SKRIPTA ZA VEŽBE IZ PREDMETA

INTERNET TEHNOLOGIJE

JavaScript, DOM i XML

Laboratorija za elektronsko poslovanje

Beograd 2008.

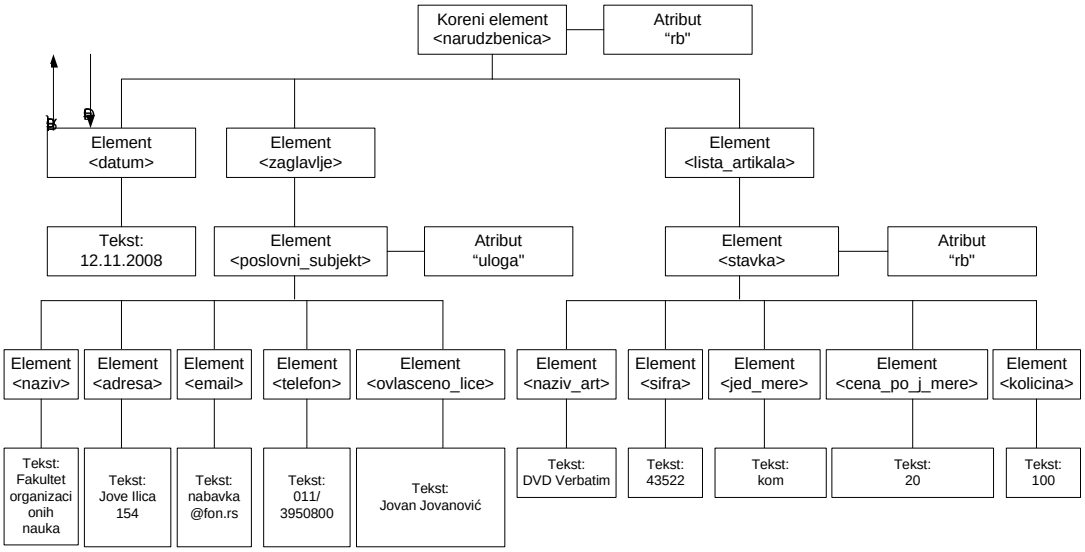
SADRŽAJ

1 XML DOM tutorial

1.1 Uvod u XML DOM

XML Document Object Model (XML DOM) definiše standard za pristup i manipulaciju XML dokumentima.

DOM posmatra XML dokument kroz strukturu stabla (stabla čvorova) sa elementima, atributima i tekstom definisanim kao čvorovi.



Slika 1 Primer XML DOM stable čvorova

1.2 Šta je DOM?

„W3C Document Object Model (DOM) je platformski i jezički neutralan interfejs koji dozvoljava programima i skriptovima da dinamički pristupaju i ažuriraju kontekst, strukturu i stil dokumenta.“

W3C DOM obezbeđuje standardan skup objekata za HTML i XML dokumente i standardan interfejs za pristup i manipulaciju ovakvim objektima.

W3C DOM je podeljen na različite delove (Core, XML i HTML) i različite nivoe (DOM nivo 1, 2 i 3):

- Core DOM – definiše standardni skup objekata za svaki strukturirani dokument
- XML DOM – definiše standardni skup objekata za XML dokumente
- HTML DOM – definiše standardni skup objekata za HTML dokumente

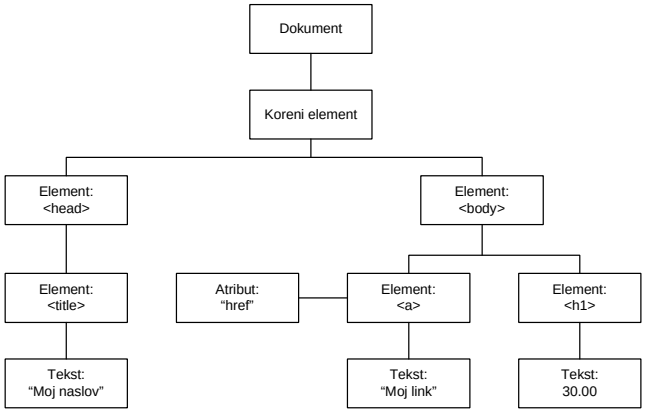
1998. godine W3C standard je objavio Nivo 1 DOM specifikaciju. Specifikacija je dozvoljavala da se pristupa i obrađuje svaki pojedinačni element na HTML strani.
Svi brauzeri su ubrzo prihvatili ovu preporuku, pa su, na taj način, nekompatibilnosti u DOM-u skoro potpuno nestale.

JavaScript može da koristi DOM kako bi se čitali i menjali HTML, XHTML i XML dokumenti.

1.3 Šta je HTML DOM?

HTML Document Object Model (HTML DOM) definiše standard za pristup i obradu HTML dokumenata.

DOM predstavlja HTML dokument u strukturi stabla (stabla čvorova) sa elementima, atributima i tekstom.



Slika 2 Primer HTML DOM stabla čvorova

1.4 W3C DOM specifikacije

Specifikacija	Svrha
DOM Level 0	Nije W3C standard. Definisan je kao ekvivalent onih funkcionalnosti koje se nalaze u Netscape Navigator 3.0 i Microsoft Internet Explorer-u 3.0.
DOM Level 1	Koncentriše se na HTML i XML modele dokumenata. Sadrži funkcionalnost navigacije i manipulacije dokumentom.
DOM Level 1 (SE)	
DOM Level 2 Core	Određuje API za pristup i ažuriranje sadržaja i strukture dokumenata. API takođe sadrži interfejs posvećen XML-u

DOM Level 2 HTML	Određuje API za obradu strukture i sadržaja HTML dokumenta
DOM Level 2 Views	Određuje API za dinamičan pristup i ažuriranje pogleda dokumenta. Pogled je alternativna prezentacija dokumenta.
DOM Level 2 Style	Određuje API za dinamičan pristup i ažuriranje stilova strana (style sheet)
DOM Level 2 Events	Određuje API za pristup događajima dokumenta
DOM Level 2 Traversal-Range	Određuje API za dinamičan prenos i identifikovanje opsega sadržaja u dokumentu
DOM Level 3	Određuje modele sadržaja (DTD i Shema) i validaciju dokumenta. Takođe određuje učitavanje i snimanje dokumenta, njegovo pregledanje, formatiranje i ključne događaje. Izgrađen je na DOM Core Level 2
DOM Level 3 Requirements	
DOM Level 3 Core	Određuje API za pristup i ažuriranje sadržaja, strukture i stila dokumenta
DOM Level 3 Events	
DOM Level 3 Load and Save	
DOM Level 3 Validation	
DOM Level 3 XPath	
DOM Level 3 Views	

1.5 Šta je XML DOM?

Ključne osobine XML DOM-a su:

- XML DOM je Document Object Model za XML
- platformski i jezički je nezavisan
- njime se definišu
- standardni skup objekata za XML
- standardni način pristupa XML dokumentima
- standardan način za manipulaciju XML dokumentima
- XML DOM je W3C standard

DOM posmatra XML dokumente kao strukturno stablo. Svim elementima, tekstu i atributima koje sadrže, može se pristupiti preko DOM stabla. Njihov sadržaj može da se modifikuje ili obriše, ili da se kreiraju novi elementi. Elementi, njihov tekst i atributi su pod jednim imenom poznati kao čvorovi.

1.6 XML čvorovi

1.6.1 Čvorovi

Prema DOM-u, sve u XML-u je čvor. Navodi se da:

- Čitav dokument je dokument čvor
- Svaki XML tag je element čvor
- Tekstovi koji se nalaze unutar XML dokumenta su tekst čvorovi
- Svaki XML atribut je atribut čvor
- Komentari su čvorovi komentari

1.6.2 Hijerarhija čvorova

Čvorovi imaju hijerarhijski odnos između sebe.

Svi čvorovi u XML dokumentu formiraju stablo dokumenta (ili stablo čvorova). Svaki element, atribut, tekst itd. u XML dokumentu predstavlja čvor u stablu. Stablo počinje čvorom dokumenta i nastavlja da se grana sve dok ne obuhvati sve tekstualne čvorove na najnižem nivou stabla.

Termini „roditelj“ (parent) i „dete“ (child) se koriste da bi opisali odnos između čvorova. Neki čvorovi mogu da imaju čvorove decu, dok drugi čvorovi nemaju decu (čvorovi listovi). Zato što je XML dokument strukturiran u formi stabla, može biti prenesen bez poznavanja tačne strukture stabla i bez poznavanja tipova koji su sadržani u njemu.

1.6.3 Primeri hijerarhije DOM čvorova

U narednom delu se daje pregled XML fajla: **narudzbenica.xml**

```
<?xml version="1.0 encoding = "UTF-8?>
<narudzbenica rb="1235">
      <datum>12.11.2008.</datum>
      <zaglavlje>
            <poslovni_subjekt uloga="narucilac">
                  <naziv>Fakultet organizacionih nauka</naziv>
                  <adresa>Jove Ilica 154</adresa>
                  <email>nabavka@fon.rs</email>
                  <telefon>011/3950800</telefon>
                  <ovlasceno_lice>Jovan Jovanovic</ovlasceno_lice>
            </poslovni_subjekt>
            <poslovni_subjekt uloga="dobavljac">
                  <naziv>Office 1 Superstore</naziv>
                  <adresa>YUBC, Bul. Mihajla Pupina 10b</adresa>
                  <email>yubc@office1.co.yu</email>
                  <telefon>011/3131003</telefon>
                  <ovlasceno_lice>Petar Petrovic</ovlasceno_lice>
            </poslovni_subjekt>
            <!--I narucilac i dobavljac predstavljaju poslovne subjekte
samo sa razlicitim ulogama u procesu nabavke-->
```

```
	</zaglavlje>
	<lista_artikala>
		<stavka rb="1">
			<naziv_art>DVD Verbatim</naziv_art>
			<sifra>43522</sifra>
			<jed_mere>kom</jed_mere>
			<cena_po_j_mere>20</cena_po_j_mere>
			<kolicina>100</kolicina>
		</stavka>
		<stavka rb="2">
			<naziv_art>Toner za ink-jet</naziv_art>
			<sifra>42057</sifra>
			<jed_mere>dl</jed_mere>
			<cena_po_j_mere>100</cena_po_j_mere>
			<kolicina>5</kolicina>
		</stavka>
		<stavka rb="3">
			<naziv_art>USB Kingston 4GB</naziv_art>
			<sifra>41862</sifra>
			<jed_mere>kom</jed_mere>
			<cena_po_j_mere>1000</cena_po_j_mere>
			<kolicina>3</kolicina>
		</stavka>
		<stavka rb="4">
			<naziv_art>Papir za štampač, 80g</naziv_art>
			<sifra>42789</sifra>
			<jed_mere>ris</jed_mere>
			<cena_po_j_mere>250</cena_po_j_mere>
			<kolicina>5</kolicina>
		</stavka>
	</lista_artikala>
</narudzbenica>
```

- Napominjemo da je koreni element u navedenom XML dokumentu nazvan <narudzbenica>. Svi drugi elementi koji se nalaze u dokumentu su sadržani unutar <narudzbenica>.
- Element <narudzbenica> predstavlja koreni element DOM stabla.
- Koreni element <narudzbenica> sadrži tiri čvora (deteta) <datum>, <zaglavlje> i <lista_artikala>. <narudzbenica> sadrži i atribut čvor rb="1235".
- Dete čvor <datum> sadrži tekstualni čvor „12.11.2008“.

Dete čvor <zaglavlje> sadrži svoja dva čvora deteta <poslovni_subjekt> sa po pet čvorova dece i jedim atribut čvorom *uloga*. <zaglavlje> sadrži i čvor komentar <!--I narucilac i dobavljac predstavljaju poslovne subjekte samo sa razlicitim ulogama u procesu nabavke-->.

Dete čvor <lista_artikala> sadrži četiri čvora deteta <stavka> sa po pet čvorova dece i jedim atribut čvorom rbs.

VAŽNO! Tekst je uvek sačuvan u tekst čvoru. Česta greška u DOM obradi je da kada se dođe do određenog čvora očekuje se da sadrži tekst. Međutim, čak i najjednostavniji element čvor ima tekst unutar sebe. Na primer, u <sifra>43522</sifra>, postoji element čvor (<sifra>) i tekst čvor unutar njega, koji sadrži tekst (43522).

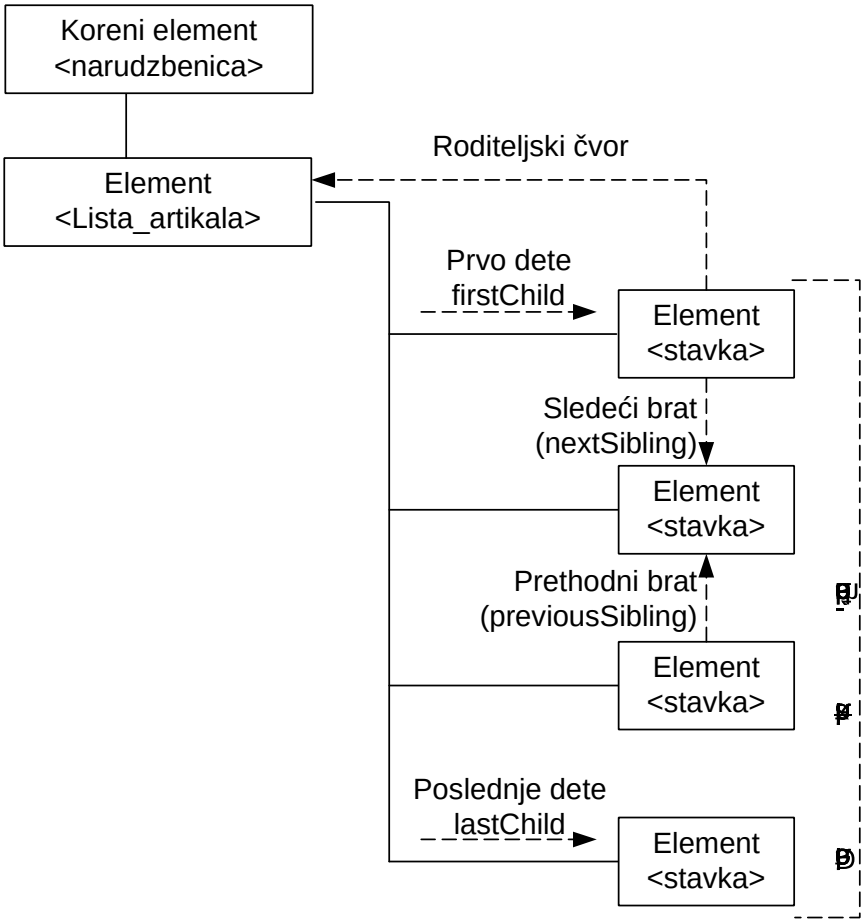
Slika 1 (iz odeljka *Uvod u XML DOM*) ilustruje fragment DOM stabla čvorova iz prethodno navedenog XML dokumenta.

1.7 XML DOM stablo čvorova

Stablo čvorova prikazuje XML dokument kao skup čvorova i njihovih međusobnih veza. Prilikom formiranja stabla čvorova, poštuju se sledeća pravila:

- U stablu čvorova, prvi čvor (čvor na vrhu) se naziva koren
- Svaki čvor, osim korena, ima tačno jedan roditeljski čvor
- Čvor može da ima proizvoljan broj dece
- List je čvor koji nema dece
- Braća su čvorovi sa istim roditeljem

Sledeća slika ilustruje deo stabla čvorova i veza koje postoje između čvorova u navedenom XML fajlu: narudzbenica.xml



Slika 3 Primer stable čvorova I odnosa koji postoje između čvorova

1.8 XML DOM pristup čvorovima

Koristeći DOM, može da se pristupi svakom čvoru u XML dokumentu.

Moguće je pronaći čvor koji želite da obradite na nekoliko načina:

- Korišćenjem getElementsByTagName() metode
- Korišćenjem svojstva parentNode (roditeljski čvor), firstChild (prvo dete) i lastChild (poslednje dete) element čvora

1.8.1 getElementsByTagName()

Metoda *getElementsByTagName()* može da pronađe bilo koji XML element u celom dokumentu.

Ova metoda zanemaruje strukturu dokumenta. Ukoliko je neophodno da se pronađu svi elementi <stavka> u dokumentu, metoda *getElementsByTagName()* će ih sve i pronaći, bez obzira na kojem nivou se element <stavka> nalazi.

Metoda *getElementsByTagName()* vraća sve elemente (kao *nodeList* tj. listu čvorova) sa navedenim imenom taga koji je predak elementa na kome se nalazite kada se koristi ova metoda.

Metoda *getElementsByTagName()* može da se koristi nad bilo kojim XML elementom.

getElementsByTagName() sintaksa

```
getElementsByTagName("imetaga");
```

Naredni primer vraća listu čvorova (nodeList) svih <stavka> elemenata u dokumentu:

```
var x=xmlDoc.getElementsByTagName("stavka");
```

1.8.2 nodeList

Kada se koristi *nodeList*, obično se lista čuva u nekoj varijabli kao na primer:

```
var x=xmlDoc.getElementsByTagName("stavka");
```

Sada varijabla x sadrži listu svih <stavka> elemenata na strani i može im se pristupati na osnovu njihovih indeksa.

Napomena: Indeks započinje od 0.

Može se proći kroz celu listu čvorova korišćenjem svojstva *length* (dužina):

```
var x=xmlDoc.getElementsByTagName("stavka");
for (var i=0;i<x.length;i++)
  {
  // uradi nešto sa svakim elementom <stavka>
  }
```

Takođe može da se pristupi određenom elementu korišćenjem indeksa.

Da bi se pristupilo trećem <p> može da se uradi sledeće:

```
var y=x[2];
```

1.8.3 parentNode, firstChild, and lastChild

Tri svojstva *parentNode*, *firstChild* i *lastChild* prate strukturu dokumenta i omogućuju prelaz na kratkim relacijama unutar dokumenta.

Dat je naredni XML fragment:

```
<lista_artikala>
	<stavka rbs="1">
		<naziv_artikla>DVD Verbatim</naziv_artikla>
		<sifra>43522</sifra>
		<jed_mere>kom</jed_mere>
		<cena_po_j_mere>20</cena_po_j_mere>
		<kolicina>100</kolicina>
	</stavka>
</lista_artikala>
```

U navedenom XML kodu, element <naziv_artikla> je *firstChild* elementa <stavka>, a element <kolicina> je *lastChild* elementa <stavka>.

Nadalje, element <stavka> je parentNode elemenata <naziv_artikla>, <sifra>, <jed_mere>, <cena_po_j_mere> i <kolicina>.

1.8.4 Koren čvorova document.documentElement

Postoji jedno posebno svojstvo dokumenta koje omogućuje pristup tagovima:

document.documentElement

Ovo svojstvo vraća koreni čvor dokumenta i postoji u svim XML i HTML dokumentima.

1.9 XML DOM informacije o čvorovima

Svaki čvor poseduje određena svojstva koja sadrže neke informacije o čvoru. Svojstva su:

- nodeName
- nodeValue
- nodeType

1.9.1 nodeName

Svojstvo *nodeName* sadrži ime čvora.

- *nodeName* element čvora je naziv taga
- *nodeName* atribut čvora je naziv atributa
- *nodeName* tekst čvora je uvek tekst
- *nodeName* dokument čvora je uvek document
- *nodeName* komenar čvora je uvek comment

Napomena: *nodeName* uvek sadrži naziv taga XML elementa i ispisan je velikim slovima.

1.9.2 nodeValue

Za tekst čvorove svojstvo *nodeValue* sadrži tekst.
Za atribut čvorove, svojstvo *nodeValue* sadrži vrednost atributa.
Za komentar čvorove, svojstvo *nodeValue* sadrži tekst komentara.
Svojstvo *nodeValue* ne postoji za dokument i element čvorove.

1.9.3 nodeType

Svojstvo *nodeType* vraća tip čvora.

Najvažniji tipovi čvorova su:

Tip elementa	nodeName	nodeValue	Tip čvora
Element	naziv taga	-	1
Atribut	naziv atributa	vrednost atributa	2
Tekst	#text	tekst čvora	3
Komentar	#comment	tekst komentara	8
Dokument	#document	-	9

1.10 XML DOM lista čvorova i NamedNodeMap

1.10.1 DOM lista čvorova

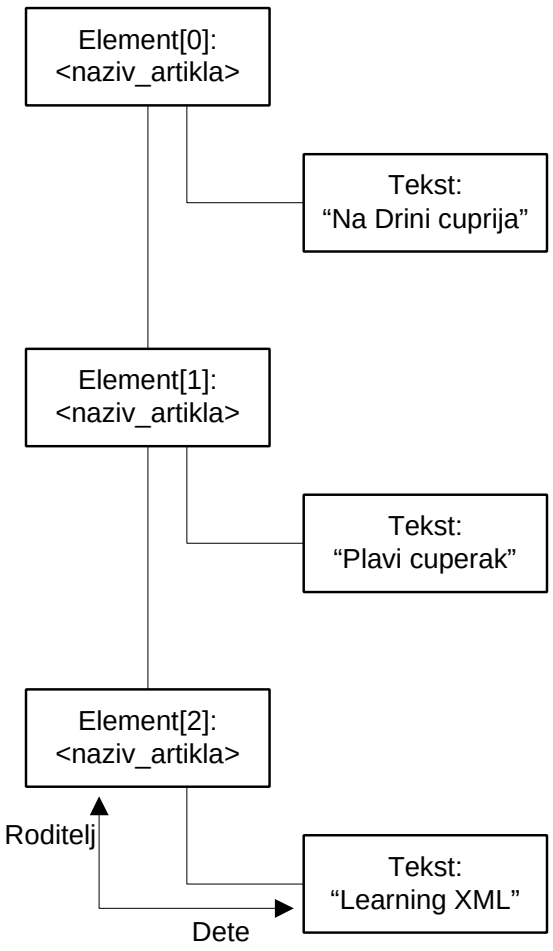
Kada se koriste svojstva ili metode kao što je *childNodes* ili *getElementsByTagName()*, vrednost koja se vrati je *NodeList* objekat.

NodeList objekat predstavlja uređenu listu čvorova.

Čvorovima je moguće pristupiti preko njihovog indeksa (započinje od 0).

Napomena: Unutar objekta *NodeList* čvorovi se nalaze u onom redosledu u kojem se oni nalaze u XML dokumentu.

Sada ćemo da kreiramo listu čvorova svih <naziv_artikla> elemenata koji se nalaze u "narudzbenica.xml", korišćenjem metode *getElementsByTagName*("sifra"). Sledeća slika predstavlja dobijenu listu čvorova:



Slika 4Primer liste čvorova

Sledeći fragment koda uzima tekst iz prvog <naziv_artikla> elementa:

```
getElementsByTagName("naziv_artikla")[0].childNodes[0].nodeValue
```

Izlaz:

```
DVD Verbatim
```

Određivanje dužine liste čvorova

Lista čvorova se održava ažurnom. Ako je neki element izbrisan ili dodat u listu čvorova XML dokumenta, lista se automatski ažurira.

Sledeći fragment koda uzima broj elemenata <naziv_artikla> iz "narudzbenica.xml":

```
getElementsByTagName("naziv_artikla").length
```

Izlaz:

```
4
```

Kada je poznata dužina liste čvorova, jednostavno se može proći kroz nju i uzeti vrednosti koje su potrebne.

Naredni fragment koda prolazi kroz sve elemente <naziv_artikla> i prikazuje njihove vrednosti:

```
//varijabla x će da sadrži NodeList
var x=getElementsByTagName("naziv_artikla")
for (i=0;i<x.length;i++)
  {
  document.write(x[i].childNodes[0].nodeValue)
  document.write("<br />")
  }
```

Izlaz:

```
DVD Verbatim
Toner za ink-jet, crni
USB Kingston 4GB
Papir za štampac, 80g
```

1.10.2 DOM NamedNodeMap

Kada se koriste svojstva atributa nad određenim elementom, vrednost koja se vraća je *NamedNodeMap* objekat.

NamedNodeMap objekat predstavlja neuređenu listu atribut čvorova. Čvorovima koji se nalaze u *NamedNodeMap* moguće je pristupiti preko njihovog imena.

Napomena: U objektu *NamedNodeMap* čvorovi se ne nalaze u nekom utvrđenom redosledu.

Određivanje dužine *NamedNodeMap* objekta

NamedNodeMap se održava ažurnim. Ako je neki element izbrisan ili dodat u listu čvorova XML dokumenta, lista se automatski ažurira.

NamedNodeMap takođe poseduje svojstvo *length* (dužina). Svojstvo *length* vraća broj čvorova u listi.

Sledeći fragment koda uzima broj atributa iz prvog elementa <naziv_artikla> iz “narudzbenica.xml“.

```
getElementsByTagName("stavka")[0].attributes.length
```

Izlaz:

```
1
```

1.10.3 Određivanje stavki objekta *NamedNodeMap*

Metoda *getNamedItem()* objekta *NamedNodeMap* može da se koristi da bi se pretražio određeni čvor.

Sledeći fragment koda prikazuje kako da se izlistaju vrednosti “kategorija“ atributa za svaki element <stavka>.

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("stavka");
for(i=0;i<x.length;i++)
  {
  //varijabla attlist će da sadrži NamedNodeMap
  var attlist=x.item(i).attributes;
  var att=attlist.getNamedItem("rb");
  document.write(att.value + "<br />")
  }
```

Izlaz:

```
1
2
3
4
```

1.11 Parsiranje XML DOM-a

Da bi se čitao, ažurirao, kreirao ili manipulisalo XML dokumentom, potreban je XML parser.

1.11.1 Parsiranje XML DOM-a

Da bi se obrađivao XML dokument, potreban je XML parser. Parser učita dokument u memoriju računara. Kada je dokument učitan, podaci koji se u njemu nalaze mogu da budu obrađivani korišćenjem DOM-a. DOM tretira XML dokument kao stablo.
Postoje neke razlike između Microsoft-ovog XML parsera i XML parsera koji se koristi u Mozilla čitačima. U ovom tutorialu će biti pokazano kako se prave skriptovi nezavisni od čitača, koji mogu da rade i u Internet Explorer-u i u Mozilla čitačima.

1.11.2 Microsoft-ov XML parser

Microsoft-ov XML parser je COM komponenta koja se pojavila sa Internet Explorer-om 5 i novijim. Kada se jednom instalira Internet Explorer, parser je dostupan skriptovima.
Mircosoft-ov XML parser podržava sve neophodne funkcije za prevođenje stabla čvorova, pristup čvorovima i vrednostima njihovih atributa, unos i brisanje čvorova i ponovnog prevođenja stabla čvorova nazad u XML.

Da bi se kreirala instanca Microsoft XML parsera, koristi se sledeći kod:

JavaScript:

```
var xmlDoc=new ActiveXObject("Microsoft.XMLDOM");
```

VBScript:

```
set xmlDoc=CreateObject("Microsoft.XMLDOM")
```

ASP:

```
set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
```

Sledeći fragment koda učitava postojeći XML dokument (“beleska.xml“) u Microsoft XML parser:

```
var xmlDoc=new ActiveXObject("Microsoft.XMLDOM");
xmlDoc.async="false";
xmlDoc.load("beleska.xml");
```

Prva linija prethodnog skripta kreira instancu XML parsera. Druga linija poništava asinhrono učitavanje, da bi se zasiguralo da parser neće nastaviti izvršavanje a da prethodno dokument još uvek nije potpuno

učitan. Treća linija govori parseru da učita XML dokument sa imenom “beleska.xml“.

1.11.3 XML parseri u Mozilla-i, Firefox-u i Opera-i

Mozilla-in XML parser podržava sve neophodne funkcije za prevođenje stabla čvorova, pristup čvorovima i vrednostima njihovih atributa, unos i brisanje čvorova i ponovnog prevođenja stabla čvorova nazad u XML.

Da bi se kreirala instanca XML parsera u Mozilla čitačima koristi se naredni kod:

JavaScript:

```
var xmlDoc=document.implementation.createDocument("ns","root",null);
```

Prvi parametar, *ns*, definiše opseg imena (namespace) koji se koristi za XML dokument. Drugi parametar, *root*, je XML koreni element u XML fajlu. Treći parametar, null, je uvek null zato što u ovom trenutku još nije implementiran.

Kod fragment koji sledi učitava XML dokument (“beleska.xml“) u Mozilla XML parser:

```
var xmlDoc=document.implementation.createDocument("","",null);
xmlDoc.load("beleska.xml");
```

Prva linija prethodnog skripta kreira instancu XML parsera. Druga linija govori parseru da učita XML dokument sa imenom “beleska.xml“.

Parsiranje XML fajla – cross-browser primer

Sledeći primer je cross-browser primer koji učitava postojeći XML dokument (“beleska.xml“) u XML parser:

```
<html>
<head>
<script type="text/javascript">
var xmlDoc;
function loadXML()
{
// kod za IE
if (window.ActiveXObject)
  {
  xmlDoc=new ActiveXObject("Microsoft.XMLDOM");
  xmlDoc.async=false;
  xmlDoc.load("beleska.xml");
  getmessage();
  }
// kod za Mozilla-u, Firefox, Opera-u, itd.
else if (document.implementation &&
document.implementation.createDocument)
```

```
  {
  xmlDoc=document.implementation.createDocument("","",null);
  xmlDoc.load("beleska.xml");
  xmlDoc.onload=getmessage;
  }
else
  {
  alert("Vas browser ne moze da obradi ovaj skript");
  }
}
function getmessage()
{
document.getElementById("za").innerHTML=
xmlDoc.getElementsByTagName("za")[0].childNodes[0].nodeValue;
document.getElementById("od").innerHTML=
xmlDoc.getElementsByTagName("od")[0].childNodes[0].nodeValue;
document.getElementById("poruka").innerHTML=
xmlDoc.getElementsByTagName("telo")[0].childNodes[0].nodeValue;
}
</script>
</head>
<body onload="loadXML()">
<h1> Beleska</h1>
<p><b>Za:</b> <span id="za"></span><br />
<b>Od:</b> <span id="od"></span><br />
<b>Poruka:</b> <span id="poruka"></span>
</p>
</body>
</html>
```

Izlaz:

```
Beleska
Za:Slavicu
Od:Dušana
Poruka: Molim te okači ovo uputstvo na sajt!
```

Važna napomena

Da biste izvukli tekst (Dušana) iz XML elementa <od>Dušana</od>, ispravna sintaksa je:

```
getElementsByTagName("od")[0].childNodes[0].nodeValue
```

VAŽNO: *getElementsByTagName* vraća niz čvorova. Niz sadrži sve elemente sa određenim imenom unutar XML dokumenta. U ovom slučaju postoji samo jedan “od“ element, ali se ipak može navesti broj indeksa u nizu ([0]).

1.11.4 Parsiranje XML stringa – cross-browser primer

Sledeći kod je coss-browser primer o tome kako se učitava i parsira XML string:

```
<html>
<body>
<script type="text/javascript">
var text="<beleska>";
text=text+"<za> Slavicu </za>";
text=text+"<od> Dušana </od>";
text=text+"<zaglavlje>Podsetnik</zaglavlje>";
text=text+"<telo> Molim te okači ovo uputstvo na sajt!</telo>";
text=text+"</beleska>";
// kod za IE
if (window.ActiveXObject)
  {
  var doc=new ActiveXObject("Microsoft.XMLDOM");
  doc.async="false";
  doc.loadXML(text);
  }
// kod za Mozilla, Firefox, Opera, etc.
else
  {
  var parser=new DOMParser();
  var doc=parser.parseFromString(text,"text/xml");
  }
// documentElement uvek predstavlja koreni čvor
var x=doc.documentElement;
document.write("Tekst prvog deteta elementa: ");
document.write(x.childNodes[0].childNodes[0].nodeValue);
document.write("<br />");
document.write("Tekst drugog deteta elementa: ");
document.write(x.childNodes[1].childNodes[0].nodeValue);
</script>
</body>
</html>
```

Izlaz:

```
Tekst prvog deteta elementa: Slavicu
Tekst drugog deteta elementa: Dušana
```

Napomena: Internet Explorer metod loadXML() parsira XML string, dok Mozilla browser-i koriste DOMParser objekat.

1.12 XML DOM prevođenje stabla čvorova

1.12.1 Prevođenje stabla čvorova

Često je potrebno da se prođe kroz elemente nekog XML dokumenta ili stringa.

Primer koji sledi prolazi kroz svu decu čvorove od čvora <beleska>, i štampa ime i vrednost čvora za svaki čvor:

```
<html>
<body>
<script type="text/javascript">
var text="<beleska>";
text=text+"<za> Slavicu </za>";
text=text+"<od> Dušana </od>";
text=text+"<zaglavlje>Podsetnik</zaglavlje>";
text=text+"<telo> Molim te okači ovo uputstvo na sajt!</telo>";
text=text+"</beleska>";
// kod za IE
if (window.ActiveXObject)
  {
  var doc=new ActiveXObject("Microsoft.XMLDOM");
  doc.async="false";
  doc.loadXML(text);
  }
// kod za Mozilla, Firefox, Opera, itd.
else
  {
  var parser=new DOMParser();
  var doc=parser.parseFromString(text,"text/xml");
  }
// documentElement uvek predstavlja koreni čvor
var x=doc.documentElement;
for (i=0;i<x.childNodes.length;i++)
  {
  document.write(x.childNodes[i].nodeName);
  document.write("=");
  document.write(x.childNodes[i].childNodes[0].nodeValue);
  document.write("<br />");
  }
</script>
</body>
</html>
```

Izlaz:

```
za=Slavicu
od=Dušana
zaglavlje=Podsetnik
telo= Molim te okači ovo uputstvo na sajt!
```

1.13 XML DOM Mozilla vs. Internet Explorer

1.13.1 Razlike u browser-ima prilikom DOM parsiranja

I Mozilla i Internet Explorer podržavaju W3C DOM specifikaciju.

Međutim, ipak postoje razlike između Internet Explorer i Mozilla DOM-a. Najvažnija razlika je u tome kako se obrađuju blanko karakteri u tekst čvorovima. Kada se generiše XML, on često sadrži blanko karaktere između čvorova. Internet Explorer, kada koristi *node.childNodes[]*, neće sadržati ove blanko čvorove. U Mozilla-i ovi čvorovi biće u nizu koji se vraća kao povratna vrednost.

Sledeći fragment koda upozorava koliko čvorova dece ima koreni element:

```
xmlDoc.load("narudzbenica.xml");
var x=xmlDoc.documentElement.childNodes;
alert(x.length)
for (i=0;i<x.length;i++)
  {
  document.write(x[i].nodeType);
  document.write("<br />");
  }
```

Internet Explorer će da preskoči blanko tekst čvorove koji su generisani između čvorova (na primer karaktere koji označavaju novi red), dok Mozilla neće. Tako da će, u prethodnom primeru Mozilla prijaviti 7 čvorova dece, dok će Internet Explorer prijaviti 3.

Da bi se prošlo kroz čvorove decu i izbegli ovi tekst čvorovi, može se proveriti tip čvora.

1.14 XML DOM navigacija čvorova

1.14.1 Navigacija DOM čvorova

Navigaciju nad čvorovima možemo da izvedemo tako što koristimo veze koje postoje među njima:

- parentNode
- childNode
- firstNode
- lastNode
- nextSibling
- previousSibling

Korišćen XML fajl: narudzbenica.xml

Skrećemo pažnju još jednom na sliku 3 (iz odeljka *XML DOM stablo čvorova*).

Napomena: Internet Explorer će da preskoči blanko tekst čvorove koji su generisani između čvorova (na primer karakteri koji označavaju početak nove linije), dok Mozilla neće. Zato, u primeru koji sledi, imaćemo funkciju koja proverava tip čvora kada se koristi *firstChild*, *lastChild*, *nextSibling*, i *previousSibling*.

1.14.2 Uzimanje prvog deteta čvoru

Sledeći fragment koda uzima prvo dete čvora <narudzbenica>:

```
//proverava da li je prvi čvor element čvor
function get_firstChild(n)
  {
  var x=n.firstChild;
  while (x.nodeType!=1)
    {
    x=x.nextSibling;
    }
  return x;
  }
xmlDoc=loadXMLDoc("narudzbenica.xml");
var y=get_firstChild(xmlDoc.documentElement);
document.write(y.nodeName);
```

Rezultat prethodnog koda bi bio:

```
datum
```

Funkcija u prethodnom primeru proverava tip čvora prvog deteta čvora.

Element čvorovi imaju *nodeType* 1, tako da ako prvi dete čvor nije element čvor, pomera se do sledećeg čvora i proverava da li je ovaj čvor element čvor. Ovo se nastavlja sve dok prvi dete čvor (koji mora da bude element čvor) ne bude pronađen. Na ovaj način, rezultat će biti prepravljen i u Internet Explorer-u i u Mozilla-i.

1.14.3 Uzimanje prethodnog brata čvora

Sledeći kod fragment uzima prethodnog brata od prvog elementa <autor>:

```
//Proverava da li je prethodni brat ujedno element čvor
function get_previoussibling(n)
  {
  var x=n.previousSibling;
  while (x.nodeType!=1)
    {
    x=x.previousSibling;
    }
  return x;
  }
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("sifra")[0];
var y=get_previoussibling(x);
document.write(y.nodeName);
```

Rezultat prethodnog koda biće:

```
naziv_artikla
```

Funkcija u prethodnom primeru proverava tip čvora prethodnog brata.

Ako prethodni brat nije element čvor, onda se pomera do “narednog“ prethodnog brata i proverava da li je čvor element čvor. Ovo se nastavlja sve dok prethodni brat (koji mora da bude element čvor) ne pronađe. Na ovaj način, rezultat će biti popravljen i u Internet Explorer-u i u Mozilla-i.

2 XML DOM obrada čvorova

2.1 XML DOM uzimanje čvorova

2.1.1 Uzimanje vrednost elementa

Metoda *getElementsTagname()* vraća listu čvorova koji sadrže elemente sa određenim imenom taga u istom redosledu u kom se nalaze u izvornom dokumentu.

Sledeći fragment koda štampa vrednosti elemenata “ naziv_artikla“ iz “narudzbenica.xml“.

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("naziv_artikla");
for (i=0;i<x.length;i++)
  {
  document.write(x[i].childNodes[0].nodeValue)
  document.write("<br />")
  }
```

Izlaz:

```
DVD Verbatim
Toner za ink-jet, crni
USB Kingston 4GB
Papir za štampac, 80g
```

2.1.2 Uzimanje vrednosti atributa

Metoda *getAttribute()* može da se koristi da bi se prikazala vrednost atributa.

Naredni fragment koda štampa vrednosti svih atributa “uloga“ iz “narudzbenica.xml“:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("poslovni_subjekt");
for (i=0;i<x.length;i++)
{
document.write(x[i].getAttribute("uloga"));
document.write("<br />");
}
```

Izlaz:

```
narucilac
dobavljac
```

2.1.3 Uzimanje vrednosti stavke

Metoda *getNamedItem()* može da se koristi da bi se pretražio određeni čvor.

Naredni fragment koda prikazuje kako je moguće odštampati vrednosti atributa “uloga“ iz svakog elementa <poslovni_subjekt>:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("poslovni_subjekt");
for(i=0;i<x.length;i++)
  {
  var attlist=x.item(i).attributes;
  var att=attlist.getNamedItem("uloga");
  document.write(att.value + "<br />")
  }
```

Izlaz:

```
narucilac
dobavljac
```

2.2 XML DOM postavljanje čvorova

2.2.1 Primeri

U narednim primerima koristiće se XML fajl narudzbenica.xml i JavaScript funkcija loadXMLDoc().

Postavljanje novog atributa i vrednosti atributa

Metoda

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("naziv_artikla");
for(i=0;i<x.length;i++)
  {
  x.item(i).setAttribute("boja","plava");
  }
```

Drugi način za kreiranje novog atributa

Metoda *createAttribute()* se koristi da bi se kreirao novi čvor atribut.

Naredni fragment koda koristi *createAttribute()* da bi kreirao novi čvor atribut i *setAttribute()* da bi ga dodao u element:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("naziv_artikla");
var noviAtribut;
for (i=0;i<x.length;i++)
  {
  noviAtribut=xmlDoc.createAttribute("boja");
  noviAtribut.value="plava";
  x[i].setAttributeNode(noviAtribut);
  }
```

2.2.2 Promena vrednosti atributa

Metoda *setAttribute()* se može koristiti da bi se promenila vrednost postojećeg atributa, ili da bi se kreirao novi atribut/vrednost atributa za neki element.

Naredni fragment koda menja vrednost postojećeg “ketegorija“ atributa (u svakom <stavka> elementu):

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("stavka");
for(i=0;i<x.length;i++)
  {
  x.item(i).setAttribute("boja","crvena");
  }
```

2.2.3 Promena vrednosti stavke

Metoda *getNamedItem()* može da se koristi da bi se promenila vrednost postojeće stavke.

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("stavka");
for(i=0;i<x.length;i++)
  {
  var atr=x.item(i).attributes.getNamedItem("boja");
  atr.value="crvena";
  }
```

2.3 XML DOM uklanjanje čvorova

2.3.1 Uklanjanje elementa

Metoda *removeChild()* može da se koristi da bi se uklonio određeni čvor.

Naredni fragment koda će da ukloni poslednji element <stavka> iz učitanog XML-a:

```
//proverava da li je poslednje dete element čvor
function get_lastchild(n)
{
var x=n.lastChild;
while (x.nodeType!=1)
  {
  x=x.previousSibling;
  }
return x;
}
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.documentElement;
x.removeChild(get_lastchild(x));
```

Napomena: Internet Explorer izostavlja blanko tekst čvorove koji se generišu između čvorova (na primer karakteri za označavanje novog reda), dok Mozilla ne izostavlja. Tako da, u prethodnom primeru, funkcija *get_lastchild()* proverava tip čvora poslednjeg deteta zadatog parametra.

Element čvor ima *nodeType* 1, tako da ako poslednje dete čvora koji je zadat kao parametar nije element čvor, pomera se prema prethodnom čvoru i proverava da li je taj čvor element čvor. To se nastavlja sve dok poslednje dete čvor (koje mora da bude element čvor) ne bude pronađeno. Na ovaj način, rezultat će da bude ispravljen i u Internet Explorer-u i u Mozilla-i.

2.3.2 Uklanjanje teksta iz elementa

Metoda *deleteData()* se koristi da bi se uklonili podaci iz tekst čvora.

Metoda *deleteData()* ima dva parametra:

- *offset* – gde da otpočne uklanjanje karaktera. *Offset* vrednost počinje od nule
- *count* – koliko karaktera da bude obrisano

Naredni fragment koda će da ukloni prvih devet karaktera iz prvog elementa <naziv_artikla> koji se nalazi u učitanom XML-u:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("stavka")[0].childNodes[0];
x.deleteData(0,9);
```

2.3.3 removeAttributeNode()

Metoda *removeAttributeNode()* se koristi da bi se uklonio atribut čvor.

Naredni fragment koda će da ukloni sve atribute “boja“ iz svakog elementa <stavka>:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("stavka");
for(i=0;i<x.length;i++)
  {
  atrCvor=x.item(i).getAttributeNode("boja")
  stari_atr=x.item(i).removeAttributeNode(atrCvor);
  document.write("Uklonjen atribut: " + stari_atr.name + "<br />");
  }
```

Izlaz:

```
Uklonjen atribut: boja
Uklonjen atribut: boja
Uklonjen atribut: boja
Uklonjen atribut: boja
```

2.4 XML DOM zamena čvorova

2.4.1 Zamena čvora u listi čvorova

Metoda *replaceChild()* se koristi da bi se zamenio čvor u listi čvorova.

Naredni fragment koda kreira novi element <stavka> koji će zameniti poslednji element <stavka>:

```
//proverava da li je poslednje dete element čvor
function get_lastchild(n)
{
```

```
var x=n.lastChild;
while (x.nodeType!=1)
  {
  x=x.previousSibling;
  }
return x;
}

xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.documentElement;
//kreira element stavka, element naziv_artikla i tekst čvor
var noviCvor=xmlDoc.createElement("stavka");
var noviNaziv=xmlDoc.createElement("naziv_artikla");
var noviTekst=xmlDoc.createTextNode("SD kartica 2GB");
//dodaje tekst čvor čvoru naziv_artikla,
//i dodaje čvor naziv_artikla čvoru stavka
noviNaziv.appendChild(noviTekst);
noviCvor.appendChild(noviNaziv);
//menja posledji čvor novim čvorom
x.replaceChild(noviCvor,get_lastchild(x));
```

Napomena: Internet Explorer izostavlja blanko tekst čvorove koji se generišu između čvorova (na primer karakteri za označavanje novog reda), dok Mozilla ne izostavlja. Tako da, u prethodnom primeru, funkcija *get_lastchild()* proverava tip čvora poslednjeg deteta zadatog parametra.

Element čvor ima *nodeType* 1, tako da ako poslednje dete čvora koji je zadat kao parametar nije element čvor, pomera se prema prethodnom čvoru i proverava da li je taj čvor element čvor. To se nastavlja sve dok poslednje dete čvor (koje mora da bude element čvor) ne bude pronađeno. Na ovaj način, rezultat će da bude ispravljen i u Internet Explorer-u i u Mozilla-i.

2.4.2 Zamena teksta u tekst čvoru

Metoda *replaceData()* se koristi da bi se zamenili podaci iz tekst čvora.

Metoda *replaceData()* ima tri parametra:

- *offset* – gde da otpočne uklanjanje karaktera. Offset vrednost počinje od nule
- *length* – koliko karaktera da bude zamenjeno
- *string* – string koji će biti dodat

Naredni fragment koda će da zameni prvih pet karaktera iz tekst čvora koji je u elementu <adresa> tekstom “Na Moravi“:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("adresa")[0].childNodes[0];
x.replaceData(0,5,"p.fah");
```

2.5 XML DOM kreiranje čvorova

2.5.1 Kreiranje elementa

Metoda *createElement()* kreira novi element čvor.

Naredni fragment koda kreira element (<boja>) i dodaje ga posle poslednjeg deteta svakog <stavka> elementa:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("stavka");
var noviElement
for (i=0;i<x.length;i++)
  {
  noviElement=xmlDoc.createElement("boja");
  x[i].appendChild(noviElement);
  }
```

2.5.2 Kreiranje atributa

Da bi se kreiralo novi atribut čvor koristiće se *createAttirbute().*

Naredni fragment koda kreira atribut “boja“ i dodaje ga svim elementima <stavka>:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("stavka");
var noviAtribut;
for (i=0;i<x.length;i++)
  {
  noviAtribut=xmlDoc.createAttribute("boja");
  noviAtribut.value="plava";
  x[i].setAttributeNode(noviAtribut);
  }
```

2.5.3 Kreiranje tekst čvora

Metoda *createText()* kreira novi tekst čvor.

Naredni fragment koda kreira element (<boja>), sa tekst čvorom ("plava") u njemu i dodaje ga nakon poslednjeg deteta svakom element čvoru <stavka>:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("stavka");
var noviElement,noviTekst
for (i=0;i<x.length;i++)
  {
  noviElement=xmlDoc.createElement("boja");
  noviTekst=xmlDoc.createTextNode("plava");
  noviElement.appendChild(noviTekst);
```

```
  x[i].appendChild(noviElement);  }
```

2.5.4 Kreiranje CDATA sekcija čvora

CDATA sekcija je deo teksta koji XML parser ne parsira. Time je omogućen unos specijalnih karaktera poput „<“ ili „&“, koji su nelegalni unutar tekst čvorova, jer „<“ označava početak taga a „&“ početak specijalnog karaktera.

CDATA sekcija počinje sa "<![CDATA[" i završava "]]>":

Primer unosa JavaScript funkcije u čvor:

```
<script>
<![CDATA[
function matchwo(a,b){
if (a < b && a < 0) then{
        return 1;
}else{
        return 0;}
}
]]>
</script>
```

CDATA sekcija ne može da sadrži string "]]>". Ugnježdene CDATA sekcije nisu dozvoljene. Oznaka "]]>" kraja CDATA sekcije ne sme sadržati prazne karaktere ili karaktere za prelom linije.

Metoda *createCDATASection()* kreira novi CDATA sekcija čvor.

Naredni fragment koda kreira CDATA sekciju i dodaje je nakon poslednjeg deteta svakom <stavka> elementu:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("stavka");
var newCDATA,noviTekst;
noviTekst="U slučaju da artikla nema na stanju, molimo vas, kontaktirajte
nas";
for (i=0;i<x.length;i++)
  {
  novaCDATA=xmlDoc.createCDATASection(noviTekst);
  x[i].appendChild(novaCDATA);
  }
```

2.5.5 Kreiranje komentar čvora

Metoda *createComment()* kreira novi komentar čvor.

Naredni fragment koda kreira komentar čvor i dodaje ga nakon poslednjeg deteta svakom <stavka> elementu:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
```

```
var x=xmlDoc.getElementsByTagName("stavka");
var noviKomentar,noviTekst;
noviTekst="Revidirano u septembru 2006";
for (i=0;i<x.length;i++)
  {
  noviKomentar=xmlDoc.createComment(noviTekst);
  x[i].appendChild(noviKomentar);
  }
```

2.6 XML DOM dodavanje čvorova,

2.6.1 Dodavanje čvora na kraj liste čvorova

Metoda *appendChild()* se koristi da bi se dodao čvor nakon poslednjeg deteta naznačenog čvora.

Ova metoda se koristi kada pozicija čvora koji se dodaje nije važna.

Naredni fragment koda kreira element (<boja>) i dodaje ga nakon poslednjeg deteta svakog <stavka> elementa:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("stavka");
var noviElement,noviTekst
for (i=0;i<x.length;i++)
  {
  noviElement=xmlDoc.createElement("boja");
  noviTekst=xmlDoc.createTextNode("plava");
  noviElement.appendChild(noviTekst);
  x[i].appendChild(noviElement);
  }
```

2.6.2 Dodavanje čvora ispred određenog čvora

Metoda *insertBefore()* se koristi da bi se čvor dodao ispred određenog čvora.

Ova metoda se koristi kada je pozicija čvora koji se dodaje važna.

Fragment koda koji sledi kreira novi element <stavka> i dodaje ga ispred poslednjeg <stavka> elementa:

```
//proverava da li je poslednje dete element čvor
function get_lastchild(n)
{
var x=n.lastChild;
while (x.nodeType!=1)
  {
  x=x.previousSibling;
```

```
  }
return x;
}
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.documentElement;
var noviCvor=xmlDoc.createElement("stavka");
var noviNaziv=xmlDoc.createElement("naziv_artikla");
var noviTekst=xmlDoc.createTextNode("sveska");
noviNaziv.appendChild(noviTekst);
noviCvor.appendChild(noviNaziv);
x.insertBefore(noviCvor,get_lastchild(x));
```

Napomena: Internet Explorer izostavlja blanko tekst čvorove koji se generišu između čvorova (na primer karakteri za označavanje novog reda), dok Mozilla ne izostavlja. Tako da, u primeru iznad, funkcija *get_lastchild()* proverava tip čvora poslednjeg deteta zadatog parametra.

Element čvor ima *nodeType* 1, tako da ako poslednje dete čvora koji je zadat kao parametar nije element čvor, pomera se prema prethodnom čvoru i proverava da li je taj čvor element čvor. To se nastavlja sve dok poslednje dete čvor (koje mora da bude element čvor) ne bude pronađeno. Na ovaj način, rezultat će da bude ispravljen i u Internet Explorer-u i u Mozilla-i.

2.6.3 Postavljanje novog atributa i vrednosti atributa

Metoda *setAttribute()* može da se koristi da bi se promenila vrednost postojećeg atributa ili da bi se kreirao novi atribut/vrednost atributa za neki element.

Naredni fragment koda dodaje novi atribut/vrednost atributa svakom elementu <stavka>:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("stavka");
for(i=0;i<x.length;i++)
  {
  x.item(i).setAttribute("boja","PLAVA");
  }
```

2.6.4 Dodavanje podataka u tekst čvor

Metoda *insertData()* se koristi da bi se dodali podaci u tekst čvor.

Metoda *insertData()* ima dva parametra:

- *offset* – označava gde počinje umetanje karaktera. *Offset* vrednosti započinju od nule
- *string* – označava string koji se umeće

Naredni fragment koda dodaje “R+ “ tekst čvoru prvog elementa <naziv_artikla> učitanog XML-a.

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var x=xmlDoc.getElementsByTagName("naziv_artikla")[0].childNodes[0];
x.insertData(0,"R+ ");
```

2.7 XML DOM kloniranje čvorova

2.7.1 Kopiranje čvora

Metoda *cloneNode()* pravi kopiju određenog čvora.

Metoda *cloneNode()* sadrži parametar (*true* ili *false*). Ovaj parametar se odnosi na to da li će klonirani čvor da sadrži sve atribute i svu decu originalnog čvora.

Fragment koda koji sledi kopira prvi čvor <stavka> i zatim dodaje kopiju na kraj liste čvorova:

```
xmlDoc=loadXMLDoc("narudzbenica.xml");
var stariCvor=xmlDoc.getElementsByTagName("stavka")[0];
var noviCvor=stariCvor.cloneNode(true);
xmlDoc.documentElement.appendChild(noviCvor);

//Izlistava sve nazive artikala
var y=xmlDoc.getElementsByTagName("naziv_artikla");
for (i=0;i<y.length;i++)
  {
  document.write(y[i].childNodes[0].nodeValue);
  document.write("<br />");
  }
```

Izlaz:

```
DVD Verbatim
Toner za ink-jet
USB Kingston 4GB
Papir za štampač, 80g
DVD Verbatim
```

3 XML DOM reference

3.1 XML DOM tipovi čvorova

DOM predstavlja dokument kao hijerarhiju objekata čvorova.
Tipovi čvorova

Naredna tabela daje listu različitih W3C tipova čvorova i koje tipove čvorova oni mogu da imaju kao svoju decu:

Tip čvora	Opis	Deca
Document	Predstavlja ceo dokument (koreni čvor stable čvorova)	Element (maksimalno jedan), ProcessingInstruction, Comment, DocumentType
DocumentFragment	Predstavja "lagani" Document objekat, koji može da sadrži deo dokumenta.	Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference
DocumentType	Obezbeđuje interfejs za entitete definisane za dati document.	Nema
ProcessingInstruction	Predstavlja instrukciju procesiranja	Nema
EntityReference	Prestavlja reference entiteta	Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference
Element	Predstavlja element	Element, Text, Comment, ProcessingInstruction, CDATASection, EntityReference
Attr	Predstavlja atribut	Text, EntityReference
Text	Predstavlja tekstualni sadržaj elementa ili atributa	Nema
CDATASection	Predstavlja CDATA sekciju u dokumentu (tekst koji neće biti parsiran od strane parsera)	Nema
Comment	Predstavlja komentar	Nema
Entity	Predstavlja entitet	Element,

		ProcessingInstruction, Comment, Text, CDATASection, EntityReference
Notation	Predstavlja notaciju deklarisanu u DTD-u	Nema

3.1.1 Tipovi čvorova – povratne vrednosti

Naredna tabela daje listu koje može da vrate svojstva *nodeName* i *nodeType* za svaki tip čvora:

Tip čvora	nodeName povratne vrednosti	nodeValue povratne vrednosti
Document	#dokument	null
DocumentFragment	#fragment dokumenta	null
DocumentType	Naziv tipa dokumenta	null
EntityReference	Naziv reference entiteta	null
Element	Naziv elementa	null
Attr	Naziv atributa	Vrednost atributa
ProcessingInstruction	target	Sadržaj čvora
Comment	#komentar	Tekst komentara
Text	#tekst	Sadržaj čvora
CDATASection	#cdata sekcija	Sadržaj čvora
Entity	Naziv entiteta	null
Notation	Naziv oznake	null

Tipovi čvorova – imenovane konstante

NodeType	Imenovana konstanta
1	ELEMENT_NODE
2	ATTRIBUTE_NODE
3	TEXT_NODE
4	CDATA_SECTION_NODE
5	ENTITY_REFERENCE_NODE
6	ENTITY_NODE
7	PROCESSING_INSTRUCTION_NODE
8	COMMENT_NODE
9	DOCUMENT_NODE
10	DOCUMENT_TYPE_NODE
11	DOCUMENT_FRAGMENT_NODE
12	NOTATION_NODE

3.2 XML DOM *Node* objekat

Node objekat je primarni tip podataka za celi DOM.

Node objekat predstavlja jedan čvor u stablu dokumenta.*Node* može biti čvor element, čvor atribut, tekst čvor ili bilo koji drugi tip čvora koji je objašnjen poglavlju „Tipovi čvorova“.

Vredno je zapaziti da dok svi objekti nasleđuju svojstva/metode od *Node* objekta i na taj način obezbeđuju sebi mogućnost da imaju roditelje i decu, to ipak ne znači da svi oni mogu imati roditelje i decu. Na primer, *Text* čvor ne može da ima decu i dodavanje dece ovakvim čvorovima prouzrokovaće DOM grešku.

IE: Internet Explorer, F: Firefox, O: Opera, W3C: World Wide Web Consortium (Internet Standard)

3.2.1 Svojstva *Node* objekta

Svojstvo	Opis	IE	F	O	W3C
baseURI	Vraća apsolutnu URI putanju čvora	Ne	1	Ne	Da
childNodes	Vraća NodeList dece čvorova izabranog čvora	5	1	9	Da
firstChild	Vraća prvo dete čvora	5	1	9	Da
lastChild	Vraća poslednje dete čvora	5	1	9	Da
localName	Vraća lokalni deo imena čvora	Ne	1	9	Da
namespaceURI	Vraća URI opseg imena čvora	Ne	1	9	Da
nextSibling	Vraća čvor koji neposredno sledi čvor	5	1	9	Da
nodeName	Vraća ime čvora zavisno od njegovog tipa	5	1	9	Da
nodeType	Vraća tip čvora	5	1	9	Da
nodeValue	Postavlja ili vraća vredost čvora, zavisno od njegovog tipa	5	1	9	Da
ownerDocument	Vraća koreni element (dokument objekat) za dati čvor	5	1	9	Da
parentNode	Vraća roditeljski čvor datog čvora	5	1	9	Da
prefix	Postavlja ili vraća opseg imena prefiksa datog čvora	No	1	9	Da
previousSibling	Vraća prvi prethodni čvor datog čvora	5	1	9	Da
textContent	Postavlja ili vraća tekstualni	N	1	N	Da

	sadržaj čvora i njegove potomke	e		e	
text	Vraća tekst čvora i njegovih potomaka. Ovo je jedino IE svojstvo.	5	Ne	Ne	Ne
xml	Vraća XML čvora i njegovih potomaka. Ovo je jedino IE svojstvo.	5	Ne	Ne	Ne

3.2.2 Metode *Node* objekta

Metoda	Opis	IE	F	O	W3C
appendChild()	Dodaje novi dete čvor na kraju liste dece čvorova datog čvora	5	1	9	Da
cloneNode()	Klonira čvor	5	1	9	Da
compareDocumentPosition()	Poredi pozicije dva čvora u dokumentu	Ne	1	Ne	Da
getFeature(feature,version)	Vraća DOM objekat koji implementira specijalizovani API navedenih karakteristika i verzije			Ne	Da
getUserData(key)	Vraća objekat koji je pridružen ključu ovog čvora. Objekat prvo mora da bude postavljen na ovaj čvor pozivom setUserData sa istim ključem			Ne	Da
hasAttributes()	Vraća true ako čvor ima neki atribut, a u svakom drugom slučaju vraća false	Ne	1	9	Da
hasChildNodes()	Vraća true ako čvor ima makar jedno dete, a u suprotnom vraća false	5	1	9	Da
insertBefore()	Dodaje nov dete čvor pre postojećeg deteta čvora	5	1	9	Da
isDefaultNamespace(URI)	Vraća da li je određeni namespaceURI podrazumevani			Ne	Da
isEqualNode()	Proverava da li su dva čvora jednaka	Ne	Ne	Ne	Da
isSameNode()	Proverava da li su dva čvora isti čvor	Ne	1	Ne	Da
isSupported(feature,version)	Vraća da li je određena karakteristika podržana od strane zadatog čvora			9	Da

lookupNamespaceURI()	Vraća opseg imena za URI koji zadovoljava određeni prefiks	Ne	1	Ne	Da
lookupPrefix()	Vraća prefiks koji se podudara sa određenim URI opsegom imena	Ne	1	Ne	Da
normalize()	Stavlja sve tekst čvorove koji se nalaze unutar čvora (uključujući i atribute) u "normalnu" formu gde jedino struktura (na primer elementi, komentari, instrukcije za procesiranje, CDATA sekcije i reference entiteta) odvaja tekst čvorove, na primer, tako da nema ni susednih tekst čvorova niti praznih tekst čvorova	5	1	9	Da
removeChild()	Uklanja dete čvor	5	1	9	Da
replaceChild()	Zamenjuje dete čvor	5	1	9	Da
setUserData(key,data,handler)	Pridružuje objekat ključu čvora			Ne	Da

3.3 XML DOM *NodeList* objekat

Čvorovima u listi čvorova je moguće pristupiti preko njihovog indeksa (koji počinje od 0). *NodeList* se održava ažurnom. Ukoliko je neki element obrisan ili dodat u listu čvorova ili u XML dokument, lista se automatski ažurira.

Napomena: U listi čvorova, čvorovi se vraćaju u onom redosledu u kojem se nalaze u XML dokumentu.

IE: Internet Explorer, F: Firefox, O: Opera, W3C: World Wide Web Consortium (Internet Standard)

3.3.1 Svojstva *NodeList* objekta

Svojstvo	Opis	IE	F	O	W3C
length	Vraća broj čvorova u listi	5	1	9	Da

3.3.2 Metode *NodeList* objekta

Metoda	Opis	IE	F	O	W3C

item()	Vraća čvor koji ima određeni ineks u listi čvorova	5	1	9	Da

3.4 XML DOM *NamedNodeMap* objekat

Čvorovima u *NamedNodeMap* je moguće pristupiti preko njihovog imena. *NamedNodeMap* se održava ažurnom. Ukoliko je neki element obrisan ili dodat u listu čvorova ili u XML dokument, lista se automatski ažurira.

Napomena: U mapi imenovanih čvorova, čvorovi se ne vraćaju u nekom određenom redosledu.

IE: Internet Explorer, F: Firefox, O: Opera, W3C: World Wide Web Consortium (Internet Standard)

3.4.1 Svojstva *NamedNodeMap* objekta

Svojstvo	Opis	IE	F	O	W3C
length	Vraća broj čvorova u listi	5	1	9	Da

3.4.2 Metode *NamedNodeMap* objekta

Metoda	Opis	IE	F	O	W3C
getNamedItem()	Vraća određeni čvor (po imenu)	5	1	9	Da
getNamedItemNS()	Vraća određeni čvor (po imenu i opsegu imena)			9	Da
item()	Vraća čvor koji ima određeni indeks	5	1	9	Da
removeNamedItem()	Uklanja određeni čvor (po imenu)	6	1	9	Da
removeNamedItemNS()	Uklanja određeni čvor (po imenu i opsegu imena)			9	Da
setNamedItem()	Postavlja određeni čvor (po imenu)			9	Da
setNamedItemNS()	Postavlja određeni čvor (po imenu i opsegu imena)			9	Da

3.5 XML DOM *Document* objekat

Document objekat je koren stabla dokumenta i omogućuje prvi pristup podacima dokumenta.

S obzirom da element čvorovi, tekst čvorovi, komentari, instrukcije za instrukcije obrade itd. ne mogu da postoje van dokumenta, *Document* objekat takođe sadrži metode za kreiranje ovih objekata. *Node* objekti imaju svojstvo *ownerDocument* koje ih pridružuje Document-u koji ih je kreirao.

IE: Internet Explorer, F: Firefox, O: Opera, W3C: World Wide Web Consortium (Internet Standard)

3.5.1 Svojstva *Document* objekta

Svojstvo	Opis	IE	F	O	W3C
async	Određuje da li download-ovanje XML fajla može da bude asinhrono ili ne	5	1.5	9	Ne
childNodes	Vraća NodeList dece čvorova za dokument	5	1	9	Da
doctype	Vraća DTD pridružen dokumentu	6	1	9	Da
documentElement	Vraća koreni čvor dokumenta	5	1	9	Da
documentURI	Postavlja ili vraća lokaciju dokumenta	Ne	1	9	Da
domConfig	Vraća konfiguraciju kada je pozvana metoda normalizeDocument()			Ne	Da
firstChild	Vraća prvo dete čvor dokumenta	5	1	9	Da
implementation	Returns the DOMImplementation object that handles this document	Ne	1	9	Da
inputEncoding	Vraća enkodiranje korišćeno za dokument (kada je parsirano)	Ne	1	Ne	Da
lastChild	Vraća poslednje dete čvor dokumenta	5	1	9	Da
nodeName	Vraća ime čvora (zavisno od njegovog tipa	5	1	9	Da
nodeType	Vraća tip čvora	5	1	9	Da
nodeValue	Postavlja ili vraća vrednost čvora (zavisno od njegovog tipa)	5	1	9	Da
strictErrorChecking	Postavlja ili vraća da li je provera grešaka uključena ili ne	Ne	1	Ne	Da
text	Vraća tekst čvora i njegovih	5	N	N	Ne

	potomaka. Podržava samo IE		e	e	
xml	Vraća XML čvora i njegovih potomaka. Podržava samo IE	5	Ne	Ne	Ne
xmlEncoding	Vraća XML enkodiranje dokumenta	Ne	1	Ne	Da
xmlStandalone	Postavlja ili vraća da li je dokument samostalan	Ne	1	Ne	Da
xmlVersion	Postavlja ili vraća XML verziju dokumenta	Ne	1	Ne	Da

3.5.2 Metode *Document* objekta

Metoda	Opis	IE	F	O	W3C
adoptNode(sourcenode)	Usvaja čvor iz drugog dokumenta i vraća ga			Ne	Da
createAttribute(name)	Kreira atribut čvor sa određenim imenom i vraća novi Attribute objekat	6	1	9	Da
createAttributeNS(uri,name)	Kreira atribut čvor sa određenim imenom i opsegom imena i vraća novi Attribute objekat			9	Da
createCDATASection()	Kreira CDATA sekcija čvor	5	1	9	Da
createComment()	Kreira komentar čvor	6	1	9	Da
createDocumentFragment()	Kreira prazan DocumentFragment objekat i vraća ga	5	1	9	Da
createElement()	Kreira element čvor	5	1	9	Da
createElementNS()	Kreira element čvor sa određenim opsegom imena	Ne	1	9	Da
createEntityReference(name)	Kreira EntitzReference objekat i vraća ga	5		Ne	Da
createProcessingInstruction(target,data)	Kreira ProcesingInstruction objekat i vraća ga	5		9	Da
createTextNode()	Kreira tekst čvor	5	1	9	Da
getElementById(id)	Vraća element koji ima ID atribut sa datom vrednošću. Ukoliko takav element ne postoji, vraća se null	5	1	9	Da
getElementsByTagName()	Vraća NedeList svih elemenata sa određenim imenom	5	1	9	Da
getElementsByTagNameNS	Vraća NodeList svih	N	1	9	Da

()	elemenata sa određenim imenom i opsegom imena	e			
importNode(nodetoimport, deep)	Importuje čvor iz jednog dokumenta u tekući dokument. Ova metoda pravi kopiju izvornog čvora. Ukoliko je parametar deep postavljen na true, importovaće svu decu tog čvora. Ukoliko je postavljen na false, importovaće samo dati čvor. Ova metoda vraća importovani čvor			9	Da
normalizeDocument()				Ne	Da
renameNode()	Menja ime element ili atribut čvoru			Ne	Da

3.6 XML DOM DocumentImplementation objekat

DOMImplementation objekat izvodi operacije koje su nezavisne od bilo koje instance Document obejct modela.

Metoda	Opis	IE	F	O	W3C
createDocument(nsURI, name, doctype)	Kreira novi DOM Document objekat određenog tipa				Da
createDocumentType(name, pubId, systemId)	Kreira prazan DocumentType čvor				Da
getFeature(feature, version)	Vraća objekat koji implementira API-je određenih karakteristika i verzije ukoliko postoje				Da
hasFeature(feature, version)	Proverava da li DOM implementacija implementira određenu karakteristiku i verziju				Da

3.7 XML DOM *DocumentType* objekat

Svaki dokument ima *DOCTYPE* atribut čija vrednost je ili null ili *DocumentType* objekat.

DocumentType obezbeđuje interfejs entitetima definisanim za XML dokument.

Svojstvo	Opis	IE	F	O	W3C
entities	Vraća NamedNodeMap koji sadrži entitete deklarisane u DTD-u	6	Ne	9	Da
internalSubset	Vraća interni DTD kao string	Ne	Ne	Ne	Da
name	Returns the name of the DTD	6	1	9	Da
notations	Vraća NamedNodeMap koji sadrži notacije deklarisane u DTD-u	6	Ne	9	Da
systemId	Vraća sistemski identifikator eksternog DTD-a	Ne	1	9	Da

3.8 XML DOM ProcessingInstruction objekat

ProcessingInstruction objekat predstavlja instrukciju obrade.

Instrukcija obrade se koristi kao način da se čuvaju procesorom određene informacije u tekstu XML dokumenta.

Svojstvo	Opis	IE	F	O	W3C
data	Postavlja ili vraća sadržaj instrukcije obrade			Ne	Da
target	Vraća odredište instrukcije obrade			Ne	Da

3.9 XML DOM *Element* objekat

Element objekat predstavlja element u XML dokumentu. *Elementi* mogu da sadrže atribute, druge elemente ili tekst. Ako element sadrži tekst, tekst je predstavljen u tekst čvoru.

VAŽNO! Tekst je uvek sačuvan u tekst čvoru. Česta greška u DOM obradi je da kada se dođe do određenog čvora očekuje se da sadrži tekst. Međutim, čak i najjednostavniji element čvor ima tekst unutar sebe. Na primer, u <godina>2005</godina>, postoji element čvor (godina) i tekst čvor unutar njega, koji sadrži tekst (2005).

Pošto je *Element* objekat takođe *Node*, nasleđuje svojstva i metode *Node* objekta.
IE: Internet Explorer, F: Firefox, O: Opera, W3C: World Wide Web Consortium (Internet Standard)

3.9.1 Svojstva *Element* objekata

Svojstva	Opis	IE	F	O	W3C
attributes	Vraća NamedNodeMap atributa za element	5	1	9	Da
baseURI	Vraća apsolutnu URI putanju elementa	Ne	1	Ne	Da
childNodes	Vraća NodeList dece čvorova elementa	5	1	9	Da
firstChild	Vraća prvo dete elementa	5	1	9	Da
lastChild	Vraća poslednje dete elementa	5	1	9	Da
localName	Vraća lokalni deo imena elementa	Ne	1	9	Da
namespaceURI	Vraća URI opseg imena elementa	Ne	1	9	Da
nextSibling	Vraća čvor koji neposredno sledi element	5	1	9	Da
nodeName	Vraća ime čvora, zavisno od njegovog tipa	5	1	9	Da
nodeType	Vraća tip čvora	5	1	9	Da
ownerDocument	Vraća koreni element (Document objekat) za element	5	1	9	Da
parentNode	Vraća roditelja elementa	5	1	9	Da
prefix	Postavlja ili vraća prefiks opsega imena elementa	Ne	1	9	Da
previousSibling	Vraća čvor koji je neposredno pre čvora	5	1	9	Da
schemaTypeInfo	Vraća informacije o tipu koji je pridružen elementu			Ne	Da
tagName	Vraća ime elementa	5	1	9	Da

textContent	Postavlja ili vraća tekstualni sadržaj elementa i njegovih potomaka	Ne	1	Ne	Da
text	Vraća tekst čvora i njegovih potomaka. Jedino podržava IE	5	Ne	Ne	Ne
xml	Vraća XML od čvora i njegovih potomaka. Jedino podržava IE	5	Ne	Ne	Ne

3.9.2 Metode *Element* objekta

Metoda	Opis	IE	F	O	W3C
appendChild()	Dodaje novi dete čvor na kraj liste dece čvorova	5	1	9	Da
cloneNode()	Klonira čvor	5	1	9	Da
compareDocumentPosition()	Poredi poziciju u dokumentu dva čvora	Ne	1	Ne	Da
getAttribute()	Vraća vrednost atributa	5	1	9	Da
getAttributeNS()	Vraća vrednost atributa (sa opsegom imena)	Ne	1	9	Da
getAttributeNode()	Vraća atribut čvor kao Attribute objekat	5	1	9	Da
getAttributeNodeNS()	Vraća atribut čvor (sa opsegom imena) kao Attribute objekat	Ne		9	Da
getElementsByTagName()	Vraća NodeList elementa koji se poklapaju i njihovu decu	5	1	9	Da
getElementsByTagNameNS()	Vraća NodeList elementa koji se poklapaju (sa opsegom imena) i njihovu decu	Ne	1	9	Da
getFeature(feature,version)	Vraća DOM objekat koji emplementira specijalizovani API određenih karakteristika i verzije			Ne	Da
getUserData(key)	Vraća objekat koji je pridružen ključu ovog čvora. Objekat prvo mora da bude postavljen na ovaj čvor pozivom setUserData sa istim ključem			Ne	Da
hasAttribute()	Vraća da li element ima neki atribut koji se poklapa sa određenim imenom	5	1	9	Da
hasAttributeNS()	Vraća da li element ima	N	1	9	Da

	neki atribut koji se poklapa sa određenim imenom i opsegom imena	e			
hasAttributes()	Vraća da li element ima bilo koji atribut	5	1	9	Da
hasChildNodes()	Vraća da li element ima bilo koje dete čvor	5	1	9	Da
insertBefore()	Ubacuje novi dete čvor ispred postojećeg čvora	5	1	9	Da
isDefaultNamespace(URI)	Vraća da li je određeni namespaceURI podrazumevan			Ne	Da
isEqualNode()	Proverava da li su dva čvora jednaka	Ne	Ne	Ne	Da
isSameNode()	Proverava da li su dva čvora isti čvor	Ne	1	Ne	Da
isSupported(feature,version)	Vraća da li je određena karakteristika podržana od strane elementa			9	Da
lookupNamespaceURI()	Vraća URI opseg imena koji se poklapa odreženim prefiksom	Ne	1	Ne	Da
lookupPrefix()	Vraća prefiks koji se poklapa sa određenim URI opsegom imena	Ne	1	Ne	Da
normalize()	Stavlja sve tekst čvorove koji se nalaze unutar čvora (uključujući i atribute) u “normalnu“ formu gde jedino struktura (na primer elementi, komentari, instrukcije za procesiranje, CDATA sekcije i reference entiteta) odvaja tekst čvorove, na primer, tako da nema ni susednih tekst čvorova niti praznih tekst čvorova	5	1	9	Da
removeAttribute()	Uklanja određeni atribut	5	1	9	Da
removeAttributeNS()	Uklanja određeni atribut (sa opsegom imena)	Ne	1	9	Da
removeAttributeNode()	Uklanja određeni atribut čvor	5	1	9	Da
removeChild()	Uklanja dete čvor	5	1	9	Da
replaceChild()	Menja dete čvor	5	1	9	Da
setUserData(key,data,handler)	Pridružuje objekat ključu elementa			Ne	Da
setAttribute()	Dodaje novi atribut	5	1	9	Da

setAttributeNS()	Dodaje novi atribut (sa opsegom imena)		1	9	Da
setAttributeNode()	Dodaje novi atribut čvor	5	1	9	Da
setAttributeNodeNS(attrnode)	Dodaje novi atribut čvor (sa osegom imena)			9	Da
setIdAttribute(name,isId)	Ukoliko je svojstvo isId Attribute objekta jednako true, ova metoda označava određeni atribut kao korisnički određeni ID atribut			Ne	Da
setIdAttributeNS(uri,name, isId)	Ukoliko je svojstvo isId Attribute objekta jednako true, ova metoda označava određeni atribut (sa opsegom imena) kao korisnički određeni ID atribut			Ne	Da
setIdAttributeNode(idAttr,isId)	Ukoliko je svojstvo isId Attribute objekta jednako true, ova metoda označava određeni atribut kao korisnički određeni ID atribut			Ne	Da

3.10 XML DOM *Attr* objekat

Attr objekat predstavlja atribut *Element* objekata. Dozvoljene vrednosti za atribute je obično definisan u DTD-u.

Pošto je *Attr* objekat takođe *Node*, nasleđuje sva svojstva i metode *Node* objekata. Međutim, atribut nema roditelja i ne može da bude dete čvor nekog elementa i vratiće null za mnoga *Node* svojstva.

Svojstvo	Opis	IE	F	O	W3C
baseURI	Vraća apsolutnu URI putanju atributa	Ne	1	Ne	Da
isId	Vraća true ako je atribut poznatog ID tipa, inače vraća false	Ne	Ne	Ne	Da
localName	Vraća lokalni deo imena atributa	Ne	1	9	Da
name	Vraća ime atributa	5	1	9	Da
namespaceURI	Vraća URI opseg imena atributa	Ne	1	9	Da
nodeName	Vraća ime čvora, zavisno od njegovog tipa	5	1	9	Da
nodeType	Vraća tip čvora	5	1	9	Da
nodeValue	Postavlja ili vraća vrednost čvora, zavisno od njegovog tipa	5	1	9	Da
ownerDocument	Vraća koreni čvor (Document objekat) atributa	5	1	9	Da
ownerElement	Vraća element čvor kome je pridružen atribut	Ne	1	9	Da
prefix	Postavlja ili vraća prefiks opsega imena atributa	Ne	1	9	Da
schemaTypeInfo	Vraća informacije o tipu koji je pridružen atributu	Ne	Ne	Ne	Da
specified	Vraća true ako je vrednost atributa postavljena u dokumentu, a false ako je njegova podrazumevana vrednost u DTD-u/Shema-i	5	1	9	Da
textContent	Postavlja ili vraća tekstualni sadržaj atributa	Ne	1	9	Da
text	Vraća tekst atributa. Podržava jedino IE	5	Ne	Ne	Ne
value	Postavlja ili vraća vrednost atributa	5	1	9	Da
xml	Vraća XML atributa. Podržava jedino IE	5	Ne	Ne	Ne

3.11 XML DOM *Text* objekat

Text objekat predstavlja tekstualni sadržaj elementa ili atributa.

3.11.1 Svojstva *Text* objekta

Svojstvo	Opis	IE	F	O	W3C
data	Postavlja ili vraća tekst elementa ili atributa	6	1	9	Da
isElementContentWhitespace	Vraća true ukoliko tekst čvor sadrži blanko karaktere, u protivnom vraća false	Ne	Ne	Ne	Da
length	Vraća dužinu tekst elementa ili atributa	6	1	9	Da
wholeText	Vraća sav tekst tekst čvorova koji su susedni ovom čvoru, sastavljen prema redosledu u dokumentu	Ne	Ne	Ne	Da

3.11.2 Metode *Text* objekta

Metoda	Opis	IE	F	O	W3C
appendData()	Dodaje podatke čvoru	6	1	9	Da
deleteData()	Briše podatke iz čvora	6	1	9	Da
insertData()	Unosi podatke u čvor	6	1	9	Da
replaceData()	Zamenjuje podatke u čvoru	6	1	9	Da
replaceWholeText(text)	Zamenjuje tekst ovog čvora i svih susednih tekst čvorova određenim tekstom	Ne	Ne	Ne	Da
splitText()	Deli čvor na dva čvora na označenom mestu i vraća novi čvor koji sadrži tekst koji se nalazi posle tog mesta deljenja	6	1	9	Da
substringData()	Izvlači podatke iz čvora	6	1	9	Da

3.12 XML DOM *CDATASection* objekat

3.12.1 CDADASection objekat

CDATASection objekat predstavlja CDATA sekciju u dokumentu.

CDATA sekcija sadrži tekst koji se ne parsira od strane parsera. Tagovi koji se nalaze unutar CDATA sekcije se ne tretiraju kao markeri. Primarna svrha je da se uključi materijal kao što su XML fragmenti, bez potrebe da se izbegnu svi delimiteri.

Jedini delimiter koji se prepoznaje u CDATA sekciji je koji označava kraj CDATA sekcije. CDATA sekcija ne može da bude ugnježdena.

IE: Internet Explorer, F: Firefox, O: Opera, W3C: World Wide Web Consortium (Internet Standard)

3.12.2 Svojstva *CDATASection* objekta

Svojstva	Opis	IE	F	O	W3C
data	Postavlja ili vraća tekst čvora	6	1	Ne	Da
length	Vraća dužinu CDATA sekcije	6	1	Ne	Da

3.12.3 Metode *CDATASection* objekta

Metoda	Opis	IE	F	O	W3C
appendData()	Dodaje podatke u čvor	6	1	Ne	Da
deleteData()	Briše podatke iz čvora	6	1	Ne	Da
insertData()	Ubacuje podatke u čvor	6	1	Ne	Da
replaceData()	Menja podatke u čvoru	6	1	Ne	Da
splitText()	Deli CDATA čvor u dva čvora	6	1	Ne	
substringData()	Izvlači podatke iz čvora	6	1	Ne	Da

3.13 XML DOM *Comment* objekat

3.13.1 *Comment* objekat

Comment objekat predstavlja sadržaj komentara čvora u dokumentu.

3.13.2 Svojstva *Comment* objekta

Svojstvo	Opis	IE	F	O	W3C
data	Postavlja ili vraća tekst čvora	6	1	9	Da
length	Vraća dužinu teksta ovog čvora	6	1	9	Da

3.13.3 Metode *Comment* objekta

Metoda	Opis	IE	F	O	W3C
appendData()	Dodaje podatke u čvor	6	1	9	Da
deleteData()	Briše podatke iz čvora	6	1	9	Da
insertData()	Ubacuje podatke u čvor	6	1	9	Da
replaceData()	Menja podatke u čvoru	6	1	9	Da
substringData()	Izvlači podatke iz čvora	6	1	9	Da

3.14 HttpRequest objekat

XMLHttpRequest objekat je podržan od strane Internet Explorer-a 5.0+, Safari-ja 1.2, Mozilla-e 1.0/Firefox-a, Opera-e 9 i Netscape-a 7.

Šta je HTTP zahtev?
Sa HTTP zahtevom veb strana može da zahteva i dobije odgovor od veb servera – bez ponovnog učitavanja strane. Korisnik može da ostane na istoj strani i da ne primeti da je skript možda zatražio strane ili poslao podatke na server u pozadini.

Korišćenjem *XMLHttpRequest* objekta veb developer može da zameni stranu podacima sa servera nakon što je strana učitana.

Google preporuka je da se koristi *XMLHttpRequest* objekat da bi se kreirao veoma dinamičan veb interfejs: kada počnemo da unosimo podatke u Google pretraživač, JavaScript šalje ukucana slova serveru i server šalje listu predloga.

3.14.1 Da li je *XMLHttpRequest* objekat W3C standard?

XMLHttpRequest objekat je JavaScript objekat i nije propisan ni jednom W3C preporukom.
Međutim, W3C DOM Level 3 "Učitavanje i snimanje" specifikacija sadrži neke slične funkcionalnosti, ali ove nisu implementirane još ni u jednom browser-u. Tako da trenutno ukoliko želite da pošaljete HTTP zahtev iz browser-a, morate da koristite XMLHttpRequest objekat.

Kreiranje XMLHttpRequest objekta
Za Mozilla-u, Firefox, Safari, Opera-u i Netscape:

```
var xmlhttp=new XMLHttpRequest()
```

Za Internet Explorer:

```
var xmlhttp=new ActiveXObject("Microsoft.XMLHTTP")
```

Primer

```
<script type="text/javascript">
var xmlhttp
function loadXMLDoc(url)
{
xmlhttp=null
// kod za Mozilla, itd.
if (window.XMLHttpRequest)
  {
  xmlhttp=new XMLHttpRequest()
  }
// kod za IE
```

```
else if (window.ActiveXObject)
  {
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP")
  }
if (xmlhttp!=null)
  {
  xmlhttp.onreadystatechange=state_Change
  xmlhttp.open("GET",url,true)
  xmlhttp.send(null)
  }
else
  {
  alert("Vas browser ne podrzava XMLHTTP.")
  }
}
function state_Change()
{
// ako xmlhttp prikaze "loaded"
if (xmlhttp.readyState==4)
  {
  // ukoliko je "OK"
  if (xmlhttp.status==200)
    {
    // ...još neki kod...
    }
  else
    {
    alert("Problem u pribavljanju XML podataka")
    }
  }
}
</script>
```

Napomena: Jedno važno svojstvo prethodnog primera je *onreadystatechange* svojstvo. Ovo svojstvo hvata događaj koji se okida svaki put kada se stanje zahteva promeni. Stanje započinje sa 0 (neinicijalizovano) do 4 (kompletirano). Funkcijom *state_Chante()* se proverava promena stanja I na osnovu toga možemo da odredimo kada je proces završen i nastavimo sa radom tek kada je u potpunosti uspešno obavljen.

3.14.2 Zašto koristimo *async* u našim promerima?

Većina primera ovde koristi *async* mod (treći parametar metode *open()* postavljen na *true*).
Async parametar određuje da li će zahtev biti obrađen asinhrono ili ne. *True* označava da će skript nastaviti da se izvršava nakon *send()* metode, bez čekanja na odgovor od servera. *False* označava da skript čeka na odgovor pre nego što nastavi izvršavanje. Postavljanjem ovog parametra na *false*, rizikuje se da se skript prekine u slučaju problema na mreži ili na

serveru, ili, ako je zahtev suviše dug (UI se zaključavaju dok se ne uputi zahtev) korisnik može da vidi poruku „Not Responding“. Bezbednije je se pošalje asinhroni zahtev i kod dizajnira oko onreadystatechange događaja!

3.15 XML/ASP

Takođe može da se otvori i pošalje XML dokument do ASP strane na serveru, analizira zahtev i vrati rezultat.

```
<html>
<body>
<script type="text/javascript">
var xmlHttp=new ActiveXObject("Microsoft.XMLHTTP")
xmlHttp.open("GET", "beleska.xml", false)
xmlHttp.send()
xmlDoc=xmlHttp.responseText
xmlHttp.open("POST", "demo_dom_http.asp", false)
xmlHttp.send(xmlDoc)
document.write(xmlHttp.responseText)
</script>
</body>
</html>
```

ASP strana, napisana u VBScript-u:

```
<%
set xmldoc = Server.CreateObject("Microsoft.XMLDOM")
xmldoc.async=false
xmldoc.load(request)

for each x in xmldoc.documentElement.childNodes
  if x.NodeName = "za" then name=x.text
next
response.write(name)
%>
```

Vraćanje rezultata nazad klijentu korišćenjem svojstva *response.write*.

3.16 *XMLHttpRequest* objekat reference

3.16.1 Metode

Metoda	Opis
abort()	Otkazuje tekući zahtev
getAllResponseHeaders()	Vraća kompletan skup http zaglavlja ko string
getResponseHeader("headername")	Vraća vrednost određenog http zaglavlja
open("method","URL",async,"uname","pswd")	Određuje metodu, URL i druge opcione atribute zahteva

	Parametar metode može da ima vrednost"GET", "POST", ili "PUT" (koristi se "GET" kada se zahtevaju podaci, a "POST" kada se šalju podaci (naročito kada je dužina podataka veća od 512 bajtova)) Parametar URL može da bude ili relativan ili kompletan URL. Async parametar određuje da li će zahtev biti obrađen asinhrono ili ne. True označava da će skript nastaviti da se izvršava nakon send() metode, bez čekanja na odgovor od servera. False označava da skript čeka na odgovor pre nego što nastavi izvršavanje.
send(content)	Šalje zahtev
setRequestHeader("label","value")	Dodaje par labela/vrednost http zaglavlju koji će biti poslat

3.16.2 Svojstva

Svojstvo	Opis
onreadystatechange	An event handler for an event that fires at every state change
readyState	Vraća stanje objekta: 0 = neinicijalizovano 1 = učitavanje 2 = učitano 3 = interaktivno 4 = kompletirano
responseText	Vraća odgovor kao string
responseXML	Vraća odgovor kao XML. Ovo svojstvo vraća XML dokument objekat koji može da bude ispitan i parsiran korišćenjem metoda i svojstava W3C DOM stabla čvorova
status	Vraća status kao broj (na primer 404 za "Not Found" ili 200 za "OK")
statusText	Vraća status kao string (na primer "Not Found" ili "OK)

3.17 XML DOM parser grešaka

Microsoft *parseError* objekat može da se koristi da bi se sakupile informacije o greškama od Microsoft XML parsera.

3.17.1 parseError objekat

Kada pokušate da otvorite XML dokument može se dogoditi da se pokrene parser grešaka.
Preko parserError objekta, mogu se pronaći greške u kodu, greške u tekstu, linije koje su prouzrokovale greške i tako dalje.

Napomena: Objekat *parseError* nije deo W3C standarda!
Greška u fajlu

U naredni kod prikazuje pokušaj učitavanja nepostojećeg fajla i prikaz nekih njegovih svojstva koja se odnose na greške:

```
var xmlDoc = new ActiveXObject("Microsoft.XMLDOM")
xmlDoc.async="false"
xmlDoc.load("ksdjf.xml")

document.write("Greska u kodu: " + xmlDoc.parseError.errorCode)
document.write("<br />Razlog greske: " + xmlDoc.parseError.reason)
document.write("<br />Linija u kojoj je greska: " +
xmlDoc.parseError.line)
```

XML greška
U narednom kodu dopustićemo da parser učita XML dokument koji nije dobro oformljen.

```
var xmlDoc = new ActiveXObject("Microsoft.XMLDOM")
xmlDoc.async="false"
xmlDoc.load("beleska_error.xml")

document.write("Greska u kodu: " + xmlDoc.parseError.errorCode)
document.write("<br />Razlog greske: " + xmlDoc.parseError.reason)
document.write("<br />Linija u kojoj je greska: " +
xmlDoc.parseError.line)
```

3.17.2 Svojstva *parseError* objekta

Svojstvo	Opis
errorCode	Vraća grešku u kodu kao long integer
reason	Vraća string koji sadrži razlog greške
line	Vraća long integer koji predstavlja broj linije u kojoj se dogodila greška
linepos	Vraća long integer koji predstavlja poziciju linije u kojoj se dogodila greška

srcText	Vraća string koji sadrži liniju u kojoj se dogodila greška
url	Vraća URL koji se odnosi na učitani dokument
filepos	Vraća long integer koji predstavlja poziciju u fajlu gde se dogodila greška

4 PRILOG - Prikazivanje XML-a pomoću XSL-a

XSL je napredan jezik stilova XML-a. XSL je mnogo savršeniji od CSS-a. Jedini način za korišćenje XSL-a je transformacija XML-a u HTML pre nego što se dobije prikaz pomoću čitača.

Izgled podataka u XML fajlu je (narudzbenica.xml):

```
<?xml version="1.0 encoding = "UTF-8?>
<narudzbenica rb="1235">
      <datum>12.11.2008.</datum>
      <zaglavlje>
            <poslovni_subjekt uloga="narucilac">
                  <naziv>Fakultet organizacionih nauka</naziv>
                  <adresa>Jove Ilica 154</adresa>
                  <email>nabavka@fon.rs</email>
                  <telefon>011/3950800</telefon>
                  <ovlasceno_lice>Jovan Jovanovic</ovlasceno_lice>
            </poslovni_subjekt>
            <poslovni_subjekt uloga="dobavljac">
                  <naziv>Office 1 Superstore</naziv>
                  <adresa>YUBC, Bul. Mihajla Pupina 10b</adresa>
                  <email>yubc@office1.co.yu</email>
                  <telefon>011/3131003</telefon>
                  <ovlasceno_lice>Petar Petrovic</ovlasceno_lice>
            </poslovni_subjekt>
            <!--I narucilac i dobavljac predstavljaju poslovne subjekte
samo sa razlicitim ulogama u procesu nabavke-->
      </zaglavlje>
      <lista_artikala>
            <stavka rb="1">
                  <naziv_art>DVD Verbatim</naziv_art>
                  <sifra>43522</sifra>
                  <jed_mere>kom</jed_mere>
                  <cena_po_j_mere>20</cena_po_j_mere>
                  <kolicina>100</kolicina>
            </stavka>
            <stavka rb="2">
                  <naziv_art>Toner za ink-jet</naziv_art>
                  <sifra>42057</sifra>
                  <jed_mere>dl</jed_mere>
                  <cena_po_j_mere>100</cena_po_j_mere>
                  <kolicina>5</kolicina>
            </stavka>
            <stavka rb="3">
                  <naziv_art>USB Kingston 4GB</naziv_art>
                  <sifra>41862</sifra>
                  <jed_mere>kom</jed_mere>
                  <cena_po_j_mere>1000</cena_po_j_mere>
                  <kolicina>3</kolicina>
            </stavka>
            <stavka rb="4">
                  <naziv_art>Papir za štampač, 80g</naziv_art>
```

```
                    <sifra>42789</sifra>
                    <jed_mere>ris</jed_mere>
                    <cena_po_j_mere>250</cena_po_j_mere>
                    <kolicina>5</kolicina>
                </stavka>
        </lista_artikala>
</narudzbenica>
```

4.1 Prilog formatiranje narudžbenice

Odgovarajući XSL fajl (formatiranje_narudzbenice.xsl):

```
<?xml version="1.0" encoding="utf-8" ?>
<html   xsl:version="1.0"   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns="http://www.w3.org/1999/xhtml" encoding="utf-8" lang="sr">
<body          style="font-family:Arial,helvetica,sans-serif;font-size:12pt;
background-color:#EEEEEE">
 <xsl:for-each select="narudzbenica/zaglavlje/poslovni_subjekt">

 <div style="background-color:teal;color:white;padding:4px">
      <span style="font-weight:bold;color:black">
            <xsl:value-of select="@uloga"/>: <br/>
      </span>
            <xsl:value-of select="naziv" /> <br/>
      <span style="font-weight:bold;color:white">
            <xsl:value-of select="adresa" />
      </span><br/>
            <xsl:value-of select="email" /> <br/>
      <span style="font-weight:bold;color:white">
            <xsl:value-of select="telefon" />
      </span><br/>
            <xsl:value-of select="ovlasceno_lice" /> <br/>
 </div>
 </xsl:for-each>
   <table border="1" padding="4">
      <tr bgcolor="#9acd32">
          <th>Rb</th>
        <th>Naziv Artikla</th>
        <th>Sifra artikla</th>
        <th>Cena po jedinici mere</th>
        <th>Jedinica mere</th>
            <th>Količina</th>
        </tr>
      <xsl:for-each select="narudzbenica/lista_artikala/stavka">
        <tr>
          <td><xsl:value-of select="@rbs"/></td>
              <td><xsl:value-of select="naziv_art"/></td>
          <td><xsl:value-of select="sifra"/></td>
              <td><xsl:value-of select="cena_po_j_mere"/></td>
          <td><xsl:value-of select="jed_mere"/></td>
          <td><xsl:value-of select="kolicina"/></td>
        </tr>
      </xsl:for-each>
    </table>
```

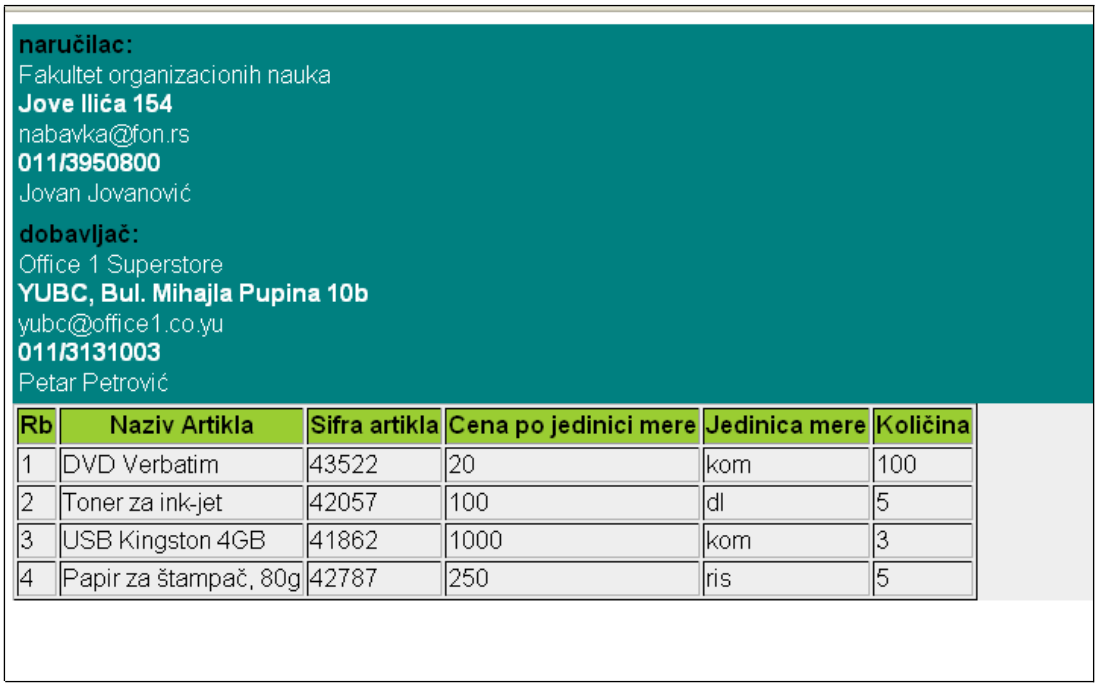
Formatiranje se primenjuje na XML podatke komandom:

```
<?xml-stylesheet type="text/xsl" href="formatiranje_narudzbenice.xsl" ?>
```

koja se postavlja u XML fajl odmah nakon deklaracije

```
<?xml version="1.0" encoding="utf-8" ?>
<?xml-stylesheet type="text/xsl" href="formatiranje_narudzbenice.xsl" ?>
```

Rezultat je transformacija XML-a u HTML i dobija se sledeći prikaz:



naručilac:
Fakultet organizacionih nauka
Jove Ilića 154
nabavka@fon.rs
011/3950800
Jovan Jovanović

dobavljač:
Office 1 Superstore
YUBC, Bul. Mihajla Pupina 10b
yubc@office1.co.yu
011/3131003
Petar Petrović

Rb	Naziv Artikla	Sifra artikla	Cena po jedinici mere	Jedinica mere	Količina
1	DVD Verbatim	43522	20	kom	100
2	Toner za ink-jet	42057	100	dl	5
3	USB Kingston 4GB	41862	1000	kom	3
4	Papir za štampač, 80g	42787	250	ris	5

Slika 1 Rezultat primene XSL na dati XML dokument