XML

Egy általános leíró (Generalized Markup) formátum elve már a '60-as évek elején megjelent, azonban csak az SGML[2] nevű változat vált széleskörűen ismertté, melyet 1986-ban ISO szabványként fogadtak el. Az elterjedést elsősorban a World Wide Web dokumentumnyelve, a HTML[3] kidolgozása gyorsította meg, amely maga is egy SGML forma.

Az SGML teljes eszközkészlete rendkívül összetett, ami nagyon megnehezítette az értelmező programok készítését, így hamarosan megjelent az igény egy egyszerűsített változat összeállítására. Maga a HTML is egy egyszerűsítés, azonban azt kezdetben elsősorban a „kézzel" (azaz nem programmal) történő dokumentumkészítésre tervezték, így ennek érdekében jelentős rövidítéseket tartalmaz. Másrészt, a HTML forma szövegek, dokumentumok összeállítására szolgál, így legtöbb eszköze többnyire felesleges, pl. adatok megadásánál.

A W3 konzorcium 1996-ban 10 tervezési célt határozott meg egy egyszerűsített SGML nyelv megalkotására. Az Extensible Markup Language (XML) első, 1.0-ás változata 1998. február 10-én jelent meg.

A célok között (az XML legyen könnyen használható az Interneten, támogassa az alkalmazások széles körét, stb.) szerepel az SGML-lel való kompatibilitás megőrzése, valamint, hogy programokkal egyszerűen elemezhető és feldolgozható legyen.

XML dokumentumok alapszerkezete

Minden XML dokumentum egy egyszerű szöveges forma, amit mind programmal, mind a legegyszerűbb szövegszerkesztőkkel is létrehozhatunk.

A dokumentum egy XML-prológgal kezdődik, hasonlóan a következőhöz:

```
<?XML version="1.0" encoding="UTF-8" ?>
```

Láthatjuk, hogy a prológ tartalmazza az alkalmazott XML verziót, valamint tartalmazhatja a további szöveg kódolását.

A továbbiakban (megjegyzésektől és egyéb kiegészítőktől eltekintve) pontosan egy elem állhat.

```
<?XML version="1.0" encoding="UTF-8" ?>
<catalog>
  <book id="1091">
    <title>XML - everywhere</title>
    <author>John Smith, Joe Johnson</author>
    <price>27.95</price>
  </book>
  <book id="1341">
    <title>Besides</title>
    <author>George Black</author>
    <price>29.95</price>
  </book>
</catalog>
```

Az *elem* elnevezését "<" jelet követően kell megadnunk. A ">" előtt szerepelhetnek ún. *attribútumok*, mégpedig név és az "=" jelet követő, idézőjelek vagy aposztrófok közötti érték formában.

XML-ben egy név (pl. elem vagy attribútum neve) kis vagy nagybetűvel, illetve aláhúzással kezdődhet, a további karakterek ezeken kívül pedig számjegyek, "." és "-" jelek lehetnek.

A ">" jelet követően egyrészt szövegrészleteket, valamint további elemeket adhatunk meg. Az elemet le kell zárnunk, a "</" és ">" közötti elemnévvel (pl. </catalog>). Ha az elem nem tartalmaz további alelemeket és szöveges elemeket, hanem legfeljebb csak attribútumokat, akkor lehetőségünk van egy összevont, rövidített leírásra (pl. <book id="9999"/>), melyben a zárást jelölő "/" jel a ">" előtt szerepel.

Az XML dokumentumban szerepelhetnek még *megjegyzések* a "<!--" és "-->" jelek között, valamint ún. *feldolgozó* utasítások "<?név" és ">" között (ennek a formátumát követi az XML prológ is). Ha a szövegekben vezérlő jeleket is szeretnénk használni, akkor azok helyett a megfelelő *entitásokat* kell alkalmaznunk (pl. "<" helyett "<", "&" helyett "&").

Hosszabb, vezérlő jeleket is tartalmazó szövegek megadását egyszerűsítik az ún. *CDATA szekciók*; a "<![CDATA[" és "]]>" közötti szövegek esetén nem kell a vezérlőjeleket entitásokkal helyettesítenünk.

XML névterek

XML dokumentumokban gyakran előfordul, hogy többféle adatot szeretnénk egyszerre szerepeltetni. Például egy dátum-jellegű érték megjelenési formáját, vagy egy titkosított szöveg kódolását. Ugyancsak előfordulhat, hogy többféle adat esetén ugyanazok az elemnevek vagy attribútum-nevek fordulnak elő.

A problémára az XML-névterek technikája nyújt megoldást. Minden névtér egy szótárként jelenik meg, így minden nevet egyértelműen azonosítani tudunk az alapján, hogy melyik szótárba tartozik. A megadás eszköze az ún. minősített név, mely esetén a névtér elnevezését követő ":" után adjuk meg a névtéren (mint szótáron) belüli elnevezést. A névterek deklarálása egy speciális, `xmlns` (az XML `namespace` alapján) elnevezésű névtérrel, attribútumként történik, ahol az attribútum értéke egy URI (WWW erőforrás azonosítója).

```
<h:text xmlns:h="http://www.w3.org/TR/REC-html40>
  … <h:b>félkövér</h:b>
  …
</h:text>
```

Példánkban `h` elnevezéssel a HTML 4.0 névtérre hivatkozunk, így a szövegben előforduló `h:b` elemet egyértelműen html-formázásként azonosíthatjuk.

A névtér elnevezését rövidítésként használjuk, s az URI fogja egyedi módon azonosítani a névteret, ezért két névtér elnevezés ugyanarra a „szótárra”, elnevezés-gyűjteményre is vonatkozhat, amennyiben a deklarációjukban az URI megegyezik.

Az `xmlns` elnevezésű attribútummal egy alapértelmezett névteret határozunk meg, amely az összes nem minősített névre érvényes lesz.

```
<html xmlns="http://www.w3.org/TR/REC-html40>
  … <b>félkövér</b>
  …
</html>
```

Dokumentum- és adatorientált XML

Az XML tervezésekor szempont volt, hogy az a HTML-hez hasonló formázott dokumentumok készítésére is alkalmas legyen. Az ilyen dokumentumok a

HTML-nél szigorúbb szabályoknak felelnek meg, ugyanakkor tetszőleges, mind informatív, mind formázó-elemeket is tartalmazhatnak. Ezt a formát dokumentumorientált XML-nek nevezik.

Dokumentumorientált XML-ben egy elemen belül vegyesen fordulhatnak elő mind a szövegek, mind az alelemek. A következő példánkban a `title` elemen belül két szöveges elem között egy "`i`" elem is szerepel.

```
<note>
  <author>Joe</author>
  <title>Writing <i>document oriented</i>  XML</title>
</note>
```

Az XML használatával egyre inkább előtérbe kerültek az adatorientált XML dokumentumok, amelyeket elsősorban programok állítanak elő, vagy programok számára készítenek, illetve (mindkettő esetén) programok közötti kommunikációra használnak. Adatorientált XML a korábbi példánk:

```
<catalog>
  <book id="1091">
    <title>XML - everywhere</title>
    <author>John Smith, Joe Johnson</author>
    <price>27.95</price>
  </book>
  <book id="1341">
    <title>Besides</title>
    <author>George Black</author>
    <price>29.95</price>
  </book>
</catalog>
```

Az adatorientált XML jóval egyszerűbb, szerkezetében sokkal szorosabb szabályokat teljesít:

- Az egymás utáni szöveges elemeket (beleértve az entitásokat és a szövegek közvetlen megadására szolgáló CDATA szekciókat is) egyetlen szöveges elemnek tekintjük.
- Az elem-jelöléseket követő újsor utáni töltőkarakterek (whitespace-ek) figyelmen kívül hagyhatók,
- Egy elemnek vagy csak legfeljebb egy (nem üres) szöveges eleme lehet, vagy csak alelemei lehetnek.
- Az értelmezés során figyelmen kívül hagyhatók a megjegyzések és a feldolgozó utasítások.

XML sémák

Az XML dokumentumok önmagukban sajnos gyakorlatilag alkalmatlanok az önleírásra. A W3C ajánlások eszköztárában az XML-sémáknak[11] a célja az XML dokumentumok külső szerkezeti leírása. Egy XML séma maga is egy XML dokumentum, amely adott névterű adott elemekhez konkrét szemantikát rendel.

A következőkben sorra vesszük az XML sémában definiált elemi típusokat és az alapvető kombinációs eszközöket. A XML séma ismertetésekor az ajánlás szövegét és példáit követjük. A példákban az `xsd` névtér-prefix az XML séma névterére vonatkozik, melyet a `http://www.w3.org/2001/XMLSchema` URI definiál.

Az XML sémák elemi értéktípusai

Az XML sémák elemi típusai a következők:

Szöveges típusok:

- `string` (pl.: `egy szöveg`): tetszőleges szöveg
- `normalizedString` (pl.: `egy szöveg`): a szövegből a többszörös töltő- (whitespace) karakterek egyetlen szóközre vannak cserélve
- `token`: mint a `normalizedString`, de a vezető és záró szóközök elhagyásával

A `normalizedString` és a `token` szabályai részben az XML szövegmegadásainak a sajátosságai miatt szükségesek, részben pedig valamely szövegként ábrázolt elnevezés vagy összetett érték elemzésének megkönnyítésére.

Bináris adattípusok:

- `base64Binary` (pl.: `GpM7`)
- `hexBinary` (pl.: `0FB7`)

A bináris típusok segítségével tetszőleges bináris adatot (pl. képeket, hangokat, stb.) szöveges módon adhatunk meg. Az előfeldolgozó a dekódolással előállíthatja a bináris adatterületet, készen a további feldolgozásra.

Szám típusok:

- `byte` (pl.: `-1`, `126`)
- `unsignedByte` (pl.: `0`, `126`)
- `integer` (pl.: `-126789`, `-1`, `0`, `1`, `126789`)
- `positiveInteger` (pl.: `1`, `126789`)
- `negativeInteger` (pl.: `-126789`, `-1`)
- `nonNegativeInteger` (pl.: `0`, `1`, `126789`)
- `nonPositiveInteger` (pl.: `-126789`, `-1`, `0`)
- `int` (pl.: `-1`, `126789675`)
- `unsignedInt` (pl.: `0`, `1267896754`)
- `long` (pl.: `-1`, `12678967543233`)
- `unsignedLong` (pl.: `0`, `12678967543233`)
- `short` (pl.: `-1`, `12678`)
- `unsignedShort` (pl.: `0`, `12678`)
- `decimal` (pl.: `-1.23`, `0`, `123.4`, `1000.00`)
- `float` (pl.: `-INF`, `-1E4`, `-0`, `0`, `12.78E-2`, `12`, `INF`, `NaN`): egyszeres pontosságú lebegőpontos érték
- `double` (pl.: `-INF`, `-1E4`, `-0`, `0`, `12.78E-2`, `12`, `INF`, `NaN`): duplapontosságú lebegőpontos érték

Ahogy az ajánlás leírása külön is felhívja a figyelmet: azonos értéknek több megadása is lehetséges (`10`, `10.0`, `1E1`, stb.), melyek az érték szöveges formája vagy tárolása esetén természetesen eltérhetnek.

Az XML séma szám-típusai között találunk gépfüggő típusokat (pl. `byte`, `int`, stb.), valamint „alap” típusokat (`integer`, `decimal`).

Valójában csak az egész (`integer`) valamint az általános szám (`decimal`) jelent ténylegesen különböző típusokat, a többi az értéktartományra vonatkozó korlátozással megadható vagy csak a tárolási módra vonatkozó utalást tartalmaz (nem decimálisan, hanem binárisan van tárolva), ami egy technikai megoldás.

Logikai típus:

- `boolean` (pl.: `true`, `false`, `1`, `0`)

Logikai típusok értékének megadásakor a `true` és `false` szövegek mellett engedélyezettek az `1` és `0` karakterek is.

Dátum, idő, időtartam:

- `date` (pl.: `1999-05-31`)
- `time` (pl.: `13:20:00.000`, `13:20:00.000-05:00`)

- `dateTime` (pl.: `1999-05-31T13:20:00.000-05:00`, azaz 1999 május 31, délután 1:20 a -5 órás (Eastern Standard Time) időzónában)
- `duration` (pl.: `P1Y2M3DT10H30M12.3S`, azaz 1 év, 2 hónap, 3 nap, 10 óra, 30 perc, és 12.3 másodperc)
- `gMonth` (pl.: `--05--`, azaz május hónap)
- `gYear` (pl.: `1999`, azaz az 1999. év)
- `gYearMonth` (pl.: `1999-02`, azaz 1999. február hónapja)
- `gDay` (pl.: `---31`, azaz bármely hónap 31-ik napja)
- `gMonthDay` (pl.: `--05-31`, azaz bármely év május 31.)

Megfigyelhetjük, hogy a dátum és időtartam értékei különböző pontossággal is megadhatók, illetve lehetőség van a dátum elemeinek a leírására is. A "`g`" prefixek a Gregorián naptárra utalnak.

Név, nyelv és URI:

- `Name` (pl.: `shipTo`) XML 1.0 név (pl. elem vagy attribútumnév)
- `QName` (pl.: `po:USAddress`) : XML névtér minősített neve
- `NCName` (pl.: `USAddress`): XML névtér lokális neve, azaz egy minősített név a prefix és a kettőspont nélkül
- `anyURI`: URI (pl.: `http://www.example.com/doc.html#ID5`, `http://www.example.com/`)
- `language` (pl.: `en-GB`, `en-US`, `fr`): az XML 1.0-ban definiált `xml:lang` attribútumban megadható nyelvkódok

Ezen típusok az XML dokumentum leírásával kapcsolatosak, annak technikáira, jelöléseire vonatkoznak.

XML 1.0 attribútum-típusok:

- `ID`
- `IDREF`
- `IDREFS`
- `ENTITY`
- `ENTITIES`
- `NOTATION`
- `NMTOKEN`
- `NMTOKENS`

Ezen típusok is az XML dokumentum leírásával kapcsolatosak, annak technikáira, jelöléseire vonatkoznak.

Az XML séma elemi típusaiból további származtatott típusok képezhetők bizonyos *korlátozók* megadásával. A lehetséges korlátozók a következők:

- `length`, `minLength`, `maxLength`: a szöveges megadás pontos, minimális vagy maximális hosszának a megadása
- `pattern`: megadhatjuk (reguláris kifejezésként) a szöveges megadás formátumát
- `enumeration`: felsorolhatjuk elemenként a felvehető értékeket
- `whiteSpace`: megadhatjuk a szóközök előfeldolgozását, ami lehet
  - `preserve`: feldolgozatlanul hagyás
  - `replace`: a töltőkarakterek (beleértve az újsort is) átcserélése szóközökre
  - `collapse`: a `replace` után az ismétlődő, valamint a kezdő és záró szóközök cseréje egyetlen szóközre.
- `maxInclusive`, `maxExclusive`, `minExclusive`, `minInclusive`: megadhatjuk a felvehető értékek tartományának alsó vagy felső határát, a határt beleértve vagy kihagyva (nyílt és zárt intervallumok)
- `totalDigits`, `fractionDigits`: kiköthetjük a számjegyek teljes, illetve a tizedesjegyek maximális számát

Az `enumeration` megadásával gyakorlatilag felsorolásos értéket definiálhatunk. A `pattern` megadásával a formátumot rögzíthetjük. A `whitespace` használata elsősorban valamely szövegként megfogalmazott összetett értékre vagy elnevezések sorozatára utal.

Tekintsünk egy rövid példát, amely egy raktári tételt olyan szövegként definiál, amely három számjegyből, majd egy kötőjelet követően két (nagy)betűből áll:

```
<xsd:simpleType name="SKU">
  <xsd:restriction base="xsd:string">
   <xsd:pattern value="\d{3}-[A-Z]{2}"/>
  </xsd:restriction>
 </xsd:simpleType>
```

A második példa az USA államok listáját határozza meg:

```
<xsd:simpleType name="USState">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="AK"/>
    <xsd:enumeration value="AL"/>
    <xsd:enumeration value="AR"/>
    <!-- and so on ... -->
  </xsd:restriction>
</xsd:simpleType>
```

Az XML sémák „alap” és „másodlagos” típusai

Megfigyelhető, hogy az XML sémák által definiált elemi típusok közül kiválasztható néhány alapvető típus, míg a többi az vagy előállítható valamely alapvető típus korlátozásával, vagy valamely informatikai technológiára utal, pl. meghatározza a tárolási vagy kódolási módot, vagy éppen az XML formátummal kapcsolatos.

Mindezek alapján az XML sémák elemi típusai a következő három csoportba sorolhatók:

- „alap” típusok (pl. `decimal`, `integer`, `date`),
- „másodlagos” típusok, amelyek valamely alaptípus (pl. értéktartomány-) korlátozásával definiáltak (pl. `positiveInteger`), vagy valamely tárolási, ábrázolási technológiával kapcsolatosak (pl. `normalizedString`, `double`, `hexBinary`),
- az XML-lel és Web-es technológiákkal kapcsolatos típusok (pl. `QNAME`, `anyURI`).

Az XML sémák „alap” típusaiként a következőket határozhatjuk meg:

- `string` (szöveg)
- `decimal` (tetszőleges szám)
- `integer` (egész érték)
- `boolean` (logikai)
- `date` és `dateTime` (dátum)
- időtartam és az idővel kapcsolatos további típusok

XML sémák – lista és unió

Az XML sémák lehetőséget adnak a listaként történő megadásra is:

```
<xsd:simpleType name="listOfMyIntType">
  <xsd:list itemType="xsd:integer"/>
</xsd:simpleType>
```

A `listOfMyInt` típusú értékek az egészek töltőkarakterekkel elválasztott sorozatát tartalmazhatja, pl.:

```
<listOfMyInt>20003 15037 95977 95945</listOfMyInt>
```

Mivel a lista elemeinél a szóköz az elválasztójel, ezért – ahogy arra az XML séma ajánlása is felhívja a figyelmet – sajnos nem adhatunk meg „több szóból álló”, azaz szóközöket tartalmazó elemeket. Az ajánlás példája szerint az `Asie Europe Amérique Latine` lista négy elemet tartalmaz, pedig a „Latin Amerika” egyetlen fogalmat jelöl.

A másik alap-kombinációs eszköz az unió, mellyel a lehetséges értékek külön megadott halmazait egyesíthetjük:

```
<xsd:simpleType name="zipUnion">
  <xsd:union memberTypes="USState listOfMyIntType"/>
</xsd:simpleType>
```

A példa szerint a `zipUnion` típusú elem vagy egyetlen USA állam kódja lehet, vagy az egészek listája.

XML sémák – összetett elemek

XML sémák segítségével természetesen összetett elemeket is leírhatunk.

Az elemek képzéséhez az alapvető eszközök a következők:

- `xsd:sequence`: az elemek a megadott sorrendben (felsorolás)
- `xsd:choice`: a felsoroltak közül valamelyik elem (alternatíva)
- `xsd:all`: az elemek tetszőleges sorrendben
- `xsd:group`: hivatkozás egy máshol definiált elemre
- `xsd:element`: XML elem
- `xsd:attribute`: XML attribútum

A következő példa egy címet tartalmaz, amelyben az irányítószám számként (`decimal`) van megadva:

```
<xsd:complexType name="USAddress">
  <xsd:sequence>
   <xsd:element name="name"   type="xsd:string"/>
   <xsd:element name="street" type="xsd:string"/>
   <xsd:element name="city"   type="xsd:string"/>
   <xsd:element name="state"  type="xsd:string"/>
   <xsd:element name="zip"    type="xsd:decimal"/>
  </xsd:sequence>
  <xsd:attribute name="country" type="xsd:NMTOKEN"
     fixed="US"/>
 </xsd:complexType>
```

Az összetett elemek képzésekor az elemek számosságát az `xsd:element` elem `minOccurs`, ill. `maxOccurs` attribútumában adhatjuk meg egy egész értékként vagy az `unbounded`, a tetszőlegességet jelző kulcsszóval. A számosság attribútumainak alapértelmezett értéke "`1`".