XML 1.0

Az angol változat nyersfordítása

Készült az Elektronikus Kereskedelmi Fórumban

A széleskörű hasznosíthatóság érdekében örömmel veszünk minden olyan észrevételt, javaslatot, amely lehetővé teszi, hogy olyan magyar változat készüljön, amelyet minden érdeklődő számára elérhetővé tehetünk

Elektronikus Kereskedelmi Fórum
1555 Budapest, Pf. 79.
Tel: 239-0760/114, Fax: 330-5995
E-mail: info@ecforum.hu , URL: http://www.ecforum.hu

TARTALOMJEGYZÉK

1. XML (Extensible Markup Language) 1.0

Az alábbi dokumentum az XML 1.0 szabvány magyar változatának vázlata. A szabvány alkalmazható magyar változatának elkészítése további széleskörű lektorálást és véleményezést igényel

A jelen dokumentum készültségi állapota

A jelen dokumentum a World Wide Web Consortium tagjai általi felülvizsgálat tárgyát képezi. A dokumentum tartalma stabil, az elmúlt év folyamán végzett XML munkálatok eredményeként kidolgozott munkaokmány tervezetek sorozatából származtatott. A dokumentum egy meglévő, a Világhálón való használathoz kifejlesztett és széles körben használt nemzetközi szabványt (Standard Generalized Markup Language (SGML), a módosított és javított ISO 8879:1986 szabvány) alkészlettel kiegészítő nyelvet határoz meg. Az XML-t már néhány (növekvő számú) kereskedelmi forgalomban is kaphatót termék is támogatja. Az XML-lel kapcsolatos megjegyzése on-line elérhetők.

A jelen specifikáció a [Berners-Lee] által meghatározott URI kifejezést használja, ez a folyamatban lévő munka várhatóan frissíteni fogja az [RFC1738] és [RFC1808] dokumentumokat. Amennyiben ez a munka RFC formájában nem kerül elfogadásra, a jelen dokumentumban az egységes erőforrás azonosítókra (URI) vonatkozó utalások egységes erőforrás lokátorokra vonatkozó utalásokká változnak.

A jelen Előterjesztett Ajánlások átdolgozásához meghatározott időszak 1998. január 5.-én ér véget. Attól a naptól számított 14 napon belül bejelentik a dokumentum elfogadott formáját: a dokumentum átváltozhat W3C ajánlássá (feltehetően kisebb változtatásokat követően), vagy visszaminősülhet Munkaokmány Tervezet állapotra, vagy W3C munkafeladatként való nyilvántartását elvetik. A jelen dokumentum a jelenlegi helyzetben semmiféle kötelezettségvállalást vagy jóváhagyást nem jelent a Konzorciumban tevékenykedő személyzet vagy a tagszervezetek részéről. *(A dokumentum lényegi változtatás nélkül W3C-ajánlássá minősült 1998. Februárjában – a ford.)*

Tartalmi kivonat

A Bővíthető jelölő nyelv (XML) egyszerűen a jelen dokumentumban részletesen és teljes körűen ismertetett SGML egy nyelvjárás változata. A cél egy általános célú SGML rendelkezésre bocsátása, amely ugyanolyan módon kiszolgálható, fogadható és feldolgozható a Világhálón, mint ahogy az jelenleg a HTML használatával lehetséges. Az XML kifejlesztésének a célja a könnyű megvalósíthatóság és mind az SGML, mind a HTML nyelvvel az együttműködési képesség megteremtése és fenntartása.

Bevezetés

A Bővíthető jelölő nyelv – angol nyelvű rövidítése XML - XML dokumentumok elnevezéssel hivatkozott adatobjektum osztályokat ír le és részben leírja az azokat feldolgozó számítógépes programok viselkedését is. Az XML az SGML [ISO8879] szűkített formája. Az XML dokumentumok felépítésükben megfelelnek az SGML dokumentumoknak.

Az XML dokumentumok entitás elnevezéssel hivatkozott tárolási egységekből tevődnek össze, amelyek mind szintaktikusan elemzett (parsed), mind szintaktikusan nem elemzett (unparsed) adatokat tartalmaznak. A szintaktikusan elemzett adatok karakterekből tevődnek össze, ezek némelyike a dokumentum karakteralapú adatait alkotja, mások pedig jelölő funkciót látnak el. A jelölő funkció kódolja a dokumentum tárolóelrendezésének és logikai felépítésének a leírását. Az XML rendelkezik egy olyan mechanizmussal, amely korlátozásokat szab meg a tárolóelrendezés és a logikai felépítés kialakíthatóságában.

XML dokumentumok olvasásához, valamint azok tartalmának és szerkezetének a hozzáférhetővé tételéhez egy szoftver modul, az úgynevezett XML processzor használatos. A feltételezések szerint egy XML processzor a tevékenységeit egy másik modul, az úgynevezett alkalmazás modul helyett és nevében végzi. A jelen specifikáció ismerteti, hogy milyennek kell lennie az XML processzor viselkedésének abból a szempontból, hogy hogyan kell kiolvasnia XML adatokat és hogyan kell az információkat átadnia az alkalmazásnak.

2.1. Az XML eredete és a kitűzött célok

Az XML fejlesztését az 1996-ban a World Wide Web Consortium (W3C) támogatásával létrehozott XML Munkacsoport (eredeti nevén SGML Szerkesztői Felülvizsgáló Testület) végezte el. A testület elnöki tisztét a Sun Microsystems által delegált Jon Bosak töltötte be a szintén a W3C által szervezett XML Szakmai Érdekcsoport (korábban SGML Munkacsoport) aktív részvételével és támogatásával. Az XML Munkacsoport tagságát egy mellékelt függelék ismerteti. A Munkacsoport és a W3C közötti összekötő Dan Connolly volt.

Az XML megtervezésével megvalósítandó célok az alábbiak:

1. Az XML az Interneten közvetlenül használható kell, hogy legyen

2. Az XML-nek az alkalmazások széles körét kell támogatnia

3. Az XML az SGML nyelvvel kompatíbilis kell, hogy legyen

4. Az XML dokumentumok feldolgozásához könnyen megírható programok szükségesek

5. Az XML opcionális szolgáltatásait a lehető legminimálisabb szintre, ideális esetben nullára kell korlátozni

6. Az XML dokumentumoknak emberek által olvashatónak és célszerűen egyértelműnek kell lenniük

7. Az XML tervezésének gyorsan meg kell történnie

8. Az XML kivitelezésének formálisnak és tömörnek kell lennie

9. Az XML dokumentumoknak könnyen létrehozhatónak kell lenniük.

10. Az XML jelölő funkció tömörségének a fontossága minimális jelentőségű

A jelen specifikáció a csatolt szabványokkal (Unicode és ISO/IEC 10646 karakterszabványok, Internet RFC 1766 nyelvazonosító címkék szabványa, ISO 639 nyelv elnevezés kódszabvány és ISO 3166 országnév szabvány) együtt az XML 1.0 verzió megértéséhez és az XML dokumentumok

feldolgozásához szükséges számítógépes programok fejlesztéséhez minden szükséges tájékoztatást megad.

Az XML specifikáció jelen változata nyilvános felülvizsgálat és megvitatás céljára készült. Mindaddig, amíg a jogi rendelkezéseket tiszteletben tartják, a specifikáció korlátozások nélkül szabadon forgalmazható.

2.2. Fogalommeghatározások

Az XML dokumentumok leírásához használt fogalommeghatározásokat a jelen specifikáció törzsszövege tartalmazza. Az alábbi felsorolásban feltüntetett szavak és kifejezések használatosak a meghatározások felépítéséhez és az XML processzor működésének a leírásához:

may (lehetséges)

A szabványt kielégítő dokumentumok és XML processzorok leírás szerinti viselkedése engedélyezett, de nem kötelező érvényű.

must (kötelező)

A szabványt kielégítő dokumentumok és XML processzorok leírás szerinti viselkedése szükséges, ellenkező esetben hibaállapot következik be.

error (hibaállapot)

A jelen specifikáció szabályainak megsértése; a hibaállapot eredménye nem meghatározott. A szabványt kielégítő szoftver felfedezhet és jelenthet, valamint helyreállíthat hibaállapotokat.

fatal error (végzetes hiba)

Olyan hibaállapot, amelyet a szabványt kielégítő XML processzornak fel kell derítenie és jeleznie kell az alkalmazás számára. Végzetes hiba észlelésekor a processzor folytathatja az adatok feldolgozását további hibaállapotok keresése céljából és azokról üzenetet küldhet az alkalmazásnak. A hibaállapotok kijavításának támogatása céljából a processzor a (karakteralapú és jelölő adatokat tartalmazó) dokumentumból feldolgozatlan adatokat bocsáthat az alkalmazás rendelkezésére. Végzetes hiba észlelését követően azonban a processzornak nem szabad folytatnia a normál feldolgozási műveleteket (tehát nem szabad további adatokat és a dokumentum logikai szerkezetére vonatkozó információkat a szokásos módon az alkalmazás számára átadnia).

at user option (a felhasználó tetszése szerinti választás)
Egy szabványt kielégítő szoftver (a mondatban meghatározott végrehajtási utasítás igéjétől függően) a leírás szerint viselkedhet vagy a leírás szerinti viselkedése kötelező; és ha azt megteszi, a felhasználó számára lehetővé kell tennie, hogy a leírás szerinti viselkedést tetszés szerint engedélyezze vagy letiltsa.

validity constraint (érvényességi korlátozás)
Az összes érvényes XML dokumentumra vonatkozó hatályos szabály. Az érvényességi korlátozások megsértése hibaállapotok keletkezésével jár, amelyeket a felhasználó választása szerint érvényességellenőrző XML processzoroknak kell jelenteni.

well-formedness constraint (jól formázottsági korlátozás)
Az összes jól formázott XML dokumentumra vonatkozó hatályos szabály. A jól formázottsági korlátozások megsértése végzetes hiba keletkezésével jár.

match (megegyezés)
(Karakterláncoké vagy neveké:) Két összehasonlításra kerülő karakterláncnak vagy névnek azonosnak kell lennie. A többféle megjelenítésű (tehát mind előre megfogalmazott, mind alapkarakter+ékezet formában jelen lévő) karakterek az ISO/IEC 10646 szabványban csak akkor megegyezők, ha mindkét karakterláncban ugyanazon megjelenítéssel szerepelnek. A felhasználó választása szerint a processzorok az ilyen karaktereket normalizálhatják bizonyos hitelesített formátumúra. Kisbetű/nagybetű egyeztetés (karakterláncokban és a nyelvtanban) nem történik. A karakterlánc akkor egyezik meg (tartalom és tartalomtípus vonatkozásában) egy nyelvtani termékkel, ha a nyelvtani

termék által előállított nyelvhez tartozik. Egy elem akkor egyezik meg a deklaráló utasításával, ha kielégíti az Elemérvényességi korlátozásban előírt módokat.

for compatibility (kompatibilitás céljából)
Olyan szolgáltatás, amelyet az XML kizárólag az XML és az SGML közötti kompatibilitás fenntartása céljából tartalmaz.

for interoperability (együttműködési képesség céljából)
Nem kötelező érvényű ajánlás annak javítása céljából, hogy az XML dokumentumok feldolgozhatók legyenek az ISO 8879 szabvány WebSGML Adaptációk című Függelékének kibocsátását megelőzően már meglévő telepített SGML processzorbázison.

2. Dokumentumok

Egy adatobjektum akkor tekinthető XML dokumentumnak, ha a jelen specifikációban meghatározottak szerint jól formázott (Well formed). Egy jól formázott dokumentum pedig mindezeken túlmenően akkor lehet érvényes, ha bizonyos további korlátozások előírásait is kielégíti.
Minden egyes XML dokumentum rendelkezik logikai és fizikai szerkezettel is. A dokumentum fizikai szerkezete entitás néven hivatkozott egységekből tevődik össze. Egy entitás hivatkozhat egyéb entitásokra is, hogy ezzel azokat kijelölje a dokumentumba való beillesztéshez. A dokumentum egy "gyökérrel", vagy más néven dokumentum-entitással kezdődik. A dokumentum logikai szempontból deklaráló utasításokat, elemeket, megjegyzéseket, karakter hivatkozásokat és feldolgozási parancsokat tartalmaz, amelyek mindegyike a dokumentumban konkrét jelölő funkcióként jelenik meg. A logikai és fizikai szerkezeteknek megfelelőképpen be kell ágyazódniuk, ahogy azt az alábbi részekben ismertetjük.

2.1. Jól formázott XML dokumentumok (Well formed XML Documents)

Egy szöveges objektum akkor lehet jól formázott XML dokumentum, ha:

1. Egészét tekintve megegyezik a dokumentum címkével ellátott termékkel.

2. Kielégíti a jól formázottságnak a jelen specifikációban meghatározott korlátozásait.

3. Minden egyes szintaktikusan elemzett entitása jól formázott.

Dokumentum
[1] document ::= prolog element Misc*

A dokumentum termékkel való megegyezés a következőket jelenti:

1. Egy vagy több elemet tartalmaz.

2. Pontosan egyetlen egy olyan elem van, a gyökér, vagy más néven dokumentum elem, amelynek egyetlen része sem jelenik meg egyetlen másik elem tartalmában sem. Az összes egyéb elem esetében, ha a kezdőcímke másik elemnek is a tartalmában szerepel, a zárócímke is megtalálható ugyanannak az elemnek a tartalmában. Más szóval: a kezdő- és zárócímkékkel behatárolt elemek megfelelő módon egymásba ágyazódnak.

Ennek következményeként minden egyes nem gyökér C elemhez a dokumentumban megtalálható egy másik P elem is olyan módon, hogy a C elem a P elem tartalmában szerepel, azonban semmiféle más olyan elem tartalmában nem található meg, amely a P elem tartalmában jelen van. A P elemre a C elem szülő elemeként, a C elemre pedig a P elem gyermek elemeként hivatkozunk.

2.2. Karakterek

Egy szintaktikusan elemzett entitás tartalmaz szöveget, karakterek olyan sorozatát, amelyek jelölő funkciójú vagy karakteralapú adatokat képviselnek. Egy karakter az ISO/IEC 10646 [ISO10646] szabványban meghatározottak szerint a szöveg elemi része. Érvényesen használható karakterek a “tabulátor”, az “enter”, “line feed”, valamint a Unicode és ISO/IEC 10646 tartalmát képező összes érvényes grafikus karakter.

Karakterkészlet tartomány	
[2] Char ::=#x9 | #xA | #xD | [#x20-#D7FF] | [#xE000-#xFFFD] | [#x10000-#x10FFFF]	/* any Unicode character, excluding the surrogate blocks, FFFE, and FFFF. */

A karakterértékek bitmintákba történő kódolásához használatos mechanizmus entitásonként eltérő lehet. Az összes XML processzornak támogatnia kell a 10646 szabvány UTF-8 és UTF-16 kódolásait; a kettő közül a használatban lévőt jelző, vagy más kódolásokat szerephez juttató mechanizmusokat a jelen specifikáció későbbi részében a karakter kódolások tárgyalásánál ismertetjük.

A használt kódolás típusától függetlenül az ISO/IEC 10646 szabvány karakterkészletének bármelyik karaktere hivatkozható a hozzárendelt UCS-4 kódérték decimális vagy hexadecimális alakjával.

2.3. Közös szintaktikus szerkezetek

A jelen részben meghatározunk egy néhány, a nyelvtanban elterjedten használt szimbólumot.

S (fehér szóköz) egy vagy több szóköz (#x20) karaktert, kocsi vissza, sorváltás vagy tab karaktert tartalmaz.

Fehér szóköz
[3] S ::= (#x20 | #x9 | #xD | #xA)+

A karaktereket kényelmi szempontok miatt betűkként, számjegyenként vagy egyéb karakterekként osztályozzuk. A betűk osztálya ABC-alapú vagy szótag karaktereket tartalmaz, amelyeket egy vagy több kombinációs karakter, vagy egy képírásos (ideografikus) karakter követhet. Az egyes osztályokhoz tartozó karakterek teljes meghatározása a karakterosztályokat leíró függelékben található.

A Név betűvel vagy egy néhány központozási karakter egyikével kezdődő vezérjel, a folytatása betűkből, számjegyekből, elválasztójelekből, aláhúzás jelekből, kettőspont vagy pont karakterekből állhat, amelyek mindegyikét karaktereknek nevezzük. Az "xml" karakterlánccal kezdődő, vagy az (('X'|'x') ('M'|'m') ('L'|'l')) karakterláncoknak bármelyikével megegyező nevek a jelen specifikáció jövőbeli verziói szabványosításainak a céljaira fenntartottak.

Megjegyzés: A kettőspont karakter XML neveken belül név helykijelölésekkel kapcsolatos kísérletezések céljaira fenntartott. A jelentése valamikor a jövőben várhatóan szabványosítva lesz, amikor a kísérleti célokra kettőspontot használó dokumentumokat feltehetően majd frissíteni kell. (Nincs garancia arra nézve, hogy az XML céljaira elfogadott bármiféle név helykijelölő mechanizmus ténylegesen használni fogja a kettőspontot név helykijelölő határoló karakterként.) A gyakorlatban ez azt jelenti, hogy a szerzőknek nem szabad a kettőspontot XML nevekben használniuk olyan esetek kivételével, amikor az a név helykijelölésével történő kísérletezéshez szükséges, ugyanakkor az XML processzoroktól elvárt, hogy a kettőspont használatát névkarakterként el fogadják el.

A Nmtoken (név vezérlőjel) a névkarakterek tetszőleges keverékét jelenti.

Nevek és vezérlőjelek

```
[4] NameChar ::= Letter | Digit | '.' | '-' | '_' | ':' | CombiningChar
                 | Extender
[5]      Name ::=  (Letter | '_' | ':') (NameChar)*
[6]     Names ::=  Name (S Name)*
[7]   Nmtoken ::=  (NameChar)+
[8]  Nmtokens ::=  Nmtoken (S Nmtoken)*
```

Literális, szó szerint értelmezhető adat bármiféle olyan idézett karakterlánc lehet, amely nem tartalmazza a karakterláncot behatároló idézőjeleket. Literális adatok használatosak belső entitások tartalmának (EntityValue), az érték attribútumainak (AttValue), és külső azonosítóinak (SystemLiteral) a meghatározásához. Bizonyos esetekben a teljes literális adat átugorható (SkipLit), a jelölő adatok keresése nélkül:

Literális adatok

```
 [9]  EntityValue ::= '"' ([^%&"] | PEReference | Reference)* '"'
                    | "'" ([^%&'] | PEReference | Reference)* "'"
[10]    AttValue ::= '"' ([^<&"] | Reference)* '"'
                    | "'" ([^<&'] | Reference)* "'"
[11] SystemLiteral ::= SkipLit
[12] PubidLiteral ::= '"' PubidChar* '"' | "'" (PubidChar - "'")* "'"
[13]   PubidChar ::= #x20 | #xD | #xA | [a-zA-Z0-9] | [-'()+,./:=?]
```

2.4. Karakteralapú és jelölő adatok

A szöveg keverten tartalmaz karakteralapú adatokat és jelölő adatokat. A jelölő adatok kezdőcímkéből, zárócímkéből, üres elemekből, entitás hivatkozásokból, karakter hivatkozásokból, megjegyzésekből, CDATA szakasz határolókból, dokumentumtípus deklarációkból és feldolgozási parancsokból tevődnek összes.

Minden olyan szöveg, amely nem jelölő adat, a dokumentum karakteralapú adatait jelenti.

Az "és" karakter (&) és a balra mutató hegyes zárójel (<) a literális formában csak akkor jelenhet meg, amikor jelölő adatok határolójaként használják, illetve megjegyzésben, feldolgozási parancsban, vagy egy CDATA szakaszban. Használata érvényes egy belső entitás deklarációs utasításában szereplő literális entitás értékben is; lásd a jól formázott entitásokról szóló részt. Ha használatára más helyen van szükség, numerikus karakter hivatkozások vagy a "&" és "<" karakterláncok használatával kell kiléptetni. A jobbra mutató hegyes zárójel (>) a ">" karakterlánccal jeleníthető meg és a kompatibilitás megtartása céljából a ">" karakterlánccal kell kiléptetni, illetve egy karakter hivatkozással, ha a "]]>" karakterláncban jelenik meg a tartalomban olyan esetekben, amikor ez a karakterlánc nem egy CDATA szakasz végét jelöli.

Elemek tartalmában a karakteralapú adatok bármilyen karakterekből álló olyan karakterláncot alkothatnak, amelyek nem tartalmazzák egyetlen jelölő adat kezdőcímkéjét sem. CDATA szakaszban a karakteralapú adatok bármilyen karakterekből álló olyan karakterláncot alkothatnak, amelyek nem tartalmazzák a CDATA szakasz záró "]]>" határoló elemét.

Ahhoz, hogy az attribútumértékek tartalmazhassanak egyszeres és kettős idézőjeleket is, az aposztróf, más néven egyszeres idézőjel karakter (') megjelenítéséhez a "'", és a kettős idézőjel karakter (") megjelenítéséhez a """ karakterláncot kell használni.

Karakteralapú adatok
[15] CharData ::= [^<&]* - ([^<&]* ']]>' [^<&]*)

2.5. Megjegyzések

Megjegyzések a dokumentumban bárhol megjelenhetnek más jelölő adatokon kívüli részekben; megjelenhetnek továbbá a nyelvtan által engedélyezett helyeken is a dokumentumtípus deklarációs utasításon belül. A megjegyzések nem alkotják a dokumentum karakteralapú adatainak részét; egy XML processzor lehetővé teheti egy alkalmazás számára a megjegyzések szövegének a kikeresését, ez azonban nem szükségszerű követelmény. A kompatibilitás megőrzése céljából a "--" (kettős elválasztójel) karakternek nem szabad megjelennie a megjegyzéseken belül.

Megjegyzések
[16] Comment ::= '<!--' ((Char - '-') | ('-' (Char - '-')))* '-->'

Példa a megjegyzésekre:

<!-- declarations for <head> & <body> -->

2.6. Feldolgozási parancsok

Feldolgozási parancsok (PI) teszik lehetővé, hogy a dokumentumok az alkalmazásokhoz parancsokat tartalmazzanak.

Feldolgozási parancsok
[16] PI ::= '<?' PITarget (S (Char* - (Char* '?>' Char*)))? '?>'
[17] PITarget ::= Name - (('X' | 'x') ('M' | 'm') ('L' | 'l'))

A feldolgozási parancsok nem részei a dokumentum karakteralapú adatainak, azonban át kell haladniuk az alkalmazáson. A feldolgozási parancs annak az alkalmazásnak az azonosítására használatos célmegjelöléssel (PITarget) kezdődik, amely alkalmazásra a parancs végrehajtása vonatkozik. Az olyan célmegjelölő nevek, mint az "XML", "xml", stb., fenntartottak a jelen specifikáción vagy annak későbbi verzióin végrehajtandó szabványosítás céljaira. Az XML jelölő mechanizmusa használható feldolgozási parancsok céljainak formális deklarációs utasításaiként.

2.7. CDATA szakaszok

CDATA szakaszok bárhol megjelenhetnek, ahol előfordulnak karakteralapú adatok; a CDATA szakaszok használatosak szövegtömbökből való kilépésre, amelyeket a rendszer egyébként jelölő adatokként ismerne fel. A CDATA szakaszok a "<![CDATA[" karakterlánccal kezdődnek és a "]]>" karakterlánccal fejeződnek be:

CDATA szakaszok
[18] CDSect ::= CDStart CData CDEnd
[19] CDStart ::= '<![CDATA['
[20] CData ::= (Char* - (Char* ']]>' Char*))
[21] CDEnd ::= ']]>'

Egy CDATA szakaszon belül csak a CDEnd karakterláncot ismeri fel a rendszer jelölő adatként, tehát balra mutató hegyes zárójel karakter, valamint az "és" (&) karakter csak literális formájában fordulhat elő; ezeket a "<" és "&" karakterláncokkal nem szükséges (és nem is lehet) kiléptetni. A CDATA szakaszok nem beágyazhatók.

A példában látható egy CDATA szakasz, amelyben a rendszer a "<greeting>" és "</greeting>" karakterláncokat karakteralapú adatokként, nem pedig jelölő adatokként ismeri fel:

```
<![CDATA[<greeting>Hello, world!</greeting>]]>
```

2.8. Előszó és a dokumentumtípus deklarációs utasítása

Az XML dokumentumok kezdődhetnek XML deklarációs utasítással, és azzal is kell kezdődniük, ez határozza meg a használt XML verziót.

Az alábbiakban bemutatunk egy teljes, elkészült, jól formázott, azonban nem érvényes XML dokumentumot:

```
<?xml version="1.0"?>
<greeting>Hello, world!</greeting>
```

és ugyanez igaz az alábbira is:

```
<greeting>Hello, world!</greeting>
```

A jelen specifikációban leírt verzióval való megegyezés jelzéséhez az "1.0" verziószámot kell használni; az "1.0" érték használata olyan esetben, amikor a dokumentum nem felel meg ennek a verziónak, hibaállapotot idéz elő a dokumentum használatában. Az XML munkacsoport szándéka, hogy a jelen specifikáció későbbi verzióit más, nem az "1.0" verziószámmal fogja ellátni, ez a szándék azonban nem jelent semmiféle olyan kötelezettségvállalást, amelynek alapján az XML újabb verziója valójában elkészül, sem pedig arra vonatkozót, hogy ha készül újabb verzió, akkor az meghatározott számozási minta szerint lesz megjelölve. Mivel a jövőbeli változatok megjelenésének a lehetősége nem lett kizárva, ez a szerkezet biztosítja az eszközöket ahhoz, hogy az automatikus verzió felismerés szükség esetén lehetséges legyen. A processzorok hibaállapotot jelezhetnek, ha olyan verziókkal megjelölt dokumentumokat fogadnak, amelyeket a processzorok nem támogatnak és ismernek fel.

A jelölő adatok feladata egy XML dokumentumban a tárolási és logikai szerkezet, valamint az attribútumérték párok és a logikai szerkezet közötti társítás leírása. Az XML rendelkezésre bocsát egy mechanizmust, a dokumentumtípus deklarációs utasítást a logikai szerkezet korlátozásainak meghatározásához és előre meghatározott tárolóegységek használatának a támogatásához. Egy XML dokumentum akkor érvényes, ha a hozzá társított dokumentumtípus deklarációs utasítást tartalmaz és a dokumentum kielégíti az abban előírt korlátozások kritériumait.

A dokumentumtípus deklarációs utasításnak a dokumentum legelső eleme előtt meg kell jelennie a dokumentumban.

Előszó

```
[23]     prolog ::= XMLDecl? Misc* (doctypedecl Misc*)?
[24]    XMLDecl ::= '<?xml' VersionInfo EncodingDecl? SDDecl? S? '?>'
[25] VersionInfo ::= S 'version' Eq ("'VersionNum'" | '"VersionNum"')
[26]        Eq ::= S? '=' S?
[27]  VersionNum ::= ([a-zA-Z0-9_.:] | '-')+
[28]       Misc ::= Comment | PI |  S
```

Az XML dokumentumtípus deklarációs utasítás jelölő adatok deklarációs utasításait tartalmazza vagy azokra utal, ezek határozzák meg a dokumentumok osztályai számára a nyelvtant. A nyelvtan a dokumentumtípus meghatározásaként ismert, ennek angol rövidítése DTD. A dokumentumtípus deklarációs utasítása rámutathat egy jelölő adatok deklarációs utasításait tartalmazó külső alkészletre

(egy különleges típusú külső entitásra), vagy a jelölő adatok deklarációs utasításait közvetlenül egy belső alkészlet tartalmazza, vagy mindkettő egyidejűleg is lehetséges. A DTD a dokumentum számára mindkét alkészletet együttesen tartalmazza.

A jelölő adatok deklarációs utasítása lehet egy elemtípus deklarációs utasítás, egy attribútumlista deklarációs utasítás , egy entitás deklarációs utasítás , vagy egy jelölő deklarációs utasítás . Ezeket a deklarációs utasításokat részben vagy egészben az alábbi jól formázottsági és érvényességkorlátozási részben ismertetettek szerint paraméter-entitások tartalmazhatják. Ezzel kapcsolatos további tájékoztatást a fizikai szerkezetet ismertető rész tartalmaz.

Dokumentumtípus definíció	
[28] doctypedecl ::= '<!DOCTYPE' S Name (S ExternalID)? S? ('[' (markupdecl | PEReference | S)* ']' S?)? '>'	[vc: Root Element Type]
[29] markupdecl ::= elementdecl | AttlistDecl | EntityDecl | NotationDecl | PI | Comment	[vc: Proper Declaration/PE Nesting] [wfc: PEs in Internal Subset]

Érvényességkorlátozás - Gyökér elem típusa

A dokumentumtípus deklarációs utasításban szereplő névnek meg kell egyeznie a gyökér elem típusával.

Érvényességkorlátozás - Deklarációs utasítás / paraméter-entitások megfelelő beágyazása

A jelölő adatok deklarációs utasításainak megfelelően be kell ágyazódniuk paraméter-entitást helyettesítő szövegbe. Ez tulajdonképpen azt jelenti, hogy ha a jelölő adatok deklarációs utasításának (a fenti jelölése markupdecl) első vagy utolsó karakterét egy paraméter-entitás hivatkozás helyettesítő szövege tartalmazza, mindkettőnek ugyanabban a helyettesítő szövegben szerepelnie kell.

Jól formázottsági korlátozás - Paraméter-entitások a belső alkészletben

A belső DTD alkészletben paraméter-entitás hivatkozások csak akkor fordulhatnak elő, ha jelölő adatok deklarációs utasításai is előfordulhatnak, a jelölő adatok deklarációs utasításain belül azonban nem. (Ez nem vonatkozik a külső paraméter-entitásokban vagy külső alkészletben előforduló hivatkozásokra.)

Ugyanúgy, mint a belső alkészlet esetében, a külső alkészletnek és a DTD tartalmában hivatkozott bármely külső paraméter-entitásnak tartalmaznia kell jelölő adatok nem terminál típusú markupdecl szimbólum által engedélyezett, fehér szóközökkel vagy paraméter-entitás hivatkozásokkal megszakított teljes deklarációs utasítássorozatát. Ugyanakkor a feltételes szakasz szerkezet használatakor a külső alkészlet vagy a külső paraméter-entitások tartalma részben vagy egészben teljesen figyelmen kívül hagyható; ez a belső alkészlet esetében nem engedélyezett.

Külső alkészlet	
[30] extSubset ::= TextDecl? extSubsetDecl	
[31] extSubsetDecl ::= (markupdecl | conditionalSect | PEReference | S)*	

A külső alkészlet és a külső paraméter-entitások abban is különböznek a belső alkészlettől, hogy a paraméter-entitás hivatkozásokat a rendszer a jelölő adatok deklarációs utasításain belül is képes felismerni, nem csak a jelölő adatok deklarációs utasításai között.

Az alábbiakban bemutatunk egy példát egy XML dokumentumra egy dokumentumtípus deklarációs utasítással:

```
<?xml version="1.0"?>
<!DOCTYPE greeting SYSTEM "hello.dtd">
<greeting>Hello, world!</greeting>
```

A "hello.dtd" rendszerazonosító megadja a dokumentumhoz a DTD egységes erőforrás azonosítóját (URI).

A deklarációs utasítások lokálisan is megadhatók, amint az az alábbi példában is látható:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE greeting [
  <!ELEMENT greeting (#PCDATA)>
]>
<greeting>Hello, world!</greeting>
```

Ha mind a külső, mind a belső alkészlet használatban van, a belső alkészletnek a külső alkészlet előtt kell megjelennie. Ennek hatására a belső alkészlet entitás- és attribútumlistái elsőbbséget élveznek a külső alkészletével szemben.

2.9. Önálló dokumentum deklarációs utasítása

Jelölő adatok deklarációs utasításai gyakran befolyásolják a dokumentum tartalmát olyankor, amikor a tartalom átadódik egy XML processzorról egy alkalmazásra; erre példák az alapértelmezett attribútumok és az Entitás deklarációs utasítások. Az önálló dokumentum deklarációs utasítása, amely az XML deklarációs utasítás alkotóelemeként jelenhet meg, jelezheti, hogy a dokumentumot nem befolyásolják jelölő adatok deklarációs utasításai (talán azért, mert nincsenek is).

Önálló dokumentum deklarálása

```
[32] SDDecl ::= S 'standalone' Eq '"' ('yes'
                    | 'no') '"' | "'" ('yes' | 'no') "'"  [vc: Standalone Document Declaration ]
```

Egy önálló dokumentum deklarációs utasításban szereplő "igen" érték arra utal, hogy a dokumentum-entitáson kívüli részen nincsenek jelölő adatoknak deklarációs utasításai (sem a DTD külső alkészletében, sem pedig olyan külső alkészletből hivatkozott külső paraméter-entitás, amely befolyásolja az XML processzorról az alkalmazásra átadásra kerülő információkat. A "nem" érték arra utal, hogy vannak vagy lehetnek ilyen külső deklarációs utasítások. Megjegyzendő, hogy az önálló dokumentum csak külső deklarációs utasítások jelenlétét írja le; belsőleg külső entitások jelenlétére utaló hivatkozások a dokumentumban, nem változtatják meg a dokumentum önálló jellegét.

Ha nincsenek külső jelölő adat deklarációs utasítások, az önálló dokumentum deklarációs utasításnak nincs értékelhető jelentése. Ha vannak külső jelölő adat deklarációs utasítások de nincs önálló dokumentum deklarációs utasítás, a "nem" érték érvényessége feltételezett.

Minden olyan XML dokumentum, amelyben az önálló dokumentumra utaló "nem" érték az érvényes, átváltható önálló dokumentummá, ami bizonyos hálózatkiszolgáló alkalmazások esetében lehet kívánatos.

Érvényességkorlátozás - Önálló dokumentum deklarációs utasítás

Az önálló dokumentum deklarációs utasításnak "nem" értékkel kell rendelkeznie olyankor, ha a külső jelölő adat deklarációs utasítások bármelyike az alábbiak valamelyikének deklarációs utasítását tartalmazza

alapértelmezett értékkel rendelkező attribútumok, ha a hozzájuk rendelt elemek a dokumentumban az attribútumok értékének a megadása nélkül jelennek meg a dokumentumban, vagy

entitások (az amp, lt, gt, apos és quot kivételével), ha az entitásokra utaló hivatkozások megjelennek a dokumentumban, vagy

normalizálásnak alávetendő értékkel rendelkező attribútumok, ha az ilyen attribútum a dokumentumban a normalizálás végrehajtása eredményeként megváltozó értékkel jelenik meg, vagy

elemtartalommal rendelkező elemtípusok, ha azok bármiféle előfordulása esetén az elemtartalom közvetlenül fehér szóközt tartalmaz.

Példa egy önálló dokumentum deklarációs utasítással rendelkező XML deklarációs utasításra:

```
<?xml version="1.0" standalone='yes'?>
```

2.10. Fehér szóköz kezelése

XML dokumentumok szerkesztésekor gyakran kényelmes megoldásnak bizonyul (a jelen specifikációban leírt nem szóvégi S által leírt szóköz, tab vagy üres sor karakterből álló) "fehér szóközök" alkalmazása a jelölő adatok könnyebb olvashatóság céljából történő különválasztása céljából. Az ilyen fehér szóközök használata a kibocsátott változatban jellemző módon a szándékok szerint nem követendő. Másrészről a kibocsátott változatban megőrizendő "lényeges" fehér szóközök használata, például költészetben és a forráskódban, általánosan elterjedt gyakorlat.

Az XML processzornak minden nem jelölő adatnak minősített karaktert át kell futtatnia az alkalmazáson. Egy érvényesség ellenőrző XML processzor processzornak kell megkülönböztetnie az elemtartalomban szereplő fehér szóközt egyéb nem jelölő adatoknak minősülő karakterektől és jeleznie kell az alkalmazás felé, hogy az elemtartalomban szereplő fehér szóköznek nincs jelentősége.

Egy különleges attribútum, az "xml:space" beillesztésével jelezhető a dokumentumokban, hogy az attribútumhoz társított elem a fehér szóköz használatát az alkalmazások által lényeges fehér szóközként igényli.

Érvényes dokumentumokban, ez az attribútum, minden más attribútumhoz hasonlóan deklarálandó ha használatára sor kerül. A deklarációs utasításban az attribútumot felsorolásba vett típusként kell meghatározni, amelynek lehetséges felvehető értékei csak "alapértelmezett" és "védett" értékek lehetnek.

Az "alapértelmezett" érték arra utal, hogy ehhez az elemhez az alkalmazások alapértelmezett fehér szóköz feldolgozási módszerei az elfogadhatók; a "védett" érték pedig arra a szándékra utal, hogy az alkalmazásoknak meg kell őrizniük az összes fehér szóközt. Ez a deklarált szándék alkalmazandó minden olyan elem esetében, amelynek tartalmán belül az attribútum deklarált, mindaddig, amíg annak érvényét az "xml:space" attribútum egy ismételt előfordulása nem hatálytalanítja

A dokumentumok gyökér eleme nem jelzi az alkalmazások szóköz kezelésével kapcsolatos szándékokat, kivéve, ha a gyökér elem határozza meg az attribútum értékét, vagy az attribútum alapértelmezett értékkel deklarált.

Például:

```
<!ATTLIST poem   xml:space (default|preserve) 'preserve'>
```

2.11. Sor vége karakter kezelése

A szintaktikusan ellenőrzött (parsed) XML entitások gyakran olyan számítógépes fájlokban tárolódnak, amelyek a szerkesztés kényelmesebbé tétele céljából sorokba szervezettek. Az ilyen sorokat jellemző módon a CR (#xD) és LF (#xA) karakterek valamilyen kombinációja választja el egymástól.

Az alkalmazások feladatainak a leegyszerűsítése céljából minden olyan esetben, amikor egy külső szintaktikusan elemzett entitás vagy egy belső szintaktikusan elemzett entitás a két karakteres "#xD#xA" literális entitás értéket, vagy egy önálló literális #xD értékét tartalmazza, az XML processzornak egyetlen #xA karaktert kell küldenie az alkalmazás számára (Ez a viselkedés kényelmesen előidézhető olyan módon, hogy a szintaktikus elemzést megelőzően minden sor vége karaktert bevitelkor #xA karakterré kell normalizálni.)

2.12. Nyelv azonosítása

A dokumentumok feldolgozásában gyakran hasznos annak a természetes vagy formális nyelvnek az azonosíthatósága, amelyben a tartalom megírásra került.

Ez az XML dokumentum bármely elemének tartalmában vagy attribútum értékeiben használt nyelv meghatározásához egy különleges "xml:lang" attribútum beszúrásával érhető el. Az attribútum felvehető értékei a "Nyelvazonosító címkék" [RFC1766] című dokumentumban meghatározott nyelvazonosító kódok:

Nyelv azonosítása

```
[33] LanguageID ::= Langcode ('-' Subcode)*
  [34]   Langcode ::= ISO639Code |  IanaCode |  UserCode
  [35] ISO639Code ::= ([a-z] | [A-Z]) ([a-z] | [A-Z])
  [36]   IanaCode ::= ('i' | 'I') '-' ([a-z] | [A-Z])+
  [37]   UserCode ::= ('x' | 'X') '-' ([a-z] | [A-Z])+
  [38]    Subcode ::= ([a-z] | [A-Z])+
```

A Langcode értéke az alábbiak valamelyike lehet:

- a "Kódok a nyelvek megnevezéseinek megjelenítéséhez" című [ISO639] szabványban előírt kétbetűs nyelvkód

- az Internet Szám Hozzárendelési Hatóság (IANA) szervezeténél nyilvántartásba vett nyelvazonosító; ezek "i-" (vagy "I-") előtaggal kezdődnek

- a felhasználó által hozzárendelt vagy felek közötti megállapodás szerint belső használatra elfogadott nyelvazonosító; ezeknek "x-" vagy "X-" előtaggal kell kezdődniük annak biztosítás céljából, hogy ne ütközhessenek a később szabványosításra kerülő vagy az IANA szervezete által nyilvántartásba vett nevekkel

Tetszőleges számú alkód szegmens alkalmazható; ha van első alkód szegmens és az alkód két betűt tartalmaz, annak a "Kódok a nyelvek megnevezéseinek megjelenítéséhez" című [ISO3166] szabványban előírt országkódnak kell lennie. Ha az első alkód kettőnél több betűt tartalmaz, olyan esetek kivételével, amikor a Langcode attribútum előtagja "x-" vagy "X-", az IANA szervezeténél nyilvántartásba vett alkódnak kell lennie.

Szokásos a nyelvkód kisbetűs és az országkód (ha használatra kerül) nagy betűs írásmódja. Megjegyzendő, hogy ezek az értékek az XML dokumentumokban használt egyéb nevekkel ellentétben megkülönböztetik a kis-/és nagybetűs írásmódokat.

Például:

```
<p xml:lang="en">The quick brown fox jumps over the lazy dog.</p>
<p xml:lang="en-GB">What colour is it?</p>
<p xml:lang="en-US">What color is it?</p>
<sp who="Faust" desc='leise' xml:lang="de">
 <l>Habe nun, ach! Philosophie,</l>
 <l>Juristerei, und Medizin</l>
 <l>und leider auch Theologie</l>
 <l>durchaus studiert mit heißem Bemüh'n.</l>
 </sp>
```

Az xml:lang attribútum használatával jelzett szándék mindaddig érvényes marad a meghatározó elem tartalmában hivatkozott minden más elem vonatkozásában, amíg annak érvényességét az xml:lang attribútum újbóli használata nem hatálytalanítja.

Érvényes dokumentumokban, ezt az attribútumot a jelen specifikáció más részeiben ismertetett módon kell meghatározni; a tipikus deklarációs utasítás a következő:

```
xml:lang  NMTOKEN  #IMPLIED
```

ugyanakkor szükség esetén sajátos alapértelmezett értékek is adhatók. Angol hallgatók számára széljegyzetekkel és megjegyzésekkel angol nyelven megírt francia versgyűjteményben például az xml:lang attribútum a következőképpen deklarálható:

```
   <!ATTLIST poem   xml:lang NMTOKEN 'fr'>
   <!ATTLIST gloss  xml:lang NMTOKEN 'en'>
   <!ATTLIST note   xml:lang NMTOKEN 'en'>
```

3. Logikai szerkezetek

Minden egyes XML dokumentum tartalmaz egy vagy több olyan elemet, amelyek határait kezdőcímke és zárócímke zárja le, illetve üres elemek esetében a határolóelem egy üres-elem címke. Minden egyes elem névvel azonosított típussal rendelkezik, ezeket bizonyos esetekben "általános azonosítónak" (GI) nevezik, és rendelkezhetnek egy attribútum specifikációkészlettel is. Minden egyes attribútum specifikáció névvel és a értékkel ellátott.

```
Elem
[39] element ::= EmptyElemTag
                 | STag content ETag          [ wfc: Element Type Match ]
                                               [ vc: Element Valid ]
```

A jelen specifikáció nem tartalmaz az elemtípusok és attribútumok szemantikájával, használatával vagy elnevezésével kapcsolatos korlátozásokat (a szintaktikai korlátozásokon kívül), azzal a megkötéssel azonban, hogy az (('X'|'x')('M'|'m')('L'|'l')) karakterekkel megegyezően kezdődő nevek szabványosítás, vagy a jelen specifikáció jövőbeli változatai céljaira fenntartottak.

Jól formázottsági korlátozás - Elemtípus egyeztetés

Egy elem zárócímkéjében szereplő névnek meg kell felelnie a kezdőcímkében szereplő elemtípusnak.

3.1. Kezdőcímke, zárócímke, és üres elem címkék

Minden nem üres XML elem kezdete kezdőcímkével ellátott.

Kezdőcímke
[40] STag ::= '<' Name (S Attribute)* S? [wfc: Unique Att Spec] '>' [41] Attribute ::= Name Eq AttValue [vc: Attribute Value Type] [wfc: No External Entity References] [wfc: No < in Attribute Values]

A kezdő- és zárócímkében szereplő név megadja az típusát. A Name-AttValue párra az elem attribútum specifikációiként hivatkozunk, ahol a név minden egyes párban az attribútum névre utaló hivatkozás, az AttValue értéke (a ' vagy " határolóelemek közötti karakterek) pedig az attribútum értéket jelenti.

Jól formázottsági korlátozás - Egyedi attribútum specifikációk

Ugyanabban a kezdőcímkében vagy üres-elem címkében attribútum név nem szerepelhet egynél többször.

Érvényességkorlátozás - Attribútum értéktípus

Az attribútumnak deklaráltnak kell lennie; az érték az attribútumhoz deklarált típusnak meg kell, hogy feleljen. (Az attribútum típusokkal kapcsolatos tájékoztatás az attribútum-lista deklarációs utasítások című részben található.)

Jól formázottsági korlátozás - Nem lehet külső entitásra hivatkozás

Attribútum értékek nem tartalmazhatnak külső entitásokra vonatkozó közvetett vagy közvetlen hivatkozásokat.

Jól formázottsági korlátozás - Nem < karakter az attribútum értékekben

Egy attribútum értékben (a < attribútum kivételével) közvetve vagy közvetlenül hivatkozott entitások helyettesítő szövegében nem lehet < karakter.

Egy példa a kezdőcímkére:

```
<termdef id="dt-dog" term="dog">
```

Minden egyes kezdőcímkével rendelkező elemet a kezdőcímkében megadott elemtípust tükröző nevet tartalmazó zárócímkével kell ellátni:

Zárócímke

```
[42] ETag ::= '</' Name S? '>'
```

Egy példa a zárócímkére:

```
</termdef>
```

A kezdőcímke és zárócímke között elhelyezkedő szöveget az elem tartalmának nevezzük:

Content of Elements
[43] content ::= (element | CharData | Reference | CDSect | PI | Comment)*

Ha egy elem üres, azt vagy egy közvetlenül egy zárócímkét megelőző kezdőcímkével, vagy pedig egy üres elem címkével kell jelezni. Az üres elem címke sajátos formátumú:

Üres elemek címkéi
[44] EmptyElemTag ::= '<' Name (S Attribute)* S? [wfc: Unique Att '/>'Spec]

Üres elem címkék bármely tartalom nélküli elemhez felhasználhatók, függetlenül attól, hogy használunk-e ÜRES meghatározást a deklarációban.

Példák az üres elemekre:

```
<IMG align="left"
 src="http://www.w3.org/Icons/WWW/w3c_home" />
<br></br>
<br/>
```

3.2. Elemtípus deklarációs utasítások

Egy XML dokumentum elem szerkezete érvényesség ellenőrzés céljából az elemtípus és attribútum-lista deklarációs utasítások megadására korlátozható.

Az elemtípus deklarációs utasítás korlátozza az elem tartalmát.

Az elemtípus deklarációs utasítások gyakran korlátozzák, hogy milyen elemtípusok jelenhetnek meg az adott elem gyermek elemeként. A felhasználó tetszése szerint kiválasztható, hogy az XML processzor adjon-e figyelmeztető jelzést olyankor, amikor a deklarációs utasítás olyan elemtípusra hivatkozik, amelyhez nem áll rendelkezésre deklarációs utasítás, ez azonban nem jelent hibaállapotot.

Az elemtípus deklarációs utasítás formátuma a következő:

Element Type Declaration
[45] elementdecl ::= '<!ELEMENT' S Name S [vc: Unique Element Type contentspec S? '>' Declaration]
[46] contentspec ::= 'EMPTY' | 'ANY' | Mixed [vc: Element Valid] | children

ahol a név adja meg a deklarálandó elemtípust.

Érvényességkorlátozás - Egyedi elemtípus deklarációs utasítás

Semmiféle elemtípus nem deklarálható egynél többször.

Példák az elemtípus deklarációs utasításokra:

```
<!ELEMENT br EMPTY>
<!ELEMENT p (#PCDATA|emph)* >
<!ELEMENT %name.para; %content.para; >
<!ELEMENT container ANY>
```

Érvényességkorlátozás - Elem érvényesség

Egy elem érvényes, ha az elemdecl attribútumnak megfelelő deklarációs utasítással rendelkezik, ahol a név egyeztetett az elemtípussal és az alábbi feltételek valamelyike érvényesül:

A deklarációs utasítás ÜRES állapotra utal és az elem nem rendelkezik tartalommal.

A deklarációs utasítás gyermek állapotra utal és a gyermek elemek a tartalom típusában található szabályos kifejezés által létrehozott nyelvhez tartoznak.

The deklarációs utasítás kevert állapotra utal és a tartalom olyan karakteralapú adatokat és gyermek elemeket tartalmaz, amelyek típusa megfelel a tartalom típusában szereplő neveknek.

A deklarációs utasítás BÁRMILYEN állapotra utal, és minden gyermek elem típusa deklarált.

3.2.1 Elemtartalom

Egy elem típus akkor rendelkezik elemtartalommal, ha az adott elemtípus elemei csak gyermek elemeket tartalmazhatnak (karakteralapú adatokat nem). Ebben az esetben a korlátozás tartalmaz egy tartalomtípust, egy egyszerű nyelvtant, amely vezérli az engedélyezett gyermek elemtípusokat, valamint egy sorrendet, amely meghatározza, hogy azok milyen sorrendben jelenhetnek meg. A nyelvtan is tartalomrészecskékre (cp) épül, ezek nevekből, a tartalomrészecskék választéklistáiból, vagy a tartalomrészecskék sorrendi listáiból állnak:

Elemtartalom

```
[47] children ::= (choice | seq) ('?' | '*'
                  | '+')?
[48]      cp ::= (Name | choice | seq) ('?' | '*'
                  | '+')?
[49]  choice ::= '(' S? cp ( S? '|' S? cp )* S?        [ vc: Proper Group/PE
                 ')'                                   Nesting ]
[50]     seq ::= '(' S? cp ( S? ',' S? cp )* S?        [ vc: Proper Group/PE
                 ')'                                    Nesting ]
```

ahol minden egyes név egy gyermek elemként megjeleníthető elemtípust jelent. A választéklistában szereplő bármelyik tartalomrészecske megjelenhet az elemtartalomban azon a helyen, ahol a választéklista megjelenik a nyelvtanban; a sorrendi listában megjelenő tartalomrészecskék mindegyikének a listában meghatározott sorrendben kell megjelennie az elemtartalomban. A nevet vagy listát követő opcionális karakter vezérli, hogy az elem vagy a listában szereplő tartalomrészecskék egyszer vagy többször (+), nullaszor vagy többször (*), illetve nullaszor vagy egyszer (?) fordulhatnak elő. A szintaxis és a jelentés ugyanaz, mint a jelen specifikációban található termékek esetében.

Egy elem tartalma akkor és csak akkor feleltethető meg egy tartalomtípusnak, ha kijelölhető egy olyan útvonal a tartalomtípuson keresztül, amely megfelel a sorrendiségi, választék és ismétlési kívánalmaknak, és a tartalomban szereplő minden egyes elem megfelel a tartalomtípusban szereplő valamelyik elemtípusnak. Ha a dokumentumban egy elem a tartalomtípusban szereplő valamelyik elemtípussal egynél több alkalommal megegyezik hibaállapot következik be. További tájékoztatás a determináns tartalomtípusokat leíró függelékben található.

Érvényességkorlátozás – Csoport/Paraméter-entitás megfelelő beágyazása

A Paraméter-entitás helyettesítő szövegébe megfelelően be kell ágyazni a zárójeles csoportokat. Ez azt jelenti, hogy ha akár a nyitó, akár a záró zárójelet egy választéklista, sorrendlista vagy kevert szerkezetben egy paraméter-entitás helyettesítő szövege tartalmazza, mindkettőnek ugyanabban a helyettesítő szövegben kell lennie. A kölcsönös működtethetőség fenntartása érdekében ha egy paraméter-entitás hivatkozás megjelenik egy választéklista, sorrendlista vagy kevert szerkezetben, annak helyettesítő szövege nem lehet üres és annak sem az első, sem az utolsó nem szóköz karaktere nem lehet összekötő jel (| vagy ,).

Példák a elem tartalomtípusokra:

```
<!ELEMENT spec (front, body, back?)>
<!ELEMENT div1 (head, (p | list | note)*, div2*)>
<!ELEMENT dictionary-body (%div.mix; | %dict.mix;)*>
```

3.2.1.Kevert tartalom

Egy elemtípus akkor rendelkezik kevert tartalommal, amikor a típushoz tartozó elemek tartalmazhatnak karaktereket, opcionálisan megszakítva gyermek elemekkel. Ebben az esetben a gyermek elemek típusai korlátozottak lehetnek, ez azonban a sorrendjükre vagy az előfordulásuk számára nem vonatkozik:

Kevert tartalom deklarálása

```
[51] Mixed ::= '(' S? '#PCDATA' (S? '|' S? Name)*
                S? ')*'
                 | '(' S? '#PCDATA' S? ')'              [ VC: Proper Group/PE
                                                          Nesting ]
                                                        [ VC: No Duplicate Types ]
```

ahol a nevek adják meg a gyermek elemekként megjeleníthető elemek típusát.

Érvényességi korlátozás – Nem lehetnek kettőzött típusok

Egyetlen kevert tartalom meghatározáson belül ugyanannak a névnek nem szabad egynél többször megjelennie.

Példák a kevert tartalom deklarációra:

```
<!ELEMENT p (#PCDATA|a|ul|b|i|em)*>
<!ELEMENT p (#PCDATA | %font; | %phrase; | %special; | %form;)* >
<!ELEMENT b (#PCDATA)>
```

3.3. Attribútumlista deklarációk

Az attribútumok a név-érték párok elemekhez társításához használatosak Az attribútum specifikációk csak kezdőcímkékben és üres elem címkékben jelenhetnek meg; tehát a felismerésükre használatos termékekről tájékoztatás a kezdőcímkék megvitatásánál található. Az attribútumlista deklarációk az alábbiakra használhatók:

Adott elemtípushoz tartozó attribútumkészlet meghatározásához.

Típuskorlátozások meghatározásához az attribútumok számára.

Az attribútumok alapértelmezett értékeinek a meghatározásához.

Az attribútumlista deklarációk határozzák meg egy adott elemtípushoz társított minden egyes attribútum nevét, adattípusát és alapértelmezett értékét (ha van):

Attribútumlista deklaráció

```
[52] AttlistDecl ::= '<!ATTLIST' S Name AttDef* S? '>'
[53]     AttDef ::= S Name S AttType S Default
```

Az AttlistDecl szabályban szereplő név egy elem típusát jelenti. A felhasználó választása szerint egy XML processzor figyelmeztetést adhat olyankor, amikor attribútumot deklarál olyan elemhez, amely saját maga nem deklarált, ez azonban nem hibaállapot. Az AttDef szabályban az attribútum neve található.

Amikor egy adott elemtípushoz egy vagy több AttlistDecl szabályt adnak meg, azok tartalma összefésülődik. Amikor egy adott elemtípusnál egy vagy több meghatározást rendelnek hozzá ugyanazon attribútumhoz, az első deklarációs utasítás lesz a kötelező érvényű, a többit a rendszer figyelmen kívül hagyja. A közös működtethetőség fenntartása érdekében a DTD írók dönthetnek úgy, hogy legfeljebb egy attribútumlista deklarációt rendelnek hozzá egy adott elemtípushoz és legfeljebb egy attribútum meghatározást egy adott attribútumhoz, valamint legalább egy attribútum definíciót minden egyes attribútumlista deklarációhoz. A közös működtethetőség fenntartása érdekében a felhasználó választása szerint egy XML processzor figyelmeztetést adhat olyankor, amikor egy adott elemtípushoz egynél több attribútumlista deklarált, vagy ha egy adott attribútumhoz egynél több attribútum meghatározás tartozik, ez azonban nem hibaállapot.

3.3.1.Attribútumtípusok

Az XML attribútumtípusok háromfélék lehetnek: karakterlánc típusúak, vezérlőjel készlet típusúak, felsorolásba vett típusúak. A karakterlánc alapú attribútum bármiféle literális karakterláncot felvehet értékként; a vezérlőjel készlet típusúak különféle lexikális és szemantikai korlátozásokkal rendelkezhetnek, amint azt már említettük:

Attribútumtípusok

```
[54]     AttType ::= StringType | TokenizedType
                   | EnumeratedType
[55]  StringType ::= 'CDATA'
[56] TokenizedType ::= 'ID' [ vc: ID ]
                                              [ vc: One ID per
                                              Element Type ]
                                              [ vc: ID Attribute
                                              Default ]
                     | 'IDREF'          [ vc: IDREF ]
                     | 'IDREFS'                [ vc: IDREF ]
                     | 'ENTITY'                [ vc: Entity Name ]
                     | 'ENTITIES'              [ vc: Entity Name ]
                     | 'NMTOKEN'               [ vc: Name Token ]
                     | 'NMTOKENS'              [ vc: Name Token ]
```

Érvényességi korlátozás – Azonosító érték (ID)

Az ilyen típusú értékeknek meg kell egyezniük a név típusával. A név egynél többször nem jelenhet meg egy XML dokumentumban ilyen típusú értékként; tehát az azonosító értékeknek egyedi módon azonosítaniuk kell a hozzájuk rendelt elemeket.

Érvényességi korlátozás – Elemtípusonként egyetlen azonosító

Semmiféle elem nem rendelkezhet egynél több meghatározott attribútum azonosítóval.

Érvényességi korlátozás – Alapértelmezett azonosító attribútum

Az azonosító attribútum alapértelmezett értéke #IMPLIED (értelemszerű) vagy #REQUIRED (megkövetelt).

Érvényességi korlátozás - IDREF

Az IDREF típus értékeinek meg kell felelniük a Név hivatkozás eredményének és az IDREFS típus értékeinek meg kell felelniük a Nevek hivatkozás eredményének; minden egyes Névnek az XML dokumentum valamelyik pontján egyeznie kell egy azonosító (ID) attribútummal; tehát az IDREF értékeknek meg kell egyezniük valamilyen azonosító ID attribútummal.

Érvényességi korlátozás - Entity Name

Az ENTITY típusú értékeknek meg kell egyezniük a Név hivatkozás eredményével és az ENTITIES típusú értékeknek meg kell egyezniük a Nevek hivatkozás eredményével; minden egyes Névnek meg kell egyeznie egy, a DTD-ben deklarált szintaktikusan nem elemzett entitás nevével.

Érvényességi korlátozás – a NMTOKEN típusú Név vezérjel értékeknek tartalmazniuk kell egy nem szóvégi Nmtoken karakterláncot; és a NMTOKENS típus értékeinek meg kell egyezniük a Nmtokens értékeivel.

Az XML processzornak az alkalmazásra történő átadást megelőzően az attribútumértékek normalizálása című részben ismertetettek szerint normalizálniuk kell az attribútumértékeket.

A felsorolásba vett attribútumok a deklarációs utasításban megadott értéklista egyik értékét vehetik fel. Két felsorolásba vett attribútumtípus van:

Felsorolásba vett attribútumtípusok

```
[57] EnumeratedType ::= NotationType | Enumeration
[58]   NotationType ::=   'NOTATION' S '(' S? Name  [ VC: Notation
                              (S? '|' Name)* S? ')'Attributes ]
[59]    Enumeration ::=    '(' S? Nmtoken (S? '|' S?  [ vc: Enumeration ]
                           Nmtoken)* S? ')'
```

Érvényességi korlátozás – Jelölő attribútumok

Az ilyen típusú értékeknek a deklarációs utasításban található jelölésnevek egyikével kell megegyezniük.; a deklarációs utasításban található minden egyes jelölésnevet deklarálni kell.

Érvényességi korlátozás – Felsorolásba vétel

Az ilyen értéktípusoknak meg kell egyezniük a deklarációs utasításban szereplő egyik Nmtoken vezérlőjel értékével.

A közös működtethetőség fenntartása érdekében ugyanazon Nmtoken egynél többször nem jelenhet meg egyetlen elemtípus felsorolásba vett attribútumtípusaiban.

3.3.2.Attribute Defaults

Egy attribútum deklarációs utasítás tájékoztatást nyújt arról, hogy az attribútum jelenléte szükséges-e vagy sem, illetve hogyan kell az XML processzornak reagálnia akkor, amikor egy deklarált attribútum a dokumentumban nem található meg.

Alapértelmezett attribútumok
[60] Default ::= '#REQUIRED' | '#IMPLIED' | (('#FIXED' S)? AttValue)[VC: Required Attribute] [VC: Attribute Default Legal] [WFC: No < in Attribute Values] [VC: Fixed Attribute Default]

Érvényességi korlátozás – Érvényes alapértelmezett attribútum

A deklarált alapértelmezésnek ki kell elégítenie a deklarált attribútum lexikális korlátozásaiban előírtakat.

A #REQUIRED érték azt jelenti, hogy a dokumentum nem érvényes, ha a processzor a kérdéses elemtípushoz olyan kezdőcímkét talál, amely nem határoz meg értéket ennek az attribútumnak. Az #IMPLIED érték azt jelenti, hogy ha az attribútum hiányzik egy ilyen típusú elemből, az XML processzornak tájékoztatnia kell az alkalmazást arról, hogy érték nem lett meghatározva; ez az alkalmazás viselkedésével kapcsolatosan nem fejt ki korlátozó hatást.

Ha az attribútum sem #REQUIRED, sem #IMPLIED értékkel nem rendelkezik, akkor az AttValue értéke tartalmazza a deklarált alapértelmezett értéket. Ha a #FIXED érték jelen van, a dokumentum érvénytelen, amennyiben az alapértelmezettől eltérő értékű attribútum van jelen. Ha alapértelmezett érték deklarált, amikor egy XML processzor attribútum hiányát észleli, úgy kell viselkednie, mintha az attribútum a deklarált alapértelmezett értékkel jelen lenne.

Példa az attribútumlista deklarációkra:

```
<!ATTLIST termdef
      id      ID      #REQUIRED
      name    CDATA   #IMPLIED>
<!ATTLIST list
      type    (bullets|ordered|glossary)  "ordered">
<!ATTLIST form
      method CDATA    #FIXED "POST">
```

3.3.3.Attribútumérték normalizációja

Attribútum értéknormalizálás

Mielőtt egy attribútumérték átadásra kerülne az alkalmazás számára, egy, XML processzornak azt az alábbiakban ismertetettek szerint normalizálnia kell:

- Először a sor vége vagy határokat jelölő karakterláncokat (bizonyos rendszerekben határoló rekordokat) kell az attribútum értékben és minden hivatkozott entitásban kicserélni egyszeres szóköz karakterekre (#x20). (Lásd a sor vége karakter kezelése című részben is.)

- Másodszor a karakter hivatkozásokat és belső szintaktikusan elemzett hivatkozásokat ki kell terjeszteni. Küldő entitásokra vonatkozó hivatkozások hibaállapotot idéznek elő.

- Végezetül, ha az attribútum nem CDATA típusú, minden fehér szóköz karakterláncot normalizálni kell egyszeres szóköz karakterekké (#x20), valamint a kezdő és záró fehér szóközöket el kell távolítani

A deklarációs utasításokkal nem rendelkező attribútumot nem érvényes szintaktikusan elemzett attribútumnak kell tekinteni, mintha a CDATA jelentette volna.

3.4. Feltételes szakaszok

A feltételes szakaszok a dokumentumtípus deklarációs utasításának külső alkészlet típusai, amelyeket az őket vezérlő kulcsszó alapján a DTD logikai szerkezete tartalmaz vagy nem tartalmaz.

Feltételes szakasz

```
[61]   conditionalSect ::= includeSect | ignoreSect
[62]       includeSect ::= '<![' S? 'INCLUDE' S? '[' extSubset ']]>'
[63]        ignoreSect ::= '<![' S? 'IGNORE' S? '[' ignoreSectContents* ']]>'
[64] ignoreSectContents ::= Ignore ('<![' ignoreSectContents ']]>' Ignore)*
[65]            Ignore ::= Char* - (Char* ('<![' | ']]>') Char*)
```

Ugyanolyan módon, mint a belső és külső DTD alkészletek, a feltételes szakasz is tartalmazhat egy vagy több teljes deklarációs utasítást, megjegyzést, feldolgozási parancsot vagy beágyazott feltételes szakaszokat fehér szóközökkel összekeverve.

Ha a feltételes szakasz kulcsszava "INCLUDE" (tartalmazza),akkor a processzornak úgy kell kezelnie a feltételes szakaszt, mint a dokumentum egy részét. Ha a kulcsszó "IGNORE" (figyelmen kívül hagy), akkor a feltételes szakasz tartalmát a processzor nem kezeli úgy, mint a dokumentum egy részét. Megjegyzendő, hogy megbízható szintaktikus elemzés elvégzéséhez még a figyelmen kívül hagyott feltételes szakaszok tartalmát is olvasni kell beágyazott feltételes szakaszok keresése céljából, és meg kell győződni arról, hogy a legkülső (figyelmen kívül hagyott) feltételes szakasz megfelelő felderítése sikeres volt-e. Ha egy nagyobb, "IGNORE" kulcsszóval jelzett feltételes szakaszon belül egy "INCLUDE" kulcsszóval jelzett feltételes szakasz jelenik meg, a processzor mind a külső, mind a belső feltételes szakaszt figyelmen kívül hagyja.

Ha a feltételes szakasz kulcsszava egy paraméter-entitás hivatkozása, a paraméter-entitást a tartalmával kell felváltani, mielőtt a processzor eldönti, hogy bevegye vagy hagyja figyelmen kívül a feltételes szakaszt.

Egy példa:

```
<!ENTITY % draft 'INCLUDE' >
<!ENTITY % final 'IGNORE' >

<![%draft;[
<!ELEMENT book (comments*, title, body, supplements?)>
]]>
<![%final;[
<!ELEMENT book (title, body, supplements?)>
]]>
```

4. Fizikai szerkezetek

Egy XML dokumentum tartalmazhat egy vagy több virtuális tárolóeszközt. Ezeknek a neve entitás; mindegyikük rendelkezik tartalommal és (az alábbiakban elmagyarázásra kerülő dokumentum-entitáson és külső DTD alkészleten kívül) névvel azonosított. Minden egyes XML dokumentum

rendelkezik egy dokumentum-entitásnak nevezett entitással, amely az XML processzor kiindulási pontja és az egész dokumentumot tartalmazhatja.

Entitások lehetnek szintaktikusan elemzettek és szintaktikusan nem elemzettek. Egy szintaktikusan elemzett entitás tartalmára annak helyettesítő szövege utal, ez a szöveg a dokumentum elválaszthatatlan részének tekintendő.

Egy szintaktikusan nem elemzett entitás olyan erőforrás, amelynek tartalma lehet vagy nem lehet szöveges is, amennyiben pedig szöveges, akkor nem lehet XML. Minden egyes szintaktikusan nem elemzett entitás rendelkezik névvel azonosított jelöléssel. Azon az elváráson túlmenően, hogy egy XML processzornak a jelölés nevét és társított azonosítóit az alkalmazás rendelkezésére kell bocsátania, az XML egyéb megkötéseket nem tartalmaz a szintaktikusan nem elemzett entitások vonatkozásában.

A szintaktikusan elemzett entitások kikeresése entitás hivatkozásokat használó nevek szerint történik; a szintaktikusan nem elemzett entitásoké pedig az ENTITY vagy ENTITIES attribútumok értékében megadott név alapján.

Az általános entitások szintaktikusan elemzett entitások a dokumentum tartalmán belüli használathoz. A jelen specifikációban az általános entitásokra történő utalás bizonyos esetekben nem minősített entitás terminológia használatával történik, amikor az kétértelműséget nem idéz elő. A paraméter-entitások szintaktikusan elemzett entitások a DTD-n belüli használathoz. Ez a két különböző entitástípus Ez a két entitás típus eltérő hivatkozási formákat használ és azok felismerése és azonosítása egymástól különböző szövegkörnyezetekben történik.

4.1. Karakter és entitás hivatkozások

A karakter hivatkozás az ISO/IEC 10646 karakterkészlet egy bizonyos karakterére utal, például olyanra, amely az adatbeviteli eszközökről közvetlenül nem érhető el.

Karakter hivatkozás
[66] CharRef ::= '&#' [0-9]+ ';' | '&#x' [0-9a-fA-F]+ ';' [wfc: Legal Character]

Jó formázottsági korlátozás – Karakter érvényesség

A karakter hivatkozásokat használó karaktereknek nem szóvégi Char parancs szerint érvényesnek kell lenniük.

Ha a karakter kezdete "&#x", akkor az érték a záró ";" karakterig terjedően az ISO/IEC 10646 karakterkészlet egy tagjának a hexadecimális megjelenítését írja le. Ha a karakter kezdete "&#", akkor a záró ";" karakterig terjedően a karakter decimális értékét írja le.

Az entitás hivatkozások névvel ellátott entitások tartalmára utalnak. Az általános entitásokra utaló hivatkozások “és”, valamint pontosvessző (;) karaktereket használnak határolóelemként.

A paraméter-entitások határolóelemei a százalékjel (%)és a pontosvessző (;).

Entitás hivatkozás	
[67] Reference ::= EntityRef | CharRef	
[68] EntityRef ::= '&' Name ';'	[WFC: Entity Declared] [VC: Entity Declared] [WFC: Parsed Entity] [WFC: No Recursion]
[69] PEReference ::= '%' Name ';'	[VC: Entity Declared] [WFC: No Recursion] [WFC: In DTD]

Jól formázottsági korlátozás – Entitás meghatározottsági státusza

Olyan dokumentumban, amely semmiféle DTD-t nem tartalmaz, vagy csak egy belső DTD alkészletet tartalmazó dokumentumban amelyben nincsenek paraméter-entitás hivatkozások, illetve "standalone='yes'" jelzővel ellátott dokumentumokban az entitás hivatkozásban megadott Név értéknek meg kell egyeznie a deklarációs utasításban megadott névvel, kivéve, a jól formázott dokumentumokban , ahol az amp, lt, gt, apos és quot entitások egyikét sem kell meghatározni. A paraméter-entitás deklarációs utasításának meg kell előznie az entitásra utaló minden hivatkozást. Hasonlóképpen egy általános entitás deklarációs utasításának is meg kell előznie az entitásra utaló minden olyan hivatkozást, amely egy attribútumlista deklarációs utasításának alapértelmezett értékében megjelenik. Megjegyzendő, hogy ha az entitások deklarációja külső alkészletben vagy külső paraméter-entitásokban történik, akkor egy érvényesség ellenőrzést nem végző processzor nem köteles azok deklarációs utasításait olvasni és feldolgozni; az ilyen dokumentumok esetében a deklaráltság előírása nem tekintendő a jól formázottsági korlátozás kritériumának.

Érvényességi korlátozás – Entitás deklaráltsági státusza

Külső alkészlettel vagy külső paraméter-entitással rendelkező, "standalone='no'" jelzővel ellátott dokumentumokban az entitás hivatkozásban megadott Névnek meg kell egyeznie az entitás deklarációs utasításában megadott névvel. A közös működés fenntarthatósága érdekében az érvényes dokumentumoknak az amp, lt, gt, apos és quot entitásokat meg kell határozni az előre meghatározott entitások című részben ismertetettek szerint. Hasonlóképpen egy általános entitás deklarációjának meg kell előznie a rá utaló hivatkozásokat az attribútumlista deklarációs utasításának egy alapértelmezett értékében.

Jól formázottsági korlátozás – Entitás szintaktikus elemzési státusza

Az entitás hivatkozás nem tartalmazhatja szintaktikusan nem elemzett entitás nevét.

Szintaktikusan elemzett entitások csak az ENTITY vagy ENTITIES attribútum értékekben meghatározott értékekkel hivatkozhatók.

Jó formázottsági korlátozás – Visszautalás tilalma

Egy szintaktikusan elemzett entitás nem tartalmazhat sem közvetlenül, sem közvetve saját magára visszautaló hivatkozást.

Jó formázottsági korlátozás – A DTD terjedelmén belüli státusz

A paraméter-entitás hivatkozások csak a DTD terjedelmén belül jelenhetnek meg.

Példák a karakter és entitás hivatkozásokra:

```
Type <key>less-than</key> (&#x3C;) to save options.
 This document was prepared on &docdate; and
 is classified &security-level;.
```

Példa egy paraméter-entitás hivatkozásra:

```
<!-- declare the parameter entity ”ISOLat2”... -->
<!ENTITY % ISOLat2
       SYSTEM "http://www.xml.com/iso/isolat2-xml.entities" >
<!-- ... now reference it. -->
%ISOLat2;
```

4.2. Entitás deklarációs utasítások

Az entitások deklarálása az alábbiak szerint történik:

Entitás deklarálása

```
[70] EntityDecl ::= GEDecl | PEDecl
[71]     GEDecl ::= '<!ENTITY' S Name S EntityDef S? '>'
[72]     PEDecl ::= | '<!ENTITY' S '%' S Name S PEDef S? '>'
[73]  EntityDef ::= EntityValue | ExternalID NdataDecl?)
[74]      PEDef ::= EntityValue | ExternalID
```

Egy entitást az entitás hivatkozásban, szintaktikusan nem elemzett entitás esetében az ENTITY vagy ENTITIES attribútum értékében megadott Név szerint lehet azonosítani. Ha ugyanaz az entitás egynél többször meg van határozva, az első megtalált deklaráció a kötelező érvényű. A felhasználó kívánsága szerint az XML processzor figyelmeztetést adhat, ha entitások sokszorosan deklaráltak.

4.2.1.Belső entitások

Ha az entitás meghatározása EntityValue, a meghatározott entitást belső entitásnak nevezzük. Külön fizikai tároló objektum nincs, és az entitás tartalmát a deklaráció írja le. Megjegyzendő, hogy az entitás és karakter hivatkozások némelyik feldolgozásához a literális értékben el kell készíteni a helyettesítő szöveget: lásd a belső entitás helyettesítő szöveg elkészítése című részben.

A belső entitások szintaktikusan elemzett entitások.

Példa egy belső entitás deklarálásához:

```
<!ENTITY Pub-Status "This is a pre-release of the specification.">
```

4.2.2.Külső entitások

Ha egy entitás nem belső entitás, akkor külső entitás, Ennek a deklarálása a következőképpen történik:

Külső entitás deklarálása
[75] ExternalID ::= 'SYSTEM' S SystemLiteral | 'PUBLIC' S PubidLiteral S SystemLiteral [77] NDataDecl ::= S 'NDATA' S Name [vc: Notation Declared]

Ha a NDataDecl paraméter jelen van, akkor ez egy szintaktikusan nem elemzett entitás, ellenkező esetben szintaktikusan elemzett entitás.

Érvényességi korlátozás – Jelölő státusz deklarált

A Névnek egyeznie kell a jelölés deklarált nevével.

A SYSTEM kulcsszót követő SystemLiteral paramétert nevezzük az entitás rendszerazonosítójának. Ez egy URI, amelynek a használatával az entitás kikereshető. Megjegyzendő, hogy a kettős kereszt karakter ("#") és az egységes erőforrás azonosítókkal (URI) gyakran használt töredékazonosító hivatalosan nem képezi az URI részét; az XML processzor hibát jelezhet, ha egy töredékazonosító a rendszerazonosító részeként van megadva. A jelen specifikáció terjedelemén kívülről (például egy bizonyos DTD által meghatározott sajátos XML elemtípusban meghatározott forrásból, vagy bizonyos alkalmazás specifikációkban meghatározott feldolgozási parancsokból) származó ellentétes értelmű rendelkezések hiányában a relatív egységes erőforrás azonosítók annak az erőforrásnak a helyére utalnak, amelyben az adott entitás deklaráció megtalálható. A belső DTD alkészleten belüli entitás deklarációs utasításokban elhelyezkedő relatív URI-k ilyen módon a dokumentumnak arra a helyére utalnak, ahol az ilyen alkészletek tartózkodnak; az entitás deklarációban a külső alkészletre utalók a külső alkészletet tartalmazó fájlok helyére.

A rendszerazonosítón túlmenően a egy külső azonosító tartalmazhat egy nyilvános azonosítót is. Az entitás tartalmát kikeresni próbáló XML processzor a nyilvános azonosító használatával kísérelheti meg egy alternatív URI létrehozását. Ha a processzor nem képes ezt megtenni, a rendszer literálisban meghatározott URI-t kell használnia. Az egyeztetés megkísérlését megelőzően a nyilvános azonosítóban előforduló minden fehér szóköz karakterláncot egy karakteres szóközzé (#x20) kell normalizálni, valamint a kezdő és záró fehér szóközöket el kell távolítani.

Példa külső entitások deklarálására:

```
<!ENTITY open-hatch
      SYSTEM "http://www.textuality.com/boilerplate/OpenHatch.xml">
<!ENTITY open-hatch
      PUBLIC "-//Textuality//TEXT Standard open-hatch boilerplate//EN"
      "http://www.textuality.com/boilerplate/OpenHatch.xml">
<!ENTITY hatch-pic
      SYSTEM "../grafix/OpenHatch.gif"
      NDATA gif >
```

4.3. Szintaktikusan elemzett entitások

4.3.1.A szöveg deklarálása

Külső szintaktikusan elemzett entitások mindegyike kezdődhet szöveg deklarálással.

Szöveg deklarálás
[77] TextDecl ::= '<?xml' VersionInfo? EncodingDecl S? '?>'

Megjegyzendő, hogy a szöveg deklarálást literálisan kell elvégezni, nem pedig egy szintaktikusan elemzett entitásra való hivatkozással.

Szöveg deklarálása nem jelenhet meg sehol máshol, csak külső szintaktikusan elemzett entitás előtt.

4.3.2.Jól formázott szintaktikusan elemzett entitások

A dokumentum-entitás jól formázott, ha megfelel a dokumentum címkével jelzett paraméternek. Egy külső, általános szintaktikusan elemzett entitás jól formázott, ha megfelel az ExtParsedEnt címkéjű paraméternek. Egy külső paraméter-entitás jól formázott, ha megfelel az ExtPE címkével ellátott paraméter értékének.

Jól formázott külső szintaktikusan elemzett entitás
[78] extParsedEnt ::= TextDecl? content [79] extPE-entity ::= TextDecl? extSubset

Egy belső, általános, szintaktikusan elemzett entitás jól formázott, ha helyettesítő szövege megegyezik a tartalom címkével ellátott paraméter értékével. Megjegyzendő, hogy ez mindaddig megbízhatóan nem tesztelhető, amíg a DTD el nem készül. Minden belső paraméter-entitás a meghatározása által jól formázott.

Az entitások jól formázottságának a következménye, hogy az XML dokumentum logikai és fizikai szerkezete megfelelő módon beágyazott; nincsenek olyan kezdőcímkék és zárócímkék, üres elem címkék, elemek, megjegyzések, feldolgozási parancsok, karakterhivatkozások vagy entitás hivatkozások, amelyek az egyik entitásban kezdődnek és másik entitásban záródnak be.

4.3.3.Karakterkódolás entitásokban

Minden egyes külső szintaktikusan elemzett entitás egy XML dokumentumban különböző kódolásokat használhat a karakterekhez. Minden XML processzornak képesnek kell lennie mind UTF-8, mind UTF-16 kódolású karakterek olvasására.

Az UTF-16 kódolású karaktereknek az ISO/IEC 10646 szabvány E Függelékében és a Unicode szabvány B Függelékében leírt Bájt-sorrend jellel (ZERO WIDTH NO-BREAK SPACE karakter, #xFEFF) kell kezdődniük. Ez egy kódolási aláírás, nem része sem a jelölő adatoknak, sem a karakteralapú adatoknak az XML dokumentumban. Az XML processzoroknak ezzel a karakterrel meg kell tudniuk különböztetni az UTF-8 kódolású dokumentumokat az UTF-16 kódolású dokumentumoktól.

Jóllehet, egy XML processzornak csak UTF-8 és UTF-16 kódolású entitásokat kell tudnia olvasni, felismerték azt a tényt, hogy világszerte másféle kódolásokat is használnak, és kívánatos lehet, hogy az XML processzorok az azokat a kódolásokat használó entitásokat is olvasni tudják. A nem UTF-8 vagy UTF-16 kódolású szintaktikusan elemzett entitásoknak egy szöveges deklarációval kell kezdődniük, amely tartalmazza a kódolási deklarációt:

Kódolási deklaráció

```
 [80] EncodingDecl ::= S 'encoding' Eq (' " ' EncName
                                ' " ' | " ' " EncName " ' " )
 [81]     EncName ::= [A-Za-z] ([A-Za-z0-9._]
                               | '-') *
/* Encoding name contains only Latin characters */
```

A dokumentum-entitásban a kódolási deklaráció az XML deklaráció részét képezi. Az EncName paraméter tartalmazza a használt kódolás nevét.

Egy kódolási deklarációban az UTF-8, UTF-16, ISO-10646-UCS-2, és ISO-10646-UCS-4 jelölések használandók a Unicode / ISO/IEC 10646 különféle kódolásainak és transzformációinak a jelzése céljaira, az ISO-8859-1,ISO-8859-2, ... ISO-8859-9 értékek az ISO 8859 szabvány részeire utaló jelzések, továbbá az ISO-2022-JP, Shift_JIS, és EUC-JP használandó JIS X-0208-1997 szabvány különféle változatainak a jelzéséhez. Az XML processzorok képesek felismerni másféke kódolásokat is; ajánlatos, hogy az Internet Szám Hozzárendelő Hatósága (IANA) által (karakterkészletekként) nyilvántartásba vett, tehát nem csak a felsorolásba belevett kódolásokra a hivatkozások a bejegyzett neveken történjenek. Megjegyzendő, hogy ezek a bejegyzett nevek kisbetűs/nagybetűs írásmódra érzéketlenekként lettek nyilvántartásba véve, tehát az azokat összehasonlítani kívánó processzoroknak az összehasonlítást kisbetűs/nagybetűs írásmódra érzéketlen módon kell végrehajtaniuk.

Egy kódolási deklarációt tartalmazó entitás részéről hiba, ha a deklarációban megnevezettől eltérő kódolást juttat el a processzorhoz, vagy egy kódolási deklaráció részéről, ha máshol jelenik meg, és nem egy külső entitás elején.

Egy olyan entitás, amely sem Bájt-sorrend jellel, sem kódolási deklarációval nem rendelkezik, bizonyára UTF-8 kódolású.

Ha egy XML processzor olyan entitást talál, amelynek kódolását nem képes feldolgozni, köteles erről az alkalmazást tájékoztatni és beszüntetni a feldolgozást, ugyanúgy, mint egy végzetes hiba keletkezésekor.

Példák a kódolási deklarációkra:

```
<?xml encoding='UTF-8'?>
<?xml encoding='EUC-JP'?>
```

4.4. Az XML Processzor entitás és hivatkozás kezelése

Az alábbi táblázat összefoglalja azokat a szövegösszefüggéseket, amelyek használatával karakterhivatkozások, entitás hivatkozások megtétele, valamint szintaktikusan nem elemzett entitások behívása előfordulhat, továbbá a processzor egyes esetekben elvárt viselkedését. A bal szélső oszlop írja le a felismerési szövegösszefüggést:

Hivatkozás a tartalomban

egy elemben a kezdőcímke után és a zárócímke előtt bárhol megjeleníthető hivatkozásként megfelel a nem szövégi tartalom kritériumának.

Hivatkozás attribútum értékben

a kezdőcímkében egy attribútum értékként, vagy egy attribútum deklarációban alapértelmezett

értékként hivatkozásként; megfelel a nem szövégi AttValue paraméter kritériumának.

Előfordulás attribútum értékként

egy ENTITY típusúként deklarált attribútum értékeként, vagy ENTITIES típusúként deklarált attribútum értékében szóközökkel elválasztott vezérlőjelek egyikeként megjelenő érték, egy Név, nem hivatkozás.

Hivatkozás entitás értékben

egy paraméterben, vagy belső entitás literális entitás értékében előforduló hivatkozásként; megfelel a nem szövégi EntityValue paraméter kritériumának.

Hivatkozás a DTD-ben

a DTD belső vagy külső alkészleteinek hivatkozásaként, azonban az EntityValue vagy AttValue meghatározásán kívül esik.

	Entitástípus				Karakter
	Paraméter	Belső Általános	Külső Szintaktikusan elemzett Általános	Szintaktikusan nem elemzett	
Hivatkozás a tartalomban	Nem ismert	Igénybe vett	Igénybe vett ha érvényes	Tiltott	Igénybe vett
Hivatkozás attribútumértékben	Nem ismert	Igénybe vett	Tiltott	Tiltott	Igénybe vett
Előfordulás attribútumértékben	Nem ismert	Tiltott	Tiltott	Jelzés	Nem ismert
Hivatkozás entitás értékben	Igénybe vett	Kihagyva	Kihagyva	Tiltott	Igénybe vett
Hivatkozás DTD-ben	Paraméter-entitásként igénybe vett	Tiltott	Tiltott	Tiltott	Tiltott

4.4.1.Nem ismert

A DTD felületén kívül a % karakternek nincs különös jelentősége; ezért ami a DTD-n belül paraméter-entitás hivatkozás lehet, jelölő tartalomként nem ismert. Ugyanilyen módon a szintaktikusan nem elemzett entitások sem ismertek, csak ha egy megfelelően deklarált attribútumban jelennek meg.

4.4.2.Igénybe vett

Egy entitást a dokumentum akkor tartalmaz, ha a hivatkozás helyett a processzor az entitás helyettesítő szövegét keresi ki és dolgozza fel, mintha az a dokumentum része lenne azon a helyen, ahol a hivatkozást a processzor megtalálta. A helyettesítő szöveg tartalmazhat mind karakteralapú adatokat, mind pedig (paraméter-entitások kivételével) jelölő adatokat, amelyeket a processzor a szokásos módon ismer fel, azzal a kivétellel, hogy az entitások helyettesítő szövege használatos a jelölő adatot kiváltó határolójelként (az entitások: amp, lt, gt,apos, quot) mindig adatokként kezeltek. (Az "AT&T;" karakterlánc az "AT&T;" bővítése, a fennmaradó “és” karakter hivatkozás határolóelemként nem ismert.) A karakterhivatkozást a dokumentum tartalmazza olyankor, amikor a jelzett karakter feldolgozása magának a hivatkozásnak a helyén történik.

Tartalmazza ha érvényes

Amikor egy XML processzor felismer egy hivatkozást szintaktikusan elemzett entitásként, a processzornak a dokumentum érvényességellenőrzése céljából igénybe kell vennie kell az entitás helyettesítő szövegét. Ha az entitás külső és a processzor nem kísérli meg az XML dokumentum érvényességellenőrzését, a processzor igénybe veheti a helyettesítő szöveget, de ez nem szükségszerű.

Ez a szabály azon a felismerésen alapul, hogy az SGL és XML entitáskezelő mechanizmusok automatikus bevételi szolgáltatása elsősorban abból a célból készült, hogy támogassa az összeszerkesztő munkát, ami nem szükségszerűen megfelelő más alkalmazások céljaira, főleg dokumentumok átfésülésére. A böngészők például, amikor külső szintaktikusan elemzett entitás hivatkozást találnak, választhatják, hogy vizuálisan jelzik az entitás jelenlétét, de a képernyőre csak külön parancs kiadása után hívják be.

4.4.3.Tiltott

Az alábbiak tiltottak és végzetes hiba keletkezéséhez vezetnek:

- hivatkozás megjelenése szintaktikusan nem elemzett entitásban.
- bármely karakter vagy általános entitás hivatkozásának megjelenése a DTD-ben kivéve EntityValue vagy AttValue paraméterben.
- hivatkozás egy külső entitásra attribútumértékben.

4.4.4.Jelzés

Amikor egy szintaktikusan nem elemzett entitás vezérlőjelként jelenik meg egy ENTITY vagy ENTITIES típusúként deklarált entitásértékben, a processzornak jeleznie kell az alkalmazás felé a társított jelölő adat nevét, a jelölő adathoz társított rendszert és (ha vannak) a nyilvános azonosítókat.

4.4.5.Kihagyva

Amikor egy általános entitás hivatkozás megjelenik egy entitás deklaráció EntityValue paraméterében a processzor azt figyelmen kívül, változtatás nélkül hagyja.

4.4.6.Paraméter-entitásként igénybe vett

Ugyanúgy, mint a külső szintaktikusan elemzett entitásokat, a paraméter-entitásokat is csak érvényeségellenőrzés céljából kell igénybe venni. Amikor egy paraméter-entitás megjelenik ismertként és igénybe vettként a DTD-ben, annak helyettesítő szövege egy kezdő és egy záró szóköz (#x20)

karakterrel megnövekszik; a szándék arra irányul, hogy a paraméter-entitások helyettesítő szövegét rákényszerítsük, egész számú nyelvtani vezérlőjelet tartalmazzon a DTD-ben.

4.5. Belső entitás helyettesítő szövegének a felépítése

A belső entitások kezelésének megvitatásakor hasznos lehet az entitás érték két formájának a megkülönböztetése. A literális entitásérték az entitás deklarációs utasításában ténylegesen jelen lévő idézett karakterlánc, amely nem szóvégi EntityValue paraméternek felel meg. A helyettesítő szöveg az entitás tartalma a karakter hivatkozások és paraméter-entitás hivatkozások felváltása után.

A literális entitás érték belső entitás deklarációként van megadva (EntityValue), tartalmazhat karaktert, paraméter-entitást, és általános entitás hivatkozást. Az ilyen hivatkozásokat teljes mértékben a literális entitás értékben kell tárolni. A fentiekben hivatkozottak szerint igénybe vett tényleges helyettesítő szövegnek tartalmaznia kell minden hivatkozott paraméter-entitás helyettesítő szövegét, valamint tartalmaznia kell a literális entitás értékben a karakterhivatkozások bármelyikének a helyén található karaktert, ugyanakkor az általános entitás hivatkozásokat változtatás és méretnövelés nélkül figyelmen kívül kell hagyni. Például adottak az alábbi deklarációk:

```
<!ENTITY % pub    "&#xc9;ditions Gallimard" >
<!ENTITY   rights "All rights reserved" >
<!ENTITY   book   "La Peste: Albert Camus,
&#xA9; 1947 %pub;. &rights;" >
```

ebben az esetben a ” &book” entitás helyettesítő szövege:

```
La Peste: Albert Camus,
(c) 1947 Éditions Gallimard. &rights;
```

Az általános entitás "&rights;" hivatkozása meg fog nőni, ha a "&book;" hivatkozás megjelenik a dokumentum tartalmában, vagy egy attribútumértékben.

Ezekkel az egyszerű szabályokkal bonyolult beavatkozások hajthatók végre; egy nehéz feladatot jelentő példafeladat megvitatása az entitás hivatkozások megnövelése című függelékben található.

4.6. Előre meghatározott entitások

Mind az entitás, mind a karakter hivatkozások használhatók a balra mutató hegyes háromszög karakter, az “és” karakter, valamint egyéb határolóelemek kiváltásához. Erre a célra általános entitások (amp,lt, gt, apos, quot) egy készlete lett meghatározva. Numerikus karakterhivatkozások is használhatók; ezek megnövelése azonnal a felismerés pillanatában megtörténik és ugyanolyan módon kezelendők, mint a karakteralapú adatok, tehát a "<" és "&" numerikus karakterhivatkozások is használhatók a < és & karakterek kiváltásához olyankor, amikor azok megjelennek a karakteralapú adatok között.

Ezeket az entitásokat minden XML processzornak fel kell ismernie, függetlenül attól, hogy azok deklaráltak-e vagy sem. A közös működtethetőség fenntarthatóságának az érdekében az érvényes XML dokumentumoknak használatba vétel előtt ezeket az entitásokat, mint ahogy más entitásokat is, deklarálni kell. A kérdéses entitásokat belső entitásokként kell deklarálni, amelyeknek a helyezettesítő szövege a kiváltandó egyetlen karakter az alábbiakban bemutatottak szerint.

```
<!ENTITY lt     "&#38;#60;">
<!ENTITY gt     "&#62;">
<!ENTITY amp    "&#38;#38;">
<!ENTITY apos   "&#39;">
<!ENTITY quot   "&#34;">
```

Megjegyzendő, kétséges, hogy az "lt" és "amp" deklarációkban szereplő "<" és "&" karakterek kiválthatók annak alapján, hogy kielégítik-e a jól formázott entitás kicserélés kritériumait.

4.7. Jelölő deklarációs utasítások

A jelölők azonosítják név szerint a szintaktikusan elemzett entitások formátumát, illetve azt az alkalmazást, amelyhez a feldolgozási parancsok címződnek.

A jelölő deklarációs utasítások megadják egy jelölő nevét entitás és attribútumlista deklarációkban való használathoz, valamint egy külső azonosítót ahhoz a jelöléshez, amely engedélyezheti az XML processzor vagy annak ügyfél alkalmazása számára, hogy az adott jelölésben adatokat dolgozzon fel.

Jelölő deklarációs utasítások
[82] NotationDecl ::= '<!NOTATION' S Name S (ExternalID | PublicID) S? '>' [83] PublicID ::= 'PUBLIC' S PubidLiteral

Az XML processzoroknak az alkalmazások rendelkezésére kell bocsátaniuk minden deklarált, illetve attribútum meghatározásban vagy entitás deklarációban hivatkozott jelölő nevét és külső azonosítóját. Ezek továbbá a külső azonosítót átalakíthatják rendszerazonosítóvá, fájlnévvé, vagy bármiféle olyan információvá, amelyekkel az alkalmazás meghívhat egy processzort a jelölőben leírt adatok szolgáltatása céljából. (Ugyan ez nem hiba, de It is not an error, however, for XML dokumentums todeclare and refer to notations for which notation-specific applications arenot available on the system where the XML processor or application isrunning.)

4.8. Dokumentum-entitás

A dokumentum-entitás az entitásfa gyökerének a szerepét játssza és ez a kiindulópont az XML processzorok számára. A jelen specifikáció nem írja le, hogy a dokumentum-entitást a XML processzornak hol kell keresnie; más entitásokkal ellentétben a dokumentum-entitásnak nincs neve és könnyűszerrel megjelenik a processzor bemeneti adatfolaymában mindféle azonosítás nélkül is.

5. Megfelelőség

4.9. Érvényességellenőrző és nem érvényességellenőrző processzorok

A megfelelőnek minősített XML processzorok két osztályra osztottak: érvényességellenőrző és nem érvényességellenőrző processzorokra. Az érvényességellenőrző és nem érvényességellenőrző rendszerek egyaránt jelentik a jelen specifikációban leírt jól formázottsággal szemben támasztott elvárások megsértését.

Az érvényességellenőrző processzoroknak, DTD-ben a deklarációkban leírt a korlátozásokban bekövetkező szabálysértéseket jelenteniük kell. Jelenteniük kell a jelen specifikációben előírt érvényességi korlátozások teljesítésének az elmulasztását is.

11. Jelölések

Az XML formális nyelvtanát a jelen specifikációban az egyszerű Bővített Backus-Naur Formátumú (EBNF) jelöléssel adjuk meg. Ebben a formátumban minden egyes szabályt egy szimbólum határoz meg.

Szimbólum ::= jelentés

A szimbólumokat gyakran nagy kezdőbetűvel írjuk, ha azokat szabályszerű kifejezés határozza meg, illetve egyéb esetekben kis kezdőbetűvel. A literális karakterláncokat idézőjel jelöli.

A szabály jobb oldalán a kifejezésen belül az alábbi jelentések használatosak karakterláncoknak egy vagy több karakterrel történő egyeztetéséhez:

#xN

ahol N egy hexadecimális egész szám, a kifejezés a karakter az ISO/IEC 10646 szabvánnyal egyezteti, amelynek hitelesítő (UCS-4) előjel nélküli bináris számértéke értelmezéskor a jelzett értéket veszi fel. A #xN szimbólumot megelőző bevezető zéró karakterek nem számítanak; a zéró karaktereket a vonatkozó kódértékben a használatos karakterkódoló vezérli, az XLM-re nincsenek kihatással.
[a-zA-Z], [#xN-#xN]
a jelzett tartomány(ok)ban bármiféále karakterrel megegyező lehet (tartalmazott).
[^a-z], [^#xN-#xN]
a jelzett tartományon kívül bármiféále karakterrel megegyező.
[^abc], [^#xN#xN#xN]
a nem a megadott karakterek közé eső bármiféle karaktart jelenti.
"string"
a kettős idézőjelek közé zárt literális karakterlánccal megegyező jelentésű.
'string'
az egyszeres idézőjelek közé zárt literális karakterlánccal megegyező jelentésű.
Ezek a szimbólumok az alábbiak szerint kombinálhatók összetettebb minták egyeztetéséhez, ahol A és B egyszerű kifejezéseket képvisel:

(kifejezés)

a kifejezés egyetlen egységként kezelendő és a jelen listában leírtak szerint kombinálható.
A?
A értékével vagy semmivel nem egyezik; opcionális A.
A B
jelentése A után következő B.
A | B
jelentése A vagy B de mindkettő nem.
A - B
bármilyen karakterláncot jelent, amely megegyezik A értékével, de B értékével nem.
A+
A egy vagy több előfordulását jelenti.
A*
jelentése zéró vagy A több előfordulása.
Az eredményekben egyéb előforduló jelölések az alábbiak:
/* ... */
megjegyzés.
[wfc: ...]
jól formázottsági korlátozás; ezt azonosítja név szerint egy korlátozás egy eredményhez társított jól formázott dokumentumon.
[vc: ...]
érvényességi korlátozás; ezt azonosítja név szerint egy korlátozás egy eredményhez társított érvényes dokumentumon.

Függelékek

A. Hivatkozások

4.10. A.1 Irányadó hivatkozások

IETF RFC 1766
IETF (Internet Engineering Task Force). RFC 1766: Tags for the Identification of Languages, ed. H. Alvestrand. 1995.
ISO 639
(International Standardization Organization). ISO 8879:1988 (E). Code for the representation of names of languages. [Geneva]: International Standardization Organization, 1988.
ISO 3166
(International Standardization Organization). ISO 3166-1:1997 (E). Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes [Geneva]: International Standardization Organization, 1997.

ISO/IEC 10646
ISO (International Standardization Organization). ISO/IEC 10646-1993 (E). Information technology -- Universal Multiple-Octet Coded Character Set (UCS) -- Part 1: Architecture and Basic Multilingual Plane. [Geneva]: International Standardization Organization, 1993 (plus amendments AM 1 through AM 7).
Unicode
The Unicode Consortium. The Unicode Standard, Version 2.0. Reading, Mass.: Addison-Wesley Developers Press, 1996.

4.11. A.2 Egyéb hivatkozások

Aho/Ullman
Aho, Alfred V., Ravi Sethi, and Jeffrey D. Ullman. Compilers: Principles, Techniques, and Tools. Reading: Addison-Wesley, 1986, rpt. corr. 1988.
Berners-Lee et al.
Berners-Lee, T., R. Fielding, and L. Masinter. Uniform Resource Identifiers (URI): Generic Syntax and Semantics. 1997. (Work in progress; see updates to RFC1738.)
Brüggemann-Klein
Brüggemann-Klein, Anne. Regular Expressions into Finite Automata. Extended abstract in I. Simon, Hrsg., LATIN 1992, S. 97-98. Springer-Verlag, Berlin 1992. Full Version in Theoretical Computer Science 120: 197-213, 1993.
Brüggemann-Klein and Wood
Brüggemann-Klein, Anne, and Derick Wood. Deterministic Regular Languages. Universität Freiburg, Institut für Informatik, Bericht 38, Oktober 1991.
IETF RFC1738
IETF (Internet Engineering Task Force). RFC 1738: Uniform Resource Locators (URL), ed. T. Berners-Lee, L. Masinter, M. McCahill. 1994.
IETF RFC1808
IETF (Internet Engineering Task Force). RFC 1808: Relative Uniform Resource Locators, ed. R. Fielding. 1995.
IETF RFC2141
IETF (Internet Engineering Task Force). RFC 2141: URN Syntax, ed. R. Moats. 1997.
ISO/IEC 8879
ISO (International Standardization Organization). ISO/IEC 8879-1986 (E). Information processing -- Text and Office Systems -- Standard Generalized Markup Language (SGML). First edition -- 1986-10-15. [Geneva]: International Standardization Organization, 1986.
ISO/IEC 10744
ISO (International Standardization Organization). ISO/IEC 10744-1992 (E). Information technology -- Hypermedia/Time-based Structuring Language (HyTime). [Geneva]: International Organization for Standardization, 1992. Extended Facilities Annexe. [Geneva]: International Standardization Organization, 1996.

B. Karakterosztályok

A Unicode szabványban meghatározott jellemzőket követve a karaktereket a következők szerint osztályozták: alapkarakterek (többek között ezek tartalmazzák a latin ABC karaktereit is ékezetek nélkül), idiografikus karakterekre és kombinált karakterekre (többek között ez az osztály tartalmazza a legtöbb ékezetet). Ezeknek az osztályoknak a kombinációja alkotja a betűket. A számjegyeket és kitöltő karaktereket szintén megkülönböztetik.

Karakterek

[84] Betűk ::= BaseChar | Ideographic

[85] Alapkarakterek ::= [#x0041-#x005A] | [#x0061-#x007A]
| [#x00C0-#x00D6] | [#x00D8-#x00F6]
| [#x00F8-#x00FF] | [#x0100-#x0131]
| [#x0134-#x013E] | [#x0141-#x0148]
| [#x014A-#x017E] | [#x0180-#x01C3]
| [#x01CD-#x01F0] | [#x01F4-#x01F5]
| [#x01FA-#x0217] | [#x0250-#x02A8]
| [#x02BB-#x02C1] | #x0386 | [#x0388-#x038A]
| #x038C | [#x038E-#x03A1] | [#x03A3-#x03CE]
| [#x03D0-#x03D6] | #x03DA | #x03DC | #x03DE
| #x03E0 | [#x03E2-#x03F3] | [#x0401-#x040C]
| [#x040E-#x044F] | [#x0451-#x045C]
| [#x045E-#x0481] | [#x0490-#x04C4]
| [#x04C7-#x04C8] | [#x04CB-#x04CC]
| [#x04D0-#x04EB] | [#x04EE-#x04F5]
| [#x04F8-#x04F9] | [#x0531-#x0556] | #x0559
| [#x0561-#x0586] | [#x05D0-#x05EA]
| [#x05F0-#x05F2] | [#x0621-#x063A]
| [#x0641-#x064A] | [#x0671-#x06B7]
| [#x06BA-#x06BE] | [#x06C0-#x06CE]
| [#x06D0-#x06D3] | #x06D5 | [#x06E5-#x06E6]
| [#x0905-#x0939] | #x093D | [#x0958-#x0961]
| [#x0985-#x098C] | [#x098F-#x0990]
| [#x0993-#x09A8] | [#x09AA-#x09B0] | #x09B2
| [#x09B6-#x09B9] | [#x09DC-#x09DD]
| [#x09DF-#x09E1] | [#x09F0-#x09F1]
| [#x0A05-#x0A0A] | [#x0A0F-#x0A10]
| [#x0A13-#x0A28] | [#x0A2A-#x0A30]
| [#x0A32-#x0A33] | [#x0A35-#x0A36]
| [#x0A38-#x0A39] | [#x0A59-#x0A5C] | #x0A5E
| [#x0A72-#x0A74] | [#x0A85-#x0A8B] | #x0A8D
| [#x0A8F-#x0A91] | [#x0A93-#x0AA8]
| [#x0AAA-#x0AB0] | [#x0AB2-#x0AB3]
| [#x0AB5-#x0AB9] | #x0ABD | #x0AE0
| [#x0B05-#x0B0C] | [#x0B0F-#x0B10]
| [#x0B13-#x0B28] | [#x0B2A-#x0B30]
| [#x0B32-#x0B33] | [#x0B36-#x0B39] | #x0B3D
| [#x0B5C-#x0B5D] | [#x0B5F-#x0B61]
| [#x0B85-#x0B8A] | [#x0B8E-#x0B90]
| [#x0B92-#x0B95] | [#x0B99-#x0B9A] | #x0B9C
| [#x0B9E-#x0B9F] | [#x0BA3-#x0BA4]
| [#x0BA8-#x0BAA] | [#x0BAE-#x0BB5]
| [#x0BB7-#x0BB9] | [#x0C05-#x0C0C]
| [#x0C0E-#x0C10] | [#x0C12-#x0C28]
| [#x0C2A-#x0C33] | [#x0C35-#x0C39]
| [#x0C60-#x0C61] | [#x0C85-#x0C8C]
| [#x0C8E-#x0C90] | [#x0C92-#x0CA8]
| [#x0CAA-#x0CB3] | [#x0CB5-#x0CB9] | #x0CDE
| [#x0CE0-#x0CE1] | [#x0D05-#x0D0C]
| [#x0D0E-#x0D10] | [#x0D12-#x0D28]
| [#x0D2A-#x0D39] | [#x0D60-#x0D61]
| [#x0E01-#x0E2E] | #x0E30 | [#x0E32-#x0E33]
| [#x0E40-#x0E45] | [#x0E81-#x0E82] | #x0E84
| [#x0E87-#x0E88] | #x0E8A | #x0E8D
| [#x0E94-#x0E97] | [#x0E99-#x0E9F]
| [#x0EA1-#x0EA3] | #x0EA5 | #x0EA7
| [#x0EAA-#x0EAB] | [#x0EAD-#x0EAE] | #x0EB0
| [#x0EB2-#x0EB3] | #x0EBD | [#x0EC0-#x0EC4]
| [#x0F40-#x0F47] | [#x0F49-#x0F69]
| [#x10A0-#x10C5] | [#x10D0-#x10F6] | #x1100
| [#x1102-#x1103] | [#x1105-#x1107] | #x1109
| [#x110B-#x110C] | [#x110E-#x1112] | #x113C
| #x113E | #x1140 | #x114C | #x114E | #x1150
| [#x1154-#x1155] | #x1159 | [#x115F-#x1161]
| #x1163 | #x1165 | #x1167 | #x1169

Az itt meghatározott karakterosztályok a Unicode karakter adatbázisból származtathatók az alábbiak szerint:

- A név kezdőkaraktereknek a Ll, Lu, Lo, Lt, Nl kategóriák valamelyikéhez kell tartozniuk.
- A név karaktereknek a név kezdőkarakterek kivételével az Mc, Me, Mn, Lm, vagy Nd kategóriák valamelyikéhez kell tartozniuk.
- A kompatibilitási területre eső karakterek (tehát, azok, amelyek nagyobbak, mint #xF900 és kisebbek, mint #xFFFE) használata nem engedélyezett az XML nevekben.
- Azok a karakterek, amelyek betűtípus vagy kompatibilitási szempontból felbomlottnak tekintendők (vagyis az adatbázis 5. mezejében "kompatibiltási formattálási címkével ellátottak" – jelölésük az 5. mezőre utaló szám, előtte "<" karakter) nem engedélyezettek.
- Az alábbi karakterek névkezdő karakterekként használatosak, nem pedig vénkarakterekként, mert a tulajdonság fájl ezeket az ABC betűiként osztályozza őket: [#x02BB-#x02C1], #x0559, #x06E5, #x06E6.
- A #x20DD-#x20E0 közötti karakterek a kizártak (a Unicode 5.14 bekezdésének a rendelkezéseivel összhangban).
- A #x00B7 karakter helykitöltő karakterként osztályozott, mert a tulajdonság lista annak minősítette.
- A #x0387 karakter névkarakterként hozzáadott, mert a #x00B7 annak hitelesítő egyenértéke.
- A ':' és '_' karakterek használata névkezdő karakterekként nem engedélyezett.
- A '-' és '.' karakterek névkarakterekként történő használata engedélyezett.

C. XML és SGML (Nem-irányadó)

Az XML az SGML alkészleteként került kifejlesztésre, amennyiben minden érvényes XML dokumentum megfelel az SGML dokumentumoknak. Az SGML dokumentumokon kívüli dokumentumokra az XML által gyakorolt összehasonlító többlet megszorítások a mellékelt figyelmeztetésben olvashatók, amely tartalmazza az XML dokumentumok használata által az SGML szintaktikus elemzőjének (parser) működésére kifejtett hatásra vonatkozó SGML nyilatkozatot is.

D. Entitások és karakter hivatkozások megnövelése (Nem-irányadó)

A jelen függelék néhány példán keresztül szemlélteti az entitás- és karakterhivatkozások felismerésére és megnövelésére vonatkozó szempontokat.

Ha a DTD tartalmazza a deklarációs utasítást

```
<!ENTITY example "<p>An ampersand (&#38;#38;) may be escaped numerically (&#38;#38;#38;) or with a general entity
(&amp;amp;).</p>" >
```

az XML processzor akkor fogja felismerni a karakterhivatkozásokat, amikor az entitás deklarációjának szintaktikus elemzését végzi, és mielőtt az “example” elnevezésű entitás elemzésének eredményéül a karakterláncot menti:

```
 <p>An ampersand (&#38;) may be escaped
numerically (&#38;#38;) or with a general entity
```

```
(&amp;amp;).</p>
```

A dokumentumban az "&example;" entitásra utaló hivatkozás hatására bekövetkezik a szöveg újabb szintaktikus elemzése, amikor a processzor felismeri a “p” elem kezdő- és zárócímkéjét és a három hivatkozást felismeri és bővíti, ennek eredménye a "p" elem létrehozása a következő tartalommal (összes adat, nincs határolóelem, sem jelölő adat):

```
Egy (&) numerikusan a (&#38;) karakterrel, vagy az általános (&amp;) entitással kiváltható.
```

Egy összetettebb példán bemutatjuk a szabályokat és azok hatásait.

A következő példában a sorok számozása csak a könnyebb eligazodást segíti.

```
1 <?xml version='1.0'?>
2 <!DOCTYPE test [
3 <!ELEMENT test (#PCDATA) >
4 <!ENTITY % xx '&#37;zz;'>
5 <!ENTITY % zz '&#60;!ENTITY tricky "error-prone" >' >
6 %xx;
7 ]>
8 <test>This sample shows a &tricky; method.</test>
```

Ez az eljárás az alábbi eredményt hozza:

- a 4. sorban a 37 karakterre utaló hivatkozás azonnal bővül, és a szimbólum táblán mentődik az "xx" paraméter-entitás "%zz;" értékkel. Mivel a helyettesítő szöveg újból nem lett letapogatva, a "%zz;" paraméter-entitás ismeretlen. (Ha ismert lenne, hibaállapot keletkezne, mert még nincs deklarálva.)
- az 5. sorban a "<" karakterhivatkozás azonnal megnövekedett és a "zz" paraméter-entitás az "<!ENTITY tricky "error-prone" >" helyettesítő szöveggel mentődött, ami egy jól formázott entitás deklaráció.
- a 6. sorban a processzor felismeri az "xx" paraméter-entitást és az "xx" helyettesítő szövegén (nevezetesen a "%zz;" értéken) szintaktikus elemzést végez el. Felismeri a "zz" értékre utaló hivatkozást és ennek hatására annak helyettesítő szövegén ("<!ENTITY tricky "error-prone" >") szintaktikus elemzést hajt végre. A "tricky" általános entitás most már deklarált az "error-prone" helyettesítő szöveggel.
- a 8. sorban a processzor felismeri a "tricky" általános entitásra utaló hivatkozást és kibővíti olyan módon, hogy a "test" elem önmeghatározó és (nyelvtan nélküli) karakterlánccá válik. Ez a példa egy hibalehetőséget hordozó módszert mutat be.

E. Determináns tartalomtípusok (Nem-irányadó)

A kompatibilitás biztosítása érdekében szükséges, hogy az elemtípusban a deklarációk determináns jellegűek legyenek.

Az SGML megkívánja a tartalomtípusok determináns jellegét (az SGML-ben ezt "egyértelműségnek");

Az SGML rendszerek használatára épült XML processzorok a nem determináns tartalomtípusokat hibajelzővel láthatják el.

Például a ((b, c) | (b, d)) tartalomtípus nem determináns, mivel a modellben szereplő kezdő b miatt a szintaktikus elemző, ha nem tekint előre, hogy a b után milyen elem következik, nem tudja eldönteni, hogy melyik b kerül egyeztetésre. Ebben az esetben a két hivatkozás egyesíthető egyetlen hivatkozássá, és ennek köszönhetően a modell (b, (c | d)) alakúra változik. Egy kezdő b most már egyértelműen egyeztethető egyetlen névvel a tartalomtípusban. Ilyenkor a szintaktikus elemzőnek nem kell előre tekintenie, hogy megnézze, milyen elem következik; mind a c, mind a d érték elfogadható.

Még formaibb módon a tartalomtípusból véges állapotú automatizmus építhető ki szabványos algoritmusok használatával, például az Aho, Sethi, and Ullman [Aho] 3.9 szakaszában található 3.5 algoritmussal. Sok ilyen algoritmusban egy követőkészletet építenek be a szabványos kifejezés minden

egyes pozíciójába (tehát a szabványos kifejezés szintaxisfájának minden levél leágazásába); ha bármelyik pozíció rendelkezik olyan követőkészlettel, amelyben egynél több követési pozíció ugyanolyan elemtípus nevével címkézett, a tartalomtípus hibaállapotba kerül és erről hibaüzenetet küldhet.

Vannak olyan algoritmusok, amelyek sokféle, de nem az összes determináns tartalomtípus automatikus csökkentését engedélyezik egyenértékű determináns modellekbe; lásd Brüggemann-Klein 1991 [ABK].

F. Karakterkódolások automatikus detektálása (Nem irányadó)

Az XML kódoló deklarációs utasítás minden egyes entitáson belső címkeként működik, jelzi, hogy melyik karakterkódoló van használatban. Mielőtt egy XML processzor a belső címkét olvashatná azonban, szemmel láthatóan tudnia kellene, hogy melyik kódoló van használatban—és éppen ez az, amit a címkének jeleznie kellene. Általános esetben ez reménytelen helyzet az XML-ben. Ugyanakkor mégsem teljesen reménytelen a helyzet, mert az XML kétféle módon korlátozza az általános eset bekövetkezésének lehetőségét. Minden egyes rendszermegvalósításnál feltételezett, hogy a rendszer csak egy teljes karakterkódolási módszert támogat, és az XML kódolási deklarációja mind pozícióját, mind tartalmát tekintve korlátozott ahhoz, normális esetekben megvalósíthatóvá tegye az automatikus karakterkódolás felderítést minden egyes entitásban. Sok esetben az XML saját adatáramlásán kívül más információforrások is rendelkezésre állnak. Két eset különböztethető meg: attól függően, hogy az XML entitás társuló (külső) információkkal együtt, vagy azok nélkül kerül-e a processzor elé. Először nézzük az első esetet.

Mivel nem minden XML entitás kódolt UTF-8 vagy UTF-16 formátumban, ezért egy XML kódolási deklarációval kell kezdeni, amelyben az első karakterekként '<?xml' értéket kell meghatározni, ezzel bármely megfelelően alkalmazkodó processzor képes felderíteni 2-4 oktett adatbevitelt követően, hogy a következő esetek közül melyik az érvényes. Ennek a listának az elolvasásakor segíthet, ha tudjuk, hogy UCS-4 kódolásban a '<' karakter "#x0000003C" és a '?' karakter "#x0000003F", az UTF-16 kódolású adatfolyamoknál a Bájt-sorrend jel "#xFEFF".

- 00 00 00 3C: UCS-4, nagy-endian gép (1234 sorrend)
- 3C 00 00 00: UCS-4, kis-endian gép (4321 sorrend)
- 00 00 3C 00: UCS-4, szokatlan oktett sorrend (2143)
- 00 3C 00 00: UCS- szokatlan oktett sorrend (3412)
- FE FF: UTF-16, nagy-endian
- FF FE: UTF-16, kis-endian
- 00 3C 00 3F: UTF-16, nagy-endian, nincs Bájt-sorrend jel (és ezért az igazat megvallva hibás)
- 3C 00 3F 00: UTF-16, kis-endian, nincs Bájt-sorrend jel (és ezért az igazat megvallva hibás)
- 3C 3F 78 6D: UTF-8, ISO 646, ASCII, az ISO 8859 bizonyos részei, a Shift-JIS, az EUC, illetve bármiféle 7-bit vagy 8-bit, vagy kevert sávszélességű kódolás használható, ami biztosítja, hogy az ASCII karakterek normál pozíciójúak, szélességűek és értékűek, a tényleges kódolási deklarációt kell olvasni annak kiderítéséhez, hogy melyik a használt típus, de miután ezen kódolási rendszerek mindegyike ugyanazon bitet, az ASCII karakterek bitmintáját használja, maga a kódolási deklaráció megbízhatóan leolvasható
- 4C 6F A7 94: EBCDIC (bizonyos összeállításoknál a teljes kódolási deklarációt be kell olvasni a használt kódlap meghatározásához)
- más: UTF-8 kódolási deklaráció nélkül, vagy az adatáramlás sérült, töredékes vagy valamilyen típusú csomagba beszorult

Az ilyen szintű automatikus felderítés elegendő az XML kódolási deklaráció olvasásához és a karakterkódoló azonosító szintaktikus elemzéséhez, ami még szükséges az egyes kódoló családok megkülönböztetéséhez (például az UTF-8 megkülönböztetéséhez a 8859-től, vagy a 8859 részeinek egymástól való megkülönböztetéséhez, vagy egyedi használatban lévő EBCDIC kódlapok megállapításához, stb.).

Miután a kódolási deklaráció tartalma az ASCII karakterek használatára korlátozott, egy processzor megbízhatóan végig tudja olvasni az egész kódolási deklarációt, mihelyt felderítette, hogy melyik kódolás van használatban. Mivel a gyakorlatban széles körben alkalmazzák a fenti kategóriák valamelyikéhez tartozó karakterkódolásokat, az XML kódolási deklaráció lehetővé teszi a karakterkódolások célszerűen megbízható címkézését még olyan esetekben is, amikor a külső információk az operációs rendszer vagy a szállító protokoll szintjén megbízhatatlanok.

Miután a processzor felderítette a használt karakterkódolási módot, megfelelőképpen intézkedhet, akár külön adatbeviteli rutin használatával minden egyes esetben, akár minden egyes bevitt karakter megfelelő konverziós műveletének a meghívásával. Az egyéb öncímkéző rendszerekhez hasonlóan az XML kódolási deklaráció működésképtelenné válik, ha bármiféle szoftver megváltoztatja az entitás karakterkészletét vagy kódolását anélkül, frissítené a kódolási deklarációt. A karakterkódoló rutinok fejlesztőinek biztosítaniuk kell az entitás felcímkézéséhez használt külső és belső információk pontosságát.

A második lehetséges eset akkor következik be, amikor az XML entitás kódoló információval társított, például bizonyos fájlrendszerekben és hálózati protokollokban. Amikor sok információforrás áll rendelkezésre, azok relatív prioritása és a konfliktuskezeléshez kiválasztandó kedvelt módszert meg kell határozni az XML szállításához használt magasabb szintű protokoll részeként. Például a belső címke és egy külső fejléc MIME kódolású címkéje esetében a relatív prioritás szabályainak a szöveges/xml és alkalmazás/xml MIME típusait meghatározó RFC dokumentum részét kell képezniük. Az együttműködési képesség fenntarthatósága érdekében ugyanakkor az alábbi szabályok betartása ajánlatos.

- Ha egy XML entitás egy fájlban helyezkedik el, a Bájt-sorrend jel és a kódolásdeklaráló PI használatos (ha jelen van) a karakterkódolás meghatározásához. Minden egyéb heurisztika és információforrás csak hibaállapotok helyreállítását szolgálja.

- Ha egy XML entitás MIME típusú szöveges/xml-hez társítva kerül szállításra, a MIME típuson elhelyezkedő charset paraméter határozza meg a karakterkódolási módszert; minden egyéb heurisztika és információforrás csak hibaállapotok helyreállítását szolgálja.

- Ha egy XML entitás MIME típusú alkalmazás/xml-hez társítva kerül szállításra, a Bájt-sorrend jel és a kódolásdeklaráló PI használatos (ha jelen van) a karakterkódolás meghatározásához. Minden egyéb heurisztika és információforrás csak hibaállapotok helyreállítását szolgálja.

Ezek a szabályok csak a protokollszintű dokumentáció hiányában alkalmazandók; elsősorban akkor, amikor a MIME típusú szöveges/xml és alkalmazás/xml meghatározása történik, a vonatkozó RFC ajánlásai ezeket a szabályokat hatálytalanítani fogják.

G. A W3C XML munkacsoport (Nem-irányadó)

A jelen specifikációt a W3C XML munkacsoport (WG) készítette el és hagyta jóvá nyilvánosságra hozatal céljából. Az a tény, hogy a WG a jelen specifikációt jóváhagyta, nem szükségszerűen jelenti azt, hogy a jóváhagyást a WG összes tagja megszavazta. Az XML WG jelenlegi és korábbi tagjai az alábbiak:
Jon Bosak, Sun (Chair); James Clark (Technical Lead); Tim Bray, Textuality and Netscape (XML társszerkesztő); Jean Paoli, Microsoft (XML társszerkesztő); C. M. Sperberg-McQueen, U. of Ill. (XML társszerkesztő); Dan Connolly, W3C; Steve DeRose, INSO; Dave Hollander, HP; Eliot Kimber, Highland; Eve Maler, ArborText; Tom Magliery, NCSA; Murray Maloney, Muzmo and Grif; Makoto Murata, Fuji Xerox Information Systems; Joel Nava, Adobe; Peter Sharpe, SoftQuad; John Tigue, DataChannel