Svetovni splet

Zapiski predavanj
Bolonjski študijski program (1. stopnja UN)
2011/2012

Jaka Sodnik
jaka.sodnik@fe.uni-lj.si

1. Internetni protokoli

Definicija interneta

- internet: medmrežje ali povezavo med omrežji različnih vrst.
- Internet: svetovno omrežje Internet, ki je zasnovano na protokolu IP (Internet Protokol).
- Svetovni splet (World Wide Web): opisuje le določeno storitev (aplikacijo), ki teče preko Internet omrežja in sloni na protokolu HTTP (Hyper Text Transport Protocol). Protokol je namenjen za dostop do dokumentov, ki so zapisani v jeziku HTML (Hyper Text Makup Language).

Internet protokolni sklad

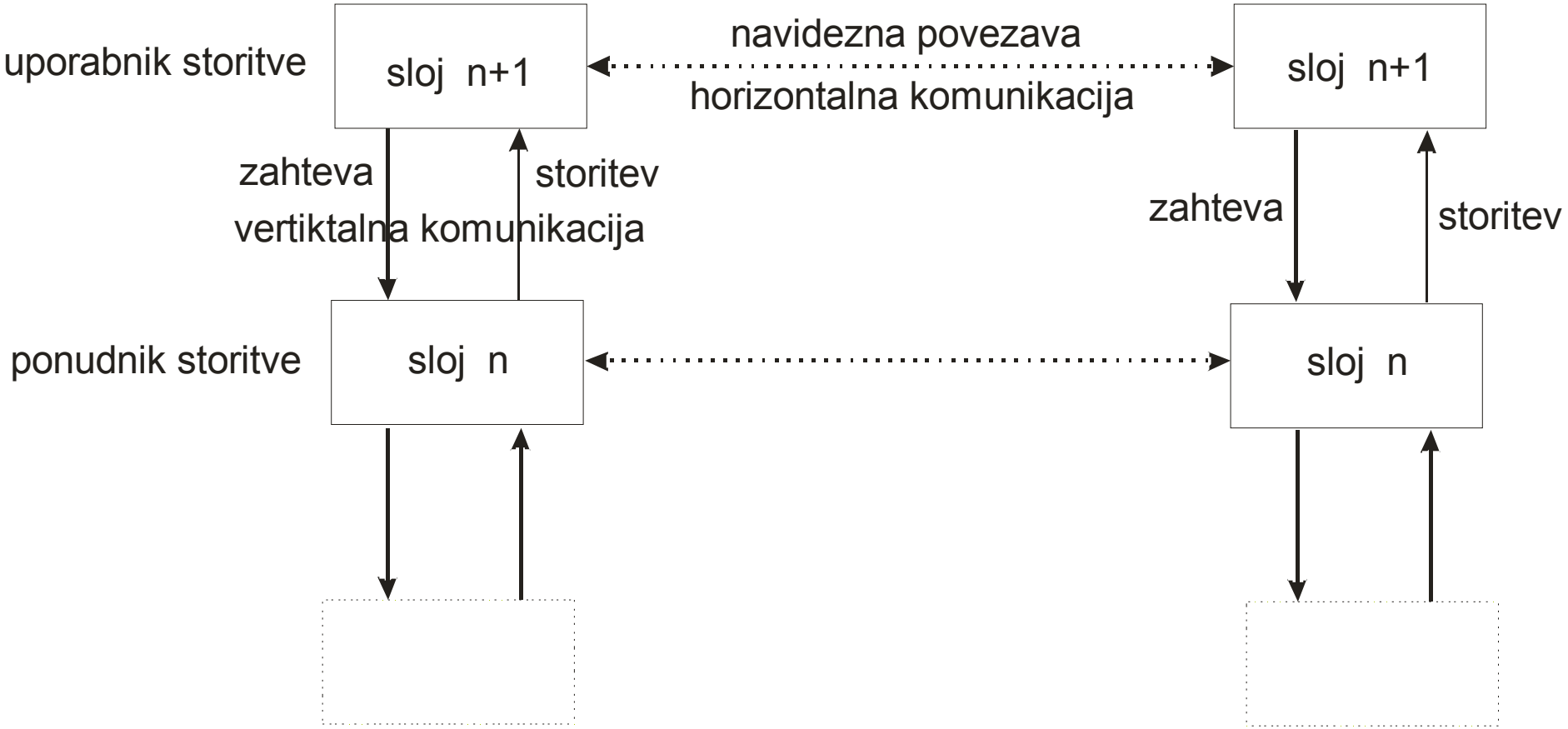
- Omrežje internet sloni TCP/IP protokolnem skladu sklad
  - Poleg TCP (Transmision Control Protocol) se uporablja tudi UDP (User Datagram Protokol).
- Za standardizacijo skrbi IETF(Internet Engineering Task Force)
  - izdaja RFC (Request For Comment), ki so oštevilčeni po vrstnem redu.
- RFC standardi so dostopni na Internet naslovu

http://www.rfc-editor.org

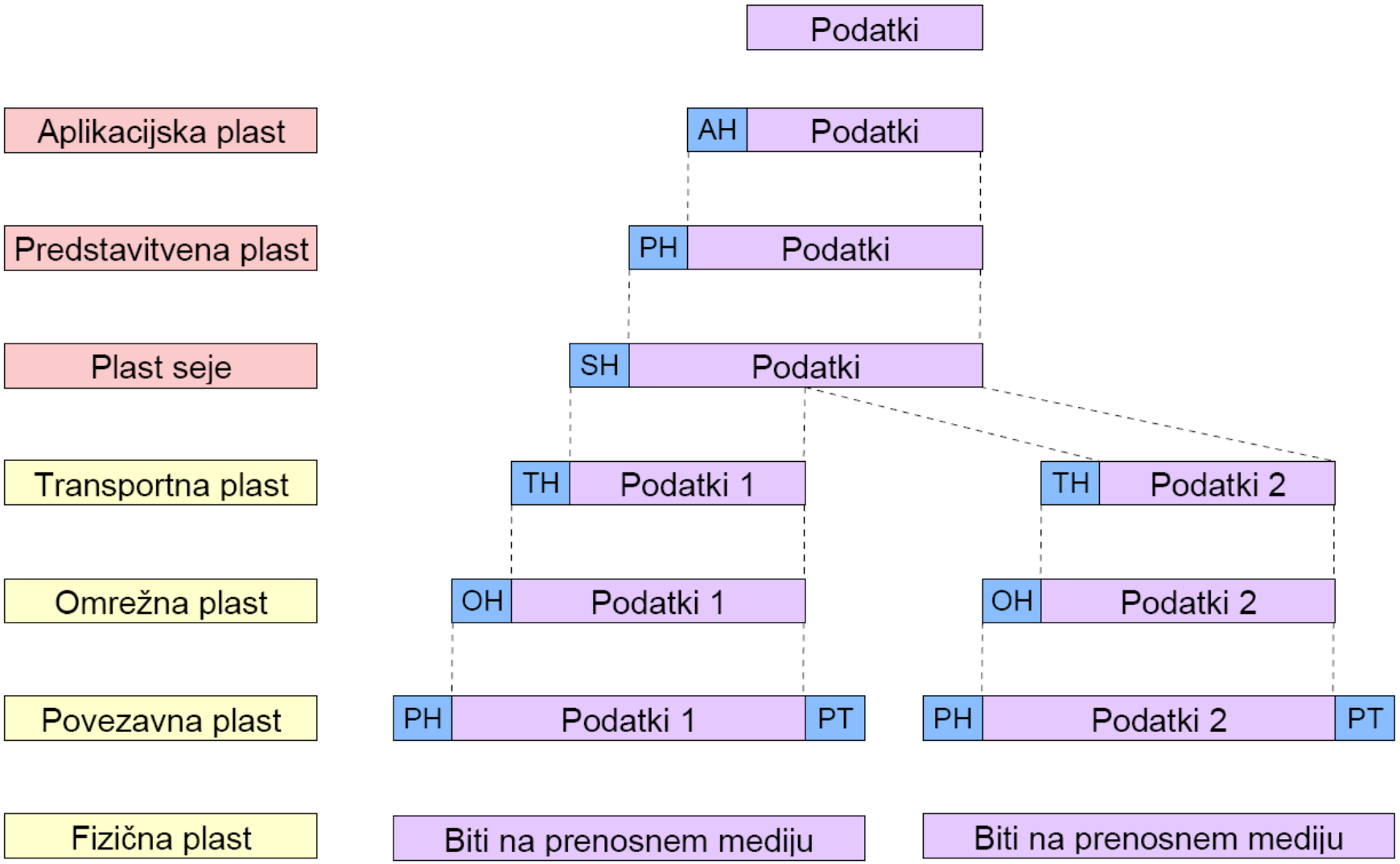
Protokol

- Nabor pravil in postopkov, ki določajo obliko in način komunikacije (prenosa podatkov) med dvema računalnikoma oz. aplikacijama

Protokolni sklad



Kontrolne informacije v protokolnem skladu

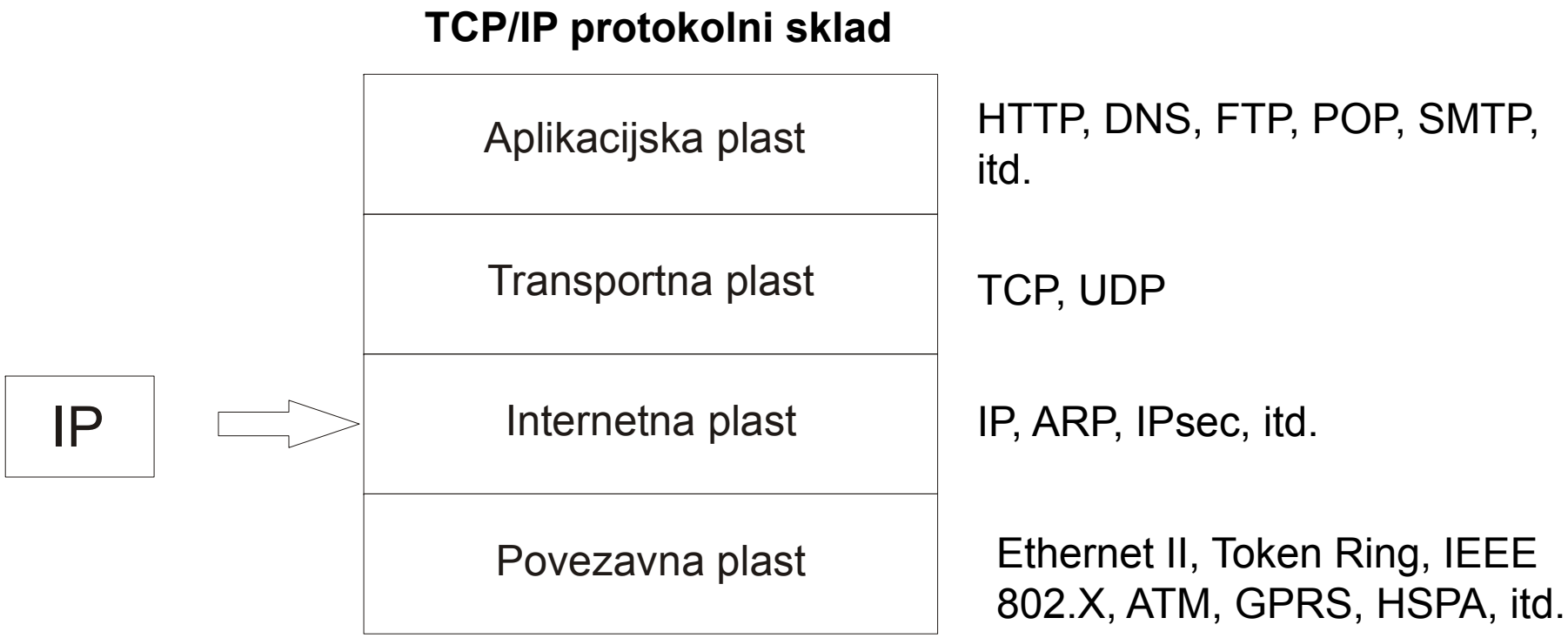


TCP/IP in OSI model

OSI model	TCP / IP
Aplikacijski sloj	Aplikacijski sloj
Predstavitveni sloj	
Sejni sloj	Transportni (prenosni) sloj
Prenosni sloj	
Omrežni sloj	Internetni (omrežni) sloj
Povezovalni sloj	Povezavni (dostopovni) sloj
Fizični sloj	

Uvrstitev TCP/IP v OSI ni enoumna!

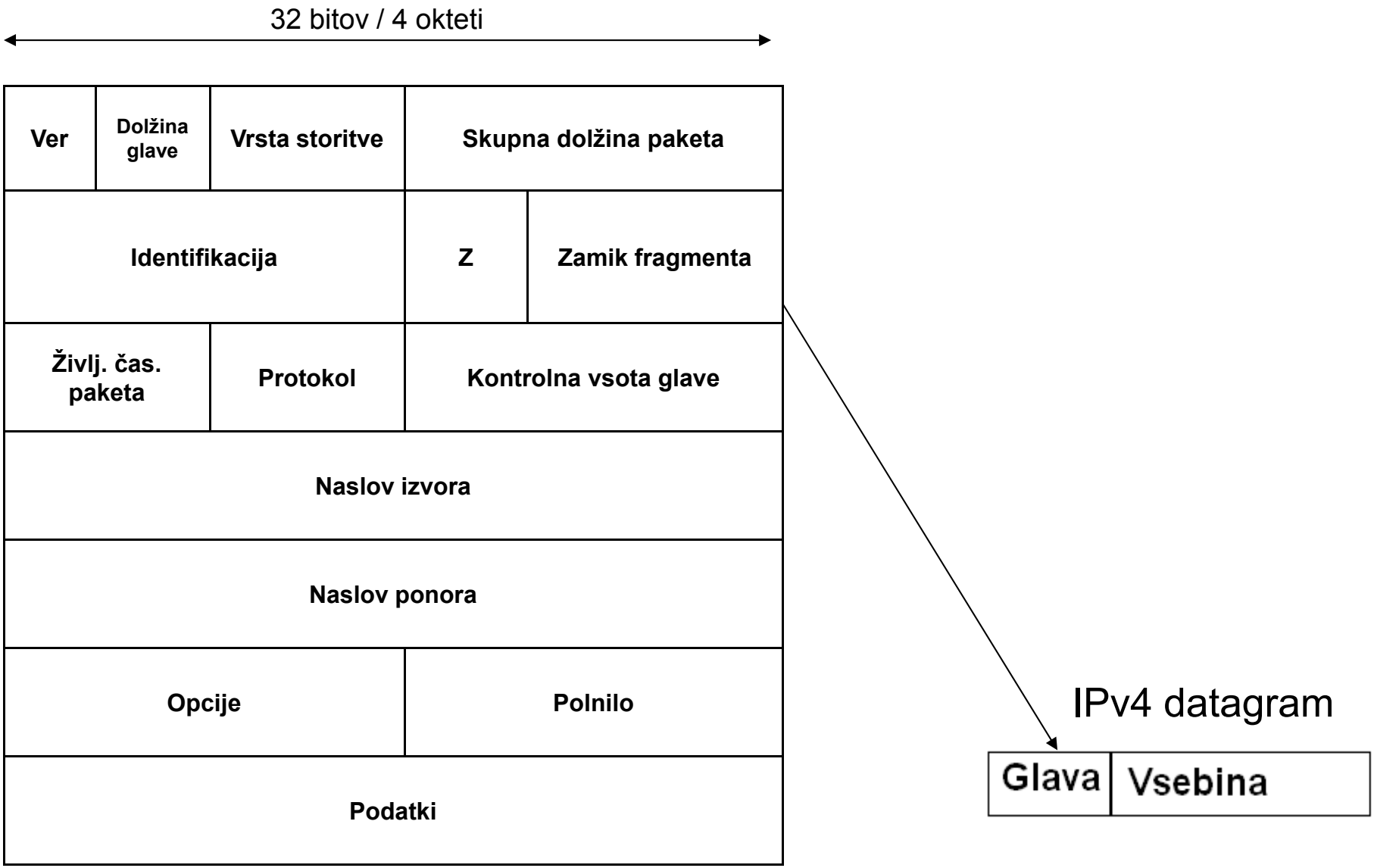
Protokoli Internetne plasti



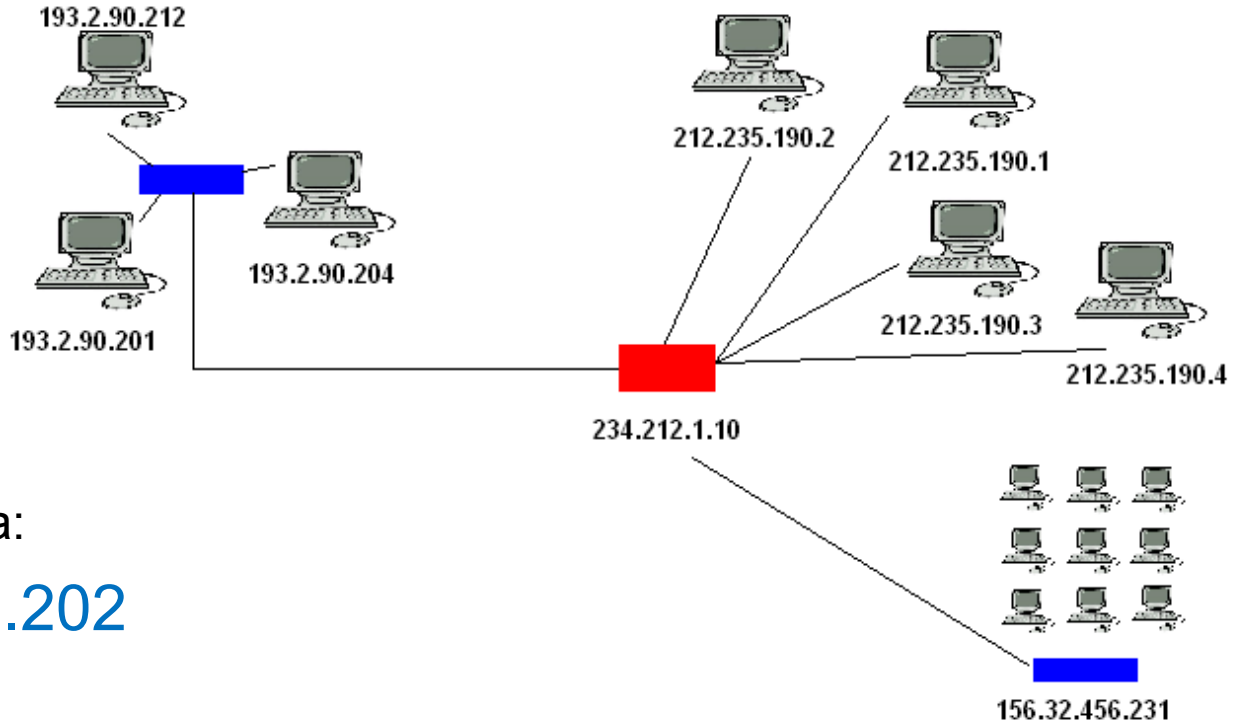
IP (Internetni protokol)

- Skrbi za prenos datagramov (paketov) od izvora do ponora preko internetnega omrežja
  - Nepovezavna storitev
  - Nezanesljiva – best effort
- Glavne naloge IP protokola
  - Logično naslavljanje omrežnih povezav
  - Usmerjanje in prenos podatkov po omrežjih
  - Razbijanje toka podatkov na različno velike pakete (glede na zahteve fizičnega omrežja)
  - Preverjanje pravilnost sprejetih paketov
    - Kontrola vsota - checksum

IP (Internetni protokol)



- IPv4 naslavljanje
  - 32-bitna unikatna oznaka računalnika v omrežju (IPv4)
  - IP naslov je sestavljen iz štirih oktetov
    - Posamezne oktete ločimo s piko



Primer naslova:

212.235.190.202

IP (Internetni protokol)

- Vsak IP naslov je sestavljen iz naslova podomrežja (subnet) in naslova naprave (host)
- Podomrežna maska (subnet mask)
  - 32 bitna beseda, ki z enicami označuje omrežje in z ničlami napravo znotraj omrežja

	Desetiški zapis	**Binarni zapis**
IP naslov	192.168.5.10	11000000.10101000.00000101.00001010
Maska podomrežja	255.255.255.0	11111111. 11111111. 11111111.00000000
Mrežni del naslova	192.168.5.0	11000000.10101000.00000101.00000000
Naslov računalnika	0.0.0.10	00000000.00000000.00000000.00001010

IP (Internetni protokol)

- Rezervirana naslova v podomrežju
  - Same 0: enoumna identifikacija omrežja
  - Same 1: hkratna oddaja vsem napravam (broadcasting)
    - Pri 7-bitni maski je torej na voljo 126 omrežnih mest

- Pred uvedbo podomrežnih mask so se uporabljali razredi
  - Prefiks določa omrežje

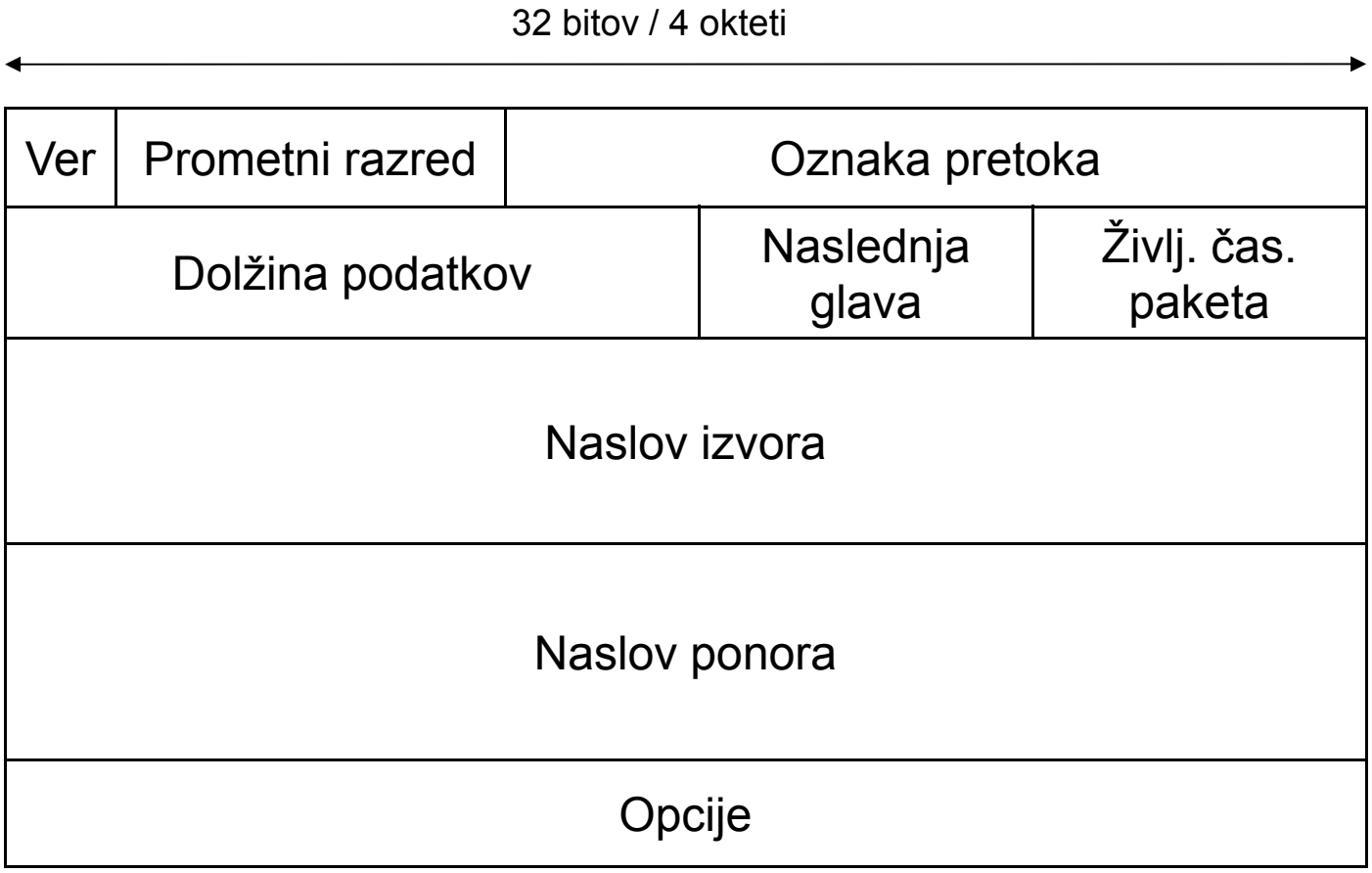
razred	prefiks	maska	št. omrežij	št. naprav
A	0	255.0.0.0	125	16.777.213
B	10	255.255.0.0	16.383	65.533
C	110	255.255.255.0	2.097.151	254
D	1110			
E	1111			

IP (Internetni protokol)

- Lokalna in globalna omrežja
  - Vsak IP paket vsebuje naslov prejemnika in naslov pošiljatelja
  - Naprava s pomočjo omrežne maske ugotovi ali je ciljna naprava v lokalnem ali v globalnem omrežju
    - Napravam v lokalnem omrežju se datagram dostavi na osnovi protokola lokalnega omrežja
    - MAC naslov (48 bitov)
      - Primer: 18-A7-03-EC-7B-E9
    - ARP (Address Resolution Protocol)
      - Protokol za preslikavo med lokalnim IP naslovom in fizičnim naslovom
      - Vsem napravam v omrežju se pošlje povpraševanje po določenem IP naslovu, ciljna naprava odgovori z svojim fizičnim naslovom

- Lokalna omrežja
  - Globalni IP naslov pridobimo od svojega ponudnika internet storitev ISP (Internet Service Provider)
  - Zaradi enostavnejšega usmerjanja se ISP naslovi dodeljujejo večjim ISP, ki jih potem dodeljujejo manjšim ISP.
  - Geografsko dodeljevanje
    - Naslovi 193.x.x.x so npr. vsi v Evropi.
  - Zaradi pomanjkanja IP naslovov, lahko v lokalnem ali privatnem MAN omrežju uporabljamo lokalne IP naslove, ki jih pri izhodu iz omrežja pretvorimo v globalne.
  - Da ne bi prišlo do mešanja lokalnih in globalnih naslovov, je za lokalne naslove v vsakem razredu rezervirano eno omrežje

- IP verzija 6 (IPv6)
  - Glava IPv6 datagrama



IP (Internetni protokol)

- IP verzija 6 (IPv6)
  - Naslavljanje
    - 128-bitni naslov ($3.4 \cdot 10^{38}$ možnih naslovov)
    - Več možnosti za združevanje naslovov geografsko, po ponudniku storitev, itd.
    - Trije načini naslavljanja
      - Usmerjeno (unicast): en naslovnik
      - Komurkoli v skupini (anycast): nekdo iz skupine (najbližji)
      - Skupinsko (multicast): vsi iz skupine
      - Ni več broadcast nasljavljanja!

IP (Internetni protokol)

- IP verzija 6 (IPv6)
  - Naslavljanje
    - 8 blokov po 16 bitov (šestnajstiški zapis)
    - Primer IPv6 naslova:

      2001:1470:fffe:FE03:0000:0000:abcd:ef12
  -
    - Kadar je so v nekem kosu same ničle, se te lahko izpustijo
    - Primer IPv6 naslova:

      2001:0000:0000:0000:0000:0000:0000:ef12

      2001::ef12

IP (Internetni protokol)

- Pv6 vs. IPv4
  - Prednosti
    - Razširjen naslovni prostor (128-bitni naslovi)
    - Manjše število polj v glavi datagrama
    - Kvaliteta storitve (QoS) – prioritetni nivoji
    - Varnostni mehanizmi
    - Mobilnost
  - Ostaja enako
    - Nepovezavni protokol
    - Nezanesljiv prenos podatkov (best effort)

Protokoli transportne plasti

TCP/IP protokolni sklad

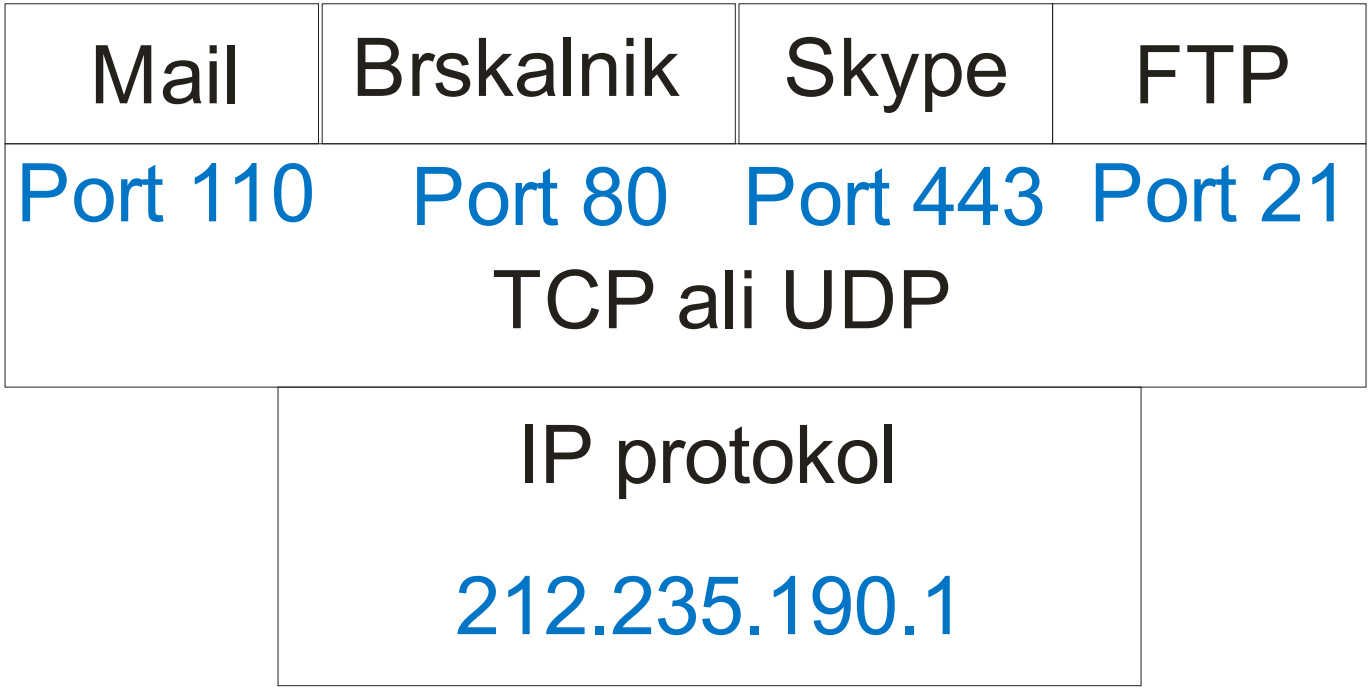
Aplikacijska plast	HTTP, DNS, FTP, POP, SMTP, itd.
Transportna plast	TCP, UDP
Internetna plast	IP, ARP, IPsec, itd.
Povezavna plast	Ethernet II, Token Ring, IEEE 802.X, ATM, GPRS, HSPA, itd.

Protokoli transportne plasti

- Skupne značilnosti
  - Višjim plastem zagotavljajo storitev prenosa podatkov
  - Skrbi za pakiranje podatkov, ki jih dobi od aplikacijskega nivoja (v datagrame)
  - Skrbi za to, aplikacija vsako omrežje pod transportnim nivojem vidi na enak način
  - Z nižjimi plastmi komunicirajo preko številke protokola (TCP=6, UDP=17)
  - Z višjimi plastmi komunicirajo preko številke vrat (port)

Protokoli transportne plasti

- PORT (vrata)
  - Unikatna 16-bitna številka posamezne aplikacije, ki komunicira preko TCP ali UDP protokola



Protokoli transportne plasti

- Seznam tipičnih TCP/UDP vrat

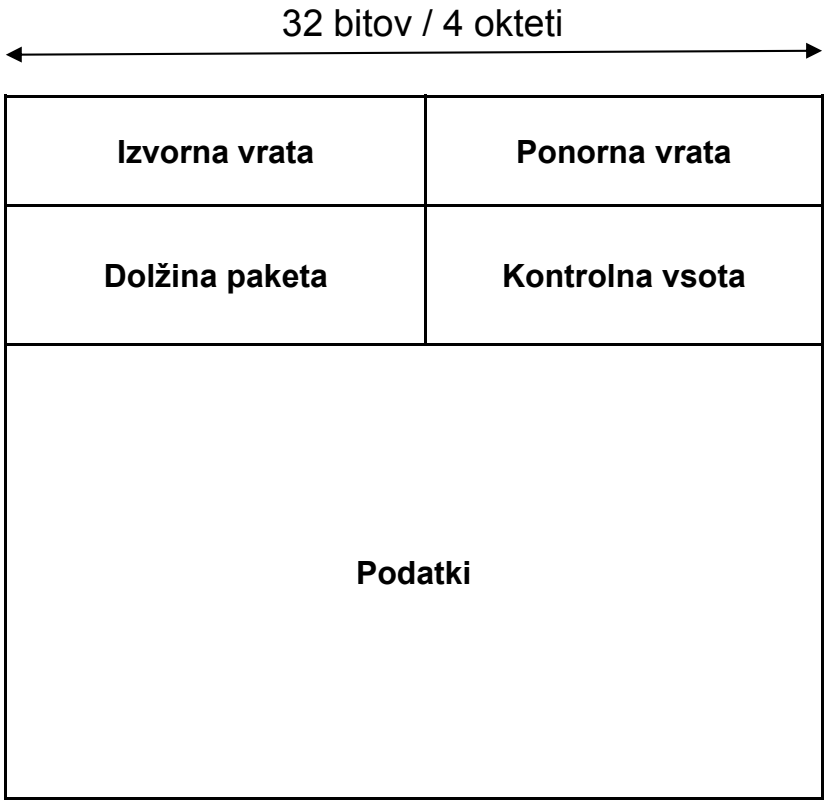
Port (vrata)	Protokol	Aplikacija
20	TCP	FTP
23	TCP	Telnet
25	TCP	El. pošta (SMTP)
53	TCP, UDP	DNS
80	TCP, UDP	HTTP, WWW
110	TCP	El. pošta (POP3)

UDP protokol

- UDP – User Datagram Protocol
  - Nepovezaven in nezanesljiv
  - Preprost in učinkovit
    - Malo režije
      - Ni potrjevanja, nadzora pretoka, označevanja vrstnega reda, itd.
    - Prevzame podatkovno enoto, jo opremi z IP naslovi in odpošlje
  - Najpogostejši primeri uporabe
    - Multimedijske vsebine
    - Internetna telefonija
    - Usmerjevalni protokoli
    - DNS

UDP protokol

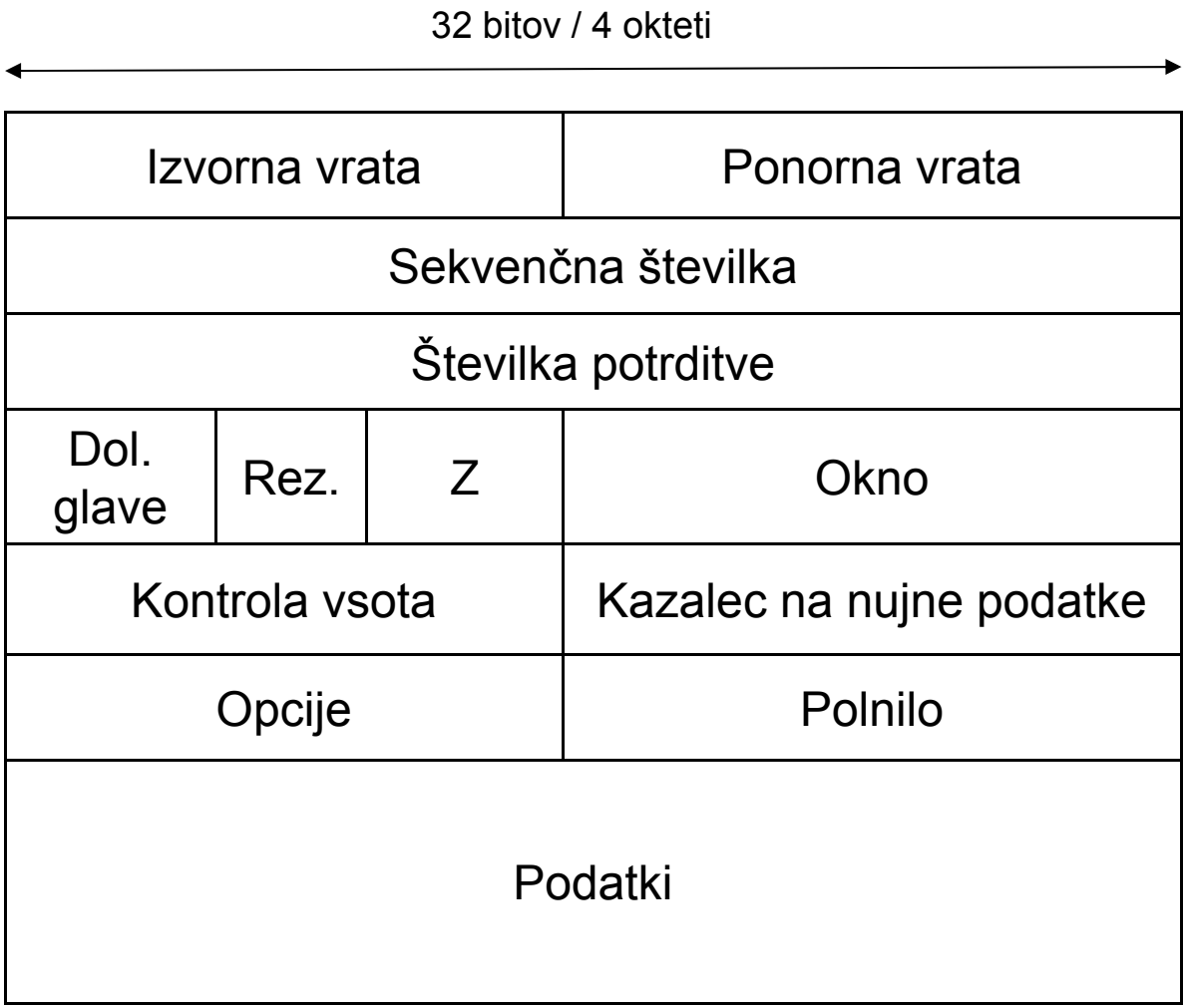
- Glava UDP datagrama



TCP protokol

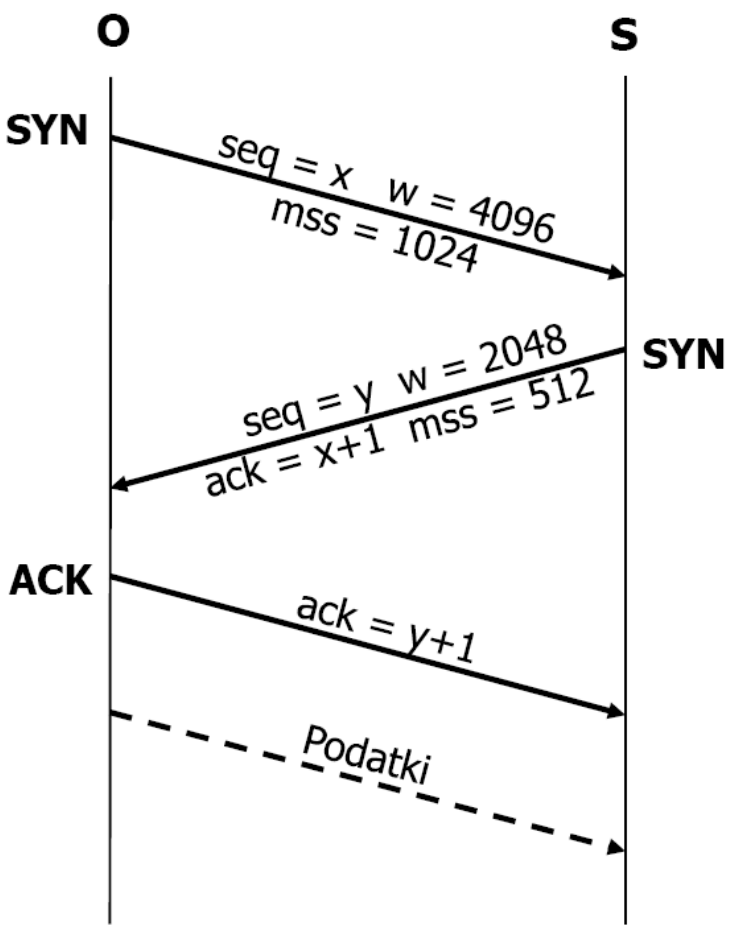
- Transmission Control Procotol (TCP)
  - Povezavno orientiran
    - navidezne povezave
  - Zanesljiv
    - Odkrivanje in odpravljanje napak pri prenosu
    - Dinamičen nadzor nad pretokom podatkov
      - Nadzor nad vrstnim redom paketov (segmentov)

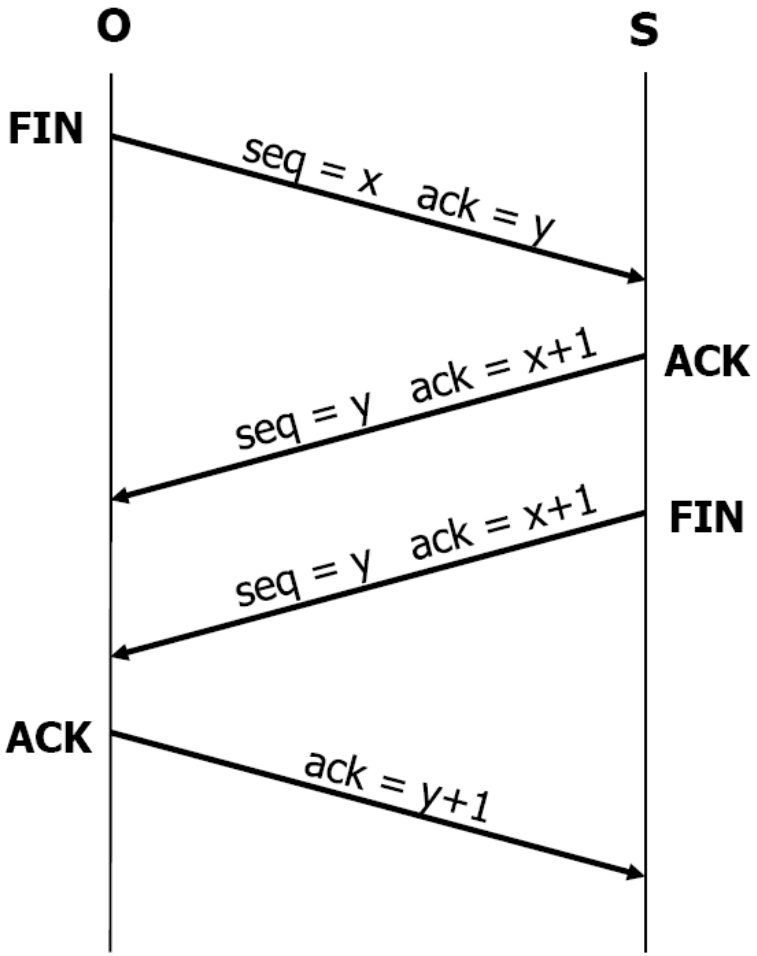
- Glava TCP segmenta



TCP protokol

- Vzpostavljanje in rušenje TCP povezave





- Lastnosti TCP protokola
  - Protokol drsečega okna (sliding window)
    - Določa največje število oktetov, ki jih oddajnik lahko pošlje brez potrditve sprejemnika
  - Počasen začetek (slow start)
    - Na začetku lahko oddajnik pošlje le en segment
    - Z vsako potrditvijo se število segmentov brez potrditve poveča za ena (do velikosti okna)
  - Časovniki
    - Timeout in Retransmission

Protokoli aplikacijske plasti

TCP/IP protokolni sklad

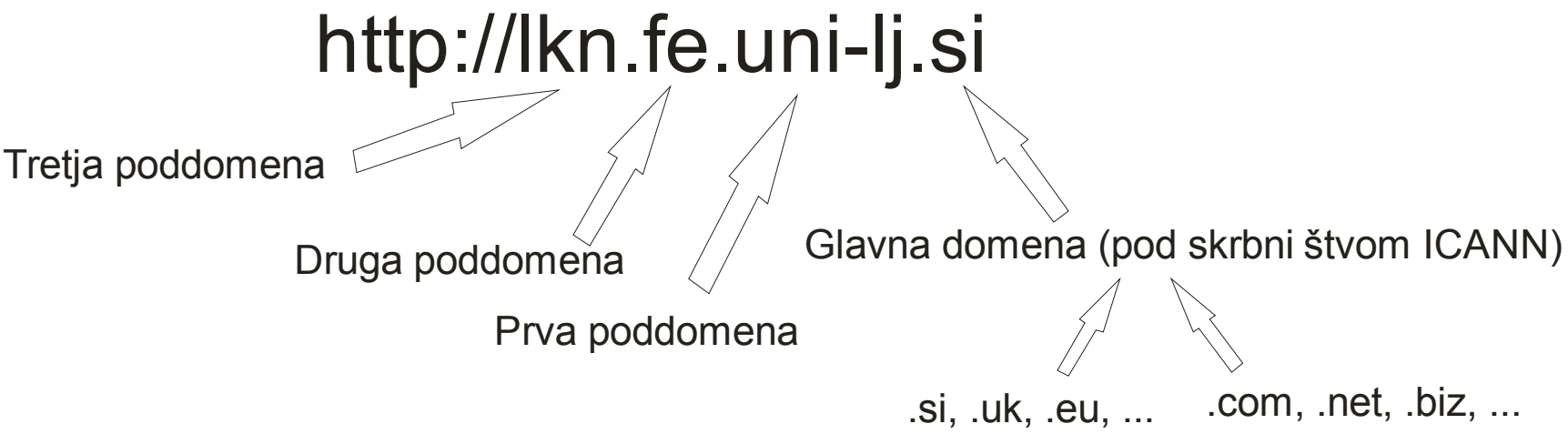
Aplikacijska plast	HTTP, DNS, FTP, POP, SMTP, itd.
Transportna plast	TCP, UDP
Internetna plast	IP, ARP, IPsec, itd.
Povezavna plast	Ethernet II, Token Ring, IEEE 802.X, ATM, GPRS, HSPA, itd.

Protokoli aplikacijskega sloja

- Nad TCP oz. UDP protokolom lahko teče veliko različnih protokolov:
  - HTTP (TCP / 80)
  - FTP (TCP/ 20, 21)
  - DNS (UDP / 53)
  - SMTP (TCP / 25)
  - IMAP (TCP / 143)
  - NFS (UDP, TCP)
  - Telnet (TCP / 23)
  - DHCP (UDP / 67, 68)

DNS protokol

- Računalniki v internetu lahko komunicirajo tudi preko **domenskih imen**



DNS protokol

- Domain Name System
  - Hierarhični sistem za poimenovanje omrežnih naprav, ki komunicirajo preko interneta
  - Skrbi za spreminjanje domenskih imen v IP naslove in obratno
  - Na eni IP številki je lahko več domen!!
  - Primer

http://www.lkn.fe.uni-lj.si ⇨ 212.235.190.202

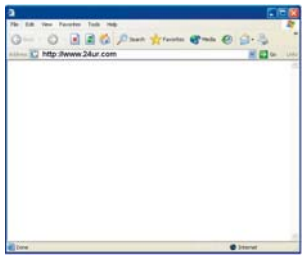
213.253.92.88 ⇨ http://www.najdi.si

HTTP protokol

- HyperText Transport Protocol
- Osnovni protokol za komunikacijo med odjemalcem in strežnikom
- Temelji na principu zahteva-odziv (request-response)
- Večina implementacij HTTP protokola teče preko TCP protokola (vrata 80)
- Tekstovna oblika (ASCII), razen če ni dodatne zaščite (SSL)

HTTP protokol

DNS (UDP)



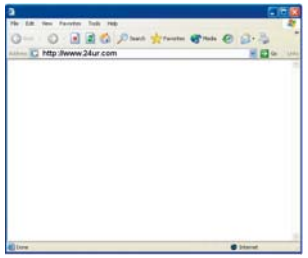
http://www.24ur.com

DNS

91.202.65.130

84.255.109.77

HTTP (TCP)



HTTP REQ. (GET)

24UR

HTTP RESP. (HTML)

91.202.65.130

URL naslov

http://www.fe.uni-lj.si:5431/kolokviji/ocene.txt

HTTP protokol

- HTTP zahteva (request)
  - Začetna vrstica (request-line)
    - Zahtevana metoda
    - URI (naslovnik zahteve)
    - Verzija HTTP protokola
  - Glava
    - …
  - Prazna vrstica (<CR><LF>)
  - Vsebina zahteve (opcijsko)

HTTP protokol

- Začetna vrstica HTTP zahteve
  - Zahtevana metoda
    - **GET**: najpogostejša zahteva za izpis neke internetne strani (preko brskalnika)
    - **POST**: zahteva, ko je strežniku ob zahtevi potrebno poslati določene podatke (izpolnjevanje nekega obrazca preko interneta)
    - HEAD
    - OPTIONS
    - PUT
    - DELETE
    - TRACE

HTTP protokol

- Začetna vrstica HTTP zahteve
  - URI (naslovnik zahteve)
    - Uniform Resource Identifier
    - Tu je ponavadi določena le podstran, včasih lahko tudi celoten naslov
    - Primer: http://www.24ur.com/sport
      - URI: /sport
  - Verzija HTTP protokola
    - Verzija, ki se trenutno uporablja je 1.1

Primer začetne vrstice HTTP zahteve: GET /sport HTTP/1.1

- Glava (header) v HTTP zahtevi
  - Primer: http://www.lkn.fe.uni-lj.si

GET / HTTP/1.1

host: www.lkn.fe.uni-lj.si

user-agent: Mozilla/5.0….

accept: text/html,application/xhtml+xml….

accept-language: en-us,en;…

accept-encoding: gzip, deflate

accept-charset: ISO-8859-1….

keep-alive: 300

connection: keep-alive

HTTP protokol

- MIME (Multiple Internet Mail Extensions) tipi
  - Označuje tipa podatkov v priponkah el. Pošti
  - WWW: Določa tip podatkov, ki je razumljiv odjemalcu
  - Določen kot **tip/podtip**
  - Tipi: application, audio, image, message, text, video, itd.
  - Primeri:

text/html
text/plain
image/gif
image/jpeg
application/octet-stream

HTTP protokol

- HTTP odgovor (response)
  - Statusna vrstica
  - Glava
  - Prazna vrstica (<CR><LF>)
  - Vsebina odgovora (opcijsko)

- HTTP odgovor (response)
  - Statusna vrstica
    - Verzija HTTP protokola
    - Status: koda + opis

Primer začetne vrstice HTTP odgovora: HTTP/1.1 200 OK

HTTP protokol

- Nekaj primerov statusov
  - 200 OK
  - 301 Moved Permanently
  - 307 Temporary Redirect
  - 401 Unauthorized
  - 403 Forbidden
  - 404 Not Found
  - 500 Internal Server Error

Prva cifra	Razred
1	Informational
2	Success
3	Redirection
4	Client Error
5	Server Error

- Glava (header) v HTTP odgovoru
  - Primer: http://www.lkn.fe.uni-lj.si

HTTP/1.1 200 OK

Server: Microsoft-IIS/5.0

Date: Thu, 18 Sep 2008 15:17:20 GMT

Expires: ….

Etag: …

Content-Length: 4389

Content-Type: text/html

Content-Encoding: gzip

Accept-Ranges: ….

Location: …

HTTP protokol

- Lokalno shranjevanje podatkov (cache control)
  - Večina brskalnikov avtomatsko “kešira” določene podatke
  - Problem neveljavnosti podatkov
    - Zahteva z HEAD metodo
    - Uporaba ETag polja
    - Uporaba Expires: polja
  - Ročno brisanje lokalnega pomnilnika

HTTP protokol

- Zaščita komunikacije
  - SSL(Secure Socket Layer)
    - SSL 1.0 (Netscape), 2.0 in 3.0
  - TLS (Transport Socket Layer)
    - TLS 1.0
      - Nadgradnja SSL 3.0
    - TLS 1.1
    - TLS 1.2 (2008)
  - Ščitenje protokolov HTTP, FTP, SMTP, itd.
    - HTTPS (vrata 443)
  - Proces rokovanja (handshake)

2. Spletni strežnik in odjemalec

Spletni strežnik

- Spletni strežnik je računalnik oz. program, ki sprejema HTTP ukaze (request) od odjemalcev, jih obdela in pošlje HTTP odgovor (response)
- Sočasno lahko obdela večje število zahtev
- Na enem fizičnem strežniku (na enem IP naslovu) se lahko nahaja več virtualnih strežnikov
- Na podlagi poddomene določi, kateri virtualni strežnik bo obdelal zahtevo (polje Host v glavi HTTP zahteve)

Spletni strežnik

- Glavne naloge spletnega strežnika
  - Čaka na morebitne zahteve za vzpostavitev TCP zvez
  - Ob zahtevi za vzpostavitev določi t.i. “subtask”, ki bo obdelal zahtevo
  - Vzpostavi TCP zvezo in sprejme HTTP zahtevo
  - Glede na vsebino v glavi HTTP zahteve določi, na kateri virtualni strežnik se zahteva nanaša
  - Obdela morebitno zahtevo za avtentikacijo odjemalca
  - Virtualni strežnik glede na zahtevani URI pripravi HTTP odgovor (samo stran ali s pomočjo programa-skripte)
  - V posebno log datoteko vpiše podatke o odjemalcu, ki je poslal zahtevo
  - Če odjemalec v HTTP glavi zahteva, da se TCP zveza ohrani strežnik čaka na morebitne nove zahteve, sicer se TCP zveza poruši

Spletni strežnik

- Najbolj razširjeni spletni strežniki
  - Microsoft Internet Information Services (IIS)
  - Apache
  - Apache Tomcat
  - Sun Java System Web Server
  - Jigsaw (Java)
  - Zeus (samo za Linux)
  - GWS (Google)
  - Jnode

Spletni strežnik

- Programi na spletnem strežniku (server-side scripting)
  - ASP, PHP, JSP, Perl, itd.
  - Rezultat programov je dinamično generirana HTML stran
  - Možnost povezovanja z različnimi podatkovnimi bazami
  - Zahtevne aplikacije, ki nadalje komunicirajo z nekim sistemom (primer: J2EE aplikacije)

Apachi spletni strežnik

- Na voljo za operacijske sisteme UNIX, Linux, Mac OS in Windows
- Programski paket XAMPP

- Osnovne značilnosti
  - Spletni dokumenti (**…/htdocs/**)
  - Konfiguracija (**…/conf/httpd.conf**)
    - Izvorna mapa (“root directory”)
    - Virtual hosts
    - “Alias”-i

Spletni odjemalec

- Spletni odjemalec je računalnik oz. program, ki pošlje HTTP zahtevo na nek spletni strežnik in potem sprejme odgovor
- Primeri spletnih odjemalcev
  - Uporabniški agent (user agent)
  - Spletni brskalnik
  - Programski roboti

Spletni brskalnik

- WorldWideWeb (Tim Berners-Lee, 1991)
- Eden prvih grafičnih brskalnikov Mosaic (1993)

Spletni brskalnik

- “Vojne” med razvijalci
  - IE in Netscape (1997)
  - Nove funkcionalnosti / hrošči
  - Specifične lastnosti (“Best viewed in …”
    - Nastanek projekta Mozilla
- Spletni brskalniki (engine)
  - Microsoft Internet Explorer (Trident)
  - Mozilla Firefox (Gecko)
  - Opera (različice za mobilne naprave) – (Presto)
  - Safari (WebKit)
  - Chrome (WebKit)
    - Chrome OS

Spletni brskalnik

- Glavne operacije spletnega brskalnika
  - Preoblikovanje zahtevanega URL (URI) naslova v HTTP zahtevo
    - HTTPS, FTP, FILE
  - Po potrebi sprememba domenskega imena v IP naslov (s pomočjo DNS protokola)
  - Vzpostavitev TCP povezave z izbranim strežnikom
    - Več povezav hkrati (z različnimi strežniki)
  - Pošiljanje HTTP zahteve na strežnik
  - Prikaz vsebine
    - Zavihki (Tabs)
    - Pojavna okna (Pop-up)

Spletni brskalnik

- Uporabniški vmesnik
  - URL okno
  - Iskalno okno
  - Navigacijski gumbi
  - Statusno okno
  - Zgodovina (piškotki)
  - Zaznamki (sinhronizacija)
  - Vtičniki
  - ***Prikaz izvorne kode***
  - ***Razhroščevalnik***
- Varnost??

Spletni brskalnik

- Predstavitev oz. prikaz dokumenta (web browser engine)
  - Določanje vsebine
    - HTML
    - XML
    - Slike
  - Določanje oblike
    - CSS
  - Programski jeziki
    - JavaScript
  - Vtičniki (Plugins)
    - Flash

3. HTML
(Hyper Text Markup Language)

Označevalni jeziki

- Označevalni jezik (Markup Language)
  - Umetni jezik, ki s pomočjo nekih opomb (značk) podaja informacije o strukturi nekega besedila ali navodila za prikaza le-tega
- Primeri označevalnih jezikov
  - Tex: za oblikovanje tehničnih besedil, ki vsebujejo enačbe in druge matematične zapise
  - SGML
  - XML
  - HTML

Označevalni jeziki

- Osnovni pojmi označevalnih jezikov
  - **Sintaksa** (skladnja): določa nabor besed oz. značk, ki so na voljo v nekem označevalnem jeziku
    - Abstraktna in konkretna sintaksa
  - **Semantika** (pomenoslovje): določa pomen posameznih besed oz. značk
  - **Metajezik**: jezik za opis sintakse nekega drugega jezika
- SGML, XML
  - Metajezika, s pomočjo katerih so opisane posamezne različice jezika HTML

Označevalni jeziki

- SGML
  - Standard Generalized Markup Language
  - Metajezik za opisovanje označevalnih jezikov
  - Določa le pravila abstraktne sintakse označevalnega jezika
  - Konkretna sintaksa je določena s pomočjo DTD datotek (Document Type Definition)
  - Določa uporabo značk za opis jezika
    - Primer: <nekaj></nekaj>
  - Zaključne značke niso nujno potrebne
  - Uporabljen za definicijo različice HTML 4.01

Označevalni jeziki

- XML
  - Extensible Markup Language
  - Definiran s pomočjo SGML jezika
  - Razširljiv
    - Možnost dodajanja novih značk
  - Stroga sintaktična pravila
    - Obvezne zaključne značke
  - Enostavna interpretacija (parsanje) dokumenta
  - Namenjen hranjenju različnih tipov besedil ali podatkov, ki imajo drevesno strukturo
  - Uporabljen za definicijo različice XHTML 1.0

Različice jezika HTML

- HTML 1.0 (1993)
  - Razvit skupaj s prvim grafičnim brskalnikom (Mosaic)
  - Podpira le najbolj osnovne ukaze za oblikovanje besedil
- HTML 2.0 (1995)
  - Vsebuje vse elemente prejšnje različice in doda nove
- HTML 3.0 (1995)
  - Veliko dodatnih možnosti, a brez podpore v brskalnikih
  - Glavna brskalnika: Netscape in Microsoft
- HTML 3.2 (1997)
  - Ustanovitev W3C (World Wide Web Consortium)
  - Očiščena verzija jezika HTML
  - V celoti podprt v vseh današnjih brskalnikih

Različice jezika HTML

- HTML 4.0 (1998)
  - dodana podpora prekrivnim slogom CSS
  - vodilno vlogo med brskalniki prevzame Microsoft IE
- HTML 4.01
  - definiran s pomočjo jezika SGML
- XHTML 1.0
  - definiran s pomočjo jezika XML
  - obvezne zaključne značke
  - obvezne vrednosti lastnosti elementov
  - ločevanje med malimi in velikimi črkami
- XHTML 1.1
- XHTML 2.0
- **HTML5**

Uvod v HTML

- HTML - Hyper Text Markup Language
- Določa osnovno strukturo spletne strani
- HTML: tekstovna datoteka, ki jo lahko urejamo s katerim koli urejevalnikom
  - končnica **.html** (DOS različica **.htm**)
- Ogled HTML datoteke:
  - Desna miškina tipka -> View source

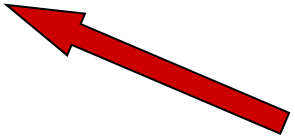
Osnovni gradniki spletne strani

- Značka (tag)
  - Značka se začne z znakom **<** in konča z znakom **>**
    - Primer: **<značka>**
  - Samostojne značke: **
**
  - Začetne in končne značke: **** vsebina ****
- Element: osnovni gradnik spletne strani
  - <h1> ***vsebina*** </h1>
- Lastnost (atribut)
  - ime = “vrednost”

```
<a ref=“http://www.najdi.si" >
…
</a>
```

Primer HTML datoteke

```
<!DOCTYPE html
        PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3org/1999/xhtml">
        <head>
                <title>
                  HelloWorld.html
                </title>
        </head>
        <body>
                <p>
                  Hello World!
                </p>
        </body>
</html>
```



Označevalna informacija: navodila brskalniku

Podatki, ki jih mora brskalnik prikazati

Značilnosti HTML sintakse

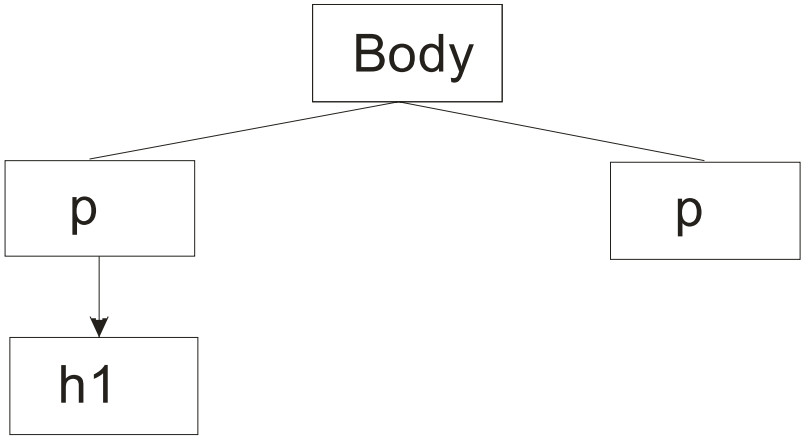
- Deklaracija tipa dokumenta <!DOCTYPE>
  - Nastopi pred glavo
  - Trije tipi dokumentov HTML 4.01 in XHTML 1.0
    - Strict DTD: kadar sta vsebina in prikaz ločena (kadar uporabljamo stile – CSS)
    - Transitional DTD: kadar je poleg vsebine v HTML datoteki tudi način prikaza (kadar brskalnik ne podpira uporabe stilov)
    - Frameset DTD: kadar uporabljamo okvire – “frameset“

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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN“
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!DOCTYPE html>
```

Značilnosti HTML sintakse

- Gnezdenje elementov
  - En HTML element predstavlja vsebino drugega HTML elementa
  - S pomočjo gnezdenja oblikujemo drevesno strukturo
  - V določenih primerih veljajo posebna pravila gnezdenja (seznami, tabele, okviri, itd.)

<body> <h1> … </h1> … </body>



Značilnosti HTML sintakse

- Neupoštevanje večkratnih presledkov
  - Večkratni presledki, nove vrstice in tabulatorji se upoštevajo kot enkratni presledki

Primer: spodnji dve sintaksi brskalnik prikaže na enak način:

```
<p>
    Prva vrstica.
    Druga vrstica.
</p>

<p>
    Prva vrstica. Druga vrstica.
</p>
```

Značilnosti HTML sintakse

- Neupoštevanje neznanih elementov
  - Brskalnik vedno skuša prikazati celotno vsebino dokumenta
  - Če ne prepozna določenega elementa, prikaže vsebino z neupoštevanjem le-tega
  - Neupoštevanje sintaktičnih napak
  - Pomembno za kompatibilnost novih in starih verzij HTML

Primera:

```
<head>
  <titlee>
        Naslov dokumenta
  </title>
</head>
```

```
<body>
  <k>
        Vsebina
  <p>
</body>
```

Značilnosti HTML sintakse

- Posebni znaki (entitete)
  - Določeni znaki (npr. <) imajo v HTML datoteki poseben pomen in jih ne moremo uporabiti v besedilu
  - Za izpis takšnih znakov uporabljamo t.i. entitete (entity)
  - Entiteta je zapisana kot: &#*stevilka*; ali &*ime*;

Znak	Ime	Številka
<	&lt;	&#60;
>	&gt;	&#62;
&	&amp;	&#38;
“	&quot;	&#34;

Značilnosti HTML sintakse

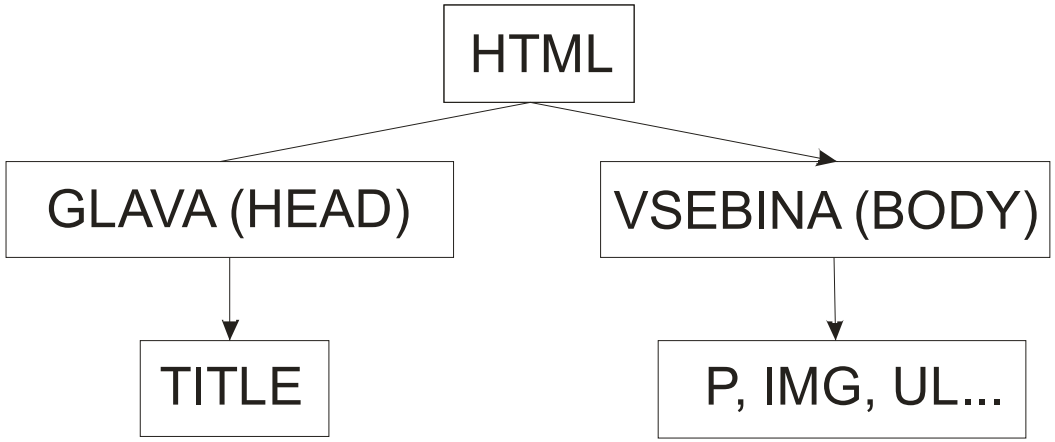
- Preverjanje pravilnosti sintakse

  http://validator.w3.org/

Zgradba HTML strani

- Drevesna struktura
- Osnovni elementi strani:
  - HTML, HEAD, BODY

```
<html>
    <head>
        …
    </head>
    <body
        …
    </body>
</html>
```



Glava - <head>

- Splošne informacije o dokumentu
  - **<title>** NASLOV **</title>**
  - **<base>** : skupni del spletnega naslova in privzeto okno za povezave

```
<base href="http://www.w3schools.com/images/" />
<base target="_blank" />
```

  - **<link>** : lastnosti povezave dveh dokumentov (za vključevanje stilov)

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

Glava - <head>

- Druge značke, ki se uporabljajo v glavi:
  - **<meta>** :dodatne informacije o strani (kodne tabele, ključne besede namenjene iskalnikom, podatki o osvežitvi strani, itd.)

```
<meta name="keywords" content= "UNI-LJ, FE, TK" />
```

  - **<script>** : vključevanje programčkov v HTML datoteko
    - Značke <script> se lahko nahajajo tudi izven glave HTML datoteke

```
<script type="text/javascript">
   document.write("Hello World!")
</script>
```

Vsebina - <body>

- Vsebuje vso glavno vsebino spletne strani

  <body>

  …

  </body>

- Brskalnik prebere vsebino spletno strani od vrha navzdol in od leve proti desni
- Za oblikovanje glavne vsebine in elementov imamo na voljo veliko število različnih značk

HTML značke

- **Naslovi**

  <h1>…</h1>, <h2>…</h2>, … , <h6>…</h6>
- **Odstavki**

  …
- **Prelomi vrstic**

 ali
- **Komentarji**:

  <!-- *komentar* -->
- **Predhodno oblikovano besedilo**

  <pre>… </pre>

- Seznami
  - Naštevanje (unordered list)

```
<ul>
    <li> izbira 1 </li>
    <li> izbira 2 </li>
</ul>
```

  - Oštevilčeno naštevanje (ordered list)

```
<ol>
    <li> prva postavka </li>
    <li> druga postavka </li>
</ol>
```

- Povezave
  - Povezava lahko kaže tudi na sliko, zvočno datoteko, filmsko datoteko, itd.

```
<a href="http://www.najdi.si" target="_blank"> Slovenski iskalnik </a>

<a href="#fokus"> Del strani FOKUS </a>

<a href="mailto:jaka@sodnik.com?subject=Vaja"> Mail Jaku </a>

<img src="slika.jpg" alt="Manjka slika"> Primer slike </img>
```

- Tabele

```
<table border="1" frame="box">
    <tr>
        <td>Vrstica 1, Stolpec 1 </td>
        <td>Vrstica 1, Stolpec 2 </td>
    </tr>
    <tr>
        <td>Vrstica 2, Stolpec 1 </td>
        <td>Vrstica 2, Stolpec 2 </td>
    </tr>
    <tr>
        <td colspan="2" align="center">Vrstica 3</td>
    </tr>
</table>
```

HTML značke

- Okviri (frames)
  - **<frameset>,<frame>**
  - **<iframe>**

```
<html>
<head> … </head>
    <frameset cols="30%, 70%">
        <frameset rows=“2*,3*”>
                    <frame src="…" name=“levo_zgoraj">
                    <frame src="…" name=“levo_spodaj">
        </frameset>
        <frame src="…" name=“desno">
    </frameset>
</html>
```

HTML značke

- Določanje strukture strani
  - **<p>**
  - ** **
  - ****
- Multimedija
  - **<embed>**
  - **HTML5**
    - **<audio>**
    - **<video>**
    - **<canvas>**

Dodatne lastnosti HTML elementov

- Nabor posebnih lastnosti, ki jih lahko določimo vsem HTML elementom
  - class: elementu določimo nek razred, ki določa način prikaza elementa
  - style: določanje stila oz. načina prikaza elementa
  - name: ime HTML elementa, ki se uporablja pri pošiljanju vrednosti spremenljivk na strežnik
  - id: unikatno poimenovanje HTML elementa, da se kasneje lahko nanj sklicujemo (iz programčkov na odjemalcu)
  - title: pojasnilo o pomenu elementa, ki se prikaže uporabniku, ko premakne miško nad element

- Vnosni obrazci
  - <form>
    - name, action, method (GET ali POST)

```
<form name="oseba" action="obdelava.asp" method="post">
```

    - enctype (le pri uporabi metode POST)

      application/x-www-form-urlencoded

      multipart/form-data

      text/plain

- Vnosni obrazci
  - <input>
    - Atributi
      - Type
        - HTML 4.01: text, password, hidden, button, submit, reset, image, file, checkbox, radio, …
        - ***HTML 5: tel, search, url, email, datetime, number, range, color, image***
      - name
      - value
      - src, alt
      - maxlength, size, disabled

HTML značke

- Novi atributi (HTML5)
  - autofocus — autofocus
  - autocomplete — on, off
  - placeholder — “poljubno besedilo”
  - required — required
  - pattern — pattern="[A-Z]{3}[0-9]{4}“
  - formnovalidate
  - formaction / formenctype / formmethod / formtarget (na *input* ali *button* elementih)

POZOR: Večina novih atributov ni podprta v IE!

- Vnosni obrazci
  - Primeri

```
<input type="text" name="priimek" value="Sodnik“ />

<input type=“password" name=“geslo” />

<input type="radio" name="cifra" value="1“ checked=“checked” /> 1
<input type="radio" name="cifra" value="2“ /> 2

Mazda: <input type="checkbox" name=“znamka" value=“mazda“ />
Ford:  <input type="checkbox" name=" znamka " value=“ford“ />
VW:  <input type="checkbox" name=" znamka " value=“vw“ />

<select name="telefoni“ multiple="multiple">
   <option value="Nokia"> Nokia </option>
   <option value="SE"> Sony Ericsson </option>
   <option value="Samsung"> Samsung </option>
</select>
```

- Vnosni obrazci
  - Primeri

```
<textarea name="besedilo" rows="5" cols="15">
  … besedilo…
</textarea>

<input type="button" name="gumb" value="" onclick="…">

<input type="reset" name="gumb" value="">


<label for="predmet">Predmet:</label>
<input type="text" name="predmet" id="predmet"/>

<input type="file" name="avatar" id="avatar" />
```

Vnosni obrazci

Primeri

EMŠO: <input type="text" name="emso" autofocus="autofocus" />

Koda države: <input type="text" name="ime" pattern="[A-Za-z]{3}" title="Trimestna koda države" />

```
<form action="obdelaj.php" method="get">
   Ime: <input type="text" name="ime" /><br />
   Priimek: <input type="text" name="priimek" /><br />
   <input type="submit" value="Poslji" />
   <input type="submit" formmethod="post"
          formaction="obdelaj_post.php" value="Poslji POST" />
</form>
```

HTML značke

- Multimedija
  - Podpora in način implementacije je odvisen od brskalnika
  - Video formati
    - AVI (.avi), WMV (.wmv), MPEG (.mpg, .mpeg), QuickTime (.mov), RealVideo (.rm, .ram), Flash (.swf, .flv), Mpeg-4 (.mp4)
  - Avdio formati
    - MIDI (.mid), RealAudio (.rm, .ram), Wave (.wav), WMA (.wma), MP3 (.mp3, .mpga)

- Avdio

```
<audio controls="controls" height="50px" width="100px">
    <source src=“glasba.mp3" type="audio/mpeg" />
</audio>


<embed height="50px" width="100px" src="song.mp3" />


 <audio controls="controls" height="50px" width="100px">
     <source src=“glasba.mp3" type="audio/mpeg" />
     <embed height="50px" width="100px" src=" glasba.mp3 " />
 </audio>


<a href=“glasba.mp3 ">Play Song</a>

<script type="text/javascript" src="http://mediaplayer.yahoo.com/js">
</script>
```

- Video

```
<video width="320" height="240" controls="controls">
    <source src=“film.mp4" type="video/mp4" />
    <source src="film.webm" type="video/webm" />
   <object data="film.mp4" width="320" height="240">
      <embed src="film.swf" width="320" height="240" />
   </object>
</video>
```

  - Youtube

```
<embed width="420" height="345"
src="http://www.youtube.com/v/PQqVEjypglE"
type="application/x-shockwave-flash">
</embed>
```

Dogodki povezani z HTML elementi

- S pomočjo dogodkov (events) vključimo v prikaz strani programčke (skripte)
  - onload
  - onchange
  - onsubmit
  - onfocus
  - onclick
  - onmouseover
  - ltd.

<a href=www.google.com onmouseover="alert("To je link na Google");return true">

1_HTML_primeri.rar

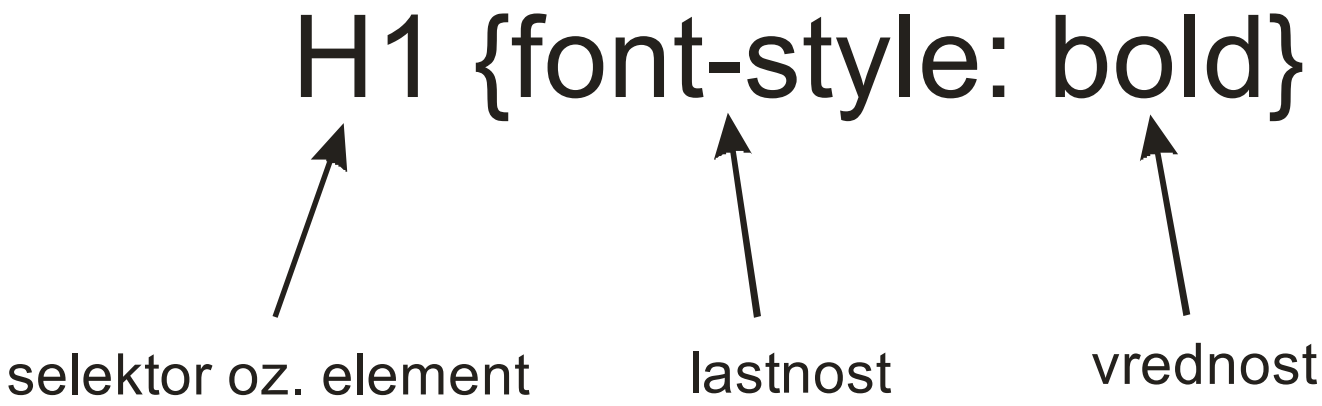
4. Uporaba stilov v HTML

CSS – Cascading Style Sheets

Splošno o stilih

- S pomočjo uporabe stilov ločimo način predstavitve informacije (predstavitev) od vsebine informacije (semantika)
- Stili določajo način prikaza posameznih HTML elementov
- Stili so lahko vsebovani v HTML datoteki sami ali v ločeni css datoteki
- Možna je uporaba skupnega stila (skupnih css datotek) za več spletnih strani

Osnovna sintaksa za določanje stila



Selektor: HTML element, kateremu želimo določiti stil

Lastnost: lastnost HTML elementa, ki jo želimo določiti

Vrednost: vrednost

PRIMER: p {font-familiy: “arial”}

Osnovna sintaksa za določanje stila

- Primeri selektorjev: **tip** elementa

```
h1
{
   color: blue;
   font-weight: bold;
   text-align: left
}
```

```
p
{
   font-size: smaller; background-color: yellow
}
```

```
h1, h2, h3, h4
{
   color: red
}
```

```
h1
{
/* Tole je stil za element h1*/
color: blue;
font-weight: bold;
text-align: left
}
```

```
* {font-weight: bold}
```

Osnovna sintaksa za določanje stila

- Primeri selektorjev: razredi (class)

Določitev:

```
.debelo {font-weight: bold}
```

Ime razreda se ne sme začeti s številko!!

Uporaba:

```
<p class="debelo"> ….. </p>
<h2 class="debelo"> … </h2>
```

Osnovna sintaksa za določanje stila

- Primeri selektorjev: **razred** (class)

Določitev:

```
p.debelo {font-weight: bold}
p.normalno {font-weight: normal}
p.centralno {text-align: center}
```

Uporaba:

```
<p class="debelo"> ….. </p>
<p class="normalno"> … </p>
<p class="normalno centralno"> … </p>
```

Osnovna sintaksa za določanje stila

- Primeri selektorjev: **id** elementa

Določitev:

```
#moj_element {color: red}
```

ali

```
div#moj_element
{
color: red;
font-weight: bold
}
```

ID se ne sme začeti s številko!!

```
<div id="moj_element">
…..
</div>
```

0_Selektorji.html

Pseudo-razredi

- Primeri selektorjev: **psevdo razredi** (pseudo-class)
- Sintaksa:

selektor: pseudo-razred {lastnost: vrednost}

Primer: a: link {color: green}

selektor.razred: pseudo-razred {lastnost: vrednost}

Primer: a.black: visitied {color: red}

```
a:link {color: black}       /*nobiskana povezava*/
a:visited {color: red}      /*obiskana povezava*/
a:hover {color: green}      /*miška nad povezavo*/
a:active {color: blue}      /*izbrana povezava*/
```

Pseudo-razredi/elementi

- Seznam pseudo razredov
  - :active: aktiviran element
  - :focus: element s fokusom (aktivni element)
  - :hover: ko je miškin kazalec nad elementom
  - :first-child: prvi otrok določenega elementa
  - :lang: določanje drugega jezika za določen element
- Seznam pseudo elementov
  - :first-letter: poseben stil za prvo črko besedila
  - :first-line: poseben stil za prvo vrstico besedila
  - :before: vključevanje določene vsebine pred nek element
  - :after: vključevanje določene vsebine za element

Naprednejša sintaksa za določanje stila

- Določanje elementa in podelementov (relacija oče – potomec)

Določitev:

```
div b {color: blue}
```

Primer:

```
<div>
    <h1> Tole je pomemben <b>naslov</b>.</h1>
    <b> Tole je debelo in modro.</b>
<div>
```

Rezultat: Tole je pomemben **naslov**. **Tole je debelo in modro.**

Naprednejša sintaksa za določanje stila

- Določanje elementa in podelementov (relacija oče – otrok)

Določitev:

```
div > b {color: blue}
```

Primer:

```
<div>
    <h1>Tole je pomemben<b>naslov</b>.</h1>
    <b>Tole je debelo in modro.</b>
<div>
```

Rezultat: Tole je pomemben **naslov**. **Tole je debelo in modro.**

Naprednejša sintaksa za določanje stila

- Drugi primeri napredne sintakse z združevanjem elementov
  - div:first-child — prvi otrok od div
  - p1 + p2 — za p2 direktno sledi p1
  - h1[color] — če je določena lastnost color
  - h1[color="red"] — če je lastnost h1 določena na red
  - div[name~="ab"] — če lastnost name vsebuje "ab"
  - :first-of-type, :last-of-type, :only-child, :nth-child(n), :empty, :enabled, :disabled, :checked, …

Preverjanje pravilnosti sintakse

- CSS validator

http://jigsaw.w3.org/css-validator/

Vključevanje stila v HTML dokument

- Informacije o stilih se lahko nahajajo v
  - **Privzete** nastavitve brskalnika
  - **Glavi** HTML datoteke z uporabo značke style (embedded style)
  - **Zunanji** css datoteki
    - Možnost uporabe v več HTML dokumentih
    - Celostna podoba spletnih strani podjetij in ustanov
  - Znački posameznega HTML elementa z uporabo atributa **style** (inline style)
    - Določanje oblike ni več ločeno od vsebine
    - Uporabimo če želimo preklicati obstoječi stil za nek element in ga določiti posebej za ta element

Vključevanje stila v HTML dokument

```
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="zunanji_stil.css" />
    <style type="text/css">
      @import url("tudi_zunanji.css");
      table {border-collapse: collapse}
      div {color: black; text-align: left}
      body {background-image: url("slike/ozadje1.jpg")}
    </style>
  </head>
  <body>
      <h2>Naslov vsega skupaj</h2>
      <div id="prvi">
      Tole je eno besedilo
        <p style="color: blue; font-family: arial">
          Za ta del velja poseben stil
        </p>
      </div>
  </body>
  <html>
```

Določanje tipa uporabniškega agenta (medija)

- S pomočjo lastnosti media lahko določimo način prikaza za posamezno vrsto odjemalca
- Vrste odjemalcev
  - all: vsi odjemalci
  - aural: sintetizatorji govora
  - braille: taktilne naprave
  - handheld: mobilni telefoni, PDAji
  - print: tiskalniki
  - projection: projektorji, veliki zasloni
  - screen: računalniški zasloni
  - tv: TV zasloni

Določanje tipa uporabniškega agenta (medija)

```
<style>
        @media screen
         {
        p.test {font-family:sans-serif; font-size:14px;}
        }

        @media print
         {
        p.test {font-family:serif; font-size:10px;}
         }

        @media all
        {
         p.test {font-weight:bold;}
        }
</style>
```

Prioritete stilov

- Prioritete stilov (ko je določenih več stilov za isti element)
  - Če ni nobene definicije veljajo privzete nastavitve brskalnika
    - Spreminjanje privzetega stila
  - Avtorski stili
    - Zunanji, v glavi ali na sami znački (velja zadnja nastavitev)
    - Uporaba uteži **!important**
  - Izbira uporabniškega agenta
    - Ročna izbira ali na osnovi “media query-a”

1_Prioritete.rar

Prioritete stilov

- Primer

```
<style type="text/css">
      p {color: black}
</style>
<link rel="stylesheet" type="text/css" href="zunanji1.css" />
<style type="text/css">
      p {color: red}
</style>
```

Vsebina datoteke **zunanji1.css:**

```
@import url("zunanji2.css");
p {color:blue}
```

Vsebina datoteke **zunanji2.css:**

```
p {color: green}
```

```
p {color: black}
p {color: green}
p {color: blue}
p {color: red}
```

Vsebina elementa p bo obarvana rdeče!

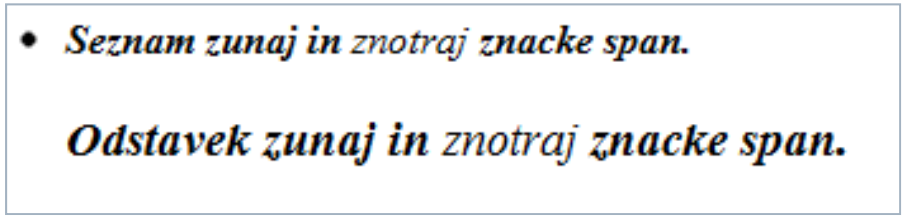
Dedovanje stilov

- Stil se v večini primerov deduje od starševskega elementa
  - Izjeme, npr. “height”
- Absolutna ali relativna definicija stila
  - Primer: “2 cm” ali “larger”
- Določena, izračunana in dejanska vrednost
- Uporaba lastnosti **“inherit”**

```
.ena {
  background-color: white;
  color: black;
}

.dva{
  background-color: inherit;
  color: inherit;
  font-weight: normal;
}
```

```
<div class=“ena">
  <p class=“dva">
    Tole je en odstavek!
  </p>
</div>
```

- Primer




```
<html>
  <head>
    <style type="text/css">
       body { font-weight: bold }
       li { font-style: italic }
       p { font-size: larger }
       span { font-weight: normal }
    </style>
  </head>
  <body>
       <ul>
          <li>
             Seznam zunaj in <span> znotraj </span> značke span.
             <p>
                Odstavek zunaj in <span> znotraj </span> značke span.
             </p>
          </li>
       </ul>
  </body>
  <html>
```

Pisave (fonti)

- Pisava (“font”)
  - Preslikava nekega znaka v njegovo vizualno predstavitev
- Družina pisav (“font familiy”)
  - Zbirka sorodnih oz. podobnih si pisav
  - Pisave v določeni družini pisav se med seboj razlikujejo v debelosti, nagnjenosti, a imajo podoben izgled

PrImeR Pisav

Primer Arial

Pisave (fonti)

- Določimo lahko več družin pisav v želenem vrstnem redu

  font-family: “Times New Roman”, “arial”, serif

  - Brskalnik uporabi prvo znano pisavo med naštetimi
  - Zadnja našteta pisava naj bo generična
  - Generične družine pisav:
    - Serif, Sans-serif, Cursive, Fantasy, Monospace
    - Privzeta pisava znotraj generične družine je nastavitev v brskalniku

Pisave (fonti)

- Spletni fonti (web fonts)
  - @font-face

http://www.font-face.com

```
@font-face { font-family: Delicious; src: url('Delicious-Roman.otf'); }

@font-face { font-family: Delicious; font-weight: bold; src: url('Delicious-Bold.otf'); }

@font-face { font-family: DeliciousRoman;
src: url(http://www.font-face.com/fonts/delicious/Delicious-Roman.otf);}
```

  - Google web fonts

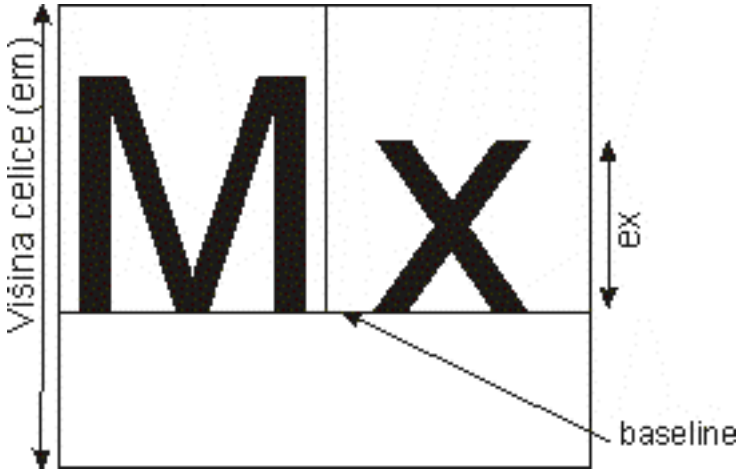
http://www.google.com/webfonts#ChoosePlace:select

```
<link href='http://fonts.googleapis.com/css?family=Codystar'
rel='stylesheet' type='text/css'>

font-family: 'Codystar', cursive;
```

2_Fonti.rar

Določanje dolžine ali velikosti

- Dolžina ali velikost je določena s številko in mersko enoto (primer: “height: 20px”)
- Merske enote
  - in – inč
  - cm – centimeter
  - mm – milimeter
  - pt – točka
  - pc – pica
  - px – pika (piksel)
  - em, ex: vezano na velikost pisave



1 in = 2.54cm = 25.4mm = 72pt = 6pc = 96px

Pisave (fonti)

- Velikost pisave določimo s pomočjo lastnosti

  font-size: …
- Možnosti določanja velikosti pisave
  - Absolutno: font-size: 10px
  - Relativno glede na očetovski element: font-size: 85%
  - S pomočjo absolutne ključne besede:
    - xx-small, x-small, small, medium, large, x-large, xx-large
    - Razlika med velikostmi je približno 20%
  - S pomočjo relativne ključne besede:
    - smaller, larger

Pisave (fonti)

- Ostale nastavitve pisave
  - font-familiy, font-size
  - font-style (normal, italic, oblique)
  - font-variant (normal, small-caps)
  - font-weight (normal, bold, bolder, lighter, številka (100, 200, …)
  - line-height (1.2em, 120%, 80%)
  - font (…)

font: bold, italic, normal, 14pt,1.5, “Arial”, sans-serif

{ font-weight: bold;
font-sytle: italic;
font-variant: normal;
font-size: 14pt;
line-height: 1.5;
font family: “Arial”, sans-serif }

Določanje barve

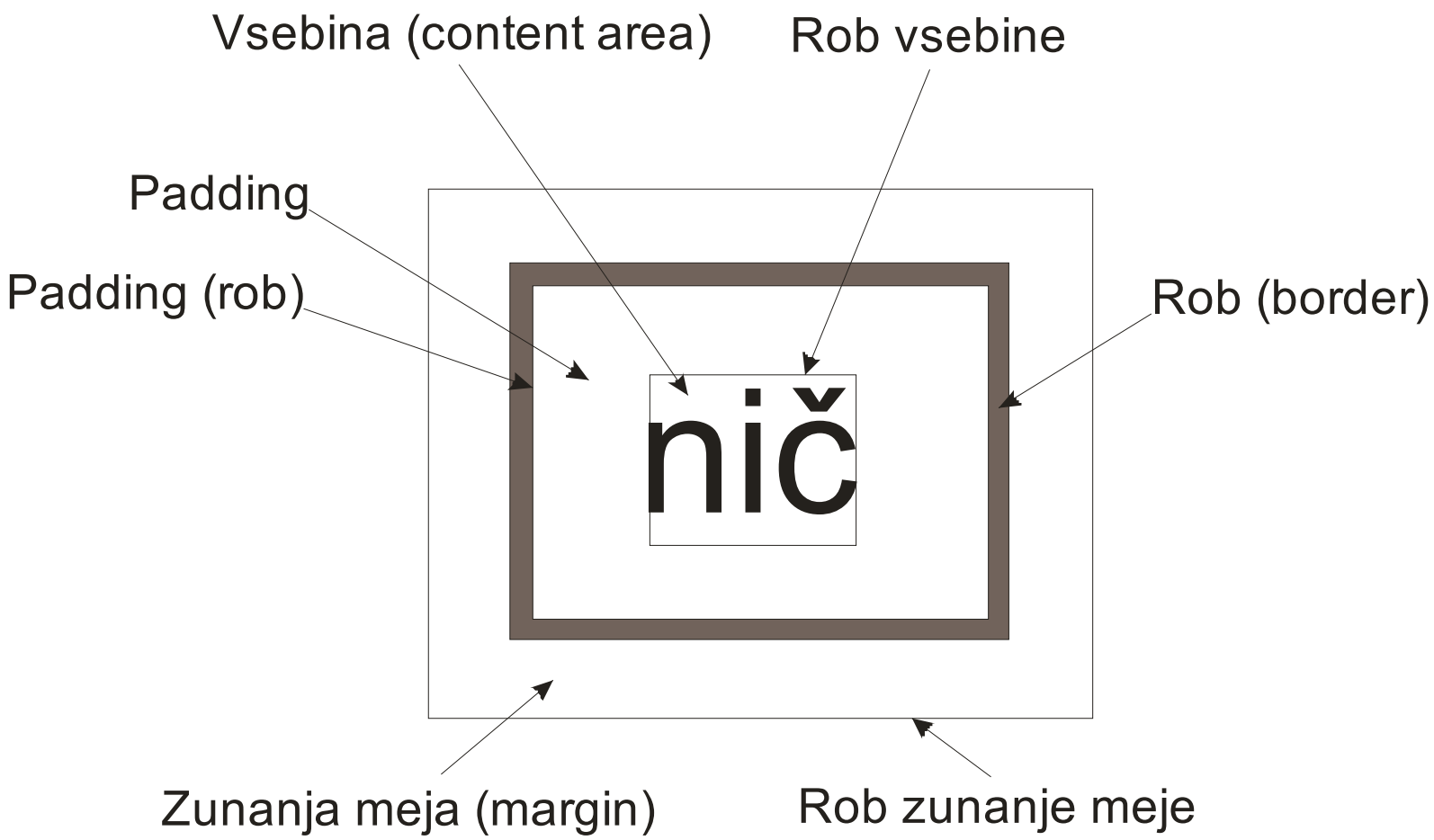
- Barva se nastavi z določitvijo intenzitete treh komponent: rdeče, zelene in modre
- Trije načini določanja
  - Desetiški zapis: rgb(255,170,0)
  - V odstotkih: rgb(100%, 66.7%, 0%)
  - Šestnajstiški (hexa): #ffaa00
  - Skrajšani zapis: #fa0
  - Z imenom: black

Določanje barve

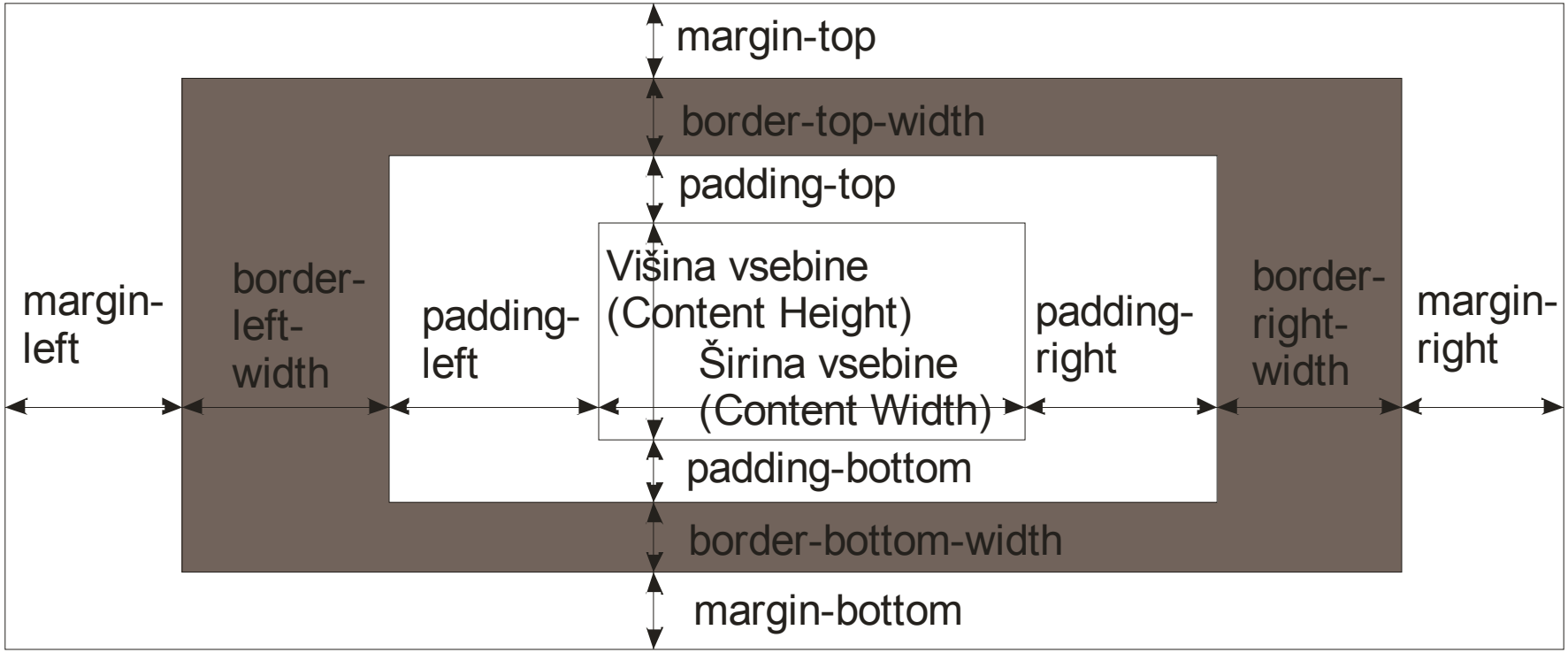
- Imena barv in njihove vrednosti

black	#000000	(črna)
gray	#808080	(siva)
silver	#c0c0c0	(srebrna)
white	#ffffff	(bela)
red	#ff0000	(rdeča)
blue	#0000ff	(modra)
yellow	#ffff00	(rumena)
green	#008000	(zelena)
purple	#800080	(vijolična)

Določanje velikosti



Določanje velikosti



- Primeri
  - Nastavitev za vsak rob posebej
    - {border-width: thin}

      (vsi robovi bodo tanki)
    - {border-width: thin medium}

      (zgornji/spodnji bosta tanka, levi/desni bosta srednje debela)
    - {border-width: thin thick medium}

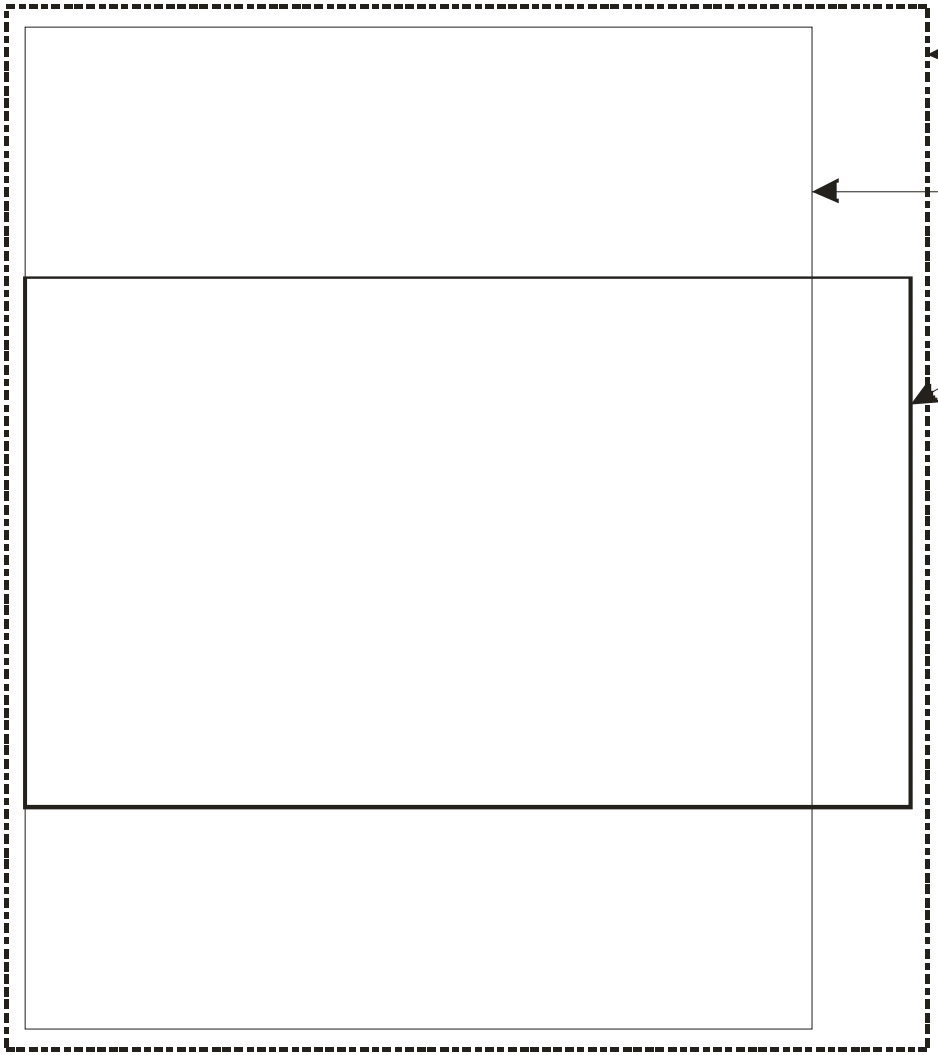
      (zgornji bo tanek, levi/desni bosta debela, spodnji bo srednje debel)
    - {border-width: thin medium thick medium}

      (vsak rob posebej: zgornji, desni, spodnji, levi)
    - *border-bottom: thick dashed blue;*
    - *border-bottom-color: blue;*
  - border-style (none, hidden, dotted, …), border-color, border-collapse, …

Ozadje elementov

- Ozadje zavzame področje vsebine (content), “padding” in robov (border)
  - Privzeto ozadje elementov je transparentno
  - Zunanja meja (margin) je **vedno** transparenta
  - Nastavitve ozadja se ne dedujejo
  - Nastavitve
    - background-color
    - background-attachment (scroll, fixed)
    - background-image (url (…))
    - background-position (top, top center, bottom left, ..)
    - background-repeat (repeat, no-repeat, repeat-x, …)

Osnovni bloki HTML dokumenta



Postavitvi: **block** in **inline**

p {display: inline }

Nastavljanje dimenzij elementov

- height, width, max-height, min-height, …
  - auto, px, %

    width = širina vsebine
    (margin-left + border-left-width+padding-left)
    (margin-right + border-right-width + margin-right)

- Določanje širine in višine s pomočjo %
  - Glede na starševski element

    {width: 40%}

  - Določanje položaja elementa s pomočjo širine (centriranje)

    {width: 50%; margin-left:auto; margin-right: auto}

3_Bloki.rar

Določanje položaja elementov

- Glavni načini določanja položajev elementov (position)
  - static: privzeti način, položaj je določen avtomatsko glede na ostale elemente
  - relative: relativno glede na prvotni položaj
  - absolute: poljuben položaj glede na canvas
  - fixed: poljuben položaj glede na okno brskalnika

  float: levo ali desno znotraj starševskega elementa (za določanje položaja vstavljenih slik, kadar uporabljamo "static" ali "relative" tip)

Določanje položaja elementov

- Druge nastavitve
  - overflow (visible, hidden, scroll, auto)
  - vertical-align (baseline, sub, super, top, text-top, middle, bottom, text-bottom, *vrednost*, *%)*
  - z-index (vrednost)
  - visibility (visible, hidden, collapse)
  - clear (left, right, both, none)
  - display (inline, block, none)

4_Polozaji.rar

5. Programska koda na odjemalcu

Programska koda na odjemalcu

- Programska koda, ki je vgrajena v samo spletno stran – HTML datoteko
- Izbiro programskega jezika določa izbira internetnega brskalnika
  - VBScript (Microsoft IE)
  - JavaScript (ECMA-262)
    - JScript (Microsoft): podpora na strežniku in odjemalcu
    - skriptni jezik (interpretirani jezik)
- Dopolnilo za atraktivnejši izgled spletne strani
  - Ključna funkcionalnost naj deluje tudi brez skript!

Osnove JavaScript jezika

- Prvi brskalnik s podporo za JavaScript je bil Netscape 2.0
- Sintaksa jezike je osnovana na programskem jeziku Java (Sun Microsystem)
- Opozorilo o napakah v kodi je na voljo šele med zagonom programa
- Pogoj za poganjanje JavaScript programov
  - Interpreter (scripting engine)
  - Gostujoče okolje (hosting environment)

- Najpogostejše naloge JavaScript-a
  - Dinamična vsebina spletnih strani
  - Odziv na akcije uporabnika
  - Spreminjanje vsebine HTML elementov
  - Pregledovanje in kontrola vnesenih podatkov pred pošiljanjem na strežnik
  - Delo s piškotki (cookies)

- Vključevanje JS programov v HTML dokument
  - V glavo dokumenta: ponavadi skripte, ki se izvedejo, ko jih pokličemo
  - V telo dokumenta: skripte se izvedejo ob nalaganju dokumenta
  - Preko zunanjih datotek

```
<head> ali <body>
  <script type="text/javascript">
  …
  </script>
<head>
```

```
<head>
  <script src="moji_programi.js"></script>
<head>
```

- Osnovna sintaktična pravila
  - Sintaksa je podobna C-jevskim jezikom
  - Občutljivost na male in velike črke
  - Podpičje na koncu vrstic je opcijsko
  - Posamezne bloke kode lahko združimo s pomočjo oklepajev

    {

    …

    }
  - Komentarji v kodi: // ali /*… */
  - Izpis v HTML: document.write (..);

- Spremenljivke
  - Spremenljivke nimajo tipov
  - Opcijska uporaba besede **var** za najavo spr.

    var x=5; var oseba="Janez";

    x=5; var zaposlen=false;
  - Najava konstant

    const Pi = 3.14;
  - Vrednost spremenljivke pripada enemu od tipov
    - Boolean, Number, String, Object, undefined, null
    - Z ukazom typeof lahko ugotovimo tip spremenljivke

- Življenjski prostor spremenljivk (scope)
  - Lokalna spremenljivka
    - Definirana znotraj funkcije s pomočjo **var**
    - Podana kot parameter funkcije
  - Globalna spremenljivka
    - Vse ostalo
    - ***window.****ime_spremenljivke*

- Tipi spremenljivk
  - Boolean
    - Dve možni vrednosti: true ali false
    - false: 0, -0, null, undefined, NaN, prazen string, false
  - Number
    - Vsa števila so tipa float
    - Cela števila (integer) so tipa float, a brez decimalne vejice
      - ***Math.floor()***
    - Različni načini zapisov
      - cifra = 314.54; (decmalno)
      - cifra = 2.14e+5; (eksponentno)
      - cifra = 0xFE; (šestnajstiško)
      - cifra = 0377; (osmiško)

- Tipi spremenljivk
  - String
    - Dolžina je eden ali več znakov
    - Ni tipa “char”
    - Posebni znaki
      - \b brisanje (backspace)
      - \n nova vrstica (new line)
      - \” narekovaj
      - \\ leva poševnica (backslash)
      - …
  - Null in Undefined
  - Avtomatska pretvorba tipov spremenljivk

- Operatorji
  - Osnovne rač. operacije:
    - + - * /
    - ++ -- (pozor: a++ ali ++a)!!
  - Prirejanje
    - = += -= *= /=
  - Pogojno prirejanje: x=(pogoj)?vr1:vr2;
  - Znak + lahko uporabimo tudi za združevanje besed (tipov String)
  - Če združimo String in Number vedno dobimo String
  - Rezultat (napaka) nelogičnih operacij je NaN

- Primerjanje spremenljivk
  - Enakost: ==
  - Stroga enakost (tip in vrednost): ===
  - Neenakost: !=
  - < > >= <=
  - Logični operatorji
    - In (AND): &&
    - Ali (OR): ||
    - Ne (NOT): !

1_scope.html

- Pogojni stavki

```
if (pogoj)
{
   …
}
else
{
  …
}
```

```
if (x ==1)
{
   x += 2;
}
```

Odločitveni stavki

```
switch (spremenljiva)
{
case 1:
    …
    break;
case 2:
    …
    break;
default:
    …
}
```

```
switch(oseba)
{
case “Janez”:
  document.write(“Zdravo Janez”);
  break;
case “Miha”
 document.write(“Zdravo Miha”);
  break;
 default:
   document.write(“Zdravo ???”);
}
```

- Ukazi za ponavljanje
  - Omogočajo, da se določen kos kode ponovi poljubno število ciklov
  - Ukaza
    - For: ponavljanje določeno število ciklov
    - While: ponavljanje dokler ni izpolnjen določen pogoj

```
ime: for (i=1; i<=10; i++)
{
   …
   break ime;
}
```

```
i=1;
while (i <=10)

{
   …
   i++;
}
```

- Funkcije
  - Funkcija loči oz. omeji določen del kode in se ta izvede le ob klicu
  - Lahko je definirana v glavi ali telesu HTML dokumenta
  - Lahko vrne vrednost ali pa tudi ne (lahko vrne več različnih vrednosti)
  - Parameter funkcije je lahko funkcija

```
function vsota(param1, param2)
{
  return param1+param2;
}
```

```
function izpis(besedilo)
{
  document.write(besedilo);
}
```

- Funkcije
  - Privzete oz. vgrajene funkcije
    - parseInt(), parseFloat()

      Pretvorba texta (string) v številko (number)
    - isNaN(), isFinite()

      Preverjanje napačnih rezultatov mat. operacij
    - URIencode(), URIdecode()

      Kodiranje oz. dekodiranje URL zapisa – odstranjevanje nedovoljenih znakov v URL zapisu

www.blabla.com/Moji dokumenti/pomembno!.doc

www.blabla.com/Moji%20dokumenti/pomembno%21.doc

2_zanke.html

REZERVIRANE BESEDE

abstract
boolean **break** byte
case catch char class const **continue**
debugger **default delete do** double
else enum export extends
false final **finally** float **for function**
goto
if implements import **in instanceof** int
interface
long
native **new null**
package private protected public
return
short static super **switch** synchronized
this throw throws transient **true try typeof**
var volatile **void**
while with

- Objekti
  - JavaScript je v svoji osnovi objektni jezik
  - Objekt vsebuje različno število lastnosti (property) in metod (method)
  - Definicija objekta in lastnosti

```
var oseba= new Object();
oseba.ime = “ Janez“;
oseba.starost = 34;
oseba.porocen = false;
```

```
var oseba = {ime:“Janez“; starost:34; porocen:false};
```

- Vgrajeni JavaScript objekti
  - window
  - Number
  - String
    - Lastnosti: length…
    - Metode: big(), bold(), fontcolor(), indexOf(), toUpperCase(), charAt(cifra), concat(beseda), …
  - Date
    - Metode: Date() – današnji datum in ura
      getDate(), getHours() – dan, ura,…
      setDate(), setHours() – nastavljanje dni, ur, …
  - Boolean
  - Array
  - Math
    - Lastnosti: E, LN2, PI, SQRT2, …
    - Metode: abs(), sin(), cos(), sqrt(), round(), random(), …

3_objekti_vgrajeni.html

Objekti

```
var oseba= new Object();
oseba.ime = “ Janez“;
oseba.starost = 34;
oseba.porocen = false;

for (var lastnost in oseba) {
   document.write(lastnost + “ je lastnost objekta oseba“);
}

document.write(oseba.ime);
document.write(oseba[“ime“]);

for (var lastnost in oseba) {
   document.write(lastnost + “ ima vrednost “ + oseba[lastnost]);
}
```

4_objekti_izpis.html

- Vse spremenljivke tipa Object so reference (kazalci)

```
var obj1 = new Object();
obj1.vr = “Lep“;
var obj2 = obj1;
obj2.vr += “ dan!“;
window.alert (obj1.vr);
```

5_objekti_reference.html

- Metode objekta

```
function povecaj_a() {
  this.a++;
}

var objekt = new (Object);
objekt.a = 1;
objekt.nekaj = povecaj_a;
```

```
var objekt = new (Object);
objekt.a = 1;
objekt.nekaj = function(){
  this.a++;
};
```

Konstruktorji

```
function ustvariOsebo(ime) {
   var obj = new Object();
   obj.ime = ime;
   obj.starost = null;
   obj.spremeniIme = function(ime) {
      this.ime = ime;
   };
   return obj;
}

var oseba = ustvariOsebo("Janez");
oseba.starost = 71;
```

```
function ustvariOsebo(ime) {
   this.ime = ime;
   this.starost = null;
   this.spremeniIme = function(ime) {
      this.ime = ime;
   };
}

var oseba = new ustvariOsebo("Janez");
oseba.starost = 71;
```

6_objekti_konstruktorji.html

Prototipi

```
function ustvariOsebo(ime) {
   var obj = new Object();
   obj.ime = ime;
   obj.starost = null;
   obj.spremeniIme = function(ime) {
      this.ime = ime;
   };
   return obj;
}

var oseba = ustvariOsebo("Janez");
oseba.starost = 71;

var oseba = ustvariOsebo("Miha");
oseba.prototype.spol = "m";
```

```
function beriNazaj() {
  for (i=this.length-1;i>=0;i--)
  document.write(this.charAt(i))
}

String.prototype.vzvratno=beriNazaj;

var stavek = "Zivjo!"
stavek.vzvratno();
```

- Nizi (arrayi)
  - Posebna vrsta objektov

```
var niz1 = new Array();

var niz2 = new Array(1, “Jaka”, true);

var niz3 = [1, “Jaka”, true];

var niz4 = [ [ “A”, “B”, “C”],
             [ “a”, “b”, “c”],
              [1, 2 ,3] ];
```

```
var dolzina = niz1.length;
niz2.length = 4;

niz2[4] = -23.5;

niz5 = new Array(5);
```

- Nizi (arrayi)
  - Metode objekta Array
    - toString(): vrne niz kot string, elementi so ločeni z vejico
    - push(nov_element): doda element (vrne dolžino)
    - pop(): vrne zadnji element in ga odstrani
    - shift(): vrne prvi element in ga odstrani iz niza
    - splice(index, 0, nov_element): vrine nov element na določeno mesto
    - splice(index, stevilo_elementov): odstrani določeno stevilo elementov od določenega indeksa naprej

7_arrayi.html

- Kontrola nad morebitnimi napakami v kodi
  - try … catch stavek (throw)

```
try
{
  …
}
catch(napaka)
{
  …
  … napaka …
}
```

```
try
{
   throw(“Moja napaka”);
}
catch (npk)
{
  … npk …
}
```

8_napake.html

- Kontrola nad morebitnimi napakami v kodi
  - onerror dogodek

```
onerror=ObdelajNapako;
…
…
function ObdelajNapako(opis, naslov, vrstica)
{
  …
  … opis …
}
```

6. Document Object Model - DOM

HTML DOM

- HTML Document Object Model – HTML DOM
  - DOM je definicija lastnosti objekta **document,** ki so prav tako objekti s svojimi lastnostmi in metodami
  - Dostop in spreminjanje strukture in vsebine HTML dokumenta s pomočjo JavaScript programov
  - HTML dokument je dostopen drevesna struktura (elementi, lastnost, besedilo), katerega koren (root) je objekt **document**
  - W3C DOM – poenoten model v vseh brskalnikih
    - Najširše je uporabljena in implementiran DOM Level 2

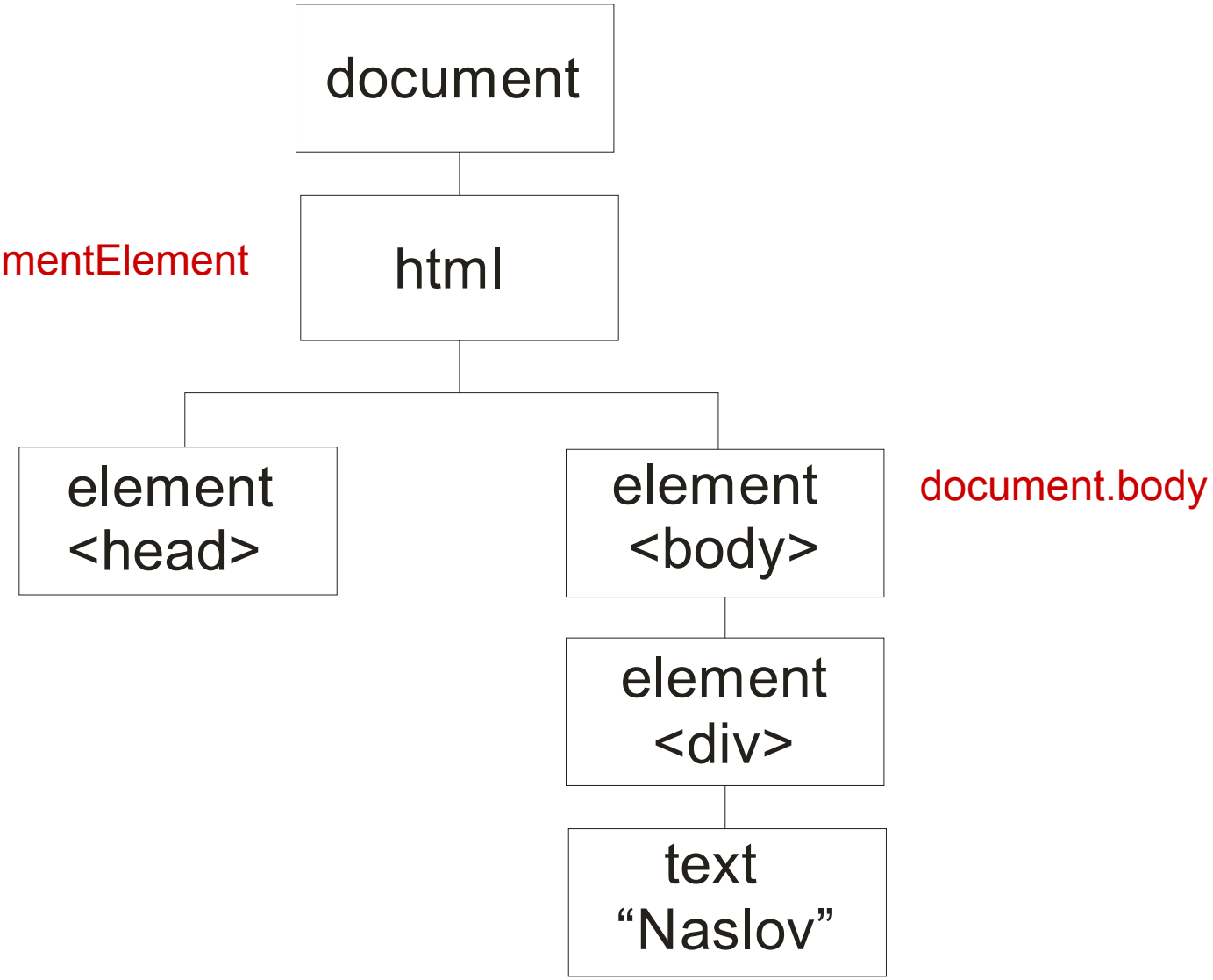
HTML DOM

- Povezave med vozlišči v drevesu
  - Izhodiščno vozlišče oz. koren drevesa se imenuje root – **document**
  - Vsako vozlišče razen korena ima natanko enega starša (**parent**)
  - Vsako vozlišče ime lahko poljubno potomcev (**children**)
  - Vozlišče brez potomcev imenujemo list (leafe)
  - Vsa vozlišča, ki imajo istega starša (bratje) se imenujejo **siblings**

HTML DOM

document

document.documentElement



document.body

HTML DOM

- Vozlišče (objekt **Node**)
  - Osnovni tipi vozlišč
    - Element (element) 1- (Node.ELEMENT_NODE)
    - Lastnost (attribute) 2 - (Node.ATTRIBUTE_NODE)
    - Besedilo (text) 3 - (Node.TEXT_NODE)
    - Komentar (comment) 8 - (Node.COMMENT_NODE)
    - Dokument (document) 9
    - DocumenType 10 -(Node.DOCUMENT_TYPE_NODE)
  - Vsa besedila v HTML datoteki so predstavljena kot vozlišča tipa **text**

- Primer DOM drevesa za HTML datoteko

```
<html>
 <head>
      <title>Moja primer</title>  <!-- tole je naslov -->
</head>
<body>
     <h1>Drevesna struktura HTML dokumenta</h1>
     <p>
       <b> Tole je pomembna vaja! </b> <br />
       <a href="www.najdi.si"> Link na NAJDI </a>
     </p>
</body>
</html>
```

- Lastnosti objekta **Node**
  - nodeType: tip vozlišča (1, 2, 3 .. 9)
  - nodeName: HTML značka
  - *innerHTML*: celotna vsebina vozlišča
  - parentNode: starš
  - childNodes: set otrok
  - firstChild, lastChild: prvi in zadnji otrok
  - previousSibling, nextSibling: bratje
  - attributes: lastnosti znotraj HTML značke
- Metode
  - hasAttributes(), hasChildNodes(), appendChild(Node), insertBefore(Node, Node), removeChild(Node), replaceChild(Node, Node)

0_uvod.html

1_izpis_drevesa.html

HTML DOM

- Vozlišče **document**
  - Predstavlja koren DOM drevesa (koren vozlišča **html**)
  - Starševski element za vse elemente izven html strukture (komentarji, DOCTYPE, itd.)
  - Dodatne lastnosti in metode
    - doctype, title
    - body
    - URL (read-only), domain (read-only), referrer
    - createElement(String)
    - createTextNode(String)
    - getElementById(String)
    - getElementsByTagName(String)

HTML DOM

- Vozlišče **element**
  - Vsi HTML elementi v dokumentu
  - Dodatne lastnosti in metode
    - tagName
    - getAttribute(String)
    - setAttribute(String, String)
    - removeAttribute(String)
    - hasAttribute(String)
    - getElementsByTagName(String)

- Vozlišče **text**
  - Vse kar ni del označevalnega jezika
  - Besedila v elementih so lahko razdeljena na več vozlišč tipa text
  - Metoda **normalize**()

- Dodatne lastnosti elementov (okrajšave)
  - Nadomestek klicev funkcij setAttribute in getAttribute

```
el.setAttribute(“id”, “vrednost”);
el.id = “vrednost”;
```

```
el.getAttribute(“value”);
el.value
```

2_obrazec.html

- Spreminjanje stila (CSS)
  - Vsakemu objektu lahko nastavljamo in spreminjamo lastnosti stila preko lastnosti style

```
document.getElementById("m").style.backgroundColor="blue";

document.getElementById("m").style.zIndex="2";
```

  - Lastnosti objekta style so enake CSS lastnostim
    - Namesto '-' je velika začetnica

```
{background-color: blue;}        el.style.backgroundColor = "blue";
{font-size: 12px;}               el.sytle.fontSize = "12px";
```

  - Metodi **getPropertyValue** in **setProperty**

```
var i = el.style.getPropertyValue("background-color");
el.style.setProperty("background-color","blue","");
```

Test.html

- Upravljanje z dogodki (event handling)
  - Akcije uporabnika, ki jih zazna brskalnik in pokliče predviden JavaScript program
  - Določitev privzetega skriptnega jezika za obravnavanje dogodkov

```
<meta http-equiv="Content-Script-Type" content="text/javascript" />">
```

  - Dogodke določimo kot
    - atribute HTML elementov

```
<a href="www….." onmouseover="moj_program();">
```

    - JavaScript metode

```
a.onmouseover = moj_program;
```

- Nabor dogodkov
  - Dogodki miške
    - onclick, ondblclick, onmousedown, onmousmove, onmouseup, onmouseover, onmouseout
  - Dogodki tipkovnice
    - onkeydown, onkeypress, onkeyup
  - Dogodki oken/objektov
    - onload, onunload
    - onabort, onerror
    - onresize, onscroll
  - Dogodki obrazcev (form)
    - onfocus, onblur
    - onreset, onsubmit, onselect

- Naprednejši načini upravljanja dogodkov
  - "Event listener" je lahko vsaka funkcija, ki kot parameter posreduje instanco objekta Event
  - **.addEventListener(dogodek, funkcija, false)**

```
var gumb = window.document.getElementById("gumb");
gumb.addEventListener("click", akcija, false);

function akcija() {
  window.alert ("Pritisnjen je bil gumb!");
  return;
}
```

  - Dogodki so enaki HTML dogodkom
    - Ni podpore za: keypress, keydown, keyup

Primer: validacija forme

```
<script type="text/javascript">
  function preveri(){
    var polje = document.getElementById(“polje”);
    if (polje.value = ‘’)  {
      alert(“Polje ne sme biti prazno”);
      return false;
    }
     return true;
  }
</script>
```

```
<form … onsubmit=“return preveri()”>
  <input id=“polje”/>
</form>
```

```
<script type="text/javascript">
  function preveri(event){
    var polje = document.getElementById(“polje”);
    if (polje.value = ‘’)  {
      alert(“Polje ne sme biti prazno”);
      event.preventDefault();
    }
 }
</script>
```

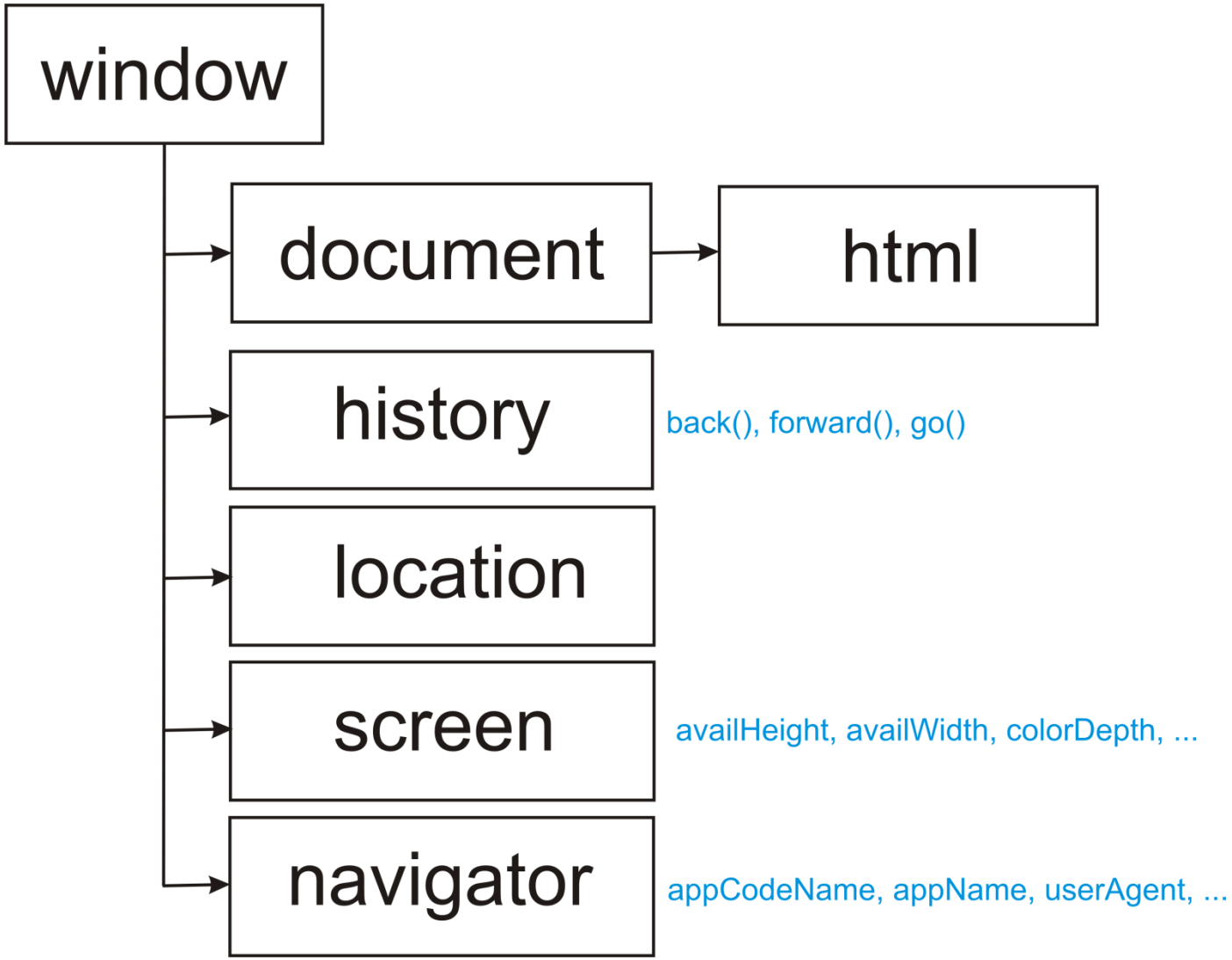
```
<form … id=“forma”>
  <input id=“polje”/>
</form>

var forma = document.getElementById(“forma”);
forma.addEventListener(“submit”, preveri, false);

var forma = document.getElementById(“forma”);
forma.onsubmit = preveri;
```

3_validacija.html

- Lastnosti objekta Event (**window.event)**
  - bubbles, cancelable, target, timeStamp, type
  - Miška in tipovnica
    - altKey, button, clientX, clientY, ctrlKey, screenX, screenY, shiftKey
- Preprečevanje proženja večkratnih dogodkov
  - **event.cancelBubble = true;**
  - **event.stopPropagation();**

4_bubbles.html

HTML DOM


window
document
html
history
back(), forward(), go()
location
screen
availHeight, availWidth, colorDepth, ...
navigator
appCodeName, appName, userAgent, ...

7. Asinhroni JavaScript in XML
AJAX

AJAX

- Osnovne značilnosti
  - Dopolnjuje klasični način komunikacije brskalnika s strežnikom (preko HTML obrazcev - POST in GET)
  - Komunikacija brskalnika s strežnikom z uporabo JavaScript programov
  - Asinhroni način – brez ponovnega nalaganja strani
  - Pogoj za delovanje AJAX tehnologije je podpora s strani brskalnika
  - Ni samostojni standard, temelji na:
    - JavaScript
    - XML
    - HTML
    - CSS

AJAX

- Objekt XMLHttpRequestObject
  - Posebni JavaScript objekt, ki omogoča asinhrono komunikacijo s strežnikom
  - Podprt v brskalnikih
    - Internet Explorer 5.5+
    - Safari 1.2
    - Mozilla 1.0+
    - Opera 8+
  - Vsak brskalnik uporablja omenjeni objekt na svoj način
    - Pri pisanju aplikacije moramo zagotoviti podporo za vse vrste brskalnikov

AJAX

- Inicializacija XMLHttpRequest objekta

```
function ajaxFunction() {
      var xmlhttp;
      if (window.XMLHttpRequest)
        {
            // podpora za IE7+, Firefox, Chrome, Opera, Safari
            xmlhttp=new XMLHttpRequest();
        } else  {
        // podpora za IE6, IE5
        xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
        }
      }
```

AJAX

- Objekt XMLHttpRequestObject
  - Lastnost ***onreadystatechange***
    - Lastnost mora določati funkcijo, ki se pokliče, ko pridejo zahtevani podatki iz strežnika

```
xmlHttp.onreadystatechange=function()
{
   // Programska koda, ki obdela podatke
}
```

AJAX

- Objekt XMLHttpRequestObject
  - Lastnost **readyState**
    - Vedno se spremeni istočasno kot onreadystatechange
    - Vsebuje informacijo o statusu odziva strežnika
      - readyState: 0,1,2,3,4 (4 pomeni končan odziv strežnika)
      - status: 200 (OK) ali 404 (Page not found)

```
xmlHttp.onreadystatechange=function()
{
        if (xmlHttp.readyState==4 && xmlHttp.status==200){
          // Odziv strežnika končan, lahko beremo podatke
        }
}
```

- Objekt XMLHttpRequestObject
  - Lastnost **responseText**
    - Vsebuje dejanski odziv strežnika (informacijo, ki jo vrne skripta na strežniku)

```
xmlHttp.onreadystatechange=function()
{
   if (xmlHttp.readyState==4 && xmlHttp.status==200){
   {
       izpis = document.getElementById("moj_element");
       izpis.innerHTML = xmlHttp.responseText;
   }
 }
```

  - Lastnost **responseXML**

AJAX

- Objekt XMLHttpRequestObject
  - Pošiljanje zahteve na strežnik
  - Ukaz ***open*** *(metoda, skripta na strežniku, asinhroni način)*
    - Ukaz ***send*** *(parametri)*

```
xmlHttp.open("GET",“skripta.php",true);
xmlHttp.send(null);
```

```
xmlHttp.open("GET",“skripta.php?param1=2&param2=joco",true);
xmlHttp.send(null);
```

```
xmlhttp.open("POST","ajax_test. php",true);
xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded");
xmlhttp.send(“ime=Janez&priimek=Novak");
```

4_Ajax_primeri.rar

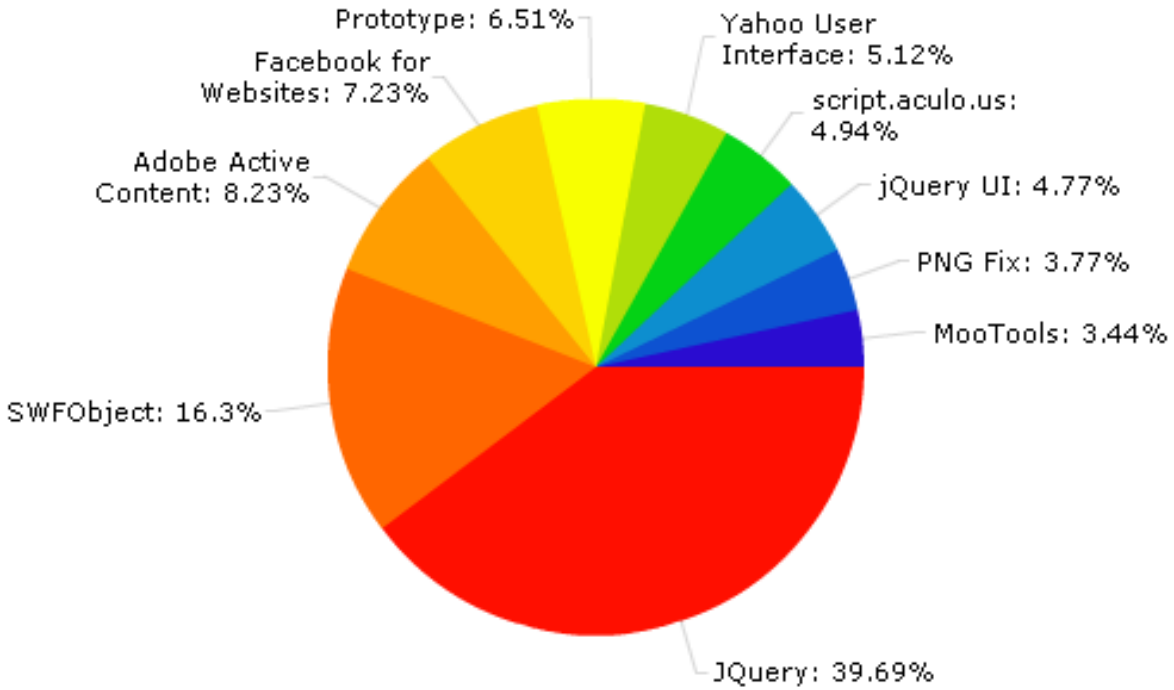
8. Javascript ogrodja (frameworks)

Javascript ogrodja

- Splošne značilnosti
  - Nabor pred-pripravljenih javascript kontrol, funkcij in metod za dodajanje različnih funkcionalnosti
  - Enostavna implementacija različnih dinamičnih elementov
  - Imitacija standardnih računalniških aplikacij
  - Ogrodja dodamo kot eno ali več zunanjih datotek
  - Dobra podpora v vseh razširjenih brskalnikih
  - Slabosti
    - Dodatne zakasnitve pri nalaganju strani
    - Nova sintaktična pravila (nov programski jezik?!)

JavaScript ogrodja

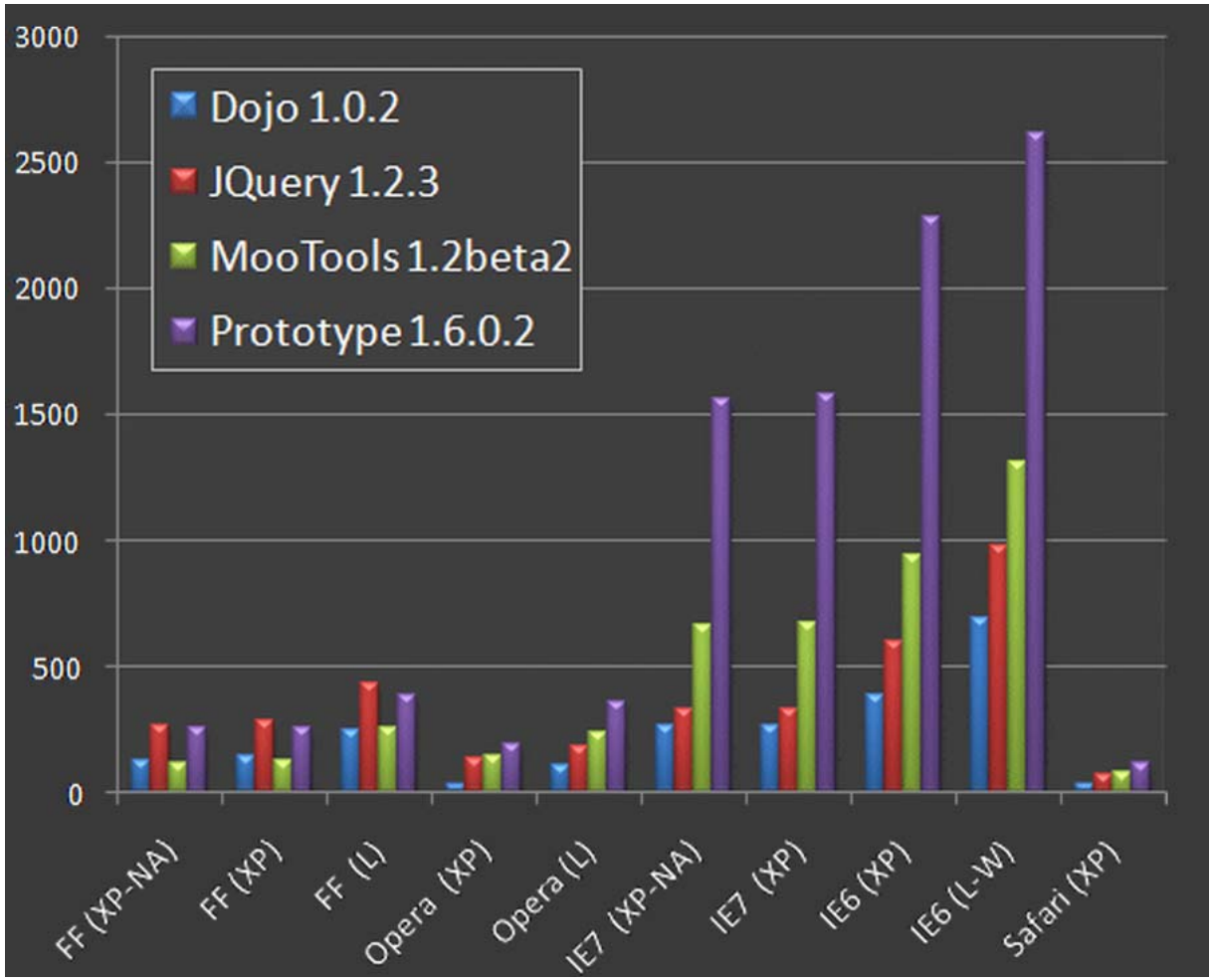
- Dojo
- Prototype in Scriptaculous
- jQuery in jQuery UI
- MooTools
- Yahoo user interface



[http://trends.builtwith.com]

Javascript ogrodja

- Hitrost izvajanja kode v brskalnikih



[http:// blog.creonfx.co]

8.1. Osnove ogrodja jQuery

Osnove ogrodja jQuery

- Celotno ogrodje se nahaja v eni js datoteki
  - Kompresirana in nekompresirana datoteka
  - Možnost vključitve preko Google API-ja
- Ogrodje določa globalni jQuery z različnimi lastnostmi in metodami
  - ***jQuery*** ali ***$*** *(window.jQuery ali window.$)*

```
var jQuery = window.jQuery = window.$ = function(selector, context)
    {
      // ...
      // other internal initialization code goes here
    };
```

  - selector: CSS selektor za izbiro DOM elementa
  - *context: začetna izbira položaja v DOM drevesu*

Primer jQuery dodatka

```
<!DOCTYPE html>
 <html lang="en">
 <head>
   <meta charset="utf-8">
   <title>jQuery demo</title>
   <script src="jquery-1.7.1.min.js"></script>
 </head>
 <body>
   <a href="http://www.najdi.si">NAJDI</a>
<script>
     $(document).ready(function(){
       $("a").click(function(event){
         alert("Link ne deluje!");
         event.preventDefault();
       });
     });
   </script>
 </body>
 </html>
```

```
$(document).ready(function() {
 …
}
```

je alternativa funkciji

```
window.onload = function () {
 …
}
```

- Izbira DOM elementov (jQuery selektorji)
  - Uporaba CSS3 selektorjev (in XPath)
  - Izbira elementov na podlagi ID-jev, razredov, elementov, atributov:

```
$(“#moj_id”)        $(“.moj_razred”)          $(“input”)
                    $(“div.moj_razred”)

$(“div#moj_id > a”)          $(“#vsebina div.poudarjeno p”)

$(“p[name=moj_p]”)           $(“p[name*=moj]”)  (^=, $=)
```

  - el:nth-child(n), el:last-child, e:enabled, e:disabled, e:not(s), e > f, e + f, *el:visible, el:hidden*, *el:parent*

```
$(“a.zunanji:first”)
$(“div:visible”)
```

- Naprednejša izbira in filtriranje (Traversing)
  - .find(“..”)
  - .each(function(..){…})
  - :has()
  - .filter() in .not()
  - .next(), .prev(), .parents(), .siblings(), …

```
$(document).ready(function() {
  $("#moj_div").find(“p").each(function(i) {
    $(this).append( " pa še tole! " + i );
  });
});
```

alternativa

$(“#moj_div p”).each(…

```
$(document).ready(function() {
  $(“div").not(":has(p)").css(“color", “red");
});
```

Osnove ogrodja jQuery

- Manipulacija elementov
  - Dogodki
    - .click(), .keypress(), .focus(), .hover(), .mouseenter(), .mouseleave(), .resize()
  - CSS
    - .css(), .addClass(), .removeClass(), .height(), .width(), .position(), …
  - Dodajanje in odvzemanje vsebine in html strukture
    - .append(), .html(), .val(), .text(), .attr(), .wrap(), …
  - Efekti
    - .animate(), .show(), .hide(), .fadeIn(), .fadeOut(), .fadeTo(), .slideDown(), .slideUp(), .delay(), …

basics_1_2.html

- jQuery in uporabniški vmesnik (jQueryUI)
  - Zbirka kontrol in posebnih efektov za nadgradnjo uporabniškega vmesnika
  - Dodatna js datoteka

```
<script src="jquery-1.7.1.min.js"></script>
<script src="jquery-ui-1.8.18.custom.min.js"></script>
```

  - Zbirka demo efektov in generatorjev teme (CSS)
    - Draggable, Droppable, Resizable, Selectable, …
    - Accordion, Datepicker, Dialog, Tabs, …

dragNdrop.html

harmonika.html

9. EXtensible Markup Language
XML

XML (EXtensible Markup Language)

- Razširljiv označevalni jezik
- V osnovi namenjen hranjenju in izmenjavi podatkov
- Značke niso vnaprej določene in jih določimo po potrebi
- XML datoteke so navadne tekstovne datoteke
- Drevesna struktura

PRIMER:

```
<?xml version = "1.0”?>
<oseba>
<priimek>Novak</priimek>
<ime>Janez</ime>
<starost>52</starost>
</oseba>
```

```
<oseba>
<priimek>Novak</priimek>
<ime>Miha</ime>
<starost>22</starost>
</oseba>
```

XML (EXtensible Markup Language)

- Osnovna sintaktična pravila
  - Začetna vrstica: <?xml version = "1.0“?>
  - Encoding: <?xml version="1.0" encoding=“UTF-8"?>
  - Obvezne zaključne značke
  - Občutljivost na male in velike črke
  - Zahtevana je drevesna struktura (pravilno gnezdenje elementov)
  - Značke lahko vsebujejo lastnosti (atribute)
    - Obvezna uporaba navednic

      PRIMER: <oseba podjetje=“IskraTEL”>…… </oseba>
  - Entitete, komentarji (enako kot v HTML)
  - …

osebe.xml

XML (EXtensible Markup Language)

- Razčlenjevanje oz. interpretacija XML datotek
  - Razčlenjevalnik (parser) XML datotek je del brskalnika
    - Spremeni XML dokument v XML DOM objekt
    - XML DOM objekt manipuliramo s pomočjo JavaScripta
  - Razčlenjevanje XML tekstovne spremenljivke

```
txt = “<xml…….”
if (window.DOMParser)
  {
  parser=new DOMParser();
  xmlDoc=parser.parseFromString(txt,"text/xml");
  }
else      // Internet Explorer
  {
  xmlDoc=new ActiveXObject("Microsoft.XMLDOM");
  xmlDoc.async=false;
  xmlDoc.loadXML(txt);
  }
```

XML (EXtensible Markup Language)

- Razčlenjevanje XML dokumenta

```
if (window.XMLHttpRequest)
  {                                  // code for IE7+, Firefox, Chrome, Opera, Safari
  xmlhttp=new XMLHttpRequest();
  }
else
  {                                  // code for IE6, IE5
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }

xmlhttp.open("GET",“dokument.xml",false);
xmlhttp.send();
xmlDoc=xmlhttp.responseXML;
```

XML (EXtensible Markup Language)

- Bralnik shrani XML datoteko v spomin v obliki DOM (Document Object Model) drevesa
- Vsak XML element predstavlja vozlišče v drevesu
- Z XML drevesom operiramo na enak način kot s HTML drevesom
  - getElementsByTagName, childNodes, parentNode, itd.
  - nodeName, nodeValue, attributes, itd.

osebe_prikaz.html

XML (EXtensible Markup Language)

- XML imenski prostor (Namespace)
  - Ločevanje podvojenih značk s pomočjo prefiksa

```
<ss:gradivo>Web technologies</ss:gradivo>
<st:gradivo>Developing web</st:gradivo>
```

  - Določanje imenskega prostora
    - URL ali URN (ali nek poljubni niz)

URL: http://www.lkn.fe.uni-lj.si/svetovni_splet
URN: www-lkn-fe-uni-lj-si:svetovni_splet

    - Uporaba atributa **xmlns**

xmlns:ss = “http://www.lkn.fe.uni-lj.si/svetovni_splet”
xmlns:st = “http://www.lkn.fe.uni-lj.si/spletne_tehnologije”

    - Privzet imenski prostor

xmlns = “http://www.lkn.fe.uni-lj.si/predmeti”

- Primer uporabe imenskih prostorov

```
<?xml version = "1.0”?>

<!-- predmet.xml -->

<predmet xmlns = “http://www.lkn.fe.uni-lj.si/predmeti”
          xmlns:ucni = “http://www.lkn.fe.uni-lj.si/ucni_predmeti”>

          <opis ime=“projektor”>
                    <letnik>2010</letnik>
                    <znamka>LG</znamka>
          </opis>
          <ucni:opis ime=“Svetovni splet”>
                    <ucni:letnik>3</ucni:letnik>
                    <ucni:semester>poletni</ucni:semester>
          </ucni:opis>
</predmet>
```

XML (EXtensible Markup Language)

- Definicije tipov dokumentov (Document Type Definitions – DTD)
  - Specifikacija XML strukture, ki omogoča validacijo
    - Nabor in vrstni red razpoložljivih elementov in atributov
    - Določitev elementov kot obveznih ali opcijskih
  - Na osnovi EBNF (Extended Backus-Naur Form)
    - Elementi: **<!ELEMENT …**
      - **+** (element se pojavi vsaj enkrat), ***** (element se pojavi poljubno krat ali nikoli), **?** (element se pojavi enkrat ali nikoli)
    - Atributi: **<!ATTLIST …**
      - **#IMPLIED** (ni obvezen)**, #REQUIRED** (obvezen)**, #FIXED** (privzet)
      - Predpisana vrednost: (M | Z)
    - Tip podatkov
      - **#CDATA** (tekst, ki se ne razčlenjuje), **#PCDATA** (element, ki se razčlenjuje), **EMTPY** (prazen element)

XML (EXtensible Markup Language)

Primer DTD datoteke

```
<?xml version = "1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE predmet SYSTEM "predmet.dtd>
<!-- predmet.xml -->

<predmet xmlns = "http://www.lkn.fe.uni-lj.si/predmeti"
xmlns:ucni = "http://www.lkn.fe.uni-lj.si/ucni_predmeti">

        <opis ime="projektor">
            <letnik>2010</letnik>
            <znamka>LG</znamka>
        </opis>
        <ucni:opis ime="Svetovni splet">
            <ucni:letnik>3</ucni:letnik>
            <ucni:semester>poletni</ucni:semester>
        </ucni:opis>
</predmet>
```

```
<!– predmet.DTD-->

<!ELEMENT predmet ( opis+ ) >

<!ELEMENT opis ( letnik, znamka?, semester? ) >
<!ATTLIST opis ime CDATA  #REQUIRED>

<!ELEMENT letnik (  #PCDATA ) >
<!ELEMENT znamka  (  #PCDATA ) >
<!ELEMENT semester (  #PCDATA ) >
```

Primer DTD datoteke

```
<?xml version = "1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE predmet [
   <!ELEMENT predmet ( opis+ ) >
   <!ELEMENT opis ( letnik, znamka?, semester? ) >
   <!ATTLIST opis ime CDATA  #REQUIRED>
   <!ELEMENT letnik (  #PCDATA ) >
   <!ELEMENT znamka  (  #PCDATA ) >
   <!ELEMENT semester (  #PCDATA ) >
]>

<predmet xmlns = "http://www.lkn.fe.uni-lj.si/predmeti"
xmlns:ucni = "http://www.lkn.fe.uni-lj.si/ucni_predmeti">

                <opis ime="projektor">
                    <letnik>2010</letnik>
                    <znamka>LG</znamka>
                </opis>
                <ucni:opis ime="Svetovni splet">
                    <ucni:letnik>3</ucni:letnik>
                    <ucni:semester>poletni</ucni:semester>
                </ucni:opis>
</predmet>
```

osebe_dtd.html

XML (EXtensible Markup Language)

- XML sheme
  - Strožja specifikacija XML strukture
  - Zasnovana na XML jeziku samem
  - Natančna specifikacija tipov vnosa
    - string, boolean, decimal, float, double, long, int, date, time, ***complexType***
    - Končnica **.xsd**
  - Spletni validatorji

XML (EXtensible Markup Language)

- Sintaktična pravila za pisanje XML shem (na primeru)

```
<?xml version = "1.0”?>

<!-- predmet.xsd -->

<schema xmlns = “http://www.w3.org/2001/XMLSchema”
            xmlns:lkn = “http://www.lkn.fe.uni-lj.si/predmeti”
           targetNamespace = “http://www.lkn.fe.uni-lj.si/predmeti”

           <element name = “predmet” type=“lkn:Opisi”/>

           <complexType name = “Opisi”>
                    <sequence>
                        <element name = “opis” type=“lkn:Predmet” minOccurs = “1”/>
                    </sequence>
           </complexType>

           <complexType name = “Predmet”>
                    <sequence>
                        <element name = “letnik” type=“int”/>
                        <element name = “znamka” type=“string”/>
                    </sequence>
           </complexType>
</schema>
```

XML (EXtensible Markup Language)

```
<?xml version = "1.0"?>

<!-- computer.xsd -->

<schema
  xmlns = "http://www.w3.org/2001/XMLSchema"
   xmlns:computer = "http://www.ccc.com/computer"
  targetNamespace = "http://www.ccc.com/computer">

        <simpleType name="gigaherzt">
            <restriction base = "decimal">
                <minInclusive value = "2.1"/>
            </restriction>
        </simpleType>

        <complexType name = "CPU">
            <simpleContent>
                <extension base = "string">
                    <attribute name = "model" type = "string" />
                </extension>
            </simpleContent>
        </complexType>
        <complexType name = "portable">
            <all>
                <element name = "processor" type = "computer:CPU"/>
                <element name = "monitor" type = "int"/>
                <element name = "CPUSpeed" type = "computer:gigahertz"/>
                <element name = "RAM" type = "int"/>
            </all>
            <attribute name = "manufacturer" type = "string"/>
        </complexType>

        <element name = "laptop" type = "computer:portable/>"
</schema>
```

```
<?xml version = "1.0"?>

<!-- laptop.xml -->

<computer:laptop xmlns:computer = http://www.ccc.com/computer
    manufacturer = "IBM">

    <processor model = "Centrino">Intel</processor>
    <monitor>17</monitor>
    <CPUSpeed>2.4</CPUSpeed>
    <RAM>2</RAM>
</computer:laptop>
```

osebe.xsd

XML (EXtensible Markup Language)

- XML stili in transformacije (XSL in XSLT)
  - XSL (Extensible Stylesheet Language)
  - XSLT (XSL Transformations)
  - Transformacija ene XML strukture v drugo
    - Izvorna in ponorna drevesna struktura
      - Primer: XML v (X)HTML
    - Xpath sintaksa za dostopanje do različnih delov dreves
      - Atributa **select** in **match**
      - Določanje relacij med vozlišči: **/**
  - Dodatna datoteka s končnico **.xsl**
    - Navedba transformacije v XML datoteki z **<?...?>**

XML (EXtensible Markup Language)

- Osnovna XSLT sintaksa
  - Deklaracija stila in imenskega prostora

```
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

  - Določitev ciljnega drevesa (template-a)

```
<xsl:template match="/">
```

  - Izpis vrednosti iz XML elementa ali atributa

```
<xsl:value-of select="oseba/ime"/>
```

  - Iteracija skozi določen del drevesa

```
<xsl:for-each select="/oseba"/>
  ....
</xsl:for-each>
```

  - Sortiranje elementov

```
<xsl:sort select="ime"/>
```

  - Pogojni stavki

```
<xsl:if test="starost &gt; 25" >
  …
</xsl:if>
```

XML (EXtensible Markup Language)

```
<?xml version = "1.0”?>
<?xml-stylesheet  type=“text/xsl”
   href=“sports.xsl”?>
<!-- sports.xml -->

<sports>
  <game id = “1”>
    <name>Basketball</name>
    <desc>Number one in USA</desc>
  </game>

 <game id = “2”>
    <name>Soccer</name>
    <desc>Number one in Germany</desc>
  </game>

<game id = “3”>
    <name>Running</name>
    <desc>Very popular</desc>
  </game>

</sports>
```

```
<?xml version = "1.0”?>
<!-- sports.xsl -->

<xml-stylesheet version=“1.0”
 xmlns:xsl=“http://www.w3.org/1999/XSL/Transform”>
<xsl:template match = “/”>

<html>
  <head>
    <title>Sports</title>
  </head>

<body>
  <table>
   <thead>
     <tr>
       <th>ID</th>
       <th>Sport</th>
       <th>Description</th>
     </tr>
   </thead>
   <xsl:for-each select = “/sports/game”>
     <tr>
       <td><xsl:value-of select = “@id”/></td>
       <td><xsl:value-of select = “name”/></td>
       <td><xsl:value-of select = “desc”/></td>
     </tr>
   </xsl:for-each>
  </table>
</body>
</html>

</xsl:template>
</xsl:stylesheet>
```

osebe.xsl

XML (EXtensible Markup Language)

- Druge XML tehnologije
  - XSL-FO (Extensible Stylesheet Language Formatting Objects)
  - XQuery
    - Poizvedbe XML strukture po vzoru SQL jezika
    - Uporablja XPath sintakso
  - XLink, XPointer, SOAP, WSDL, RDF, RSS, SVG

10. JavaScript Object Notation
JSON

JSON

- Enostavni (lightweight) standard za izmenjavo podatkov
  - **.json**
- Temelji na jeziku JavaScript
  - Objekti in nizi (arrayi)
- Douglas Crockford (2001)
- Osnovna sintaksa
  - Osnovni tipi: String, Number, Boolean, Array, Object, null
  - Poljubno število presledkov

JSON

- Primer:

```
{
    "ime": "Uroš",
    "priimek" : "Slokar",
    "starost"  : 25,
    "naslov"  :
    {
        "ulica": "Ljubeljska 15",
        "mesto"         : "Ljubljana",
        "država"        : "Slovenija",
        "številka"    : 1000
    },
    "telefon":
    [
        {
          "tip"  : "domači",
          "številka": "01 3454-567"
        },
        {
          "tip"  : "mobilni",
          "številka": "064 567 234"
        }
    ]
}
```

JSON

```
{
  "menu": "File",
  "commands": [
      {
          "title": "New",
          "action":"CreateDoc"
      },
      {
          "title": "Open",
          "action": "OpenDoc"
      },
      {
          "title": "Close",
          "action": "CloseDoc"
      }
   ]
}
```

```
<?xml version="1.0" ?>
<root>
  <menu>File</menu>
  <commands>
     <item>
        <title>New</value>
        <action>CreateDoc</action>
     </item>
     <item>
        <title>Open</value>
        <action>OpenDoc</action>
     </item>
     <item>
        <title>Close</value>
        <action>CloseDoc</action>
     </item>
  </commands>
</root>
```

JSON_izpis.html

JSON

- Primerjava XML in JSON formata

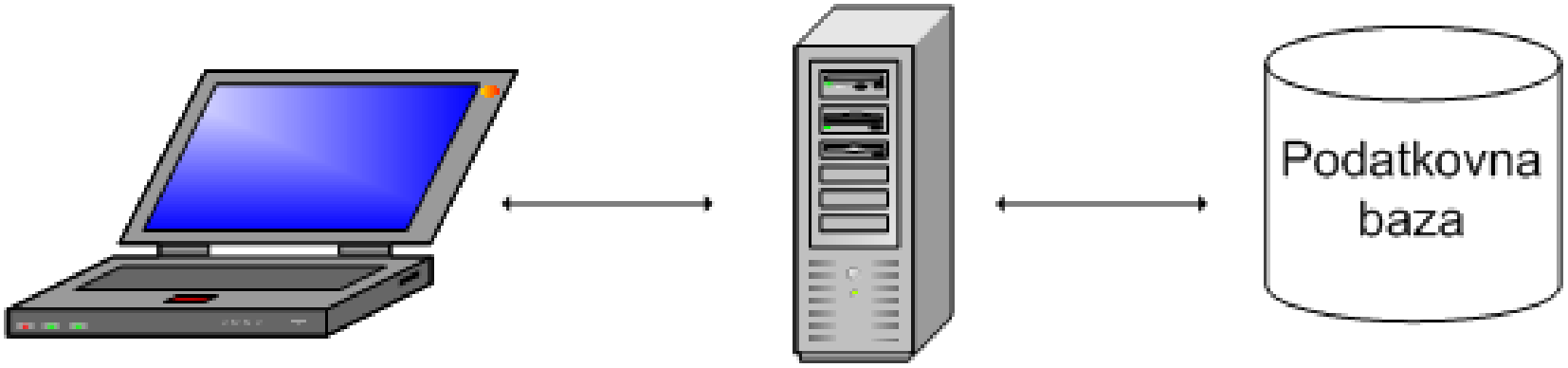
JSON_ajax.html

  - Prednosti JSON formata
    - Hitrost procesiranja in razčlenjevanja
    - Enostavnost uporabe
    - Naravna podpora v JS okolju
  - Prednosti XML formata
    - Razširljivost
    - Dobra podpora v vseh razširjenih programskih jezikih
    - Boljša berljivost strukture
- Razčlenjevanje v brskalniku
  - Vgrajen JSON objekt

```
var obj = eval('(' + json_obj + ')');

var obj = JSON.parse(json_obj);
```

11. Aktivne spletne strani – programska koda na strežniku

Trislojna arhitektura

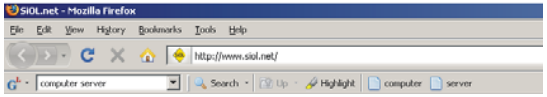


Odjemalec
Uporabniški sloj

Strežnik
Logični sloj

Informacijski
sloj

Strežnik - odjemalec

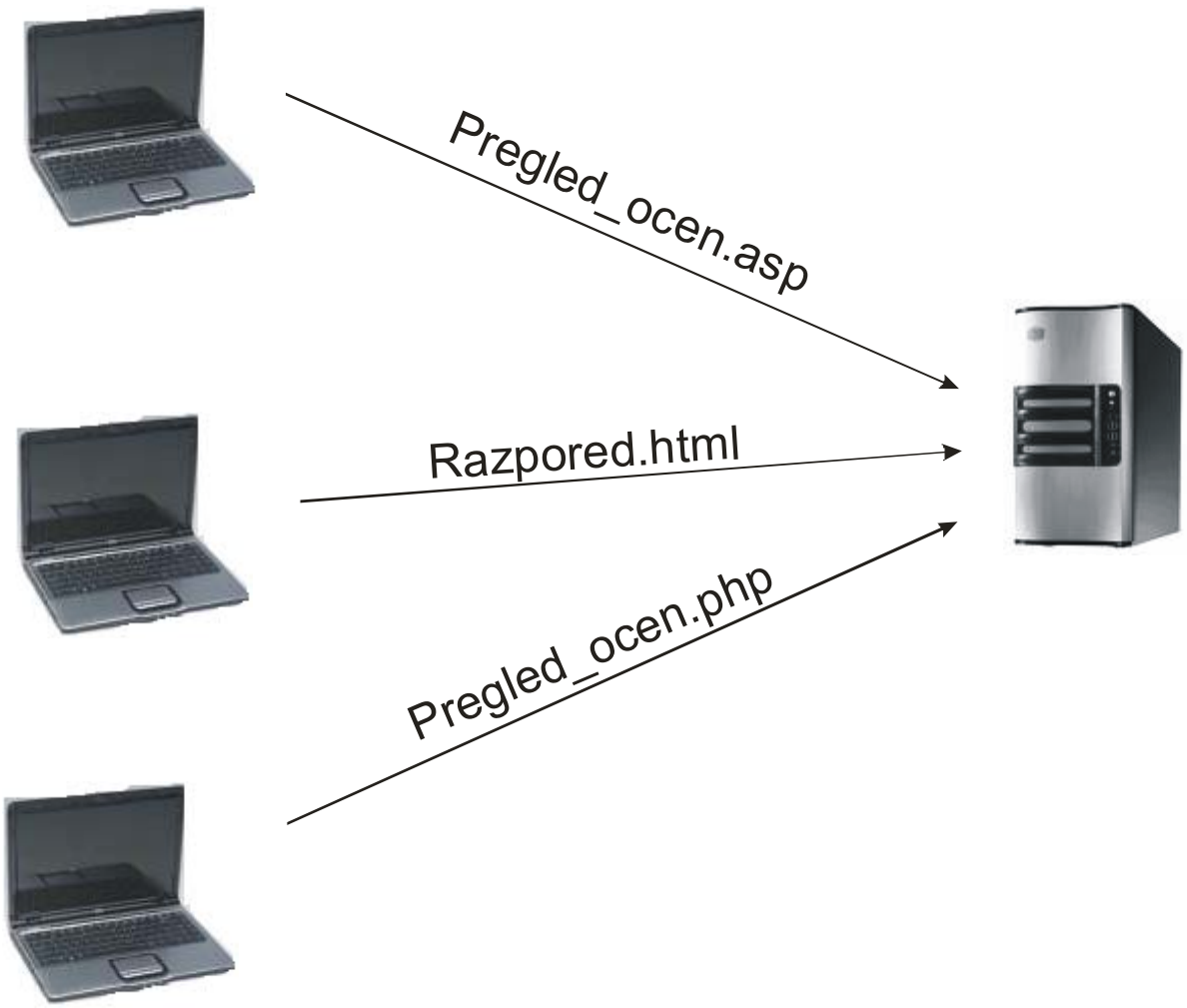


GET / HTTP/1.1
Host: www.siol.net
User-Agent: Mozilla/5.0….
Accept: text/html,application/xhtml+xml….
Accept-Language: en-us,en;…

HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Thu, 18 Sep 2008 15:17:20 GMT
X-Powered-By: ASP.NET
Content-Length: 4389
Content-Type: text/html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="sl">
<head>
        <title id="head_title">SiOL.net</title>
        <meta http-equiv="content-type" content="text/html; charset=
        <link rel="Stylesheet" type="text/css" href="http://itm.siol
        <link rel="Stylesheet" type="text/css" href="http://itm.siol
        <script type="text/javascript" src="http://itm.siol.net/js/f
        <link rel="alternate" type="application/rss+xml" title="RSS
</head>
<body>
        <script src="http://utm.siol.net/u.aspx?page=PORTAL" type="t
<div class="home" id="header">
        <div class="holder">
                <ul class="nav_corpo">  <li class="sel"><a title="Do
                <ul class="nav_tools">
```

Strežnik - odjemalec



- Programska koda na strežniku
  - Se izvede na strežniku pred pošiljanjem odgovora – strani odjemalcu
  - Odjemalec vidi le rezultat izvajanje programske kode
  - Različni programski jeziki
    - Prevedeni (C, C++, Delphi) ali skriptni (ASP, PHP, JSP, Perl, itd.)
    - Podpora s strani strežnika

Programska koda na strežniku

- Tipične naloge programske kode na strežniku
- Glavne naloge skript
  - Dinamična generacija vsebine spletne strani
    - Števci obiskov
    - Datum, ura in drugi “živi” podatki
  - Programiranje odziva na vnesene podatke uporabnika
  - Prilagoditev vsebine glede na zahteve ali zmožnosti odjemalca
  - Dostop do podatkovnih baz in prikaz rezultata v brskalniku na odjemalcu
  - Omogočanje varnosti, saj programska koda ni vidna odjemalcu

Apachi spletni strežnik

- Na voljo za operacijske sisteme UNIX, Linux, Mac OS in Windows
- Programski paket XAMPP

- Osnovne značilnosti
  - Spletni dokumenti (**…/htdocs/**)
  - Konfiguracija (**…/conf/httpd.conf**)
    - “Root” direktorij
    - “Alias”-i
    - Pravice

Skriptni jeziki za strežnike

- PHP (Hypertext Preprocessor)
  - Odprtokodni (open source) skriptni jezik
  - Posebna PHP sintaksa, ki je podobna Perl-u in C-ju
  - Strežniki, ki podpirajo PHP skripte
    - Apache
    - IIS 7.0 ali IIS 6.0 (z dodatnim FastCGI modulom)
  - Zmogljivejši od ASP jezika

Skriptni jeziki za strežnike

- ASP (Active Server Pages)
  - Programski jeziki za pisanje ASP skript
    - VBScript (poenostavljena različica Visual Basica)
    - JScript (Microsoftova različica JavaScripta)
    - Perl
  - Spletni strežniki, ki podpirajo ASP
    - Microsoft IIS
    - Sun Java System ASP
    - Personal Web Server (PWS)

Skriptni jeziki za strežnike

- JSP (Java Server Pages)
  - Temelji na programskem jeziku Java
  - Strežniki, ki podpirajo JSP skripte
    - IBM WebSphere
    - Java System Application Server
    - Apache Tomcat
    - IIS (s pomočjo Tomcat-a)
  - Zelo zmogljivo orodje v kombinaciji z Java servleti in J2EE tehnologijo

11.1. PHP skriptni jezik

PHP skriptni jezik

- Rasmus Lerdorf (1994)
  - Sledenje uporabnikov njegove spletne strani
  - Napisan v programskem jeziku C
- PHP 1 .. PHP 5.x
- PHP razčlenjevalnik
  - Del inštalacije Apachi strežnik
  - Ločena inštalacija
- Odprtokodna tehnologija
- Enostavno povezovanje in delo z bazami

PHP skriptni jezik

- PHP datoteka
  - HTML datoteka, kateri so dodani segmenti PHP kode
  - Tipična končnica .php (.php3 ali .phtml)
- Razlika med PHP in (X)HTML datoteko na strežniku
  - Ko brskalnik zahteva HTML datoteko, mu jo strežnik direktno posreduje
  - Ko brskalnik zahteva PHP datoteko, se ta posreduje t.i. PHP interpreterju (engine)
  - PHP interpreter obdela kodo vrstico po vrstico in izvede vse skripte
  - Odjemalcu se posreduje navadna HTML datoteka

- PHP sintaksa
  - PHP datoteka je zelo podobna HTML datoteki, le da vsebuje php skripte
  - Posamezne skripte so označene z

    <?php

    …

    ?>

  - **Obvezno** zaključevanje ukazov s **;**
  - Komentarji: **//** ali **/* … */**
  - Izpis besedila iz PHP: **echo** in **print**

    Primer:

```
echo “Izpis iz kode”;
print (“Izpis is kode“);
```

- Primer enostavne PHP skripte

```
<html>
    <body>

        <?php
        echo “Lep pozdrav”
        ?>

    </body>
</ html >
```

- Spremenljivke v PHP
  - Ime spremenljivke se začne z znakom $
  - Ime lahko vsebuje vse alfanumerične znake in “_”
    - Občutljivost na velike in male črke

  Primer:

```
<?php
$ime = “Janez”;
$starost = 50;
?>
```

  - “loosley typed”
    - Določanje tipa ni potrebno
- Konstante `define ( “starost”, 33);`

- Tipi spremenljivk
  - integer
  - float, double, real
  - string
  - bool, boolean
  - array
  - object
  - NULL
- Incializacija spremenljivk
  - undef

- Določanje tipa

  gettype($ime_spremenljivke)

  settype($ime_spremenljivke, “integer”)

- Pretvarjanje tipa – “casting”

  (double) $ime_spremenljivke

  (string) $ime_spremenljivke

  Ne spremeni vrednosti spremenljivke!!

Primeri:

```
$visina = “183 cm”;
settype($visina, “integer”);
```

```
$visina = “183 cm”;
print( (integer) $visina);
```

1_sintaksa.php

- Aritmetični operatorji
  - +, - , *, / y = x*5;
  - % (po modulu), ++ , -- y=10%8; y++;
- Operatorji za prirejanje
  - =, += , -= , *= , /= , .= , %= y*=x; (y=y*x;)
- Operatorji za primerjanje
  - ==, !=, <>, > , < , >= , <= 2>3 3!=4 x==2
- Logični operatorji
  - && , || , !

```
$x=1;
$y=8;
($x<3 && $y>5)
```

- Tekstovna spremenljvka – string
  - V spremenljivki ali direktno v navednicah “…” *(‘..’)*
  - Združevanje z operatorjem **.**

```
$ime = “Branka”;          $priimek = “Novak”;

echo $ime . “ “ . $priimek . “ je v službi”;
```

  - Osnovne funkcije
    - strlen($ime)

```
strlen(“Kaj delamo tu?”);                    (14)
```

    - strpos($ime, $iskani_string)

```
$stavek = “Janez je priden”;
strpos($stavek,”je”);                        (6)
```

- Ostale funkcije za delo s string-i
  - strcmp ($beseda1, $beseda2); (-1, 0, 1)
    - strncmp($beseda1, $beseda2,n);
  - strstr ($beseda, $iskano); (vrne preostanek stringa)
    - stristr ($beseda, $iskano);
  - trim ($beseda, *seznam*); (odstrani presledke na robovih)
  - str_shuffle ($beseda);
  - chr(80) in ord(“b”);

http://www.w3schools.com/php/php_ref_string.asp

- Niz (array)
  - Spremenljivka, ki hrani več vrednosti
  - Trije tipi nizov
    - Numerični (numeric)
    - Asociativni (associative)
    - Večdimenzionalni (multidimensional)

4_nizi.php

- Pogojni stavki
  - If, if … else, if … elseif … else
  - switch
- Programske zanke
  - for, foreach, while, do while

```
$imena=array(“Joc”,”Bor”,”Miha”);
 foreach ($imena as $nekdo)
{
print($nekdo);
}
```

```
do
{
   i++;
   print(“Cifra:“ . $i);
} while($i<=10)
```

2_switch.php

3_foreach.php

Funkcije

- Podajanje parametrov po referenci: **&**

```
function ImeFunkcije()
        {
         … ;
        }
```

```
function ImeFunkcije($p1, $p2)
        {
         … ;
        }
```

```
<?php
  function sestej($x, $y)
  {
     $vsota = $x+$y;
     return $vsota;
  }
  echo “12 + 23 = “ . sestej(12,23);
?>
```

```
<?php
   function funk(&$spr)
  {
      $spr++;
  }
  $a=5;
  funk($a);
  // $a ima vrednost 6
?>
```

5_funkcije.php

- Globalne in lokalne spremenljivke / SCOPE
  - Spremenljivke, ki so definirane v funkcijah so lokalne
  - Ostale spremenljivke so globalne
  - $GLOBALS: dostop do globalnih spremenljivk

```
<?php
   function test() {
      $ime = “Lokalno";
      echo “Iz globalne: “ . $GLOBALS[“ime”] . “< br />";
      echo “Iz lokalne: “ . $ime;
   }

   $ime= “Globalno”;
   test();
?>
```

6_global.php

- Obrazci
- Pošiljanje podatkov na strežnik
  - Metodi
    - GET: spremenljivke in vrednosti se prenesejo kot del URL naslova
    - POST: spremenljivke in vrednosti se prenesejo v zaglavju in niso vidne uporabniku

```
<form action=“registracija.php" method=“post">
    Uporabniško ime: <input type="text" name=“username" />
    Geslo: <input type=“password" name=“password" />
    <input type="submit" value=“Prijavi se”/>
</form>
```

- Dostop do podatkov iz obrazca v PHP skripti
  - Globalne spremenljivke / asociativni nizi
    - $_GET
    - $_POST
    - $_REQUEST
  - Vnosna polja tipa *select* in *checkbox* morajo biti definirana kot niz (array)

Validacija podatkov!!

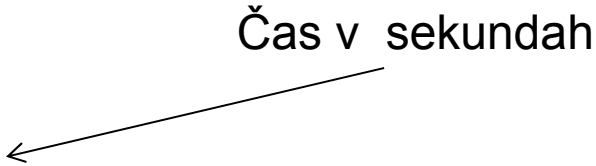
```
<?php
  echo “Uporabniško ime je: “ . $_POST[“username”];
  echo “Geslo je: “ . $_POST[“password”];
  …
?>
```

obrazec.html
obdelava.php

- Piškotki / Cookies
  - Koščki informacij, ki jih strežnik shrani na odjemalčevem računalniku
    - Se avtomatsko pošljejo na strežnik ob ponovnem obisku
  - Tekstovne datoteke brez zaščite
  - Datum veljavnosti
  - Strežnik ima dostop samo do svojih piškotkov

Piškotki

Ustvarjanje piškotka




```
<?php
  setcookie(“uporabnik", “Janez Novak", time()+3600);
?>
```

Branje piškotka

```
<?php
   // Izpit piškotka
   echo $_COOKIE[“uporabnik"];

   // Izpit vseh piškotkov
   print_r($_COOKIE);
?>
```

Brisanje piškotka

```
<?php
  setcookie(“uporabnik", “", time() - 3600);
?>
```

7_piskotki.php

- Seje
  - Hranjenje poljubnih podatkov določenem uporabniku, ki so na voljo v vseh php datotekah
  - Začetek seje
    - <?php session_start(); ?>
  - Zapis podatkov v sejno spremenljivko

```
<?php
  session_start();
  $_SESSION[“‘uporabnik”]=“janez”;
?>
```

  - Konec seje

```
unset($_SESSION[“‘uporabnik”]);
…
session_destroy();
```

seja0.html
seja1.php
seja2.php

- SSI – Server Side Includes
  - Vključevanje zunanjih PHP skript oz. združevanje različnih datotek
  - Modularna zgradba spletnih strani
  - Ukaza
    - include(), include_once()
      - opozorilo o napakah
    - require(), require_once()
      - prekinitev izvajanja skripte

vtic1.php
vtic2.php

- Napake med izvajanjem skript
  - Prekinitev izvajanja s pomočjo funkcije **die( )**

```
<?php
   if(!file_exists(“opis.txt"))
   {
      die(“Datoteka ne obstaja");
   }
  else
  {
     $file=fopen("opis.txt","r");
   }
?>
```

- Drugi načini prestrezanja napak
  - “Custom error handler”: sami določimo funkcijo, ki se pokliče, ko nastopi napaka

```
error_function(error_level,error_message,
error_file,error_line,error_context)


function mojaNapaka($st_napake, $opis)
  {
  echo "<b>Napaka:</b> [“ . $st_napake . “] “ . $opis . “/>";
  echo “Konec skripte";
  die();
  }


set_error_handler(" mojaNapaka ");
```

napaka.php

Primerjava ASP in PHP

- Cena
  - ASP – Potrebna MS licenca zaradi IIS
  - PHP – brezplačno (Linux)
- Hitrost
  - PHP se v povprečju izvaja hitreje kot ASP
- Okolje (platforma)
  - ASP – V večini primerov MS okolje (Linux + Apache)
  - PHP – Linux, Windows, Unix
- Povezovanje s podatkovno bazo
  - ASP – MS SQL
  - PHP - MySQL

ASP skriptni jezik

- ASP datoteka
  - HTML datoteka, kateri so dodani segmenti VBScript kode
  - Tipična končnica .asp
- ASP sintaksa
  - ASP datoteka je zelo podobna HTML datoteki, le da vsebuje t.i. VB skripte
  - Posamezne skripte so označene z <% … %>

    <script language="vbscript" runat="server"> …. </script>
  - Komentarji: ‘
  - Skriptni jeziki
    - VBScript (privzeti jezik)
    - JavaScript
    - PERL, REXX, itd.

ASP skriptni jezik

- Pošiljanje vsebine odjemalcu

```
<html>
    <body>
        <%
        response.write(“Lep pozdrav”)
        %>
    </body>
</ html >
```

- Skrajšana oblika

```
…
<%=“Lep pozdrav”%>
…
```

ASP skriptni jezik

- Najava spremenljivke (**brez tipa !!**)

```
<%
   Dim starost
   Dim i
   starost = “65 let”
   i = 0
%>
<%
  Const pi=3,14
%>
```

```
<%
  dim prijatelji(4)
  prijatelji(1)=“Miha”
  prijatelji(2)=“Bojan”
  prijatelji(3)=“Marjan”
  prijatelji(4)=“Aleks”
%>
```

- Globalna in lokalna spremenljivka

```
<%
    Dim letalo
    Sub Vozila(Tip)
        Dim vozilo
        vozilo = “Golf”
        letalo = “AIRBUS”
    End Sub
%>
```

- Procedure in funkcije

```
<%
Sub sestej_stevili(stevilo1, stevilo2)
    response.write(stevilo1 + stevilo2)
End Sub
%>
<%call sestej_stevili(3,5)%>

<%
Function zdruzi_besedi(beseda1,beseda2)
    zdruzi_besedi=beseda1 & beseda2
End Function
%>
<%=zdruzi_besedi (“Lep”,”Dan”)%>
```

- If, For in Case stavki

```
If vrednost = 12 Then
    …
End if
```

```
For i=0 to 10
   …
Next
```

```
Select Case (vrednost)
    Case 1
      ….
    Case 2
      …
    Case Else
      …
End Select
```

12. Podatkovne zbirke

Osnove podatkovnih zbirk

- Podatkovna zbirka – baza podatkov
  - Podatki -> informacije -> znanje
  - Strukturirana zbirka podatkov na nekem računalniku ali strežniku
    - Zbirka podatkov
    - Metapodatki
  - DMBS (Database Management System)
    - Shranjevanje
    - Povezovanje
    - Urejanje iskanje
    - Nadzor dostopa

Osnove podatkovnih zbirk

- Vrste podatkovnih zbirk glede na število uporabnikov
  - En uporabnik (single-user)
  - Več uporabnikov (multiuser)
    - Workgroup (< 50)
    - Enterprise (> 50)
- Vrste podatkovnih zbirk glede na lokacijo
  - Centralizirane
  - Porazdeljene

- Arhitektura podatkovne zbirke
  - Strojna oprema
  - Programska oprema
    - Operacijski sistem, DBMS, pomožni programi (GUI)
  - Uporabniki
    - Administratorji, načrtovalci, programerji, uporabniki
  - Postopki oz. pravila uporabe zbirke
  - Podatki

Osnove podatkovnih zbirk

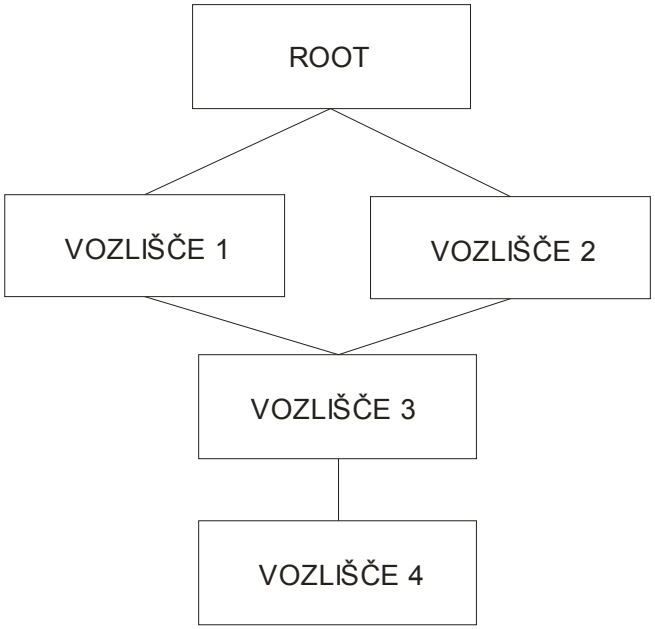
- Redundanca (redundancy)
- Nedoslednosti (inconsistency) ali podatkovne anomalije (anomaly)
  - Popravljanje, vpisovanje ali brisanje
- Integriteta podatkov (integrity)
  - Popolnost, ažurnost, točnost, zanesljivost (avtoriziranost)
  - Tri vrste integritete
    - Entitetna
    - Referenčna
    - Domenska

- Modeli podatkovnih zbirk
  - Določajo strukturo podatkov
  - Določajo atribute in podatkovne tipe
  - Določajo povezave med njimi
- Primeri modelov
  - Relacijski model
  - Hierarhični model
    - XML
  - Entitetni (mrežni) model
  - Objektni model

- Hierarhični model
  - Tipična drevesna struktura (koren, vozlišča in veje)
  - Relacija one-to-many
    - Vsako vozlišče-starš (parent) ima lahko VEČ pod-vozlišč (child)
    - Vsak otrok ima lahko SAMO ENEGA starša

    Otroci in starši so med seboj povezani z vejami (kazalci)
  - Vsak starš vsebuje kazalce do svojih otrok

- Mrežni model
  - Razširjeni hierarhični model
  - Vsak starš lahko vsebuje več otrok in vsak otrok ima lahko več staršev
    - Relacija many-to-many
  - Relacije med vozlišči so predstavljene s polji (set)



- Objektni model
  - Izvira iz objektnih programskih jezikov (Java, C++)
  - Objekti vsebujejo podatke in metode za delo z njimi
  - Objekti pripadajo razredom
  - Objekti so instance razredov
  - Med razredi in podrazred je princip dedovanja
  - Možne so kombinacije objektno-relacijskih modelov
    - Če definiramo relacije med objekti

- Relacijski model
  - RDBMS (Relational Database Management System)
  - Osnovna enota je **tabela** (table)
    - Stolpci in vrstice
  - Stolpci predstavljajo atribute
    - Enoznačna imena
    - Eden od atributov je identifikator zapisa (ID)
  - Vrstice v tabeli so zapisi
    - Vsak zapis vsebuje vse atribute
  - Povezave med podatki so del podatkov

- Relacijski model
  - Tabela
    - Zaključena enota za hranjenje podatkov
    - Unikatno ime znotraj podatkovne baze
    - Podatki so organizirani v zapise/vrstice (**records**) in polja/stolpce (**fields**)
    - Vsak stolpec (atribut)
      - Ima enoznačno določeno ime
      - Ima točno določen tip (tekst, celo število, datum, itd.)
      - Vrednosti so lahko dodatno omejene
        - Omejen obseg vrednosti (pri številih)
        - Določene preko nekega šifranta
    - Logična struktura: dejanska lokacija shranjene tabele je nepomembna

Osnove podatkovnih zbirk

- Relacijski model
  - Ključi
    - Identifikator zapisa (primary key)
    - Lahko je avtomatsko generirana zaporedna številka
    - Lahko je kombinacija več atributov
  - Relacije (povezave)
    - Shranjene v tabelah
    - Na osnovi povezovalnih atributov tabel
    - Na osnovi zunanjih ključev (povezovalne tabele)
    - Vrste relacij: 1:1, **1:***, *:*
  - Relacijski diagram
    - Grafična predstavitev relacij
  - Indeksi

- Manipulacija podatkov preko poizvedb (query-jev)
  - Poizvedbe preko RDBMS
  - Določanje kriterijev
    - Primer: “Poišči vse zapise v tabeli ***Politiki*** kjer polje ***Ime*** vsebuje ***Janez***”

      Rezultat:

001	Janez Podobnik	SLS
007	Janez Bond	CIA
011	Janez Drobnič	NSI

Komunikacija s podatkovno zbirko - SQL

- Structured Query Language
  - Standardiziran jezik za komunikacijo z relacijskimi bazami
- Osnovne funkcionalnosti jezika SQL
  - Manipulacija podatkov v bazi
    - Iskanje, vstavljanje, spreminjanje, brisanje
  - Ustvarjanje novih baz in tabel
  - Zagotavljanje integritete podatkov
  - Definicije shranjenih procedur (stored procedures)
  - Avtentikacija in avtorizacija
  - Programski stavki (for, while, if, spremenljivke, itd.)

- Osnovna SQL sintaksa
  - Spreminjanje vsebine zbirke (DML)
    - SELECT
    - UPDATE
    - DELETE
    - INSERT INTO
  - Spreminjanje strukture zbirke (DDL)
    - CREATE DATABASE
    - CREATE TABLE
    - DROP TABLE
    - CREATE INDEX
    - …

- SELECT stavek
  - Iskanje podatkov v zbirki glede na podane kriterije

```
SELECT Ime FROM Osebe

SELECT * FROM Osebe

SELECT * FROM Osebe WHERE Ime="Janez"
```

- SELECT stavek
  - FROM: ciljna tabela(e)
  - Primerjave v WHERE stavku
    - =, <>
    - < , >
    - >=, <=
    - BETWEEN, LIKE, IN
  - Logične primerjave: AND in OR

```
SELECT * FROM Osebe
WHERE (Ime="Janez" AND Priimek<>"Drobnič")
```

- SELECT stavek

```
SELECT * FROM Osebe
WHERE Ime LIKE “%ane%”
```

```
SELECT * FROM Osebe
WHERE Starost BETWEEN 13 AND 23
```

```
SELECT Priimek FROM Osebe
WHERE Ime = “Janez”
```

- INSERT stavek

```
INSERT INTO Osebe VALUES (“Jaka”, “Lakovič”, …)

INSERT INTO Osebe (Ime, Priimek) VALUES (“Manca”, “Špik”)
```

- UPDATE stavek

```
UPDATE Osebe SET Ime=“Jakec”
WHERE Ime=“Jaka” AND Priimek=“Lakovič”
```

- DELETE stavek

```
DELETE FROM Osebe WHERE Priimek=“Drobjnak”

DELETE FROM Osebe
```

SQL

- Združevanje rezultatov
  - COUNT, MIN, MAX, SUM, AVG
- Sortiranje zapisov
  - ORDER BY, GROUP BY

```
SELECT * FROM Osebe
WHERE (Ime="Janez" AND Priimek<>"Drobnič")
ORDER BY Priimek
```

- Združevanje tabel
  - JOIN, LEFT JOIN, RIGHT JOIN

```
SELECT Osebe.Ime, Podjetja.Ime
FROM Osebe LEFT JOIN Podjetja
ON Osebe.ID_podjetja = Podjetje.ID
```

SQL

- Standardni podatkovni tipi
  - varchar, smallint, int, real, float, date, time, timestamp, itd.
- Funkcije za delo s spremenljivkami oz. tabelami
  - avg(), count(), first(), last(), sum(), itd.
  - ucase(), lcase(), mid(), len(), itd.

PHP in podatkovne baze

- MySQL
  - Najpopularnejša odprtokodna podatkovna baza
  - RDBMS
    - Ni uporabniškega vmesnika!!
    - Dodatna orodja za administracijo
      - phpMyAdmin
  - Deluje na skoraj vseh operacijskih sistemih (Unix, Linux, Mac OS, Windows, itd.)
  - Del inštalacije XAMPP paketa
  - Ni popolnoma skladna s standardno SQL sintakso

PHP in podatkovne baze

- PHP upravlja z MySQL bazo s pomočjo jezika SQL
- Vzpostavljanje povezave

```
$povezava = mysql_connect(servername,username,password);

mysql_select_db(“ime_baze", $povezava);
```

- Rušenje povezave

```
mysql_close($povezava);
```

- Primeri poizvedb

```
$povezava = mysql_connect(“localhost”,”jaka”,”ggg111”);
 mysql_select_db(“ime_baze", $povezava);

 $rezultat= mysql_query("SELECT * FROM Avtomobili");

 mysql_query("INSERT INTO Avtomobili(Znamka, Model, Letnik)
 VALUES (‘Renault', ‘Clio', 2009)");

 mysql_query("UPDATE Avtomobili SET Letnik= 2010
 WHERE Znamka= ‘Renault' AND Model = ‘Twingo'");

 mysql_close($povezava);
```

- Primer obdelave vrnjenih podatkov

```
$povezava = mysql_connect(“localhost”,”jaka”,”ggg111”);
 mysql_select_db(“ime_baze", $povezava);

 $rezultat= mysql_query("SELECT * FROM Avtomobili");

 while($row = mysql_fetch_array($rezultat))
 {
 echo $row[‘Znamka'] . " " . $row[‘Model'];
 echo "<br />";
 }

 mysql_close($povezava);
```

baza.php
obrazec_baza.php
vpis.php

PHP in podatkovne baze

- Open Database Conectivity – ODBC
  - Zagotavlja standardni vmesnik za dostop do RELACIJSKIH podatkovnih zbirk
  - Vmesni sloj med aplikacijo in gonilnikom (driverjem) podatkovne zbirke
  - Neodvisen od OS, vrste podatkovne zbirke in programskega jezika aplikacije
  - Temelji na jeziku SQL (Structured Query Language) -
  - Podpora velikemu številu vrst podatkovnih zbirk

- PHP in ODBC

```
$povezava =odbc_connect(‘moja_baza','','');
$sql="SELECT * FROM Avtomobili";
$zapis=odbc_exec($ povezava,$sql);


 while (odbc_fetch_row($zapis))
  {
   $znamka=odbc_result($zapis,“Znamka");
   $model=odbc_result($zapis,“model");
   echo "<tr><td>$znamka</td>";
   echo "<td>$model</td></tr>";
 }

 odbc_close($zapis);
```

13. CGI – Common Gateway Interface

Common Gateway Interface - CGI

- Program, ki teče na internetnem strežniku
- Vmesnik med zunanjimi aplikacijami in internetnim strežnikom
- Prilagajanje vsebine glede na zahteve odjemalca
  - Procesiranje vhodnih podatkov odjemalca
  - Dostop do podatkovnih baz na strežniku
  - Ustvarjanje vsebine v realnem času (on the fly)
  - Nadzor nad dostopom in pravicami odjemalcev

Common Gateway Interface

- CGI programi
  - Napisani v poljubnem programskem jeziku (C, C++, Perl, itd.)
    - Skriptni ali prevedeni jeziki
      - Hitrost in zanesljivost programskih jezikov
      - Berljivost kode in odpravljanje napak
      - Prenosljivost programov
  - Sprejemajo vhodne podatke od internetnega strežnika
    - Podpora standardnim vhodno-izhodnim tokovom: STDIN in STDOUT
  - Izhodni podatki se posredujejo direktno odjemalcu

Common Gateway Interface

- Okoljske spremenljivke (“environment variables”)
  - Posredovanje podatkov o zahtevi ustreznemu CGI programu
  - Strežnik nastavi vrednosti teh spremenljivk, ko odjemalec pokliče določeno CGI skritpo
  - SERVER_NAME
  - SERVER_PORT
  - REQUEST_METHOD
  - QUERY_STRING
  - REMOTE_HOST
  - REMOTE_ADDR
  - CONTENT_LENGTH

Common Gateway Interface

- Sprejemanje podatkov od odjemalca
  - S pomočjo metod:
    - GET: podatki so del URL naslova in sledijo znaku “?”
    - POST: podatki so dodani na konec glave HTTP zahteve
  - GET
    - Podatki se shranijo v spremenljivko QUERY_STRING
  - POST
    - Podatki se shranijo v običajni vhodni tok STDIN
  - Sintaksa shranjenih podatkov

    ime1=vrednost1&ime2=vrednost2
  - CGI program vse podatke sprejme kot en niz (string)

- Primer klica CGI programa
  - S pomočjo metode POST

```
<form method="post" action="http://moj_streznik.com/cgi-bin/Obdelaj_obrazec.cgi">
        <div>
        <label>Ime: <input type="text" name="ime" /></label>
        <label>Priimek: <input type="text" name="priimek" value="Novak" /></label>
        …
        </div>
</form>
```

  - S pomočjo metode GET

```
<a href="http://moj_streznik.com/cgi-bin/
Obdelaj_obrazec.cgi?Ime=Janez&Priimek=Novak">Obdelaj</a>
```

- Sprejemanje podatkov od odjemalca
  - POZOR: posebni znaki v zahtevi so zapisani s pomočjo ASCII kode: <, >, ", ', {, }, |, \, ^, ~, [,], `
  - Posebni znaki: %XX
  - Presledek: +
  - Primer zapisa podatkov:

    Ploščina kroga se izračuna kot pi*r^2.

    Ploščina+kroga+se+izračuna+kot+pi*r%5E2%2E

  - V QUERY_STRING oz. STDIN se nahajajo podatki v kodirani obliki

Common Gateway Interface

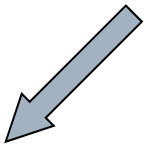
- Stvari, na katere je potrebno paziti
  - Preverjanje pravilnosti oz. smiselnosti vhodnih podatkov
  - Skrivanje informacij o programih in servisih na strežniku
  - Onemogočanje uporabe tehnologije Server Side Includes (SSI)

    <!--#exec cgi="/cgi-bin/format.cgi" -->
  - Možnost zlonamerne uporabe programov na strežniku

Common Gateway Interface

- Pošiljanje odgovora odjemalcu
  - Strežnik odgovori odjemalcu preko standardnega izhodnega toka STDOUT
  - Odgovor je lahko HTML dokument, slika, povezava na nek drug dokument, itd.
  - V odgovoru je potrebno definirati tip dokumenta, ki je vsebovan v odgovoru

POZOR: prazna vrstica



Primer HTML dokumenta:

Content-type: text/html\n\n

Common Gateway Interface

- V odgovoru odjemalcu moramo v celoti določiti vrnjeno HTML vsebino

  PRIMER:

```
printf("Content-type: text/html\n\n");
printf("<html><body>");
printf("<p>Tole je preprost CGI odgovor!</p>\n");
printf( "</body></html>\n");
```

- Drugi tipi vrnjenih dokumentov:

  Content-type: image/gif\n\n

  Location: http://www.drug_streznik.com\n\n

Common Gateway Interface

Primer CGI programa

```
#include <iostream.h>
#include <stdlib.h>

void main()
{
  char* lpszRemoteHost = getenv("REMOTE_HOST");

  cout << "Content-type: text/html" << endl << endl
   << "<html>" << endl
   << "<body>" << endl
   << "<p>" < endl
   << "Hello, "
   << lpszRemoteHost
   << "!" << endl
   << "</p>" << endl
   << "</body>" << endl
   << "</html>";
}
```

Klic CGI programa

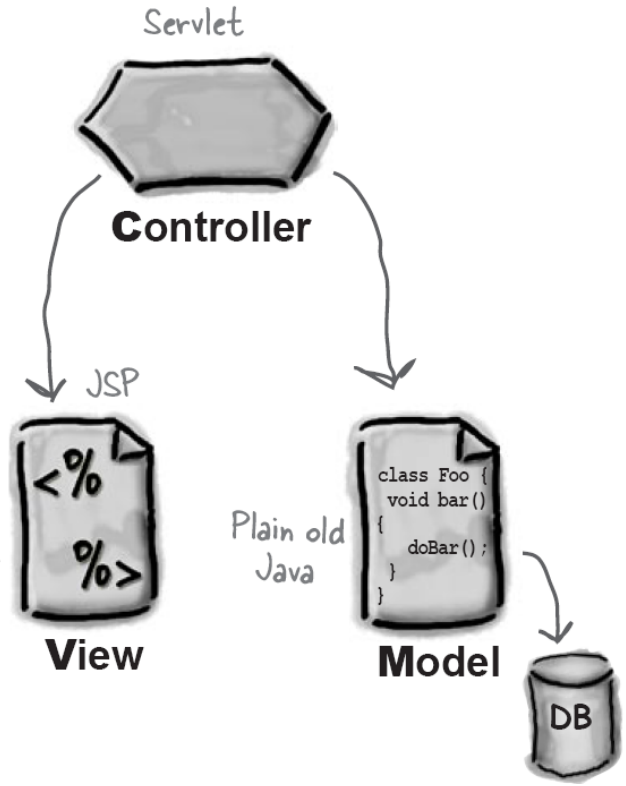
```
<form method="post"
action="http://path.to/my/cgi/program">
…
```

14. Java SERVLET tehnologija

Java Servlet tehnologija

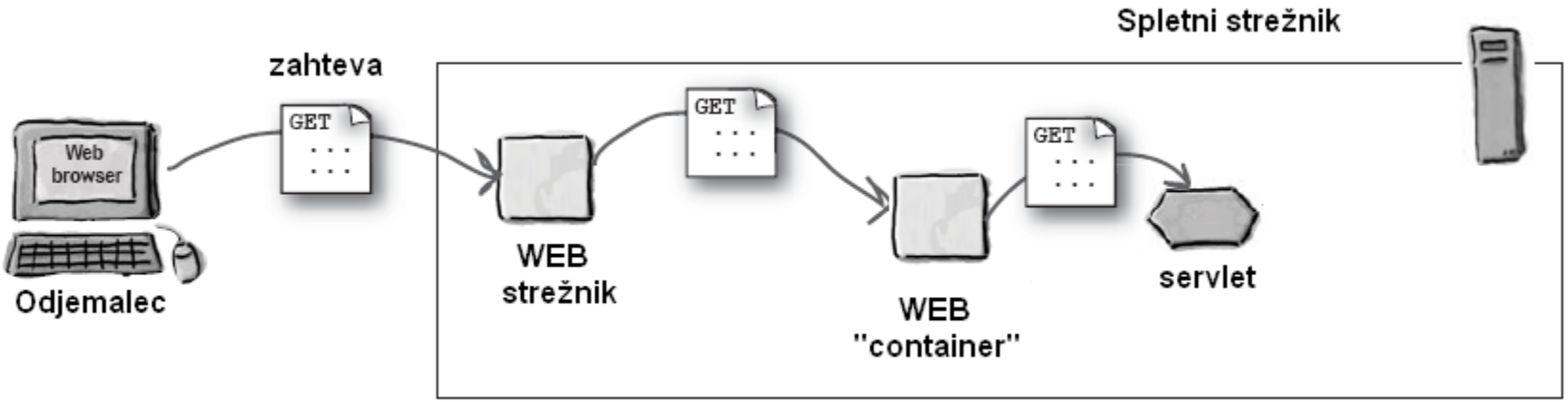
- Namenjena dinamični generaciji spletnih strani oz. vsebine, ki jo strežnik vrne odjemalcu
- Servlet tehnologija je podobna CGI tehnologiji
- Servleti so programi razviti v Javi
  - Za poganjanje je potreben JVM na strežniku
  - Poleg spletnega strežnika potrebujemo na strežniku tudi t.i. aplikacijski strežnik (container)

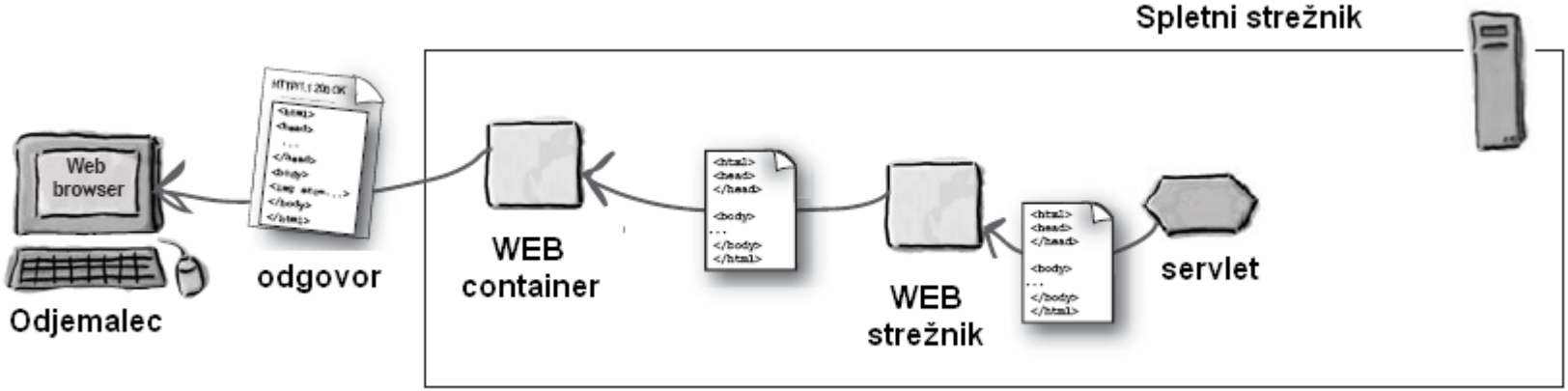
- MVC model spletne aplikacije (Model-View-Controller)
  - Model
    - Programska logika celotne aplikacije
    - Komunikacija z bazo
  - View
    - Prikaz, izpis (output)
  - Controller
    - Komunikacija z uporabnikom
    - Posrednik med moduloma Model in View



[O'REILLY, Head First: Servlets & JSP]

Java Servlet tehnologija

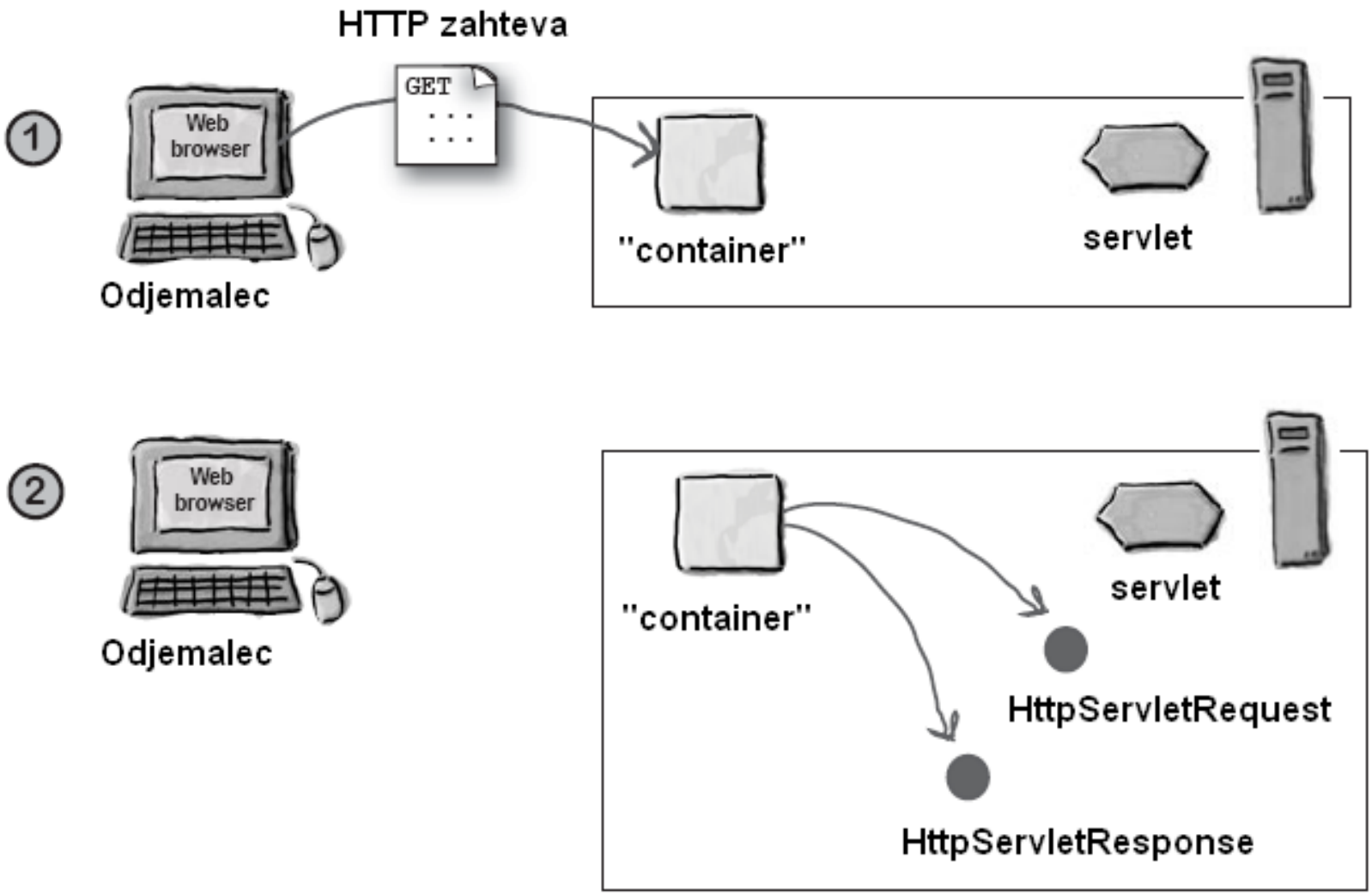




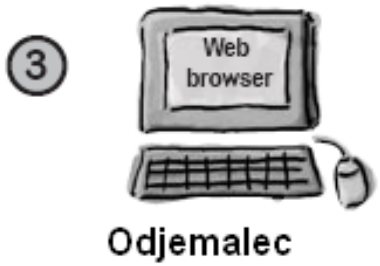
Java Servlet tehnologija

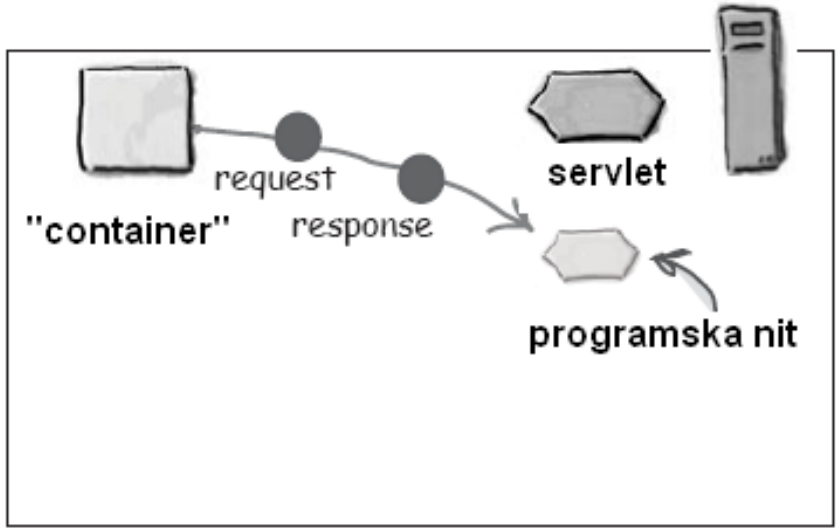
- “Container”
  - Skrbi za komunikacijo med spletnim strežnikom in servletom
  - Skrbi za inicializacijo in nalaganje in uničenje Java razredov
  - Skrbi za zaganjanje vzporednih programskih niti ob vsakem klicu servleta
  - Skrbi za varnost in nadzor dostopa do programov na strežniku
  - Skrbi za povezavo med servleti in JSP skriptami

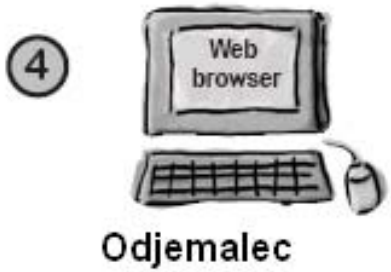
Java Servlet tehnologija

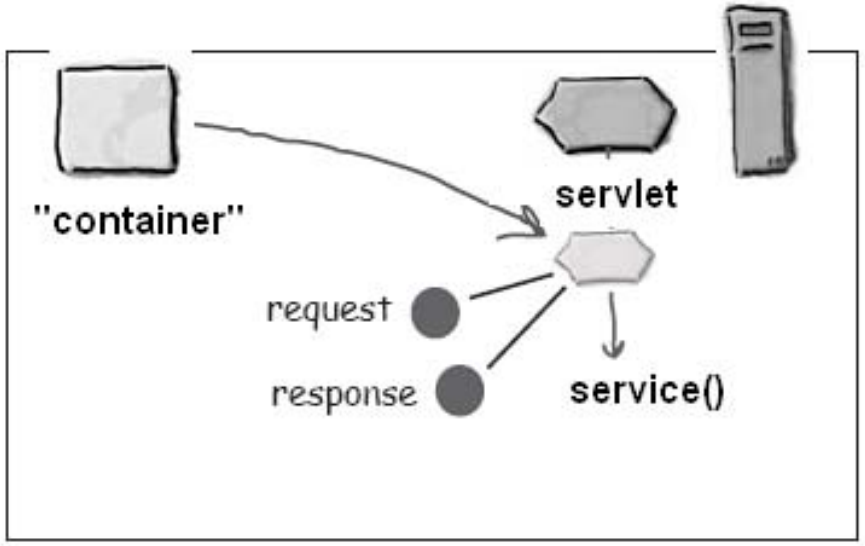


Java Servlet tehnologija

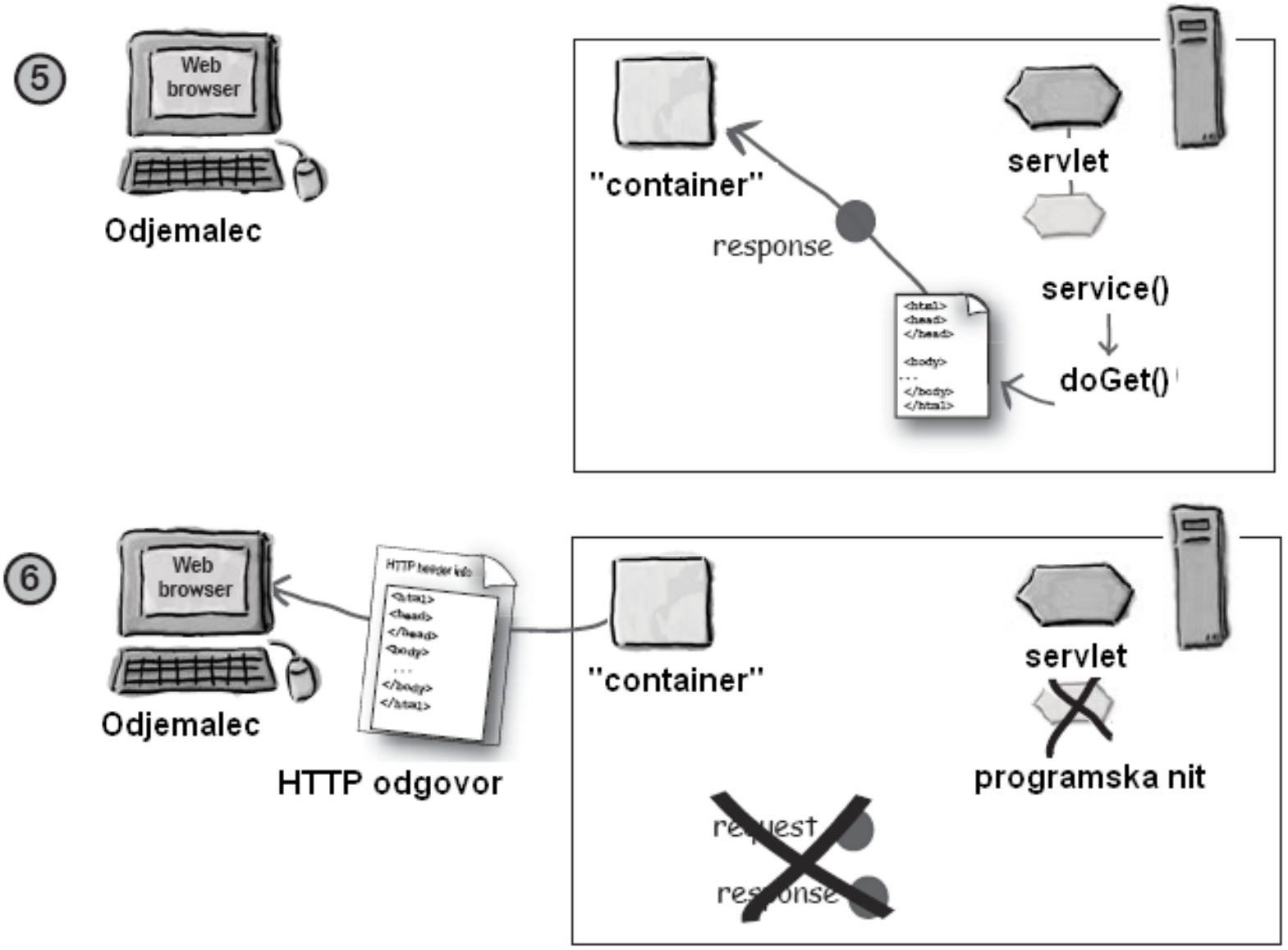








Java Servlet tehnologija



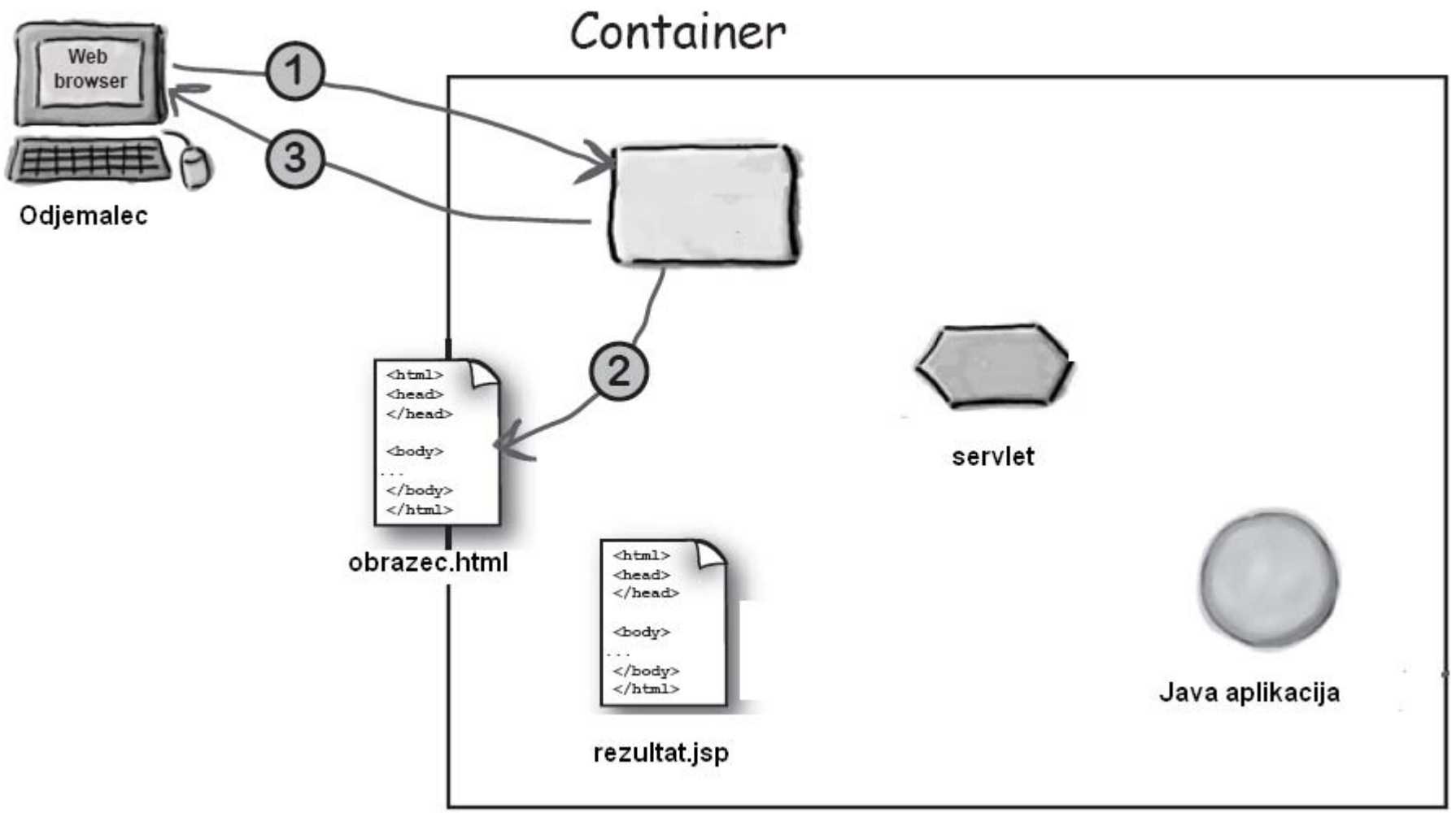
Primer servlet programa

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class Moj_Servlet extends HttpServlet {

    public void doPost(HttpServletRequest request, HttpServletResponse
    response) throws IOException
        {
        PrintWriter izpis = response.getWriter();
        java.util.Date danes = new java.util.Date();
        izpis.println("<html> <body> <p> Današnji datum je: " +
        danes + "</p> </body> </html>");
        }
}
```

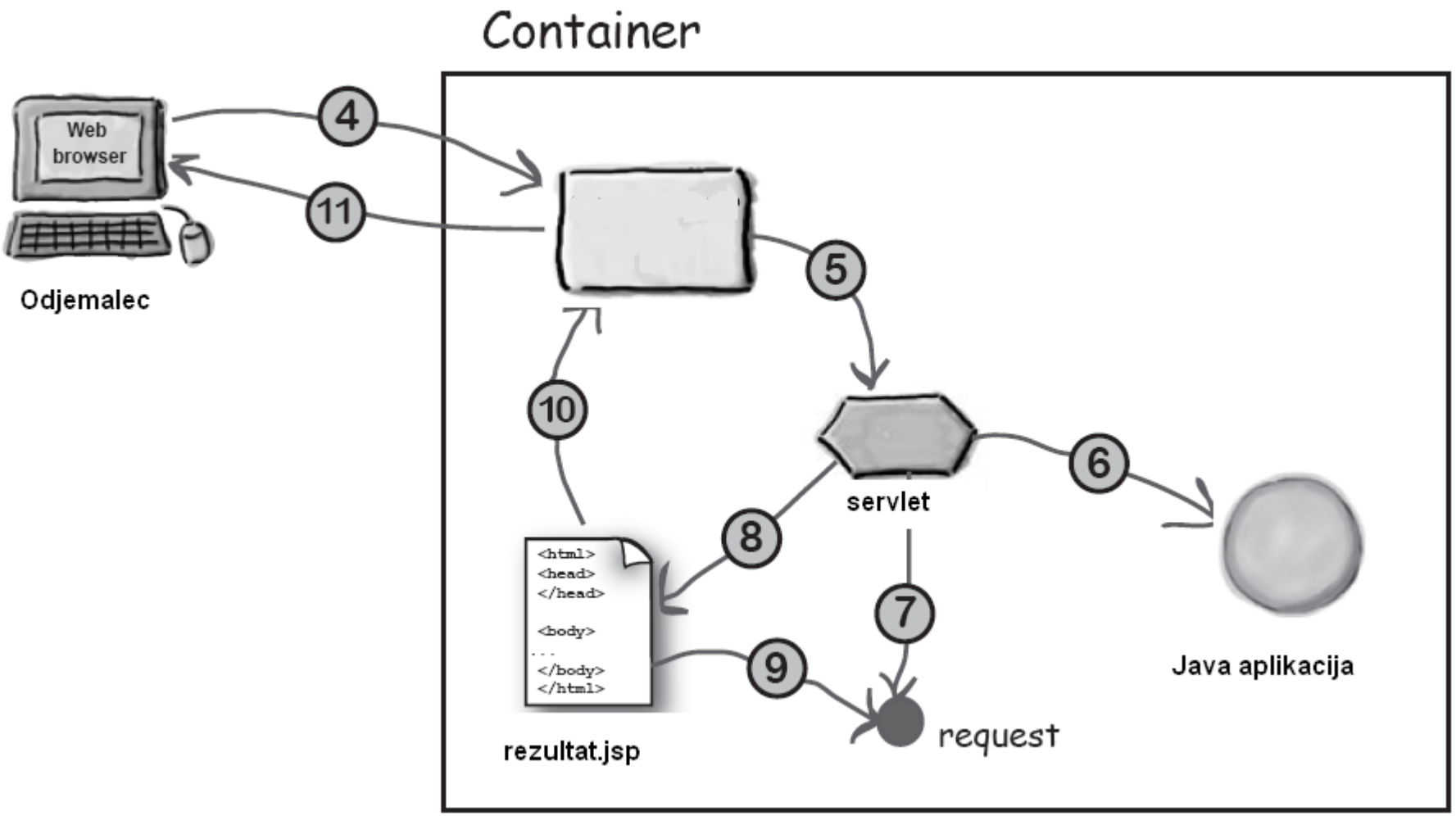
- Deployment Descriptor (DD)
  - Določa povezavo med Java razredom v aplikaciji (servletom) in URL naslovom preko katerega je ta program dostopen
  - Temelji na XML sintaksi

```
<web-app ...>
    <servlet>
        <servlet-name>Moje lokalno ime</servlet-name>
        <servlet-class>com.si.primer.Moj_servlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>Moje lokalno ime</servlet-name>
        <url-pattern>/Primer</url-pattern>
    </servlet-mapping>
</web-app>
```

Java Servlet tehnologija – primer aplikacije



Java Servlet tehnologija – primer aplikacije



Java Servlet tehnologija – primer aplikacije

obrazec.html

```
<html><body>
...
<form method="POST" action=“Noga">
      <select name="velikost_noge">
            <option value="41"> 41 </option>
            <option value="42"> 42 </option>
            <option value="43"> 43 </option>
            ...
            <option value="46"> 46 </option>
</form>
</body></html>
```

Java aplikacija

```
public class javaAplikacija {
        public String komentirajVelikost(int Velikost)
        {
                if (Velikost > 44){
                        return "BIG FOOT";
                } else {
                        return "SMALL FOOT";
                }
        }
}
```

Servlet

```
import javax.servlet.*;
...
public class Servlet_VelikostNoge extends HttpServlet {
    public void doPost(HttpServletRequest request, HttpServletResponse
    response) throws IOException
    {
        int noga = (int)request.getParameter("velikost_noge");
        javaAplikacija aplikacija = new javaAplikacija();
        String komentar = aplikacija.komentirajVelikost(noga);
        request.setAttribute("mojKomentar", komentar);
        RequestDispatcher izgled =
        request.getRequestDispatcher("rezultat.jsp")
        izgled.forward(request, response);
    }
}
```

Deployment descriptor

```
<web-app ...>
    <servlet>
        <servlet-name>Servlet_VelikostNoge</servlet-name>
        <servlet-class>lkn.si.sampleApp.VelikostNoge</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>Servlet_VelikostNoge</servlet-name>
        <url-pattern>/Noga</url-pattern>
    </servlet-mapping>
</web-app>
```

Java Servlet tehnologija – primer aplikacije

JSP datoteka (rezultat.jsp)

```
<html>
<body>
...
<%
    String komentar = request.getAttribute("komentar");
    out.print("<p> KOMENTAR: " + komentar + " </p>");
%>

</body>
</html>
```

8_Servlet_primer.rar

15. Spletne storitve in deljeni viri informacij

Spletne storitve

- Service (storitev)
  - Del programske opreme, ki nudi določeno funkcionalnost drugi programski opremi
  - Deljeni vir (shared resource)
  - Nima “človeškega” uporabniškega vmesnika
  - Servisni strežniki
    - Print server, file server, database server, application server, itd.
- SOA (Service-oriented architecture)
  - Ločitev vmesnika (interfaces) od implementacije (bussines logic)
  - Vmesnik omogoča uporabo storitve in hkrati skriva njene podrobnosti

Spletne storitve

- Webservice
  - Aplikacija, ki ima svoj spletni aplikacijski programski vmesnik (API)
  - API (Application Programming Interface)
    - Komunikacija med dvema aplikacijama
  - Komunikacija med aplikacijami preko spleta
    - Na višjem (aplikacijskem) nivoju
    - Standardni internetni protokoli (HTTP ali SMTP)
    - XML struktura podatkov
    - Neodvisno od okolja in programskega jezika
  - Posplošenje tehnologij kot so
    - RPC, CORBA, RMI

- Osnovni pojmi
  - SOAP (Simple Object Access Protocol)
    - XML dokument
    - Zahteva: Določa operacijo oz. funkcijo, ki se ob klicu izvede na strežniku in vsebuje vhodne podatke za klic funkcije
    - Rezultat: Izhodni podatki oz. rezultat izvedbe funkcije
  - WSDL (Web Service Definition Language)
    - XML dokument + XML Schema
    - Definicija storitve (webservice-a) – definicija API-ja
      - Vhodni in izhodni podatki (tipi podatkov)
      - Nabor operacij oz. funkcij, ki jih storitev nudi

- Razvoj webservice-a
  - Uporaba različnih orodij, ki avtomatsko generirajo service ali njegove posamezne dele
  - Bottom-up princip
    - Razvoj na osnovi obstoječe programske kode oz. modulov
    - Java class-i in interface-i
    - Avtomatska generacija WSDL dokumenta
  - Top-down princip
    - Avtomatska generacija programske kode na osnovi obstoječega WSDL dokumenta

- Primer
  - Valutni pretvornik (euro, dolar, kuna)
  - Uporaba orodja Eclipse
    - Programski jezik Java
    - Apachi Tomcat aplikacijski strežnik
  - Razvoj servisa
    - Java interface
      - Določa operacije, ki jih bo nudil service (API)
    - Implementacija interface-a
    - Avtomatska generacija WSDL-a in servisa
  - Razvoj odjemalske aplikacije
    - Na osnovi WSDL dokumenta
    - Omogoča testiranje servisa

- Java interface
  - Definicija

```
public interface CurCon extends java.rmi.Remote {
    public ExchangeValues fromDollars(double dollars)
        throws java.rmi.RemoteException;
    public ExchangeValues fromEuros(double euros)
        throws java.rmi.RemoteException;
    public ExchangeValues fromKunas(double kunas)
        throws java.rmi.RemoteException;
}

public class ExchangeValues {
        public double dollars;
        public double euros;
        public double kunas;
}
```

- Java interface
  - Implementacija

```
public class CurConImpl {

public ExchangeValues fromDollars(double dollars)
throws java.rmi.RemoteException
{
        ExchangeValues ev = new ExchangeValues();
        ev.dollars = dollars;
        ev.euros = dollars * 0.7;
        ev.kunas = dollars * 0.7 * 7;
        return ev;
}
public ExchangeValues fromEuros(double euros)
throws java.rmi.RemoteException
        {

        …

}
```

- Struktura WSDL dokumenta
  - Root element: <definition>
  - Element <types>
    - Definicija vseh tipov s pomočjo XML Scheme
  - Elementi <message>
    - Nabor vhodnih parametrov za neko funkcijo ali ime in tip izhodnih podatkov (tudi tipe)
    - Vsak parameter je določen z značko <part>
  - <portType>
    - Določa operacije servisa
    - Niz elementov <operation>, ki vsebujejo po en <input> in <output>
    - <input> in <output> določajo prej definirana sporočila

- Struktura WSDL dokumenta
  - <binding>
    - Določa način klicev funkcij, ki so določene preko elementov <operation>
    - Način pretvorbe WSDL definicije v SOAP sporočilo
    - Določa protokole za komunikacijo (ni nujna uporaba SOAP-a)
  - <service>
    - Določa ime celotnega servisa
    - Element <port> določa URL naslov servisa

- Tipi podatkov
  - Osnovni (javanski) tipi podatkov se pretvorijo v XML tipe, ki so določeni s pomočjo XML Scheme
  - Vgrajeni tipi (integer, decimal, double, string, dateTime, itd.)
  - Kompleksni tipi (zgrajeni s pomočjo osnovnih tipov)

```
<complexType name="ExchangeValues">
   <sequence>
      <element name="dollars" type="xsd:double"/>
      <element name="euros" type="xsd:double"/>
      <element name="kunas" type="xsd:double"/>
   </sequence>
</complexType>
```

- Struktura SOAP sporočila
  - Pravilni XML dokument
  - Root element <Envelope>
    - Obvezna definicija imenskega prostora
      <xmlns:env="http://www.w3.org/2001/12/soap-envelope">
      <xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
    - Opcijsko polje <Header>
      - Specifični podatki aplikacije (avtentikacija)
    - Obvezno polje <Body>
      - Nabor operacij in parametrov
      - Lahko določi svoj imenski prostor
      - Opcijsko polje <Fault>

- Primer SOAP zahteve

```
POST /konverter HTTP/1.1
Host: www.moj_streznik.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soapenv:Envelope
    xmlns:soapenv="http://www.w3.org/2001/12/soap-envelope"

   <soapenv:Body xmlns:d="http:// www.moj_streznik.org/konverter">
      <d:EvroVDolar>
         <d:Evro>12,35</d:Evro>
      </d:EvroVDolar>
   </soapenv:Body>

</soapenv:Envelope>
```

Primer SOAP odgovora

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soapenv:Envelope
    xmlns:soapenv="http://www.w3.org/2001/12/soap-envelope"

   <soapenv:Body xmlns:d="http:// www.moj_streznik.org/konverter">
      <d:EvroVDolarResponse>
         <d:Dolar>16,01</d:Dolar>
      </ d:EvroVDolarResponse>
   </soapenv:Body>

</soapenv:Envelope>
```

9_WebService_primer.rar

Webservice

- REST (Representational State Transfer)
  - Poenostavljen (lightweight) način webservice arhitekture
  - Temelji na HTTP protokolu
    - Metodi GET (pogosteje uporabljena) in POST
  - Omogoča branje, pisanje, popravljanje, brisanje podatkov na oddaljenem strežniku
  - Neodvisen od platforme in programskega jezika

Primerjava REST in SOAP zahteve

SOAP

```
<?xml version="1.0"?>
<soap:Envelope
      xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
      soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
   <soap:body pb="http://www.telekom.si/imenik">
       <pb:PodatkiOOsebi>
          <pb:IDOsebe>12345</pb:IDOsebe>
       </pb:PodatkiOOsebi>
   </soap:Body>
</soap:Envelope>
```

REST

http://www.telekom.si/imenik/Oseba/12345

http://www.telekom.si/imenik/ Oseba?ime=Janez&priimek=Novak

- REST odgovor
  - Oblika ni predpisana
    - Enostaven tekst, XML, CVS, JSON
    - HTML odgovor ni primeren!

```
<parts-list>
   <part id="3322">
      <name>ACME Boomerang</name>
      <desc>
         Used by Coyote in <i>Zoom at the Top</i>, 1962
      </desc>
      <price currency="usd" quantity="1">17.32</price>
      <uri>http://www.acme.com/parts/3322</uri>
  </part>
 …
</parts-list>
```

Webservice

- Dokumentacija REST storitve
  - WSDL (se ne uporablja pogosto)
  - WADL (Web Application Description Language)

```
<method name="GET" id="ItemSearch">
 <request>
   <param name="Service" style="query“ fixed="AWSECommerceService"/>
   <param name="Version" style="query" fixed="2005-07-26"/>
   <param name="Operation" style="query" fixed="ItemSearch"/>
   <param name="SubscriptionId" style="query“ type="xsd:string" required="true"/>
   <param name="SearchIndex" style="query"
</request>
<response>
  <representation mediaType="text/xml“  element="aws:ItemSearchResponse"/>
</response>
</method>
```

Webservice

- Večina strežnikov ponuja REST in SOAP vmesnik
  - Primeri:
    - Google search REST API

      https://ajax.googleapis.com/ajax/services/search/web?v=1.0&q=Kobe%20Bryant
    - Facebook API (Graph API)

      https://graph.facebook.com/cocacola

      https://graph.facebook.com/jakasodnik

- Splošna navodila pri razvoju REST storitev
  - URL-ji ne smejo biti fizični (brez končnice)
  - Omejena količina informacije v odgovoru
    - Razbitje na manjše kose
  - Odgovor mora biti dobro dokumentiran
    - Enostavnejša in preglednejša implementacija odjemalca
  - Odgovor lahko vsebuje linke na nove odgovore (podrobnosti)

RSS

- RSS (Really Simple Syndication)

  - Omogoča objavo ali prikaz izbrane vsebine določene spletne strani
  - “Naročilo“ na neko vsebino oz. nit (ang. feed)
  - Avtomatsko osveževanje in pridobivanje sveže vsebine
  - Prilagojen prikaz določene vsebine
  - Primeren za spletne strani
    - Podjetij
    - Spletnih medijev
    - Spletnih koledarjev

- Struktura RSS vsebine
  - XML datoteka
    - Veljajo vsa pravila XML sintakse
    - Več verzij RSS formata (0.91, 1.0, 2.0)
  - Elementi
    - <rss version="2.0“>: root drevesa
    - <channel>: označuje nit
      - <title>, <link>, <description>: obvezni elementi
      - <item>: poljubno število novic
        - <title>, <link>, <description>: obvezni elementi
        - <category>, <author>, <enclosure> :opcijski elementi

RSS

```
<rss xmlns:mrss="http://www.rssboard.org/media-rss" version="2.0">
    <channel>
        <title>Dnevnik.si</title>
        <link>http://www.dnevnik.si</link>
        <description>Dnevnikov RSS, zadnje novice</description>
    <item>
        <title>
          Ljubljanski podmladek SD Pahorju: "Se vam ne zdi, da je počasi dovolj?"
        </title>
        <link>http://www.dnevnik.si/novice/slovenija/1042528087</link>
         <description>
          Pred kongresom SD, ki bo 2. junija, bo konec tedna tudi seja podmladka
          stranke. …
        </description>
        <category>Slovenija</category>
        <enclosure url="http://www.dnevnik.si/uploads/image_cache/4da166d2ec10
                    68b3c263c7363accbddf.jpeg" length="9965" type="image/jpeg"/>
        <pubDate>Tue, 08 May 2012 17:57:10 +0200</pubDate>
    </item>
        ….
```

RSS

- Objava RSS niti
  - Pripravimo XML datoteko in dodamo povezavo na spletni strani
    - Obstajajo spletni validatorji za pravilnost datoteke
  - Dodamo povezavo v spletni agregator (RSS Feed Directory)
  - Dodamo povezavo na voljo iskalnikom
  - Redno osveževanje vsebine!!
    - Orodja za avtomatsko osveževanje
- Branje RSS niti
  - RSS readerji (programčki)
  - Dodatek v spletni strani
    - Orodja oz. predloge

FeedDamon
RSS_viewer.php
Rss2html.zip

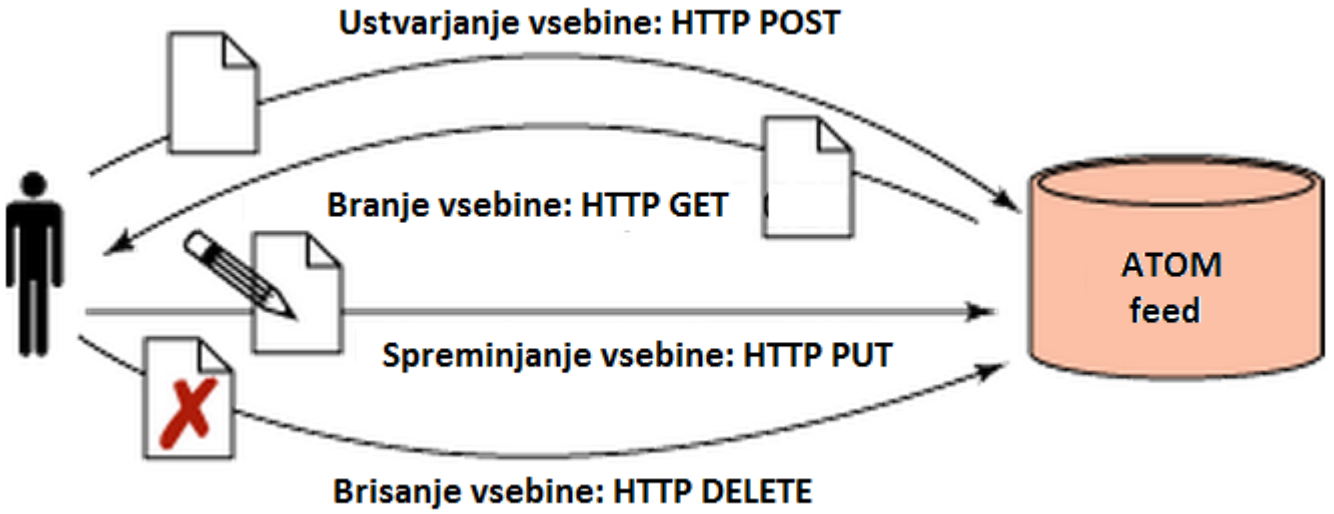
ATOM

- ATOM (Atom Syndication Format)
  - Alternativa RSS nitim
  - Bolj natančna in stroga specifikacija
  - Lahko vsebuje različne formate podatkov
    - HTML, XHTML, XML, binary, itd.
  - Razširljiva sintaksa
  - Standard (IETF)
    - Atom Syndication Format (2005)
    - Atom Publishing Protocol (2007)

ATOM

```
<feed xmlns="http://www.w3.org/2005/Atom"
      xml:lang="en"
      xml:base="http://www.example.org">
     <id>http://www.example.org/myfeed</id>
     <title>My Simple Feed</title>
     <updated>2005-07-15T12:00:00Z</updated>
     <link href="/blog" />
     <link rel="self" href="/myfeed" />
     <entry>
       <id>http://www.example.org/entries/1</id>
       <title>A simple blog entry</title>
       <link href="/blog/2005/07/1" />
       <updated>2005-07-15T12:00:00Z</updated>
       <summary>This is a simple blog entry</summary>
     </entry>
     <entry>
       <id>http://www.example.org/entries/2</id>
       <title />
       <link href="/blog/2005/07/2" />
       <updated>2005-07-15T12:00:00Z</updated>
       <summary>This is simple blog entry without a title</summary>
     </entry>
</feed>
```

- Atom Publishing Protocol
  - Dinamično spreminjanje in posodabljanje ATOM niti (feed-ov)
  - Uporaba HTTP protokola
    - Metode GET, POST, PUT, DELETE



ATOM

- Primer branje vsebine

```
GET /servicedocument HTTP/1.1
Host: example.org
```

- Primer dodajanja vsebine

```
POST /blog/entries HTTP/1.1
Host: www.example.org
Content-Type: application/atom+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0" ?>
<entry xmlns="http://www.w3.org/2005/Atom">
  <title>Atom-Powered Robots Run Amok</title>
  <link href="http://example.org/2003/12/13/atom03"/>
  <id>urn:uuid:1225c695-cfb8-4ebb-aaaa-80da344efa6a</id>
  <updated>2003-12-13T18:30:02Z</updated>
  <author><name>James</name></author>
  <summary>Some text.</summary>
</entry>
```

16. Web 2.0

Web 2.0

- Kaj je web 2.0
  - Ni več stroge delitve na ponudnike vsebin (strežnike) in odjemalce
  - Uporabnik spleta je aktivni soustvarjalec vsebine
    - Kolektivna inteligenca
    - Blogi
    - Označevanje in komentiranje (tagging)
    - Socialna omrežja

- Iskalniki
  - Glavno orodje za iskanje vsebine na spletu
  - SERP (search engine results page)
  - Web crawler (ants, bots, Web spiders, itd.)
  - Različni algoritmi
    - Google search
      - PageRank: patentiran algoritem
        - Vhodni (inbound linki)
        - Vsebina (ključne besede, naslovi, podstrani)
      - Unikatna infrastruktura na osnovi več kot milijona strežnikov
      - AsWords: pay-per-click (PPC)
    - Yahoo!
      - Imenik priljubljenih spletnih strani
      - Zaradi prevelikega obsega dodan iskalnik

- Iskalniki
  - Lokacijsko pogojeno iskanje
  - SEO (Search Engine Optimization)
    - Povečevanje rankinga in vidljivosti strani
    - Upoštevanje delovanja iskalnikov
    - Dovoljene (white hat) in prepovedane (black hat) metode

Web 2.0

- Vsebina ustvarjena s strani uporabnikov
  - Zbiranje in hranjenje podatkov o uporabnikih
    - Primer Amazon, Google, Apple, itd.
  - Wikis
    - Uporabniki spreminjajo in dodajajo vsebino na strani
      - Sodelovanje velikega števila uporabnikov (collaborative filtering)
    - Primer: Wikipedia vs. Britanica

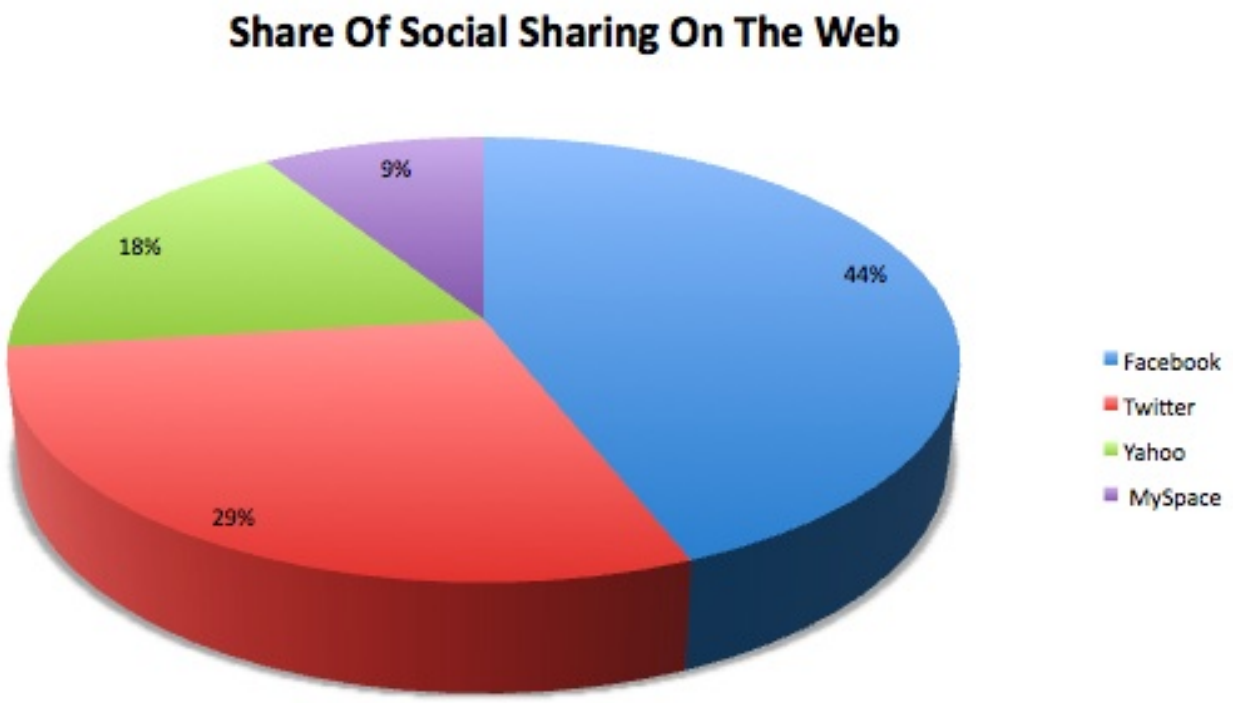
- Vsebina ustvarjena s strani uporabnikov
  - Blogi
    - Webblogs: ažurni seznam zanimivih spletnih strani
    - Kronološki urejeni zapisi posameznikov (poljubna in zelo različna tematika)
      - Razmišljanja, eseji, tehnične informacije, itd.
    - Sestavni deli blogov
      - Komentarji bralcev
      - Permalinks: unikatni URL naslovi za posamezne dele
      - Trackbacks: informacije o obiskovalcih
      - Blogroll: seznam drugih sorodnih ali priljubljenih blogov
      - RSS ali ATOM niti
    - Programska oprema
      - WordPress, Joomla, Drupal, itd.

Web 2.0

- Vsebina ustvarjena s strani uporabnikov
  - Socialna omrežja
    - Socialna omrežja vs. spletna socialna omrežja
    - Network Effects
      - Vrednost omrežja narašča s številom uporabnikov (Metcalf-ov zakon)
    - Tendenca ponudnikov je deljenje čim večje količine informacij
    - Varnost in zasebnost?!
    - Najpopularnejša omrežja
      - Facebook, Friendster, MySpace, LinkedIn, Mixi, itd.

Web 2.0

- Vsebina ustvarjena s strani uporabnikov
  - Socialna omrežja



[Socialmediaexaminer.com]

Web 2.0

- Vsebina ustvarjena s strani uporabnikov
  - Multimedija
    - Youtube
      - Nalaganje in ocenjevanje videov
      - Številna orodja za številne platforme
    - Spletna televizija in radio
      - Pandora (omejeno na ZDA)
    - Slike
      - Flickr, Picasa

- Vsebina ustvarjena s strani uporabnikov
  - Tagging (označevanje)
    - Označevanje vsebine z namenom dodajanja informacij in organizacije
    - Tag Clouds
    - Folksonomije (folksonomy = taxonomy + folk)
      - Klasifikacije na osnovi oznak
      - Drugačen način iskanja vsebine
      - Osnova za semantični splet
      - Flickr: označevanje slik
    - Social bookmarking
      - Deljenje svojih povezav in oznak

17. (Ne)Varnost na spletu

Primeri spletnih napadov

Napadi na spletni strežnik in spletne aplikacije

- Cilji
  - Dostop do nepooblaščenih informacij
  - Pogajanje lastnih aplikacij
- Glavni problemi
  - Dva tipa spletnega strežnika pokrivata 80% trga
    - Apache (HTTPD) in Microsoft IIS
  - Hitra rast spletnih vsebin brez razmisleka o varnosti
    - Spletni dostop do pomembnih vsebin
  - Spletne aplikacije
    - Vedno enostavnejši načini vdorov in zlorab

Napadi na spletni strežnik in spletne aplikacije

- Več vrst varnostnih lukenj
  - Pomanjkljivosti spletnega strežnika
  - Privzete nastavitve na strežniku in CGI skripte
  - Pomanjkljivosti v spletni aplikaciji

Napadi na spletni strežnik in spletne aplikacije

- Odkrivanje pomanjkljivosti spletnega strežnika
  - Iskanje znanih pomanjkljivosti
    - Posebni programčki - ˝skenerji˝
      - Pregledovanje portov in enumeracija
    - Primer: Napad na točno določeno stran na strežniku
      - Ukaz HEAD za pridobitev informacij o strežniku
      - Ukaz GET za pridobitev informacij o obstoju datoteke
      - Interpretacija odgovora: “ Server error“ ali “404“
  - Iskanje novih in neznanih pomanjkljivosti
    - Na osnovi izvorne kode strežnika
    - Na osnovi “reverse“ inženiringa

Napadi na spletni strežnik in spletne aplikacije

- Odkrivanje privzetih datotek na strežniku in CGI skript
  - Privzete datoteke na strežnikih
    - Demonstracija delovanja in zmogljivosti strežnika
    - Seznami datotek
  - Seznami nezaščitenih CGI skript
    - Uporaba “ skenerjev“

GET /neka_skripta.cgi HTTP/1.0

Odgovori:

404 File not Found ali

200 OK

php.ini

Napadi na spletni strežnik in spletne aplikacije

- Pomanjkljivosti v spletnih aplikacijah
  - Velika verjetnost napak v kodi in potencialnih varnostnih lukenj
- Vrste napadov na spletno aplikacijo
  - Pridobivanje informacij iz spletnih strani
    - HTML komentarji, izpisi napak
  - Dostop do nepooblaščenih datotek
    - Primer:

      http://streznik/cgi-bin/izpis?ime_datoteke=vreme

Napadi na spletni strežnik in spletne aplikacije

- Vrste napadov na spletno aplikacijo
  - Izvrševanje nedovoljenih ukazov

    http://streznik/cgi-bin/ping?ip=192.168.1.1

  - Injicirani ukazi za delo z bazo (query injection)
    - SQL sintaksa omogoča združevanje ukazov

    http://streznik/cgi-bin/poizvedba?pogoj='Miha'

    SELECT * from Osebe WHERE ime = *pogoj*

cgi.exe ping
registracija.html

Napadi na spletni strežnik in spletne aplikacije

- Vrste napadov na spletno aplikacijo
  - Skriptni napadi med različnimi stranmi (cross-site scripting)
    - Podtaknjena JavaScript koda (aplikacija)
    - ~ 80% vseh napadov na spletu
    - Nepooblaščen dostop, kraja vsebine piškotkov
    - Dva tipa podtaknjene kode
      - Začasna: enkratni napadi pogosto preko HTML obrazcev (primer: iskalniki po straneh)
      - Stalna: škodljiva koda se shrani na strežnik (primer: blogi in forumi)
    - Naprednejši napadi z uporabo DOM-a in AJAX-a
    - Napadi preko vtičnikov (Flash, ActiveX, VBScript, itd.)

Napadi na spletni strežnik in spletne aplikacije

- Vrste napadov na spletno aplikacijo
  - Nepooblaščeni dostop s spreminjanjem avtorizacijskih in avtentikacijskih podatkov
  - Spreminjanje vsebine spletnih obrazcev pred pošiljanjem na strežnik

- GLAVNA REŠITEV
  - Verifikacija in validacija podatkov na strežniku in odjemalcu

Beef DEMO

Napadi preko odjemalca

- Pridobitev dostopa do nekega sistema
- Najšibkejši varnostni člen je uporabnik!
  - Uporabnik sam zažene škodljivo kodo
    - Klik na lažno povezavo ali zagon pripete datoteke
- Trije načini napadov
  - Phishing (ribarjenje)
  - Uporaba socialnih omrežij
  - Lastni in prilagojeni programčki za vdor

Napadi preko odjemalca

- Phishing
  - Osnovni in najenostavnejši način izkoriščanja človeških slabosti
  - “Fishing”
  - Pridobivanje informacij o “tarči”
  - Dve fazi
    - Prehodno zbiranje informacij
    - Usmerjen napad z uporabo teh informacij
  - Dva načina napada
    - Usmerjen napad na točno določeno “tarčo”
    - Posplošen napad z iskanjem najšibkejših členov

Phishing

- Phishing
  - Usmerjen napad na točno določeno “tarčo”
    - Zelo osebno usmerjen napad
      - Osebni phishing: se nanaša na neko težavo osebne narave, ki naj bi jo pomagali rešiti (se predstavimo kot nek stanovski kolega ali somišljenik)
      - Profesionalni phishing: se nanaša na nek točno določeno profesionalno področje ali uspeh tarče, prošnja po pomoči na nekem strokovnem področju
  - Splošen napad
    - Usmerjen na veliko množico ljudi (več potencialnih žrtev)

Phishing

Primer phishing pošte VISA

Subject: Attention! Several VISA Credit Card bases have been LOST!

Good afternoon, unfortunately some processings have been cracked by hackers, so a new secure code to protect your data has been introduced by Visa. You should check your card balance and in case of suspicious transactions immediately contact your card issuing bank. If you don't see any suspicious transactions, it doesn't mean that the card is not lost and cannot be used. Probably, your card issuers have not updated information yet. That is why we strongly recommend you to visit our website and update your profile, otherwise we cannot guarantee stolen money repayment. Thank you for your attention. Click here and update your profile.

Phishing

Date:	Fri, 10 Sep 2004 02:11:32 +0500
From:	"Citi" <antifraud.ref.num757237203220543@citibank.com>
To:	Hampshirestefanieinsd@yahoo.com
Subject:	Attention To All Citibank Clients [Fri, 10 Sep 2004 02:19:32 +0500]

citi

Dear CitiBank customer,

Recently there have been a large number of identity theft attempts targeting CitiBank customers. In order to safeguard your account, we require that you confirm your banking details.

This process is mandatory, and if not completed within the nearest time your account may be subject to temporary suspension.

To securely confirm your Citibank account details please go to:

https://web.da-us.citibank.com/signin/scripts/login/user_setup.jsp

Thank you for your prompt attention to this matter and thank you for using CitiBank!

Citi® Identity Theft Solutions

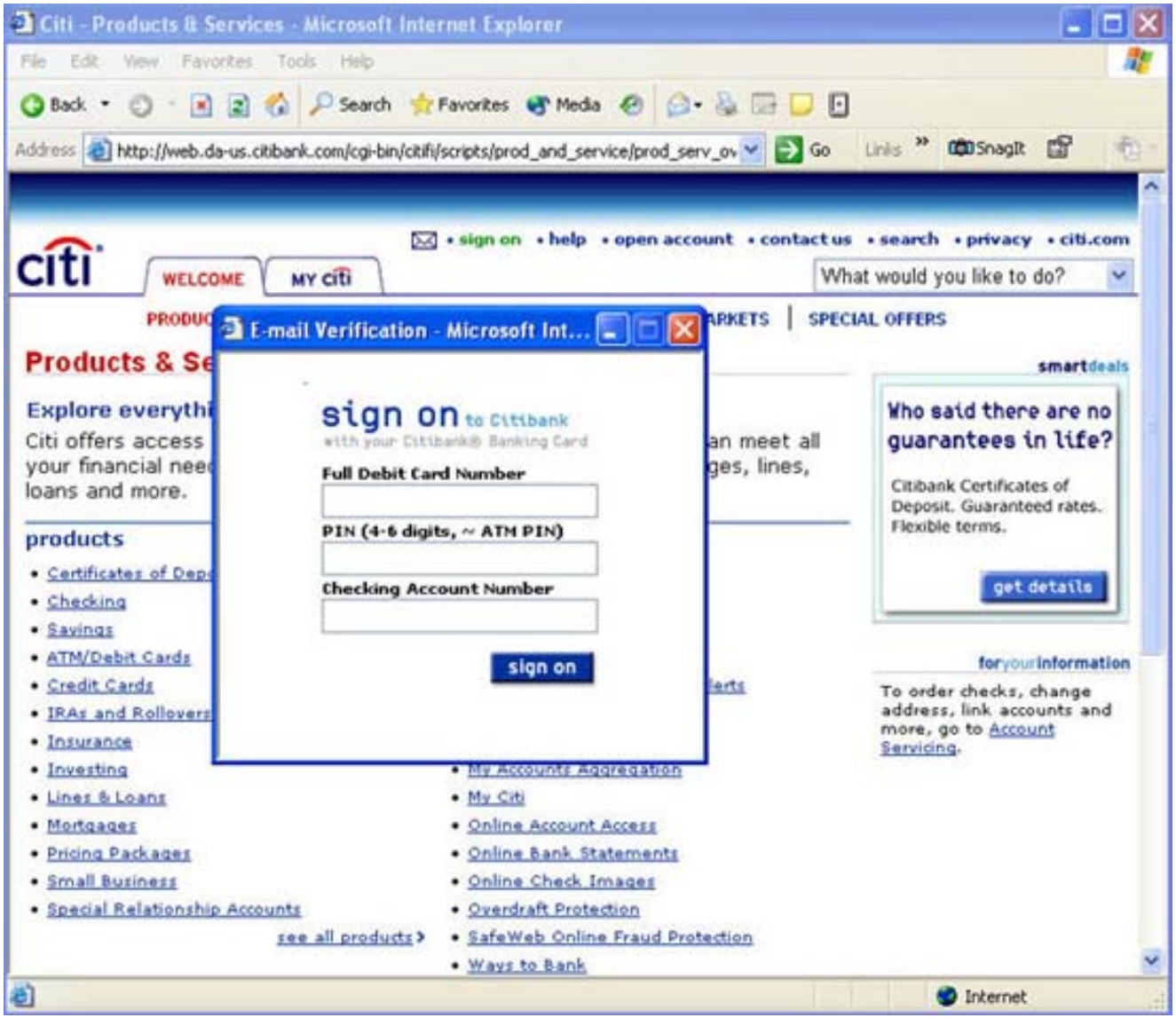
Do not reply to this email as it is an unmonitored alias

A member of citigroup

Phishing

- Osnovne tehnologije phishinga
  - Spletni obrazci
    - Kraja osnovnih podatkov
    - Osnovni obrazci za vnos imena, maila, naslova, telefonskih številk raznih ID-jev
  - Enostavne spletne aplikacije
    - Kraja podrobnosti (gesla, PIN kode, itd.)
    - Spletne aplikacije za registracijo
      - Več korakov vnosa, lažna verifikacija oz. zaščita z geslom
    - Psihologija!!

Phishing


Citi - Products & Services - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://web.da-us.citibank.com/cgi-bin/citifi/scripts/prod_and_service/prod_serv_ov
Go Links SnagIt
• sign on • help • open account • contact us • search • privacy • citi.com
citi
WELCOME
MY citi
What would you like to do?
SPECIAL OFFERS
Products & Se
Explore everythi
Citi offers access
your financial nee
loans and more.
products
Certificates of Dep
Checking
Savings
ATM/Debit Cards
Credit Cards
IRAs and Rollovers
Insurance
Investing
Lines & Loans
Mortgages
Pricing Packages
Small Business
Special Relationship Accounts
see all products
My Accounts Aggregation
My Citi
Online Account Access
Online Bank Statements
Online Check Images
Overdraft Protection
SafeWeb Online Fraud Protection
Ways to Bank
E-mail Verification - Microsoft Int...
sign on to Citibank
with your Citibank® Banking Card
Full Debit Card Number
PIN (4-6 digits, ~ ATM PIN)
Checking Account Number
sign on
smartdeals
Who said there are no guarantees in life?
Citibank Certificates of Deposit. Guaranteed rates. Flexible terms.
get details
foryourinformation
To order checks, change address, link accounts and more, go to Account Servicing.
Internet

Phishing

- Osnovne tehnologije phishinga
  - Malware
    - Škodljiva koda, ki jo želimo podtakniti
    - Izkoriščanje varnostnih lukenj v brskalnikih
      - Zastarele verzije
      - Vtičniki (Flash, Java)
    - Prevzem nadzora nad napadenim računalnikom
    - Način nameščanja
      - Posnemanje dobronamernih programov
      - Lažne izvršilne datoteke (lahko tudi PDF ali DOC!)

Phishing - malware

- Virusi
  - Dodatek “regularnim” programom
  - Se sami razmnožujejo
    - Na druge datoteke v računalniku
    - Na druge računalnike
  - Škodljivi ali neškodljivi
- Črvi (worms)
  - Ne potrebujejo uporabniške akcije za zagon
  - Se sami kopirajo na druge računalnike in razmnožujejo

Phishing - malware

- Trojanski konji
  - Maskirani kot neki uporabni programi
  - Nalaganje in zagon drugih programov (spyware)
  - Keyloger-ji
- Spyware
  - Sledenje in beleženje uporabnika na regularnih straneh
    - Keyloger-ji
    - Obiskane spletne strani, logi, itd.
  - Zbiranje in pošiljanje osebnih podatkov na oddaljen naslov za potrebe reklam
- Adware
  - Oglaševanje (hitro lahko preraste v spyware)

Napadi z uporabo socialnih omrežij

- Splošne značilnosti socialnih omrežij
  - Povezave in skupine oseb na osnovi
    - Poznanstev ali prijateljstev
    - Pripadnosti organizaciji ali skupnosti
    - Družinskih vezi
    - Interesnih podobnosti
    - Itd.
  - Povezave v “online” socialnih omrežjih v veliki večini prekašajo povezave v realnem življenju
  - Enostavna komunikacija in vzdrževanje stikov z velikim številom oseb hkrati
  - Elementi: fotografije, povezave, aplikacije, status

Napadi z uporabo socialnih omrežij

- Phishing preko socialnih omrežij
  - Napad na osebo, določeno skupino uporabnikov ali naključno skupino
  - Varnostna politika določenega socialnega omrežja
    - Omejitve zaradi obojestranskega interesa
    - Pošiljanje sporočil (“prijateljstvo”)
  - Pridobitev prijateljstva in pridružitev skupni
    - Osebni oz. personalizirani napadi
- Malware v socialnih omrežjih
  - Lažni programčki, ki jih “tarča” sama požene
    - Igre, chat programi, itd.

Napadi z uporabo socialnih omrežij

- Uporaba prijateljskih povezav v omrežju
  - Lažna “persona”, ki pridobi zaupanje
  - Izraba interesnih aktivnosti tarče
    - Status update z vabilom

Primer:

“SDS je javno užalil vse lastnike športnih trenirk! Upri se takšnemu ravnanju in podpri našo skupino TRENIRKA LOVERS”

Napadi z uporabo socialnih omrežij

- Fotografije
  - Vsebina fotografij
  - EXIF (Exchangeable image file format)
    - Časovni žig, informacije o kameri, GPS koordinate
- Povezave in odnosi (relationships)
  - Odražajo način povezave med osebami
  - Trdnost povezav (primer: married)
  - Prijateljev prijatelj (friend of a friend)

Napadi z uporabo socialnih omrežij

- Aplikacije
  - Večina socialnih omrežij dovoli uporabo aplikacij drugih razvijalcev (third-party)
  - Različne pravice dostopa do uporabniški podatkov
  - Iskanje varnostnih lukenj v obstoječih aplikacijah (ki jih “tarča” uporablja)
  - Identifikacija interesnega področja
  - Primer: “cheat” za FarmVille

Napadi z uporabo socialnih omrežij

- Status
  - Odraža trenutno stanje uporabnika
  - Vir dodatnih informacij
  - Prisotnost ali odsotnost na določeni lokaciji
  - Fizično in psihično počutje

Napadi s pomočjo lastnih in prilagojenih programov

- Napadi na točno določeno osebo ali sistem
  - Iskanje določenih informacij na računalnikih in omrežjih
  - Dostava s pomočjo phishinga
  - Prehodne raziskave in pridobivanje informacij
    - Osebne ali zdravstvene težave
    - Interesi in hobiji
  - Dejanske aplikacije z določenimi funkcionalnostmi in podtaknjeno zlonamerno kodo
    - Povečanje zaupanja, nove možnosti napadov

Napadi s pomočjo lastnih in prilagojenih programov

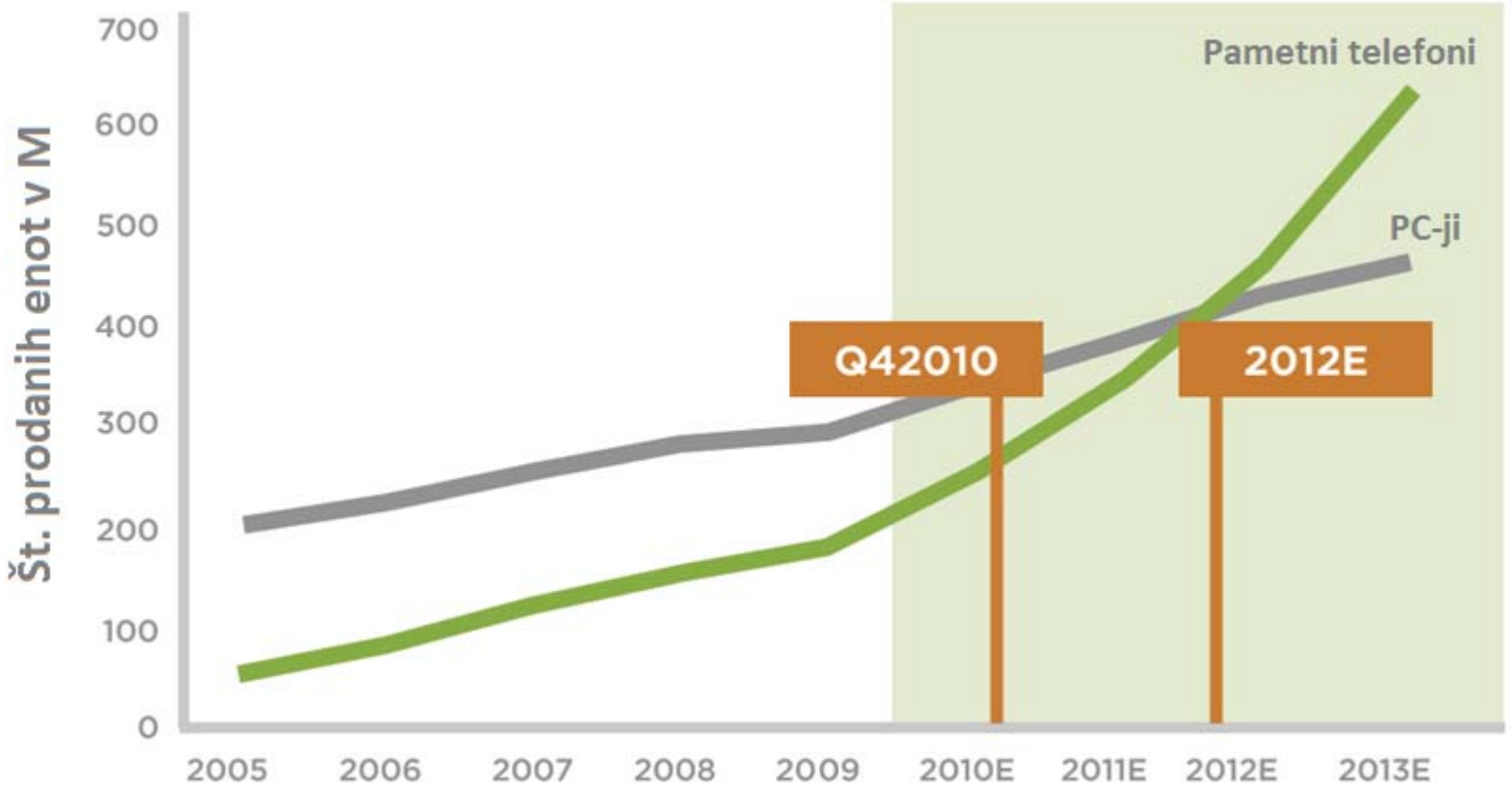
- Napadi na točno določeno osebo ali sistem
  - Prikriti programčki z zelo omejeno funkcionalnostjo, ki tečejo v ozadju
    - Se sami odstranijo po določenem času
    - Kompresija, enkripcija, randomizacija
  - Zbiranje točno določenih informacij ali datotek določenega tipa
  - “Hrupni” malware
    - Veliko hrupa za prikritje dejanske akcije v ozadju
    - Sočasna uporaba velikega števila gostiteljevih računalnikov

18. Mobilni splet

Splet na mobilnih napravah

Mobilni splet

- Pametni telefoni vs. namizni računalniki



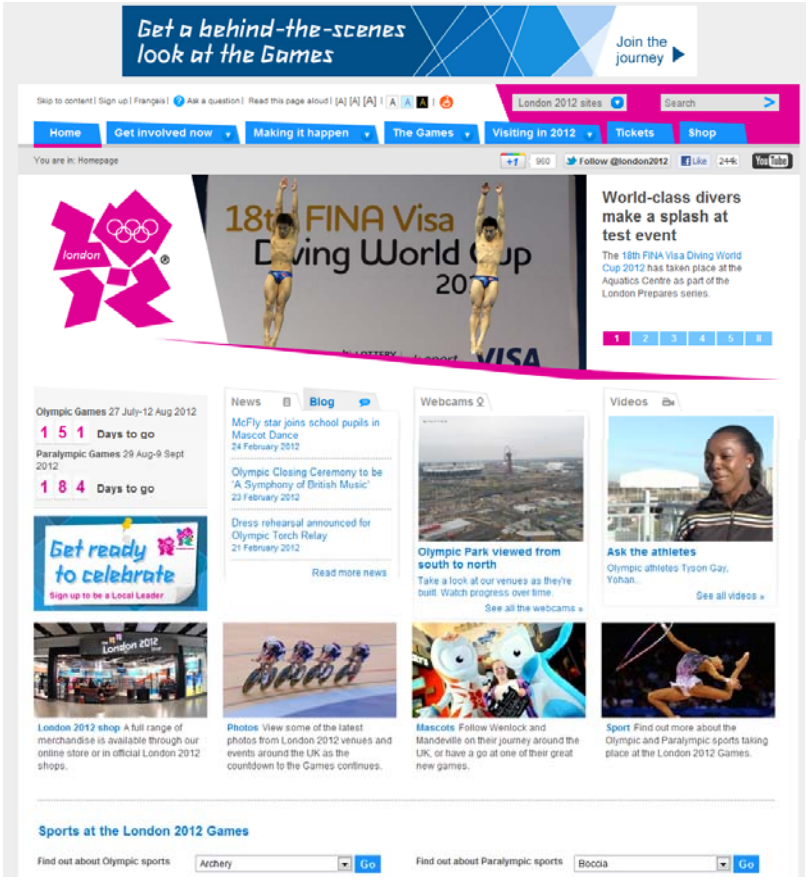
[Wroblewski, L.: *Mobile First, 2011*]

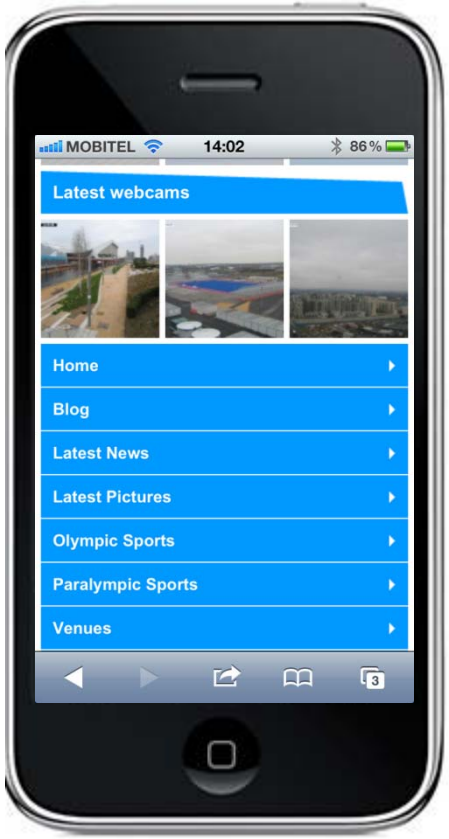
Mobilni splet

- Rast
  - Facebook: 43% aktivnih uporabnikov je mobilnih
  - Twitter: 40% twitov je poslanih iz mobilnih telefonov
  - Mixi: 85% vseh dostopov je preko mobilnih naprav
  - Pandora: 50% novih registracij je preko mobilnih naprav
  - Email: 70% vseh dostopov je mobilnih
  - Google: 130% povečanje iskanja preko mobilnih telefonov (2010)
  - Amazon: več kot milijardo USD nakupov preko mobilnih telefonov
  - Ebay: Najdražji mobilni nakup: Porsche Carrera GT ($370.000)
- Uporaba mobilnega spleta
  - 600% rast prometa do mobilnih spletnih strani
  - Povprečni pametni telefon obišče v povprečju do 24 strani na dan
  - 50 najbolj obiskanih strani ima 40% mobilnih obiskov

Omejitve mobilnih naprav

- Velikost zaslona in ločljivost
  - 320 x 480 pik (20% ločljivosti namiznih rač.)

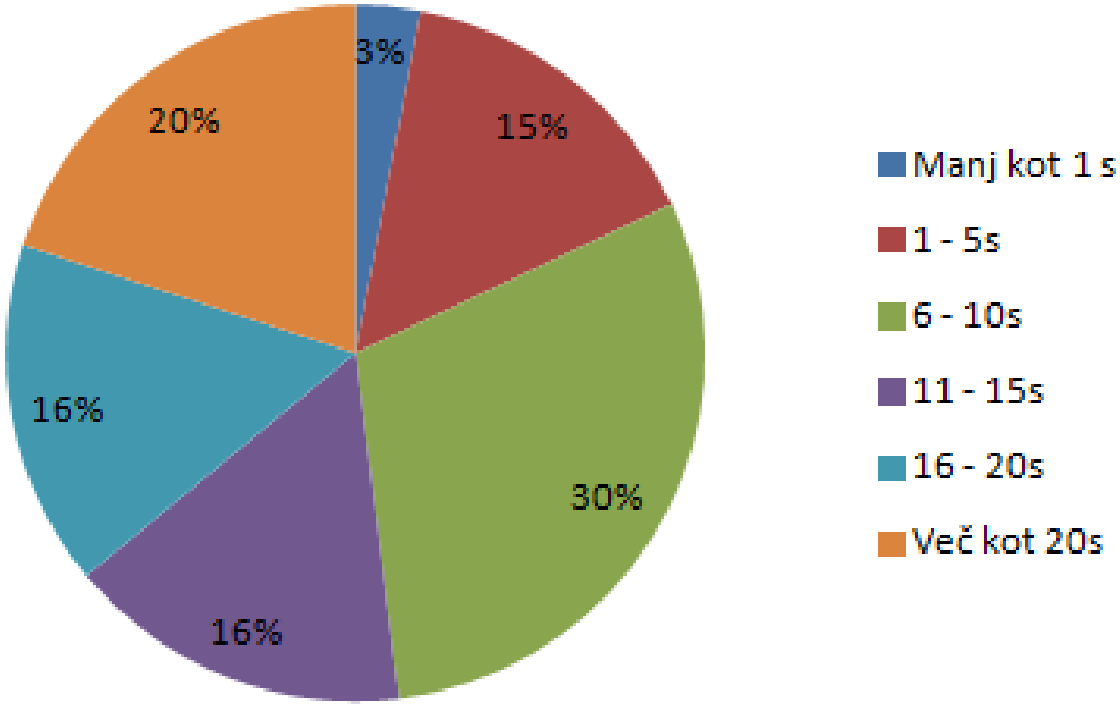




[http://www.london2012.com/]

Omejitve mobilnih naprav

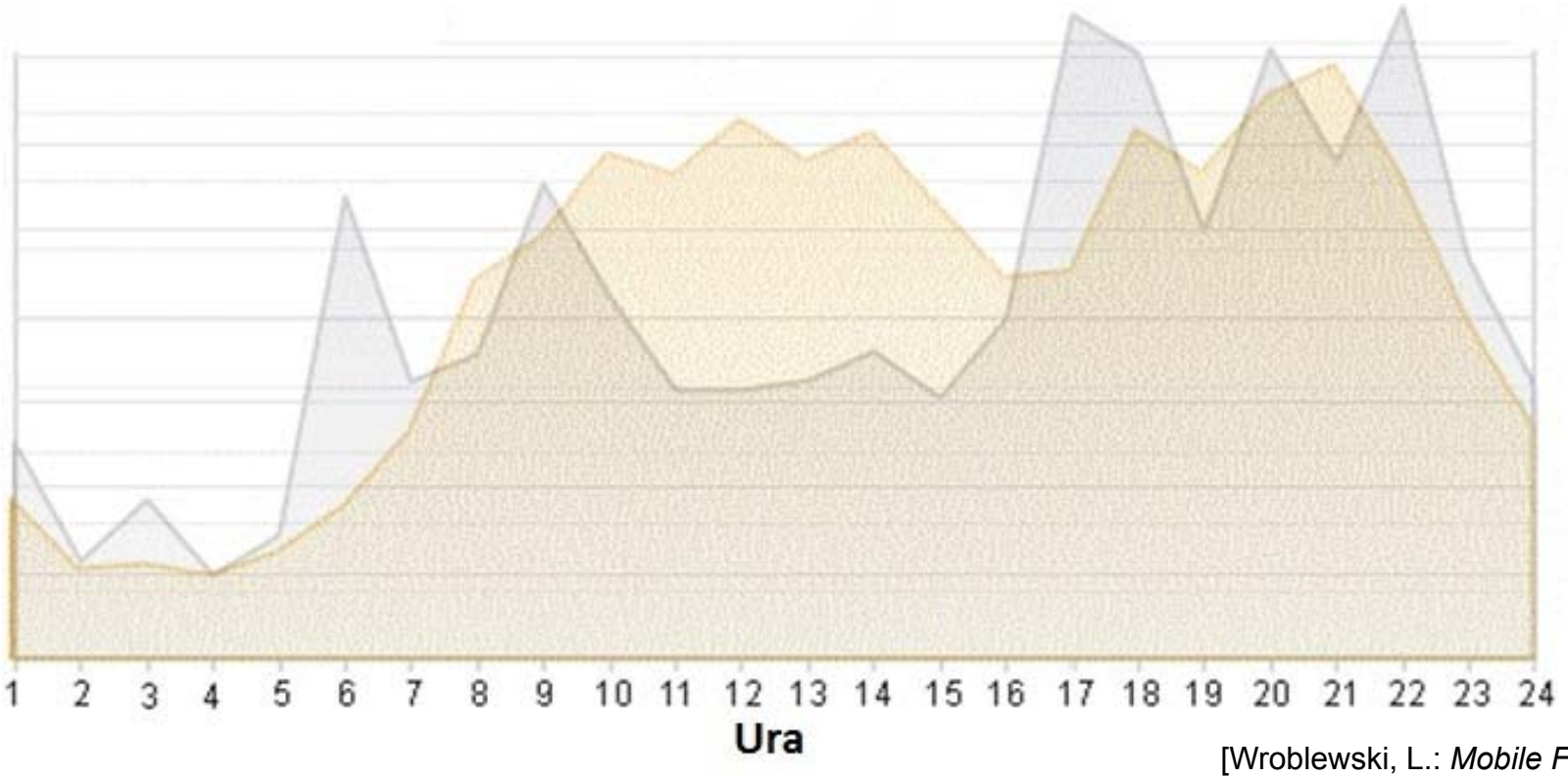
- Omrežna hitrost
  - Čas nalaganja spletne strani in potrpljenje uporabnikov



[http://blog.kissmetrics.com/loading-time/]

Omejitve mobilnih naprav

- Načini in vzorci uporabe
  - Uporaba namiznih rač. in pametnih telefonov
  - 84% (doma), 74% čakanje v vrstah), 64% (služba), itd.
  - “one eyeball, one thumb” (eno oko, en prst)



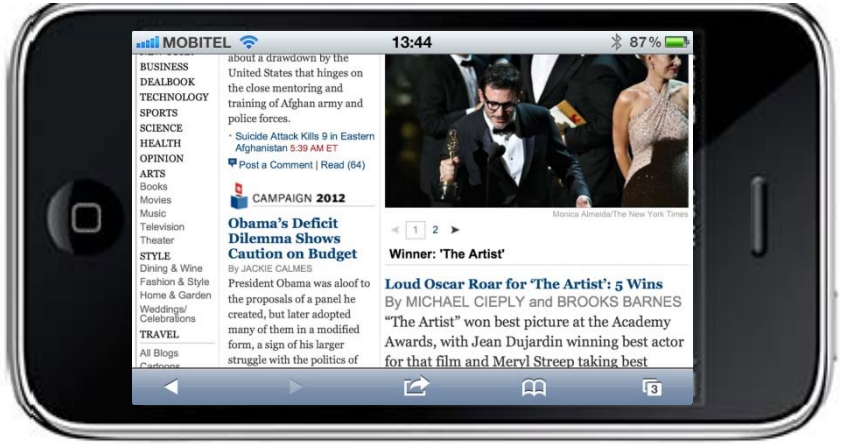
[Wroblewski, L.: *Mobile First, 2011*]

Zmogljivosti mobilnih naprav

- Slika in video
- Avdio
- Več tehnologij za prenos podatkov
- Različne orientacije
  - “Portrait” in “landscape”
- Informacija o lokaciji
  - Lokacija, gibanje, orientacija
- Zaslon na dotik
  - Direktna manipulacija
  - Geste
- Senzor bližine
- NFC

Zmogljivosti mobilnih naprav

- Različne orientacije



Zmogljivosti mobilnih naprav

- Informacija o lokaciji

Tehnologija	Natančnost	Čas določanja	Poraba energije
GPS	10 m	2 -10 minut	Zmogljivost baterije: 5-6 ur
WiFi	50m (odvisno od gostote)	Takoj	Ni učinka
Bazne postaje (triangulacija)	100 – 1400 m (odvisno od pokritosti)	Takoj	Ni učinka
Ena sama bazna postaja	500 – 2500 m (odvisno od pokritosti)	Takoj	Ni učinka
IP naslov	Država: 99% Mesto: 45 % Kraj: ~ 0%	Takoj	Ni učinka

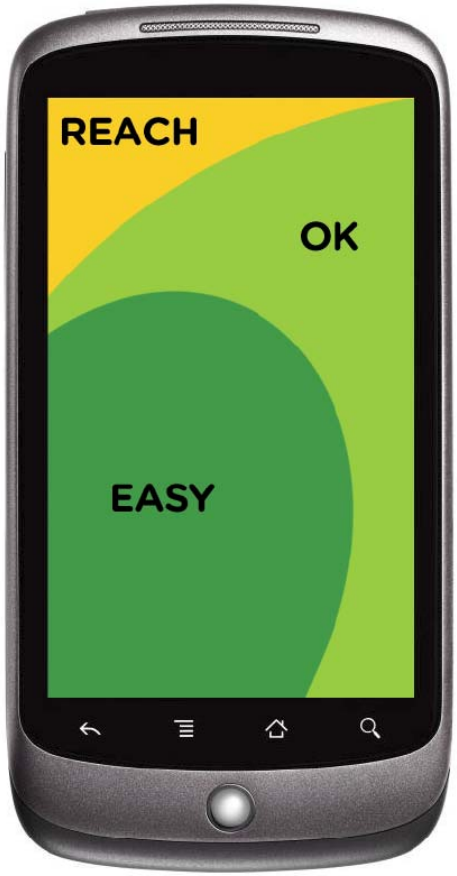
Zmogljivosti mobilnih naprav

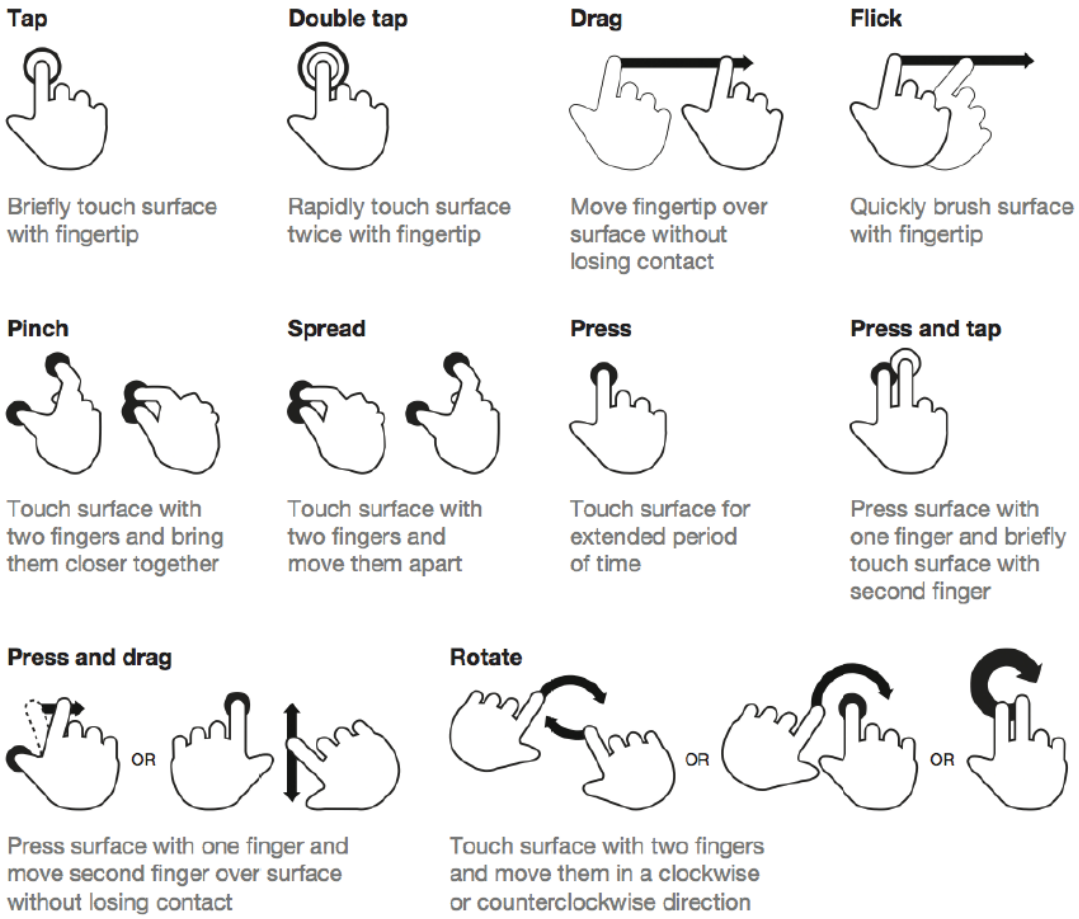
- Informacija o lokaciji
  - Lokacija, gibanje, orientacija

[GOMO News]

[Christos Gatzidis' scientific diary]

- Zaslon na dotik





[Wroblewski, L.: *Mobile First, 2011*]

Mobilne aplikacije

- “Native vs. Web“
  - Veliko število platform oz. operacijskih sistemov
  - Stroški razvoja
  - Dostop do zmogljivosti terminala
  - Hitrost izvajanja programske kode
  - Distribucija
    - Evaluacija programske kode (odobritev)
  - Hibridi?!
    - Sencha, PhoneGap

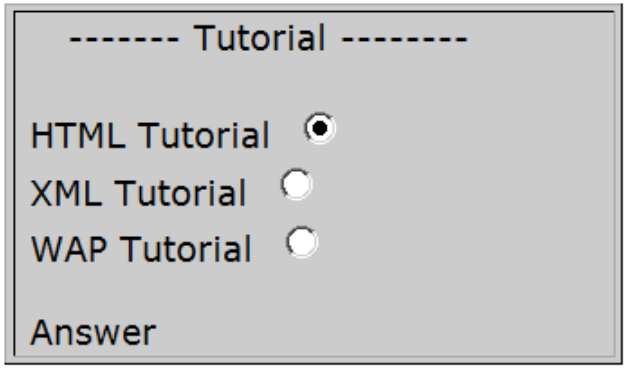
Standardi

- Mobilni splet
  - WAP 1.0 (Wireless Application Protocol) – 1998
    - WTP in WTLS transportna protokola
    - WML (Wireless Markup Language) – namesto HTML
  - WAP 2.0 – 2002
    - HTTP, TCP/IP
    - XHTML-MP (XML - Mobile Profile)
      - OMA (Open Mobile Alliance)
    - XHTML-Basic (1.1)
      - W3C
    - Wireless CSS (OMA)
    - CSS MP (W3C)
    - Kombinacija XHTML MP / WCSS (delovanje na vseh platformah)

Primeri

WML

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
  <card id="card1" title="Tutorial">
    <do type="accept" label="Answer">
      <go href="#card2"/>
    </do>
    <p><select name="name">
      <option value="HTML">HTML Tutorial</option>
      <option value="XML">XML Tutorial</option>
      <option value="WAP">WAP Tutorial</option>
    </select></p>
  </card>
  <card id="card2" title="Answer">
    <p>You selected: $(name)</p>
  </card>
</wml>
```



Standardi

- Mobile Web Initiative (MWI) – 2005
  - .mobi (TLD – Top Level Domain)
    - Dve vzporedni različici spleta
  - Mobile Web Best Practices (2010) – W3C
    - Dobre prakse pri ustvarjanju mobilnih spletnih strani za izboljšanje uporabniške izkušnje
    - Prilagajanje različnim odjemalcem
    - Orodja za preverjanje ustreznosti
      - mobileOK: http://validator.w3.org/mobile/
      - mobiReady: http://ready.mobi/launch.jsp?locale=en_EN

        (ni del W3C)

Standardi

- Dva standarda, ki “tekmujeta” za prevlado na tem področju sta XHTML 2.0 in HTML 5
  - XHTML 2.0 (2002)
    - W3C
    - Temelji na XML sintaksi
      - Strogo ločevanje strukture in vsebine
  - HTML5
    - WHATWG (Web Hypertext Application Technology Working Group)
    - Del W3C (od 2007 dalje)
    - Boljša podpora s strani brskalnikov
    - Temelji na HTML4
      - Nadgrajuje pomankljivosti
      - Lahko tudi na osnovi XML

HTML5

- Novi elementi
  - Struktura strani
    - <nav>, <footer>, <header>, <section>, <aside>
  - Multimedija
    - <video>, <avdio>, <figure>, <canvas>, SVG (Scalable Vector Graphics), itd.
  - Novi atributi
    - autofocus, autocomplete, placeholder, required, itd.
  - Dodatne zmožnosti brskalnikov
    - Javascript API-ji
    - Upravljanje z zgodovino
    - Dostop do različnih zmogljivosti telefona (kontakti, SMS, MMS, e-pošta)

HTML5

- Javascript aplikacijski programski vmesniki
  - Lokalna podatkovna baza
    - Web Storage (sessionStorage, localStorage)
    - WebSQL (temelji na SqlLite)
    - IndexedDB (shranjevanje JS objektov)
  - Dvosmerna komunikacija strežnik/odjemalec
    - WebSockets (nov protokol ws oz. wss)
    - Server Sent Events (preko HTTP)
  - Geolocation API
  - Web Workers

Organizacija mobilne strani

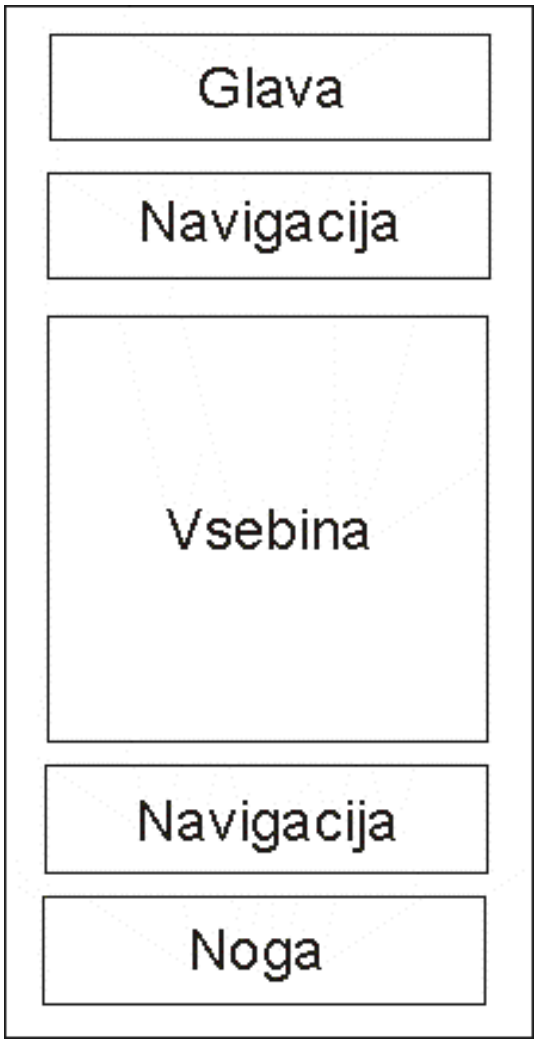
- 4 tipi interakcije mobilnih uporabnikov
  - Iskanje nujne, zelo specifične informacije
    - pogosto lokacijsko odvisne
  - Raziskovanje in igranje
    - Čakanje, dolgočasje (zapravljanje časa)
  - Ažuriranje statusa (check-in)
    - Spremljanje neke aktivnosti, ki teče vzporedno
  - Pregledovanje ali spreminjanje dokumenta (pošte)
    - Nujno zadeva, ki ne more počakati

Organizacija mobilne strani

- Vsebina je pomembnejša od navigacije

Organizacija mobilne strani

- Vertikalna postavitev strani
  - Drsniki le za vertikalno dimenzijo
- Navigacija
  - Do največ 5 nivojev globine
  - Največ 10 povezav na eni strani
    - Razporeditev povezav po prioritetah
- Vračanje (“Back button“)
- Virtualni gumbi na dnu
  - Enostavni za pritisk
  - Zamenjava s fizičnimi gumbi



Organizacija mobilne strani

- Zaslon na dotik
  - Različne velikost prstov in spretnosti uporabnikov
  - “Pretiravanje“ z velikostjo
    - Apple: 44px x 44px
    - Microsoft: 9mm (min 2mm praznega prostora)
    - Vizualni prikaz gumba zaseda 50%-100% občutljive površine
    - Velikost je odvisna od pogostosti uporabe in položaja na zaslonu
    - Geste
      - “Tap“, “drag“ in “swipe“
    - Hover dogodki
    - Tranzicija od WIMP k NUI vmesnikom

Organizacija mobilne strani

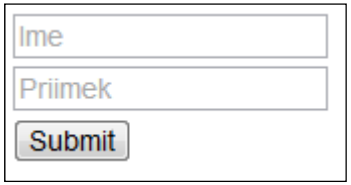
- Vnos: okrnjen in omejen
  - Milijarde SMS sporočil, ki se pošljejo vsak dan
- Povezave s telefonskimi številkami
  - Direktna vzpostavitev klica iz bližnjice

```
<a href="tel:+38641555666">+386 41 555 666</a>
```

- Obrazci
  - Le najnujnejše informacije
  - Kratka vnosna polja, po možnosti izbirna
  - Uporaba vnosnih mask (formatirani vnosi)
    - HTML5 atributi za zančko <input>
      - color, date, email, file, number, pattern, text, url, itd.

```
<input type="text" name="koda_drzave" pattern="[A-Za-z]{3}"
```

```
<input type="text" name="ime" placeholder="Ime" />
<input type="text" name="priimek" placeholder="Priimek" />
```



Organizacija mobilne strani

- Vnosna polja

<input type="text" …

<input type="url" …

<input type="number" …

Organizacija mobilne strani

- Elementi, ki se ne uporabljajo
  - Tabele
  - Zavihki (“tab-i”)
  - Okvirji (“frame-i”)
  - Pojavna okna (“pop-up-i”)
  - Vgrajeni objekti (“embeded”)
  - Skripte
  - Preusmeritve
- Minimalno število zunanjih virov!!
  - CSS datoteke, slike, itd.
  - CSS3

Osnovne značilnosti mobilne spletne aplikacije

- Optimizacija hitrosti nalaganja strani
  - Uporaba kaskadnih stilov CSS
  - Združevanje CSS in JS datotek
  - Uporaba kompresijskih algoritmov (Gzip)
  - Uporaba CSS3 tehnik
    - Okrogli robovi, barvni prehodi, sence, itd.
  - Slike
    - Najmanjša možna dimenzija (širina pod 120px)
      - Majhna ločljivost
      - Fiksna določitev dimenzij (pomoč brskalniku za hitrejši izris)
    - Časovno pogojeno nalaganje slik (le dejansko vidnih)
    - Nadomestno besedilo (lastnost “alt”)
  - Lokalno shranjevanje (caching)
  - Orodja
    - PageSpeed, yslow (Firefox), Blaze Mobile

Prilagodljiva spletna stran

- “Responsive web” – prilagodljiva spletna stran
  - “Tekoča” vsebina, ki se sama prilagaja velikosti terminala (fluid layout)
    - Relativne dimenzije: %, smaller, larger, em, itd.
    - Uporaba ogrodij - “gridov”
    - Prilagodljive slike: `img {max-width:100%}`
    - Prilagajanje glede na ločljivost in orientacijo terminala
    - “Media queries”

Prilagodljiva spletna stran

- CSS3 media queries
  - Problemi hitrih sprememb velikosti in ločljivosti zaslona
  - Različni stili in pogledi glede na vrsto brskalnika in velikost terminala
  - Primer:

```
@media screen and (max-width:320px)
{
/* Stil za orientacijo portrait (za pametne telefone, npr. iPhone in Android)
…
}

@media only screen and (min-width:321px) and (max-width:480px)
{
/* Stil le za landscape orientacijo (za pametne telefone, npr. iPhone in Android)
…
}
```

Prilagodljiva spletna stran

- CSS3 media queries

```
@media only screen and (min-device-width:320px)
   and (maxdevice-width:569px)
{
#sidenav ul {display:none;}
#sidenav form {display:block;}
}
```

Vrsta naprave	Največja širina zaslona
Veliki namizni zasloni	> 1280px
Prenosniki	570px – 1280px
Tablice	800px – 1280px (odvisno od orientacije)
Pametni telefoni	Manj kot 570px

PRIMER: http://www.smashingmagazine.com/ /responsive

Ločena mobilna stran

- Ločena mobilna spletna stran
  - Fiksne dimenzije za različne odjemalce
  - Omejuje uporabniško izkušnjo
  - Dvojni stroški razvoja in vzdrževanja
  - Branje “User-agent” polja v glavi HTTP zahteve

Primer: http://www.lkn.fe.uni-lj.si

GET / HTTP/1.1
Host: www.lkn.fe.uni-lj.si
User-Agent: Mozilla/5.0….
Accept: text/html,application/xhtml+xml….
Accept-Language: en-us,en;…
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1….
Keep-Alive: 300
Connection: keep-alive

Detekcija terminala na strežniku

- Prilagajanje vsebine na strežniku
  - Primeri agentov:

IE7: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0)

FF 3.0: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.0.1) Gecko/2008070208 Firefox/3.0.1

Chrome: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.6 (KHTML, like Gecko) Chrome/20.0.1092.0 Safari/536.6

iPhone: Mozilla/5.0 (iPhone; U; CPU like Mac OS X; en) AppleWebKit/420+ (KHTML, like Gecko) Version/3.0 Mobile/1A543a Safari/419.3

Android (HTC): Mozilla/5.0 (Linux; U; Android 2.1-update1; de-de; HTC Desire 1.19.161.5 Build/ERE27) AppleWebKit/530.17 (KHTML, like Gecko) Version/4.0 Mobile Safari/530.17

Detekcija terminala na strežniku

- Branje informacije o agentu

  Primer ASP: Request.ServerVariables("http_user_agent")

  Primer PHP: $_SERVER[“HTTP_USER_AGENT“].

- Na večini naprav se agenta da nastaviti
  - Dodatek brskalniku

Prilagajanje zmogljivostim različnih terminalov

- Detekcija zmogljivosti in lastnosti terminala
  - Podatkovna baza: Device Description Repository
  - Nestandardni dodatek v HTTP glavi
    - UAPROF: url z informacijami o lastnostih terminala

```
<?xml version="1.0"?> <rdf:RDF xmlns:rdf="http://.
…
<prf:Vendor>High Tech Computer Corporation</prf:Vendor>
<prf:BitsPerPixel>16</prf:BitsPerPixel> <prf:ColorCapable>Yes</prf:ColorCapable>
<prf:ScreenSize>320x240</prf:ScreenSize>
<prf:ImageCapable>Yes</prf:ImageCapable>
<prf:PixelAspectRatio>1x1</prf:PixelAspectRatio>
<prf:ScreenSizeChar>10x25</prf:ScreenSizeChar>
<prf:StandardFontProportional>Yes</prf:StandardFontProportional>
<prf:SoundOutputCapable>Yes</prf:SoundOutputCapable>
<prf:TextInputCapable>Yes</prf:TextInputCapable>
<prf:VoiceInputCapable>Yes</prf:VoiceInputCapable> <prf:InputCharSet>
…
```

Razvoj in testiranje mobilnih spletnih strani

- Namizni računalnik
  - Prilagojena velikost oken
    - Uporaba okna (“iframe”)

      <iframe scr=“mobilna_stran/index.html” width=“240” height=“320”> </iframe>
  - Brskalnik
    - Opera: “Small screen view”
    - FireFox: “User Agent Switcher”
    - Chrome: “UA Spoofer”
  - Emulator

    http://www.opera.com/mobile/demo/
- Mobilne naprave
- Orodja (jQuery Mobile)