Univerza v Ljubljani

Fakulteta za elektrotehniko

Laboratorij za komunikacijske naprave

Študijsko gradivo in navodila za laboratorijske vaje pri predmetu Spletne tehnologije

Bolonjski študijski program (1. stopnja VS)
MMK – Vrtojba

Grega Jakus

š.l. 2013/2014

Tržaška 25
1000 Ljubljana
Slovenija

Tel.: (01) 476 84 11
Fax: (01) 426 46 30

Vaja 1: Uvod v spletne tehnologije

1. Naslavljanje v internetu

Protokolni sklad TCP / IP

aplikacijska plast	HTTP, DNS, FTP, POP, SMTP, …
transportna plast	**TCP**, UDP, …
internetna plast	**IP**, IPsec, …
povezovalna plast	ARP, PPP, OSPF, MAC (Ethernet, DSL, ISDN, …)
fizična plast (ni več del TCP/IP sklada)	

Naslavljanje na internetnem in povezovalnem sloju

Omrežne nastavitve lokalnega računalnika

1. Izberite **Start** in nato **Zaženi…(Run…)**
2. Zaženite okno konzole z ukazom **cmd.**
3. V konzoli z ukazom **ipconfig /all** izpišite mrežne nastavitve operacijskega sistema Windows.

   Poiščite in zapišite:
   - IP naslov lokalnega računalnika
   - Masko podomrežja
   - Privzeti prehod
   - Naslov DNS strežnika
   - Fizični (MAC) naslov omrežnega vmesnika v lokalnem računalniku

4. Kaj je IP naslov in čemu služi?

5. Kaj je maska podomrežja in čemu služi?

6. Kaj je fizični naslov omrežnega vmesnika in čemu služi?

Domensko naslavljanje

Hierarhični sistem domen omogoča berljivo naslavljanje naprav v internetu. Ime naprave, priključene v internet, je sestavljeno iz

1. vrhnje (glavne) domene
2. registrirane poddomene
3. ostalih poddomen

Domenska imena se prevajajo v IP naslove s pomočjo imenskega (DNS) strežnika.

1. Kakšna je vloga imenskega strežnika?

2. V sledečih naslovih določite vrhnjo domeno, registrirano domeno in ostale poddomene.

   www.arnes.si www.lkn.fe.uni-lj.si/files/datoteka2.txt

3. Na spletu poiščite 5 oznak vrhnjih domen in navedite področja, na katera se nanašajo!

Naslavljanje na transportnem sloju

Naslavljanje na transportnem nivoju je izvedeno s pomočjo vrat (ang. *port*). Vrata so navidezna povezava, preko katere si programi izmenjujejo podatke, največkrat preko transportnega sistema.

1. Poiščite oznake vrat, ki jih navadno uporabljajo naslednji internetni protokoli aplikacijskega sloja:

http (protokol za prenos medsebojno povezanih dokumentov):

ftp (protokol za prenos datotek):

https (varni http):

2. Naštejte in na kratko opišite še 5 drugih protokolov aplikacijskega nivoja in na spletu poiščite, katera vrata navadno uporabljajo pri komunikaciji.

Naslavljanje z uporabo URL naslovov

Vsak internetni vir ima svoj lasten spletni naslov, znan pod imenom URL (*Uniform Resource Locator*).

URL naslovi so sestavljeni iz:

1. mehanizma dostopa (aplikacijski protokol)
2. oznake računalnika, kjer se nek vir nahaja, v obliki IP ali domenskega naslova
3. oznake vrat
4. poti do vira na računalniku
5. imena vira

Primer URL: http://www.lkn.fe.uni-lj.si:81/podatki/vaje/datoteka2.doc

Pozor: URL naslovi so občutljivi na velikosti črk!

V spodnjih URL naslovih določite uporabljen protokol, ime računalnika, naslov uporabljenih vrat, pot do vira in ime vira:

http://www.lkn.fe.uni-lj.si/slike/slika3.jpg

http://212.235.190.204:80/index.php

ftp://mp3.glasba.si/mojeSkupine/

Preko katerih vrat skuša spletni brskalnik komunicirati, če številke vrat v naslovu ne določimo?

2. Postavitev spletnega strežnika

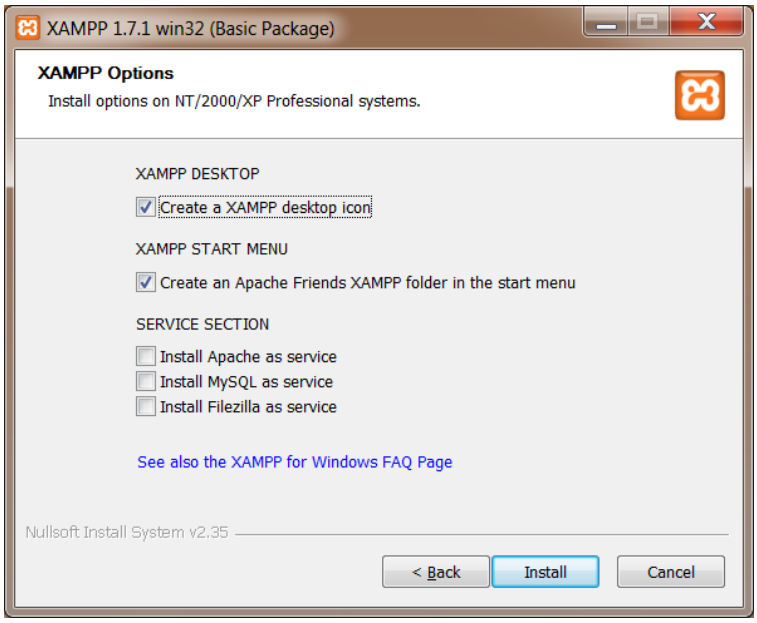
Spletni strežnik (ang. *web server*) je računalniški program, ki je odgovoren za sprejemanje zahtev spletnih odjemalcev, njihovo obravnavo in posredovanje odgovorov na zahteve. Odgovor navadno vsebuje spletno stran (v jeziku HTML) in druge, z njo povezane objekte (na primer skripte, slike, video itd).

Na vajah bomo uporabljali programski paket XAMPP, ki združuje

- spletni strežnik Apache,
- sistem za upravljanje s podatkovnimi bazami MySQL,
- datotečni strežnik FileZilla in
- poštni strežnik Mercury.

Če želite namestiti in nastavljati module XAMPP-a, potrebujete uporabniški račun z administratorskimi pravicami.

1. S spletnega mesta http://www.apachefriends.org/en/xampp-windows.html na svoj računalnik prenesite zadnjo različico kompleta XAMPP za Windows (Izberite možnost »Installer«).
2. XAMPP namestite v mapo *c:\xampp*. Med namestitvijo izberite nastavitve, kot jih prikazuje Slika 2.

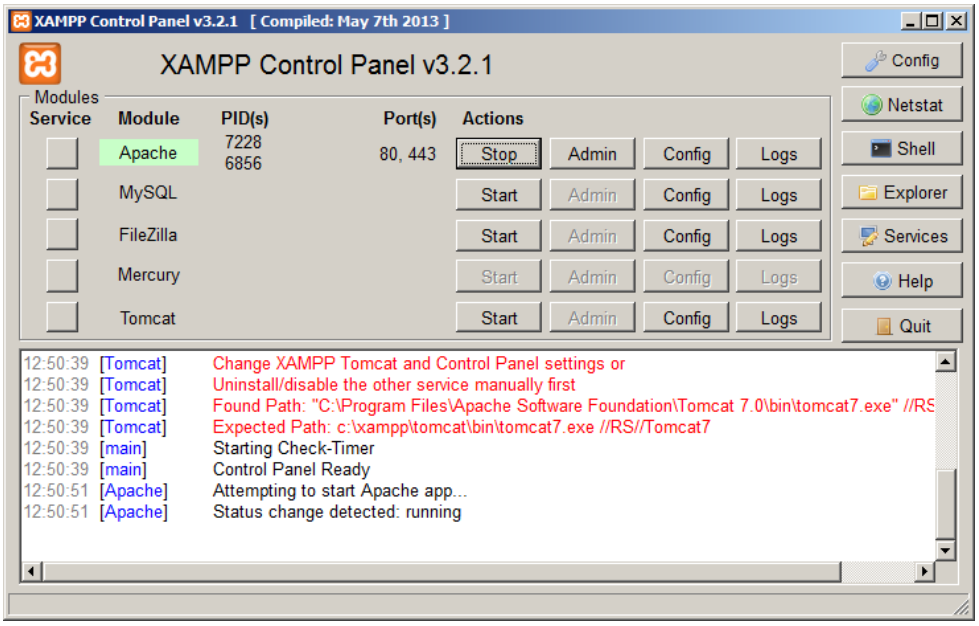


Slika 1

Nadzorna plošča

Do XAMPP-ove nadzorne plošče lahko dostopate preko ikone v opravilni vrstici, preko bližnjice na namizju ali pa tako, da zaženete program **xampp-control.exe.** Slednjega najdete v mapi, v katero ste namestili XAMPP.

1. Zaženite spletni strežnik Apache.
2. Če je zagon uspel, lahko preverite tako, da v spletni brskalnik vpišete http://localhost/. Ob uspeli namestitvi se prikaže strežnikova pozdravna stran.



Slika 2. Nadzorna plošča

Vprašanje:

Kaj označuje ključna beseda *localhost*?

Namestitev spletnih strani

Spletno stran namestimo na strežnik *apache* tako, da jo prenesemo v mapo **htdocs**. Zaradi preglednosti vsako spletno stran postavimo v ločeno mapo znotraj mape **htdocs**.

1. Z Raziskovalcem v mapi **htdocs** ustvarite novo mapo s svojim priimkom.
2. Na spletu poiščite poljubno spletno stran in jo shranite v pravkar ustvarjeno mapo.
3. V spletnem brskalniku prikažite shranjeno spletno stran tako, da posredujete lokalnemu strežniku zahtevo po spletni strani.

Pomembno je, da **do shranjene spletne strani dostopate s pomočjo zahteve, ki jo posredujete lokalnemu strežniku** (prek njegovega URL naslova v brskalniku) in ne prek datotečnega sistema (npr. s klikom na datoteko).

3. Spletni odjemalec

Spletni odjemalec (ali bolj popularno spletni brskalnik) je aplikacija, ki omogoča prikaz teksta, slik in drugih multimedijskih vsebin ter informacij, ki so vsebovane na spletnih mestih.

Najpopularnejši brskalniki so

- Internet Explorer: http://www.microsoft.com/windows/products/winfamily/ie/default.mspx
- Mozilla Firefox: http://www.mozilla.com/en-US/firefox/
- Opera: http://www.opera.com/
- Safari: http://www.apple.com/safari/
- Google Chrome: http://www.google.com/chrome/

Uporabniški vmesnik brskalnika omogoča upravljanje z:

- osnovno funkcionalnostjo brskalnika
  - osvežitev strani
  - navigacija naprej / nazaj, navigacijska vrstica
  - sledenje povezavam (v istem/novem oknu/zavihku)
  - domača stran
  - zaznamki / priljubljene strani
  - **zgodovina**
  - **prikaz izvorne HTML kode**
  - **shranjevanje spletne strani in slik**
- nastavitvami

  - domača stran
  - **varnost**
  - **zgodovina**
  - **zasebnost (piškotki)**
  - certifikati za varne povezave (uvoz certifikata)
  - samodejno izpolnjevanje obrazcev
  - nastavitev prikaza itd.
- dodatki (*add-ons, plug-ins*)
  - **Firebug** (Mozilla Firefox)
  - itd.

- Na domačem računalniku, na katerem boste izdelali spletno stran, namestite XAMPP.
- Na domačem računalniku namestite in preizkusite vsaj enega izmed brskalnikov, ki ga še ne uporabljate.

Naslednjič:

- Jezik za označevanje nadbesedila HTML. **Ponovite osnove jezika HTML** (značke, atributi, elementi, osnovni gradniki strani)!

4. Viri in literatura

- Spletna stran paketa XAMPP:
  http://www.apachefriends.org

- Porazdelitve vrhnjih domen:
  http://ftp.isc.org/www/survey/reports/current/bynum.txt

- Primerjava spletnih brskalnikov:
  http://en.wikipedia.org/wiki/Comparison_of_web_browsers

- Deleži uporabe spletnih brskalnikov:
  http://en.wikipedia.org/wiki/Usage_share_of_web_browsers

- Spletni tečaji, Wikipedia

- Prosojnice s predavanj

Vaja 2: HTML

Ponovitev:

- HTML je označevalni jezik za zapis spletnih strani.
- Osnovni gradniki jezika so HTML elementi.
- Vsebina HTML elementa je omejena z začetno in končno značko (oznako).
- Večini elementov lahko določimo atribute, ki se nahajajo v začetnih značkah.
- Osnovne značke na spletni strani so:
  <html> za začetek in **</html>** za konec HTML dokumenta
  <head> in **</head>** za začetek in konec podatkov o strani (glava)
  <body> in **</body>** za začetek in konec vsebine strani (telo)

1. Strukturiranje HTML dokumenta

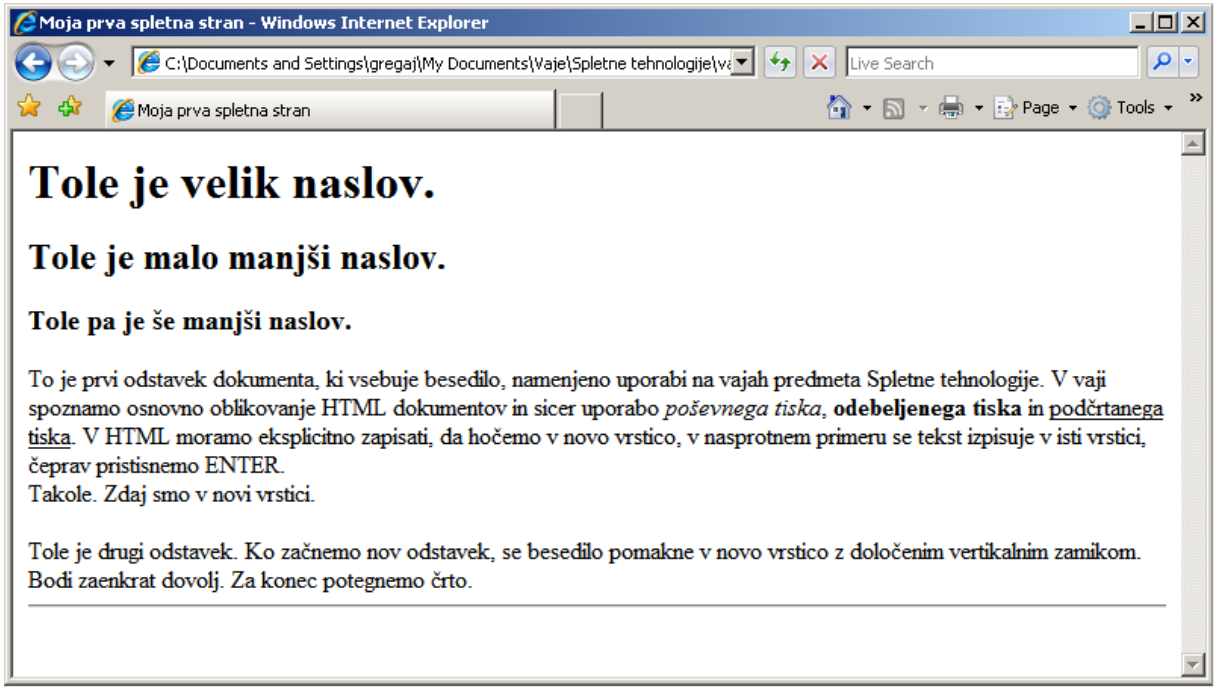
1. V beležnici (*notepad*) ustvarite nov dokument.
2. Dokument shranite **kot html dokument** pod poljubnim imenom, pri tem pa **izberite kodiranje UTF-8**.

Če želite ustvariti dokument poljubnega tipa, morate pri shranjevanju kot vrsto datoteke izbrati možnost 'Vse datoteke'. Vrsto dokumenta pri tem določa končnica datoteke, v našem primeru html.

3. V datoteki označite začetek in konec HTML dokumenta.
4. Dodajte elementa, ki določata glavo in vsebino spletne strani.
5. V glavi s pomočjo elementa **title** določite naslov spletne strani.
6. Besedilo iz datoteke http://www.lkn.fe.uni-lj.si/gradiva/ST/vaja2/tekst.txt postavite v telo spletne strani.
7. Spletno stran preoblikujte v obliko, ki jo prikazuje Slika 3. Pri tem uporabite naslednje elemente:

   h1, h2, h3, br, hr, p, b, i, u

**Vsak HTML element mora biti zaključen s končno značko. Nekateri elementi nimajo končne značke (npr. element za novo vrstico *br*). V tem primeru je začetna značka hkrati tudi končna, kar nakazujemo s poševnico ob imenu značke (
).**



Slika 3

2. *Uporaba atributov*

HTML elementi imajo lahko atribute, ki natančneje določajo njihove lastnosti. Atributi in lastnosti se vedno nahajajo v začetni znački elementa.

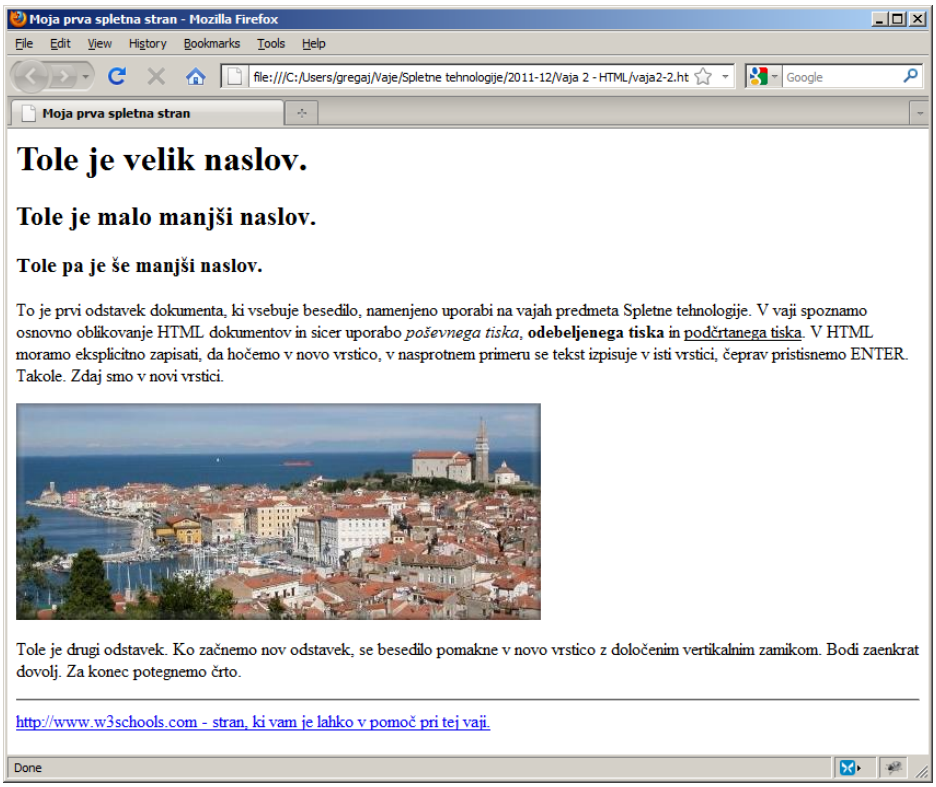
```
<img src="mojaSlika.png" alt="To je moja slika!">
```

V zgornjem primeru je *img* ime značke, *src* in *alt* pa sta atributa z vrednostma *'mojaSlika.png'* in *'To je moja slika!'*.

Z uporabo elementov **a** in **img** ter atributov **alt, href**, **src** in **target**:

1. Med prvi in drugi odstavek vrinite poljubno sliko in ji določite nadomestno besedilo. To besedilo se prikaže v primeru, če brskalnik slike ne najde.
2. Pod vodoravno črto dodajte povezavo na poljubno spletno stran. Stran naj se prikaže v novem oknu (zavihku).

Opise elementov in atributov najdete na http://www.w3schools.com/html/



Slika 4

3. *Seznami in tabele*

1. Z uporabo elementov **ul**, **ol** in **li** na stran dodajte oštevilčen in neoštevilčen seznam s poljubno vsebino, podobno kot ju prikazuje Slika 5.
2. Z uporabo elementov **table**, **tr**, **td** in **th** na stran dodajte podobno tabelo, kot jo prikazuje Slika 5, in jo napolnite s poljubno vsebino.

   Pomoč: stolpce gnezdite znotraj vrstic in ne obratno.
3. Z uporabo ustreznega atributa nastavite tabeli rob širine 1.

1. Mleko
2. Kava
3. Čaj

- Mleko
- Kava
- Čaj

Ime	Starost
Tonček	14
Marica	43
Jure	33
Mateja	21

Slika 5: Oštevilčen seznam, neoštevilčen seznam in tabela

4. Okvirji

Okvirji (*frames*) so namenjeni prikazu več HTML dokumentov znotraj istega okna brskalnika. Klasični okvirji (element **frame**) so danes že »zastareli«, zato jih prihodnje različice jezika HTML ne bodo več podpirale.

Po drugi strani pa so še vedno aktualni okvirji *iframe*, ki so namenjeni gnezdenju neke spletne strani v obstoječi strani. Prednost uporabe *iframe* okvirjev je ta, da lahko določen del strani naredimo samo enkrat (na primer meni z navigacijo), v *iframe* okvirju pa prikazujemo različne podstrani (ki jih izbiramo v meniju z navigacijo). Pogosta uporaba okvirjev *iframe* je tudi vključevanje vsebin, ki jih ponujajo razni spletni portali.

1. Na spletno stran postavite *iframe* okvir z naslednjimi lastnostmi:
   a. okvir naj ima obrobo;
   b. njegova višina in širina naj znašata 200 slikovnih elementov;
   c. omogoča naj drsenje (»*scrollanje*«);
   d. v okvirju prikažite poljubno vsebino (spletno stran).
2. Ustvarite preprosto navigacijo. Na spletno stran vključite dve povezavi. Ko uporabnik klikne na povezavo, naj se vsebina, na katero se povezava nanaša, prikaže v okvirju.
3. Na spletno stran vključite zemljevid, ki ga ponuja spletna storitev *Google Maps*. Ko na zemljevidu izberete želeno lokacijo, izberite možnost *'Link'* oziroma '*Povezava*', nato pa v svojo stran vključite kodo (*copy/paste*), ki vam jo *Google Maps* v ta namen pripravi (*Paste HTML to embed in website* oziroma *Prilepite HTML, če ga želite vdelati v spletno mesto*).
4. Na spletnem portalu *Youtube* poiščite svoj najljubši video. Pod oknom z videom kliknite na polje »*Share*« in nato »*Embed*« (»*Skupna raba*«, »*Vdelaj*«). Prikazano HTML vsebino vključite v svojo spletno stran.

1. Izdelano spletno stran namestite na lokalni strežnik in jo prikažite s pomočjo zahteve lokalnemu strežniku.

Naslednjič

Oblikovanje spletnih strani s CSS. **Ponovite osnove jezika CSS: skladnjo jezika, selektorje (za elemente, razrede, id-je, psevdo razrede, …), lastnosti, vrednosti lastnosti itd.!**

Viri in literatura

- Prosojnice s predavanj
- Spletni tečaji W3Schools: http://www.w3schools.com/html/

Vaja 3: Oblikovanje spletnih strani s CSS

CSS (*Cascading Style Sheet*) je jezik za oblikovanje elementov na spletni strani. Z uporabo jezika CSS elementom določimo oblikovne lastnosti, kot so barva, pisava, ozadje, razmiki itd. CSS je bil ustvarjen z namenom ločevanja oblike dokumenta od vsebine, ki je sicer zapisana v jeziku HTML.

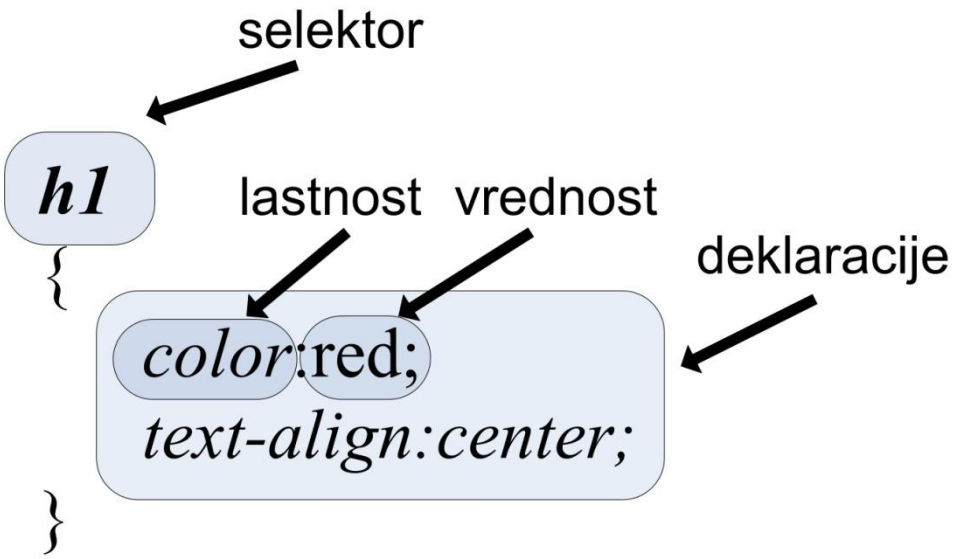
CSS stili se lahko nahajajo:

- v znački posameznega HTML elementa (t.i. *inline* stili);
- v glavi HTML datoteke;
- v ločeni CSS datoteki.

1. Povzetek CSS skladnje

CSS pravila so sestavljena iz dveh delov in sicer iz:

- selektorja in
- ene ali več deklaracij.



Slika 6: Skladnja jezika CSS

*Selektor **posameznih** elementov, ki pripadajo podanemu razredu:* ***element.razred***	*Selektor elementa s podanim id-jem:* ***#id1***
*Selektor **vseh** elementov, ki pripadajo podanemu razredu:* ***.razred***	*Uporaba psevdo razreda* ***element.razred:psevdoRazred***

Tabela 1: Primeri uporabe različnih selektorjev

- Elemente in razrede ločujemo s pikami, pred *id*-jem pa postavimo znak #.
- V nek razred lahko uvrstimo **več** elementov, *id* pa lahko dodelimo le **enemu** elementu.
- Če želimo nek stil prirediti več elementom hkrati, selektorje ločujemo z vejicami.
- Pare *lastnost-vrednost* znotraj deklaracije ločujemo s podpičji.
- Če želimo nek stil prirediti vsem elementom, ki smo jih uvrstili v nek razred, ne glede na vrsto elementa, začnemo izraz s piko, ki ji sledi ime razreda:

  .razred *{lastnost*: *vrednost}*
- HTML koda za uvrstitev elementa v razred in dodelitev unikatnega id-ja:

  <*nek_HTML_element* ***class="mojRazred"***>

  <*nek_HTML_element* ***id="idElementa"***>
- HTML dokument povežemo s pripadajočo CSS datoteko, tako da **v glavo HTML dokumenta** dodamo sledečo vrstico:

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

2. Osnovno oblikovanje strani

Tabela 2 vsebuje lastnosti, vrednosti in utežne enote, ki vam bodo v pomoč pri spodnjih nalogah.

lastnosti	*background-color, border-style, font-family, font-style, list-style-type, text-decoration, text-indent*
vrednosti	*10, brown, lightGrey, comic sans ms, underline, italic, circle, dotted*
utežne enote	*px*

Tabela 2: lastnosti, vrednosti in utežne enote

1. Prenesite na svoj računalnik HTML dokument http://www.lkn.fe.uni-lj.si/gradiva/ST/vaja3/vsebina.html in sliko, ki je z dokumentom povezana.
2. V urejevalniku teksta ustvarite nov dokument, ga po želji poimenujte in ga shranite kot CSS dokument. Za kodiranje izberite UTF-8.
3. V glavo HTML dokumenta vključite referenco na CSS skripto, v kateri boste oblikovali stil dokumenta.
4. V CSS skripti določite barvo ozadja HTML dokumenta na svetlo sivo. To storite tako, da elementu **body** nastavite lastnost **background-color** na ustrezno vrednost.
5. Nastavite obliko pisave v celotnem dokumentu na *comic sans ms*.

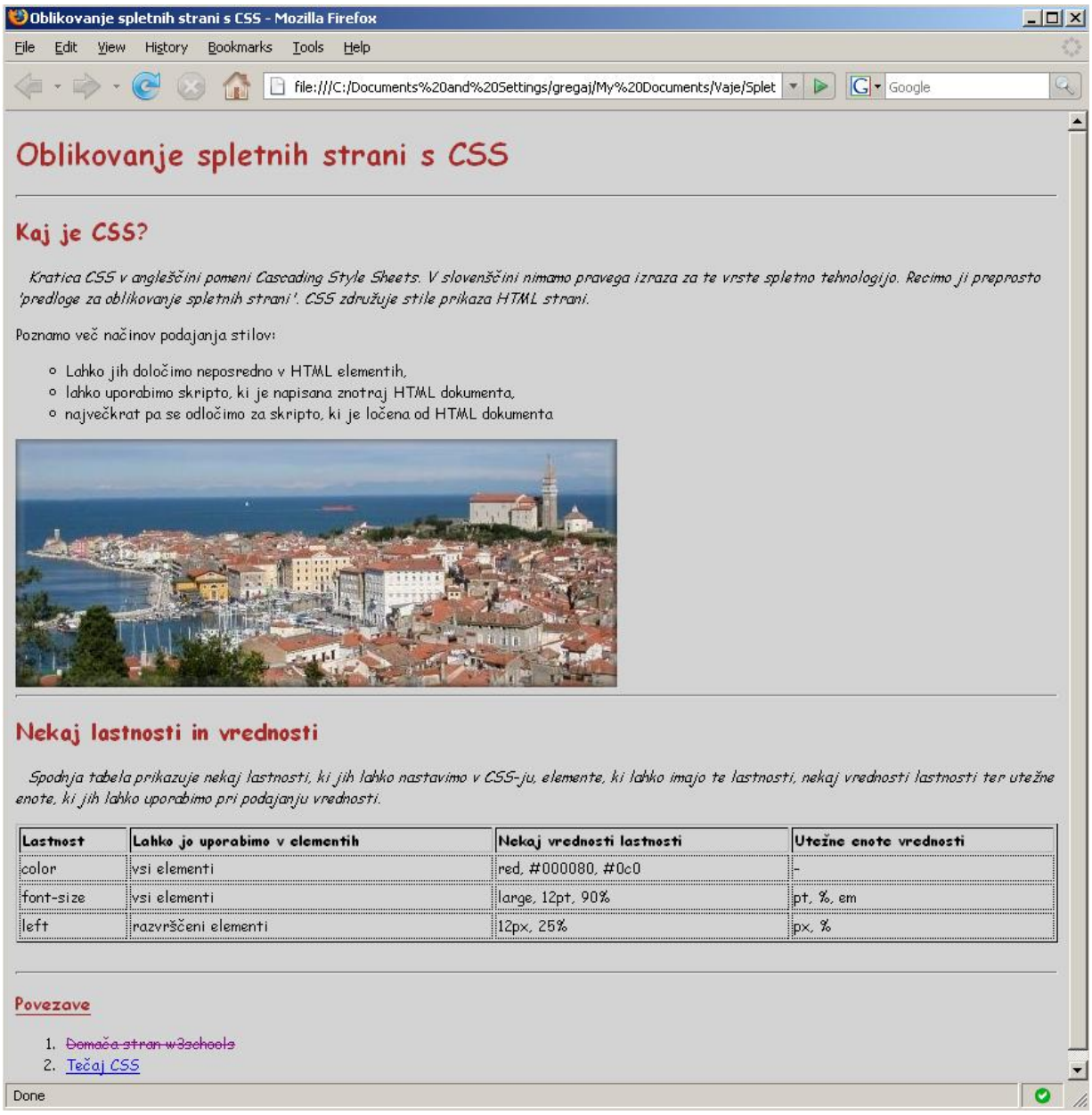
Kadar je vrednost lastnosti sestavljena iz več besed (torej vsebuje presledek), jo zapišite v dvojnih narekovajih (npr. *"comic sans ms"*).

6. Vsi naslovi (**h1, h2** in **h3**) naj se prikazujejo v rjavi barvi. Uporabite CSS skladnjo, ki vsem omenjenim elementom stil priredi v enem koraku (z enim samim selektorjem).
7. Vsem naslovom **h3** nastavite, da se bo tekst v njih prikazoval podčrtano.
8. Popravite stil neoštevilčenega seznama, tako da se bo namesto privzetega znaka za novo postavko (običajno ●) prikazoval nepobarvan krog (○).
9. Vse notranje obrobe tabele naj bodo namesto s polno črto prikazane s pikami (……). Prikaz zunanjih obrob tabele in obrob glave tabele naj ostane nespremenjen.
10. V vsakem »poglavju« naj bo prva vrstica prvega odstavka zamaknjena za 10 slikovnih elementov, odstavek pa prikazan v poševnem tisku.

Namig: uvrstite vsak prvi odstavek poglavja v razred (npr. »prviOdstavek«), nato pa temu razredu v CSS določite ustrezen stil.

Ne uporabljajte presledkov med vrednostjo in enoto (npr. 20px in ne 20 px), saj nekateri brskalniki takega zapisa ne razumejo.

Ne začenjajte imena razreda s številom, saj nekateri brskalniki takega zapisa ne razumejo.



Slika 7: Rezultat oblikovanja s CSS

3. Uporaba psevdo razredov

Psevdo razredi so že pripravljeni razredi, ki omogočajo dodajanje »posebnih efektov« elementom. Od navadnih razredov se razlikujejo po tem, da je njihovo prirejanje dinamično. Nek element lahko psevdo razred »pridobi« ali »izgubi« glede na uporabnikovo interakcijo s spletnim dokumentom. Najbolj pogosto uporabljeni psevdo razredi so: ***link***, ***visited***, ***active*** in ***hover***.

1. Ugotovite, kakšen je učinek uporabe štirih zgoraj omenjenih psevdo-razredov.
2. Nastavite, da vsakič, ko gre uporabnik z miško čez sliko na spletni strani, slika do določene mere zbledi (bledenje nastavite z lastnostjo ***opacity***).
3. Nastavite, da se tekst v povezavah prečrta, ko so povezave že obiskane.

*4. Napredno oblikovanje**

1. Z uporabo selektorjev za določanje relacij med elementi nastavite, da se vse povezave, ki se pojavljajo kjerkoli znotraj oštevilčenega seznama (torej so »potomci« elementa, ki predstavlja oštevilčen seznam), prikazujejo rdeče.
2. Z uporabo psevdo-elementov v CSS nastavite, da se vsaka prva črka v odstavku prikaže povečana (npr. 16 px), odebeljena in v rdeči barvi.

5. Naslednjič

Razvrščanje elementov s pomočjo CSS. **Ponovite področja HTML elementa** (padding, margin, border, vsebina itd.) **in načine določanja položaja elementov** (static, relative, absolute itd.).

6. Literatura

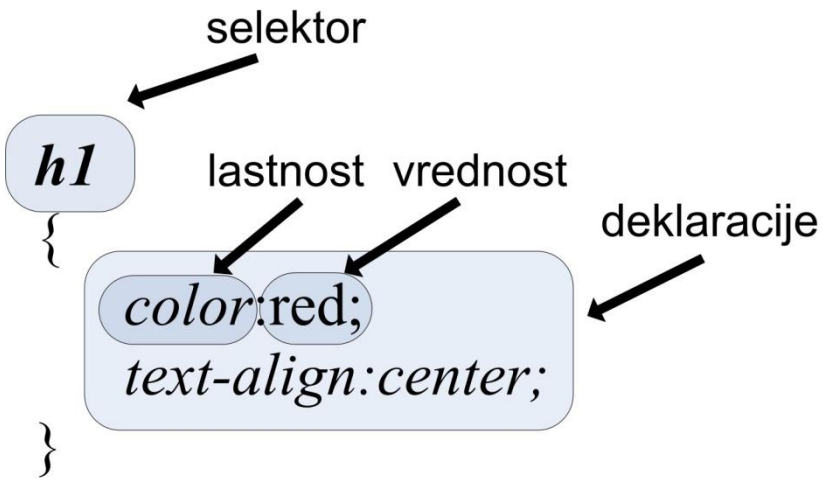
- Spletni tečaji:
  - http://www.w3schools.org/
  - http://www.htmldog.com/
- Primer uporabe več stilov pri prikazu iste vsebine
  - http://www.w3schools.com/css/demo_default.htm
- Seznam CSS lastnosti:
  - http://www.w3schools.com/css/css_reference.asp
- Prosojnice s predavanj

Vaja 4: Oblikovanje spletnih strani s CSS (2)

1. *Povzetek CSS skladnje*

CSS pravila so sestavljena iz dveh delov in sicer iz:

- selektorja in
- ene ali več deklaracij.



Slika 8: Skladnja jezika CSS

*Selektor **posameznih** elementov, ki pripadajo podanemu razredu:* ***element.razred***	*Selektor elementa s podanim id-jem:* ***#id1***
*Selektor **vseh** elementov, ki pripadajo podanemu razredu:* ***.razred***	*Uporaba psevdo razreda* ***element.razred:psevdoRazred***

Tabela 3: Primeri uporabe različnih selektorjev

- Elemente in razrede ločujemo s pikami, pred *id*-jem pa postavimo znak **#**.
- V nek razred lahko uvrstimo **več** elementov, *id* pa lahko dodelimo le **enemu** elementu.
- Če želimo nek stil prirediti več elementom hkrati, selektorje ločujemo z vejicami.
- Pare *lastnost-vrednost* znotraj deklaracije ločujemo s podpičji.
- Če želimo nek stil prirediti vsem elementom, ki smo jih uvrstili v nek razred, ne glede na vrsto elementa, začnemo izraz s piko, ki ji sledi ime razreda:

  .razred *{lastnost: vrednost}*

- HTML koda za uvrstitev elementa v razred in dodelitev unikatnega id-ja:

  <*nek_HTML_element* ***class="mojRazred"***>

  <*nek_HTML_element* ***id="idElementa"***>

- HTML dokument povežemo s pripadajočo CSS datoteko, tako da **v glavo HTML dokumenta** dodamo sledečo vrstico:

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

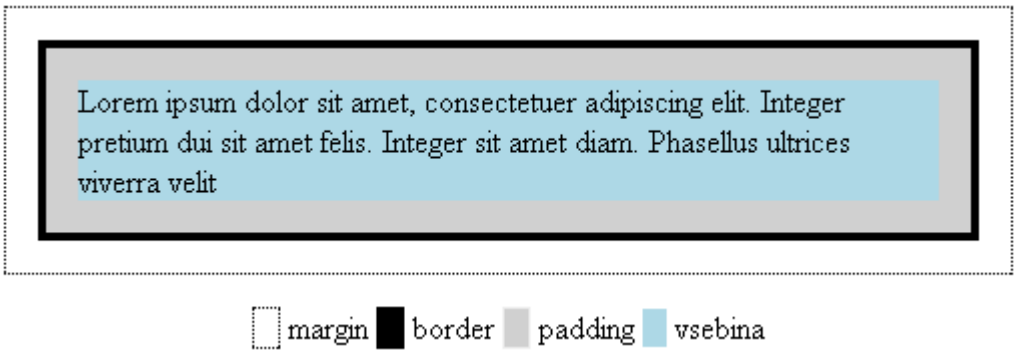
2. *Element div in področja elementov*

Element ***div*** je »splošen« element, ki služi združevanju HTML elementov z namenom njihovega razvrščanja po spletni strani. Element ***div*** lahko podobno kot ostale elemente uvrstimo v nek razred oziroma mu dodelimo *id*. Za razliko od ostalih elementov elementa ***div*** spletni brskalnik ne zna prikazati, če ne podamo informacij o njegovem prikazu (npr. v jeziku CSS).

Vsak element v HTML (tudi ***div***) je obravnavan kot pravokotnik z vsebino, ki jo obdaja več področij (glej Sliko 1):

- **padding** označuje prostor med robom elementa in njegovo vsebino. Področje ima lahko obarvano ozadje.
- **border** predstavlja rob elementa;
- **margin** označuje prazen prostor okoli elementa (na zunanji strani roba elementa). Področje **margin** nima obarvanega ozadja in je popolnoma prozorno.

Velikost in obliko omenjenih področij definiramo z uporabo istoimenskih lastnosti ***padding***, ***border*** in ***margin.*** Vrednosti, ki jih priredimo tem lastnostim, veljajo hkrati za vse štiri stranice elementa. Če želimo neko lastnost nastaviti samo na eni stranici, potem imenu lastnosti dodamo pomišljaj (-) in navedemo želeno stranico (npr. ***padding-left***).



Slika 9: Področja HTML elementa

1. S spletnega mesta http://www.lkn.fe.uni-lj.si/gradiva/ST/vaja4/ prenesite na svoj računalnik dokumenta *vsebina.html* in *stil.css*.
2. Dokument *vsebina.html* vsebuje tri elemente ***div***: element z *id* ***zunanjiDiv*** je nadrejen elementoma ***notranjiDiv1*** in ***notranjiDiv2***.

   Stil HTML dokumenta je določen v datoteki *stil.css*. V tej datoteki so vsem omenjenim ***div*** elementom že nastavljene nekatere oblikovne lastnosti.

   V nadaljevanju CSS datoteko dopolnite, tako da boste dosegli, kar od vas zahtevajo naloge.

CSS lastnosti in njihove vrednosti, ki jih boste potrebovali, poiščite na spletni strani http://www.w3schools.com/Css/.

1. Elementu z *id* ***notranjiDiv1*** s pomočjo CSS skrijte (odrežite) besedilo, ki se prikazuje čez področje elementa.
2. Elementu z *id* ***notranjiDiv2*** nastavite, da se lahko po besedilu, ki se sicer izpisuje čez področje elementa, pomikamo s pomočjo drsne vrstice (*scroll bar*) znotraj elementa.
3. Besedilo v elementu z *id* ***notranjiDiv1*** naj bo odmaknjeno 10 slikovnih elementov od vseh robov. Opazujte, kaj se pri tem dogaja z dimenzijami omenjenega elementa.
4. Vzorec obrobe elementov z *id* ***notranjiDiv1*** in ***notranjiDiv2*** naj bo prekinjena črta (*dashed*).
5. Element z *id* ***notranjiDiv2*** naj bo
   - na levi strani odmaknjen od ostalih elementov za 10 slikovnih elementov,
   - na zgornji strani odmaknjen od ostalih elementov za 20 slikovnih elementov.

3. Prikazovanje elementov

1. Ko se z miškinim kazalcem nahajate na področju elementa z *id* ***notranjiDiv1,*** naj se spremeni oblika kazalca v poljubno obliko.
2. Ko se z miškinim kazalcem ustavite na področju elementa z *id* ***notranjiDiv2***, naj ta element izgine (postane neviden).

4. Razvrščanje elementov

Postavitev elementa na spletni strani določimo z naslednjimi vrednostmi lastnosti ***position***:

- ***static*** postavi element na privzeto mesto – na tisto mesto na spletni strani, kjer bi moral biti prikazan glede na strukturo HTML dokumenta.
- ***relative*** postavi element relativno glede na privzeto mesto. Določiti moramo odmik od privzetega mesta.
- ***absolute*** postavi element relativno glede na prvi nadrejeni element, ki ni postavljen na privzeto mesto (torej ni ***static***)
- ***fixed*** postavi element relativno glede na okno brskalnika.

1. Pomaknite element z *id* ***notranjiDiv2*** 20 slikovnih elementov proti desni glede na njegovo privzeto postavitev.
2. Pomaknite levi zgornji rob elementa z *id* ***notranjiDiv1*** 300 slikovnih elementov od gornjega roba in 300 od desnega roba nadrejenega elementa. Kaj se pri tem dogaja z elementom z *id* ***notranjiDiv2*** ?
3. Nastavite, da se ***notranjiDiv2*** vedno izrisuje 100 slikovnih elementov od zgornjega in 20 elementov od desnega roba okna brskalnika. Rezultat lahko preverite tako, da toliko zmanjšate okno brskalnika, da se prikažeta oba drsnika, s katerima se pomikate po dokumentu.

1. Nastavite, da bo element ***zunanjiDiv*** (po širini) poravnan na sredini spletne strani.

Namig: Takšno poravnavo določite z uporabo lastnosti margin.

5. Prilagoditev prikaza mediju

Včasih je obliko strani potrebno prilagoditi za prikaz na različnih medijih. CSS to omogoča z uporabo pravila ***@media***.

1. Nastavite, da se pri tiskanju dokumenta vsebina prikaže v drugačni pisavi kot pri prikazu v brskalniku.
2. Nastavite, da pri tiskanju dokumenta element ***notranjiDiv2*** ni viden.

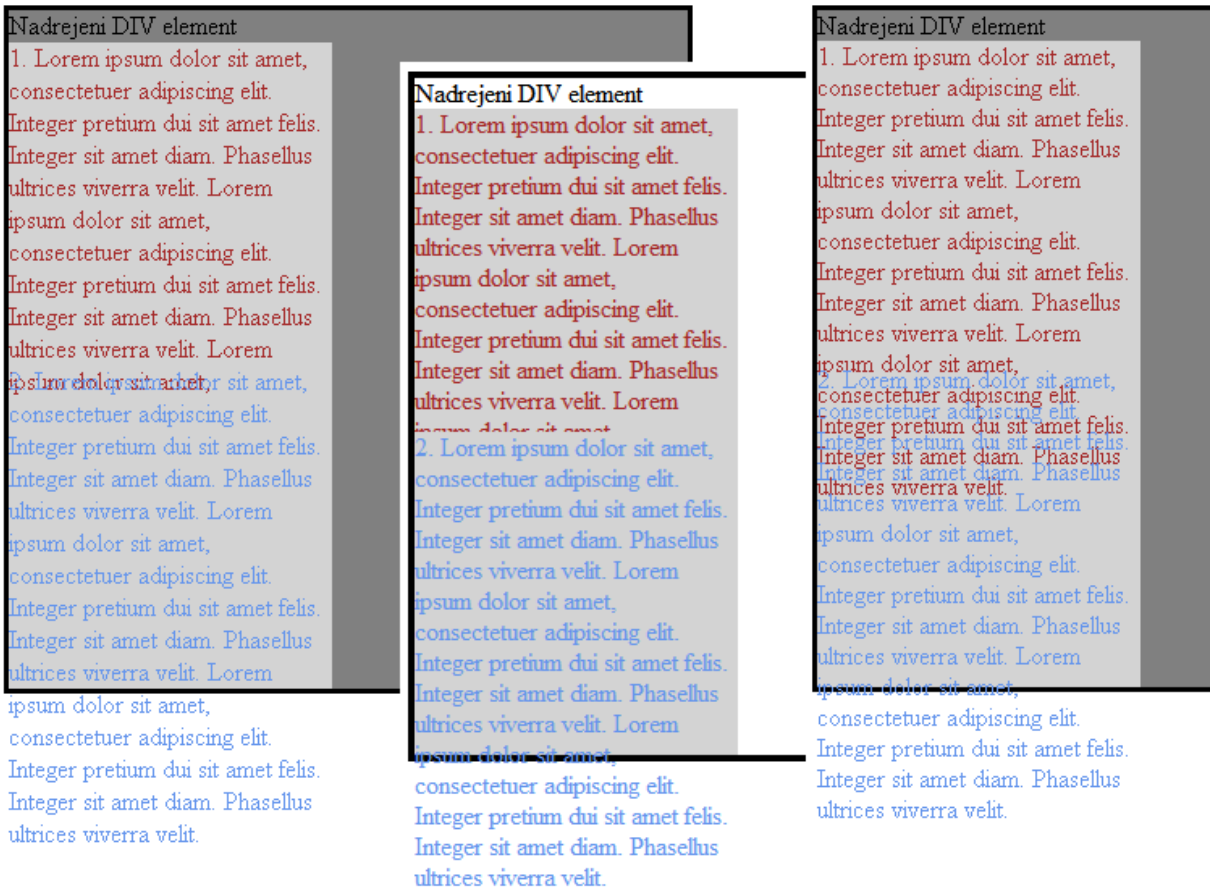
Namig: Obliko strani pri tiskanju lahko v brskalniku preverite, če izberete možnost 'Predogled tiskanja'.

6. Domača naloga

1. Preučite delovanje CSS lastnosti ***float*** in ***clear***.
2. Preučite ***block*** in ***inline*** prikaza elementov ter preklapljanje med njima.

7. Za konec…

Oblikovanje strani s CSS je lahko zelo mučno opravilo, saj vsi brskalniki ne podpirajo vseh lastnosti, čeprav so te standardizirane. Poleg tega si različni brskalniki posamezne lastnosti različno razlagajo. Spodnja slika tako prikazuje dokument *vsebina.html* v brskalnikih *Mozilla Firefox 2*, *Internet Explorer 7.0* in *Opera 9.50*. Lepa oblika dokumenta je velikokrat odvisna od izkušenj razvijalca, sprotnega »poizkušanja«, pogosto pa je potrebno predvideti pomanjkljivosti posameznih brskalnikov in za vsak podprt brskalnik uporabiti kodo, ki jo brskalnik preizkušeno razume.



Slika 10: Prikaz istega elementa v brskalnikih Mozilla Firefox 2, Internet Explorer 7.0 in Opera 9.50

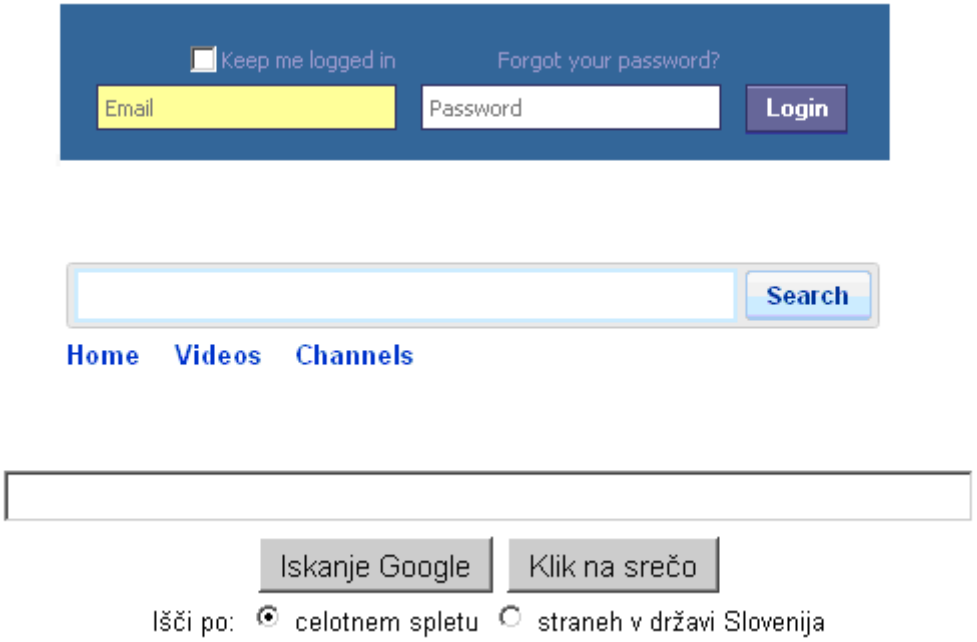
8. *Literatura*

- Spletni tečaji:
  - http://www.w3schools.org/css/
  - http://www.htmldog.com/
- Primer uporabe več stilov pri prikazu iste vsebine
  http://www.w3schools.com/css/demo_default.htm
- Seznam CSS lastnosti:
  http://www.w3schools.com/css/css_reference.asp
- Prosojnice s predavanj

Vaja 5 : HTML obrazci

1. Element form

HTML obrazci so namenjeni zbiranju uporabniških podatkov in njihov prenos med spletnimi mesti. HTML obrazec označuje element **form**.



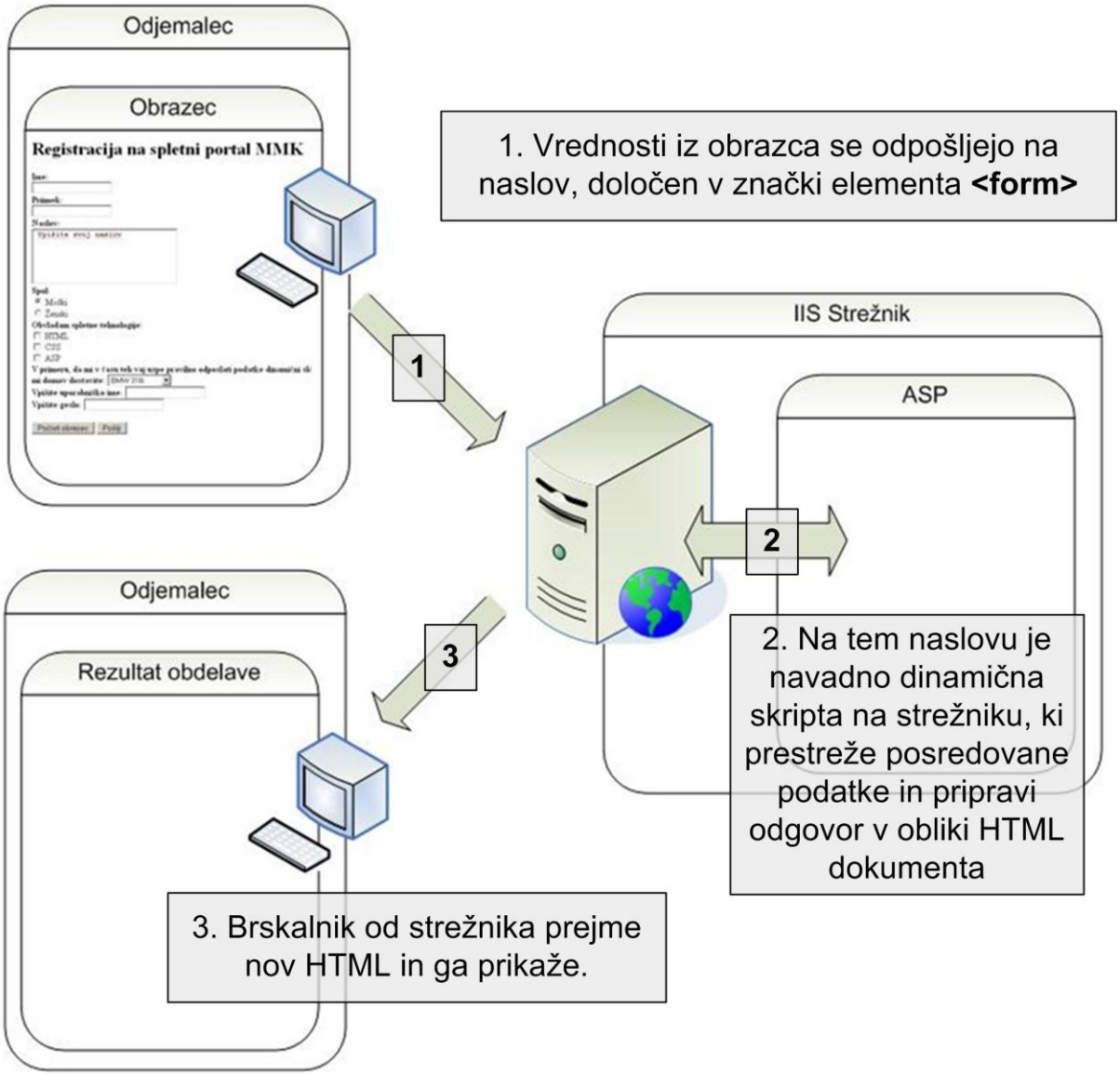
Slika 11: Primer obrazcev na znanih spletnih portalih: Facebook, Youtube in Google

Najpomembnejša atributa elementa **form** sta **method** in **action**.

- **method** določa način prenosa podatkov, ponavadi z eno izmed vrednosti:
  - GET (privzeta vrednost) - podatki se pošljejo na strežnik pripeti na URL naslov, zato so uporabniku vidni, ali
  - POST - podatki se pošljejo v telesu HTTP zahteve, uporabnik »jih ne vidi«.
- **action** določa naslov dinamične skripte ali aplikacije na strežniku, kamor se pošljejo podatki iz obrazca.

Uporabnik obrazec odpošlje dinamični skripti ali programu na strežniku, ki podatke prebere in pripravi odgovor (Slika 2). Glede na Sliko 1 tako na primer:

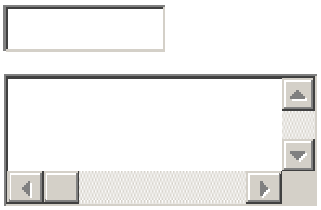
- Facebookov strežnik na podlagi uporabniškega imena in gesla vrne osebni profil,
- Youtubeov strežnik na podlagi iskalnega niza vrne rezultat iskanja po video vsebinah,
- Googlov strežnik na podlagi iskalnega niza in ostalih izbranih možnosti vrne rezultate iskanja po spletu.



Slika 2: Princip obrazcev

2. Gradniki obrazca

Obrazec je sestavljen iz elementov, ki omogočajo:

• vnos teksta: o element **input** tipa **text** o element **textarea**	
• izbiro med več možnostmi: o element **input** tipov **checkbox** in **radio** o element **select**	
• neko dejanje: o element **button** o element **input** tipa **reset**	
• vnos gesla: o element **input** tipa **password**	
• odpošiljanje obrazca o element **input** tipa **submit**	
• pošiljanje skritih podatkov: o element **input** tipa **hidden**	

- Elementom obrazca z uporabo HTML atributov lahko nastavimo parametre, kot so ime, tip, dolžina vnosnega polja itd.

- Obrazec mora vsebovati gumb, ki sproži odpošiljanje obrazca:

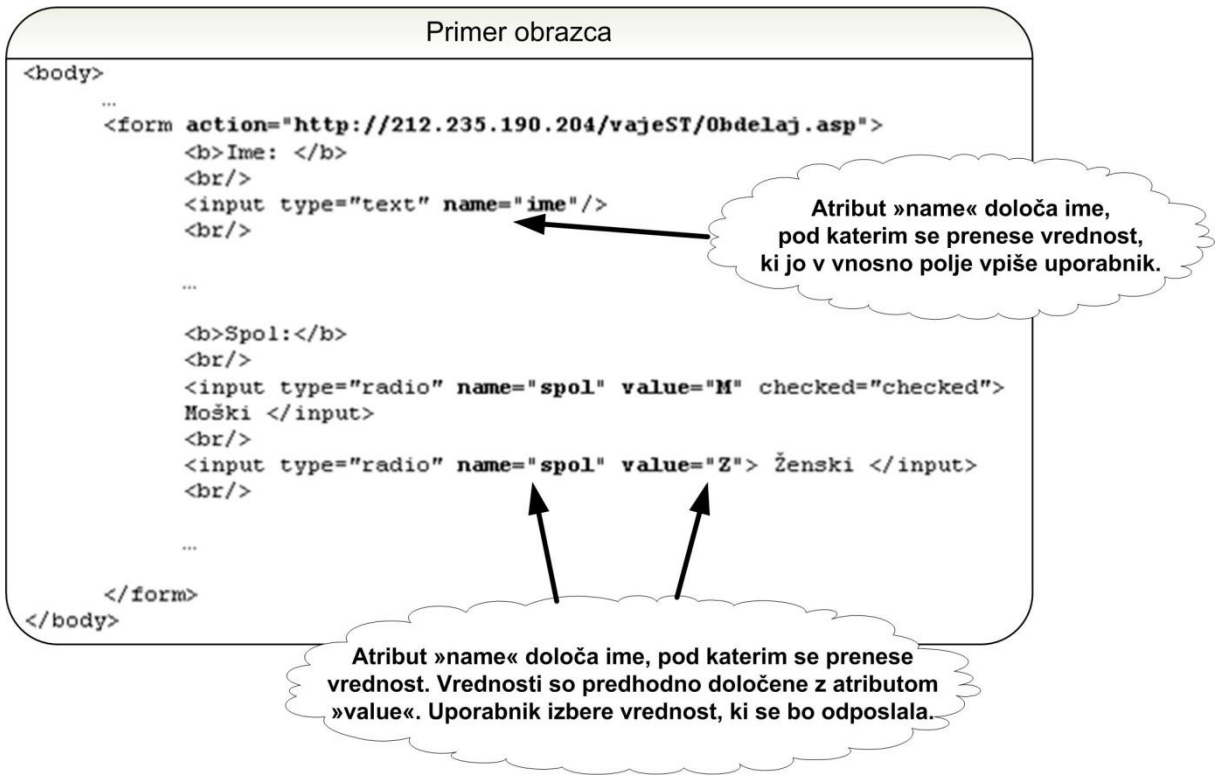
<input type="submit" value="napis_na_gumbu"/>

3. Poimenovanje gradnikov obrazca in pošiljanje podatkov strežniku

Vsak gradnik obrazca je potrebno poimenovati z uporabo atributa **name**. Poimenovanje je pomembno, saj damo z njim podatkom »ime«. Podatki se bodo namreč odposlali strežniku, ki jih mora znati ločevati. Tako je na primer primerno, če element, ki služi zbiranju podatkov o starosti, poimenujemo »starost«, »age« itd.

Podatki iz obrazcev se odpošiljajo v obliki »ime – vrednost«:

- »ime« daje podatkom pomen, posameznemu gradniku pa ga določimo z atributom **name**,
- pri vrednostih pa imamo dve možnosti:
  - vrednost določi uporabnik sam – vpiše jo na primer v vnosno polje (glej zgornji del Slike 3). Ko se obrazec odpošlje, se strežniku pošlje trenutni vpis v polju.
  - vrednost je že predhodno določena z atributom **value**. Uporabnik izbira med vnaprej pripravljenimi možnostmi (npr. pri gradnikih *checkbox*, *radio*, *select*). Vsaka izmed možnosti ima vnaprej določeno neko vrednost (v gradniku *radio* na spodnjem delu Slike 3 sta na primer pripravljeni vrednosti »M« in »Z«, ki se odpošljeta, če uporabnik izbere moški ali ženski spol).



Slika 3: V obrazec lahko vključimo gradnik, v katerega uporabnik vpiše sam poljubno vrednost (vnosno polje na zgornjem delu slike), ali pa gradnik, ki ima že nastavljene vrednosti, med katerimi uporabnik izbere želeno (radio gumb na spodnjem delu slike).

1. Ustvarite novo spletno stran in vanjo vključite obrazec. Podatki iz obrazca naj se prenašajo na spletno mesto http://www.lkn.fe.uni-lj.si/gradiva/ST/vaja5/obdelaj.asp **pripeti na URL**.

Za potrebe današnje vaje bomo pošiljali podatke iz obrazca pripete na URL. V praksi tega navadno ne počnemo, ampak raje uporabimo metodo POST, ki omogoča pošiljanje podatkov v telesu HTTP zahteve.

2. Obrazcu dodajte gradnike, tako da bo podoben tistemu, ki ga prikazuje Slika 4. Gradnike poimenujte v skladu s spodnjo tabelo.

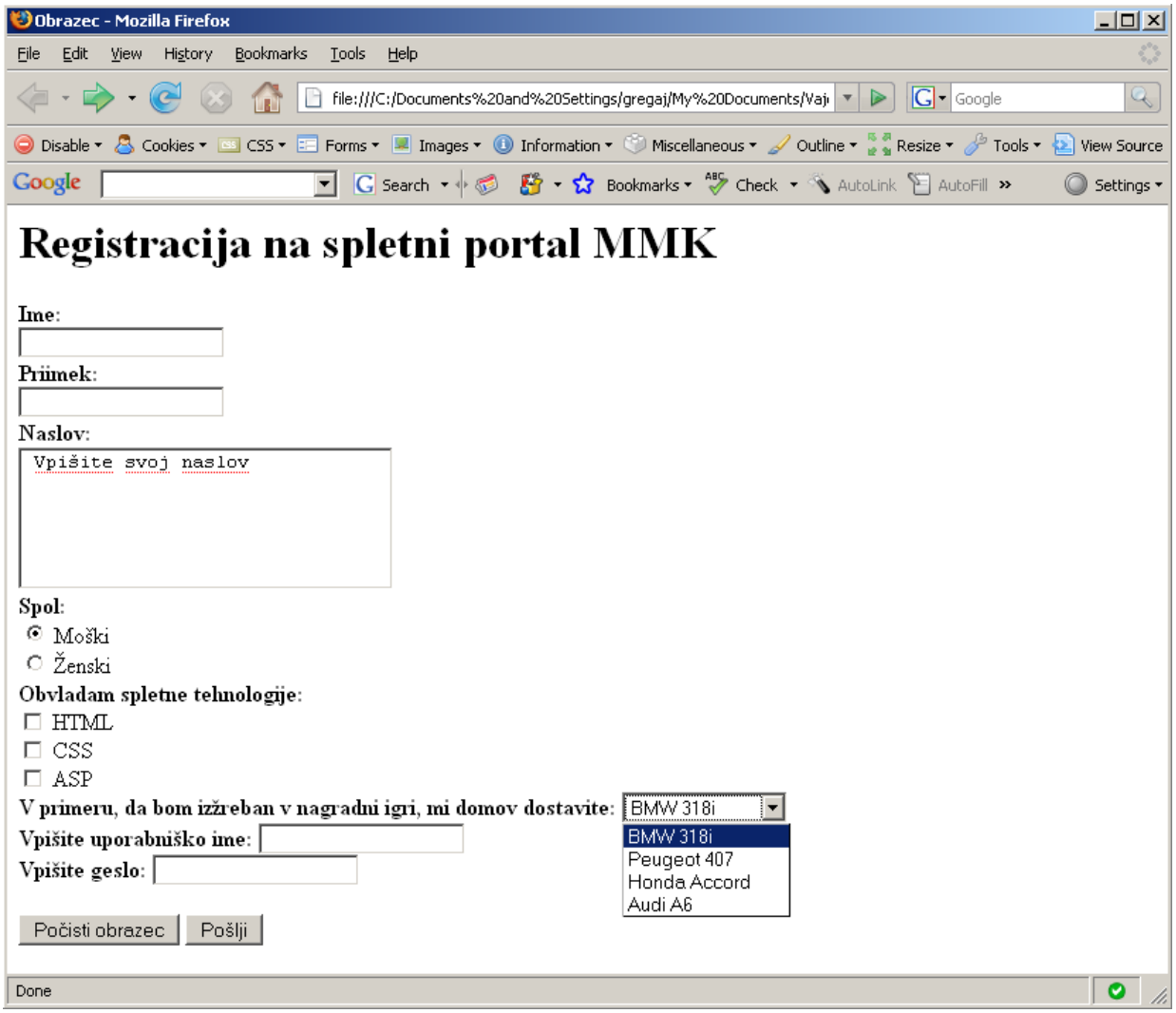
Ime	ime
Priimek	priimek
Naslov	naslov
Uporabniško ime	upIme
Geslo	geslo
Spol	spol
Poznavanje spletnih tehnologij	poznavanjeST
Izbrani avtomobil	avtomobili
Skrito sporočilo	skrito

Tabela 4: Imena podatkov, ki jih pričakuje strežniška skripta

Skripta na strežniku pričakuje podatke, ki so poimenovani v skladu z gornjo tabelo. Če ne uporabite imen iz tabele, bo skripta javila napako.

Čeprav uporabnik pri vnosu gesla ne bo videl znakov, ki jih vpisuje v temu namenjeno polje, pa je geslo mogoče enostavno razbrati iz prestrežene komunikacije (npr. z Wiresharkom). Pri pošiljanju občutljivih podatkov zato v praksi te vedno ustrezno kodiramo.

3. Pri elementih, kjer uporabnik izbira med več možnostmi, moramo poleg imena elementa, pripraviti tudi vrednosti, ki se bodo ob izbiri neke možnosti odposlale:
   - Če uporabnik izbere ženski spol, naj bo poslana vrednost "*Z*", v nasprotnem primeru pa vrednost "*M*". Ko se stran naloži, mora biti ena izmed obeh možnosti v obrazcu že izbrana (glej Sliko 4).
   - Pri izbiri poznavanja spletnih tehnologij, naj se prenesejo vrednosti "*HTML*", *CSS* in/ali *ASP* (odvisno od uporabnikove izbire).
   - Vrednosti pri izbiri avtomobila naj bodo "*BMW*", "*Peugeot*", "*Audi*" ali "*Honda*" (odvisno od uporabnikove izbire).



Slika 4: Končni rezultat vaje

4. Preizkusite delovanje obrazca.
    - Ob kliku na gumb '*Počisti obrazec*' se morajo vrednosti v vseh elementih vrniti v prvotno stanje.
    - Ob kliku na gumb '*Pošlji*' se podatki odpošljejo pripravljeni skripti na strežniku. Rezultat izvajanja skripte bo HTML stran, ki se bo prikazala v vašem brskalniku. Na strani bodo tudi opisane morebitne težave, na katere je skripta naletela pri prebiranju podatkov, ki ste jih poslali. V primeru težav, preglejte elemente, ki jih skripta navaja, in popravite napake.

Naslednjič:

Dinamične PHP skripte.

Sami boste napisali PHP skripto, ki sprejema podatke iz danes izdelanega obrazca, in pripravlja odgovor.

Ponovite PHP: osnovno skladnjo, deklaracijo spremenljivk, inicializacijo spremenljivk, kontrolne stavke, branje podatkov iz obrazca, pripravo odgovora.

Ker je PHP strežniška tehnologija, bomo potrebovali spletni strežnik, zato **ponovite nastavljanje in delovanje paketa XAMPP (Vaja 1).**

Literatura

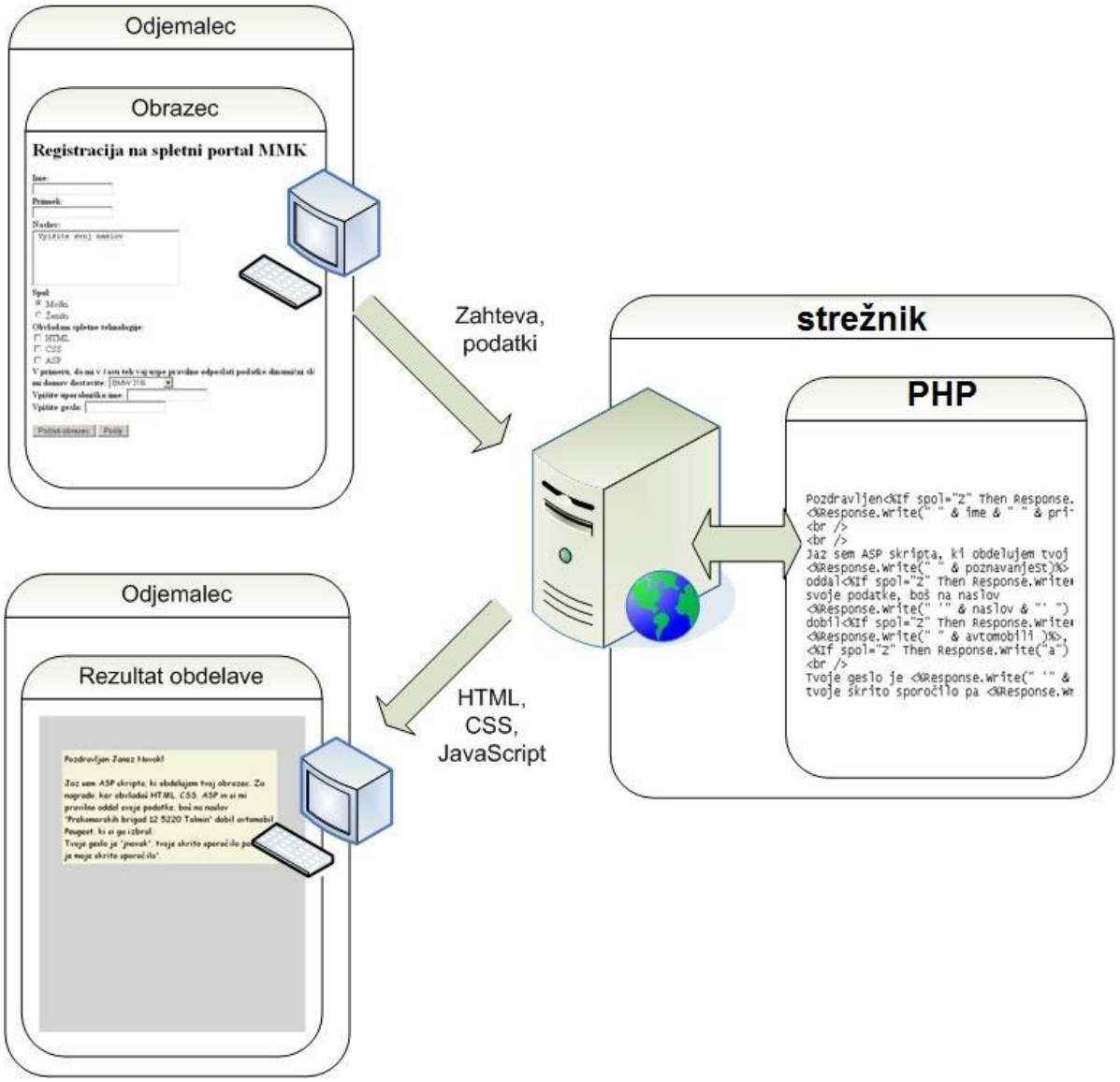
- Spletni tečaji W3Schools: http://www.w3schools.com/html/
- »Forms in HTML documents«, W3C Priporočila, http://www.w3.org/TR/html401/interact/forms.html

Vaja 6: Izdelava dinamičnih spletnih strani s PHP

PHP je tehnologija za dinamično tvorjenje spletnih strani na strežniku. PHP dokument je sestavljen iz

- HTML elementov (statični del) in
- PHP skript (dinamični del).

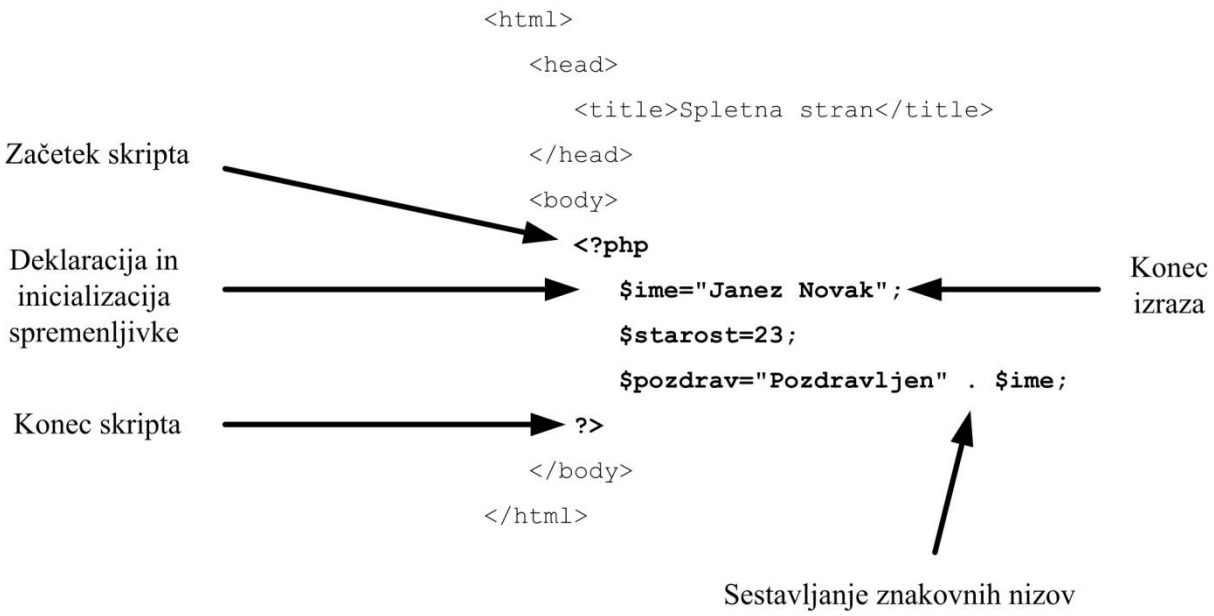
Ko odjemalec poda zahtevo po PHP dokumentu, se na strežniku izvrši dinamični del PHP dokumenta. Rezultat njegovega izvajanja je HTML, ki se doda preostalemu (statičnemu) HTML in odpošlje brskalniku kot odgovor na njegovo zahtevo.



Slika 12: Princip spletnega odjemalca in strežnika.

Programiranje v PHP

- Izraza **<?php** in **?>** ločujeta programsko kodo od HTML.
- Ime spremenljivke se začne z znakom **$**.
- Izraz je potrebno zaključiti s podpičjem (**;**)
- Spremenljivki priredimo vrednost z enačajem (**=)**
- Vrednosti v if stavkih primerjamo z dvema enačajema (==)
- Znakovne nize pišemo v dvojnih narekovajih (")
- Znakovne nize sestavljamo z uporabo pike (**.**)



Slika 13: Osnove PHP.

deklaracija funkcije →
```
function imeFunkcije()
{
   //programska koda
}
```

klic funkcije →
```
imeFunkcije();
```

if stavek →
```
if (pogoj)
{
   //programska koda, ki se
   //izvrši, če je pogoj resničen
}
else
{
   //programska koda, ki se
   //izvrši, če pogoj ni resničen
}
```

primer if stavka →
```
if(niz=="Janez")
{
   echo "Janez";
}
```

Slika 14: Osnove PHP.

1. Zaženite lokalni strežnik Apache.
2. Na spletnem mestu http://www.lkn.fe.uni-lj.si/gradiva/ST/vaja6/ najdete

- datoteko s HTML obrazcem

- predlogo PHP dokumenta.

Oba dokumenta prenesite v mapo s svojim priimkom na ustrezno mesto v datotečnem sistemu, tako da bosta dostopna preko lokalnega strežnika.

3. Obrazec v HTML dokumentu nastavite tako, da se bodo podatki iz njega pošiljali na lokalni strežnik PHP datoteki.
4. V PHP dokumentu deklarirajte spremenljivke za vse podatke iz obrazca (ime, priimek, …).

Prebiranje podatkov iz obrazca in pošiljanje odgovora odjemalcu

V PHP preberemo prejete podatke iz obrazca s pomočjo spremenljivk:

- **$_GET["ime_podatka"]**, če prebiramo vrednosti, odposlane preko metode GET in
- **$_POST["ime_podatka"]**, če prebiramo vrednosti, odposlane preko metode POST.
- **ime_podatka** določimo z atributom **name** v gradniku obrazca, ki služi pridobivanju podatka

Odgovor odjemalcu pripravimo s pomočjo funkcije **echo**:

echo "niz znakov"; ali **echo spremenljivka;**

1. Deklariranim spremenljivkam v PHP dokumentu priredite vrednosti, ki so bile poslane skripti iz obrazca.
2. Vrednosti spremenljivk izpišite v odgovoru brskalniku.

Imena podatkov, ki jih prebirate, morajo ustrezati imenom gradnikov obrazca, ki služijo za vpisovanje teh podatkov!

HTML dokument v brskalniku zahtevajte prek strežnika in ne preko datotečnega sistema (to počnete tudi pri brskanju po spletu)! V tem primeru lahko uporabite relativno naslavljanje PHP dokumenta.

Preverjanje podatkov na strežniku

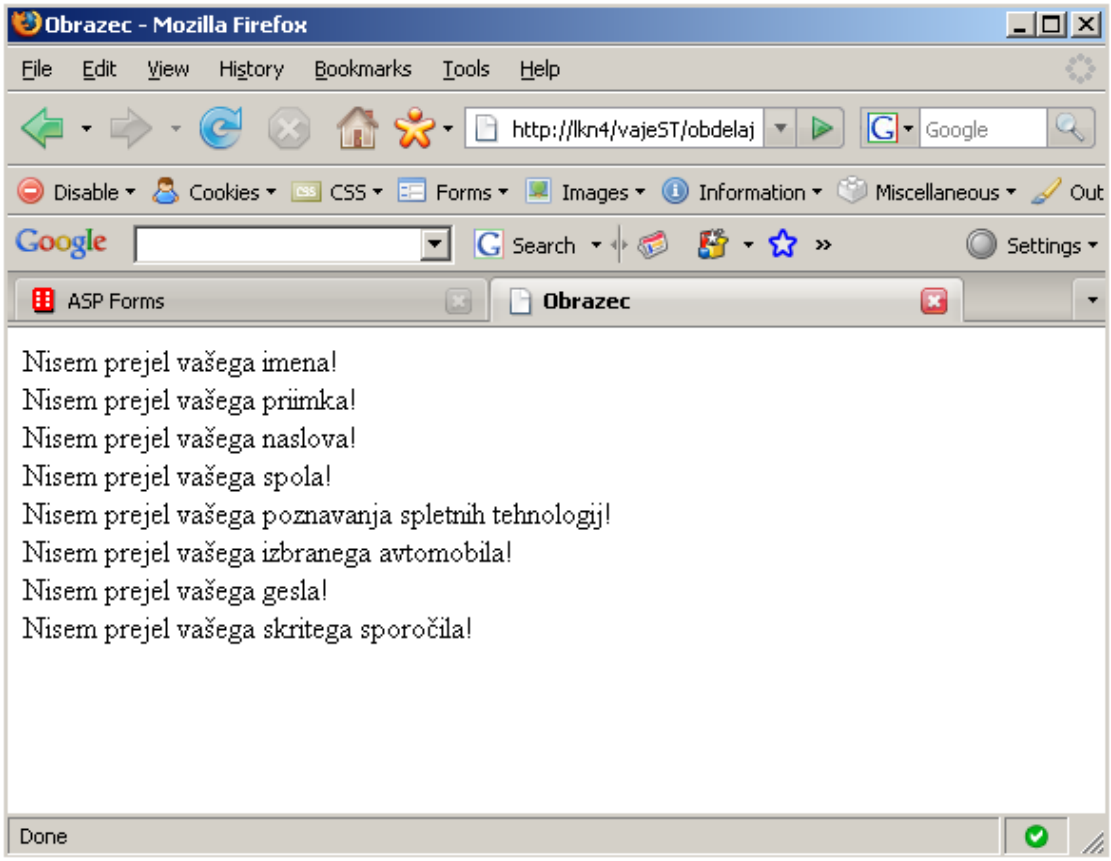
Preverjanje vsebine poslanih podatkov (validacija) se navadno izvede pred pošiljanjem podatkov v odjemalcu s pomočjo JavaScripta. Preverjanje lahko izvedemo tudi na strežniku, kar je še posebej primerno, če se podatki vpisujejo v podatkovno bazo.

1. V PHP dokumentu preverite, če je uporabnik v obrazec vpisal vse zahtevane podatke. To naredite tako, da z ***if*** stavkom preverite vrednosti spremenljivk. Če je vrednost neke spremenljivke prazen niz (t.j. ""), potem naj skripta vrne ustrezno sporočilo, kot to prikazuje Slika 15.

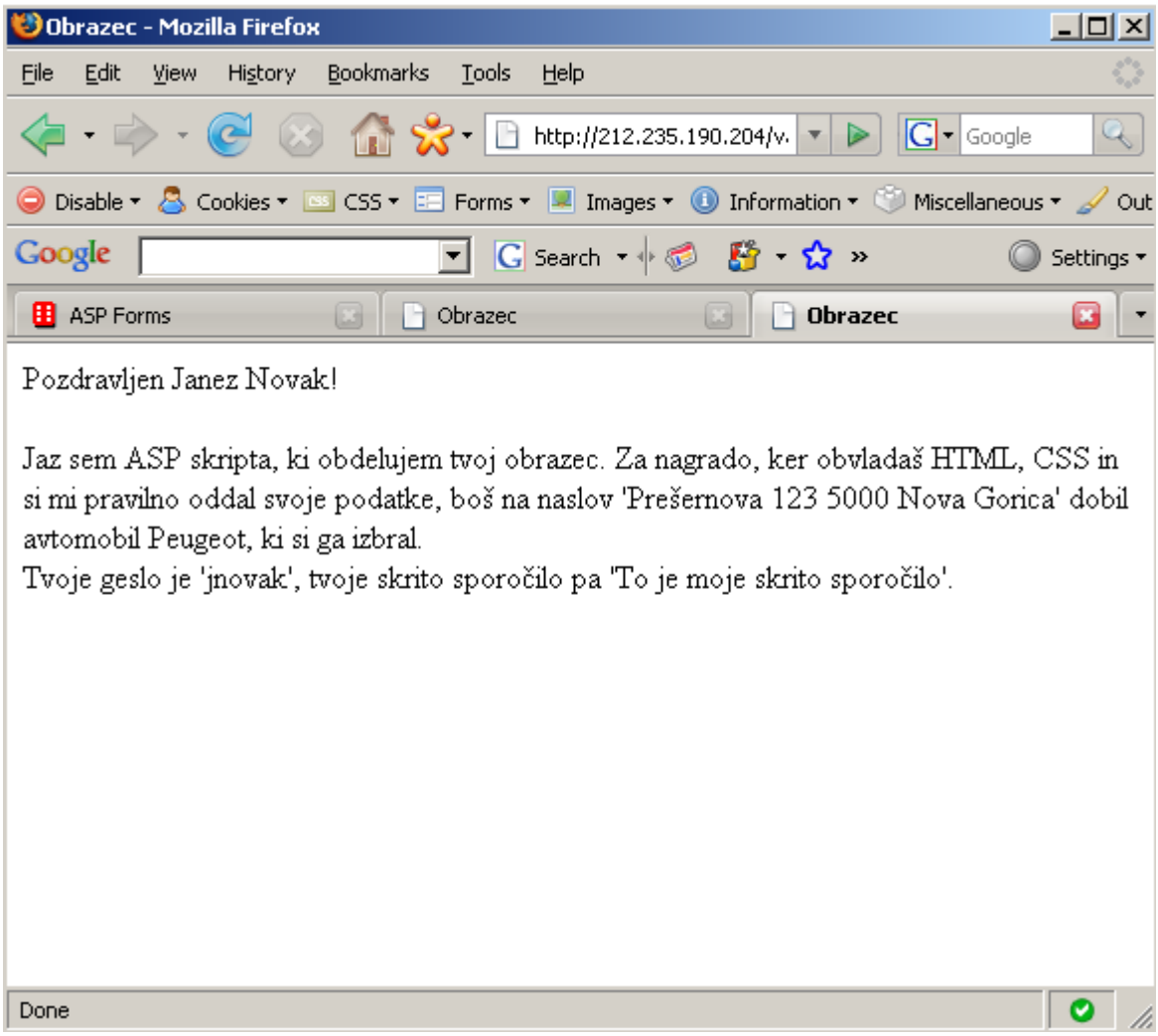
2. Dopolnite PHP dokument tako, da bo brskalniku vrnila vsebino, podobno tisti na Sliki 5. Pri tem upoštevajte naslednje:
   - Vsebino strani sestavite s pomočjo kombinacije HTML elementov in PHP skript.
   - Vsebina na Sliki 5 naj se izpiše le, če je uporabnik v obrazec vpisal vse podatke (torej nobena od PHP spremenljivk s podatki iz obrazca ni prazen niz). V nasprotnem primeru naj se izpišejo le sporočila o napakah iz Slike 4.

   Pomoč:

   - **Deklarirajte novo »kontrolno« spremenljivko in ji nastavite začetno vrednost (npr. 0).**
   - **Spremenite vrednost »kontrolne« spremenljivke (npr. na 1), če ugotovite, da je neko vnosno polje obrazca ostalo prazno (uporabite *if* stavek).**
   - **Preden pošljete odjemalcu odgovor, preverite vrednost »kontrolne« spremenljivke. Glede na njeno vrednost določite odgovor, ki ga bo videl uporabnik.**
   - Upoštevajte različne oblike besed za ženski in moški spol ("oddal"/"oddala" itd.)
3. Stran poljubno oblikujte, pri tem uporabite jezik CSS.



Slika 15: Preprosta validacija: če je vrednost neke spremenljivke prazen niz, je odgovor strežnika HTML dokument s sporočili o nepravilno sprejetih podatkih.



Slika 16: Pravilno sprejeti podatki: strežnik odjemalcu vrne HTML z besedilom, v katerega so smiselno vključeni podatki, ki jih je uporabnik predhodno vpisal v vnosni obrazec.

Naslednjič

Strežniško aplikacijo bomo nadgradili z:

- branjem in vpisovanjem v podatkovno bazo: ponovite osnove relacijskih podatkovnih baz, jezik SQL, vzpostavitev povezave s podatkovno bazo, branje iz nabora zapisov itd.
- uporabo sejnih spremenljivk: ponovite, kaj je seja, kako jo začnemo, zaključimo, kaj so sejne spremenljivke, kako jih uporabimo itd.

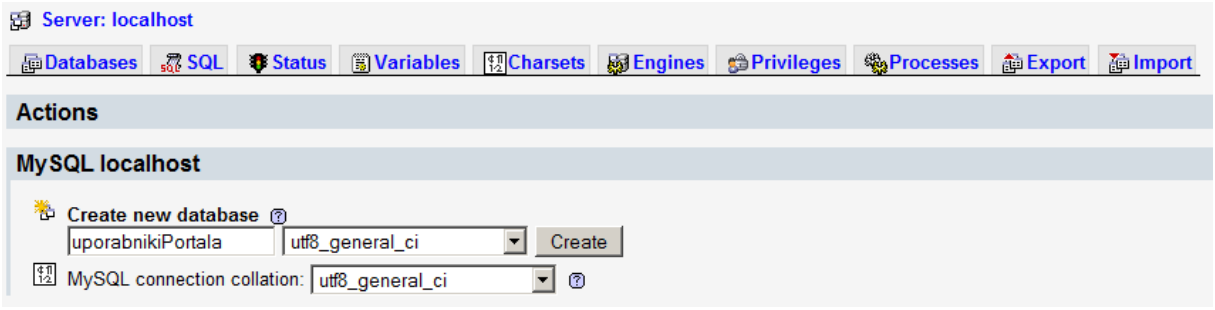
Vaja 7 : Podatkovne zbirke in jezik SQL

1. Zaženite lokalni strežnik Apache.
2. Na spletnem mestu http://www.lkn.fe.uni-lj.si/gradiva/ST/vaja7/ najdete
   - HTML dokument z obrazcem in
   - več datotek s programsko kodo v jeziku PHP, ki na podlagi podatkov iz obrazca pripravi dinamičen odgovor odjemalcu.

   Vse dokumente prenesite v mapo s svojim priimkom na ustrezno mesto v datotečnem sistemu, tako da bodo dostopni preko lokalnega strežnika.

1. Ustvarjanje podatkovne zbirke

- Paket XAMPP vsebuje podatkovni strežnik MySQL, ki ga zaženete v nadzorni plošči paketa XAMPP.
- Do nadzorne plošče sistema MySQL lahko dostopate preko URL naslova http://localhost/phpmyadmin/
- Novo podatkovno zbirko lahko ustvarite v razdelku *MySQL localhost* (Slika 1).

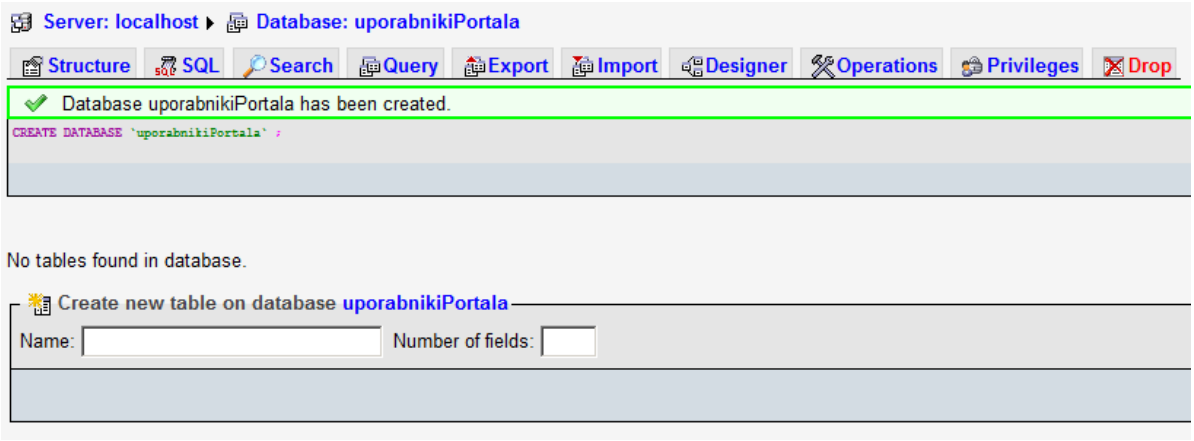


Slika 17

3. Zaženite podatkovni strežnik MySQL in v brskalniku prikažite njegovo nadzorno ploščo.
4. Ustvarite novo podatkovno zbirko in jo po želji poimenujte (ne uporabljajte presledkov). Uporabite zbirko znakov *utf_8_general_ci*.

- Relacijske podatkovne zbirke so sestavljene iz *tabel*. Podatki v tabelah so zbrani v *stolpcih* (ang. fields) in *vrsticah* (ang. rows).

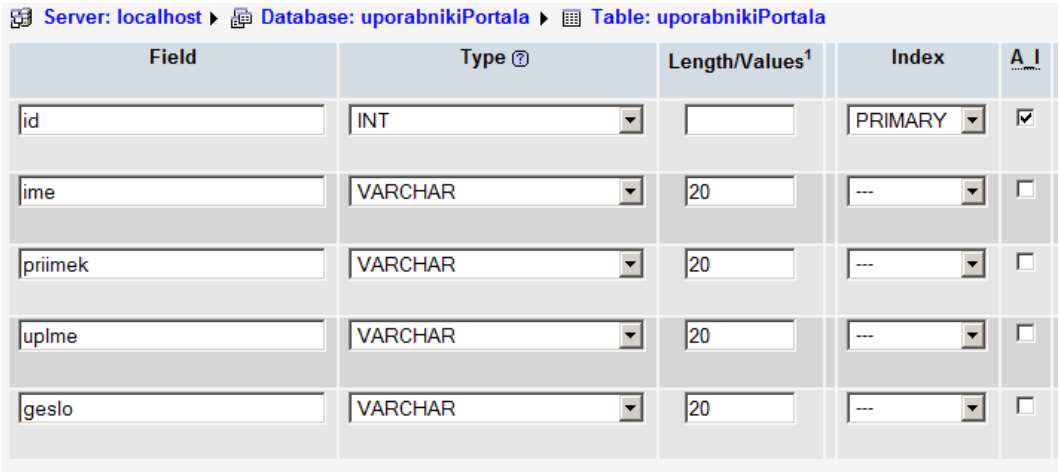
5. V podatkovni zbirki ustvarite novo tabelo in jo poimenujte 'uporabniki'. Tabela naj ima 5 stolpcev (Slika 2).



Slika 18

- Najpomembnejši lastnosti stolpca sta njegovo ime in tip. Najpomembnejši tipi so INT (nepredznačeno število), VARCHAR (znakovni niz spremenljive dolžine) in TEXT (besedilo).
- Vsaka vrstica v tabeli ima navadno enolični identifikator (id). Vrednost identifikatorja je navadno pozitivno celo število (tip INT), ki se poveča za 1 ob vsakem novem zapisu v tabelo (nastavitev »AUTO INCREMENT«). Za vrednosti tega polja običajno poskrbi pogon podatkovne zbirke.

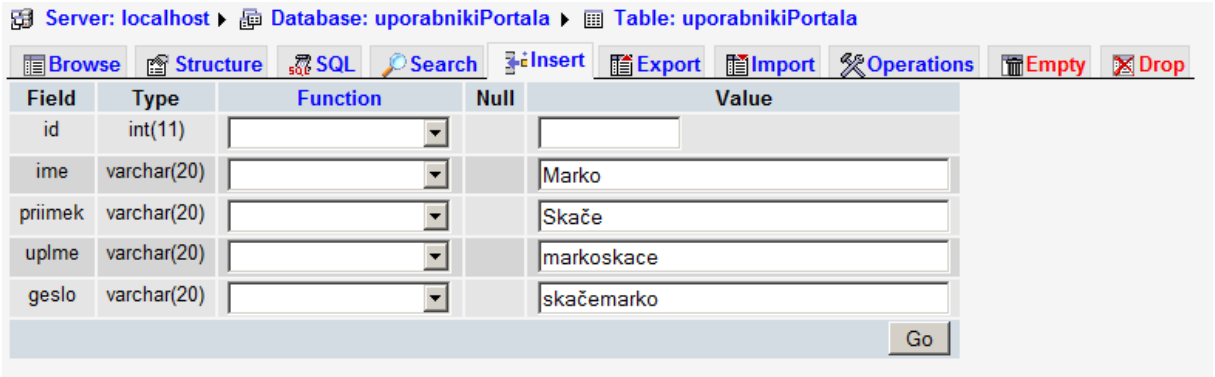
6. Pravkar ustvarjene stolpce poimenujte v skladu s Sliko 3. Določite jim tudi tip in dolžino. Pri stolpcu *id* nastavite še lastnosti Index na vrednost PRIMARY ter izberite možnost A_I (kratica za *Auto Increment*). Nadaljujte s klikom na *Save*.



Slika 19

7. Izberite zavihek *Insert* in v tabelo vpišite nekaj vrednosti (Slika 4).

Ko smo stolpec ustvarili, smo izbrali nastavitev A_I, zato pogon podatkovne zbirke sam vpisuje naraščajoče vrednosti v stolpec id. Stolpec *id* zato ob vpisovanju lahko pustite prazen.



Slika 20

2. *Uporaba podatkovne zbirke v PHP*

Pred delom s podatkovno zbirko v PHP je potrebno ustvariti povezavo do podatkovnega strežnika, kar prikazuje spodnja programska koda. Ustvarjanju povezave sledi programska koda, preko katere izvajamo poizvedbe v podatkovni zbirki (označena rdeče). Ko delo s podatkovno zbirko končamo, je povezavo potrebno zaključiti.

```php
<?php

// ustvarjanje povezave do podatkovnega strežnika na lokalnem
// računalniku z uporabniškim imenom »root« in praznim geslom:
$povezava = mysql_connect("localhost","root","");

// če povezovanje ni uspelo (napačen naslov, uporabniško, ime,
// geslo, itd,) izpišemo sporočilo o napaki:
if (!$povezava)
{
  die('Povezava ni uspela: ' . mysql_error());
}

// izberemo podatkovno zbirko, s katero bomo delali
mysql_select_db("uporabnikiPortala", $povezava);

// programska koda s poizvedbami v zbirki

// zaključimo povezavo
mysql_close($povezava);
?>
```

3. *Poizvedbe*

Poizvedbe v podatkovnih zbirkah opravljamo z uporabo jezika SQL.

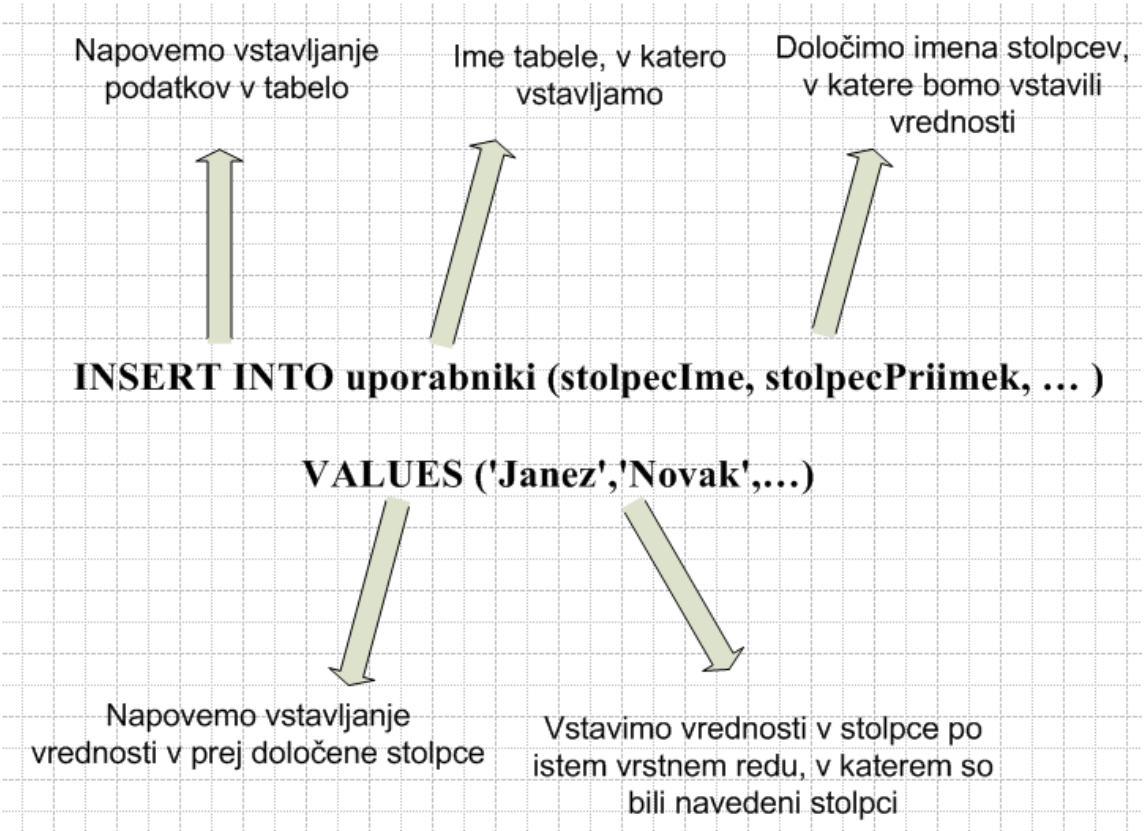
Podrobnosti o jeziku SQL lahko najdete na spletni strani http://www.w3schools.com/sql/

Vpisovanje v podatkovno zbirko

Slika 5 prikazuje SQL stavek za vpisovanje v podatkovno zbirko, spodnja koda pa izvedbo SQL stavka.

```php
if (!mysql_query($mojSQLstavek,$povezava))
{
  die('Napaka: ' . mysql_error());
}
```

Vrednost spremenljivke **`$mojSQLstavek`** v zgornji vrstici kode je znakovni niz v jeziku SQL.



Slika 21: Primer SQL stavka za vstavljanje vrednosti v tabelo

Znakovne nize v podatkovno zbirko vnašajte v enojnih narekovajih (glej Sliko 5)! Narekovaje uporabite tudi, če v SQL stavku uporabite PHP spremenljivko, katere vrednost je znakovni niz.

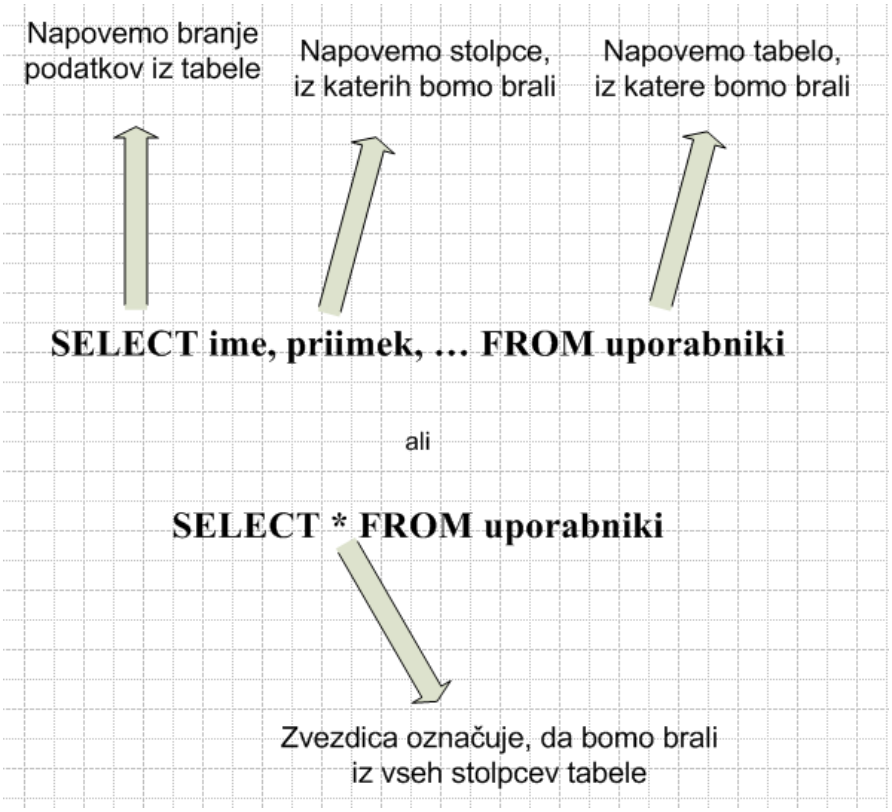
8. Dopolnite PHP dokument tako, da se vsakič, ko oddate popolne podatke, ti vpišejo v podatkovno zbirko. Pomagajte si s Sliko 5 ter kodo za vzpostavitev povezave in izvedbo poizvedbe.

Branje iz podatkovne zbirke

Slika 6 prikazuje SQL stavek za branje iz podatkovne zbirke, spodnja koda pa izvedbo SQL stavka ter izpis rezultatov.

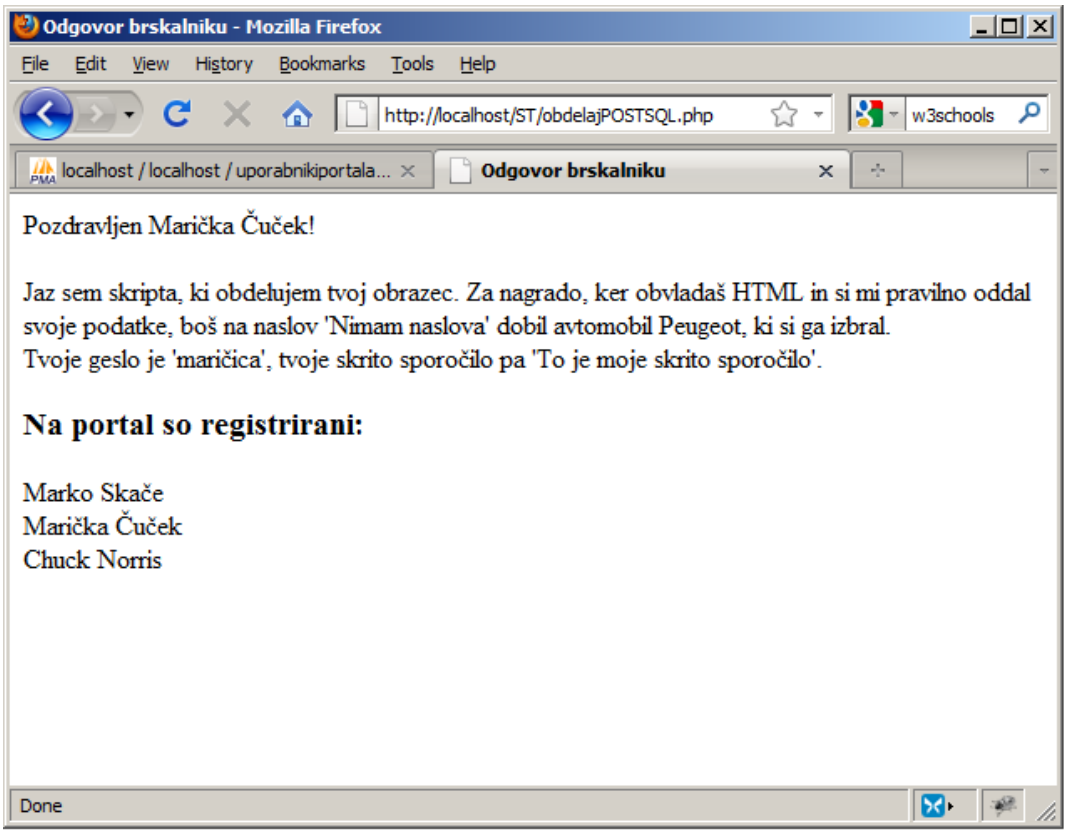
```
// rezultati poizvedbe se shranijo v spremenljivko $rezultat
$rezultat = mysql_query($mojSQLstavek);

// ker je rezultat poizvedbe polje vrednosti, se moramo skozi polje
// sprehoditi z while stavkom
while($vrstica = mysql_fetch_array($rezultat))
{
   echo "<br />";
   echo $vrstica['ime'] . " " . $vrstica['priimek'];

}
```

Vrednost spremenljivke **`$mojSQLstavek`** v zgornji vrstici kode je znakovni niz v jeziku SQL.



Slika 22: Primer SQL stavkov za branje iz tabele

9. PHP skripta naj vsakič, ko vpišete nove podatke, izpiše vse podatke vseh uporabnikov, shranjenih v vaši podatkovni zbirki (glej Sliko 7).
10. Odgovor odjemalcu primerno oblikujte, pri tem pa uporabite tudi jezik CSS.



Slika 23

4. *Domača naloga – sejne spremenljivke **

Preučite uporabo sejnih spremenljivk v PHP.

1. PHP skript dopolnite tako, da bo odgovor odjemalcu vseboval podatek, koliko krat je je uporabnik iz HTML obrazca poslal nepopolne podatke.
2. Ko uporabnik 3x pošlje nepopolne podatke, naj se uporabniška seja zaključi, uporabniku pa naj se izpiše sporočilo.
3. Poskusite poslati nepopolne podatke še četrtič. Kaj ugotovite? Kateri podatek je tisti, ki se v našem primeru ohranja znotraj ene seje?

Naslednjič

- Z JavaScriptom bomo spletno stran naredili bolj interaktivno. Ponovite:
    - kaj je JavaScript in za kaj se uporablja,
    - kako se »vgrajuje« v spletno stran,
    - skladnjo JavaScripta (deklaracija spremenljivk in funkcij, prirejanje vrednosti, itd.),
    - uporabo popup oken,
    - uporabo dogodkov za proženje JavaScript funkcij.

Literatura

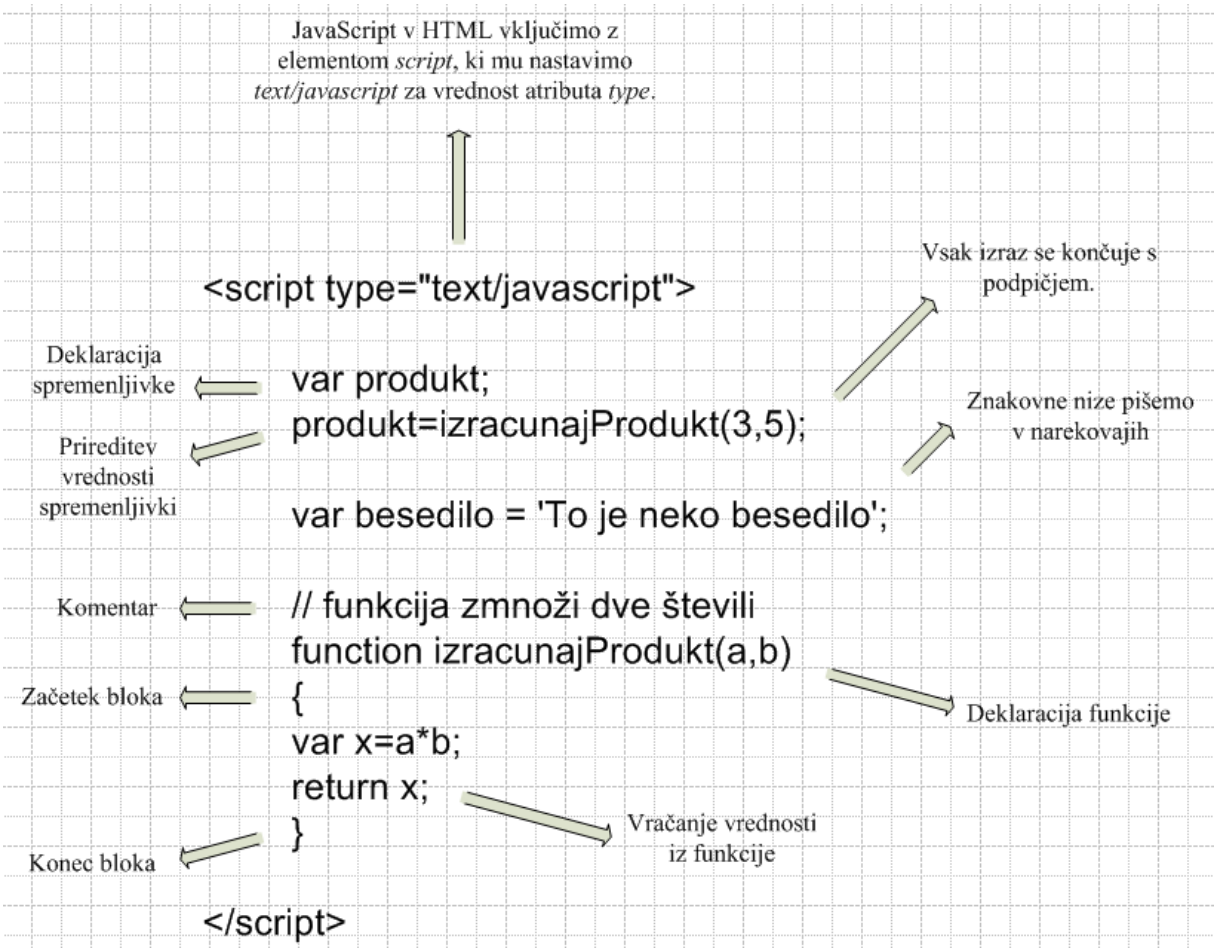
- Spletni tečaji W3Schools:
    - PHP http://www.w3schools.com/php/default.asp
    - SQL http://www.w3schools.com/sql/default.asp
- Prosojnice s predavanj

Vaja 8 - JavaScript

JavaScript

- je skriptni jezik, ki se izvaja na spletnem odjemalcu;
- je bil ustvarjen z namenom dodajanja večje interaktivnosti spletnim stranem.

1. Osnove jezika



Slika 24: Osnove jezika JavaScript

- HTML vsebino generiramo z izrazom ***document.write()***.
- Za spajanje nizov uporabljamo operator +

2. *Uporaba JavaScripta v spletni strani*

JavaScript kodo lahko v spletno stran vključimo na sledeče načine.

- **V glavo HTML dokumenta -** To kodo je navadno **potrebno poklicati** iz telesa dokumenta, ko se zgodi nek dogodek (npr. stran se naloži, uporabnik pritisne na gumb, spremeni se neka vrednost v obrazcu) ali pa iz druge JavaScript kode.
- **V telo dokumenta** - Ta koda **se izvede sproti** ob nalaganju dokumenta, običajno pa pripomore k dinamičnemu tvorjenju vsebine dokumenta (npr. izpis datuma)
- Če isto JavaScript kodo uporablja več spletnih strani hkrati, potem jo vključimo **v ločeno datoteko**, ki ima končnico *js*. To datoteko povežemo s spletnimi stranmi, tako da dodamo **v glavo HTML dokumentov** vrstico

```
<script type="text/javascript" src="datotekaZJavaScriptKodo.js"></script>
```

3. *Dogodki*

- Izvajanje JavaScript kode v spletnih straneh navadno sprožijo *dogodki*.
- Dogodki (ang. *events*) (večinoma) ustrezajo uporabnikovim dejanjem, ki jih lahko programsko prestrežemo in pripravimo odziv nanje.
- Dogodke vedno opazujemo na nekem HTML elementu, kar nakažemo z atributom, ki ustreza opazovanemu dogodku. Nekatere pogoste dogodke prikazuje Tabela 1.

```
<ime_elementa ime_dogodka="programska_koda">
```

- Kot vrednost atributov vpišemo JavaScript kodo, ki se izvede, ko (če) se dogodek zgodi. Koda navadno vsebuje klic neke predhodno definirane funkcije. **Funkcije v JavaScriptu se navadno izvedejo le ob nekem dogodku**.

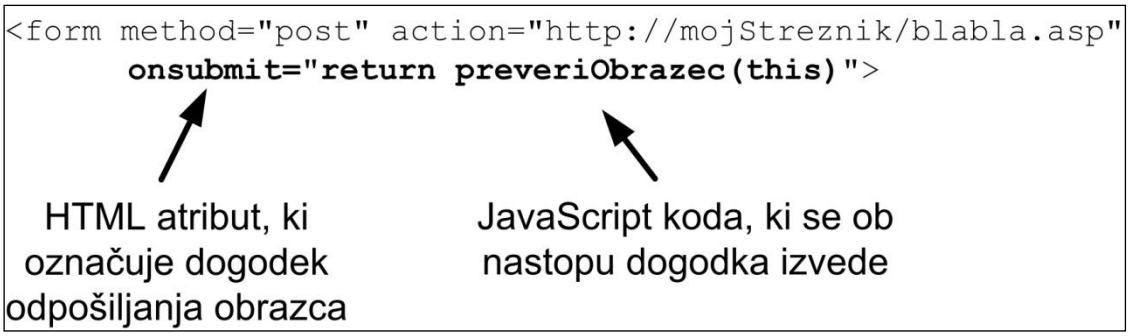
HTML atribut	pomen dogodka
onload	stran ali slika je naložena
onfocus	element je pridobil pozornost (*focus*)
onblur	element je izgubil pozornost
onchange	spremenila se je vsebina polja
onsubmit	uporabnik je sprožil odpošiljanje podatkov iz obrazca. Dogodek navadno uporabimo, če želimo preveriti podatke, preden se ti dejansko odpošljejo na strežnik.
onclick	uporabnik je »kliknil« na element
onmouseover	miškin kazalec se je premaknil nad element
onkeydown	pritisnjena je bila tipka na tipkovnici

Tabela 5: Dogodki in pripadajoči HTML atributi

Nekega dogodka ne moremo opazovati prav na vsakem HTML elementu. Dogodek *onsubmit* lahko tako na primer opazujemo samo na elementu *form*.

Celoten seznam dogodkov in elementov, s katerimi so posamezni dogodki povezani, je dostopen na spletnem naslovu http://www.w3schools.com/jsref/jsref_events.asp.

Primer uporabe dogodka



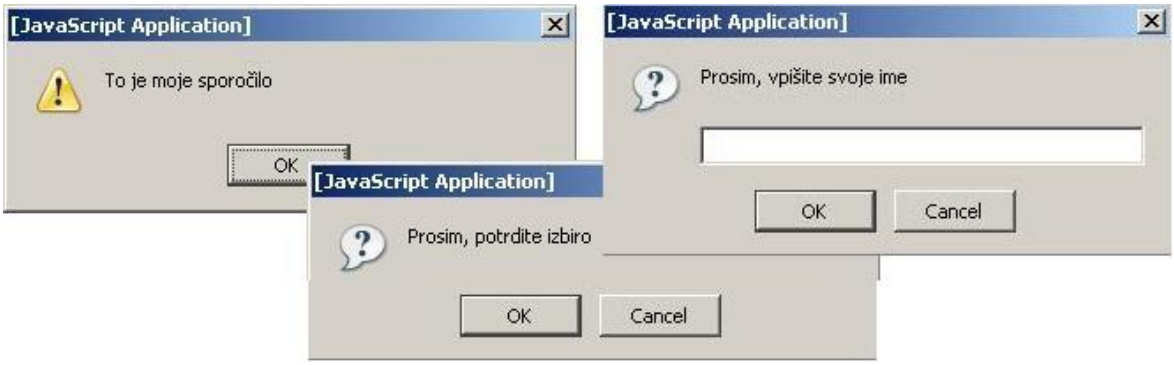
Slika 25: Primer uporabe dogodka pošiljanja podatkov iz obrazca

Slika 2 prikazuje primer uporabe dogodka, ki se nanaša na odpošiljanje podatkov iz obrazca:

1. Uporabnikov klik na gumb za odpošiljanje podatkov iz obrazca ustvari nov dogodek **onsubmit**.
2. Izvede se JavaScript koda, ki je vpisana kot vrednost atributa **onsubmit**.
3. V primeru na Sliki 2 ta koda vsebuje klic funkcije **preveriObrazec**, ki mora biti predhodno definirana (npr. v glavi dokumenta ali v ločeni datoteki).
4. Z uporabo rezervirane besede **return** dosežemo, da se odpošiljanje podatkov izvede le, če funkcija **preveriObrazec** vrne vrednost *true*. Če funkcija vrne *false*, je odpošiljanje preklicano.

4. Pojavna okna

Pojavna (»popup«) okna so prikazana na Sliki 3.



Slika 26: Pojavna okna *alert, confirm* in *prompt.*

alert('sporočilo');	v oknu izpiše podano sporočilo
confirm('sporočilo');	v oknu izpiše sporočilo in ponudi uporabniku izbiro *OK* oziroma *Cancel*. Izbira *OK* vrne vrednost *true*, izbira *Cancel* pa *false*.
prompt('sporočilo','privzeta vrednost');	od uporabnika zahteva vnos vrednosti. Izbira *OK* vrne vpisano vrednost, izbira *Cancel* pa vrne prazno vrednost (*null*).

Za zapis znakovnih nizov v JavaScriptu uporabljajte enojne narekovaje, da se izognete konfliktom z dvojnimi narekovaji v HTML značkah.

Če želimo besedilo v pojavnih oknih pomakniti v novo vrstico, uporabimo t.i. »ubežno zaporedje« **\n**.

Uporaba pojavnih oken v spletnih straneh ni priporočljiva, saj je za uporabnike moteča. Pojavna okna so priročna predvsem za izpisovanje pri hitrem testiranju strani.

1. S spletnega mesta http://www.lkn.fe.uni-lj.si/gradiva/ST/vaja8/ prenesite datoteko *obrazec.html*, lahko pa za reševanje nalog uporabite tudi lastno spletno stran.
2. S pomočjo dogodkov in pojavnih oken dopolnite prenesen dokument:
   a. Izberite si nek HTML element. Vsakič, ko se z miško zapeljete čez izbran element, naj se v pojavnem oknu izpiše poljubno sporočilo.
   b. Ko odpošljete podatke iz obrazca, naj se pojavi okno, ki od vas zahteva potrditev vaše odločitve.

5. *Delo z »vgrajenimi« JavaScript objekti*

1. **Objekt *String*** omogoča delo z znakovnimi nizi.
   - Primera:
     - ***znakovni_niz.length*** vrne dolžino znakovnega niza.
     - ***znakovni_niz.toUpperCase()*** pretvori vse črke niza v velike.

Celoten seznam metod objekta *String* je dostopen na spletnem naslovu http://www.w3schools.com/jsref/jsref_obj_string.asp.

2. **Objekt *Date*** omogoča delo z datumom in časom.
   - Pred uporabo moramo objekt *Date* ustvariti, npr.: ***var mojDatum=new Date()***
   - Trenutni datum in čas lahko izpišemo tako, da izpišemo kar vrednost spremenljivke, ki smo ji priredili objekt *Date* (v našem primeru ***mojDatum***).
   - Ostale metode kličemo na objektu samem. ***mojDatum.getFullYear()*** tako na primer vrne trenutno leto.

Celoten seznam metod objekta *Date* je dostopen na spletnem naslovu http://www.w3schools.com/jsref/jsref_obj_date.asp.

3. **Objekt *Math*** vsebuje matematične funkcije in konstante.
    - Primera:
        - ***Math.PI;*** vrne število *pi*.
        - ***Math.round();*** zaokroži število na najbližje celo število.

Celoten seznam metod in lastnosti objekta *Math* je dostopen na spletnem naslovu http://www.w3schools.com/jsref/jsref_obj_math.asp.

1. Na vrh spletne strani izpišite trenutni datum in razliko v minutah med lokalnim časom in časom GMT (*Greenwich Mean Time*).
2. Razliko pomnožite z naključnim številom med 0 in 1 in rezultat zaokrožite. Uporabite funkcije iz razreda *Math*.
3. Dodajte nekaj naključnosti tudi prikazu spletne strani. Na spletu poiščite dve sliki. Vsakič, ko stran naložite, naj prikaže ena izmed obeh slik, pri čemer je verjetnost prikaza vsake slike 50%.

   Pomoč:

   - **V HTML vključite element, ki predstavlja sliko, ter mu dodelite *id*. V JavaScriptu lahko nato pot do slike nastavljate z izrazom *document.getElementById('id_ki_ste_ga_dodelili').src='pot_do_slike'***
   - **Za ustvarjanje naključnosti in odločanja, katero sliko boste prikazali, si pomagajte z *Math.random()* in *if* stavkom.**

6. Validacija na odjemalcu

Validacija (preverjanje vsebine podatkov) se navadno izvede pred pošiljanjem podatkov na strežnik.

Preverite podatke iz obrazca, preden so ti poslani strežniku:

1. Ugotovite, kateri dogodek ustreza odpošiljanju obrazca.
2. Dogodek vključite kot atribut v tisti HTML element, kamor sodi.
3. Vrednosti atributa priredite rezultat izvajanja funkcije, ki bo preverjala obrazec (glej *Dogodki / Primer uporabe dogodka*).
4. V glavi HTML dokumenta ustvarite ogrodje funkcije, v kateri boste izvedli preverjanje.
5. Podatke iz obrazca v funkcijo posredujete tako, da pri klicu funkcije kot parameter uporabite ključno besedo ***this***, torej ***ime_vase_funkcije(this)***.

   Rezervirana beseda **this (**»ta«**)** predstavlja programski objekt, ki je programska predstavitev HTML elementa, v katerem je ključna **this** besedna uporabljena. V našem primeru bo torej **this** predstavljal programski objekt vrste **form**, saj bo ta ključna beseda uporabljena v znački istoimenskega HTML elementa. Objekt vrste **form** predstavlja programsko predstavitev obrazca skupaj z vsemi njegovimi gradniki in vrednostmi, ki so v gradnike trenutno vpisane.

6. Znotraj funkcije lahko do podatkov iz obrazca dostopate preko imena spremenljivke, ki ste ga navedli v glavi funkcije (torej *obrazec* v primeru na Sliki 4). Do vrednosti posameznega polja v obrazcu lahko dostopate prek imena polja v obrazcu, ki mu dodate lastnost **value**. Na Sliki 4 je prikazan izraz, s pomočjo katerega dostopate do vrednosti, ki je bila vpisana v polje »ime«.

```
function preglejObrazec(obrazec)
{
//spodnji izraz vrne vrednost, vpisano v polje 'ime':
obrazec.ime.value;
}
```

Slika 27: Prebiranje vrednosti iz obrazca

7. Dopolnite funkcijo tako, da bo preverila vrednosti polj v obrazcu. Če vsaj ena vrednost manjka (prazen niz), naj funkcija vrne *false*, v nasprotnem primeru pa *true*.
8. V primeru, da je uporabnik pozabil izpolniti določeno vnosno polje, naj se ob njem izpiše obvestilo.

7. Uporaba časovnika

Z uporabo metode ***setTimeout()*** je mogoče zakasnjeno izvesti programsko kodo. V omenjeno metodo posredujemo kodo in časovno zakasnitev v milisekundah.

1. Ustvarite funkcijo, ki bo vsako sekundo spremenila sliko na spletni strani. Izvedete lahko ciklično preklapljanje med slikama, ki ste ju že uporabili pri naključnem prikazu. Ciklično preklapljanje naj se sproži, ko kliknete na sliko.

   Pomoč: Prvič naj se funkcija pokliče ob nekem dogodku. Nato naj v časovnih intervalih ene sekunde funkcija z uporabo metode *setTimeout()* kliče sama sebe.

Več informacij o metodi ***setTimeout()*** in časovnikih lahko dobite na spletni strani http://www.w3schools.com/js/js_timing.asp.

Naslednjič

- Pregledali bomo hierarhijo vgrajenih objektov v JavaScriptu in nekaj teh objektov uporabili.
- Obravnavali bomo objektni model dokumenta (t.i. DOM, *Document Object Model*) in ga uporabili za branje in nastavljanje lastnosti obstoječih elementov na spletni strani.

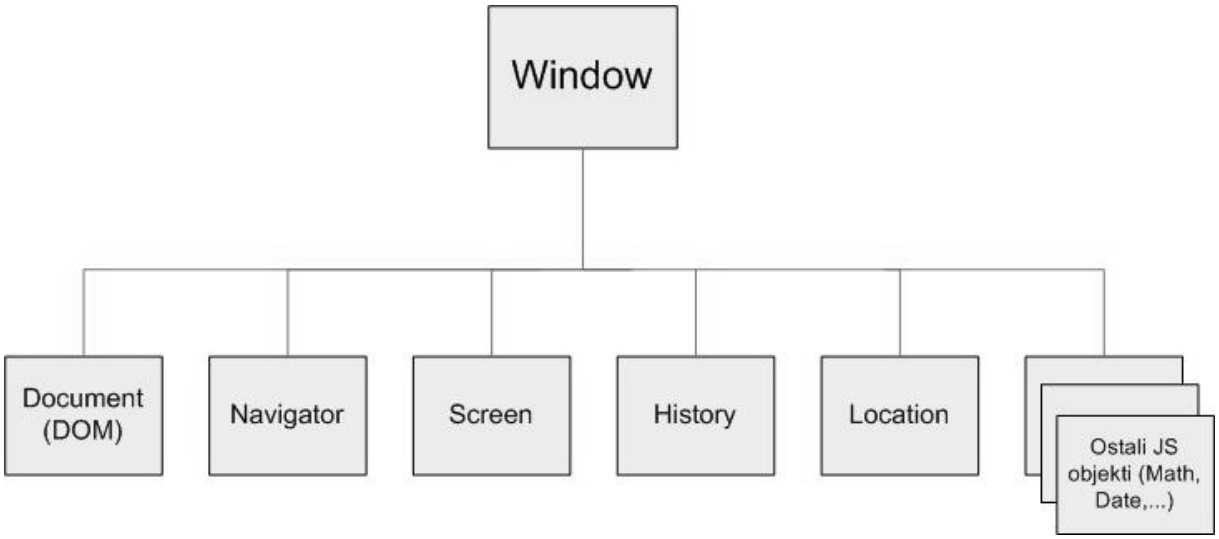
Temeljito se seznanite s hierarhijo vgrajenih objektov in principom objektnega modela DOM!

Literatura

1. Spletni tečaji JavaScripta:
   - http://www.w3schools.com/js/default.asp
   - http://www.howtocreate.co.uk/tutorials/javascript/important
2. Prosojnice s predavanj

Vaja 9 : Objektni model spletne strani

1. Hierarhija objektov brskalnika



Slika 28: Hierarhija objektov brskalnika

- Objekt **Window** je najvišji v hierarhiji objektov brskalnika in predstavlja okno brskalnika. Objekt je ustvarjen samodejno, ko brskalnik naleti na element **body** ali **frameset**. Prek lastnosti in metod objekta **Window** lahko ugotavljamo ali nastavljamo lastnosti okna brskalnika.

Metode in lastnosti objekta **Window** lahko uporabite brez predpone **Window** (pišemo lahko na primer **Window.alert('Sporočilo!');** ali samo **alert('Sporočilo!');**).

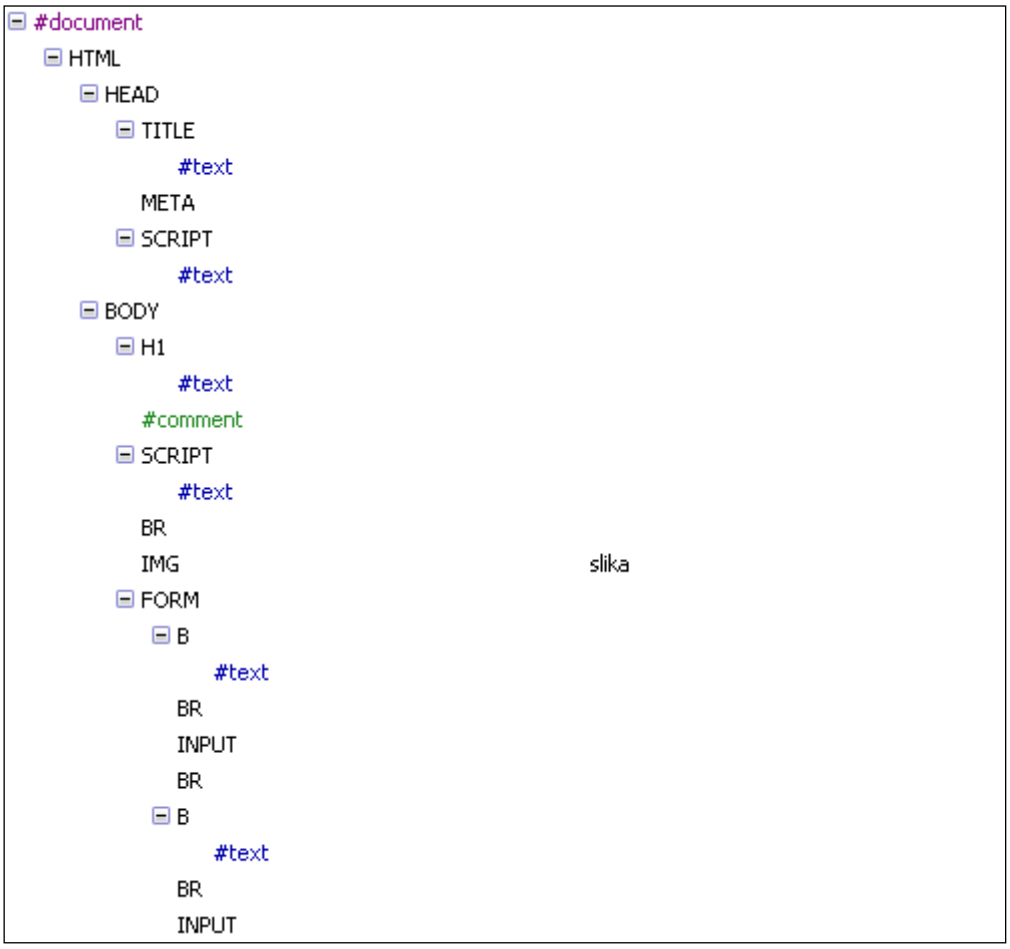
Seznam lastnosti in metod objekta Window je dostopen na spletnem naslovu http://www.w3schools.com/jsref/obj_window.asp.

- Objekt **Navigator** vsebuje informacije o uporabnikovem brskalniku.
- Objekt **Screen** vsebuje informacije o uporabnikovem zaslonu.
- Objekt **History** vsebuje informacije o obiskanih URL-jih.
- Objekt **Location** vsebuje informacije o trenutnem URL-ju.

2. Objekt *Document* in *HTML DOM*

- HTML DOM (*Document Object Model*) je standard, s pomočjo katerega lahko dostopamo do HTML elementov, njihovih atributov in vsebine.

- DOM opisuje HTML dokument kot drevesno strukturo, pri čemer vsakemu HTML elementu, atributu ali tekstu ustreza lastno vozlišče v obliki programskega objekta (glej Sliko 2).



Slika 29: Del DOM drevesa HTML dokumenta *obrazec.html*

- Objekt **Document** kot korenski element HTML DOM drevesa predstavlja celoten HTML dokument. Pomembnejše funkcije objekta **Document** so:
  - **write('HTML_tekst')** dodaja HTML vsebino v dokument.
  - **getElementById('id_elementa')** vrne programski objekt, ki ustreza HTML elementu s podanim unikatnim identifikatorjem (*id*-jem).

- **getElementsByTagName('ime_značke')** vrne polje vseh objektov, ki predstavljajo HTML elemente s podano značko (npr. **'img'** za vse slike v HTML dokumentu).

Seznam lastnosti in metod objekta **Document** je dostopen na spletnem naslovu http://www.w3schools.com/jsref/dom_obj_document.asp.

Seznam objektov v DOM drevesu z njihovimi lastnostmi lahko najdete na spletni strani http://www.w3schools.com/jsref/default.asp.

3. Naloge

1. S spletnega mesta http://www.lkn.fe.uni-lj.si/gradiva/ST/vaja9/ prenesite datoteko *obrazec.html*.

2. Dopolnite kodo tako, da se bo obrazec odposlal, ko bo uporabnik kjerkoli na spletni strani pritisnil desni gumb na miški:
   a. ugotovite, kateri dogodek ustreza pritisku gumba na miški;
   b. izberite element, v katerega je smiselno vključiti atribut z dogodkom;
   c. v glavi HTML dokumenta definirajte funkcijo, ki se bo ob dogodku izvedla;
   d. v funkcijo posredujte podatke o dogodku

      *To storite z uporabo parametra '**event**' (pri klicu funkcije vanjo posredujete objekt tipa **Event**, ki se samodejno ustvari, ko se zgodi dogodek, povezan z opazovanim elementom).*

   e. ugotovite, kateri gumb na miški je bil pritisnjen

      To ugotovite z branjem vrednosti lastnosti 'button' objekta Event (torej event.button). Če je bil pritisnjen desni gumb, ima lastnost 'button' vrednost 2.

   f. preglejte metode objekta **Form** in uporabite tisto, ki odpošlje obrazec.

      *Ker je potrebno omenjeno metodo poklicati na objektu, ki v DOM modelu predstavlja obrazec, morate pridobiti referenco na ta objekt. To storite tako, da najprej obrazcu, dodelite nek id, nato pa do pripadajočega objekta programsko dostopate z uporabo metode **getElementById('id_elementa')**.*

3. Izberite si poljuben dogodek in HTML element. Dogodek naj sproži branje ali nastavljanje poljubne lastnosti izbranega elementa.

   *Postopek je podoben tistemu pod točko 2. Pazite na tip objekta, ki ga vrača metoda **getElementById('id_elementa')**, saj je ta odvisen od izbranega elementa.*

4. Ko kliknete na naslov »*Registracija na spletni portal*«, naj se tekst v naslovu spremeni.

   *Vsebino nekega elementa spremenite tako, da objektu, ki programsko predstavlja izbran element, nastavite lastnost **innerHTML** na novo vrednost.*

5. Izberite si dva elementa in jima nastavite poljubni lastnosti prikaza.

   *Stil prikaza nekega elementa določimo prek objekta **Style**. HTML elementu z id **'id_elementa'** bi tako na primer nastavili rdečo barvo z izrazom:*

   document.getElementById('id_elementa').style.color='red';

6. Z uporabo metode **getElementsByTagName** spremenite barvo ozadja v vseh poljih tipa *input*.

   *Metoda **getElementsByTagName** vrne polje vseh objektov v dokumentu, ki ustrezajo podanemu imenu elementa (znački). Skozi polje se lahko pomikate s pomočjo for zanke.*

4. Literatura

1. W3Schools, Spletni tečaji JavaScripta in HTML DOM.
2. Prosojnice s predavanj