavanje multimedijalnog karaktera stranici. Nakon opisa tehnologija, prikazana je sama izrada i finalni izgled prezentacije „Kuća snova“.

Ključne reči

Web, *HTML*, *PHP*, *MySQL*.

Abstract

In this paper some of the most important web programming and web design technologies are described. From *(X)HTML* and *CSS* technologies for creating static pages, through combination of *PHP* and *MySQL* for adding dynamic content, ending with *Flash* for adding multimedia to the page. Finally, the process of creation and the final view of web presentation “Dream house” are presented.

Key words

Web, *HTML*, *PHP*, *MySQL*.

SADRŽAJ

1. UVOD

U ovom radu baviću se tehnologijama za izradu web prezentacija koje su u osnovi statičke, ali poseduju i određene elemente kod kojih se sadržaj generiše dinamički. Nakon Uvoda, uopšteno ću izložiti osnovne karakteristike tehnologija neophodnih za razvoj dinamičkih web prezentacija, a u kasnijim poglavljima detaljno ću opisati svaku od tehnologija koju sam odlučio da implementiram u svoju web prezentaciju. To su *XHTML*, *CSS*, *PHP*, *MySQL* i *Flash* tehnologije. O njima ću govoriti tako što ću najpre predstaviti njihov istorijat, osobenosti, način upotrebe, glavne prednosti i mane.

U praktičnom delu rada opisaću proces izrade i objavljivanja web prezentacije „Kuća snova“. Ovu prezentaciju sam izradio aprila 2008. godine za potrebe istoimenog *reality show*-a u produkciji produkcijsko-marketinške agencije *Advantage* iz Beograda. Emisija je svakodnevno emitovana tokom dvanaest nedelja na televiziji *B92* koja poseduje nacionalnu frekvenciju i bila je veoma gledana. U skladu sa tim prezentacija je morala da odgovori na visoke zahteve u smislu velikog broja poseta, multimedijalnih sadržaja, svakodnevnog ažuriranja sadržaja i administracije foruma.

Imajući u vidu da u dotadašnjem radu nisam imao iskustva sa ovako velikim projektima, čekalo me je dosta posla i širenja saznanja. Upravo zbog toga smatram da je to zanimljivo elaborirati u ovom diplomskom radu.

2. INTERNET RAZVOJ

Pod pojmom Internet razvoja podrazumeva se bilo koja aktivnost vezana za razvoj web prezentacija. U te aktivnosti spadaju web dizajn, razvoj web sadržaja, kodiranje na klijentskoj i/ili serverskoj strani, konfigurisanje web servera, razvoj elektronskog poslovanja, itd. Međutim, među profesionalcima, najčešće vlada mišljenje da je pojam Internet razvoja vezan isključivo za nedizajnerske aspekte razvoja, kao što su pisanje *markup*-a i kodiranje. Taj segment Internet razvoja naziva se još i web programiranje.

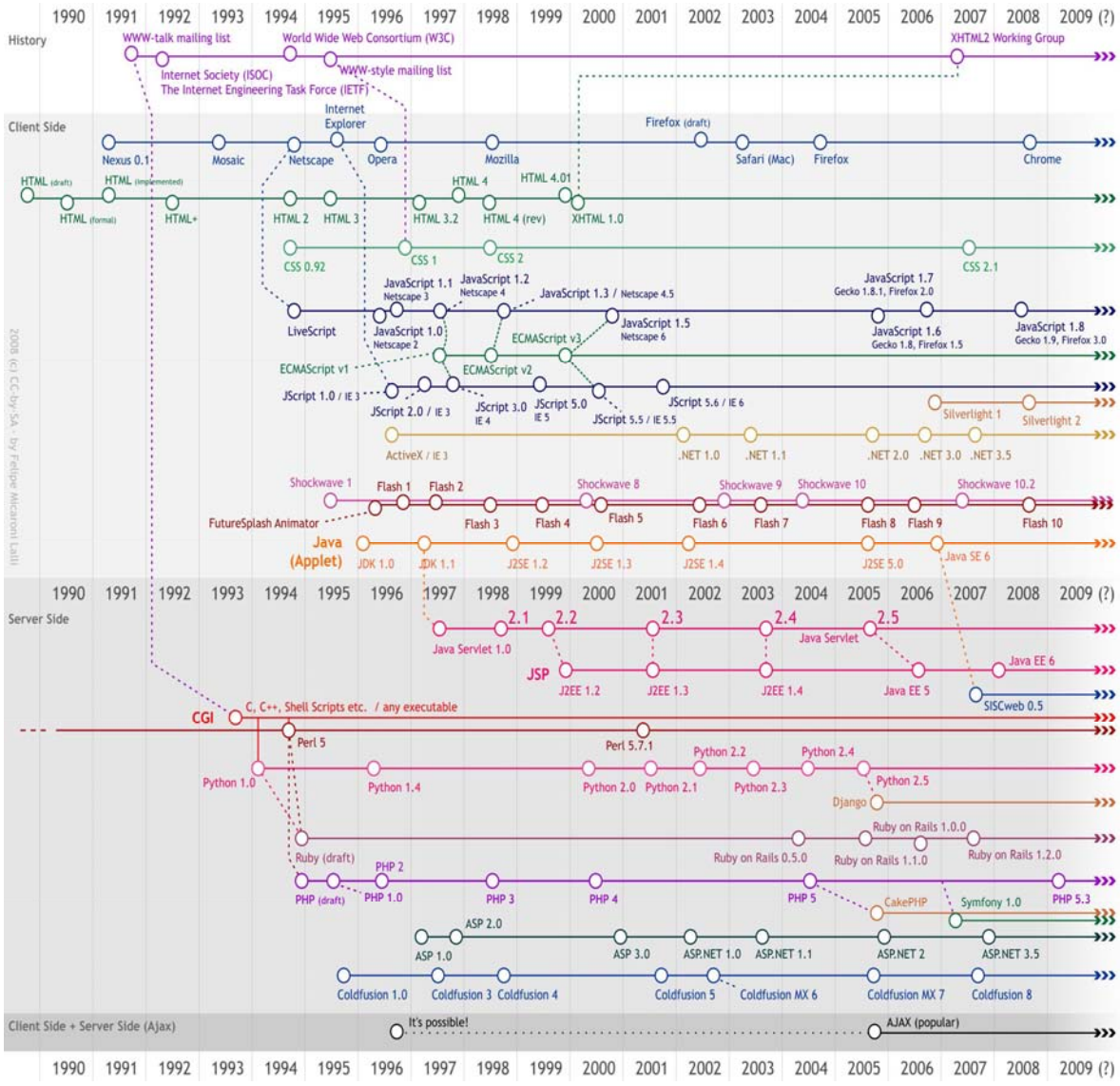
Segment Internet razvoja koji se bavi dizajnom web prezentacija naziva se web dizajn. Web dizajn predstavlja veštinu kreiranja načina prezentovanja određenog sadržaja krajnjem korisniku kroz web pregledače. Može se posmatrati kao podvrsta grafičkog dizajna, namenjena razvoju i stilizovanju objekata za upotrebu u sklopu web prezentacija. Često uključuje više disciplina kao što su grafički dizajn, fotografija, animacija, tipografija, korporativni identitet, komunikacioni dizajn, interakcioni dizajn, marketing, itd.

Kompleksnije web prezentacije i veće organizacije, obično, zahtevaju čitave Internet razvojne timove koji mogu da se sastoje od stotina programera, dizajnera i IT tehničara. Male organizacije, vrlo često, imaju jednog programera, koji je ujedno i IT tehničar, i jednog dizajnera.

Tehnologije koje se koriste u Internet razvoju su:

- *Markup* jezik (*HTML*, *XHTML* i *XML*);
- Jezik liste stilova (*CSS*, *XSL*);
- Kodiranje na serverskoj strani (*PHP*, *ASP*, *CGI*, *Java*, *.NET*);
- Kodiranje na klijentskoj strani (*JavaScript*, *VBScript*, *AJAX*);
- Baza podataka (*MS SQL*, *MySQL*, *Oracle*);
- Multimedija (*Flash*, *Silverlight*).

Koje od navedenih tehnologija će biti upotrebljene u razvoju web prezentacije zavisi u mnogome od tipa prezentacije koja se razvija. Web prezentacije mogu biti sa statičkim sadržajem ili se njihov sadržaj, pa čak i izgled, može menjati u zavisnosti od raznovrsnih faktora kao što su unos podataka od strane krajnjeg korisnika ili web programera ili promena u kompjuterskom okruženju (npr. promena u bazi podataka koja je povezana sa web prezentacijom). Takve prezentacije se nazivaju dinamičke, odnosno web prezentacije sa dinamičkim sadržajem.



Slika 1: Hronološki razvoj web programiranja

2.1. *Markup* jezik

Markup jezik predstavlja skup beleški koje služe da opišu na koji način će neki dokument biti struktuiran, postavljen ili formatiran. Ovaj jezik može biti u formi rukopisa, kada se beleške nalaze između ili pored teksta, ili u formi *markup* koda koji se koristi u računarskim sistemima za podešavanje i obradu teksta.

Najpoznatiji primer *markup* jezika koji se danas koristi u programiranju je *HTML* (*HyperText Markup Language*). On se zasniva na *SGML* (*Standard Generalized Markup Language*) jeziku i prati mnoge konvencije korišćene u izdavačkoj industriji u komunikaciji štampanih radova između autora, urednika i štampe.

2.2. Jezik liste stilova

Jezik liste stilova (*style sheet language*) predstavlja kompjuterski jezik koji se upotrebljava za opisivanje načina prezentovanja struktuiranih dokumenata, odnosno onih dokumenata čiji su delovi u potpunosti definisani i kategorizovani. Program koji treba da prezentuje određeni dokument može to učiniti u različitim stilovima zato što je sadržaj dokumenta dobro formulisan. Najpoznatiji jezik liste stilova koji se trenutno upotrebljava je *CSS* (*Cascading Style Sheets*). Kaskadne liste stilova se koriste za stilizovanje dokumenata napisanih u *HTML*, *XHTML* ili nekom drugom *markup* jeziku. Jedna od ključnih karakteristika struktuiranih dokumenata je ta da se sadržaj može koristiti u različitim kontekstima i prezentovati na različite načine. Različite liste stilova mogu se priključiti logičkoj strukturi za izradu raznih prezentacija.

Da bi se struktuirani dokument prezentovao neophodno je na sadržaj dokumenta primeniti skup stilskih pravila kao što su boje, fontovi, izgled, itd. Skup stilskih pravila naziva se lista stilova. Liste stilova u slučaju pisanih dokumenata imaju dugu istoriju upotrebe od strane urednika i tipografa kako bi se osigurala jednoličnost prezentacije, pravopisa i interpunkcije. U slučaju elektronskih dokumenata, liste stilova se koriste za vizuelnu prezentaciju, a dosta manje za pravopis i interpunkciju.

2.3. Kodiranje na serverskoj/klijentskoj strani

Kodiranje na serverskoj ili klijentskoj strani (*server-side/client-side coding*) jesu tehnologije u Internet razvoju koje omogućavaju generisanje dinamičkog sadržaja web prezentacija. Osnovna razlika između ova dva načina kodiranja je ta što se prilikom nekog korisnikovog zahteva, u prvom slučaju, skripti pokreću i obrađuju na serveru, a u drugom, na lokalnom računaru odnosno u Internet pregledaču. Tipični predstavnici prve tehnologije su *PHP*, *ASP*, *CGI*, *Java*, *.NET*, a druge *JavaScript*, *VBScript*, *AJAX*. Prednost prve tehnologije je u tome što sam proces nije zavisan od platforme na kojoj radi korisnik zato što se kompletna obrada odvija na serveru, a zahtevi ka serveru i rezultati obrade sa servera se šalju nekim standardnim protokolom koji je podržan u većini Internet pregledača (npr. *HTTP* ili *FTP*).

Obe tehnologije su jako bitan element dinamičkog *HTML*-a zato što omogućavaju web prezentacijama da budu programabilne, tj. da imaju različite i promenljive sadržaje u zavisnosti od unosa korisnika, uslova okruženja (npr. vreme) ili drugih promenljivih i ostvare komunikaciju sa bazama podataka ili nekim drugim vidom čuvanja podataka.

2.4. Baza podataka

Baza podataka predstavlja skup podataka organizovanih za brzo pretraživanje i pristup koji, zajedno sa sistemom za administraciju, organizovanje i memorisanje tih podataka, čini sistem baze podataka. Iz ugla korisnika, podaci unutar baze su na logički način povezani i obično predstavljaju neke aspekte realnog sveta (npr. članovi i knjige u biblioteci).

Korisnici pristupaju bazi podataka prvenstveno preko upitnika (*query*). Korišćenjem ključnih reči i svrstavanjem komandi korisnici mogu brzo da pronađu,

preurede, grupišu i odaberu oblast u mnogim zapisima koje treba prikazati ili pomoću kojih treba generisati izveštaje o određenom skupu podataka.

Postoje više vrsta baza podataka, u zavisnosti od načina na koji su podaci interno organizovani. Tako postoje hijerarhijske, mrežne, relacione, objektno-orijentisane i druge baze podataka. Danas se najčešće sreće model relacione baze podataka. To je tabelarna baza podataka u kojoj su podaci definisani tako da se mogu reorganizovati i može im se pristupiti na više načina. Baze se mogu razvrstati i po tipu sadržaja na bibliografske, tekstualne, numeričke, slikovne, itd.

Baze podataka danas imaju jako široko polje primene u kompjuterskim aplikacijama. One su najčešći oblik čuvanja podataka u velikim višekorisničkim sistemima gde je potrebna koordinacija između više korisnika, ali ih i samostalni korisnici smatraju korisnim, a i mnoge aplikacije za elektronsku poštu i lične organizatore koriste ovu tehnologiju.

Bazama podataka upravlja se pomoću menadžera baze podataka koji omogućava administratorima da definišu prava pristupa (čitanja/upisa) određenim korisnicima, generišu određene izveštaje i vrše analize. Baze podataka i menadžeri su zastupljeniji u velikim sistemima, ali postoje i u manjim i srednjim sistemima. *SQL* (*Structured Query Language*) je standardni jezik za kreiranje interaktivnih upita i ažuriranje baza podataka.

2.5. Multimedija

Multimedija (ili multimedij) predstavlja sadržaj koji je sačinjen od kombinacije više različitih tipova sadržaja. To može biti kombinacija teksta, zvuka, slike, animacije, videa i interaktivnog sadržaja.

Multimedija se obično snima, prikazuje ili joj se pristupa putem uređaja za obradu informacionih sadržaja u vidu kompjuterizovanih ili elektronskih uređaja, ali može biti prezentovana i uživo. Isto tako, može se posmatrati i kao elektronski medij koji se koristi za čuvanje i doživljavanje multimedijalnog sadržaja.

Neki od najkorišćenijih softvera za kreiranje multimedijalnih sadržaja su: *Adobe Flash*, *Adobe Photoshop*, *Sony SoundForge*, *Autodesk Maya*, *Microsoft Silverlight*, *Microsoft PowerPoint*, a postoje i mnogi drugi.

3. *(X)HTML*

3.1. *HTML*

Hipertekst *markup* jezik ili *HTML* (*HyperText Markup Language*) je dominantan *markup* jezik za programiranje web prezentacija koji obezbeđuje sredstva za kreiranje struktuiranih dokumenata koji sadrže tekstualne, multimedijalne i druge tipove sadržaja. On omogućava definisanje određenih struktura u dokumentu kao što su naslovi, paragrafi, liste, linkovi, itd. Takođe, daje mogućnost da se u okviru dokumenta nađu i ugnežđene slike, interaktivne forme, ugnežđeni kodovi nekog drugog jezika (npr. *JavaScript*) i drugi objekti. *HTML* se piše u formi tagova obuhvaćenih izlomljenim zagradama (`<tag>`).

Tim Berners-Li (Berners-Lee) razvio je i definisao *HTML* na osnovu *SGML markup* jezika i razvio prvi Internet pregledač/editor krajem 1990. godine. Prva verzija pregledača radila je isključivo na *NeXT* platformi i obrađivala samo tekstualne datoteke. Krajem 1991. godine objavio je na Internet-u dokument pod nazivom „HTML Tags“, koji opisuje dvadeset elemenata koji predstavljaju osnovu *HTML*-a. Svi elementi, izuzev *hyperlink*-a, bili su pod snažnim uticajem *SGML*-a. Narednih godina došlo je do oživljavanja ovog sistema u Internet zajednici i značajnog proširenja originalnih specifikacija, ali i dalje nije bio postavljen pravi standard.

HTML 2.0 završen je 1995. godine i predstavljao je prvu *HTML* specifikaciju namenjenu kao standard za buduće implementacije. Od 1996. godine *HTML* specifikaciju održava *W3C* (*World Wide Web Consortium*) uz uticaj proizvođača komercijalnog softvera. Januara 1997. godine objavljen je *HTML 3.2* koji je u sebi posedovao veliki broj tagova za opis vizuelnog izgleda elemenata u dokumentu, preuzetih od *Netscape*-a. Već decembra iste godine objavljena je verzija *4.0* iz koje je izuzet veliki broj *Netscape*-ovih tagova, zarad popularizacije upotrebe listi stilova.

Poslednja specifikacija izdata je 1999. godine od strane *W3C* i nosi oznaku *HTML 4.01*. *HyperText Markup Language*, 2000. godine, postaje međunarodni standard (*ISO/IEC 15445:2000*).

```
# <html>
# <body>
#
# <h1>Moj prvi naslov</h1>
#
# <p>Moj prvi paragraf.</p>
#
# </body>
# </html>
```

Primer 1: Izgled *HTML* koda

HTML markup se sastoji od nekoliko ključnih komponenti. To su elementi, atributi elemenata, karakterno bazirani tipovi podataka, karakterne i objektne reference i deklaracija tipa dokumenta.

```
# <!DOCTYPE html>
# <html>
# <head>
# <title>Hello HTML</title>
# </head>
# <body>
# <p>Hello World!!</p>
# </body>
# </html>
```

Primer 2: Izgled *HTML* dokumenta

3.1.1. Elementi

Elementi predstavljaju osnovnu strukturu *HTML*-a. Oni imaju dve osobine: atribut i sadržaj. Svaki atribut i sadržaj svakog elementa ima određene restrikcije koje se moraju poštovati kako bi *HTML* dokument bio validan. Svaki element obično ima početni tag (`<element>`) i završni tag (`</element>`). Atributi elementa se nalaze u okviru početnog taga, a sadržaj elementa između tagova.

```
# <element atribut="vrednost">Sadržaj</element>
```

Primer 3: Izgled elementa

Neki elementi, kao što je <br>, nemaju sadržaj i ne moraju imati završni tag. Postoji nekoliko tipova *markup* elemenata u *HTML*-u, od kojih su neki: strukturni *markup* (<h1>Naslov</h1>), prezentacioni *markup* (italic) i hipertekst *markup* (Link).

3.1.2. Atributi elemenata

Atributi elemenata, u većini slučajeva, predstavljaju parove oblika atribut="vrednost" i pisani su u početnom tagu, nakon naziva elementa. Vrednost atributa može biti uokvirena navodnicima, apostrofima, a može ostati i neuokvirena (poslednja opcija se ne preporučuje iz sigurnosnih razloga). Takođe, postoje i atributi koji nemaju vrednost, već samim svojim prisustvom utiču na element u kom se nalaze (npr. atribut ismap u elementu <img>). Neki od atributa koje gotovo svaki element može da ima su: id, class, style i title.

```
# <span id="ID" class="Klasa" style="color:yellow;"
# title"HyperText Markup Language">HTML</span>
```

Primer 4: Upotreba atributa

3.1.3. Karakterne i objektne reference

Imajući u vidu da su određeni karakteri rezervisani za pisanje *markup*-a, definisane su karakterne reference koje omogućavaju da se ti karakteri nađu u tekstu. Na primer, karakteri < i & u *markup* jeziku predstavljaju početak taga i početak karakterne reference, respektivno. Da bi ovi karakteri bili prikazani u tekstu moraju biti napisani kao < i &. Na ovaj način moguće je ispisati i karaktere koji se teško kucaju ili uopšte nisu dostupni u setu karaktera datog dokumenta. Znak © bio bi napisan kao &copy, a € kao &#8364.

3.1.4. Deklaracija tipa dokumenta

Obavezno je da svaki *HTML* dokument počne deklaracijom tipa dokumenta (*Document Type Declaration*, skr. *doctype*). U pregledačima *doctype* selektuje režim iščitavanja. Originalna namena *doctype*-a je da omogući validaciju

dokumenata *SGML* alatima baziranu na *DTD* (*Document Type Definition*). *DTD* na koji ukazuje *doctype* sadrži mašinski-čitljivu gramatiku na osnovu koje se određuje dozvoljen odnosno nedozvoljen sadržaj za dati dokument. Web pregledači ne čitaju *DTD*.

```
# <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
# "http://www.w3.org/TR/html4/strict.dtd">
```

Primer 5: Deklaracija tipa dokumenta

Navedena deklaracija ukazuje na striktni *DTD HTML* jezika u verziji *4.01* (*Strict DTD*). U ovoj verziji izostavljeni su svi prezentacioni elementi, a kompletno formatiranje dokumenta prepušteno je *CSS*-u i `<span>` i `<div>` tagovima. Pored striktne, postoje još i prelazne i frejmset verzije *DTD*-a. *Transitional DTD* namenjen je postepenom prelasku na striktni režim, dopuštajući upotrebu prezentacionih elemenata, a *Frameset DTD* omogućava upotrebu okvira u *HTML* dokumentima.

3.2. *XHTML*

Proširivi *HTML* ili *XHTML* (*Extensible HyperText Markup Language*) je *markup* jezik koji poseduje mogućnosti identične onim u *HTML* jeziku, ali koristi *XML* (*Extensible Markup Language*) sintaksu. *XHTML* se može tretirati kao presek *HTML* i *XML* jezika u mnogim pogledima. *HTML* je bio definisan kao primena veoma fleksibilnog *SGML* jezika, dok je *XHTML* aplikacija *XML* jezika, restriktivnijeg podseta *SGML*-a. Imajući u vidu da *XHTML* dokumenti moraju biti dobro formirani, za njihovo tumačenje mogu se koristiti standardni *XML* tumači, dok je za *HTML* dokumente neophodno postojanje posebnih, relativno kompleksnih, popustljivijih tumača.

3.2.1. *XHTML 1.0*

U decembru 1998. godine *W3C* objavio je radni nacrt pod nazivom „Reformulating HTML in XML“. Time se uvodi radni naziv „Voyager“ za novi *markup* jezik baziran na *HTML 4.01* jeziku, ali sa striktnim pravilima sintakse *XML*

jezika. Februara 1999. godine specifikacija je promenila naziv u *XHTML 1.0*, a januara 2000. godine postala je zvanična *W3C* preporuka. Postoje tri zvanična *DTD*-a za *XHTML 1.0*:

- *Strict XHTML 1.0*,
- *Transitional XHTML 1.0* i
- *Frameset XHTML 1.0*.

Strict XHTML 1.0 (ekvivalent *Strict HTML 4.01*) uključuje elemente i atribute koji nisu obeleženi kao nepogodni u *HTML 4.01* specifikaciji. *Transitional XHTML 1.0* (ekvivalent *Transitional HTML 4.01*) uključuje prezentacione elemente isključene iz striktne verzije. *Frameset XHTML 1.0* (ekvivalent *Frameset HTML 4.01*) omogućava definisanje frejmova unutar dokumenata.

3.2.2. Modularizacija *XHTML*-a

Modularizacija *XHTML* jezika obezbeđuje više mogućnosti podešavanja i proširivanja samog jezika. Zamisao ove karakteristike je da pomogne *XHTML* jeziku da se proširi na nove platforme. Početni nacrt specifikacije pod nazivom „Modularization of XHTML“ postao je dostupan aprila 1999. godine, a dostigao status preporuke aprila 2001. godine. Neki od jezika razvijenih ovom tehnikom su: *XHTML 1.1*, *XHTML Basic 1.0* i *XHTML Print* (namenjen štampi stranica sa mobilnih telefona na nisko-budžetne štampače). Ovu specifikaciju je u oktobru 2008. godine potisnula specifikacija „XHTML Modularization 1.1“.

3.2.3. *XHTML 1.1*

XHTML 1.1 je nastao kao rezultat evolucije specifikacije pod nazivom „Modularization of XHTML“. U septembru 1999. godine *W3C* je objavio prve nacrte, a u maju 2001. *XHTML 1.1* je postao zvanična preporuka. Moduli kombinovani unutar *XHTML 1.1* jezika efektivno rekreiraju *Strict XHTML 1.0* sa dodatnim elementima za bolju podršku istočnoazijskih jezika (`<ruby>`, `<rbc>`, `<rtc>`, `<rb>`, `<rt>`, `<rp>`). Pored toga, uklonjen je `lang` atribut (u korist `xml:lang`) i `name` atribut iz `<a>` i `<map>` elemenata.

Iako je većinom kompatibilan sa *XHTML 1.0* i *HTML 4.01* specifikacijama, preporuka je da se *XHTML 1.1* ne prenosi obeležen kao *HTML* tip medija. Zbog ograničene podrške u pregledačima za `application/xhtml+xml` tip medija, ova specifikacija nije doživela širu upotrebu. Januara 2009. izdata je nova verzija specifikacije (*XHTML 1.1 Second Edition*), sa smanjenim ograničenjima i mogućnošću da se *XHTML 1.1* prenosi kao `text/html` tip medija.

3.2.4. *XHTML 2.0*

U periodu od avgusta 2002. do jula 2006. godine *W3C* je objavio osam nacrta *XHTML 2.0* specifikacije, verzije *XHTML* jezika koja je trebalo da mu omogući novi početak eliminišući kompatibilnost sa starijim verzijama. Ova odluka je u početku izazvala veliku polemiku među web programerima. Kasnije su određeni delovi jezika izuzeti iz specifikacije i razvijani kao posebni moduli, kako bi se olakšao prelazak iz *XHTML 1.x* u *XHTML 2.0*. Deveti nacrt specifikacije očekuje se u 2009. godini.

Neke od karakteristika *XHTML 2.0* su:

- *HTML* forme će biti zamenjene X formama (*XForms*);
- *HTML* okviri će biti zamenjeni X okvirima (*XFrames*);
- Novi tip liste elemenata (`<nl>`) će biti dodat da odredi navigacionu listu;
- Svaki element će moći da se ponaša kao *hyperlink*;
- Svaki element će moći da podrži alternativne medije atributom `src`;
- Atribut `alt` elementa `<img>` će biti izuzet, a alternativni tekst će postojati u sadržaju samog elementa, slično elementu `<object>`;
- Biće dodat jedinstveni element <h> za ispisivanje naslova, a njihov nivo će zavisiti od dubine ugnežđavanja tog elementa. To će omogućiti neograničeno korišćenje naslova;
- Prezentacioni elementi `<i>`, `<b>` i `<tt>` biće izostavljeni, a jedini preostali prezentacioni elementi će biti `<sup>` i `<sub>` za superskript i subskript. Svi ostali tagovi biće semantički, dok će programeru biti dozvoljeno da kroz *CSS* kontroliše prezentaciju elemenata.

3.3. Validacija *XHTML* dokumenata

XHTML dokument koji podleže *XHTML* specifikaciji naziva se validnim dokumentom. Validnost obezbeđuje konzistentnost koda u dokumentima, što za uzvrat olakšava njihovo obrađivanje, ali ne uvek i identičnost prikaza u različitim pregledačima. Dokumenti se mogu proveriti putem *W3C Markup Validation* servisa. U praksi, gotovo svi alati za razvoj web prezentacija zasnivaju proveru koda na osnovu *W3C* standarda.

3.3.1. Osnovni element

Prvi element *XHTML* dokumenta mora biti `<HTML>` i mora sadržati atribut `xlmns` koji će ga povezati sa bibliotekom *XHTML* elemenata. Adresa odnosno *URI* (*Uniform Resource Identifier*) biblioteke je http://www.w3.org/1999/XHTML. Od verzije *1.1*, *XHTML* ima atribut `version` koji ukazuje na verziju *XHTML* jezika koja se koristi u dokumentu. Moguće je dodati i atribut `xml:lang` koji govori o prirodnom jeziku dokumenta.

```
# <html xmlns="http://www.w3.org/1999/xhtml" version="XHTML 1.1"
# xml:lang="en">
```

Primer 6: Osnovni element *XHTML* dokumenta

U *XHTML* verzijama *1.1* i *2.0* moguće je dodati atribut `schemaLocation`, koji povezuje dokument sa *XML* šemom.

```
# <html
# xmlns="http://www.w3.org/2002/06/xhtml2/" version="XHTML 2.0"
# xml:lang="en"
# xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
# xsi:schemaLocation="http://www.w3.org/1999/xhtml
# http://www.w3.org/MarkUp/SCHEMA/xhtml2.xsd"
# >
```

Primer 7: Upotreba *XML* šema

Atribut xmlns:xsi ukazuje da svaki element ili atribut sa prefiksom xsi: potpada pod biblioteku *XML* šeme, a ne *XHTML* biblioteku. Na ovaj način može se koristiti više različitih *XML* rečnika u okviru jednog dokumenta bez bojazni da će se neki nazivi podudarati.

3.3.2. *Doctype*

Da bi se izvršila validacija *XHTML* dokumenata, može se koristiti `DOCTYPE`. On ukazuje pregledaču na definiciju tipa dokumenta kojoj dokument odgovara. Deklaracija tipa dokumenta se postavlja ispred osnovnog elementa.

```
# <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
# "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Primer 8: *Doctype*

URL (*Uniform Resource Locator*) koji počinje sa `http://` i nalazi se u okviru deklaracije ima zadatak samo da upućuje na kopiju definicije tipa dokumenta koja se koristi, ukoliko validator ne može da pronađe definiciju na osnovu javnog identifikatora (`-//W3C//DTD XHTML 1.0 Transitional//EN`). Ovaj *URL* ne mora da bude specifična adresa data u primeru, naprotiv, programerima se preporučuje da, kad god je to moguće, koriste kopije *DTD* datoteka u lokalu.

3.3.3. *XML* deklaracija

Dekodiranje karaktera može se navesti na početku *XHTML* dokumenta koristeći *XML* deklaraciju, ukoliko dokument koristi `application/xhtml+xml` tip medija.

```
# <?xml version="1.0" encoding="UTF-8"?>
```

Primer 9: *XML* deklaracija

Ova deklaracija se može izostaviti zato što deklariše podrazumevano dekodiranje. Međutim, ako dokument koristi *XML 1.1* dekodiranje ili neki drugi kodni raspored, deklaraciju je neophodno navesti. Na primer, *Internet Explorer* (pre verzije *7*) pravi greške ukoliko naiđe na *XML* deklaraciju u dokumentu koji koristi `text/html` tip medija.

3.3.4. Greške u kodiranju

XHTML je striktnija verzija *HTML* jezika, tako da se moraju poštovati sva pravila prilikom kreiranja dokumenata. Svi tagovi se moraju zatvarati, imena pisati malim slovima, a vrednosti atributa pod apostrofima ili navodnicima.

Neke od najčešćih grešaka su:

- Nezatvaranje praznih elemenata (elementi bez završnog taga u *HTML*)
  - nepravilno: `<br>`
  - pravilno: `<br />`;
- Nezatvaranje ostalih elemenata (elementi koji imaju završni tag)
  - nepravilno: `<p>Paragraf.<p>Drugi paragraf.`
  - pravilno: `<p>Paragraf.</p><p>Drugi paragraf.</p>`;
- Nepravilno ugneždavanje elemenata (važi i za *HTML*)
  - nepravilno: `<em><strong>Tekst.</em></strong>`
  - pravilno: `<em><strong>Tekst.</strong></em>`;
- Nekorišćenje navodnika oko vrednosti atributa
  - nepravilno: `<td rowspan=3>`
  - pravilno: `<td rowspan="3">` ili `<td rowspan='3'>`;
- Korišćenje posebnih karaktera izvan objekata (važi i za *HTML*)
  - nepravilno: `<title>Hrana & piće</title>`
  - pravilno: `<title>Hrana & piće</title>`;
- Elementi i atributi su osetljivi na mala i velika slova
  - nepravilno: `<BODY><P ID="ONE">Tekst.</P></BODY>`
  - pravilno: `<body><p id="ONE">Tekst.</p></body>`;
- Korišćenje minimizacije atributa
  - nepravilno: `<textarea readonly>AB</textarea>`
  - pravilno: `<textarea readonly="readonly">AB</textarea>`.

```
# <?xml version="1.0" encoding="UTF-8"?>
# <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
# "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
# <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
# <head>
# <title>XHTML 1.0 Example</title>
# <script type="application/javascript">
# <![CDATA[
# function loadpdf() {
#    document.getElementById("pdf-object").src=
#    "http://www.w3.org/TR/xhtml1/xhtml1.pdf";
# }
# ]]>
# </script>
# </head>
# <body onload="loadpdf()">
# <p>This is an example of an
# <abbr title="Extensible HyperText Markup Language">XHTML</abbr>
# 1.0 Strict document.<br/>
# <img id="validation-icon"
#    src="http://www.w3.org/Icons/valid-xhtml10"
#    alt="Valid XHTML 1.0 Strict"/><br/>
# <object id="pdf-object"
#    type="application/pdf"
#    data="http://www.w3.org/TR/xhtml1/xhtml1.pdf"
#    width="100%"
#    height="500"/>
# </p>
# </body>
# </html>
```

Primer 10: Izgled pravilno napisanog *XHTML* dokumenta

4. *CSS*

Kaskadne liste stilova ili *CSS* (*Cascading Style Sheets*) predstavljaju jezik koji se koristi za opis prezentacije dokumenata pisanih *markup* jezikom. Njegova najuobičajenija primena je za stilizovanje web prezentacija pisanih u *HTML* i *XHTML* jezicima, ali se može primeniti i za bilo koji *XML* dokument.

CSS je prvenstveno dizajniran da omogući razdvajanje sadržaja dokumenta (napisanog *markup* jezikom) od prezentacije dokumenta (boja, font, izgled). Ovo razdvajanje poboljšava pristup sadržaju, omogućava više fleksibilnosti i kontrole u specifikaciji karakteristika prezentacije, omogućava mnogobrojnim dokumentima da koriste iste formate i smanje kompleksnost i ponavljanje u strukturnom sadržaju. Takođe, *CSS* dozvoljava istim dokumentima da budu prezentovani na različit način, prilagođen ekranu, štampi, govoru ili Brajevo-baziranim uređajima.

Kako se *HTML* razvijao, tako je obuhvatao sve više stilskih mogućnosti da bi zadovoljio zahteve programera. Ovo je dizajnerima omogućilo više kontrole nad izgledom dokumenta, ali po cenu *HTML* jezika koji je postao složeniji za pisanje i održavanje.

W3C je izabrao dva od devet predloga kao osnovu za *CSS*: *CHSS* (*Cascading HTML Style Sheet*) i *SSP* (*Stream-based Style Sheet Proposal*). *CHSS* je predložio Hakon Vijum Li (Lie) oktobra 1994. godine i to je jezik najsličniji današnjem *CSS* jeziku. Bert Bos (Bos) radio je na pregledaču *Argo* koji je koristio sopstveni jezik liste stilova, *SSP*. Li i Bos radili su zajedno na razvoju *CSS* standarda.

Za razliku od dotadašnjih jezika listi stilova, *CSS* je dozvolio da prezentacija dokumenta bude pod uticajem više stilova. Jedan stil mogao je da nasledi ili „kaskadira“ iz drugog stila. U ovim kaskadama postojala je šema prioriteta koja je služila da razgraniči koja pravila se primenjuju, ukoliko postoji više pravila za prikaz određenog elementa u dokumentu.

Li je predlog prvi put predstavio 1994. godine, a drugi put zajedno sa Bosom, 1995. godine. Ubrzo nakon toga, *W3C* preuzeo je razvoj i objavio *CSS 1* krajem 1996. godine. *CSS 2* je dobio status *W3C* preporuke u maju 1998. godine. Trenutno je u fazi razvoja treća verzija *CSS* standarda.

4.1. Upotreba

Pre pojave *CSS* standarda skoro svi prezentacioni atributi sadržani su, zajedno sa *markup*-om, u *HTML* dokumentu. Fontovi, veličine, boje, pozadine, poravnanja i mnogi drugi atributi morali su biti jasno opisani u samom dokumentu, često i u više navrata. *CSS* omogućava autorima da pomere deo ovih informacija u zasebne liste stilova, čime *HTML markup* postaje jednostavniji i lakši za održavanje. Naslovi (<h1>), podnaslovi (<h2>), paragrafi (<p>) i drugi elementi strukturno su definisani *HTML* jezikom, a atributi kao što su izbor fonta, veličina i boja fonta ili boja pozadine predstavljaju prezentacione atribute. *CSS* razdvaja strukturu dokumenta od njegove prezentacije. Takođe, omogućava da se na potpuno različite načine prezentuje dokument, u zavisnosti da li je u pitanju štampa ili prikaz na ekranu. *W3C* trenutno gleda na prednosti upotrebe *CSS* standarda za formatiranje prezentacije dokumenta kao na superiorniju metodu od bilo koje druge. Formatiranje unutar samog *markup*-a u potpunosti je potisnuto.

4.2. Izvori

CSS stilovi mogu se pozvati iz različitih izvora. Podaci o stilovima mogu biti priključeni dokumentu u vidu posebne datoteke ili upisani unutar samog *HTML* dokumenta. Više različitih listi stilova može biti povezano sa istim dokumentom. Različiti stilovi mogu se primeniti na isti dokument u zavisnosti od tipa izlaznog uređaja (npr. verzija za prikaz na ekranu može biti drastično drugačija od verzije za štampu).

Ukoliko se koristi više izvora stilova, postoji definisana lista prioriteta po kojima se stilovi primenjuju (od najvišeg ka najnižem):

- Stilovi autora (lista stilova definisana od strane autora dokumenta) u formi

    - *inline* stilova (stilovi definisani unutar *HTML* dokumenta, pojedinačno za svaki element, upotrebom atributa `style`)
    - ugrađenih stilova (blokovi *CSS* podataka unutar *HTML* dokumenta, ali van *markup*-a) ili
    - spoljnih listi stilova (posebne *CSS* datoteke pozvane iz *HTML* dokumenta),
- Stilovi korisnika (lokalna *CSS* datoteka definisana od strane korisnika),
- Stilovi pregledača (predefinisana lista stilova nekog pregledača).

4.3. Sintaksa

CSS ima jednostavnu sintaksu koja koristi veliki broj engleskih ključnih reči da definiše imena različitih karakteristika stila. Svaka lista stilova se sastoji od spiska pravila, a svako pravilo, ili skup pravila, od selektora i bloka deklaracije. Blok deklaracije sadrži niz izjava odvojenih tačka-zarezom u zagradi. Izjave se sastoje iz osobine i vrednosti koje su razdvojene dvotačkom.

```
# h1 { color: white;}
```

Primer 11: *CSS* pravilo

Selektori se koriste da definišu na koji element *markup*-a se primenjuje stil. Mogu se odnositi na sve elemente određenog tipa ili samo na one elemente koji odgovaraju određenom atributu.

Pseudo-klase su još jedan oblik pravila koja se koriste da identifikuju *markup* elemente i, u nekim slučajevima, određene akcije korisnika za koje se dati blok deklaracije primenjuje. Najčešći primer je `:hover`, pseudo-klasa koja primenjuje stil samo kada korisnik ukaže na element, obično pozicioniranjem kursora miša iznad njega. Neke od pseudo-klasa i pseudo-elemenata su `:first-line`, `:visited` i `:before`. Posebna pseudo-klasa je `:lang(c)`.

Pseudo-klasa bira ceo element (npr. `:link` ili `:visited`), dok se pseudo-element može sastojati od delimičnog elementa (npr. `:first-line` ili `:first-letter`).

4.4. Kompatibilnost

Imajući u vidu da svi pregledači ne pružaju identičnu podršku *CSS* kodovima, koristi se tehnika kodiranja poznata kao *CSS filter* za prikazivanje ili skrivanje određenih delova koda u različitim pregledačima. Upotrebom *CSS* filtera, neki dizajneri su otišli toliko daleko da isporučuju kompletno drugačiji *CSS* u zavisnosti od korišćenog pregledača, kako bi osigurali pravilan prikaz dizajna. Zbog prvih verzija Internet pregledača koji su bili potpuno nesposobni da prikažu *CSS*, ili su to radili na jako loš način, dizajneri danas često rutinski koriste *CSS* filtere da u potpunosti spreče takve pregledače da pristupe *CSS* kodu. *Internet Explorer* je od verzije *3.0* počeo da pruža podršku za *CSS* i svakom sledećom verzijom ona se povećavala.

CSS 1 specifikacija završena je 1996. godine, kada je i *Internet Explorer 3.0* objavljen, uvodeći ograničenu podršku za *CSS*, više od tri godine pre nego što je bilo koji pregledač dostigao potpunu implementaciju specifikacije. *Internet Explorer 5.0* za *Macintosh* (objavljen 2000. godine) bio je prvi pregledač sa punom (više od 99%) podrškom za *CSS 1*. Ostali pregledači su usledili ubrzo nakon toga, od kojih su neki nudili i podršku za delove *CSS 2* specifikacije. Čak i uz ozbiljniju ponudu pregledača sa implementiranim *CSS*-om, i dalje je postojao problem visokog nivoa nedoslednosti i grešaka u tim implementacijama, što je u mnogome otežavalo dizajnerima da postignu konzistentan prikaz dokumenata na različitim platformama. To je bila glavna prepreka za usvajanje *CSS*-a.

Problemi sa nejednakom implementacijom i postojanjem štamparskih grešaka u originalnoj specifikaciji naveli su *W3C* da napravi reviziju *CSS 2* standarda, *CSS 2.1*. Neke karakteristike *CSS 2* standarda koje nisu uspešno implementirane u pregledače su isključene, a u nekim slučajevima i promenjene da bi ispratile postojeću implementaciju. *CSS 2.1* postao je kandidat za preporuku 25. februara 2004. godine, ali je povučen u status radnog stanja 13. juna 2005. godine, da bi ponovo bio vraćen na status kandidata za preporuku 19. jula 2007. godine. Od avgusta 2009. godine, *Internet Explorer* u verziji *8* i *Firefox* u verzijama *2* i *3* nude zadovoljavajuću implementaciju *CSS 2.1* standarda.

4.5. Prednosti i mane

Neke od prednosti upotrebe *CSS* standarda su:

- Fleksibilnost

  Kombinovanjem *CSS*-a sa funkcijama *CMS*-a (*Content Management System*), znatna količina fleksibilnosti može biti programirana u forme za predaju sadržaja. To omogućava da osoba koja prilaže sadržaj ne mora da bude upoznata sa *CSS* i *HTML* jezicima, već da na jednostavan način može da izabere da li, na primer, želi sadržaj formatiran u dve ili tri kolone, da li sadržaj ima propratnu sliku, prikačenu datoteku, itd. Te informacije se dalje prosleđuju u *CMS* gde logika samog programa za upravljanje sadržajem određuje koji će se predefinisani stilovi primeniti u formatiranju dokumenta u zavisnosti od klase i identifikacije koju poseduju *HTML* elementi. Ova mogućnost dobija na značaju u slučaju velikih sajtova kod kojih se sadržaji često dodaju;

- Razdvajanje sadržaja od prezentacije

  CSS omogućava formatiranje sadržaja na različite načine na osnovu nominalnih parametara. Neki od nominalnih parametara su eksplicitne preference korisnika, različiti Internet pregledači, tip uređaja koji se koristi za pregledanje sadržaja (npr. računar ili mobilni telefon), geografski položaj korisnika ili trenutno vreme;

- Rasprostranjena konzistentnost prezentacije

  Kada se upotrebljava na pravi način, u smislu pravilnog korišćenja nasleđivanja i „kaskada“, jedna lista stilova može se primeniti za stilizovanje elemenata širom sajta. Na taj način se, u slučaju želje za promenom stila sajta, kompletan rad svodi na promenu nekoliko pravila u listi stilova. Pre pojave *CSS*-a, ove promene su zahtevale više rada, vremena i novca;

- Propusna moć

  Liste stilova se obično čuvaju u kešu pregledača, tako da se mogu koristiti za više dokumenata bez ponovnog učitavanja. Na ovaj način se smanjuje preuzimanje podataka sa mreže i povećava brzina prenosa podataka;

- Reformatiranje dokumenata

  Jednostavnom promenom u jednoj liniji *HTML* koda, različite liste stilova mogu se primeniti na istom dokumentu. To pruža mogućnost da se prezentacija dokumenta prilagodi različitim pregledačima i/ili uređajima. Uređaji koji ne mogu da razumeju *CSS* će i dalje prikazati sadržaj.

Neki zapaženi nedostaci prilikom upotrebe *CSS* standarda su:

- Neusaglašena podrška u različitim pregledačima

  Kao rezultat pogrešne ili nepotpune implementacije *CSS* standarda, različiti pregledači će različito prezentovati dokumente. Na primer, ranije verzije *Internet Explorer*-a su implementirale određene *CSS 2* osobine kao sopstvene, na nekompatibilan način. Tako su izostavljene `width`, `height` i `float` osobine. Da bi se dobio konzistentan prikaz u najpopularnijim pregledačima često se moraju upotrebiti određene *CSS* „olakšice“ (*CSS hacks*);
- Ograničena vertikalna kontrola

  Dok je horizontalni položaj elemenata lako kontrolisati, postići vertikalno centriranje ili postaviti element na dno strane često je vrlo teško, ili čak nemoguće. Obično je potrebna upotreba komplikovanih i neintuitivnih pravila. Ovo je moguće izvesti i na lak način, ali to zahteva upotrebu pravila koja nisu široko podržana;
- Odsustvo izraza

  Trenutno ne postoji mogućnost određivanja vrednosti atributa korišćenjem jednostavnih izraza (npr. `margin-left: 10% - 3em + 4px;`). Ovo je korisno u raznim slučajevima, kao što je izračunavanje širine kolona ograničene zbirom širina svih kolona;
- Nedostatak ortogonalnosti

  Više atributa često radi istu stvar. Na primer, `position`, `display` i `float` određuju model pozicioniranja i uglavnom se ne mogu kombinovati smisleno. Element `display: table-cell` ne može biti plutajući ili mu se ne može dodeliti `position: relative`, i element `float: left` ne bi trebalo da reaguje na promene

elementa `display`. Osim toga, neki atributi nisu definisani na fleksibilan način da bi se izbeglo stvaranje novih atributa. Na primer, trebalo bi koristiti atribut `border-spacing` na elementu tabele umesto atributa `margin-*` na elementu ćelije tabele, zato što prema *CSS* specifikaciji unutrašnji elementi tabele nemaju margine;

- Kontrola oblika elemenata

  CSS trenutno podržava samo pravougaone oblike. Zaobljeni uglovi ili drugi oblici bi mogli da zahtevaju nesemantički *markup*;

- Nedostatak deklaracije kolona

  Iako su moguće, šeme sa više kolona mogu biti složene za implementaciju. Sa trenutnom verzijom *CSS*-a, ovaj proces se često obavlja upotrebom plutajućih elemenata koji se često različito prikazuju u različitim pregledačima i na različitim ekranima;

- Slaba kontrola prikaza za fleksibilne prikaze

  Dok novi dodaci u *CSS 3* standardu pružaju jači i robusniji set atributa za definisanje razmeštaja elemenata, *CSS* je i dalje jako ukorenjen kao stilski jezik, a ne jezik razmeštaja.

5. *PHP*

PHP hipertekst preprocesor ili *PHP* (*PHP: HyperText Preprocessor*) je široko rasprostranjen, višenamenski skript jezik, prvenstveno dizajniran za kodiranje dinamičkih web stranica.

Rasmus Lerdorf (Lerdorf), danski programer, započeo je rad 1994. godine na projektu „Alati za ličnu prezentaciju“ ili „PHP Tools“ (*Personal Home Page Tools*). On je kreirao prve *PHP* alate (skup *CGI* binarnih datoteka napisanih u *C* programskom jeziku) kako bi zamenio skup *Perl* skripta koje je koristio za održavanje svoje prezentacije. Alati su korišćeni da izvrše zadatke kao što su prikazivanje njegovog rezimea i beleženje broja poseta njegovoj web prezentaciji. On je iskombinovao ove alate sa sopstvenim *Form Interpretator*-om i stvorio *PHP/FI*, koji je imao veću funkcionalnost. *PHP/FI* je uključivao veću implementaciju *C* programskog jezika i mogao je da komunicira sa bazom podataka, što je omogućavalo kreiranje jednostavnih dinamičkih web aplikacija. 8. juna 1995. godine, Lerdorf objavljuje *PHP* u javnosti kako bi ubrzao pronalaženje grešaka i unapredio kod. Ovo izdanje je nazvano *PHP 2* i već tada je sadržalo osnovnu funkcionalnost današnjeg *PHP*-a (varijable slične kao u *Perl*-u, rukovanje formama i mogućnost ugrađivanja u *HTML*). Sintaksa je bila slična *Perl*-u ali sa mnogo ograničenja, jednostavnija i manje konzistentna.

Dva izraelska programera, Zev Suraski (Suraski) i Endi Gutmans (Gutmans) ponovo su napisali tumač 1997. godine i postavili osnovu za *PHP 3*, menjajući naziv jezika u *PHP: HyperText Preprocessor*. *PHP/FI 2* ovaj tim izdaje novembra 1997. godine, nakon nekoliko meseci testiranja. Ubrzo zatim, počinje testiranje treće verzije i *PHP 3* se zvanično pojavljuje juna 1998. godine. Suraski i Gutmans tada počinju ponovo da pišu jezgro i stvaraju *Zend Engine* 1999. godine. Takođe osnivaju i *Zend Technologies* u Izraelu.

Maja 2000. godine izdaju *PHP 4* kojeg pokreće *Zend Engine 1.0*. Do avgusta 2008. godine on se razvio do verzije *4.4.9*, nakon koje se razvoj *PHP 4* prekida i

ukida se pružanje podrške. *PHP 5* kojeg pokreće *Zend Engine II* izdat je jula 2004. godine. On donosi poboljšanu podršku za objektno orijentisano programiranje, *PHP Data Objects* ekstenzije (za lakše i stabilnije povezivanje sa bazom podataka) i brojna poboljšanja performansi.

U drugoj polovini 2008. godine *PHP 5* postaje jedina stabilna verzija pod razvojem. Poslednja objavljena stabilna verzija je *5.2.11*. U međuvremenu, izdata je i *PHP 5.3* stabilna verzija koja ima ulogu u tranziciji na *PHP 6* koji je trenutno u razvoju i u kojem će pojedine funkcije, kao što su `register_globals`, *magic quotes* i *safe mode*, biti izbačene. Glavnu implementaciju *PHP*-a sada proizvodi *The PHP Group* i služi kao *de facto* standard, budući da ne postoji formalna specifikacija.

PHP skript jezik se može postaviti na većini servera i na skoro svakom operativnom sistemu. Trenutno je instaliran na više od milion servera i opslužuje više od dvadeset miliona web prezentacija. *PHP* je besplatno softversko izdanje pod *PHP* licencom, ali zbog ograničenja u vezi korišćenja termina *PHP*, nije kompatibilan sa *GNU General Public License*-om.

5.1. Upotreba

PHP se prvenstveno ponaša kao filter, uzimajući ulazne podatke iz datoteke ili niza koji sadrži tekst i/ili *PHP* instrukcije i vraćajući drugi niz podataka, najčešće u obliku *HTML*-a. Od verzije *4*, *PHP* tumač kompajlira ulazne podatke kako bi dobio bajt-kod koji će se procesirati u *Zend Engine*-u, što daje mnogo bolje performanse u odnosu na raniju upotrebu interpretera.

Originalno zamišljen da kreira dinamičke web prezentacije, glavni fokus *PHP*-a sada je na kodiranju na serverskoj strani. On ima velike sličnosti sa drugim programskim jezicima koji pružaju dinamički sadržaj od servera ka klijentu, kao što su *Microsoft ASP* (*Active Server Pages*), *Sun Microsystems JSP* (*JavaServer Pages*) i *mod_perl*. *PHP* je privukao razvoj mnogih sistema za kreiranje programskih blokova i strukture dizajna radi promocije *RAD* (*Rapid Application*

Development) tehnologije. Neki od njih su *CakePHP*, *Symfony*, *CodeIgniter* i *ZendFramework*.

LAMP i *WAMP* arhitekture su postale veoma popularne kao način postavljanja web aplikacija. P se obično odnosi na *PHP*, L i W na *Linux* i *Windows*, A na *Apache* i M na *MySQL*. U nekim slučajevima, P može da predstavlja i *Python* ili *Perl*.

Od aprila 2007. godine, preko dvadeset miliona internet domena je postavljeno na serverima sa instaliranim *PHP*-om. Veliki broj prezentacija je napisan delom u ovom jeziku: *Facebook*, *Wikipedia*, *Yahoo*, *Digg*, *Joomla*, *WordPress*, *YouTube*, itd.

5.2. Sintaksa

PHP tumači samo ono što se nalazi unutar njegovih graničnika, a sve što je van prosleđuje direktno na izlaz. Najčešći graničnici su `<?php` i `?>`, koji predstavljaju početni i završni graničnik i odnose se jedan na drugi. Moguće je koristiti i kombinaciju `<script language="php">` i `</script>`. Takođe, postoje i skraćene verzije početnih graničnika `<?` i `<?=` (druga se koristi za `echo` funkciju), kao i graničnici u *ASP* stilu, `<%` i `%>`. Upotreba ovih graničnika nije preporučljiva zato što je u konfiguraciji *PHP*-a moguće isključiti podršku za njih. Jedina namena graničnika je da odvoji *PHP* od ostatka koda, najčešće *HTML*-a.

Jedna od karakteristika *PHP*-a je implicitna deklaracija promenljivih. Promenljive imaju prefiks `$` i ne treba da budu unapred definisane. Imena promenljivih su, za razliku od imena funkcija i klasa, osetljiva na mala i velika slova. Stringovi pod navodnicima (`""`) i *heredoc* dopuštaju da se vrednost promenljive ugnezdi u string. *PHP* tretira nove redove kao razmake u maniru jezika slobodne forme (osim kada se nalaze pod navodnicima), a izjave se završavaju znakom `;`. U *PHP*-u postoji tri tipa komentara: `/* */` za obeležavanje bloka komentara, a `//` i `#` za komentare koji su u jednoj liniji. Jedna od najčešće

korišćenih funkcija za ispis teksta je `echo`. Pored nje, postoji i `print` funkcija, ali je ona nešto sporija pošto vraća status da li je bila uspešna ili ne.

U smislu ključnih reči i sintakse jezika, *PHP* je sličan mnogim jezicima visokog nivoa koji prate *C* sintaksu. Uslovi `if`, `for` i `while` petlje i `return` funkcije su slične u sintaksi sa jezicima kao što su *C*, *C++*, *Java* i *Perl*.

```
# <?php echo "Hello World!\n"; ?>
```

Primer 12: Upotreba `echo` funkcije

Umesto korišćenja `<?php` graničnika i `echo` funkcije, moguća je upotreba skraćenice `<?=` koja implicitno ispisuje podatke.

```
# <?=$tekst; ?>
```

Primer 13: Ispisivanje sadržaja promenljive `$tekst`

5.3. Tipovi podataka

PHP čuva cele brojeve u opsegu zavisnom od platforme. Tipično, ovo je opseg 32-bitnih celih brojeva sa predznakom. Celi brojevi bez predznaka se, u određenim situacijama, konvertuju u brojeve sa predznakom. Ovakvo ponašanje je drugačije u odnosu na ostale programske jezike. Celobrojne promenljive mogu biti određene upotrebom decimalnih (pozitivnih i negativnih), oktalnih ili heksadecimalnih oznaka. Brojevi sa plutajućim zarezom se, takođe, beleže u opsegu zavisnom od platforme. Oni mogu biti zadati korišćenjem decimalne tačke ili dva oblika naučnog obeležavanja.

PHP poseduje urođen logički tip (*Boolean*) koji je sličan logičkom tipu u *Java* i *C++* jezicima. Korišćenjem pravila konverzije logičkih tipova, vrednosti različite od nule se tumače kao istinite (*true*), a nule kao lažne (*false*). Nulti tipovi podataka predstavljaju promenljive koje nemaju vrednost. Jedina vrednost u ovom tipu podataka je *NULL*. Promenljive tipa izvora (*resources*) imaju vrednost koja ukazuje na lokaciju nekog eksternog izvora podataka (datoteka, slika, baza podataka, itd.).

Nizovi mogu sadržati elemente bilo kog tipa koji *PHP* podržava, uključujući izvore (*resources*), objekte (*objects*), pa čak i druge nizove (*arrays*). Poredak se čuva u listama vrednosti i u *hash* datotekama. *PHP* podržava i stringove koji se mogu koristiti sa apostrofima, navodnicima ili *heredoc* sintaksom.

5.4. Funkcije

PHP ima stotine osnovnih funkcija i na hiljade kroz razne ekstenzije. Ove funkcije su dobro dokumentovane na web prezentaciji *PHP*-a, ali bez obzira na to, ugrađena biblioteka poseduje širok spektar nazivnih konvencija i puna je nekonzistentnosti.

Funkcije nisu funkcije prve klase i mogu biti pozvane samo svojim imenom, bilo direktno, bilo preko promenljive koja sadrži ime funkcije. Funkcije definisane od strane korisnika mogu biti napravljene u bilo kom trenutku bez kreiranja prototipa. Funkcije mogu biti određene unutar blokova koda, čime se dozvoljava da se odluka o tome da li će funkcija biti definisana ili ne, donese prilikom izvršavanja koda. Pozivi funkcija moraju koristiti zagrade sa izuzetkom funkcija bez argumenata koje se pozivaju new operatorom, gde su zagrade opcione.

```
# <?php
# function hello()
# {
#  echo "Hello World!\n";
# }
#
# hello();
# ?>
```

Primer 14: Definisanje funkcije

Pozivi funkcija mogu biti napravljeni i upotrebom promenljivih, gde vrednost promenljive sadrži ime funkcije koju poziva.

```
# <?php
# function hello()
# {
#  return 'Hello';
# }
# function world()
# {
#  return "World!\n";
# }
#
# $fn1 = 'hello';
# $fn2 = 'world';
#
# echo $fn1() . ' ' . $fn2();
# ?>
```

Primer 15: Pozivanje funkcija upotrebom promenljivih

5.5. Objekti

Osnovne funkcije objektno orijentisanog programiranja dodate su već u trećoj verziji *PHP* jezika, a poboljšane u *PHP 4*. Rukovanje objektima iznova je napisano u verziji *PHP 5*, čime je proširen skup mogućnosti i poboljšane su performanse. U ranijim verzijama, objektima se rukovalo kao sa primitivnim tipovima podataka. Nedostatak ove metode bio je taj da se ceo objekat kopirao kada je promenljiva dodeljivana ili prosleđivana kao parametar metodi. Prema novom pristupu, objekti se referenciraju ukazivačima, a ne vrednostima.

PHP 5 uvodi privatne i zaštićene promenljive i metode, zajedno sa apstraktnim klasama i privatnim klasama kao i apstraktnim metodama i privatnim metodama. On takođe uvodi i standardni način deklarisanja konstruktora i destruktora kao i model upravljanja greškama (*exception handling*), slično drugim objektno orijentisanim jezicima kao što je *C*++.

5.6. Optimizacija brzine

Kao i u svakom interpretiranom jeziku, *PHP* skripti se čuvaju kao ljudsko-čitljivi izvorni kodovi koji se u hodu kompajliraju korišćenjem *PHP engine*-a. Ovo predstavlja mogući bezbednosni rizik koji kompajlirani jezici (kao što je *C*, jezik u kome su *PHP* i njegove ekstenzije napisane) nemaju zato što su sačuvani u binarnom formatu. Da bi se smanjilo vreme izvršenja i eliminisala potreba za

kompajliranjem *PHP* izvornog koda svaki put kada se pristupi web stranici, *PHP* skripti se takođe mogu čuvati u binarnom formatu upotrebom *PHP* kompajlera.

Optimizeri koda imaju za cilj da smanje vreme izvršenja kompajliranog koda smanjivanjem njegove veličine i drugim izmenama. Priroda *PHP* kompajlera je takva da često postoji mogućnost za optimizaciju koda, a jedan od optimizera može biti *PHP* ekstenzija *eAccelerator*.

Drugi pristup smanjivanju opterećenja *PHP* servera je upotreba *Opcode cache*-a. On funkcioniše tako što kešira kompajlirane forme *PHP* skripta (*opcodes*) u deljenoj memoriji i time izbegava ponovno tumačenje i kompajliranje koda svaki put kada se skript pokrene.

6. *(My)SQL*

6.1. *SQL*

Struktuirani jezik upita ili *SQL* (*Structured Query Language*) je programski jezik namenjen za upravljanje podacima u sistemima za upravljanje relacionim bazama podataka, *RDBMS* (*Relational Database Managment System*). Tu spada iščitavanje i upisivanje podataka, pravljenje i modifikovanje šema baze podataka i kontrola pristupa objektima baze podataka. Baziran je na principima relacione algebre. *SQL* je bio jedan od prvih jezika za relacioni model Edgara F. Koda (Codd), opisan 1970. godine u njegovom radu „A Relational Model of Data for Large Shared Data Banks“, i postao je jedan od najrasprostranjenijih jezika za rad sa relacionim bazama podataka.

SQL je razvijen u *IBM*-u od strane Donalda D. Čemberlina (Chamberlin) i Rejmonda F. Bojsija (Boyce) ranih 70-ih godina dvadesetog veka. Ova verzija, prvobitno nazvana *SEQUEL* (*Structured English Query Language*), bila je namenjena *IBM*-ovom sistemu za upravljanje relacionim bazama podataka, nazvanom *System R*. Kasnije, zbog autorskih prava, jezik menja ime u *SQL*.

Prvi sistemi za upravljanje relacionim bazama podataka bili su *RDMS*, razvijen ranih 70-ih godina u *Massachusetts Institute of Technology*, i *Ingres*, razvijen 1974. godine na *University of California, Berkeley*. *Ingres* je koristio *QUEL* koji je kasnije na tržištu potisnut od strane *SQL*-a.

Kasnih 70-ih godina, kompanija *Relational Software, Inc.* (današnji *Oracle Corporation*), uvidela je potencijal koncepata opisanih od strane Koda, Čemberlina i Bojsija i razvila sopstveni *RDBMS* zasnovan na *SQL*-u, sa ciljem da ga proda Američkoj mornarici, *CIA*-i i drugim američkim vladinim agencijama. Leta 1979. godine kompanija predstavlja prvu komercijalno dostupnu implementaciju *SQL*-a, *Oracle V2*, samo nekoliko nedelja pre *IBM*-a.

Nakon završenog testiranja *SQL*-a, *IBM* počinje da razvija prve komercijalne proizvode bazirane na prototipu *System R*. Tako nastaju *System/38* (1979. godine), *SQL/DS* (1981. godine) i *DB2* (1983. godine).

SQL je 1986. godine standardizovan po *ANSI*, a 1987. i po *ISO* kriterijumima. Sam standard je sačinjen iz više delova, od kojih su neki: *SQL/Foundation*, *SQL/CLI*, *SQL/PSM*, *SQL/MED*, *SQL/OLB*, *SQL/Schemata*, *SQL/JRT* i *SQL/XML*. Zbog obimnosti standarda svaki proizvođač *DBMS*-a na sopstveni način implementira *SQL*.

Česte kritike upućene na račun *SQL*-a uključuju nedostatak kompatibilnosti između rešenja različitih proizvođača, neprikladno rukovanje podacima koji nedostaju i nepotrebno kompleksnu i povremeno nejasnu gramatiku i semantiku jezika.

6.1.1. Elementi jezika

SQL jezik se sastoji iz:

- Klauzula (*Clauses*)

  To su, u nekim slučajevima opcione, konstitutivne komponente izjava i upita;
- Izraza (*Expressions*)

  Oni mogu proizvoditi ili skalarne vrednosti ili tabele koje se sastoje od kolona i redova podataka;
- Iskaza (*Predicates*)

  Oni određuju uslove koji mogu biti ocenjivani trovrednosnom logikom *SQL*-a i na taj način ograničavaju efekte izraza i upita ili menjaju tok izvršavanja programa;
- Upita (*Queries*)

  Oni vraćaju podatke na osnovu prethodno zadatih kriterijuma;

- Izjava (*Statements*)

  One mogu imati stalan efekat na šeme i podatke ili kontrolisati transakcije, tok izvršavanja programa, konekcije, sesije ili dijagnostiku;
- Završetaka izjava (*Statement Terminators*)

  Svaka *SQL* izjava se završava tačka-zarezom. To nije obavezno na svakoj platformi, ali je standardni deo *SQL* gramatike;
- Beznačajnih razmaka (*Insignificant Whitespaces*)

  Oni su obično zanemareni u *SQL* izrazima i upitima, ali olakšavaju čitljivost *SQL* koda.

6.1.2. Upiti

Najčešća operacija u *SQL*-u je upit, koji se izvodi deklarativnom `SELECT` izjavom. Komanda `SELECT` vraća podatke iz jedne ili više tabela, odnosno izraza. Uobičajene `SELECT` izjave nemaju trajan efekat na bazu podataka. Neke nestandardne implementacije `SELECT` naredbe mogu imati trajan efekat, kao što je `SELECT INTO` u nekim bazama podataka.

Upiti omogućavaju korisniku da opiše željene podatke, a sistemu za upravljanje bazom podataka prepušta se odgovornost oko planiranja, optimizacije i izvršenja fizičkih operacija neophodnih da se ti podaci dostave.

Upit sadrži listu kolona koje treba da budu prikazane odmah nakon ključne reči `SELECT`. Zvezdica (`*`) se može upotrebiti da ukaže da u rezultat treba uključiti sve kolone date tabele. `SELECT` je najkompleksnija izjava u *SQL*-u sa opcionim ključnim rečima i klauzulama:

- `FROM`

  Ukazuje na tabelu iz koje treba pročitati podatke. Opciono sadrži i `JOIN` podklauzulu koja definiše pravila spajanja tabela;

- `WHERE`

  Definiše iskaz za poređenje, kojim se ograničavaju redovi koji se vraćaju kao rezultat upita. Ova klauzula eliminiše one redove kod kojih rezultat poređenja sa iskazom nije istinit (*true*);

- `GROUP BY`

  Ova klauzula se koristi da redove koji imaju zajedničke vrednosti grupiše u manje grupe redova. Često se koristi zajedno sa *SQL* funkcijama agregacije i za eliminisanje duplikatnih redova. Klauzula `WHERE` se uvek upotrebljava ispred ove klauzule;

- `HAVING`

  Ova klauzula uključuje iskaz na osnovu koga se filtriraju rezultati dobijeni `GROUP BY` klauzulom. Zato što ima efekat na rezultate `GROUP BY` klauzule, u iskazu `HAVING` klauzule mogu se koristiti funkcije agregacije;

- `ORDER BY`

  Ovom klauzulom se određuje kolona po kojoj će se sortirati dobijeni rezultati u rastućem ili opadajućem nizu (ASC/DSC). Bez ove klauzule neće biti definisan redosled redova rezultata.

```
# SELECT *
#          FROM televizor
#          WHERE cena > 199.00
#          ORDER BY tip;
```

Primer 16: `SELECT` upit

6.1.3. Trovrednosna logika

Ideja o nultom tipu podatka (*NULL*) implementirana je u SQL kako bi rukovala nedostajućim podacima u relacionom modelu. *NULL* vrednost (nepoznato) zajedno sa *TRUE* (istinito) i *FALSE* (lažno) vrednostima predstavlja osnovu trovrednosne logike ili *3VL* (*Three-Valued Logic*). *NULL* nema svoju vrednost, već samo ukazuje na nepostojanje informacije. Stoga, poređenja sa *NULL*-om uvek rezultiraju nepoznatim, kao trećom mogućom vrednošću.

6.1.4. Manipulacija podacima

Jezik za manipulaciju podacima ili *DML* (*Data Manipulation Language*) je podset *SQL*-a koji se koristi za upisivanje, ažuriranje i brisanje podataka. Njegove komande su:

- `INSERT`

  Koristi se za dodavanje redova u postojeću tabelu;

- `UPDATE`

  Koristi se za modifikaciju postojećih redova u tabeli;

- `DELETE`

  Koristi se za brisanje redova iz tabele;

- `TRUNCATE`

  Koristi se za brisanje svih podataka iz tabela;

- `MERGE`

  Koristi se za mešanje podataka iz različitih tabela. Predstavlja kombinaciju `INSERT` i `UPDATE` komandi.

```
# INSERT INTO televizor
#          (proizvodjac, tip, cena)
#      VALUES
#          ('Sony', 'LCD', 200);
```

Primer 17: Upotreba *DML* komandi

6.1.5. Kontrola transakcija

Transakcije, ako su dostupne, zaokružuju *DML* komande. One uključuju:

- `START TRANSACTION` (`BEGIN WORK` ili `BEGIN TRANSACTION`)

  Koristi se za obeležavanje početka transakcije u okviru baze podataka i izvršava se u celosti ili se uopšte ne izvršava;

- `COMMIT`

  Ova komanda uzrokuje da sve promene nastale prilikom transakcija u bazi podataka budu trajno sačuvane;

- `ROLLBACK`

  Ova komanda uzrokuje da se sve promene nastale od poslednje `COMMIT` ili `ROLLBACK` komande zanemare.

Kada se jednom komanda COMMIT izvrši, promene nastale transakcijom se ne mogu poništiti komandom ROLLBACK. Komande COMMIT i ROLLBACK završavaju tekuću transakciju i otključavaju podatke.

```
# START TRANSACTION;
#          UPDATE racun SET iznos=iznos-200
#          WHERE br_racuna=1234;
#          UPDATE racun SET iznos=iznos+400
#          WHERE br_racuna=5678;
#          IF ERRORS=0 COMMIT;
#          IF ERRORS<>0 ROLLBACK;
```

Primer 18: Upotreba komandi kontrole transakcija

6.1.6. Definicija podataka

Jezik definicije podataka ili *DDL* (*Data Definition Language*) upravlja strukturom tabela i indeksa. Njegove osnovne komande su:

- CREATE

  Ova komanda stvara objekat u bazi podataka (npr. tabelu);

- ALTER

  Ova komanda modifikuje strukturu postojećih objekata (npr. dodaje kolonu tabeli);

- RENAME

  Ova komanda menja nazive objekata;

- DROP

  Ova komanda briše objekte iz baze podataka, najčešće nepovratno.

```
# CREATE TABLE televizor
# (
#          id INT NOT NULL,
#          proizvodjac VARCHAR(12),
#          tip VARCHAR(3),
#          cena DOUBLE PRECISION,
#          PRIMARY KEY (id),
# );
```

Primer 19: Upotreba *DDL* komandi

6.1.7. Kontrola podataka

Jezik kontrole podataka ili DCL (Data Control Language) dodeljuje korisnicima ili grupama korisnika ovlašćenja za pristup i manipulaciju podacima. Njegove komande su:

- GRANT

  Ova komanda autorizuje jednog ili vise korisnika za određene operacije nad objektima;

- REVOKE

  Ova komanda onemogućava pristup operacijama nad objektima.

```
# GRANT SELECT, UPDATE
#          ON televizor
#          TO user1, user2;
```

Primer 20: Upotreba *DCL* komandi

6.1.8. Tipovi podataka

Svaka kolona u SQL tabeli deklariše tip podataka koji ona može sadržati. Podaci mogu biti:

- Stringovi karaktera
  - CHARACTER(n) ili CHAR(n)
  - CHARACTER VARYING(n) ili VARCHAR(n)
  - NATIONAL CHARACTER(n) ili NCHAR(n)
  - NATIONAL CHARACTER VARYING(n) ili NVARCHAR(n);
- Stringovi bitova
  - BIT (n)
  - BIT VARYING(n);
- Brojevi
  - INTEGER i SMALLINT
  - FLOAT, REAL i DOUBLE PRECISION
  - NUMERIC(precision, scale) ili DECIMAL(precision, scale);

- Datum i vreme
    - DATE
    - TIME
    - TIMESTAMP
    - INTERVAL.

6.2. *MySQL*

Moj struktuirani upitni jezik ili *MySQL* (*My Structured Query Language*) je sistem za upravljanje relacionim bazama podataka. Ovaj program se pokreće kao server koji omogućava višekorisnički pristup određenom broju baza podataka.

Izvorni kod *MySQL*-a je dostupan pod *GNU General Public License*-om, kao i raznim drugim sporazumima o vlasništvu. *MySQL* je u vlasništvu profitabilne švedske kompanije *MySQL AB*, filijale *Sun Microsystems*-a. 2009. godine *Oracle Corporation*, koji inače drži veći deo autorskih prava nad *MySQL* projektom, započinje preuzimanje kompanije *Sun Microsystems*.

Razvoj MySQL-a je započet 1994. godine od strane Majkla Videniusa (Widenius) i Dejvida Eksmarka (Axmark). Prvo izdanje je usledilo maja 1995. godine. Trenutno aktuelna verzija, izdata novembra 2008. godine, jeste *MySQL 5.1*, a u razvoju se nalazi verzija *6.0*.

6.2.1. Upotreba

MySQL se najčešće upotrebljava u projektima besplatnog softvera koji zahteva potpuno opremljene sisteme za upravljanje bazama podataka, kao što su *phpBB*, *WordPress* i drugi softveri zasnovani na *LAMP* arhitekturi. Takođe, nalazi upotrebu na velikim sajtovima, kao što su *Google*, *Facebook*, *Wikipedia*, itd. Njegova popularnost je u velikoj meri povezana sa popularnošću *PHP* jezika, sa kojim se *MySQL* najčešće kombinuje. Trenutno je instaliran na više od šest miliona sajtova.

6.2.2. Platforme i interfejsi

MySQL je pisan u *C* i *C++* jezicima i radi na mnogim platformama od kojih su neke: *Linux*, *FreeBSD*, *OpenBSD*, *NetBSD*, *Novell NetWare*, *Mac OS X*, *Solaris*, *Symbian*, *OS/2 Warp* i *Microsoft Windows*. Svi veći programski jezici sa specifičnim *API*-ima (*Application Programming Interface*) pružaju biblioteke za rad sa *MySQL* bazama. Osim toga, *ODBC* (*Open Database Connectivity*) interfejs pod imenom *MyODBC* omogućava i drugim programskim jezicima sa podrškom za *ODBC* interfejs da komuniciraju sa *MySQL* bazom (npr. *ASP* i *ColdFusion*).

Za administraciju *MySQL* baza podataka može se koristiti postojeći alat za administraciju iz komandne linije (komande `mysql` i `mysqladmin`). Takođe, sa *MySQL* sajta može se preuzeti paket *GUI* (*Graphical User Interface*) administrativnih alata: *MySQL Administrator*, *MySQL Query Browser* i *MySQL Migration Toolkit*.

U vezi sa pomenutim alatima koje je razvio *MySQL AB*, postoji i više komercijalnih i nekomercijalnih rešenja koja mogu da se integrišu u *MySQL*. Jedno od najpopularnijih besplatnih rešenja jeste *phpMyAdmin*, administratorski interfejs napravljen u *PHP* jeziku.

6.2.3. Karakteristike

Ono što *MySQL* odvaja od drugih sistema za upravljanje relacionim bazama podataka su sledeće karakteristike:

- posedovanje višestrukih sistema za smeštanje, pri čemu je moguć odabir najefikasnijeg sistema za određenu primenu, čak i u toku samog rada servera
    - urođeni sistemi (npr. *MyISAM*, *Falcon*, *CSV*)
    - sistemi razvijeni u saradnji sa partnerima (npr. *InnoDB*, *solidDB*, *NitroEDB*)
    - sistemi razvijeni u saradnji sa zajednicom (npr. *memcached*, *httpd*, *PBXT*)
    - sistemi prilagođeni specifičnim potrebama korisnika;

- posedovanje funkcije grupisanja izvršenja, pri čemu se prikupljaju višestruke transakcije iz višestrukih konekcija, sa ciljem povećanja izvršenja u sekundi.

Aprila 2009. godine *MySQL* je ponudio *MySQL 5.1* u dve varijante, *MySQL Community Server* (besplatna verzija) i *MySQL Enterprise Server* (komercijalna verzija namenjena velikim korisnicima). Obe varijante poseduju zajednički osnovni kod i sledeće osobine:

- širok podskup *ANSI SQL 99*, kao i ekstenzije;
- podrška radu na različitim platformama;
- procedure sačuvane u samoj bazi podataka;
- okidače;
- pokazivače;
- mogućnost ažuriranja pogleda;
- prava VARCHAR podrška;
- `INFORMATION_SCHEMA`;
- striktni režim;
- *X/Open XA* raspodeljena obrada transakcija ili *DTP* (*Distributed Transaction Processing*) i kao deo ovoga dvofazno izvršavanje upotrebom *Oracle InnoDB* sistema;
- nezavisni sistemi za smeštanje (*MyISAM* za brzinu čitanja, *InnoDB* za transakcije i referencijalni integritet, *MySQL Archive* za čuvanje arhivskih podataka na malom prostoru);
- transakcije sa *InnoDB*, *BDB* i *Cluster storage* sistemima
- *SSL* podrška;
- keširanje upita;
- podselekcije (ugnežđene selekcije);
- ugrađena biblioteka baze podataka;
- potpuno tekstualno indeksiranje i pretraživanje upotrebom *MyISAM* sistema;
- delimična *Unicode* podrška, itd.

7. *FLASH*

Adobe Flash (nekadašnji *Macromedia Flash*) je multimedijalna platforma prvobitno kreirana od strane *Macromedia*-e, a trenutno je razvija i distribuira *Adobe Systems*. Od predstavljanja 1996. godine *Flash* je postao jedan od najpopularniji načina za dodavanje animacije i interaktivnosti web prezentacijama. Najčešće se koristi za izradu animacija, reklama i različitih komponenti za web stranice (npr. za integrisanje video zapisa u web prezentacije).

Flash može da radi sa vektorskom i rasterskom grafikom, i podržava dvosmerni *streaming* audio i video zapisa. Nekoliko softverskih proizvoda, sistema i uređaja može da kreira i prikazuje *Flash* sadržaje, uključujući *Adobe Flash Player*, koji je besplatan i dostupan za većinu web pregledača, neke mobilne telefone i druge elektronske uređaje. *Adobe Flash Professional* program za kreiranje multimedijalnih sadržaja koristi se za pravljenje web aplikacija, igara i filmova, kao i sadržaja za mobilne elektronske uređaje.

Datoteke u *SWF* (*ShockWave Flash*) formatu poseduju sadržaj koji može biti objekat uključen u web stranicu, pušten kroz *Flash Player* ili pak zapakovan u *Projector* (samoizvršavajuću datoteku sa *Flash* sadržajem). *Flash* video sadržaji se nalaze u *FLV* (*Flash Video*) formatu koji može biti pušten ili kroz *SWF* datoteku ili putem nekog plejera koji ga podržava (npr. *VLC*, *QuickTime*, *Windows Media Player* sa dodatkom eksternog kodeka).

Flash aplikacija je bila originalna zamisao Džonatana Geja (Gay) koju je sa Čarlijem Džeksonom (Jackson) i Mišel Velš (Welsh) realizovao 1993. godine u vidu proizvoda *SmartSketch*. Razvojem Interneta, ovaj tim je uvideo potrebu za dodavanjem alata za animaciju svom programu za crtanje. Tako je nastao *FutureSplash Animator*, kojeg je krajem 1996. godine *Macromedia* otkupila i preimenovala u *Flash*. *Macromedia* je razvijala *Flash* sve do verzije *8*, kada je i zvanično preuzeta od strane *Adobe*-a. Danas je aktuelna verzija *10* (*Adobe Flash CS4 Professional*).

7.1. Programski jezik

Primarno fokusirane na animaciju, prve verzije *Flash*-a imale su ograničene mogućnosti za rad sa interaktivnim sadržajem, a samim tim i slabe opcije pisanja skripta.

Novije verzije uključuju *ActionScript*, implementaciju *ECMAScript* standarda, pa stoga ima sintaksu kao *JavaScript*, ali u drugačijem okruženju i sa drugačijim setom biblioteka klasa. *ActionScript* se koristi za kreiranje gotovo svih interaktivnih elemenata u *Flash*-u (dugmad, padajuće liste, polja za unos teksta, itd.). Trenutno aktuelna verzija je *ActionScript 3.0* koja je donela mnogo poboljšanja, veću fleksibilnost i objektno orijentisani pristup skriptovanju.

7.2. Karakteristike

U odnosu na druga rešenja na tržištu (*Java*, *Acrobat Reader*, *QuickTime*, *Windows Media Player*) instalacija *Flash Player*-a je manja, brže se preuzima i brže pokreće. Upoteba vektorske grafike u kombinaciji sa programskim kodom čini *Flash* datoteke manjim i samim tim manje zahtevnim za propusni opseg Internet veze u odnosu na bitmapiranu grafiku i video klipove. Za sadržaje koji koriste samo jedan tip medija (tekst, video ili audio), druga rešenja mogu pokazati bolje performanse. *Flash* podržava dva formata video zapisa (*On2 VP6* i *Sorenson Spark*), *MP3*-bazirani audio i četiri tipa slika (*JPEG*, *Progressive JPEG*, *PNG* i *GIF*).

Flash format je postao jako rasprostranjen na tržištu i stvorio je dominaciju, toliko da statistike pokazuju da je na preko 95% računara u svetu instalirana neka verzija, a čak na više od 85% najnovija verzija *Flash*-a. *Adobe Flash Player* postoji za veliki broj platformi i uređaja (npr. *Windows*, *Mac OS X*, *Linux*, *Solaris*, *Symbian*, itd.).

7.3. Video na web stranicama

Flash se može upotrebiti za ubacivanje video sadržaja u web prezentacije. Potrebno je kreirati *SWF* datoteku koja će se ponašati kao plejer za određeni video fajl. Ovo je osnova mnogih popularnih web prezentacija, kao što su *YouTube* i *Google Video*. Pravi video sadržaj se nalazi u *FLV* ili *F4V* datotekama, koje generički video plejer može pustiti. Video sadržaj se može gledati u web pregledaču ako se u njega ugradi (*embed*) video plejer.

```
# <html>
# <body>
# <object width="640" height="505">
#         <param name="movie"
#         value="http://www.youtube.com/v/Yx2zwlj4wK0&hl=en&fs=1&rel=0&
#         color1=0xe1600f&color2=0xfebd01"></param>
#         <param name="allowFullScreen"
#         value="true"></param>
#         <param name="allowscriptaccess" value="always"></param>
#         <embed
#         src="http://www.youtube.com/v/Yx2zwlj4wK0&hl=en&fs=1&rel=0&
#         color1=0xe1600f&color2=0xfebd01"
#         type="application/x-shockwave-flash"
#         allowscriptaccess="always" allowfullscreen="true"
#         width="640" height="505">
#         </embed>
# </object>
# </body>
# </html>
```

Primer 21: Ugrađivanje video plejera u *HTML* kod

Za kodovanje video sadržaja najčešće se koriste *Sorenson Spark*, *On2 VP6* i *H.264* kodeci. Za kodovanje audio sadržaja najčešće se koriste *MP3* i *AAC* formati.

8. IZRADA PREZENTACIJE

8.1. Pripremna faza

Izrada web prezentacije podrazumeva nekoliko pripremnih koraka. Najpre, potrebno je zakupiti domen po kojem će se web prezentacija nalaziti na Internetu. Za web prezentaciju *reality show*-a „Kuća snova“ izabran je domen „kucasnova.tv“. Ekstenzija „.tv“ je internacionalna ekstenzija koju uglavnom koriste sajtovi za prezentaciju TV formata (televizijske i produkcijske kuće, serijski programi, *talk show*-ovi, itd.). Domen je, pri međunarodnom registru domena, zakupila produkcijska kuća *Advantage*.

Sledeće što je potrebno uraditi je analizirati resurse koje će prezentacija trošiti. Pod tim se podrazumeva količina prostora na serveru, kapacitet, brzina, protok i pouzdanost servera. Na samom početku projekta se znalo da će prezentacija imati veliku i konstantnu posećenost, kao i veliki broj multimedijalnih sadržaja (prevashodno videa). Imajući u vidu veličinu i broj video zapisa koji je trebalo da se nađe na serveru, donet je zaključak da bi iznajmljivanje jednog takvog servera bilo preskupo. Kao logično rešenje nametnuo se servis za koji je već bilo poznato da jako pouzdano radi i opslužuje izuzetno veliki broj ljudi – *YouTube*. *Advantage* je od ranije posedovao *Producer* nalog na *YouTube*-u, koji je koristio za promociju svojih formata. Doneta je odlika da se novi sadržaji dodaju na postojeći nalog, budući da je njegov kapacitet bio zadovoljavajuć. Ovim je eliminisana potreba za velikim kapacitetom servera za samu prezentaciju i značajno smanjeno opterećenje na njega.

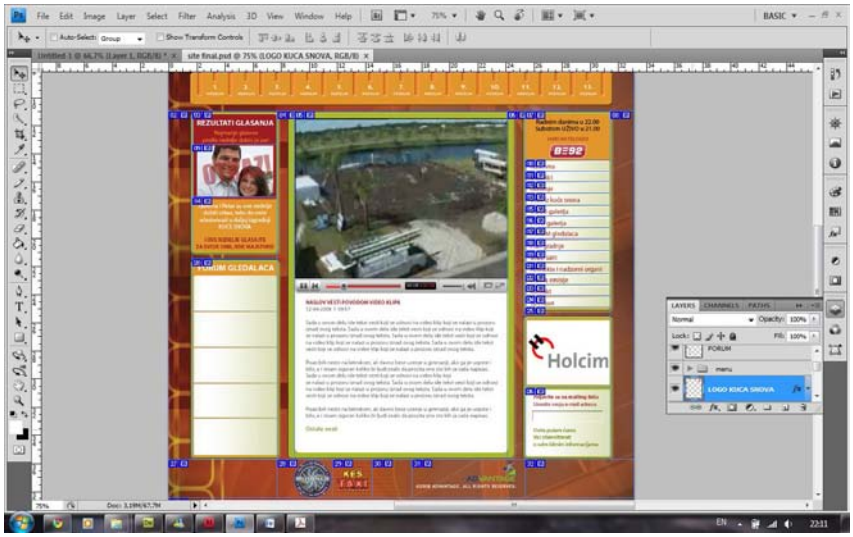
Kao konačno rešenje izabran je server koji se fizički nalazio u Nemačkoj, a preporuka za pouzdanost je bila ta da je isti koristio jedino sajt „KobajaGrande“ i pri tome trošio samo četvrtinu resursa. Nakon pomenutog izbacivanja video sadržaja, planirano je da sajt sa svim propratnim sadržajima i bazom podataka dostigne veličinu od oko 100 MB. Takođe, očekivani broj poseta korisnika nije prelazio hiljadu u toku jednog dana, a trideset u istom trenutku. Budući da je

najveći deo opterećenja prebačen na *YouTube*, pitanje protoka i brzine servera postalo je irelevantno.

Još jedan od zahteva u pogledu servera bio je taj da podržava *PHP* i *MySQL* tehnologije, budući da je bilo neophodno kreiranje dinamičkih elemenata prezentacije i instaliranje foruma. Na serveru su bili instalirani *PHP 5.2*, *MySQL 5.0* i *phpMyAdmin*.

8.2. Dizajn

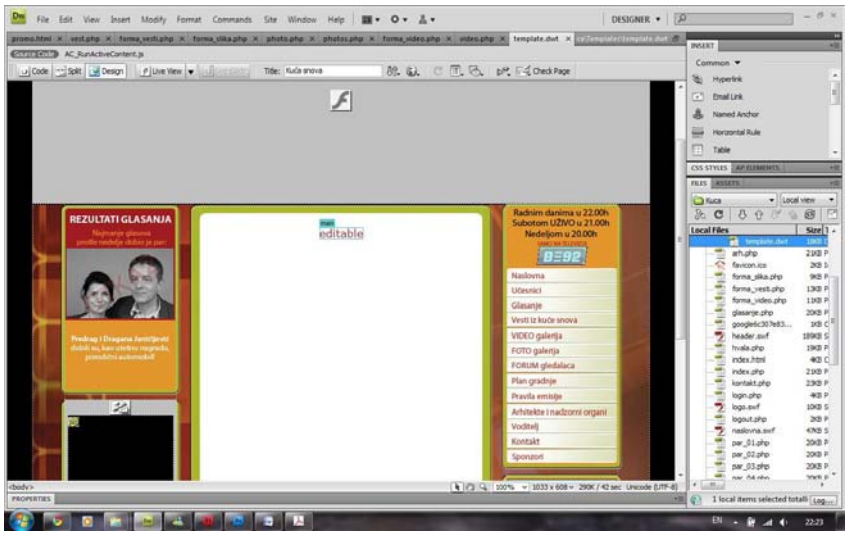
Za dizajn prezentacije je bila zadužena druga osoba, tako da će ovde biti navedene samo najosnovnije informacije. Prvi dizajn prezentacije nije bio adekvatan za primenu na web-u, tako da je konačno rešenje postignuto saradnjom dizajnera i web programera. Nakon dobijanja konačnog izgleda sajta, usledila je njegova obrada u *Adobe Photoshop*-u. Sajt je bilo neophodno prvo podeliti (*slice* alatom) u manje segmente koji će se kasnije lakše koristiti. Kao izlazni format sličica (delova sajta) izabran je *JPEG* sa visokim kvalitetom kompresije, koji je obezbeđivao odličan kvalitet prikaza uz odgovarajuću, tj. malu veličinu datoteka. Sajt je bio optimizovan za prikazivanje u rezoluciji većoj od 1024 x 768 piksela.



Slika 2: Izgled podeljenog sajta u *Photoshop*-u

8.3. Izrada statičkog dela prezentacije

Za posao web programiranja izabran je *Adobe DreamWeaver*, koji je bio poznat kao pouzdano rešenje i koji je mogao da podrži sve zahteve. Kao markup jezik izabran je *XHTML 1.0 Transitional* u kombinaciji sa *CSS 2.1*. Težnja je bila da sam *XHTML* kod ostane što jednostavniji, a da se svi parametri koji se tiču vizuelne prezentacije prebace u posebnu *CSS* datoteku. Imajući u vidu da je trebalo da sve stranice prezentacije imaju istu formu, prvo je napravljen obrazac (*template*) u *DreamWeaver*-u koji je sadržao elemente zajedničke za sve stranice i definisani su stilovi u okviru posebne datoteke.



Slika 3: Izgled *template*-a u *DreamWeaver*-u

U ovoj fazi definisani su meniji, kao i prostor za zaglavlje, rezultate glasanja, termine emitovanja, banere i formu za prijavu na *mailing* listu. Takođe, ostavljen je prostor sa leve strane u kojem je zamišljeno da se prikazuju najnoviji postovi poslednjih pet tema na forumu. U centralnom delu stranice kreiran je jedan editabilni region u kojem će se postavljati sadržaji specifični za pojedine stranice. Meniji su napravljeni upotrebom *rollover* sličica, tj. upotrebom dve različite sličice u zavisnosti od toga da li se kursor miša nalazi iznad stavke menija. Na dnu strane postavljeni su *copyright* informacija i linkovi ka prezentacijama produkcijske kuće *Advantage* i kvizova „Želite li da postanete Milioner?“ i „Keš taxi“. Svi hiperlinkovi u *template*-u su definisani unapred i izvedeni upotrebom *hotspot*-ova, koji nastaju mapiranjem slika sa ciljem da link ne bude cela slika već samo željeni deo slike

(npr. logotip nepravilnog oblika na četvrtastoj slici). Ipak, u meniju su cele sličice bile definisane kao hiperlink, zbog nemogućnosti primene mapiranja na *rollover* parove.

Nakon toga, pristupljeno je izradi svih stranica sa statičkim sadržajem na osnovu prethodno kreiranog obrasca. To su bile: Naslovna, Učesnici, Glasanje, Plan gradnje, Uradi sam, Arhitekte i nadzorni organi, Pravila emisije, Kontakt i Sponzori. Pored ovih strana, napravljene su i posebne strane za svaki od parova učesnika.

Slika 4: Izgled statičke stranice jednog od parova učesnika

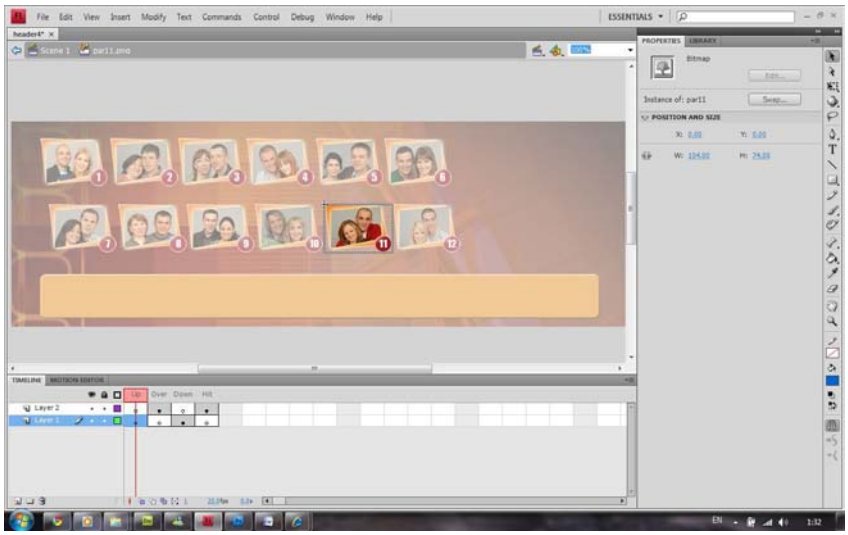
Na svim stranicama prezentacije u donjem desnom uglu nalazila se forma za prijavljivanje korisnika na *mailing* listu. Znatno složenija forma nalazila se na stranici Kontakt. Obe forme su realizovane upotrebom gotove aplikacije *phpMailer-FE*, prilagođene konkretnim potrebama i dizajnu prezentacije, koja sve unešene podatke šalje u vidu poruke na unapred definisanu *e-mail* adresu. Kontakt forma je sadržala i *CAPTCHA*-u (*Completely Automated Public Turing Test To Tell Computers and Humans Apart*), koja je imala zadatak da smanji broj zloupotreba forme (*spam*). Nakon uspešnog popunjavanja forme i prosleđivanja poruke, korisnik je preusmeravan na tzv. *thankyou* stranicu.

Slika 5: Kontakt forma

Jedna od glavnih prednosti upotrebe *template*-a je ta da se u slučaju nekih promena u njemu, sve te promene automatski preslikavaju u datoteke nastale na osnovu njega. Tako su, na primer, jednom nedeljno ažurirani rezultati glasanja samo na jednom mestu, a bili su vidljivi u svim dokumentima prezentacije.

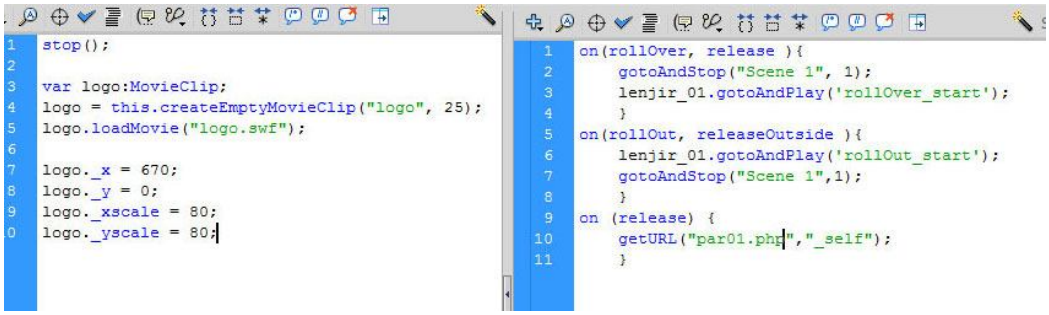
8.4. Izrada zaglavlja

Za izradu zaglavlja korišćen je *Adobe Flash Professional* alat. Zaglavlje je sastavljeno iz dve povezane *SWF* datoteke. Jedna datoteka je sadržala samo transparentni animirani logotip „Kuća snova“, dok se u drugoj nalazilo dvanaest slika parova, slika lenjira, *ActionScript* kod i pozadina.



Slika 6: Definisanje stanja dugmadi u *Flash*-u

Pozadina je precizno pozicionirana kako bi se nadovezala na postojeću pozadinu u *XHTML* kodu. Slike parova su pretvorene u dugmad sa definisanim *up*, *over*, *down* i *hit* stanjima. *Over* stanje je zamenjeno *movie clip*-om, trajanja 12 frejmova (pola sekunde), u kojem je na šestom frejmu uvećana slika. Pravljenjem *motion tween*-a između prvog i poslednjeg frejma, postignut je efekat da prelaskom kursora miša preko para dolazi do zumiranja i odzumiranja slike. Definisanjem *hit* stanja omogućeno je dodavanje hiperlinka, u *ActionScript* kodu, koji vodi do stranice odgovarajućeg para. *ActionScript* kodom je omogućeno i povezivanje *SWF* datoteke u kojoj se nalazio logotip *reality show*-a sa glavnom *SWF* datotekom. Uloga lenjira je bila da ukazuje na vreme (u nedeljama) od početka serijala, sve dok se kursorom miša ne pređe preko slike nekog para, kada bi se sadržaj lenjira menjao i prikazivao imena, starost i grad iz kojeg je par došao.



Slika 7: Povezivanje sa drugom *SWF* datotekom (levo) i definisanje promene lenjira i hiperlinka za *release* akciju dugmeta (desno)

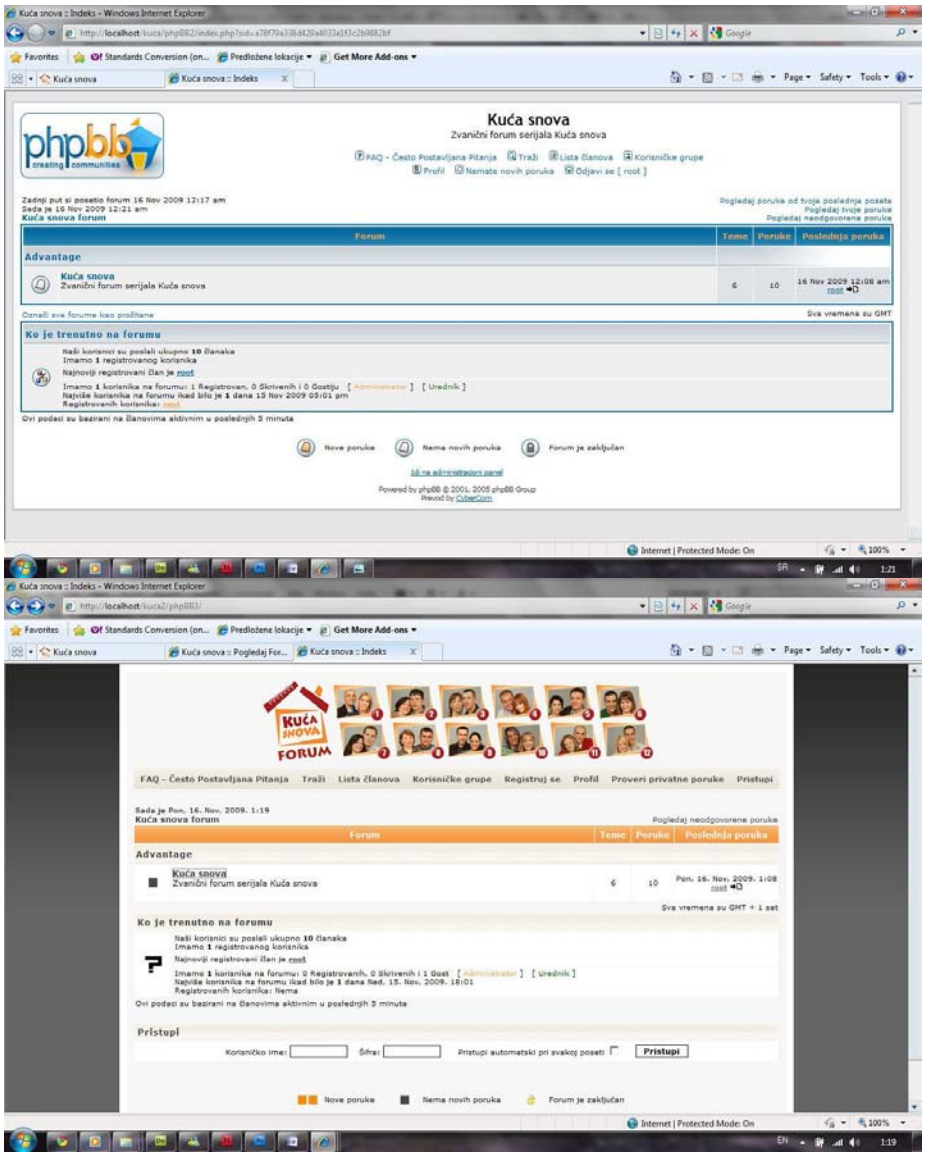
8.5. Instalacija foruma

Jedan od zahteva prilikom planiranja prezentacije bio je postojanje foruma, putem kojeg bi gledaoci mogli da diskutuju o serijalu. Za ovu potrebu izabrano je gotovo rešenje u vidu *phpBB* foruma u verziji *2*. U trenutku izrade ove prezentacije, postojala je i *phpBB 3* verzija, ali su podrška i dodaci za nju bili na daleko nižem nivou. Ovaj forum za svoj rad zahteva instaliran *PHP* i *MySQL* bazu podataka.

Sama instalacija foruma traje kratko. Nakon pozivanja datoteke *install.php* putem web pregledača, otvara se stranica sa osnovnim podešavanjima gde se vrši izbor jezika i unose parametri baze podataka (server, naziv, korisničko ime i šifra),

prefiks za ime svake tabele koju će *phpBB* napraviti u bazi (što je pogodno ako se ne instalira u praznu bazu podataka) i podaci o administratoru foruma.

Po završetku instalacije, prikazuje se početna strana foruma i, nakon prijavljivanja sa prethodno definisanim parametrima, link ka administracionom panelu. U ovom panelu može se podesiti jezik foruma i prikaz vremena i datuma, kreirati podforum, definisati kategorije, upravljati korisnicima i grupama korisnika, uticati na izgled foruma, definisati razne zabrane, itd. Prvi korak je bio promeniti izgled foruma i približiti ga izgledu prezentacije. To je postignuto menjanjem *CSS* datoteka samog foruma, kao i promenom određenih slika unutar foruma.



Slika 8: Osnovni izgled foruma (gore) i prilagođeni izgled foruma (dole)

U drugom koraku definisana su neka pravila funkcionisanja foruma, kao što su način registracije korisnika, pravila otvaranja tema, postovanja, cenzure, itd. Za kontrolu sadržaja koji su korisnici objavljivali izabran je i instaliran dodatni modul *SpamWords* u kojem se na lak način definišu reči i kombinacije reči koje nisu dozvoljene.

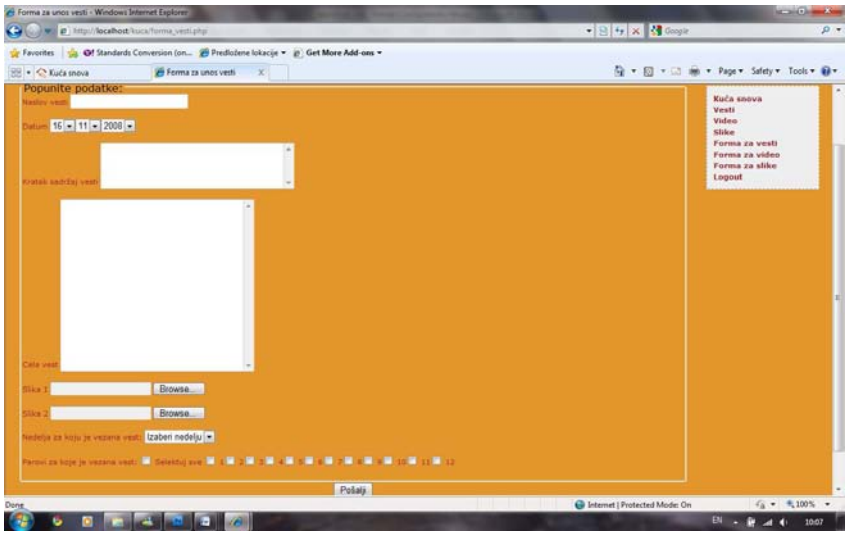
Treći korak je bio omogućiti prikaz najnovijih postova u ranije dodeljenom prostoru na stranicama osnovne prezentacije. Nakon malo podrobnijeg upoznavanja sa strukturom *phpBB 2* foruma i konsultacijom sa zajednicom korisnika, uspešno je napisan *PHP* kod koji pruža traženu funkcionalnost. Kod je zatim umetnut na odgovarajuće mesto u *template*-u, a samim tim i u sve stranice prezentacije.

Slika 9: Prikaz najnovijih postova sa foruma

8.6. Izrada dinamičkog dela prezentacije

Dinamički deo prezentacije „Kuća snova“ trebalo je da zadovolji nekoliko zahteva. Prvi od njih je bila mogućnost da urednici produkcijske kuće *Advantage* (bez poznavanja tehnologija web programiranja) na brz i jednostavan način dodaju sadržaje na sajt. Postojalo je tri tipa sadržaja koje je trebalo dodavati: vesti, fotografije i video zapise. Prvi zadatak je bio napraviti forme za unos podataka, povezati ih sa *PHP* kodom, a zatim pohraniti podatke u bazu podataka. Bilo je

potrebno kreirati tri različite forme. Primer kompletnog koda forme za unos vesti može se videti u Prilogu 1. Kod je u osnovi identičan za sve tri forme, a razlike se odnose samo na razlike u poljima forme i, samim ti, kolonama u tabelama baze podataka. Zajedničke funkcionalnosti su: konekcija sa bazom, dodeljivanje vrednosti iz forme *PHP* promenljivima, provera validnosti podataka, *upload* datoteka i čuvanje podataka u bazu.

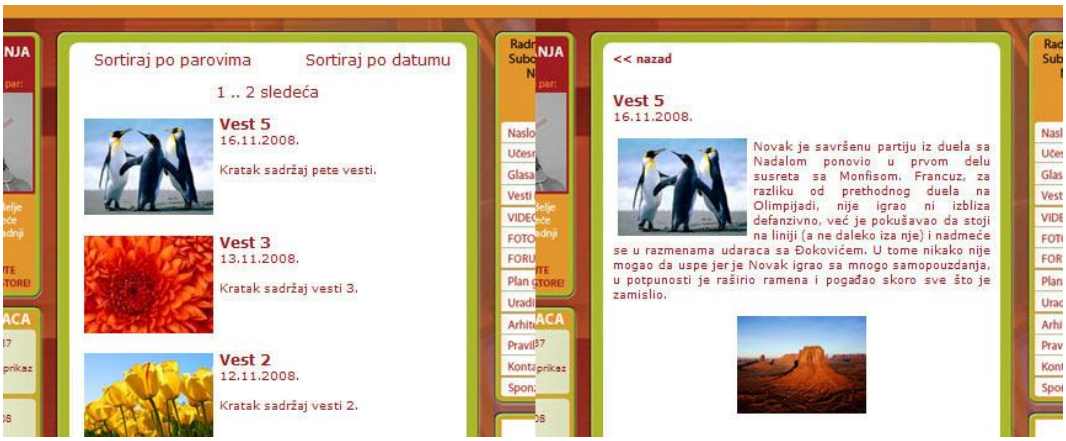


Slika 10: Izgled forme za unos vesti

Da bi se sajt obezbedio od neovlašćenih dodavanja sadržaja, u bazi podataka postojala je tabela korisnika koji su morali da se prijave na sistem pre unosa podataka. To je izvedeno upotrebom *login* forme i prostim poređenjem unešenih podataka sa onima u bazi. Nakon uspešnog unosa podataka, upotrebom opcije *logout*, ranije napravljen *cookie* se brisao iz sistema i time eliminisao mogućnost neovlašćenog pristupa.

Slika 11: Forma za prijavu na sistem

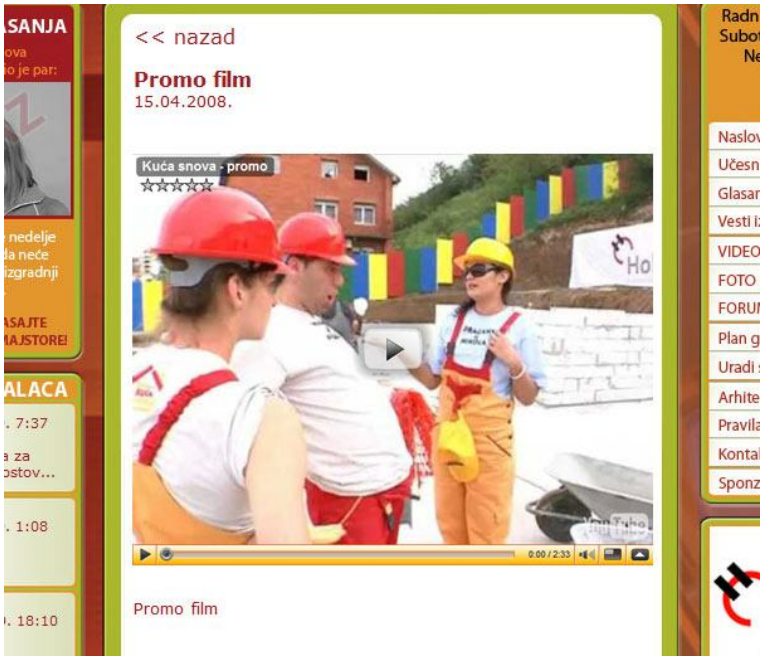
Drugi zadatak je bio unešene podatke prikazati i sa novim unosima automatski ažurirati. To je, ponovo, realizovano kombinacijom *PHP* i *MySQL* tehnologija. *PHP* kod je prosleđivao upite bazi podataka koja je zatim vraćala podatke. Vraćeni podaci su kroz *PHP* kod umetani u prethodno definisane elemente *XHTML markup*-a koji su se na jednostavan način mogli ponavljati. Time je dobijena mogućnost izbora broja prikazanih rezultata na stranici. Ukoliko bi taj broj bio premašen, pojavio bi se *pager* koji bi nam omogućio da listamo rezultate raspoređene na više stranica. Pored toga, postojala je i mogućnost sortiranja rezultata na bazi jednog od dva ponuđena kriterijuma, bilo u opadajućem ili rastućem nizu. Primer koda korišćenog za prikaz vesti može se videti u Prilogu 2.



Slika 12: Izgled liste vesti (levo) i izgled pojedinačne vesti (desno)

Prilikom dodavanja video zapisa postupak je bio nešto drugačiji. Naime, video klipovi su od strane video montažera već bili *upload*-ovani na *YouTube*, tako da je urednik preuzimao samo *URL* video klipa i ubacivao ga sa drugim tekstualnim podacima u odgovarajuće polje forme i dalje u bazu. U delu za prikaz video sadržaja bio je ugrađen video plejer koji je kao promenljivu iz baze podataka dobijao samo *URL* video klipa.

Sistem je radio izuzetno pouzdano i jedina mana koja je primećena odnosila se na ljudski faktor. Ukoliko bi došlo do pogrešnog unosa od strane urednika, prepravke su morale biti izvedene od strane administratora direktno u samoj bazi podataka, upotrebom alata *phpMyAdmin*. Zbog nedostatka vremena ovaj problem nikada nije rešen, ali nije predstavljao smetnju u radu.



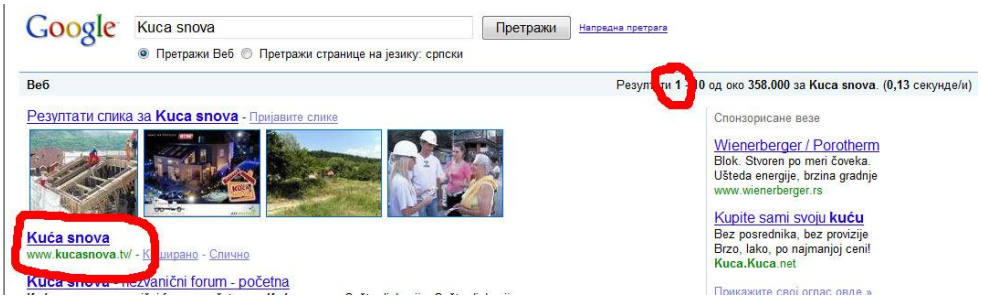
Slika 13: Izgled video plejera

8.7. Objavljivanje i optimizacija

Tokom kompletnog perioda izrade, ova prezentacija se nalazila na serveru i funkcionisala u realnim uslovima, samo daleko od očiju javnosti. To je postignuto postavljanjem posebne *index.html* datoteke koja je sadržala samo najavu novog *reality show*-a i novog sajta. Naslovna strana prezentacije u izradi nosila je ekstenziju „.php“ i nije bila učitavana automatski, zbog prioriteta koji je bio postavljen na serveru (prednost „.html“ datoteka u odnosu na „.php“). Rad prezentacije u realnim uslovima tokom razvoja omogućio je minimizaciju vremena potrebnog za objavljivanje prezentacije u javnost, tj. sveo ga je na nulu. Prezentacija je svakodnevno testirana tokom razvoja tako da nije bilo potrebno odvojiti dodatno vreme za te potrebe, već je prezentacija mogla biti objavljena odmah po završetku razvoja.

Slika 14: Konačni izgled naslovne strane

Nakon objavljivanja prezentacije, pristupilo se optimizaciji rezultata pretrage putem najpopularnijeg pretraživača, *Google*-a. Iskorišćena je opcija kreiranja mape sajta u *XML* formatu, koju nudi *DreamWeaver*. Usledilo je prijavljivanje sajta na *Google*, omogućavanje *Google* robotu da pretražuje sadržaj prezentacije i definisanje opisa sajta i ključnih reči. Prvi rezultati bili su vidljivi već posle nekoliko dana, a prezentacija je dospela na prvo mesto posle dvanaest dana, pet dana nakon početka emitovanja *show*-a.



Slika 15: Prvo mesto na *Google*-u

9. ZAKLJUČAK

Na kraju ovog projekta mogu da zaključim da mi je ovo iskustvo bilo značajno iz nekoliko razloga. Najpre, proširio sam svoje znanje u oblasti web programiranja i ovladao trenutno najpopularnijim i najzastupljenijim tehnologijama za izradu dinamičkih sadržaja web prezentacija. U planu mi je da i dalje radim na pospešivanju svojih znanja iz ove oblasti i istražujem nove mogućnosti. Kao drugo, po prvi put sam učestvovao u ovako velikom projektu i veoma uspešno sarađivao sa čitavim timom ljudi u veoma teškim uslovima. Naime, vremenski rokovi su bili izuzetno kratki i, imajući u vidu šta je sve urađeno za samo desetak dana, mislim da je obavljen odličan posao.

O uspešnosti projekta govore i podaci da je prezentacija dok je *show* bio aktuelan bila veoma posećena i da je odgovorila najbitnijem zahtevu – funkcionalnosti. Moj rad na prezentaciji „Kuća snova“ produkcijska kuća Advantage ocenila je kao vrlo dobar, tako da smo nastavili saradnju koja i dalje traje na projektima kao što su „Želite li da postanete Milioner?“, „Keš taxi“, „Narod protiv“, itd. Takođe, održavam i prezentaciju same produkcijske kuće.

10. PRILOG

10.1. Prilog 1: Izgled koda forme za unos vesti

```
1   <?php
2   ob_start();
3   $page_title = 'Forma za unos vesti';
4   include ('./includes/header.html');
5   if (!isset($_SESSION['first_name'])) {
6   $url = 'http://' . $_SERVER['HTTP_HOST'] .
    dirname($_SERVER['PHP_SELF']);
7   if ((substr($url, -1) == '/') OR (substr($url, -1) == '\\') ) {
8   $url = substr ($url, 0, -1);
9   }
10  $url .= '/index.php';
11  ob_end_clean();
12  header("Location: $url");
13  exit();
14  }
15  ?>
16  <?php
17  if (isset($_POST['submitted'])) {
18  $dbc = @mysql_connect ('localhost', 'kucasnova', 'kucasnova123') OR
    die ('Could not connect to MySQL: ' . mysql_error() );
19  @mysql_select_db ('kucasnova') OR die ('Could not select the
    database: ' . mysql_error() );
20  $errors = array();
21  if (empty($_POST['naslov_vesti'])) {
22  $errors[] = 'Zaboravili ste da unesete naslov vesti.';
23  } else {
24  $nasl = $_POST['naslov_vesti'];
25  }
26  $dan = '' . $_POST['year'] . '-' . $_POST['month'] . '-' .
    $_POST['day'];
27  if (empty($_POST['kratka_vest'])) {
28  $errors[] = 'Zaboravili ste da unesete kratak sadržaj vesti.';
29  } else {
30  $kv = $_POST['kratka_vest'];
31  }
32  if (empty($_POST['dugacka'])) {
33  $errors[] = 'Zaboravili ste da unesete celu vest.';
34  } else {
35  $vest = $_POST['dugacka'];
36  }
37  if (isset($_FILES[$slika1]) && ($_FILES[$slika1]['error'] != 4)) {
38  $errors[] = 'Zaboravili ste da unesete prvu sliku.';
39  } else {
40  }
41  if (isset($_FILES[$slika2]) && ($_FILES[$slika2]['error'] != 4)) {
42  $errors[] = 'Zaboravili ste da unesete drugu sliku.';
43  } else {
44  }
45  if (empty($_POST['nedelja'])) {
46  $errors[] = 'Zaboravili ste da unesete nedelju za vest.';
47  } else {
48  $ned = $_POST['nedelja'];
49  }
```

```
50  if (!isset($_POST['par'])) {
51  $errors[] = 'Zaboravili ste da izaberete par ili parove za vest.';
52  } else {
53  $par = $_POST['par'];
54  }
55  if (empty($errors)) {
56  $query1 = "INSERT INTO slike (file_name, file_size, file_type) VALUES
    ('{$_FILES[slika1]['name']}', {$_FILES[slika1]['size']},
    '{$_FILES[slika1]['type']}')";
57  $result1 = mysql_query($query1);
58  if ($result1) {
59  $slika_id1 = mysql_insert_id();
60  $name_niz1 = explode('.',$_FILES['slika1']['name']);
61  $name1 = $slika_id1 . '.' . $name_niz1[1];
62  if (move_uploaded_file($_FILES['slika1']['tmp_name'],
    "./images/vesti/$name1")) {
63  echo '<p>Prva slika je sačuvana!</p>';
64  $query2 = "INSERT INTO slike (file_name, file_size, file_type) VALUES
    ( '{$_FILES[slika2]['name']}', {$_FILES[slika2]['size']}, '{$_FILE
    S[slika2]['type']}')";
65  $result2 = mysql_query($query2);
66  if ($result2) {
67  $slika_id2 = mysql_insert_id();
68  $name_niz2 = explode('.',$_FILES['slika2']['name']);
69  $name2 = $slika_id2 . '.' . $name_niz2[1];
70  if (move_uploaded_file($_FILES['slika2']['tmp_name'],
    "./images/vesti/$name2")) {
71  echo '<p>Druga slika je sačuvana!</p>';
72  $query5 = "INSERT INTO vest (naslov, datum, kratka, dugacka,
    slika_id1, slika_id2, nedelja) VALUES ( '$nasl', '$dan', '$kv',
    '$vest', $slika_id1, $slika_id2, $ned)";
73  $result5 = mysql_query($query5);
74  if ($result5) {
75  $vest_broj = mysql_insert_id();
76  foreach ($par as $key => $value) {
77  $query6 = "INSERT INTO vesti (vest_id, par) VALUES ( $vest_broj,
    $value)";
78  $result6 = mysql_query($query6);
79  if ($result6) {
80  } else {
81  echo '<p><font color="red">Neki par je trsao.</font></p>';
82  }
83  }
84  } else {
85  echo '<p><font color="red">Your submission could not be processed due
    to a system error. We apologize for any inconvenience.</font></p>';
86  }
87  } else {
88  echo '<p><font color="red">Druga slika nije sačuvana!</font></p>';
89  $query3 = "DELETE FROM slike WHERE slika_id = $slika_id2";
90  $result3 = mysql_query ($query3);
91  $query4 = "DELETE FROM slike WHERE slika_id = $slika_id1";
92  $result4 = mysql_query ($query4);
93  unlink ('./images/vesti/$name1');
94  }
95
96  } else {
97  echo '<p><font color="red">Druga slika je trsla.</font></p>';
98  }
99  } else {
100 echo '<p><font color="red">Prva slika nije sačuvana!</font></p>';
```

```
$query = "DELETE FROM slike WHERE slika_id = $slika_id1";
$result = mysql_query ($query);
}
} else {
echo '<p><font color="red">Prva slika je trsla.</font></p>';
}
} else {
echo '<h1 id="mainhead">Error!</h1>
<p class="error">Pri unosu podataka desile su se sledeće
greške:<br/>';
foreach ($errors as $msg) {
echo " - $msg<br />\n";
}
echo '</p><p>Pokušajte ponovo.</p><p><br /></p>';
}
mysql_close();
}
?><style type="text/css">
<!--
body,td,th {
font-family: Verdana, Arial, Helvetica, sans-serif;
color: #9D1E25;
}
body {
background-color: #E3962C;
}
a:link {
text-decoration: none;
color: #9D1E25;
}
a:visited {
text-decoration: none;
color: #9D1E25;
}
a:hover {
text-decoration: none;
color: #9D1E25;
}
a:active {
text-decoration: none;
color: #9D1E25;
}
-->
</style>
<script type="text/javascript">
<!--
pageMem = false;
function SetAllCheckBoxes(FormName, FieldName)
{
if(!document.forms[FormName])
return;
var objCheckBoxes = document.forms[FormName].elements[FieldName];
if(!objCheckBoxes)
return;
if (pageMem == false) pageMem = true; else pageMem = false;
var countCheckBoxes = objCheckBoxes.length;
if(!countCheckBoxes)
objCheckBoxes.checked = pageMem;
else
for(var i = 0; i < countCheckBoxes; i++)
objCheckBoxes [i].checked = pageMem;
```

```
161 }
162 // -->
163 </script>
164 <form enctype="multipart/form-data" name="myForm"
    action="forma_vesti.php" method="post">
165 <fieldset><legend>Popunite podatke:</legend>
166 <input type="hidden" name="MAX_FILE_SIZE" value="524288">
167 <p>Naslov vesti
168 <input name="naslov_vesti" type="text" id="naslov_vesti" size="24"
    maxlength="48" /><br />
169 </p>
170 <p>Datum
171 <?php
172 function make_calendar_pulldowns ($m = NULL, $d = NULL, $y = NULL) {
173 echo '<select name="day">' ;
174 for ($day = 1; $day <= 9; $day ++) {
175 echo "<option value=\"0$day\"" ;
176 if ($day == $d) {
177 echo ' selected="selected"' ;
178 }
179 echo ">$day</option>\n" ;
180 }
181 for ($day = 10; $day <= 31; $day ++) {
182 echo "<option value=\"$day\"" ;
183 if ($day == $d) {
184 echo ' selected="selected"' ;
185 }
186 echo ">$day</option>\n" ;
187 }
188 echo '</select>' ;
189
190 echo '<select name="month">' ;
191 for ($month = 1; $month <= 9; $month ++) {
192 echo "<option value=\"0$month\"" ;
193 if ($month == $m) {
194 echo ' selected="selected"' ;
195 }
196 echo ">$month</option>\n" ;
197 }
198 for ($month = 10; $month <= 12; $month ++) {
199 echo "<option value=\"$month\"" ;
200 if ($month == $m) {
201 echo ' selected="selected"' ;
202 }
203 echo ">$month</option>\n" ;
204 }
205 echo '</select>' ;
206 echo '<select name="year"><option
    value="2008">2008</option></select>' ;
207 }
208 $dates = getdate ();
209 make_calendar_pulldowns ($dates ['mon' ], $dates ['mday' ], $dates
    ['year' ]);
210 ?>
211 </p>
212 <p>Kratak sadržaj vesti
213 <textarea name="kratka_vest" id="kratka_vest" cols="34"
    rows="4"></textarea><br />
214 </p>
215 <p>Cela vest
```

```
216 <textarea name="dugacka" id="dugacka" cols="34"
    rows="16"></textarea><br />
217 </p>
218 <p>Slika 1
219 <input type="file" name="slika1" id="slika1" /><br />
220 </p>
221 <p>Slika 2
222 <input type="file" name="slika2" id="slika2" /><br />
223 </p>
224 <p>Nedelja za koju je vezana vest:
225 <select name="nedelja" id="nedelja">
226 <option value="0">Izaberi nedelju</option>
227 <option value="1">1</option>
228 <option value="2">2</option>
229 <option value="3">3</option>
230 <option value="4">4</option>
231 <option value="5">5</option>
232 <option value="6">6</option>
233 <option value="7">7</option>
234 <option value="8">8</option>
235 <option value="9">9</option>
236 <option value="10">10</option>
237 <option value="11">11</option>
238 <option value="12">12</option>
239 <option value="13">13</option>
240 </select><br />
241 </p>
242 <p>Parovi za koje je vezana vest:
243 <input type="checkbox" name="check_all"
    onclick="SetAllCheckBoxes('myForm', 'par[]')"> Selektuj sve
244 <?php
245 for ($i = 1; $i < 13; $i++) {
246 echo '<input type="checkbox" name="par[]" id="par" value="' . $i . '"
    /> ' . $i;
247 }
248 ?>
249 </p>
250 </fieldset>
251 <input type= "hidden" name="submitted" value= "TRUE" />
252 <div align= "center" ><input type= "submit" name="submit" value=
    "Pošalji" />
253 </div>
254 </form>
255 <?php
256 include ('./includes/footer.html' );
257 ?>
```

10.2. Prilog 2: Izgled dela koda stranice za prikaz vesti

```
292 <table width="410" border="0" cellspacing="0" cellpadding="0">
293 <?php
294 $dbc = @mysql_connect ('localhost', 'kucasnova', 'kucasnova123') OR
    die ('Could not connect to MySQL: ' . mysql_error() );
295 @mysql_select_db ('kucasnova') OR die ('Could not select the
    database: ' . mysql_error() );
296 $display = 4;
297 if (isset($_GET['np'])) {
298 $num_pages = $_GET['np'];
299 } else {
300 if (isset($_GET['sort'])) {
301 switch ($_GET['sort']) {
302 case 'lna':
303 $query = "SELECT COUNT(*) FROM vesti ORDER BY vesti_id ASC";
304 break;
305 case 'lnd':
306 $query = "SELECT COUNT(*) FROM vesti ORDER BY vesti_id DESC";
307 break;
308 case 'fna':
309 $query = "SELECT COUNT(*) FROM vest ORDER BY vest_id ASC";
310 break;
311 case 'fnd':
312 $query = "SELECT COUNT(*) FROM vest ORDER BY vest_id DESC";
313 break;
314 default:
315 $query = "SELECT COUNT(*) FROM vest ORDER BY vest_id DESC";
316 break;
317 }
318 $sort = $_GET['sort'];
319 } else {
320 $query = "SELECT COUNT(*) FROM vest ORDER BY vest_id DESC";
321 }
322 $result = @mysql_query ($query);
323 $row = mysql_fetch_array ($result, MYSQL_NUM);
324 $num_records = $row[0];
325 if ($num_records > $display) {
326 $num_pages = ceil ($num_records/$display);
327 } else {
328 $num_pages = 1;
329 }
330 }
331 if (isset($_GET['s'])) {
332 $start = $_GET['s'];
333 } else {
334 $start = 0;
335 }
336 $link1 = "{$_SERVER['PHP_SELF']}?sort=lna";
337 $link2 = "{$_SERVER['PHP_SELF']}?sort=fna";
338 if (isset($_GET['sort'])) {
339 switch ($_GET['sort']) {
340 case 'lna':
341 $order_by = "SELECT vesti_id, vest.vest_id, par, vest.naslov,
    vest.datum, kratka, slika_id1, nedelja FROM vesti, vest WHERE
    vesti.vest_id = vest.vest_id ORDER BY par ASC, vest.datum DESC LIMIT
    $start, $display";
342 $link1 = "{$_SERVER['PHP_SELF']}?sort=lnd";
343 break;
344 case 'lnd':
```

```
345 $order_by = "SELECT vesti_id, vest.vest_id, par, vest.naslov,
    vest.datum, kratka, slika_id1, nedelja FROM vesti, vest WHERE
    vesti.vest_id = vest.vest_id ORDER BY par DESC, vest.datum DESC LIMIT
    $start, $display";
346 $link1 = "{$_SERVER['PHP_SELF']}?sort=lna";
347 break;
348 case 'fna':
349 $order_by = "SELECT vest_id, naslov, datum, kratka, slika_id1 FROM
    vest ORDER BY datum ASC LIMIT $start, $display";
350 $link2 = "{$_SERVER['PHP_SELF']}?sort=fnd";
351 break;
352 case 'fnd':
353 $order_by = "SELECT vest_id, naslov, datum, kratka, slika_id1 FROM
    vest ORDER BY datum DESC LIMIT $start, $display";
354 $link2 = "{$_SERVER['PHP_SELF']}?sort=fna";
355 break;
356 default:
357 $order_by = 'vest.datum DESC';
358 break;
359 }
360 $sort = $_GET['sort'];
361 } else {
362 $order_by = "SELECT vest_id, naslov, datum, kratka, slika_id1 FROM
    vest ORDER BY datum DESC LIMIT $start, $display";
363 $sort = 'fnd';
364 }
365 echo '<tr>
366 <p align="center"><a href="' . $link1 . '">  Sortiraj po
    parovima</a>        &
    nbsp;  <a href="' . $link2 . '">Sortiraj po datumu</a></p>
367 </tr>';
368 $query1 = $order_by;
369 $result1 = mysql_query ($query1);
370 while ($row1 = mysql_fetch_array($result1, MYSQL_ASSOC)) {
371 $datum1 = explode('-',$row1['datum']);
372 $datum = $datum1[2] . '.' . $datum1[1] . '.' . $datum1[0] . '.';
373 $slika_id_1 = $row1['slika_id1'];
374 $query3 = "SELECT slika_id, file_name, file_size, file_type FROM
    slike WHERE slika_id = $slika_id_1";
375 $result3 = @mysql_query($query3);
376 $row3 = mysql_fetch_array($result3, MYSQL_ASSOC);
377 $name_niz1 = explode('.',$row3['file_name']);
378 $slika1 = $row3['slika_id'] . '.' . $name_niz1[1];
379 echo '
380 <tr>
381 <td><p align="left"><p align="left"><a href="vest.php?id=' .
    $row1['vest_id'] . '&s=' . $start . '&np=' . $num_pages . '&sort=' .
    $sort . '"><img src="./images/vesti/' . $slika1 . '" alt=""
    width="140" height="105" border="0" hspace="6" vspace="3"
    align="left" /><span class="head">' . $row1['naslov'] . '<br/>
    </span><span class="text">' . $datum . '</span><br />';
382 if ($sort == 'lnd' or $sort == 'lna') echo '<span class="text">' .
    'Par broj: ' . $row1['par'] . '</span>';
383 echo '</a></p>
384 <p align="justify" class="text"><a href="vest.php?id=' .
    $row1['vest_id'] . '">' . $row1['kratka'] . '</a></p><p></p>
385 </td>
386 </tr>';
387 }
388 mysql_close();
389 if ($num_pages > 1) {
```

```
390 echo '<tr>';
391 $current_page = ($start/$display) + 1;
392 if ($current_page != 1) {
393 echo '<a href="vesti.php?s=' . ($start - $display) . '&np=' .
    $num_pages . '&sort=' . $sort .'">prethodna </a> ';
394 }
395 if ($current_page != 1) {
396 echo '<a href="vesti.php?s=' . '0' . '&np=' . $num_pages . '&sort=' .
    $sort .'">' . '1' . ' .. </a> ';
397 } else {
398 echo ' 1 ';
399 }
400 for ($i = 2; $i <= $num_pages-1; $i++) {
401 if ($i != $current_page) {
402 if (($i == $current_page - 3) or ($i == $current_page - 2) or ($i ==
    $current_page - 1) or ($i == $current_page + 1) or ($i ==
    $current_page + 2) or ($i == $current_page + 3)) {
403 echo '<a href="vesti.php?s=' . (($display * ($i - 1))) . '&np=' .
    $num_pages . '&sort=' . $sort .'">' . $i . '</a> ';
404 }
405 } else {
406 echo $i . ' ';
407 }
408 }
409 if ($current_page != $num_pages) {
410 echo '<a href="vesti.php?s=' . (($display * ($num_pages -1))) .
    '&np=' . $num_pages . '&sort=' . $sort .'">.. ' . $num_pages . '</a>
    ';
411 } else {
412 echo ' ' . $num_pages . ' ';
413 }
414 if ($current_page != $num_pages) {
415 echo '<a href="vesti.php?s=' . ($start + $display) . '&np=' .
    $num_pages . '&sort=' . $sort .'"> sledeća</a>';
416 }
417 echo '</tr>';
418 }
419 ?>
420 </table>
```

11. LITERATURA

1. *World Wide Web Consortium* (*W3C*) – http://www.w3.org
2. *W3 Schools* – http://www.w3schools.com
3. *PHP* – http://www.PHP.NET
4. *MySQL* – http://www.MySQL.com
5. David Sawyer McFarland, *Dreamweaver CS3: Uputstvo koje vam nedostaje*, Kompjuter biblioteka, Beograd, 2007.
6. Luke Welling, Laura Thomson, *PHP i MySQL – Razvoj aplikacija za Web*, Mikro knjiga, Beograd, 2009.
7. Wikipedia: *Web development* – http://en.wikipedia.org/wiki/Web_development
8. Wikipedia: *Web design* – http://en.wikipedia.org/wiki/Web_design
9. Wikipedia: *HTML* – http://en.wikipedia.org/wiki/HTML
10. Wikipedia: *XHTML* – http://en.wikipedia.org/wiki/XHTML
11. Wikipedia: *PHP* – http://en.wikipedia.org/wiki/PHP
12. Wikipedia: *SQL* – http://en.wikipedia.org/wiki/SQL
13. Wikipedia: *MySQL* – http://en.wikipedia.org/wiki/MySQL
14. Wikipedia: *Flash* – http://en.wikipedia.org/wiki/Adobe_Flash