Ispravno formiranje i validacija XML-a

Kroz anatomiju jezika XML u prethodnoj lekciji su predstavljena i brojna sintaksna pravila XML-a. XML dokument koji zadovoljava sva sintaksna pravila jezika drugačije se naziva ispravno formiran (*well-formed*) XML dokument. Pored pojma ispravno formiranog, u XML svetu postoji i pojam validnog XML dokumenta (*valid XML*). U lekciji pred vama biće objašnjeno šta je to validan XML dokument i koja je razlika između ispravno formiranog i validnog XML dokumenta.

Ispravno formiran XML dokument

XML poseduje jasno definisana sintaksna pravila. Kako bi programi koji se bave čitanjem i interpretacijom XML koda mogli adekvatno da istretiraju kod, neophodno je da napisani XML dokument u potpunosti zadovolji sva sintaksna pravila XML jezika. Štaviše, jedino takav dokument i može biti pročitan. Bilo kakav pokušaj čitanja XML dokumenta koji poseduje makar jednu sintaksnu grešku završiće se neuspehom. U prethodnoj lekciji se tako nešto moglo videti prilikom pokušaja prikaza XML dokumenta sa greškom unutar web pregledača (slika 2.1).

Slika 2.1. Greška prilikom čitanja XML dokumenta

XML dokument koji u potpunosti zadovolji sintaksna pravila jezika drugačije se naziva ispravno formiran dokument. Kako bi XML dokument poneo takav epitet, neophodno je da zadovolji sledeća pravila:

- svaki XML dokument mora imati jedan koreni element;
- svi elementi moraju imati otvarajući i zatvarajući tag ili samozatvarajući tag;
- nazivi tagova mogu biti sastavljeni iz slova, brojeva i karaktera donja crta, srednja crta i tačka;
- nazivi tagova moraju započeti slovom ili karakterom donja crta;
- nazivi tagova ne smeju započeti tekstom xml, XML, Xml, odnosno bilo kojom kombinacijom velikih i malih slova u nizu xml;
- nazivi tagova ne smeju sadržati razmake;
- postavljanje jednog elementa unutar nekog drugog se mora ispravno obaviti; neispravno: `<recipe><title>My Recipe</recipe></title>`; ispravno: `<recipe><title>My Recipe</title></recipe>`;

- vrednosti atributa moraju biti navedene između navodnika: mogu se koristiti jednostruki (') ili dvostruki (") navodnici;
- svaki otvarajući tag mora biti propraćen i zatvarajućim tagom; drugim rečima, ne postoji element koji je sačinjen samo iz otvarajućeg taga (što je slučaj u HTML-u – element `<br>`);
- XML je osetljiv na velika i mala slova, tako da su `<name>` i `<Name>` dva različita taga;
- specijalni karakteri `<` i `&` ne mogu se u svom izvornom obliku navesti kao sadržaj elemenata ili vrednost atributa, već se moraju koristi referentni entiteti.

Ukoliko se unutar XML dokumenta definiše i neki prostor imena, to nazivima elemenata i atributa omogućava da sadrže i jedan karakter koji bez prostora imena ne bi mogli. Reč je o karakteru dve tačke (:). Ipak, u takvim situacijama, imena elemenata i atributa ne smeju da sadrže više od jednog karaktera dve tačke (:).

XML dokument koji zadovoljava navedena pravila naziva se ispravno formiran dokument.

Validan XML dokument

Jedna od osnovnih odlika XML-a tiče se mogućnosti samostalnog definisanja naziva elemenata. To je inače osnovna osobina koja XML razlikuje od HTML-a. Ipak, prilikom kreiranja XML dokumenata postoji mogućnost i da se eksplicitno definišu pravila koja se moraju poštovati prilikom izgradnje njegove strukture. Takva pravila koja se odnose na upotrebu elemenata, njihov raspored, skup dozvoljenih atributa i sl. drugačije se nazivaju validaciona pravila, a samim tim dokument koji ih ispunjava – **validan XML dokument**.

Upravo spomenuta validaciona pravila postoje i unutar HTML-a. Tako je prilikom kreiranja HTML dokumenata unapred poznato koje je elemente moguće koristiti, na koji način se oni mogu kombinovati i koji atributi su dostupni različitim elementima.

Upravo takva validaciona pravila je moguće definisati i za XML dokumente. U našem primeru XML dokumenta sa receptom, jedno takvo pravilo bi moglo da definiše da se element `ingredients` može pojaviti tek nakon pojavljivanja elementa `title`.

Validaciona pravila koja se primenjuju nad XML dokumentima moguće je definisati na dva načina, upotrebom sledećih pojmova:

- Document Type Definition
- XML Schema

Kada koristiti, a kada ne koristiti validaciona pravila?

U prethodnoj lekciji ste mogli videti da se kreiranje XML dokumenta može obaviti i bez prethodnog definisanja bilo kakvih validacionih pravila. To je potpuno legitimno. U takvoj situaciji odgovornost za ispravno kreiranje elemenata i atributa je u potpunosti na onom koji kreira dokument. Drugim rečima, bez unapred definisanih validacionih pravila, XML čitač neće biti u mogućnosti da prepozna grešku ukoliko, recimo, prilikom formiranja elemenata umesto `recipe` napišemo npr. `receipt`.

XML dokumenti bez unapred utvrđenih validacionih pravila često se drugačije nazivaju **DTDless**, a takva je većina jednostavnih XML dokumenata. Ipak, ukoliko se unutar neke aplikacije intenzivno koristi XML za baratanje podacima i ukoliko su takvi podaci nešto složeniji, česta je praksa definisanje validacionih pravila, kako bi sami XML čitači bili u mogućnosti da samostalno provere validnost napisanog ili generisanog XML koda.

DTD (Document Type Definition)

Document Type Definition (DTD) izvorni je način za definisanje validacionih, odnosno gramatičkih pravila XML dokumenta. Korišćenjem DTD-a moguće je eksplicitno definisati elemente koji mogu biti deo dokumenta, njihov raspored, broj pojavljivanja, sadržaj, atribute koje mogu imati...

Slika 2.2. Document Type Definition

Kao primer za demonstraciju mogućnosti DTD-a, biće korišćen sledeći XML dokument:

```
<?xml version="1.0" encoding="UTF-8" ?>
<root>
    <country countryCode="sr">
        <name>Serbia</name>
        <capital>Belgrade</capital>
        <description>Serbia is.....</description>
    </country>
    <country countryCode="fr">
        <name>France</name>
        <capital>Paris</capital>
        <description>France is.....</description>
    </country>
</root>
```

Prikazani XML dokument koristi se za opis jednostavnih podataka država. Koreni element dokumenta je `root`. Elementima `country` predstavljaju se pojedinačne države. Atribut `countryCode` koristi se za definisanje koda države. Elementi `name`, `capital` i `description` su podelementi elementa `country`.

Dokument za definisanje gramatičkih pravila ovakvog XML dokumenta mogao bi da izgleda ovako:

```
<!ELEMENT root (country)*>
<!ELEMENT country (name, capital, description)>
<!ATTLIST country countryCode CDATA #IMPLIED >

<!ELEMENT name (#PCDATA)>
<!ELEMENT capital (#PCDATA)>
<!ELEMENT description (#PCDATA)>
```

Prikazanim kodom obavlja se sledeće:

- definišu se elementi koji se mogu naći unutar XML dokumenta;
- definiše se redosled kojim se elementi mogu pojaviti;
- definiše se sadržaj koji elementi mogu imati;
- definišu se atributi i elementi na kojima se takvi atributi mogu pojaviti.

Kako biste razumeli na koji način je prikazanim DTD kodom obavljeno sve što je navedeno, neophodno je da se upoznamo se pravilima za formiranje DTD koda.

DTD dokument se sastoji iz **deklaracija**. Prikazani DTD poseduje ukupno šest deklaracija. Deklaracijama se definišu elementi i atributi koji se mogu pojaviti unutar XML dokumenta.

DTD deklaracije elemenata

Unutar DTD-a, elementi se definišu deklaracijom koja ima sledeći oblik:

```
<!ELEMENT name content-specifier>
```

- `name` se odnosni na bilo koje validno XML ime, uz pridržavanje pravila koja važe za postavljanje imena elemenata;
- `content-specifier` se odnosni na definiciju elemenata koji se mogu pojaviti unutar elementa.

Prva deklaracija elementa unutar prikazanog DTD-a izgleda ovako:

```
<!ELEMENT root (country)*>
```

Na ovaj način definiše se da će koreni element dokumenta biti element sa nazivom `root`. Pri tome, definiše se i da će element `root` kao svoj sadržaj moći da ima proizvoljan broj elemenata `country`.

Definisanje broja pojavljivanja elementa u primeru se postiže korišćenjem karaktera *, koji specificira da se element `country` unutar elementa `root` može pojaviti proizvoljan broj puta (odnosno nijednom, jednom ili više puta).

Broj pojavljivanja elementa unutar nekog drugog elementa

Navođenje karaktera zvezdica nije jedini način za definisanje broja pojavljivanja elemenata. Načini za definisanje različitog broja pojavljivanja elemenata prikazani su tabelom 2.1.

Broj pojavljivanja	Sintaksa
tačno jedno pojavljivanje	<!ELEMENT element-name (child-name)>
minimalno jedno pojavljivanje	<!ELEMENT element-name (child-name+)>
bilo koji broj pojavljivanja (nijedno, jedno ili više)	<!ELEMENT element-name (child-name*)>
nijedno ili jedno pojavljivanje	<!ELEMENT element-name (child-name?)>

Tabela 2.1. Definisanje različitog broja pojavljivanja elemenata unutar nekog drugog elementa

Drugom deklaracijom unutar DTD-a definiše se element `country`:

```
<!ELEMENT country (name, capital, description)>
```

Za definisanje sadržaja elementa `country` sada je upotrebljen nešto drugačiji pristup. U zagradi su navedeni nazivi tri nova elementa. S obzirom na to da se ne koristi nikakav dodatni karakter koji bi definisao broj pojavljivanja, ovakvi podelementi će moći da se pojave samo jednom.

DTD deklaracija atributa

Pored elemenata, DTD omogućava definisanje i deklaracija atributa. Sintaksa deklaracije atributa izgleda ovako:

```
<!ATTLIST element-name attribute-name attribute-type attribute-value>
```

- `element-name` – naziv elementa na koji će atribut moći da se postavi
- `attribute-name` – naziv atributa
- `attribute-type` – tip atributa
- `attribute-value` – pravila koja se odnose na vrednost atributa

U našem primeru, treća DTD naredba jeste naredba kojom se definiše jedan atribut:

```
<!ATTLIST country countryCode CDATA #IMPLIED>
```

Naziv elementa za koji se definiše postojanje atributa je `country`, a naziv samog atributa je `countryCode`. Preostale dve vrednosti tiču se tipa i vrednosti atributa. Za tip atributa postavljena je vrednost **CDATA**. Ovo znači da će atribut `countryCode` moći da prihvati bilo koju tekstualnu (string) vrednost koja je sintaksno validna u XML jeziku.

Četvrtim parametrom DTD deklaracije atributa utiče se na osobine vrednosti koje će atribut moći da ima. Tako je korišćenjem ovog parametra moguće definisati podrazumevanu ili obaveznu vrednost, ali i da li je definisanje vrednosti opciono ili obavezno.

Definisanje osobina vrednosti atributa

Četvrti parametar DTD deklaracije atributa može imati vrednosti koje su ilustrovane tabelom 2.2.

Vrednost	Opis
default_value	kao vrednost četvrtog parametra moguće je uneti vrednost koja predstavlja podrazumevanu vrednost
#REQUIRED	atribut je neophodan
#IMPLIED	atribut je opcioni
#FIXED value	vrednost atributa je fiksna, odnosno ona koja se navede nakon ključne reči #FIXED

Tabela 2.2. Definisanje osobina vrednosti atributa

U našem primeru je kao vrednost četvrtog parametra DTD deklaracije atributa definisano `#IMPLIED`, što znači da je atribut opcioni. Na kraju, kompletna DTD deklaracija atributa iz našeg primera se može pročitati ovako: na elementu `country` je moguće definisati atribut `countryCode`, koji je opcioni i čija vrednost može biti bilo koji sintaksno ispravni XML tekst.

Poslednje tri DTD deklaracije definišu još tri elementa koja će moći da se pojave unutar XML dokumenta:

- `<!ELEMENT name (#PCDATA)>`
- `<!ELEMENT capital (#PCDATA)>`
- `<!ELEMENT description (#PCDATA)>`

Na ovaj način je rečeno da će XML dokument moći da sadrži i elemente `name`, `capital` i `description`. Ipak, ovoga puta, njihov sadržaj je definisan oznakom **PCDATA**, što je skraćenica za *parsed character data*. Stoga nije teško zaključiti da se na ovaj način definiše da će elementi moći da imaju bilo koji tekstualni sadržaj koji XML čitač može da parsira. Iz prethodne lekcije je poznato da sadržaj XML elemenata ne može da bude parsiran samo ukoliko se unutar njega nađu karakteri manje (`<`) i ampersend (`&`).

Korišćenje DTD-a

U prethodnim redovima prikazano je kreiranje DTD koda koji se može koristiti za validaciju XML dokumenta. Ipak, kako bi se prikazani DTD zaista i koristio za obavljanje takvog posla, njega je neophodno povezati sa XML dokumentom koji želimo da bude validiran. To je moguće obaviti na dva načina:

- smeštanjem DTD-a unutar zasebnog fajla, u kom slučaju je reč o eksternom DTD-u;
- smeštanjem DTD-a unutar istog fajla u kome se nalazi i XML, kada se govori o internom DTD-u.

Eksterni DTD

Eksterni DTD podrazumeva njegovo smeštanje unutar zasebnog fajla, sa `.dtd` ekstenzijom:

```
<!ELEMENT root (country)*>
<!ELEMENT country (name, capital, description)>
<!ATTLIST country countryCode CDATA #IMPLIED >

<!ELEMENT name (#PCDATA)>
<!ELEMENT capital (#PCDATA)>
<!ELEMENT description (#PCDATA)>
```

DTD fajl je potrebno povezati sa XML dokumentom, a za obavljanje takvog posla se koristi DOCTYPE deklaracija:

```
<!DOCTYPE root SYSTEM "country.dtd">
```

DOCTYPE deklaracijom obavlja se identifikovanje DTD dokumenta koji sadrži gramatička pravila za validiranje. `root` se odnosi na naziv elementa iz DTD-a koji će biti korišćen kao koreni element, a `country.dtd` predstavlja putanju na kojoj se nalazi fajl sa DTD kodom.

DOCTYPE deklaraciju je potrebno postaviti na početak dokumenta, nakon eventualnog XML prologa:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE root SYSTEM "country.dtd">
<root>
   <country countryCode="sr">
         <name>Serbia</name>
         <capital>Belgrade</capital>
         <description>Serbia is.....</description>
   </country>
   <country countryCode="fr">
         <name>France</name>
         <capital>Paris</capital>
         <description>France is.....</description>
   </country>
</root>
```

DTD i DOCTYPE

Obratite pažnju da su *document type definition* (DTD) i *document type declaration* (DOCTYPE) dva različita pojma. *Document type definition* odnosi se na validaciona pravila, dok pojam *document type declaration* označava deklaraciju koju je potrebno dodati u XML dokument kako bi se nad dokumentom primenila validaciona pravila definisana DTD-om.

Smeštanje DTD-a unutar eksternog fajla omogućava da se taj fajl upotrebi za validaciju većeg broja XML dokumenata. Ukoliko tako nešto nije potrebno, validaciona DTD pravila se mogu definisati i unutar samog XML dokumenta.

Interni DTD

DTD je moguće integrisati i unutar XML dokumenta. To se obavlja korišćenjem DOCTYPE deklaracije u nešto drugačijem obliku:

```
<!DOCTYPE element
[
   declaration1
   declaration2
   ........
]>
```

Osnovna razlika u odnosu na primer definisanja eksternog DTD-a ogleda se u postojanju tela DOCTYPE deklaracije, koje se definiše između uglastih zagrada – `[]`. Tako kompletan primer XML dokumenta sa internim DTD-om izgleda ovako:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE root [
    <!ELEMENT root (country)*>
    <!ELEMENT country (name, capital, description)>
    <!ATTLIST country countryCode CDATA #IMPLIED >
    <!ELEMENT name (#PCDATA)>
    <!ELEMENT capital (#PCDATA)>
    <!ELEMENT description (#PCDATA)>
]>
<root>
   <country countryCode="sr">
         <name>Serbia</name>
         <capital>Belgrade</capital>
         <description>Serbia is.....</description>
   </country>
   <country countryCode="fr">
         <name>France</name>
         <capital>Paris</capital>
         <description>France is.....</description>
   </country>
</root>
```

XML Schema Definition

Pored DTD-a, za definisanje validacionih pravila XML dokumenata moguće je koristiti još jedan, moderniji pristup – XML schema. XML schema koristi se za opisivanje strukture, sadržaja i semantičkih pravila za kreiranje XML dokumenata. XML šema se drugačije naziva *XML schema definition* ili skraćeno XSD.

Slika 2.3. XML Schema Definition

XML šeme zapravo su XML dokumenti koji moraju da zadovolje validaciona pravila XML schema DTD-a, koji postoji na sledećoj adresi:

```
http://www.w3.org/2001/XMLSchema.dtd
```

Organizacija W3C ponudila je XML šeme kao napredniji način za definisanje validacionih pravila XML dokumenata. Prednosti koje XML šeme donose su:

- za razliku od DTD-a, XML šeme su XML dokumenti, pa se za njihovo kreiranje koriste pravila XML jezika;
- XML šeme podržavaju tipove podataka;
- XML šeme podržavaju prostore imena.

Definisanje validacionih pravila identičnih onim iz prethodnog primera korišćenjem XML šema se može postići na sledeći način:

```
<?xml version="1.0"?>
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="root">
         <xs:complexType>
               <xs:sequence>
                     <xs:element ref="country" minOccurs="0"
maxOccurs="unbounded" />
               </xs:sequence>
         </xs:complexType>
   </xs:element>
   <xs:element name="country">
         <xs:complexType>
               <xs:sequence>
                     <xs:element name="name" type="xs:string" />
                     <xs:element name="capital" type="xs:string" />
                     <xs:element name="description" type="xs:string" />
               </xs:sequence>
               <xs:attribute name="countryCode" type="xs:string"/>
         </xs:complexType>
   </xs:element>
</xs:schema>
```

Kao i prilikom korišćenja DTD-a, i sada je prikazanim XSD dokumentom obavljeno definisanja elemenata, njihovog redosleda i sadržaja i atributa koje mogu imati.

XSD dokumenti uvek započinju korenim elementom `schema`:

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
      ...
</xs:schema>
```

Najčešća je praksa da se na korenom XSD elementu `schema` definiše prostor imena `http://www.w3.org/2001/XMLSchema`, sa prefiksom `xs`. Na taj način se stavlja do znanja da svi elementi koji budu korišćeni u XSD dokumentu pripadaju prostoru imena `http://www.w3.org/2001/XMLSchema`. Upravo zbog toga i svi ostali elementi u XSD dokumentu započinju prefiksom `xs`.

Unutar `schema` elementa, obavlja se definisanje validacionih pravila. Prvo ćemo se pozabaviti načinom na koji se definišu validaciona pravila za kreiranje elemenata.

Deklaracija elemenata korišćenjem XSD-a

XML šema poznaje dva osnovna tipa elemenata: proste i složene (kompleksne).

Prosti elementi su oni koji poseduju tekstualni sadržaj. Kreiraju se korišćenjem sledeće sintakse:

```
<xs:element name = "x" type = "y"/>
```

Značenje definisanih atributa je sledeće:

- `name` – naziv elementa
- `type` – tip vrednosti elementa

Kompleksni elementi su oni koji kao svoj sadržaj mogu posedovati neke druge XML elemente. I oni se kreiraju korišćenjem elementa `element`, ali uz upotrebu još nekih elemenata:

```
<xs:element name = "x">
   <xs:complexType>
         <compositor>
               elements declaration...
         <compositor>
   </xs:complexType>
</xs:element>
```

Definisanje kompleksnih tipova obavlja se postavljanjem elementa `complexType` unutar elementa `element`. Zatim se definiše jedan od specijalnih elemenata koji se drugačije nazivaju kompozitori. Unutar kompozitora se zatim smeštaju i konkretni prosti elementi.

U našem primeru su definisana dva kompleksna i tri prosta elementa. Kompleksni elementi su `root` i `country`, a prosti `name`, `capital` i `description`.

Prvi deklarisani element jeste `root`. Reč je o kompleksnom elementu koji unutar sebe može imati elemente `country`. Pri tome je definisano da broj takvih `country` elementa može biti proizvoljan (odnosno nijedan, jedan ili više). To je postignuto korišćenjem XSD indikatora.

XSD indikatori

XSD indikatorima utiče se na raspored i broj pojavljivanja XML elemenata. Na primer, nešto ranije, prilikom priče o deklarisanju prostih i kompleksnih XSD elemenata, spomenut je pojam kompozitora (*compositors*). Kompozitori su zapravo jedna vrsta indikatora kojima se definiše redosled, ali delimično i broj pojavljivanja elemenata unutar nekog drugog elementa:

- `<all>` – elementi se mogu pojaviti u bilo kom rasporedu, ali samo jednom
- `<choice>` – može se pojaviti samo jedan od svih definisanih elemenata
- `<sequence>` – elementi se mogu pojaviti isključivo definisanim redosledom

Pored kompozitora, kao indikatori se koriste i atributi za definisanje minimalnog i maksimalnog broja pojavljivanja elemenata:

- `maxOccurs` – maksimalni broj pojavljivanja
- `minOccurs` – minimalni broj pojavljivanja

Kao vrednosti atributa `maxOccurs` i `minOccurs` mogu se koristiti brojevi ili odrednica `unbounded`, koja definiše da nema ograničenja u minimalnom ili maksimalnom broju definisanih elemenata.

Podrazumevane vrednosti `maxOccurs` i `minOccurs` atributa su `1`.

Na osnovu priče o XSD indikatorima, može se zaključiti da je prilikom deklaracije `root` elementa obavljeno sledeće: definisano je da se unutar `root` elementa može pojaviti proizvoljan broj `country` elemenata. To je postignuto kombinovanjem `<sequence>` kompozitora i atributa `minOccurs` i `maxOccurs` sa vrednostima `0` i `unbounded`, respektivno.

Kada se govori o deklaraciji `root` elementa, neophodno je primetiti još nešto – `country` element nije deklarisan direktno unutar `root` elementa, već je samo referenciran upotrebom atributa `ref`. Naime, i `country` element je kompleksni tip, te je stoga on deklarisan nakon deklaracije `root` elementa, a unutar `root` elementa je obavljeno samo njegovo referenciranje.

Deklaracija kompleksnog elementa `country` obavljena je po identičnom principu kao i kod elementa `root`. Jedina razlika je ta što su elementi koji predstavljaju sadržaj `country` elementa unutar njega direktno i definisani, a ne referencirani. Reč je o tri prosta elementa (`name`, `capital`, `description`), koja su predviđena kao konkretni nosioci sadržaja.

Deklaracijom `country` elementa je definisano da će takvi elementi morati da kao svoj sadržaj imaju elemente `name`, `capital` i `description`, poštujući takav redosled. Pri tome, svaki element će moći da se pojavi samo jednom. To je postignuto korišćenjem `<sequence>` kompozitora i odsustvom atributa `minOccurs` i `maxOccurs`, koji u takvom slučaju imaju vrednost 1.

Novina prilikom deklarisanja prostih elemenata `name`, `capital` i `description` je upotreba atributa `type`, kojim se definiše tip elementa, pa samim tim i osobine sadržaja koji element može imati.

XSD tipovi

XSD tipovi se mogu koristiti za definisanje tipa sadržaja prostih elemenata i atributa. Neki od najkorišćenijih tipova prikazani su u tabeli ispod.

Tip	Opis	Primer
`xs:string`	tekst	New York, NY
`xs:integer`	ceo broj	+234, –345, 678987
`xs:boolean`	logička vrednost	true, false, 1, 0
`xs:decimal`	decimalni broj	–42.5, 67, 92.34, +54.345
`xs:date`	datum u formatu CCYY-MM-DD	2006-05-05
`xs:time`	vreme u formatu hh:mm:ss-hh:mm	10:27:34-05:00

Tabela 2.3. XSD tipovi

Na kraju, prikazani XSD dokument poseduje i deklaraciju jednog atributa:

```
<xs:attribute name="countryCode" type="xs:string"/>
```

Atributi se deklarišu na isti način kao i prosti elementi, uz razliku upotrebe elementa `attribute` za njihovo deklarisanje. Tako je prikazanom linijom rečeno sledeće: elementi `country` će moći da imaju po tačno jedan atribut `countryCode`, čija će vrednost biti tipa `string`.

XSD kod prikazan u prethodnim redovima potrebno je smestiti unutar zasebnog fajla sa ekstenzijom **.xsd** i referencirati unutar XML dokumenta koji će njime biti validiran. Definisanje XSD validacionog dokumenta unutar XML-a obavlja se definisanjem specijalnog prostora imena:

```
<root xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

   ...

</root>
```

Nad elementom `root` definisan je specijalni prostor imena `http://www.w3.org/2001/XMLSchema-instance`, kome je dodeljen prefiks xsi. Reč je o prostoru imena koji za XML čitače ima specijalno značenje. Kada XML čitač detektuje ovakav prostor imena, on zna da je XML dokument potrebno validirati korišćenjem XML šeme. Gde se takva šema nalazi, definiše se korišćenjem sledećih atributa:

- `schemaLocation` – kada je XSD namenjen tačno određenom prostoru imena;
- `noNamespaceSchemaLocation` – kada XSD nije namenjen nijednom konkretnom prostoru imena.

S obzirom na to da smo kreirali XSD koji nije namenjen nekom konkretnom prostoru imena, proces definisanja XSD validacionog dokumenta će izgledati ovako:

```
<root xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="country.xsd">

   ...

</root>
```

Pored definisanog prostora imena, i atribut `noNamespaceSchemaLocation` za XML čitač ima posebno značenje. On u našem primeru govori parseru gde se nalazi dokument sa definisanim XSD validacionim pravilima.

Kompletan izgled XML dokumenta je:

```
<?xml version="1.0" encoding="UTF-8" ?>
<root
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="country.xsd">
   <country countryCode="sr">
         <name>Serbia</name>
         <capital>Belgrade</capital>
         <description>Serbia is.....</description>
   </country>
   <country countryCode="fr">
         <name>France</name>
         <capital>Paris</capital>
         <description>France is.....</description>
   </country>
</root>
```

Pitanje

Koji je najstariji jezik za definisanje validacionih pravila XML dokumenta?

a) XSD
b) DTD
c) XSTL
d) SGML

Objašnjenje:

Document type definition (DTD) je najstariji jezik za definisanje validacionih – ili, drugim rečima, gramatičkih – pravila XML dokumenta. Ovaj jezik karakterišu stroga sintaksna pravila, koja definišu koji elementi mogu biti deo dokumenta, šta oni mogu sadržati i kakve atribute mogu imati.

Rezime

- XML dokument koji zadovoljava sintaksna pravila jezika naziva se ispravno formiran dokument.
- Svi XML elementi moraju imati otvarajući i zatvarajući tag ili samozatvarajući tag.
- XML tagovi moraju da budu pravilno ugnežđeni.
- Sve vrednosti XML atributa moraju biti navedene između navodnika.
- XML je osetljiv na velika i mala slova.
- Pravila koja se odnose na upotrebu elemenata, njihov raspored, skup dozvoljenih atributa i slično drugačije se nazivaju validaciona pravila.
- XML koji zadovoljava definisana validaciona pravila naziva se validan XML dokument.
- Validaciona pravila koja se primenjuju nad XML dokumentima moguće je definisati upotrebom document type definitiona ili XML schema.
- Document type definition (DTD) je izvorni način za definisanje validacionih pravila XML dokumenta.
- XML schema je moderniji pristup za opis strukture, sadržaja i semantičkih pravila XML dokumenta, a i sama je XML dokument.