HTML5 i JavaScript za video igre
CSS, DOM

Predavač:
doc. dr Jelena Graovac

Stilski listovi

- Vizuelna prezentacija HTML dokumenata podešava se korišćenjem stilskih listova (*stylesheets*) opisanih u jeziku CSS (*Cascading Style Sheets*)
- Prva verzija objavljena 1996. godine (CSS 1)
- Današnje konture dobija sa verzijom CSS 2 (CSS 2.1)
- Aktuelna verzija CSS 3
- U razvoju (od kraja 2017. godine): CSS 4
- Verzija CSS 4 još uvek nije potpuno podržana od strane veb pregledača

Opšta sintaksa stilskih listova

- Mogu se zadavati u zaglavlju HTML dokumenata, u okviru elementa `style` ili u posebnim CSS dokumentima
- Stilski list se sastoji od niza pravila

```
p { color: red; }
h1 { font-family: Arial; margin: 20px; }
```

- Beline nemaju uticaja; stilski list se često nazubljuje radi preglednosti

```
h1 {
  font-family: Arial;
  margin: 20px;
}
```

- Svako pravilo je oblika:
  selektor {niz deklaracija međusobno razdvojenih znakom ';' }
- Svaka deklaracija je oblika:
  svojstvo: vrednost

Opšta sintaksa stilskih listova

- Selektori mogu biti složeniji nego samo navođenje imena elemenata
- Više selektora može se navesti zajedno (razdvajaju se zapetama)

```
h1, h2, h3 { color: blue; }
```

- Komentari se navode između simbola /* i */

```
p { /* podešavamo sve pasuse */
  color: red; /* crvena boja teksta */
  margin: 10px; /* margina od 10 piksela */
}
```

Uključivanje stilskih listova u HTML dokumente

- Tri načina za uključivanje CSS opisa u HTML dokument
- *Opisi na nivou elementa* (atribut `style`)
  - pojedinačnom elementu može se promeniti stil navođenjem atributa `style` čija je vrednost niz CSS svojstava i njihovih vrednosti
  - opis vizuelne prezentacije je isprepleten sa opisom njene strukture

```
<p style="color:red; margin-left:10px;">Ovo je pasus</p>
```

- *Opisi na nivou dokumenta* (element `style`)
  - CSS opis se može navesti u zaglavlju dokumenta, kao sadržaj elementa `style`

```
<head>
  ...
  <style type="text/css">
    p { color : blue; }
  </style>
  ...
</head>
```

Uključivanje stilskih listova u HTML dokumente

- *Spoljašnji opisi*
  - Koriste se za stilizaciju većeg broja veb-strana na isti način
  - Zapisuju se u vidu tekstualne datoteke sa ekstenzijom .css
  - Pojednostavljuje izmenu vizuelne prezentacije celog veb-sajta
  - Uključuje se korišćenjem elementa link u zaglavlju dokumenta, navođenjem atributa `rel` sa vrednošću `stylesheet`

    ```
    <link rel="stylesheet" type="text/css" href="stil.css" />
    ```
  - Jezik CSS dozvoljava uvoz nekog drugog stilskog lista u dati stilski list, korišćenjem direktive @import

    ```
    <style type="text/css"> @import url("stil.css"); </style>
    ```

Nasleđivanje stilskih listova

- Za neka svojstva važi da ako elementu pridružimo neku deklaraciju stila, nju automatski nasleđuju svi elementi sadržani u tom elementu

```
body { color : red; }
```

- Nasleđeno svojstvo se može promeniti

```
p { color : blue; }
```

- Neka svojstva se ne nasleđuju

```
body { margin : 20px; }
```

Selektori

- Najjednostavniji selektor je naziv elementa

```
p { color : blue; }
```

- Kao selektor se može koristiti jedinstveni identifikator elementa: elementu pridružujemo identifikator korišćenjem atributa id, a zatim ga koristimo kao selektor oblika #id

```
<p id="opis">U ovom pasusu biće opisana glavna svojstva...</p>
```

```
p#opis { color : blue; }
```

isto je što i:

```
#opis { color : blue; }
```

Selektori

- Kao selektor može se koristiti i naziv klase: svakom elementu koji želimo na isti način da stilizujemo dodeljuje se klasa korišćenjem atributa class, a zatim se ta klasa koristi kao selektor oblika .class

```
<p class="rezime">Jezik HTML je...</p>
...
<p class="rezime">Jezik CSS je...</p>
```

```
p.rezime { color : blue; border: 1px solid black; }
```

nije nužno isto što i:

```
.rezime { color : blue; border: 1px solid black; }
```

- Opštije: `E[atribut="vrednost"]` – svi elementi *E* koji imaju dati atribut postavljen na datu vrednost
- Slično: `E[atribut~="vrednost"]` – svi elementi *E* koji imaju dati atribut sa vrednošću koja sadrži datu reč (`E.klasa` je ekvivalentno sa `E[class~="klasa"]`)

Pseudoklase i pseudoelementi

- Pseudoklase i pseudoelementi služe za finija podešavanja u jeziku CSS
- Unapred su definisani i nipošto ih ne treba dodeljivati elementima u sklopu HTML opisa
- Pseudoklase služe za stilizovanje elemenata dok su u posebnom stanju, dok pseudoelementi služe za stilizaciju dela nekog elementa
- Pseudoklase se označavaju pomoću dvotačke (:hover), a pseudoelementi pomoću dvostruke dvotačke (::first-line)
- Najčešće pseudoklase i pseudoelementi:
  - :link, :active, :visited
  - :hover
  - :checked
  - :first-child
  - :nth-child(An+B), :nth-child(even), :nth-child(odd)
  - ::first-line
  - ::first-letter
  - ::before, ::after

```
:hover { color : green;  }
```

Ugnežđeni elementi

- Ugnežđeni elementi se koriste kada je potrebno promeniti stil samo onih elemenata koji se nalaze u nekom širem elementu
- Sintaksa: `selector1 selector2` – ovim se označavaju elementi opisani selektorom `selector2` koji se nalaze u okviru selektora `selector1`
- Sintaksa: `selektor1 > selektor2` – ovim se označavaju elementi opisani selektorom `selector2` koji su neposredni potomci selektora `selector1`
- Sintaksa: `selektor1 + selektor2` – ovim se označavaju elementi opisani selektorom `selektor2` čiji je prethodni brat u HTML stablu opisan selektorom `selektor1`
- Sintaksa `selektor1 ~ selektor2` – ovim se označavaju elementi opisani selektorom `selektor2` čiji je neki od prethodne braće u HTML stablu opisan selektorom `selektor1`

```
main p { color: blue; }
div#gallery > img.small { width: 50px; }
```

Font

- Naziv (familija) fonta se zadaje svojstvom font-family; može se navesti:
  - tačan naziv fonta: imena fontova od više reči navode se pod navodnicima (npr. "Times New Roman")
  - ime familije fontova (npr. Times)
  - ime vrste fontova (npr. serif)
- Razlikujemo:
  - serifne fontove (*serif*) koji na ivicama znakova imaju neke detalje
  - neserifne fontove (*sans-serif*) koji to nemaju
  - neproporcionalne fontove (*monospace*) kod kojih su sva slova iste širine
- Moguće je navesti više opisa u opadajućem prioritetu

```
p { font-family: "New Century Schoolbook", Times, serif }
```

Font

- Veličina fonta se zadaje svojstvom font-size, na dva načina:
  - apsolutno: izraženo u pikselima (px) ili u ,,tačkama" (pt). Veličina piksela zavisi od rezolucije. 1pt=1/72 inča
  - relativno: izraženo u procentima ili em-ovima (1em=100%), u odnosu na veličinu fonta nasledjenu od roditeljskog elementa

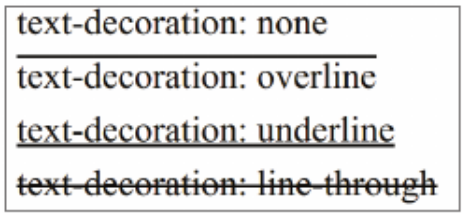
```
p { font-size: 12pt; }
p { font-size: 120%; }
p { font-size: 1.2em; }
```

- Varijante fonta
  - font-style određuje iskošenost karaktera; moguće vrednosti: `normal`, `italic`, `oblique`
  - font-weight određuje debljinu slova; najčešće vrednosti: `normal`, `bold`, `lighter`
- Dozvoljeno je više karakteristika fonta zadati jednom deklaracijom

```
p { font: italic bold 12pt Times, serif; }
```

Stilizovanje teksta – ukrašavanje teksta

- U CSS-u se (za razliku od procesora teksta) podešavanje teksta razlikuje od podešavanja fonta
- U podešavanje teksta spadaju: podvlačenje teksta, precrtavanje, uvlačenje prve linije, poravnanje teksta, podešavanje razmaka između reči i slova,...
- U CSS-u se svojstvima fonta određuje izbor glifa za ispis karaktera, a svojstvima teksta kako se glifovi raspoređuju i da li je potrebno još nešto docrtati



- Dodatno ukrašavanje teksta zadaje se svojstvom text-decoration; moguće vrednosti su `none`, `underline`, `overline`, `line-through`

```
a { text-decoration: none; }
a:hover { text-decoration: underline; }
```

Stilizovanje teksta – razmak između karaktera

- U CSS-u moguće je fino podešavanje horizontalnog razmaka između karaktera
- Za podešavanje razmaka između karaktera u jednoj reči koristi se svojstvo letter-spacing, a za podešavanje razmaka između susednih reči svojstvo word-spacing; zadaje se u pt, px ili em

l e t t e r - s p a c i n g : 5 p x
word-spacing: 25px

- Vertikalni razmak nazivamo prored i on se može podešavati korišćenjem svojstva line-height; zadaje se kao decimalni broj ili kao procenat

```
p { line-height: 1.5em; }
```

U ovom primeru se 'line-height: 200%' koristi da bi se povećao prored u pasusu.

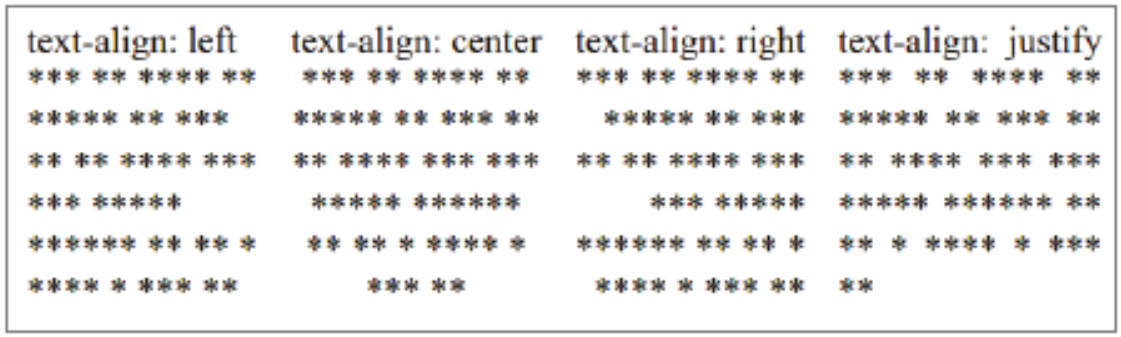
Stilizovanje teksta – uvlačenje prve linije teksta, poravnanje

- Uvlačenje prve linije teksta zadaje se svojstvom text-indent; vrednost se zadaje u px, pt ili u procentima, odnosno u em

```
p { text-indent: 3em; }
```

U ovom primeru je postavljeno 'text-indent: 3em' tako da je prva linija pasusa uvučena.

- Poravnanje teksta u okviru elementa se zadaje svojstvom text-align; moguće vrednosti su `left`, `right`, `center`, `justify`



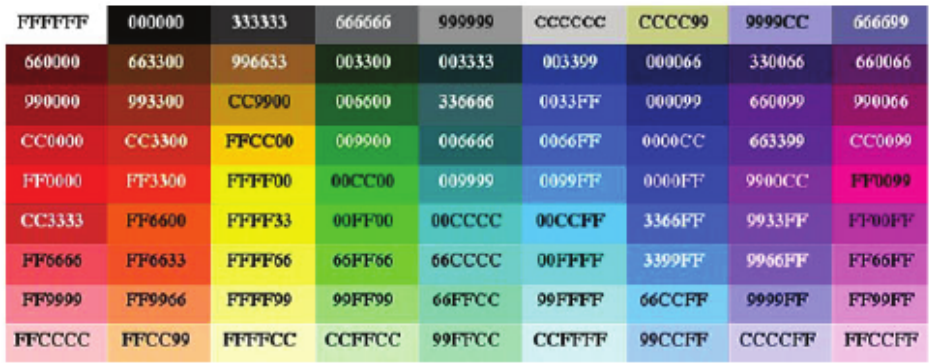
- Poravnanje jednog elementa u odnosu na drugi element koji ga okružuje zadaje se na drugi način

Boje

- Boja teksta zadaje se svojstvom color
- Boje se mogu zadati na više načina:
  - korišćenjem imena

    ```
    p { color: red; }
    ```
  - heksadekadnim kôdom oblika #rrggbb – tri dvocifrena heksadekadna broja

    ```
    p { color: #00ff00; }
    ```

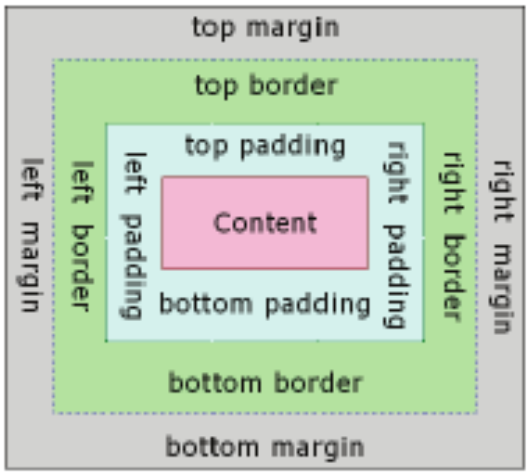


  - heksadekadnim kôdom oblika #rgb (#27c = #2277cc)
  - dekadnom specifikacijom oblika rgb(r,g,b) – tri dekadna broja

    ```
    p { color: rgb(0,0,255); }
    ```

Model kutije

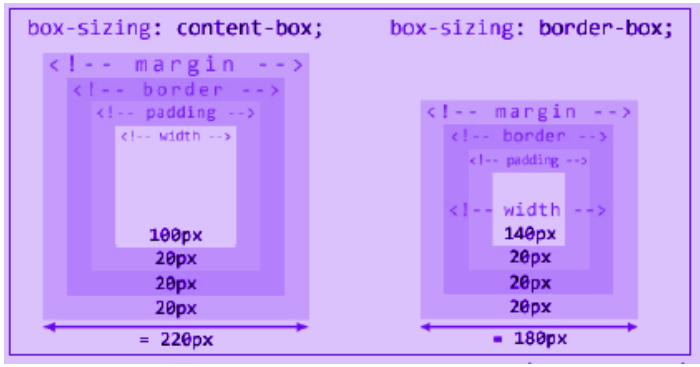
- Svi HTML elementi mogu da se posmatraju kao pravougaone površine – kutije (*box*)
- Svaka kutija ima sadržaj (*content*) i može da ima okvir (*border*)
- Okvir je razdvojen od sadržaja unutrašnjom marginom, tj. punjenjem (*padding*), a od okolnih elemenata spoljašnjom marginom (*margin*)



Модел кутије

Širina i visina

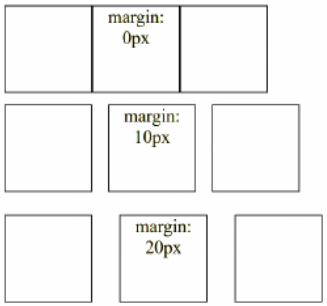
- Širina i visina elementa zadaju se svojstvima width i height; vrednost se zadaje u px
  - Podrazumevano se punjenje, okvir i margine ne računaju u širinu i visinu
  - Svojstvom box-sizing se podešava šta se računa u širinu i visinu elementa: podrazumevana vrednost je content-box i tada se računa samo sadržaj, ako se navede border-box onda se računa visina i širina sadržaja, punjenja i okvira (bez spoljašnjih margina)



- Nekada je zgodno ne fiksirati širinu i visinu već samo zadati najmanje ili najveće dopuštene vrednosti: to se postiže svojstvima max-width, min-width, max-height, min-height

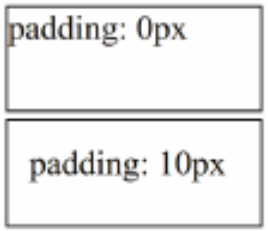
Spoljašnje margine

- Elementima je moguće podesiti spoljašnje i unutrašnje margine
- Spoljašnju marginu je moguće podesiti svojstvom margin:
  - ako se navede samo jedna vrednost, ona se odnosi na sve margine
  - ako se navedu dve vrednosti, prva se odnosi na gornju i donju marginu, a druga na levu i desnu
  - ako se navedu tri vrednosti, prva se odnosi na gornju, druga na levu i desnu, a treća na donju marginu
  - ako se navedu četiri vrednosti, one se odnose na gornju, desnu, donju, levu marginu redom
- Postoje i pojedinačna svojstva: margin-top, margin-right, margin-bottom, margin-left
- ako leva i desna margina imaju vrednost `auto`, margine će se automatski podesiti da budu jednake
- Gornje i donje margine susednih elemenata se ne sabiraju, računa se veća od vrednosti; leve i desne margine susednih elemenata se sabiraju



Unutrašnje margine

- Unutrašnju marginu je moguće podesiti svojstvom padding
- Koristi se analogno svojstvu `margin`
- Postoje i pojedinačna svojstva: padding-top, padding-right, padding-bottom, padding-left

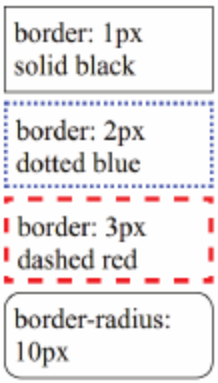


Okviri

- Oko svakog elementa moguće je prikazati okvir
- Debljina okvira podešava se svojstvom border-width
- Tip linije okvira podešava se svojstvom border-style: moguće vrednosti su `solid`, `dashed`, `dotted`,...
- Boja okvira podešava se svojstvom border-color
- Sva tri svojstva moguće je zadati korišćenjem svojstva border

```
p { border: 1px solid black; }
```

- Postoje i svojstva border-top, border-right, border-bottom, border-left
- Zaobljenost okvira se može zadati svojstvom border-radius: vrednost odgovara poluprečniku krugova upisanih u svako teme pravouganika



Pozadina elemenata

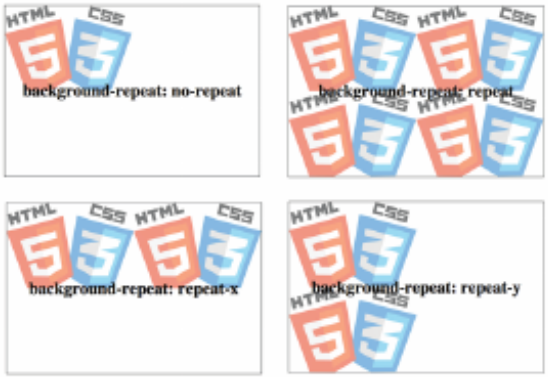
- Svakom elementu moguće je podesiti pozadinu: boju ili sliku
- Boja pozadine elementa može se podesiti svojstvom background-color: vrednost je boja ili `transparent`

background-color: yellow

background-color: #ccccff

background-color: rgb(255, 200, 200)

- Kao pozadina elementa može se postaviti slika svojstvom background-image: vrednost se zadaje u obliku `url(...)`
- Svojstvom background-repeat kontroliše se da li da se slika ponavlja dok ne ispuni širinu/visinu elementa ili ne; moguće vrednosti su `repeat`, `repeat-x`, `repeat-y`, `no-repeat`



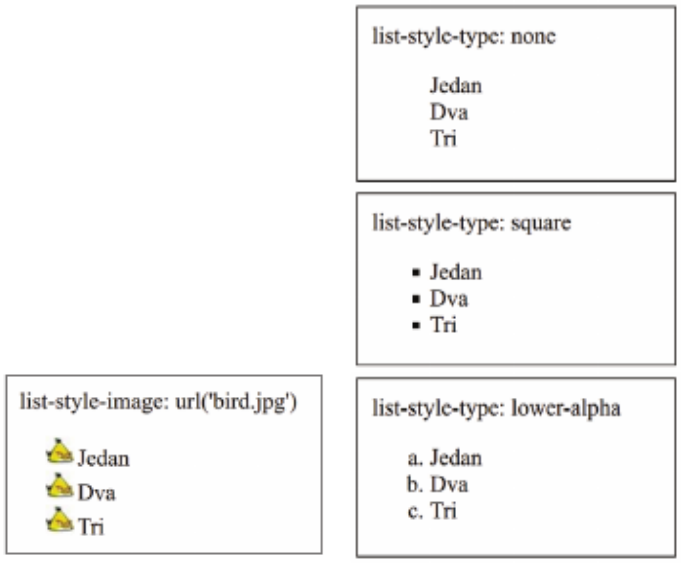
Pozadina elemenata

- Pozicija slike u pozadini elementa može se podesiti svojstvom background-position: moguće je navesti dve vrednosti (horizontalna, vertikalna) ili samo jednu (horizontalna, podrazumevano `center`)
- Vrednosti mogu biti date kao:
  - procenat – x% znači da poravnava tačku koja se nalazi na x% širine (dužine) slike sa tačkom koja se nalazi na x% širine (dužine) elementa
  - dužina – gornje levo teme slike postavlja se na tačku pomerenu za ovu vrednost u odnosu na gornje levo teme elementa
  - `top`, `bottom` – 0% (100%) za vertikalnu poziciju
  - `left`, `right` – 0% (100%) za horizontalnu poziciju
  - `center` – 50% za horizontalnu/vertikalnu poziciju

Stilizovanje lista

- Najčešće se stilizuju oznake stavke liste
  - oblik znaka ispred stavki nabrajanja u nenumerisanoj listi ili broja u numerisanoj listi se podešava svojstvom list-style-type: moguće vrednosti su: `disc`, `circle`, `square`, `none`, `decimal`, `lower-alpha`, `lower-roman`, `upper-alpha`, `upper-roman`, ...
  - kod nenumerisanih slika umesto znaka za nabrajanje može se postaviti slika korišćenjem svojstva list-style-image

```
ul { list-style-image: url("slika.png"); }
```



Stilizovanje tabela

- Stilizacija tabela obuhvata postavljanje dimenzija, okvira, boje pozadine, unutrašnjih i spoljašnjih margina i sl.
- Podešavanja se vrše na nivou tabele, na nivou pojedinačnih redova i pojedinačnih ćelija
- Opšta svojstva `border`, `background-color`, `padding`, `margin`

```
table {
  margin-left: auto;
  margin-right: auto;
}
td {
  background-color: #eeeeee;
  padding: 5px;
  border: 1px solid red;
}
```

Ime	Prezime	Poeni
Petar	Petrović	100
Ana	Anić	95
Mihailo	Mihailović	82
Mara	Marić	73

Stilizovanje tabela

- Postoje i svojstva koja su karakteristična samo za tabele
- Svojstvom border-collapse sa vrednošću `collapse` postavlja se da se susedne ćelije slepe i da imaju jedinstveni okvir

Devojčice	Dečaci
120	125
118	123

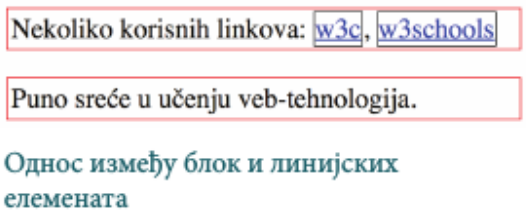
Devojčice	Dečaci
120	125
118	123

- Za poravnanje sadržaja ćelija tabele koriste se svojstva text-align i vertical-align: vrednosti `top`, `middle`, `bottom`

text-align: left	text-align: center	text-align: right
vertical-align: top	vertical-align: middle	vertical-align: bottom

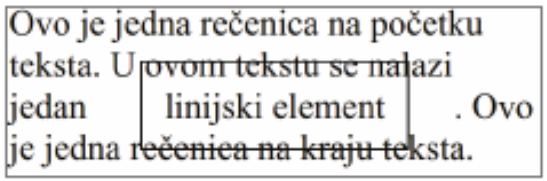
Prikaz

- Razlikujemo dva načina prikaza elemenata:
  - blok elementi
    - `display: block;`
    - mogu da sadrže tekst, linijske elemente i druge blok elemente
    - prostiru se celom širinom bloka, slažu se jedan ispod drugog
    - `div`, `section`, `article`, `header`, `footer`, `main`, `aside`, `form`, `p`, `ul`, `ol`, `li`
  - linijski elementi
    - `display: inline;`
    - mogu da sadrže tekst i linijske elemente
    - zauzimaju koliko i sadržaj, slažu se jedan pored drugog
    - `span`, `a`, `img`, `em`, `strong`, `small`, `i`, `b`, `u`, `sub`, `sup`



Prikaz

- Svojstvo display može imati različite vrednosti:
  - none – element se u potpunosti izostavlja iz prikaza (ne zauzima nikakav prostor na strani)
  - block – element se prikazuje kao blok element; mogu mu se postavljati širina, visina, okvir i margine
  - inline – element se prikazuje kao linijski element; može se podešavati okvir, margine, ali ne i visina i širina (osim kod eksternih objekata, poput slika, kod kojih može, iako su inline elementi); ima smisla podešavati samo levu i desnu marginu jer samo one pomeraju okolni sadržaj

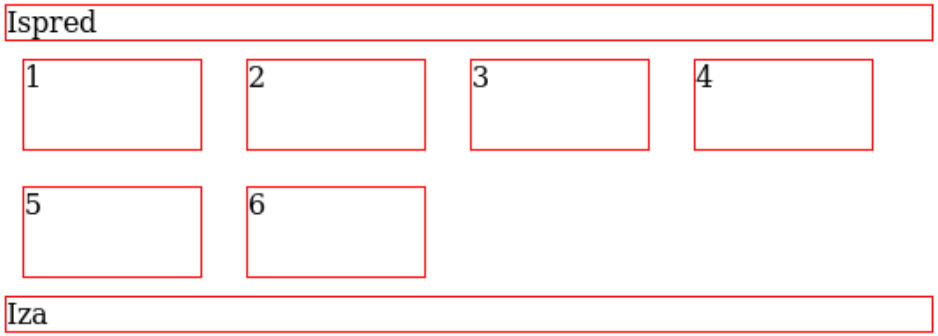


Маргине линијског елемента

Prikaz

- inline-block – element se prikazuje kao linijski blok element; ne prostire se celom širinom, ali mu se mogu podešavati i širina i visina i margine

```
div {
  border: 1px solid red;
}
div.inblock {
  display: inline-block;
  width: 100px;
  height: 50px;
  margin: 10px;
}
<div>Ispred</div>
<div class="inblock">1</div>
<div class="inblock">2</div>
<div class="inblock">3</div>
<div class="inblock">4</div>
<div class="inblock">5</div>
<div class="inblock">6</div>
<div>Iza</div>
```



Uvod

- Objektni model dokumenta (*Document Object Model, DOM*) je interfejs koji omogućava skriptovima da dinamički pristupe i izmene sadržaj, strukturu ili stil veb dokumenta
- Elementi dokumenta predstavljaju se objektima koji imaju svoja svojstva i metode; promenom vrednosti svojstava i pozivima metoda menja se veb dokument
- DOM je nezavisan od jezika iz kojeg se koristi i od platforme na kojoj se koristi

Istorijat DOM-a

- Nastao je u vreme ratova pregledača
- 1996. Netscape u okviru njihovog pregledača ugrađuje podršku za jezik JavaScript; Microsoft u IE 3.0 ugrađuje podršku za jezik JScript
- Definiše "DOM Level 0" odnosno "Legacy DOM" koji omogućava pristup samo nekim elementima HTML dokumenta
- 1997. sa novijim verzijama pregledača javlja se bolja podrška za dinamički HTML i DOM se proširuje; svaka kompanija vrši nezavisno proširenje i ove verzije su poznate pod nazivom "Intermediate DOM"
- Krajem 1990-tih pod okriljem W3C započinje standardizacija klijentskih skript jezika i DOM-a; razvijen standard ECMAScript i standardna verzija DOM-a poznata kao DOM Level 1 kojom se definiše kompletan model za HTML i XML dokumente
- 2000. godine DOM Level 2, 2004. godine DOM Level 3

Raspoređivačke mašine

- DOM predstavlja sliku HTML/XML dokumenta u obliku stabla čiji su čvorovi DOM objekti
- Svaki pregledač mora da sadrži raščlanjivač ili parser koji čita tekst datoteke i predstavlja ga u obliku DOM-a
- Ove komponente pregledača nazivaju se raspoređivačke mašine (*layout engine*)
- Danas najpoznatije:
  - Trident/MSHTML koji koristi Internet Explorer
  - Presto koji koristi Opera
  - Webkit koji koriste Safari, Google Chrome, Google Android...
  - Gecko koji koriste Mozilla alati (Firefox,...)

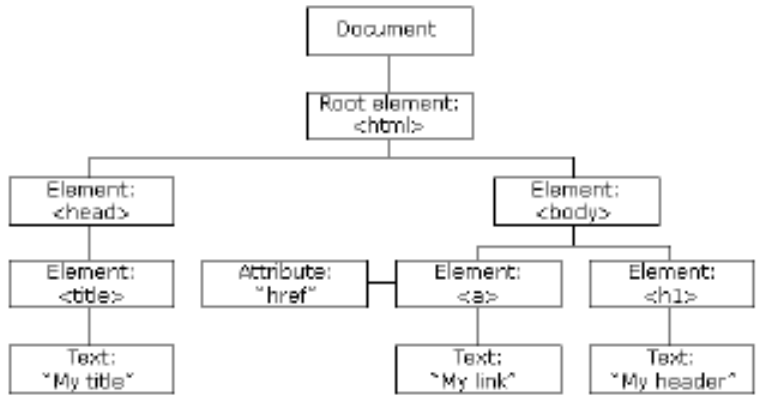
Struktura DOM

- Svakom delu dokumenta pridružen je zaseban DOM objekat
- Svojstvima objekta pristupa se sa `objekat.svojstvo`, a metodama sa `objekat.metod(parametri)`
- Svaki DOM objekat ima svoj tip (kojim su određena njegova svojstva i metode)
- Tipovi su određeni interfejsima, a interfejsi su organizovani u hijerarhiju nasleđivanja
- Npr. svi DOM objekti su čvorovi DOM stabla i implementiraju interfejs `Node`; objekti koji odgovaraju elementima dokumenta implementiraju interfejs `Element` koji nasleđuje interfejs `Node`

DOM stablo

- DOM objekti su međusobno povezani i čine strukturu stabla
- Svaki objekat predstavlja jedan čvor stabla; postoje različite vrste čvorova:
  - čvor dokumenta
  - čvor elementa
  - čvor teksta
  - čvor atributa
  - čvor komentara

```
<html>
  <head>
    <title>My title</title>
  </head>
  <body>
    <a href="page.html">My link</a>
    <h1>My header</h1>
  </body>
</html>
```



DOM stablo

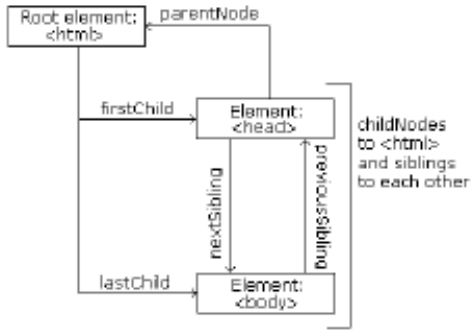
- Čvorovi u stablu su u odnosima: roditelj, dete, brat
- U stablu postoji jedinstveni čvor koji se označava kao koren
- Svaki čvor može da ima proizvoljan broj dece; lista dece svakog čvora je uređena
- List je čvor bez dece
- Jedan čvor je brat drugog čvora ako imaju istog roditelja

DOM stablo: interfejs `Node`

- Svojstva kojima se pristupa osnovnim podacima o čvoru:
  - nodeType – tip čvora (1 - element, 2 - atribut, 3 - tekst, 8 - komentar, 9 - dokument)
  - nodeName – ime čvora, nepromenljivo svojstvo (ime elementa, ime atributa, `#text`, `#comment`, `#document`)
  - nodeValue – vrednost čvora (za atribute – vrednost atributa, za tekst i komentare – sam tekst, za element i dokument vraća `NULL`)
  - innerHTML – za svaki čvor sadrži HTML kôd koji ga opisuje (uključujući i njegove naslednike)
- Celom dokumentu odgovara čvor kome se može pristupiti sa document
- Korenom čvoru dokumenta može se pristupiti sa document.documentElement

DOM stablo: interfejs Node

- Svojstva za navigaciju kroz DOM stablo:
  - firstChild – prvo dete čvora, za list vraća NULL
  - lastChild – poslednje dete čvora, za list vraća NULL
  - childNodes – niz čvorova dece datog čvora
  - parentNode – roditeljski čvor, u slučaju korena vraća NULL
  - nextSibling – naredni brat čvora, za poslednji čvor u listi dece vraća NULL
  - previousSibling – prethodni brat čvora, za prvi čvor u listi dece vraća NULL
  - attributes – niz čvorova atributa čvora
  - ownerDocument – jedinstveni čvor dokumenta (svi čvorovi su sadržani u ovom čvoru)



DOM stablo: interfejs Node

```
<html>
  <head>
    <title>My title</title>
  </head>
  <body>
    <a href="page.html">My link</a>
    <h1>My header</h1>
  </body>
</html>
```

- Primer: tekstu `My header` moguće je pristupiti na sledeći način:

```
document.documentElement.firstChild.nextSibling
        .childNodes[1].firstChild.nodeValue
```

DOM stablo: interfejs Node

- Metode za manipulaciju stablom:
  - var c = document.createElement("p") – kreira novi čvor (element p) ali ga ne dodaje u stablo
  - node.appendChild(c) – dodaje novo dete *c* čvoru node (na kraj)
  - node.removeChild(c) – uklanja dete *c* čvora node
  - node.insertBefore(nc, rc) – ubacuje novo dete *nc* pre postojećeg deteta *rc*
  - node.insertAfter(nc, rc) – ubacuje novo dete *nc* posle postojećeg deteta *rc*

DOM stablo: interfejs `Document`

- Objekat koji predstavlja dokument, pored interfejsa `Node` nasleđuje i interfejs `Document`
- Ovaj interfejs sadrži metode za lociranje čvorova bez direktne navigacije kroz DOM stablo:
  - document.body – body element
  - document.getElementById(id) – locira se čvor sa datim identifikatorom
  - document.getElementsByTagName(tag) – vraća se lista svih čvorova koji odgovaraju datom elementu
  - document.getElementsByClassName(cls) – vraća se lista svih čvorova koji pripadaju datoj klasi
  - document.querySelector(sel) – vraća se prvi čvor koji zadovoljava dati selektor
  - document.querySelectorAll(sel) – vraća se lista svih čvorova koji zadovoljavaju selektor
- Ovo je najčešći način na koji se pristupa elementima u stablu

Promene stilova elemenata

- Svaki čvor koji odgovara elementu ima svojstvo style
- Svojstvo style sadrži svojstva koja odgovaraju CSS svojstvima
  - color
  - backgroundColor
  - display
  - borderWidthTop
  - ...

```
var el = getElementById("mydiv");
el.style.color = "red";
el.style.borderWidth = "2px";
```

Događaji

- Događaji (engl. events) su mehanizam koji omogućava interakciju veb strane sa korisnikom
- Najčešće nastaju kao reakcija na odgovarajuću radnju korisnika (klik i prelaz mišem, pritisak tastera na tastaturi)
- Postoje i događaji koji se dešavaju nevezano za ponašanje korisnika (npr. učitavanje strane ili nekog elementa na strani je takođe događaj)
- Najzad, događaji se mogu emitovati i ručno iz JavaScript programa, kreiranjem odgovarajućeg objekta događaja (Event konstruktor) i pozivanjem metode el.dispatchEvent(event) za odgovarajući element
- Element koji emituje događaj zove se ciljni (engl. target) element

Tipovi događaja

- Svaki događaj ima svoj tip koji određuje njegovo značenje
- Tipovi događaja su zadati niskama koje sadrže ime tipa (npr. click, load, mouseover, keypress i sl.)
- Interno, događaji su predstavljeni objektima koji se kreiraju Event konstruktorom sa argumentom koji predstavlja naziv tipa događaja
- Nazivu tipa događaja za objekat događaja *e* se može pristupiti preko svojstva *e.type*
- Postoje i izvedeni konstruktori MouseEvent, KeyboardEvent i sl. koji kreiraju objekte događaja sa dodatnim svojstvima koje bliže određuju odgovarajuće događaje (npr. koordinate miša, kôd pritisnutog tastera)
- Događaj se uvek vezuje za neki element – ciljni element kome se može pristupiti pomoću *e.target* (npr. element na koji je kliknuto mišem, element koji je u fokusu prilikom pritiska tastera na tastaturi, i sl.)
- Tipično, veb pregledač automatski kreira odgovarajući objekat i emituje događaj kao reakciju na neko dešavanje ili akciju korisnika

Obrada događaja

- Događaji se obrađuju tako što se definišu posebne funkcije koje služe kao upravljači događajima (engl. event handlers)
- Za svaki element mogu se postaviti različiti upravljači za različite tipove događaja
- Kada se događaj desi, tada se najpre poziva upravljač za odgovarajući tip događaja za element koji emituje događaj (ciljni element)
- Nakon toga se događaj u talasu prenosi kroz pretke ciljnog elementa i za svaki od njih se u redosledu ka korenu pozivaju odgovarajući upravljači događajem tog tipa, ako postoje
- Upravljač događaja kao argument ima objekat događaja
- Unutar upravljača događaja ključna reč this se odnosi na element za koji se upravljač izvršava
- Dogadjaji se tipično obradjuju na nivou elementa koji je izazvao dogadjaj, ili na nivou document čvora

Registrovanje upravljača događajima

- Prvi način: pomoću HTML atributa:

```
<span onclick="this.style.color = 'red';"> Klikni me </span>
```

- Drugi način: pomoću onclick svojstva DOM čvora:

```
<span id="s"> Klikni i mene </span>
<script type="text/JavaScript">
 var elem = document.getElementById("s");
 elem.onclick = function (e) { this.style.color = 'red'; };
</script>
```

- Treći način: pomoću addEventListener metode DOM čvora (preporučljivo):

```
<span id="s"> Klikni i mene </span>
<script type="text/JavaScript">
 var elem = document.getElementById("s");
 elem.addEventListener("click",
      function (e) { this.style.color = 'red'; });
</script>
```

Neki značajni tipovi događaja

- load: emituje se kada se završi sa učitavanjem dokumenta ili elementa
- submit i reset: vezani za forme (vidi dole)
- resize: promena dimenzija prozora ili elementa
- scroll: skrolovanje dokumenta ili elementa
- keydown: taster pritisnut dok je element u fokusu
- keypress: taster emitovao štampajući karakter
- keyup: taster otpušten
- mouseover: miš je postavljen iznad elementa
- mousedown: levi taster miša je pritisnut dok je iznad datog elementa
- mouseup: levi taster miša je otpušten
- click: klik na nekom elementu
- dbclick: dvostruki klik na nekom elementu
- contextmenu: desni klik na nekom elementu
- ...

Formulari

- U HTML-u postoji podrška za pravljenje formulara u koje posetioci veb-sajtova mogu da unose podatke
- Formular se sastoji iz kontrola – elemenata koji omogućavaju unos vrednosti ili izbor opcija
- Osnovni elementi:
  - form – predstavlja formular, atribut method odredjuje metod HTTP zahteva prilikom slanja upita koji je prikuljen formularom (GET ili POST), a atribut action odredjuje URL na koji se šalje upit
  - input – predstavlja veći broj različitih vrsta kontrola; atribut name predstavlja ime (ključ) parametra upita, value sadrži pridruženu vrednost, a atribut type predstavlja tip kontrole i može imati vrednosti:
    - text – polje za unos teksta (atribut value je početna vrednost)
    - password – polje za unos lozinke
    - radio – radio-dugme (sva dugmad iz iste grupe imaju istu vrednost atributa name)
    - checkbox – polje za štrikliranje (atribut checked ako je podrazumevano štikliran)
    - button – obično dugme (koje podrazumevano ne radi ništa)
    - submit – dugme čijim se aktiviranjem prikupljeni podaci šalju na server
    - reset – dugme koje resetuje formu
    - hidden – sakriveno polje sa fiksiranom vrednošću
    - date – izbor datuma
    - email – dodatna validacija ispravnosti e-mail adrese

Formulari

- Pored kontrola koja se mogu kreirati input elementom, postoje i drugi elementi koji mogu kreirati kontrole u formularu:
  - select – predstavlja padajuću listu:
    - atribut name odredjuje ime parametra upita koji je pridružen kontroli
    - atributom size postavlja se broj opcija koje se vide (podrazumevano 1)
    - pojedinačne stavke u listi se zadaju elementom option koji ima atribut value koji predstavlja vrednost pridruženu parametru
  - textarea – polje za unos teksta u više redova; atributi name, rows, cols
  - label – tekstualni opis pridružen kontroli; atibut for sadrži id kontrole koja dobija fokus klikom na labelu
  - button – kreira dugme koje može imati proizvoljan sadržaj. Atribut type: button, submit, reset

Primer formulara (1. deo)

```
<form>
  <label for="ime">Ime i prezime: </label>
  <input type="text" id="ime" name="ime" />
  <br />

  <label for="ime">Lozinka: </label>
  <input type="password" id="lozinka" name="lozinka" />
  <br />

  <label for="adresa">Adresa: </label>
  <textarea id="adresa" name="adresa"></textarea>
  <br />

  <label for="vrsta">Vrsta pice: </label>
  <select id="vrsta" name="vrsta">
    <option value="kaprićoza">Kaprićoza</option>
    <option value="margarita">Margarita</option>
    <option value="vegetarijana">Vegetarijana</option>
  </select>
  <br />
```

Primer formulara (2. deo)

```
  <input type="radio" name="velicina" id="velika" value="velika"
         checked />
  <label for="velika">Velika</label> <br/>
  <input type="radio" name="velicina" id="srednja" value="srednja" />
  <label for="srednja">Srednja</label> <br/>
  <input type="radio" name="velicina" id="mala" value="mala" />
  <label for="mala">Mala</label> <br/>

  <input type="checkbox" name="kecap" id="kecap" />
  <label for="kecap">Kečap</label>
  <input type="checkbox" name="origano" id="origano" />
  <label for="origano">Origano</label>
  <br />

  <input type="button" value="Poništi"
         onclick="ponisti()"/>

  <input type="submit" value="Naruči"
         onclick="naruciPicu()"/>
</form>
```

Izgled formulara