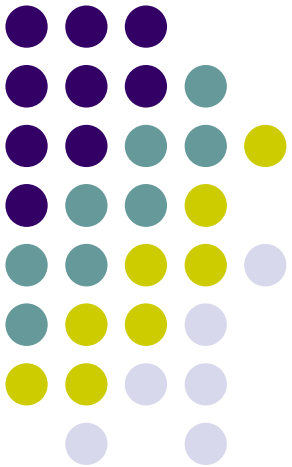
Klijentske tehnologije

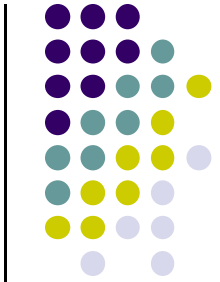
DHTML, DOM, JavaScript

Maja Štula

ak. god. 2011/2012

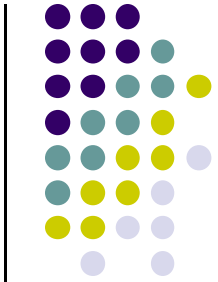


DHTML



- DHTML (*Dynamic HTML*) nije W3 konzorcijum standard već je to pojam koji su uveli Netscape i Microsoft za označavanje kombinacija novih tehnologija podržanih od 4 generacije pretraživača (IE 4.x, Netscape 4.x,…).
- DHTML je kombinacija tehnologija za kreiranje dinamičkih Web stranica na strani klijenta.
- DHTML se odnosi na kombinaciju HTML 4.0, CSS-a, DOM-a i JavaScript-a.

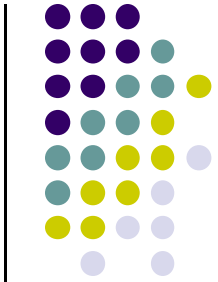
DHTML primjer



PRIMJER:

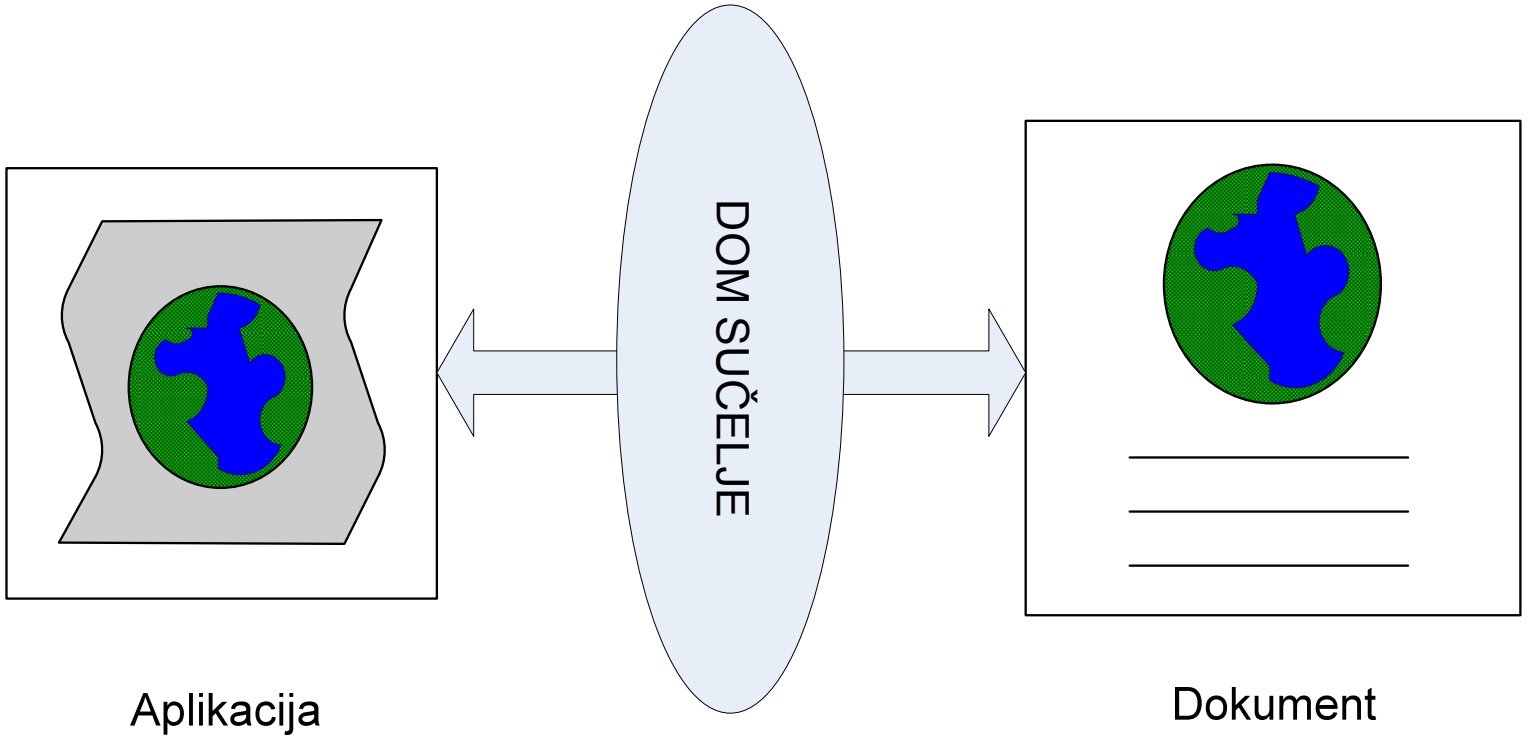
```
<html>
<head>
<script type="text/javascript">
function bgChange(bg)
{
document.body.style.background=bg
}
</script>
</head>
<body>
<b>Mouse over the squares and the background color will change!</b>
<table width="300" height="100">
 <tr>
  <td onmouseover="bgChange('red')" onmouseout="bgChange('transparent')" bgcolor="red">
  </td>
  <td onmouseover="bgChange('blue')"
      onmouseout="bgChange('transparent')"
      bgcolor="blue">
  </td>
  <td onmouseover="bgChange('green')"  onmouseout="bgChange('transparent')"  bgcolor="green">
  </td>
</tr>
</table>
</body>
</html>
```

DOM

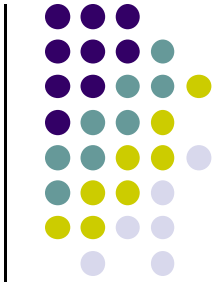


- DOM (*Document Object Model*) je W3 konzorcijum standard koji definira sučelje (API) između programa (skripti) i HTML (XHTML) i XML dokumenata koje omogućava aplikaciji dinamički pristup i mijenjanje sadržaj, strukture i stila dokumenta. Sučelje je neovisno o platformi i jeziku.
- DOM definira logičku strukturu dokumenta i načine rada sa dokumentom.
- DOM definira standardni skup “objekata” za predstavljanje HTML i XML dokumenata i standardno sučelje za pristup i manipulaciju njima. Elementi DOM-a nisu klasični objekti (u smislu objektno-orijentiranog programskog jezika), ne mogu se instancirati i sl., ali su objekti u smislu da se dijelovi dokumenta ne promatraju u DOM-u kao podaci, nego kao “objekti” određenih svojstava i funkcionalnosti.
- http://www.w3.org/DOM/

DOM



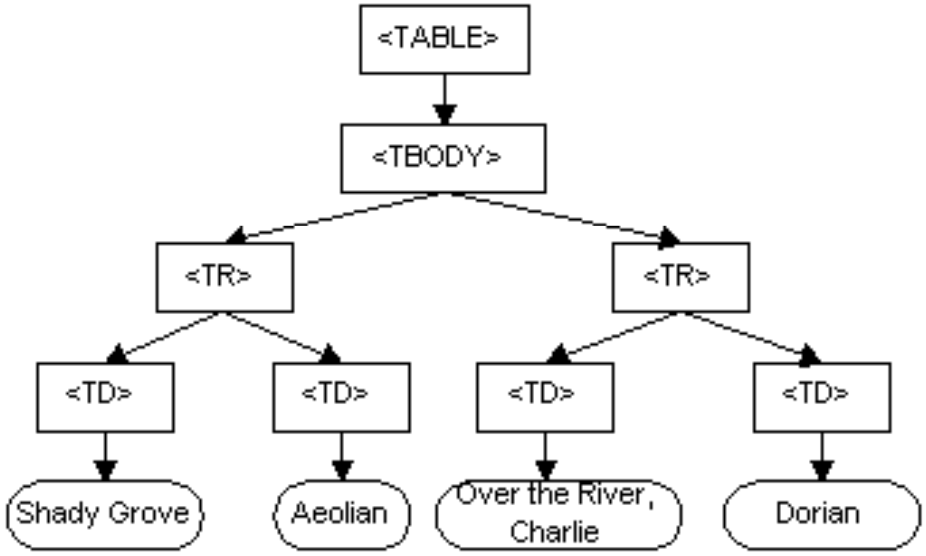
DOM



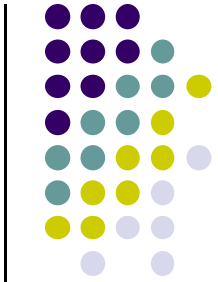
- Prije no što je W3 konzorcijum definirao W3 DOM pretraživači su koristili različita sučelja preko kojih je JavaScript rukovao HTML elementima.
- Struktura dokumenta prema DOM-u je stablasta (*tree*).
- Elementi dokumenta se nazivaju čvorovi (*nodes*).
- Takva struktura zahtjeva da čitav dokument bude učitan u memoriju i parsiran prije no što aplikacije mogu sa takvim dokumentom raditi. (loše svojstvo DOM-a)

DOM

```
<TABLE>
<TBODY>
<TR>
<TD>Shady Grove</TD>
<TD>Aeolian</TD>
</TR>
<TR>
<TD>Over the River,
    Charlie</TD>
<TD>Dorian</TD>
</TR>
</TBODY>
</TABLE>
```

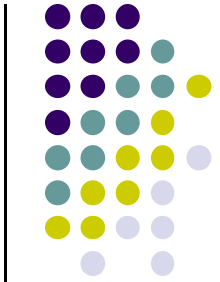


DOM



- W3 konzorcijum izdaje specifikacije DOM standarda. Trenutno postoje tri verzije:
  - DOM razina 1 (DOM Level 1) - se odnosi na osnovu HTML modela dokumenata, te na XML dokumente. Sadrži funkcionalnost za navigaciju i manipulaciju dokumentima. Ta specifikacija je omogućila pristup i manipulaciju sa svakim elementom u HTML stranici. Svi pretraživači imaju implementiranu tu preporuku. DOM Level 1 je objavljen kao W3C preporuka 1. listopada 1998. Radni prijedlog preporuke za drugo izdanje je objavljen 29. rujna 2000.

DOM



- DOM razina 2 (DOM Level 2) - dodaje objektni model stranica stila (CSS) DOM razini 1 i definira funkcionalnost za mijenjanje informacije u ovisnosti o stilu pridijeljenom dokumentu. Definira model događaja i omogućava podršku za XML kolekcije imena. DOM Level 2 specifikacija je objavljena kao W3C preporuka 13. studenog 2000:
- DOM razina 3 (DOM Level 3) - adresira učitavanje i spremanje dokumenta, kao i sadržaj modela s podrškom za potvrdu dokumenta. Također adresira pogled dokumenta (*document view*), formatiranje dokumenta. W3C preporuka 2004.

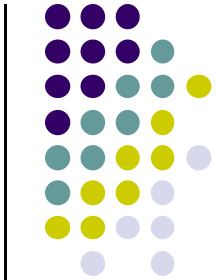
MEHANIZAM PRIKAZA

- Mehanizam prikaza (*layout engine, rendering engine*) je softver koji dohvaća web sadržaj (HTML, XML, slike, …) i prikaz sadržaja (CSS, XSL, …) i prikazuje formatirani sadržaj na ekranu. Mehanizam prikaza "iscrtava" sadržaj stranice na ekranu ili printeru.
- Mehanizam prikaza koriste pretraživači, e-mail klijentske aplikacije i slične aplikacije koji trebaju prikazati web sadržaj.

	Creator	Cost (USD)	Open Source	Software license	Leading application
Gecko	Mozilla Corporation	Free	Yes	MPL, MPL/GPL/LGPL tri-license	Mozilla Firefox
iCab	Alexander Clauss	Free / US $29 (Pro)	No	Proprietary	iCab
KHTML	KDE	Free	Yes	LGPL	Konqueror
Presto	Opera Software	Free	No	Proprietary	Opera
Tasman	Microsoft	Free	No	Proprietary	Internet Explorer for Mac
Trident	Microsoft	Free †	No	Proprietary	Internet Explorer
WebCore	Apple Computer	Free	Yes	LGPL	Safari
	Creator	Cost (USD)	Open Source	Software license	Leading application

Slika: Različiti mehanizmi prikaza

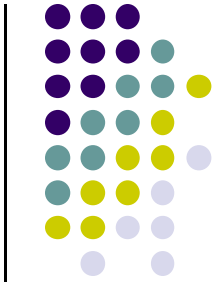
MEHANIZAM PRIKAZA



	Trident	Tasman	Gecko	WebCore	KHTML	Presto	iCab
DOM1	6.0	0	1.0	85	Yes	7.0	Yes
DOM2	Minor	Minor	Major	Major	Major	Major	Major
DOM3	No	No	Minor	Minor	Minor	Minor	Minor
	Trident	Tasman	Gecko	WebCore	KHTML	Presto	iCab

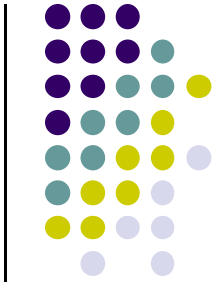
Testiranje podrške za DOM
http://www.w3.org/2003/02/06-dom-support.html

DOM



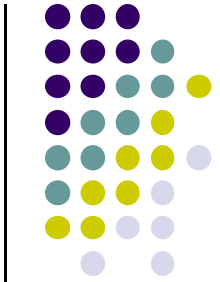
- DOM standard se sastoji od nekoliko dijelova. Osnovni dijelovi DOM standarda podržani u DOM razini 2 i 3 su:
  - *Core* DOM – definira standardni set objekata za bilo koji strukturirani dokument
  - HTML DOM - definira standardni set objekata za HTML dokumente
  - XML DOM – definira standardni set objekata za XML dokumente
  - DOM događaji (*events*) – definira standardni set događaja
  - DOM CSS – ovo sučelje je definirano kako bi se omogućio pristup CSS elementima

Core DOM elementi



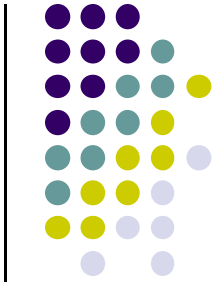
- DOM predstavlja dokument kao hijerarhijski organizirane objekte (čvorove) koji mogu implementirati određena sučelja. Neki tipovi čvorova mogu imati čvorove djecu, a neki ne. Za HTML i XML tipovi čvorova su:
  1. Dokument (*Document*) – je osnovni čvor XML ili HTML dokumenta koji predstavlja čitav dokument.(http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/core.html#i-Document), može imati djecu tipa Element (maksimalno jedan), Naredba, Komentar, Tip Dokumenta (maksimalno jedan)
  2. Fragment dokumenta (*DocumentFragment*) može imati djecu tipa Element, Naredba, Komentar, Tekst, CDATASection, Referenca entiteta
  3. Tip dokumenta (*DocumentType*) ne može imati djecu

Core DOM elementi



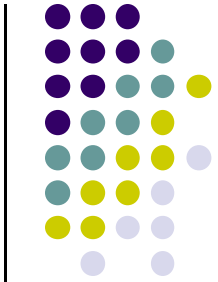
4. Referenca entiteta (*EntityReference*) može imati djecu tipa Element, Naredba, Komentar, Tekst, CDATASection, Referenca entiteta
5. Element može imati djecu tipa Element, Naredba, Komentar, Tekst, CDATASection, Referenca entiteta
6. Atribut (*Attr*) može imati djecu tipa Tekst, Referenca entiteta
7. Naredba (*ProcessingInstruction*) ne može imati djecu
8. Komentar (*Comment*) ne može imati djecu
9. Tekst ne može imati djecu
10. CDATASection ne može imati djecu
11. Entitet (*Entity*) može imati djecu tipa Element, Naredba, Komentar, Tekst, CDATASection, Referenca entiteta
12. Notacija (*Notation*) ne može imati djecu

HTML DOM



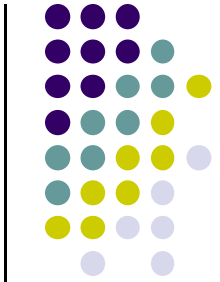
- HTML DOM definira objekte i metode koji su vezani specifično uz HTML i XHTML.
- Osnovna funkcionalnost rada sa dokumentom definirana je u Core DOM-u, a HTML DOM uvodi specifične elemente npr. HTMLDocument je element izvedene iz Document Core elementa koji predstavlja čitav HTML dokument (isto kao i Document element).
- HTML DOM definira HTML dokument kao kolekciju objekata. Objekti imaju svojstva i metode. Objekti mogu odgovarati na događaje.
- HTML DOM vidi HTML dokument kao stablastu strukturu elemenata ugrađenih unutar drugih elemenata.

HTML DOM



- Svim elementima, tekstu koji sadrže i njihovim atributima, može se pristupiti preko DOM stabla. Njihov sadržaj može biti mijenjan i izbrisan, te se mogu kreirati novi elementi.
- Prema DOM-u, sve u HTML dokumentu je čvor.
  - cijeli dokument je čvor dokumenta;
  - svaki HTML tag je čvor element;
  - tekstovi koji se nalaze u HTML elementima su čvorovi teksta;
  - atributi elemenata su posebni čvorovi atributa;
  - komentari su čvorovi.

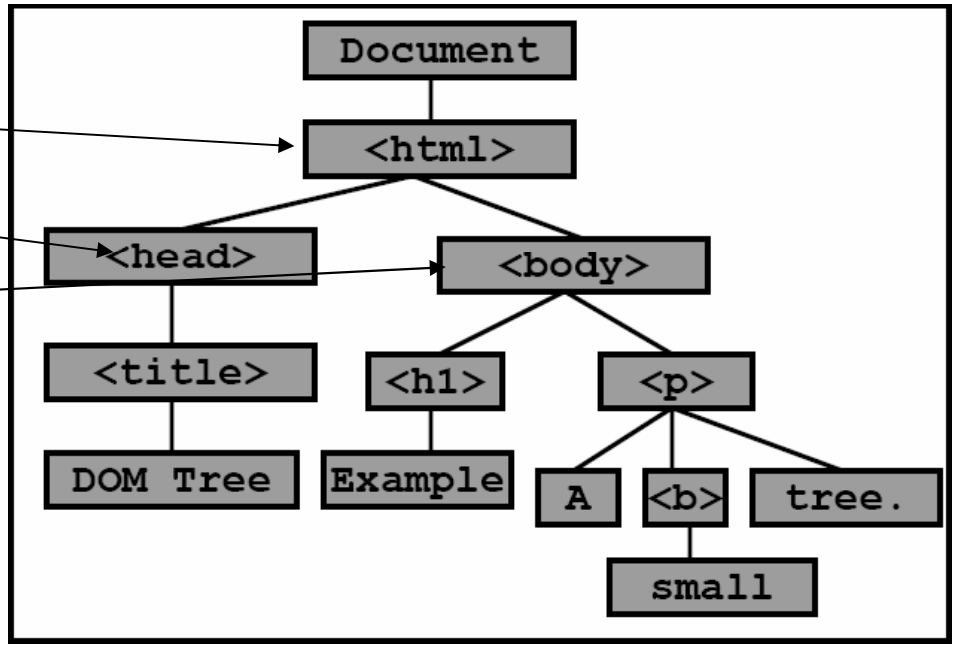
HTML DOM



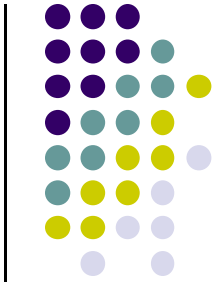
- Čvorovi imaju hijerarhijski odnos jedan prema drugome.
- Svi čvorovi u HTML dokumentu tvore stablo dokumenta (ili stablo čvorova).
- Stablo počinje u čvoru Dokument i nastavlja se granati dok ne dosegne sve čvorove teksta u najnižoj razini stabla.
- Svi čvorovi u stablu čvorova su u određenim međuodnosima.

HTML DOM

```
<html>
  <head>
    <title>DOM Tree</title>
  </head>
<body>
  <h1>Example</h1>
  <p>A <b>small</b> tree.</p>
</body>
</html>
```

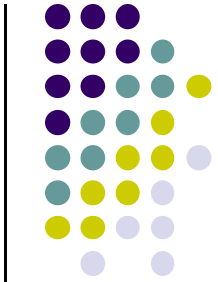


HTML DOM



- Što to znači da su svi čvorovi u stablu u određenim međuodnosima?
- Svaki čvor osim čvora Dokument ima svog čvora roditelja.
- Npr. sa prethodnog slajda *<html>* čvor je čvor roditelj od *<head>* i *<body>* čvorova, čvor roditelj od tekstualnog čvora "*small"* je *<b>* čvor, a čvor roditelj od tekstualnog čvora "*Example"* je *<h1>* čvor.
- Većina čvorova ima čvorove djecu.
- Npr. sa prethodnog slajda čvor *<head>* čvor ima jedan čvor dijete, *<title>* čvor. *<title>* čvor također ima jedan čvor dijete, tekstualni čvor "*DOM Tree*".
- Čvorovi su braća ako imaju istog roditelja. Npr. *<h1>* i *<p>* čvorovi su braća, jer je njihov roditelj *<body>* čvor.

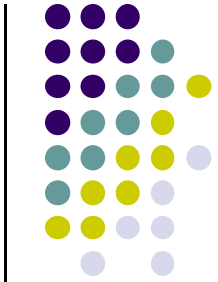
DOM



DOM svojstvo	Vrijednost koju vraća	Objašnjenje
firstChild	čvor prvog potomka	svi potomci su uključeni u *childNodes* kolekciju
lastChild	čvor posljednjeg potomka	svi potomci su uključeni u *childNodes* kolekciju
nextSibling	sljedeći potomak od čvora roditelja	objekt
parentNode	referenca na čvor roditelja	objekt
previousSibling	referenca na prijašnjeg potomka od čvora roditelja	objekt

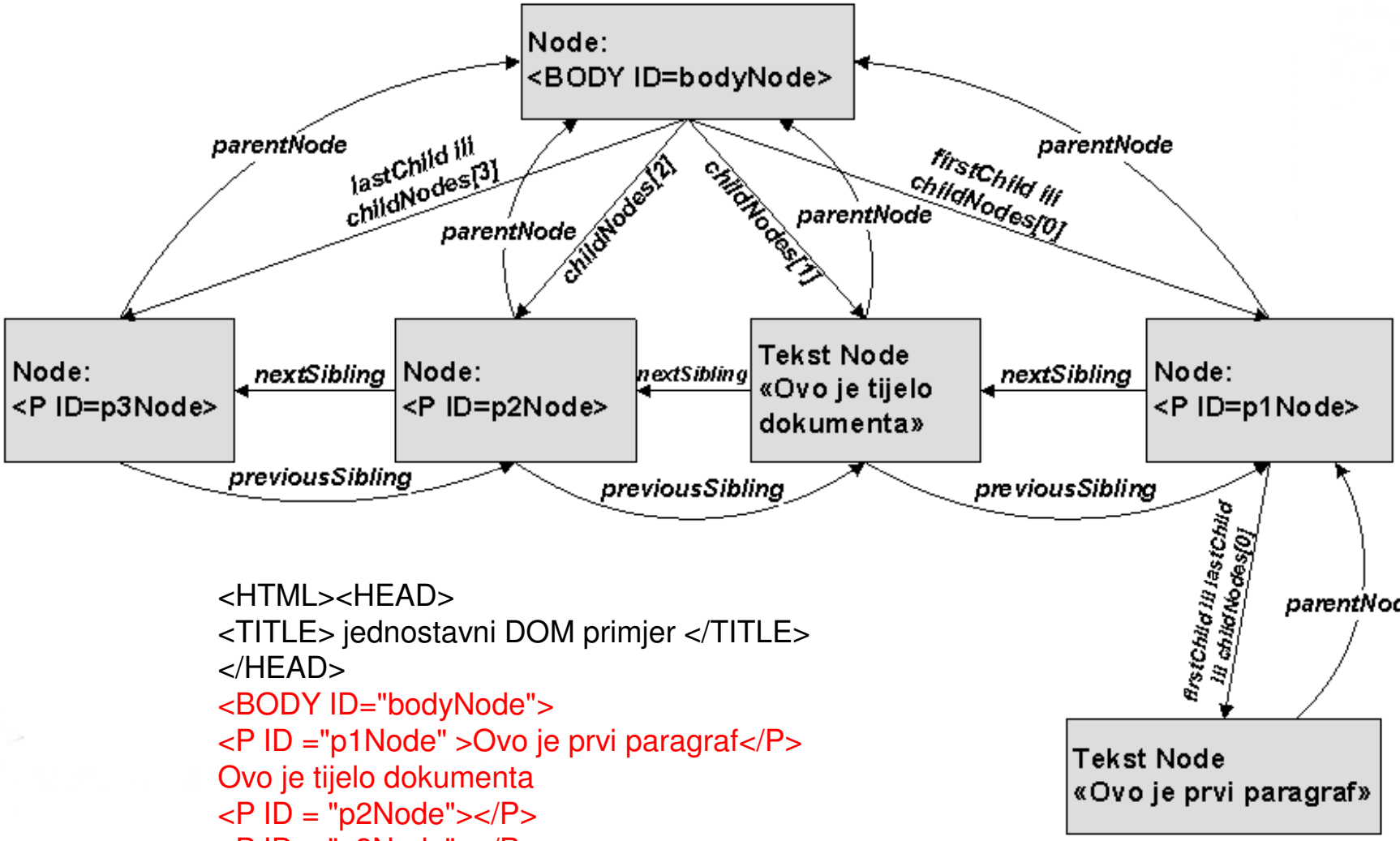
- DOM (*Document Object Model*) osigurava niz svojstava (*properties*) odnosno metoda objekata koja olakšavaju kretanje kroz sam dokument, te pristup određenim elementima. Dohvaćanje čvora dokumenta radi se ili preko gore navedenih svojstava čvora ili metodama *getElementById* ili *getElementsByTagName*() . *getElementById*() metoda vraća element sa ID atributom navedenim u pozivu metode. *getElementsByTagName*() metoda vraća sve elemente sa imenom navedenim u pozivu metode.

DOM



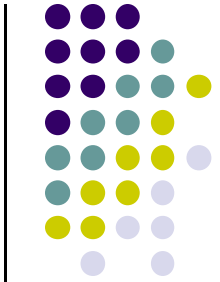
PRIMJER:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE> New Document </TITLE>
<META NAME="Generator" CONTENT="EditPlus">
<META NAME="Author" CONTENT="">
<META NAME="Keywords" CONTENT="">
<META NAME="Description" CONTENT="">

<script type="text/javascript">
function dohvati_prvi_child()
{ alert(document.firstChild.nodeValue);}
function dohvati_zadnji_child()
{ alert(document.lastChild.nodeName); }
</script>

</HEAD>
<BODY>
<INPUT TYPE=button onclick="dohvati_prvi_child()" VALUE="Dohvati  prvi child element">
<BR><BR>
<INPUT TYPE=button onclick="dohvati_zadnji_child()" VALUE="Dohvati zadnji child element">
</BODY>
</HTML>
```



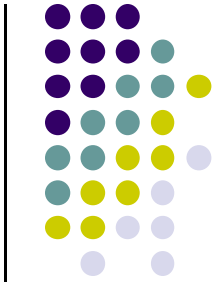

```
<HTML><HEAD>
<TITLE> jednostavni DOM primjer </TITLE>
</HEAD>
<BODY ID="bodyNode">
<P ID ="p1Node" >Ovo je prvi paragraf</P>
Ovo je tijelo dokumenta
<P ID = "p2Node"></P>
<P ID = "p3Node"></P>
</BODY>
</HTML>
```

Kretanje kroz DOM dokument



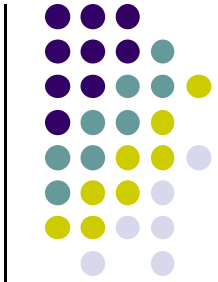
- Strelice na prethodnoj slici pokazuju kojim je različitim rutama moguće doći do određenog čvora.
- Npr. sa prethodnog slajda čvor *<BODY>* ima ID atributu vrijednosti *bodyNode*. Od čvora *<BODY>* može se doći do njegovog prvog potomka upotrebom *bodyNode.firstChild* ili *bodyNode.childNodes[0]*, a do drugog potomka (tekstualnog čvora) *bodyNode.childNodes[1]* itd. Također se može pristupiti četvrtom potomku (zadnjem) sa *bodyNode.childNodes[3]* ili *bodyNode.lastChild*.

Kretanje kroz DOM dokument



- Pozicioniranje po DOM dokumentu je izuzetno složeno korištenjem svojstava *parentNode*, *lastChild*, ...
- Stoga se najčešće sa pozicioniranje na određeni DOM element koristi ID atribut kojim se jedinstveno identificira određeni DOM element i metoda *getElementbyId*.
- ID atribut olakšava pristup čvoru, tj. preko id atributa HTML elementa može se direktno pristupiti čvoru.

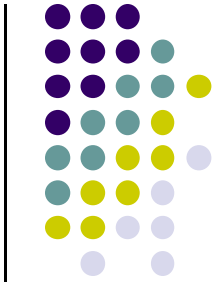
Kretanje kroz DOM dokument



PRIMJER:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE> jednostavni DOM primjer </TITLE>
<script type="text/javascript">
function dohvati_body_prvi_child()
{ alert(document.getElementById("bodyNode").firstChild.nodeName);}
function dohvati_body_zadnji_child()
{ alert(document.getElementById("bodyNode").childNodes[4].nodeName); }
</script>
</HEAD>
<BODY ID="bodyNode">
<P ID ="p1Node" >Ovo je prvi paragraf</P>
Ovo je tijelo dokumenta
<P ID = "p2Node"></P>
<P ID = "p3Node"></P>
<INPUT TYPE=button onclick="dohvati_body_prvi_child()" VALUE="Dohvati od body elementa prvi child element">
<BR><BR>
<INPUT TYPE=button onclick="dohvati_body_zadnji_child()" VALUE="Dohvati od body elementa peti child element">
</BODY>
</HTML>
```

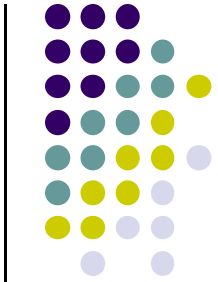
Svojstva čvora



- Osim svojstava za pozicioniranje u DOM dokumentu svaki čvor sadrži i sljedeća tri svojstva:

1. nodeName svojstvo sadrži ime čvora. Ako je čvor tipa element tada je to ime HTML ili XML oznake (P, FONT, UL). Ako je čvor tipa atribut tada je to ime atributa. Ako je čvor tekst tada je to #text, a ako je čvor dokument (osnovni čvor) tada je to #document.
2. nodeType svojstvo sadrži tip čvora. Ako je čvor element to je svojstvo postavljeno na 1, ako je čvor atribut svojstvo je 2, za tekst je 3, komentar 8 i dokument 9.
3. nodeValue svojstvo sadrži vrijednost čvora. Za čvor tipa Dokument i elemente vraća null vrijednost, a za sve ostale čvorove može imati vrijednost.

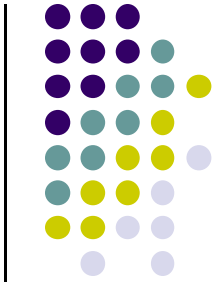
Kretanje kroz DOM dokument



PRIMJER:

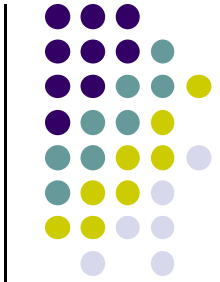
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE> jednostavni DOM primjer </TITLE>
<script type="text/javascript">
function dohvati_prvi_brat()
{
alert(document.getElementById("p1Node").nextSibling.nodeName);
alert(document.getElementById("p1Node").nextSibling.nodeValue);
alert(document.getElementById("p1Node").nextSibling.nextSibling.nodeName);
alert(document.getElementById("p1Node").nextSibling.nextSibling.nodeValue);
}
</script>
</HEAD>
<BODY ID="bodyNode">
<P ID ="p1Node" >Ovo je prvi paragraf</P>
Ovo je tijelo dokumenta
<P ID = "p2Node"></P>
<P ID = "p3Node"></P>
<INPUT TYPE=button onclick="dohvati_prvi_brat()" VALUE="Dohvati prvi i drugi sybling od prvog p element">
</BODY>
</HTML>
```

Kretanje kroz DOM dokument



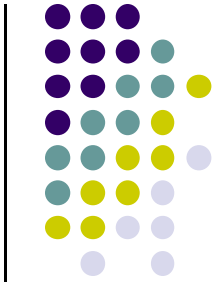
- Sve *<P>* oznake su označene jedinstvenim ID: *p1Node*, *p2Node* i *p3Node*. Ako se krene od *p1Node* čvora, moguće je dohvatiti drugog potomka *<BODY>* čvora (korijenskog čvora) sa *p1Node.nextSibling*. Treći potomak se dohvaća preko: *p1Node.nextSibling.nextSibling*, a posljednji preko: *p1Node.nextSibling.nextSibling.nextSibling*.
- Ako bi četvrti potomak imao potomka (iako nema) tada bi se prvom potomku pristupilo kao: *p1Node.nextSibling.nextSibling.nextSibling.childNodes[0]*.
- Ako se krene sa trećim *<P>* čvorom, moguće je vratiti se natrag na drugog upotrebom *previousSibling* svojstva: *p3Node.previousSibling.previousSibling.previousSibling.childNodes[0]*.
- Ako se opet krene sa *<BODY>*, on ima jednog unuka, a to je sadržaj prvog *<P>* čvora. On se dohvaća sa *bodyNode.firstChild.firstChild*.

Kretanje kroz DOM dokument



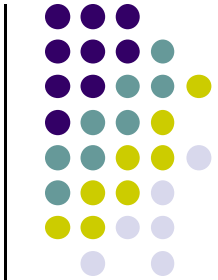
- Slijedeća vrsta navigacije je od potomka prema roditelju. Za pristup čvoru roditelja koristi se *parentNode* svojstvo. Da bi se došlo od bilo kojeg *<P>* čvora do *<BODY>* oznake koristi se *p1Node.parentNode*, *p2Node.parentNode* ili *p3Node.parentNode*.
- Može se ići u krug od *<BODY>* čvora do unuka i natrag sa *bodyNode.firstChild.firstChild.parentNode.parentNode*.

Mijenjanje DOM dokumenta



- Kreiranje čvorova je moguće sa skupom metoda koje se upotrebljavaju nad *document* objektom. Te metode su:
  - *createElement(String tip)*: Kreira novi element određenog tipa i vraća referencu na taj element.
  - *createTextNode(String tekst)*: Kreira novi tekstualni čvor sa sadržajem koji je specificiran kao tekst.
- Svaki čvor nasljeđuje metodu *appendChild*, koja omogućuje umetanje novog čvora na mjesto djeteta čvora nad kojim se pozvala metoda *appendChild* na točno određeno mjesto.
- Svaki čvor nasljeđuje i metodu *removeChild*, (i još cijeli niz metoda) koja omogućuje brisanje djeteta.

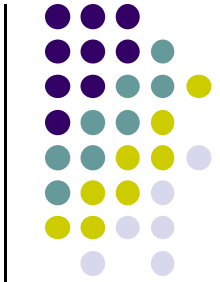
Mijenjanje DOM dokumenta



PRIMJER:

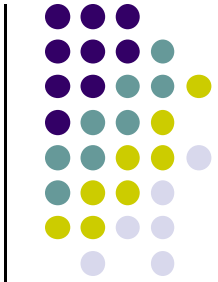
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML><HEAD><TITLE> jednostavni DOM primjer </TITLE>
<script type="text/javascript">
function umetni_tekst()
{
var p_elementi = document.getElementsByTagName("P");
var noviTekst = document.createTextNode("Ovo je neka rečenica");
p_elementi[1].appendChild(noviTekst);
// Zamjena postojećeg sadržaja
//p_elementi[1].innerHTML = "Ovo je neka rečenica";
}
function izbrisi_tekst()
{
var p_elementi = document.getElementsByTagName("P");
if (p_elementi[1].hasChildNodes())
  p_elementi[1].removeChild(p_elementi[1].firstChild);
}
</script>
</HEAD>
<BODY ID="bodyNode">
<P ID ="p1Node" >Ovo je prvi paragraf</P>
Ovo je tijelo dokumenta
<P ID = "p2Node"></P><P ID = "p3Node"></P>
<INPUT TYPE=button onclick="umetni_tekst()" VALUE="Umetni tekst u drugi P element">
<br><br><INPUT TYPE=button onclick="izbrisi_tekst()" VALUE="Izbrisi tekst iz drugog P elementa">
</BODY></HTML>
```

XML DOM



- XML DOM definira objekte (element) i metode koji su vezani specifično uz XML.
- Osnovna funkcionalnost rada sa dokumentom definirana je u Core DOM-u.
- XML DOM definira standardni način pristupa i manipulacije XML dokumenta. XML DOM vidi XML dokument kao stablastu strukturu elemenata ugrađenih unutar drugih elemenata.
- Svim elementima, tekstu koji sadrže i njihovim atributima, može se pristupiti preko DOM stabla. Njihov sadržaj može biti mijenjan i izbrisan, i mogu se kreirati novi elementi.

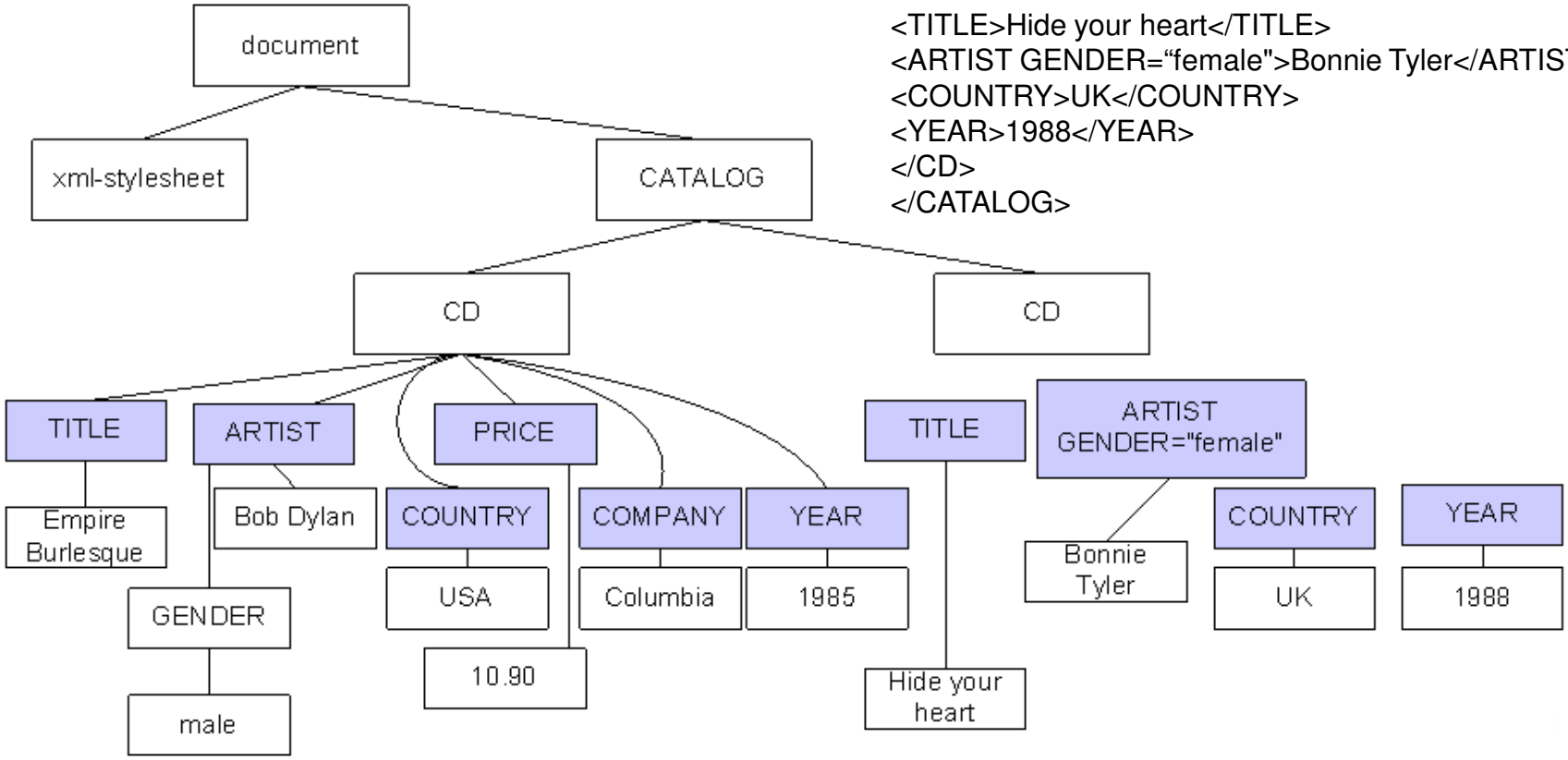
XML DOM

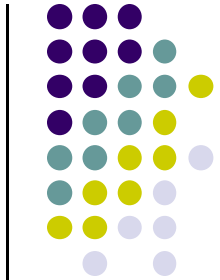


- Stablo ima jedan *root* čvor. Svi čvorovi osim *root* čvora imaju jedan čvor roditelj. Svaki čvor može imati više čvorova djece. I atributi se promatraju kao čvorovi XML DOM stabla.
- Prema DOM-u, sve u XML dokumentu je čvor.
  - cijeli dokument je čvor dokumenta;
  - svaki XML tag je čvor elementa;
  - tekstovi koji se nalaze u XML elementima su čvorovi teksta;
  - atributi elemenata su posebni čvorovi atributa;
  - komentari su čvorovi.

XML DOM

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/css" href="cd_catalog.css"?>
<CATALOG>
<CD>
        <TITLE >Empire Burlesque</TITLE>
        <ARTIST GENDER="male">Bob Dylan</ARTIST>
        <COUNTRY>USA</COUNTRY>
        <COMPANY>Columbia</COMPANY>
        <PRICE>10.90</PRICE>
        <YEAR>1985</YEAR>
</CD>
<CD>
<TITLE>Hide your heart</TITLE>
<ARTIST GENDER="female">Bonnie Tyler</ARTIST>
<COUNTRY>UK</COUNTRY>
<YEAR>1988</YEAR>
</CD>
</CATALOG>
```

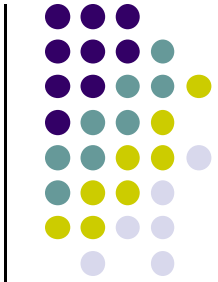




PRIMJER:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE> jednostavni DOM primjer </TITLE>
</HEAD>
<BODY>
<script type="text/javascript">
var xmlDoc;
// code for IE
if (window.ActiveXObject)
{
xmlDoc=new ActiveXObject("Microsoft.XMLDOM");
}
// code for Mozilla, Firefox, Opera, etc.
else if (document.implementation && document.implementation.createDocument)
{
xmlDoc=document.implementation.createDocument("","",null);
}
else
{
alert('Your browser cannot handle this script');
}
xmlDoc.async=false;
xmlDoc.load("cd_catalog.xml");
var x=xmlDoc.getElementsByTagName('TITLE');
for (i=0;i<x.length;i++)
{
        document.write(x[i].childNodes[0].nodeValue);
        document.write("<br />");

}
</script>
</BODY>
</HTML>
```

CSS DOM

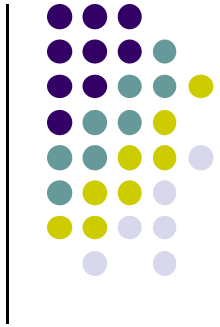


- CSS DOM definira objekt style preko kojeg je moguće pristupati stilu elementa dokumenta i mijenjati ga.
- Sintaksa je element.style.styleName = vrijednost

  Npr: document.body.style.background= “yellow”
- Postojeći stil je moguće izbrisati postavljanjem na prazni string “”.
- Postavljanje stila pojedinačnih elemenata preko class atributa moguće je sintaksom:

  element.className = ''; Brisanje stila klase

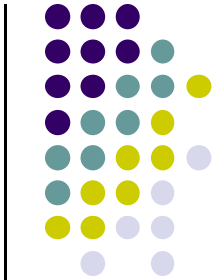
  element.className = 'newClassName'; Postavljanje stila klase



PRIMJER:

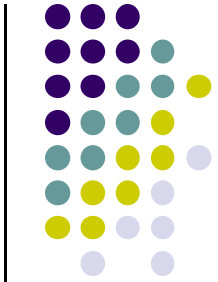
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE> New Document </TITLE>
</HEAD>
<script type="text/javascript">
function izbrisi_stil_jedan()
{
var p_elementi = document.getElementsByTagName("p");
p_elementi[0].jedan = ""; // Nije baš podržano
}
function postavi_stil_jedan()
{
var p_elementi = document.getElementsByTagName("p");
p_elementi[0].style.color = ‘green';
p_elementi[0].style.fontSize = '20pt';
}
</script>
<style type="text/css">
p.jedan {color: red; }
p.dva {color: blue;}
</style>
<BODY>
<p class="jedan">p klasa jedan</p>
<p class="dva">p klasa dva</p>
<INPUT TYPE=button onclick="izbrisi_stil_jedan()" VALUE="Izbrisi stil prvog p elementa">
<BR><BR>
<INPUT TYPE=button onclick="postavi_stil_jedan()" VALUE="Postavi stil prvog p elementa">
<BR><BR>
</BODY>
</BODY>
</HTML>
```

JavaScript



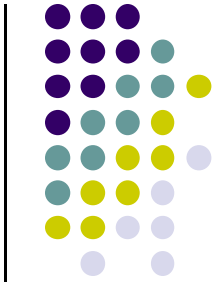
- JavaScript je klijentski (to znači da se izvršava na strani web klijenta u pretraživaču) skriptni objektno-orijentirani programski jezik.
- Skriptni kôd se ne kompajlira već se izvršava u trenutku poziva koda pomoću interpretera integriranog u browser.
- Jezik je razvijen u tvrtci Netscape 1995. godine. Netscape Navigator 2.0 je prvi podržavao JavaScript. Originalni naziv je bio “LiveScript”, a poslije je promijenjen u JavaScript da bi dostigao popularnost Jave. 1997. godine JavaScript je standardiziran od strane ECMA (*European association for standardizing information and communication systems*).

JavaScript



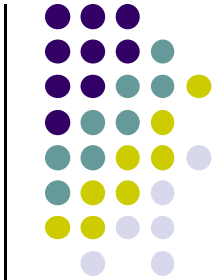
- Koristi se kako bi se HTML stranicama dodala interaktivnost i dinamika na strani web klijenta kroz:
  - JS može reagirati na događaje (može se podesiti tako da se izvrši tek nakon što se nešto dogodi, npr. nakon što HTML stranica završi učitavanje, nakon korisnikovog klika...)
  - JS može čitati i mijenjati sadržaj HTML elemenata preko DOM stabla
  - JS se može koristiti za analizu (*validate*) forme podataka (npr. provjeravanjem forme podataka prije nego što se proslijede serveru, rasteretiti ćemo server dodatnog obrađivanja podataka)
  - ....

JavaScript



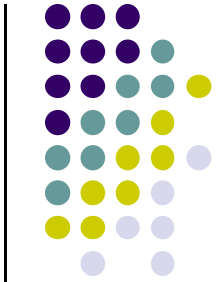
- Za uključivanje JS u kôd HTML stranice koristi se tag <script>.
- Unutar stranice može se nalaziti proizvoljan broj skripti.
- Skripte se mogu uključivati unutar HEAD dijela HTML stranice ili unutar BODY dijela stranice.
- Skripte se navode u HEAD dijelu ukoliko sadrže kôd koji želimo da se izvrši na određeni događaj. Skripte u HEAD dijelu pretraživač može učitati u memoriju bez izvršavanja i čekati da se negdje u stranici pozove učitana skripta.
- Skripte navedenu u BODY dijelu pretraživač izvrši kada naiđe na njih pri parsiranju HTML dokumenta.
- Skripta se može smjestiti i u zasebnu datoteku s ekstenzijom .js. U tom slučaju se u tagu <script> treba koristiti atribut src sa imenom skripte tj. URI-jem skripte.

JavaScript



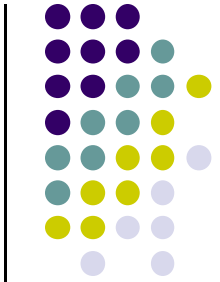
```
<html>
<head>
<script type="text/javascript">
                    JavaScript naredbe
</script>
</head>

<body>
<script type="text/javascript">
                    JavaScript naredbe
</script>
</body>
</html>

<script src=“xxxx.js"> </script>
```

JavaScript



- Varijable
- Literali
- Operatori
- Kontrolne strukture
- Objekti

JavaScript



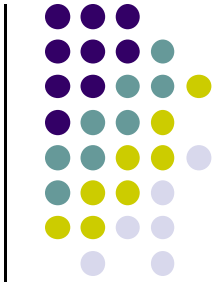
- Tip varijable se određuje na osnovu sadržaja.
- Varijabla se može deklarirati s ključnom riječi *var*:

  var mojavarijabla; ← opcionalno ;
- Može se kreirati automatski dodjelom vrijednosti:

  mojavarijable=1;

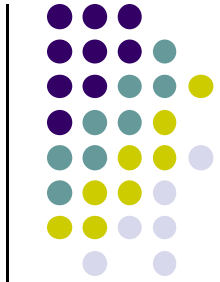
  mojdrugavarijabla=“Neki tekst”;
- Ključna riječ *var* deklarira lokalnu varijablu (u funkciji). Bez nje je varijabla globalna.

JavaScript



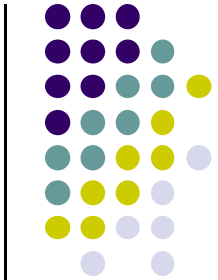
- Literali:
  - Brojevi 22 456.77
  - Stringovi “neki tekst”
  - Boolean true false

JavaScript



- Operatori (slično kao u C-u)
  - aritmetički +, -, *, /
  - usporedbe ==, !=, <, >
  - pridruživanja =
  - na razini bita <<, >>, &, |
  - konkatenacija stringova +

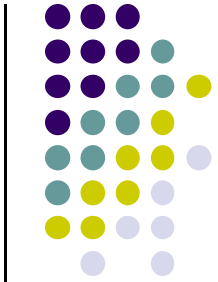
JavaScript grananja



- if (izvršava se ako je vrijednost uvjeta *true)*
- if...else (izvršava se jedan od dva bloka naredbi)
- switch (jedan od više blokova naredbi)

```
if (uvjet)
{ kod se izvršava ako je uvjet zadovoljen}
else
{ kod se izvršava ako uvjet nije zadovoljen}

switch (izraz)
{case vrijednost1:
kod će se izvršiti ako je izraz= vrijednost1
break
case vrijednost2:
kod će se izvršiti ako je izraz= vrijednost2
break
 default:kod će se izvršiti ako je izraz različit i od vr1 i vr2}
```

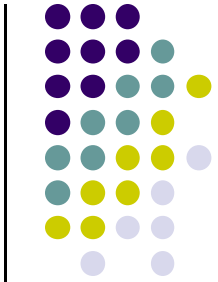
JavaScript petlje



- while – ponavlja određeni blok koda dok je uvjet zadovoljen
- do...while – blok koda se izvrši jednom, a tada se ponavlja dok je zadovoljen uvjet
- *for* – ponavlja set naredbi određen broj puta; *for in* koristi se za iteraciju kroz elemente polja ili kroz svojstva objekta

```
for (inicijalizacija; uvjet; inkrement)
{ kod koji će biti izvršen}

for (varijabla in objekt)
{ kod koji će biti izvršen}
```

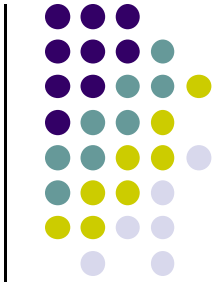
JavaScript privremeni prozori



- U JavaScriptu je moguće kreirati tri vrste privremenih prozora:
  - *Alert Box* (prozor za upozoravanje) se često koristi kako bi programer bio siguran da će informacija doći do korisnika. Kada se prozor za upozoravanje privremeno pojavi, korisnik mora kliknuti «OK» za nastavak rada. Sintaksa: alert("sometext")
  - *Confirm Box* (prozor za potvrdu) se često koristi kako bi korisnik mogao verificirati ili prihvatiti nešto. Kada se prozor za potvrdu privremeno pojavi, korisnik mora kliknuti ili «OK» ili «Cancel» da bi nastavio rad. Ako korisnik klikne «OK», prozor vraća *true*, a ako klikne «Cancel», prozor vraća *false*. Sintaksa: confirm("sometext")
  - *Prompt Box* (prozor za unos) se obično koristi kako bi korisnik mogao unijeti neku vrijednost prije ulaska na web stranicu. Kada se prozor za unos privremeno pojavi, korisnik mora kliknuti ili «OK» ili «Cancel» kako bi nastavio rad nakon unosa ulazne vrijednosti. Ako korisnik klikne «OK» prozor vraća ulaznu vrijednost, a ako klikne «Cancel», prozor vraća *null*. Sintaksa: prompt("sometext","defaultvalue")

http://www.w3schools.com/js/js_popup.asp

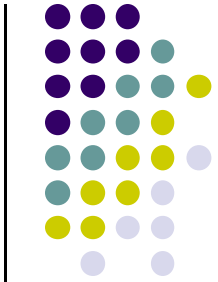
JavaScript



- Funkcije se definiraju na početku dokumenta, u HEAD dijelu, na taj način osiguravamo da se funkcija učita prije njenog poziva.
- Definiranje funkcija vrši se tako da se funkciji dodjeli ime, definiraju argumenti i u tijelu funkcije napišu sve naredbe za koje želimo da se izvrše kada se funkcija pozove.

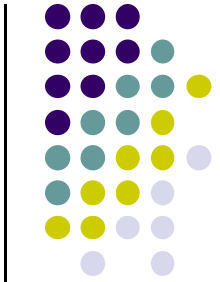
```
function ime_funkcije (arg1,arg2,...)
{ naredba }
```

Predefinirani JavaScript objekti



- Postoji cijeli niz objekata koji su ugrađeni (predefinirani) u JS.
- Neki od predefiniranih objekata koji nisu vezani uz DOM strukturu HTML dokumenta su:
  - Array
  - Boolean
  - Date
  - Math
  - String
  - .....

Predefinirani JavaScript objekti



- Sintaksa korištenja objekata je kao u C++/Java:

  imeobjekta.imeatributa

  imeobjekta.imemetode()

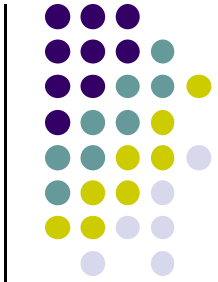
- Npr. ugrađeni objekt Math (tj. klasa Math sa statičkim metodama, ali u JavaScriptu se nazivaju ugrađeni objekti) sadrži matematičke metode:

  prvi = 4;

  drugi = 4;

  rezultat = Math.pow(prvi,drugi) – rezultat izvršavanja je $prvi^{drugi}$ tj. u ovom slučaju $4^4 = 256$;

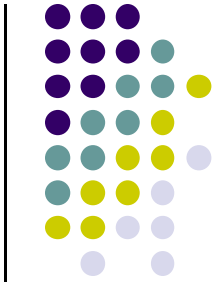
Predefinirani JavaScript objekti



- Ugrađeni objekt String služi za rad sa stringovima.
- Ovaj se objekt treba instanicirati ili korištenjem new ključne riječi ili pridruživanjem string vrijednosti varijabli:

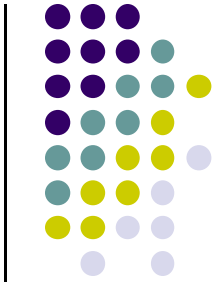
```
tekst1 = new String(“Ovo je neki string”);
tekst2 = “Ovo je drugi string”;
```

JavaScript vlastiti objekti



- Osim korištenja ugrađenih objekata, korisnik također može stvarati i vlastite objekte i to na sljedeća dva načina:
  1. stvaranjem direktne instance objekta korištenjem ugrađenog složenog tipa podataka Object.
  2. stvaranjem predloška objekt. Predložak je funkcija unutar koje je potrebno izvršiti pridruživanja sa *this.propertyName*. Ključna riječ “this” je instaca trenutnog objekta. Jednom kada postoji predložak, moguće je stvarati nove instance objekta.

Stvaranje direktne instance objekta



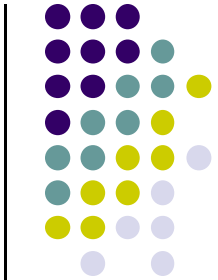
- Korištenje Object tipa:

```
osobaObj=new Object();
osobaObj.ime="John“;
osobaObj.prezime="Doe“;
osobaObj.dob=50;
osobaObj.boja_ociju="blue"
function jelo(a)
{ alert(“Ja jedem” + a);}
osobaObj.jelo=jelo; // povezivanje metode i objekta
```

- Pozivanje metode na objektu:

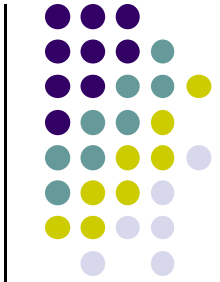
```
osobaObj.jelo(“ručak”);
```

Stvaranje predloška objekta



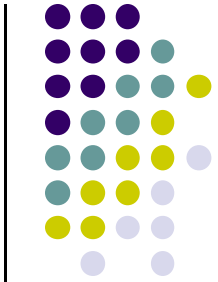
```
function osoba(ime,prezime,dob,boja_ociju)
{
this.ime=ime;
this.prezime=prezime;
this.dob=dob;
this.boja_ociju=boja_ociju;
}


myFather=new osoba("John","Doe",50,"blue");
myMother=new osoba("Sally","Rally",48,"green");
```

JavaScript DOM objekti



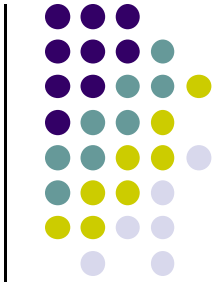
- JavaScript također uključuje podršku i za DOM kroz niz ugrađenih DOM JavaScript objekata poput:
  - Document
  - Body
  - Button
  - Table .....
- Svaki JavaScript DOM objekat odgovara elementu DOM stabla HTML dokumenta.

JavaScript browser objekti



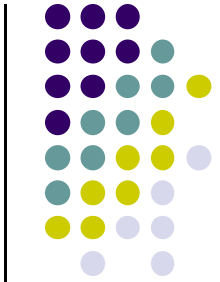
- JavaScript također uključuje podršku i za rad sa browserom kroz pet ugrađenih objekata:
  - Navigator
  - Screen
  - Window
  - Location
  - History

JavaScript browser objekti



- Objekt window predstavlja cijeli prozor browsera.
- Ako dokument otvoren u browseru sadrži frame-ove tada browser kreira za svaki frame jedan window objekt + window objekt za cijelu HTML stranicu koja uključuje okvire.
- Objekt window ima cijeli niz metoda i svojstava koji se mogu koristiti za rad s browserom.

JavaScript browser objekti

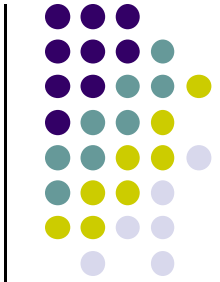


- Primjer otvaranja novog prozora korištenjem metode open na window objektu:

window.open(“http://www.google.hr”,“_blank”, “width=200,height=100”);

- Većina parametara ima predefinirane vrijednosti tako da ih ne morate postavljati.

JavaScript DOM objekti



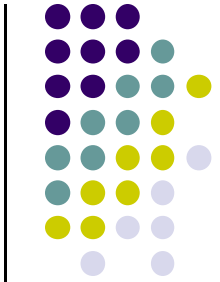
- Primjer mijenjanja boje pozadine stranice preko body objekta.
- Ne možete napisati samo:

```
body.bgColor = "green";
```

- Jer se svim DOM elementima treba pristupiti preko DOM stabla, stoga se uvijek kreće od document objekta i onda se dohvaćaju child elementi:

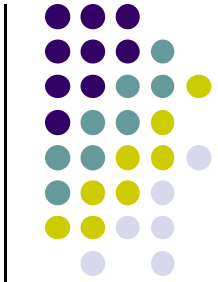
```
// ILI
tijelo = document.getElementsByTagName("body");
tijelo[0].bgColor = "red";
// ILI
document.body.bgColor = "blue";
```

JavaScript HTML DOM Event objekt



- Objekt Event definira događaj koji se desio na HTML DOM elementu.
- Ne podržavaju svi elementi sve događaje.
- Obično se događaji kombiniraju s nekom funkcionalnošću isprogramiranom u funkciji.
- Događaju se pristupa preko predefiniranih atributa koji označavaju određeni događaj (onclick, onkeypress,.....).

JavaScript HTML DOM Event objekt



- Primjer promjene sadržaja prvog *p* elementa u stranici kada korisnik klikne na botun u stranici (onclick event):

```
function funkcija()
{
    p_element = document.getElementById("prvi");
    p_element.innerHTML="Ovo je neki novi sadržaj p
    elementa";
}
<p id="prvi">Ovo je sadrzaj p elementa </p>
<button onclick="funkcija()">Klikni na me </button>
```

Poziv funkcije na događaj klika miša na botun