

Az XML 1.0 ajánlás

Jeszenszky Péter

Debreceni Egyetem, Informatikai Kar

jeszenszky.peter@inf.unideb.hu

Utolsó módosítás: 2022. szeptember 30.

Szabvány

- Az aktuális szabvány:
 - *Extensible Markup Language (XML) 1.0 (Fifth Edition)* (W3C ajánlás, 2008. november 26.)
<https://www.w3.org/TR/xml/>
- Leírja az XML dokumentumoknak nevezett objektumokat és részben előírja az ezeket feldolgozó programok viselkedését.

XML dokumentumok

- Olyan szöveges objektumok, melyek a szabvány előírásai szerint jólformáltak.
- Fizikai és logikai szerkezetük van.
 - Fizikailag egyedeknek nevezett tárolási egységekből állnak.
 - Logikailag deklarációkból, elemekből, megjegyzésekből, feldolgozási utasításokból és további szerkezeti alkotóelemekből állnak.

Elemek (1)

- Minden elemet nyitó és záró címke határol vagy egyetlen üres elem címke alkot.
 - Példák:

```
<author>Sir Arthur Conan Doyle</author>
```

```
<message xml:lang="en">Hello, World!</message>
```

```

```

Elemek (2)

- A nyitó, záró és üres elem címkében adott nevet elemtípusnak nevezzük.
 - **Jólformáltsági megszorítás:** a nyitó és záró címkében adott nevek meg kell, hogy egyezzenek.
- A nyitó és a záró címke fogja közre az elem tartalmát.
- Az elemekhez meg lehet adni attribútum-specifikációknak nevezett név-érték párokat.

Elemek (3)

- Az üres elem tartalom nélküli elem.
 - Megadható közvetlenül egymást követő nyitó és záró címkével vagy üres elem címkével.
 - Példa: `<elem></elem>`, `<elem/>`

Jólformáltság

- Nagyjából az alábbiakat jelenti:
 - Egyetlen felső szintű elem, az úgynevezett **gyökérelem** tartalmazza az összes többi elemet.
 - Minden nyitó címkéhez tartozik egy megfelelő záró címke.
 - Az elemek megfelelő módon egymásba ágyazottak, nem fedhetik át egymást.
 - Minden a dokumentumban hivatkozott elemzett egyed jólformált.
- Sok további követelménynek – úgynevezett **jólformáltsági megszorításnak** – kell teljesülnie.

Karakterek

- Az XML dokumentumok Unicode karakterekből állnak.
- Minden XML feldolgozó kötelezően kell, hogy támogassa az UTF-8 és UTF-16 kódolásokat.

Whitespace karakterek (1)

- *Whitespace* karakterek az alábbiak:
 - U+0009 (vízszintes tabulátor, ' \t ')
 - U+000A (új sor, LF, ' \n ')
 - U+000D („kocsi vissza”, CR, ' \r ')
 - U+0020 (szóköz)

Whitespace karakterek (2)

- XML dokumentumok szerkesztésekor szokás az elemek egymásba ágyazottságát az emberi szem számára jól átláthatóan mutató *whitespace* karaktereket használni.
 - A sorok elején megfelelő számú szóköz vagy vízszintes tabulátor karakterek elhelyezése.
- Ezek a *whitespace* karakterek az alkalmazások számára általában nem lényegesek.

Whitespace karakterek (3)

- Példa:

```
<?xml version="1.0" encoding="UTF-8"?>
<movie id="0078748"><title xml:lang="en">Alien</title>
<year>1979</year><genres><genre>horror</genre>
<genre>sci-fi</genre></genres></movie>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<movie id="0078748">
  <title xml:lang="en">Alien</title>
  <year>1979</year>
  <genres>
    <genre>horror</genre>
    <genre>sci-fi</genre>
  </genres>
</movie>
```

Névkarakterek

- A nevekben megengedett karakterek pontos felsorolását lásd a szabványban.
 - Megengedettek a betű és a decimális számjegy karakterek.
 - A programozási nyelvekben megszokott '_' karakter mellett a '.', '-', ':' és '·' (U+00B7) karakterek is rendelkezésre állnak.
- **Névtoken:** egy vagy több névkarakterből álló karaktersorozat.

Nevek (1)

- Olyan névtokenek, melyek első karakterei csak a szabványban megadott karakterek lehetnek.
 - Nem kezdődhetnek például számjegy, ' - ', ' . ' és ' . ' (U+00B7) karakterrel sem.
- Névben szereplő ' : ' karakter speciális jelentést nyer az XML névterek specifikáció révén.
 - Kizárólag ennek megfelelően használjuk ezt a karaktert!

Nevek (2)

- Nevek például az alábbi karakterláncok:
 - `születési.dátum`
 - `születési-dátum`
 - `jaxb:version`
 - `elem1, elem2, elem3, ...`
 - `β0`
 - `_` (igen, akár egyetlen aláhúzójel is!)
 - `_ - _ - _` (igen, akár ez is!)
- Nem nevek például az alábbi karakterláncok:
 - `1st, 2nd, 3rd, ...` (mert név nem kezdődhet számjegy karakterrel)
 - `on/off` (mert név nem tartalmazhat `' / '` karaktert)

Nevek (3)

- A nevek (például az elem és attribútumnevek) kisbetű-nagybetű érzékenyek az XML-ben.
 - Például a `title`, `Title` és `TITLE` különböző nevek.

Literálok

- ' " ' vagy ' ' ' karakterek által határolt karaktersorozatok, melyek nem tartalmazzák magát a határoló karaktert.
 - Példa: `"/style.css"`, `'/style.css'`
- Használatuk például az alábbiak megadásánál:
 - Attribútumértékek
 - Belső egyedek tartalma

Jelölők

- Nyitó, záró és üres elem címke
- Karakterhivatkozás
- Egyedhivatkozás
- Megjegyzés
- Feldolgozási utasítás
- CDATA-szakasz határolók
- XML deklaráció
- Szövegdeklaráció
- Dokumentumtípus- deklaráció

Karakteres adat

- A dokumentum szövegének az a része, mely nem jelölő.

Szöveg nyelvének jelzése (1)

- Az `xml:lang` attribútummal jelezhető az elemek tartalmának és attribútumértékeink nyelve.
 - Érvényes dokumentumokban az attribútumot a DTD-ben kell deklarálni használat esetén.
 - Az attribútum az elemre és gyermekeire is vonatkozik (beleértve az elem attribútumainak értékeit), azonban a gyermekeknél is megadható más értékkel (lyuk a hatáskörben).

Szöveg nyelvének jelzése (2)

- Az attribútum értéke egy természetes nyelvet azonosító címke kell, hogy legyen, a címkéket az alábbi szabvány definiálja:
 - Addison Phillips (ed.), Mark Davis (ed.). *RFC 5646: Tags for Identifying Languages*. September 2009.
<https://www.rfc-editor.org/rfc/rfc5646>
- Példák az attribútum értékére: de-AT, en-US, hu, az-Arab, az-Cyrl, az-Latn, ...
- Speciálisan megadható az üres karakterlánc annak jelzésére, hogy nem áll rendelkezésre információ a nyelvről.
 - Mintha az `xml:lang` attribútum egyáltalán nem került volna megadásra az elemen vagy valamely felmenőjén.

Szöveg nyelvének jelzése (3)

- Példa:
 - `<para xml:lang="en">The more <phrase xml:lang="fr">outré</phrase> and grotesque an incident is the more carefully it deserves to be examined, and the very point which appears to complicate a case is, when duly considered and scientifically handled, the one which is most likely to elucidate it.</para>`
 - Arthur Conan Doyle, *The Hound of the Baskervilles*.

Speciális karakterek

- Az '&' és '<' karakterek ebben a formában kizárólag jelölő-határolóként, megjegyzésekben, feldolgozási utasításokban és CDATA-szakaszokban fordulhatnak elő.
 - Helyettünk minden egyéb helyen karakterhivatkozásokat, illetve az & és < egyedhivatkozásokat kell használni!
 - A '>' karakter megadható az > egyedhivatkozással.
- Hogy attribútumértékek is tartalmazhassanak aposztrófokat és idézőjeleket, az ''' és '"' karakterek megadhatók az ' és " egyedhivatkozásokkal.

Nyitó, záró és üres elem címke (1)

- **Nyitó címke:**
 - Példa: `<title>`, `<title xml:lang="en">`
- **Záró címke:**
 - Példa: `</title>`
- **Üres elem címke:**
 - Példa: `
`, `<hr />`, ``

Nyitó, záró és üres elem címke (2)

- **Jólformáltsági megszorítás:** nyitó és üres elem címkében minden attribútumnév legfeljebb egyszer szerepelhet.
- Lényegtelen az attribútumok megadásának sorrendje.

Karakterhivatkozások

- Szövegben, attribútum értékekben és literális egyed értékekben Unicode karakterek kifejezhetők az alábbi formájú karakterhivatkozásokkal:
 - `&#nnnn;`, ahol *nnnn* a kódpontot ábrázoló decimális számjegysorozat.
 - Példa: `©` (©), `☯` (☯), `😺`
 - `&#xhhhh;`, ahol *hhhh* a kódpontot ábrázoló hexadecimális számjegysorozat.
 - Példa: `©` (©), `☯` (☯), `😺`

Egyedhivatkozások

- Hivatkozás egy névvel azonosított egyed tartalmára.
 - Hivatkozás elemzett általános egyedre: `&név ;`
 - Példa: `&`, `Á`, `©right;`
 - Paraméteregyed-hivatkozás: `%név;`
 - Példa: `%inline;`, `%ImgAlign;`
- Az egyedhivatkozásokról részletesen később, a dokumentum fizikai szerkezete kapcsán lesz szó.

Megjegyzések

- A dokumentumban bárhol szerepelhetnek más jelölőkön kívül.
 - Az egyetlen kivétel a dokumentumtípus-deklaráció, melyben bizonyos helyeken előfordulhatnak.
- Példa:
 - `<!-- Ez egy megjegyzés -->`

Feldolgozási utasítások

- Az alkalmazások számára tartalmaznak utasításokat.
- Példa:
 - `<?xml-stylesheet type="text/css" href="style.css"?>`

CDATA-szakaszok

- Bárhol előfordulhatnak a dokumentumban, ahol előfordulhat karakteres adat.
 - Olyan karaktereket tartalmazó szövegrészek levédésére szolgálnak, melyek egyébként jelölőként lennének tekintve.
 - A CDATA-részben csak a ']] > ' karakterlánc tekintett jelölőnek.
- Példa:
 - `<![CDATA[if (0 < n && n <= 10)]]>`

XML deklaráció

- Az XML dokumentumoknak egy XML deklarációval ajánlott kezdődniük, mely meghatározza az XML használt verziószámát.
 - A karakterkódolást meg kell adni, ha a használt kódolás nem az UTF-8 vagy az UTF-16, kivéve ha a kódolást egy magasabb szintű protokoll (például a HTTP) határozza meg.
- Példa:
 - `<?xml version="1.0"?>`
 - `<?xml version='1.0' encoding='UTF-8'?>`

Dokumentumtípus-deklaráció (1)

- Dokumentumok egy osztályához egy nyelvtant meghatározó jelölő deklarációkat tartalmaz és/vagy ilyen deklarációkra mutat.
 - Ezt a nyelvtant **dokumentumtípus-definíció**nak (vagy röviden DTD-nek) nevezik.
- A név a dokumentumtípus-deklarációban a gyökérelem elemtípusát írja elő.

Dokumentumtípus-deklaráció (2)

- Egy **külső DTD alkészlet**nek nevezett, jelölő deklarációkat tartalmazó külső egyedre mutató dokumentumtípus-deklaráció:
 - `<!DOCTYPE score-partwise SYSTEM "http://www.musicxml.org/dtds/partwise.dtd">`
 - `<!DOCTYPE score-partwise SYSTEM "partwise.dtd">`
 - `<!DOCTYPE score-partwise PUBLIC "-//Recordare//DTD MusicXML 3.1 Partwise//EN" "http://www.musicxml.org/dtds/partwise.dtd">`
- A fenti példák a MusicXML DTD-t használják.
 - Lásd: <https://www.musicxml.com/>

Dokumentumtípus-deklaráció (3)

- Jelölő deklarációkat – az úgynevezett **belső DTD alkészletet** – tartalmazó dokumentumtípus-deklaráció:

```
– <!DOCTYPE message [  
    <!ELEMENT message (#PCDATA)>  
    <!ATTLIST message  
        xml:lang CDATA #IMPLIED>  
]>
```

Dokumentumtípus-deklaráció (4)

- Egy külső DTD alkészletre mutató dokumentumtípus-deklaráció, mely belső DTD alkészletet is tartalmaz:
 - ```
<!DOCTYPE play SYSTEM "play.dtd" [
 <!ENTITY r "Rosencrantz">
 <!ENTITY g "Guildenstern">
 <!ENTITY rag "&r; and &g;">
]>
```

# Dokumentumtípus-definíció (1)

- Lehetővé teszi megszorítások előírását a dokumentum logikai szerkezetére, valamint tárolási egységek használatát támogatja.
- Jelölő deklarációkból áll.

# Dokumentumtípus-definíció (2)

- A dokumentumtípus-deklarációban társítható egy dokumentumhoz.
  - A dokumentumtípus-deklaráció mutathat egy külső DTD alkészletre, tartalmazhat egy belső DTD alkészletet, vagy teheti mindkettőt.
- A két alkészlet együtt alkotja egy dokumentum DTD-jét.
  - Ha a külső és belső alkészlet is van, akkor a belső alkészlet megelőzi a külső alkészletet.
    - Ez lehetővé teszi a külső alkészletben lévő egyed- és attribútumlista-deklarációk felülírását.

# Érvényesség (1)

- Egy XML dokumentum érvényes akkor, ha tartozik hozzá dokumentumtípus-deklaráció és a dokumentum eleget tesz a DTD által kifejezett megszorításoknak.
  - A specifikáció pontosan előírja, hogy mely megszorítások kell, hogy teljesüljenek. Ezek az úgynevezett **érvényességi megszorítások**.
  - Az érvényességi megszorítások megsértése hiba.

# Érvényesség (2)

- Nagyjából az alábbiakat jelenti:
  - A dokumentumtípus-deklarációban adott név meg kell, hogy egyezzen a gyökérelem elemtípusával.
  - A dokumentum minden egyes eleme deklarált kell, hogy legyen a DTD-ben és az elem tartalma meg kell, hogy feleljen a deklarációnak.
  - A dokumentum minden egyes attribútuma deklarált kell, hogy legyen a DTD-ben és az attribútum értéke meg kell, hogy feleljen a deklarációnak.
  - Ha egy attribútum kötelezőnek deklarált a DTD-ben, akkor az elemtípus minden egyes előfordulásához explicit módon meg kell adni a dokumentumban.

# Jelölő deklarációk

- Elemtípus-deklaráció
- Attribútumlista-deklaráció
- Egyeddeklaráció

# Elemtípus-deklaráció

- Egy elemtípus-deklaráció megszorítást ír elő az elem tartalmára.
- **Érvényességi megszorítás:** egy elemtípus nem deklaráálható egynél többször.



# Elemtípus-deklaráció: üres elemek

- **Érvényességi megszorítás:** az így deklarált elemeknek nem lehet tartalma.
- Példa üres elem deklarálására és használatára:

```
<!ELEMENT br EMPTY>
```

```
...
```

```


```

```

</br>
```

# Elemtípus-deklaráció: elemtartalom (1)

- Egy elemtípus elemtartalmú, ha az ilyen típusú elemek csak elemgyermekeket tartalmazhatnak (karakteres adatot nem), melyek opcionálisan *whitespace* karakterekkel választhatók el.
  - Megjegyzések és feldolgozási utasítások is megengedettek az elemgyermekek között.
- A deklaráció egy **tartalommodell**t határoz meg, egy olyan egyszerű nyelvtant, mely az elemgyermekek típusát és ezek megengedett sorrendjét szabályozza.
  - A tartalommodell egy reguláris kifejezéshez hasonló minta.

# Elemtípus-deklaráció: elemtartalom

## (2)

- Tartalommodell megadásánál használható konstrukciók:
  - Sorozat, mint például  
(street, city, zip, country)
  - Alternatíva lista, mint például  
(u1 | o1 | d1)
- Az előfordulások számát szabályozó speciális karakterek (a megelőző részkifejezésre vonatkoznak):
  - **?**: nulla vagy egy (0, 1) előfordulás
  - **+**: tetszőleges számú, de legalább egy (1, 2, 3, ...) előfordulás
  - **\***: tetszőleges számú (0, 1, 2, ...) előfordulás
- Összetett kifejezések alkothatók ezekből a konstrukciókból.

# Elemtípus-deklaráció: elemtartalom (3)

- Tegyük fel, hogy az f, g és h elemek üresként deklaráltak.

Deklaráció	The content of e	Az e elem érvényes példányai
<code>&lt;!ELEMENT e (f)&gt;</code>	Pontosan egy f elem	<code>&lt;e&gt;&lt;f/&gt;&lt;/e&gt;</code>
<code>&lt;!ELEMENT e (f?)&gt;</code>	0 vagy 1 f elem	<code>&lt;e&gt;&lt;/e&gt;</code> <code>&lt;e&gt;&lt;f/&gt;&lt;/e&gt;</code>
<code>&lt;!ELEMENT e (f+)&gt;</code>	1 vagy több f elem	<code>&lt;e&gt;&lt;f/&gt;&lt;/e&gt;</code> <code>&lt;e&gt;&lt;f/&gt;&lt;f/&gt;&lt;/e&gt;</code> <code>&lt;e&gt;&lt;f/&gt;&lt;f/&gt;&lt;f/&gt;&lt;/e&gt;</code> ...
<code>&lt;!ELEMENT e (f*)&gt;</code>	0 vagy több f elem	<code>&lt;e&gt;&lt;/e&gt;</code> <code>&lt;e&gt;&lt;f/&gt;&lt;/e&gt;</code> <code>&lt;e&gt;&lt;f/&gt;&lt;f/&gt;&lt;/e&gt;</code> ...
<code>&lt;!ELEMENT e (f,g,h)&gt;</code>	Egy f, egy g és egy h elem, ebben a sorrendben	<code>&lt;e&gt;&lt;f/&gt;&lt;g/&gt;&lt;h/&gt;&lt;/e&gt;</code>
<code>&lt;!ELEMENT e (f g h)&gt;</code>	Pontosan egy f, g vagy h elem	<code>&lt;e&gt;&lt;f/&gt;&lt;/e&gt;</code> <code>&lt;e&gt;&lt;g/&gt;&lt;/e&gt;</code> <code>&lt;e&gt;&lt;h/&gt;&lt;/e&gt;</code>

# Elemtípus-deklaráció: elemtartalom (4)

- **Példa:** olyan a elem deklarálása, mely páros (0, 2, 4, ...) számú b elemet kell, hogy tartalmazzon:
  - `<!ELEMENT a (b,b)*>`
- **Példa:** olyan a elem deklarálása, mely páratlan (1, 3, 5, ...) számú b elemet kell, hogy tartalmazzon:
  - `<!ELEMENT a (b,(b,b)*)>`
- **Példa:** olyan a elem deklarálása, mely 1, 2 vagy 3 b elemet kell, hogy tartalmazzon:
  - `<!ELEMENT a (b,(b,b?))>`

# Elemtípus-deklaráció: elemtartalom (5)

- **Érvényességi megszorítás:** az így deklarált elemek tartalma meg kell, hogy feleljen a deklaráció tartalommodelljének.
  - Az első elemgyermek előtt, az elemgyermek között és az utolsó elemgyermek után megengedettek *whitespace* karakterek, megjegyzések és feldolgozási utasítások.

# Elemtípus-deklaráció: vegyes tartalom (1)

- Egy elemtípus vegyes tartalmú, ha az ilyen típusú elemek karakteres adatot tartalmazhatnak elemgyermekkel vegyítve.
  - Korlátozható az elemgyermkek típusa, de a sorrendjük és az előfordulási számuk nem.

# Elemtípus-deklaráció: vegyes tartalom (2)

- Példa vegyes tartalmú elem deklarálására és használatára:

```
<!ELEMENT para (#PCDATA | link)*>
<!ELEMENT link (#PCDATA)>

...
<para>See
 <link>https://sass-lang.com/</link>
 and
 <link>https://stylus-lang.com/</link>.
</para>
```



# Elemtípus-deklaráció: vegyes tartalom (3)

- **Érvényességi megszorítás:** az egyedhivatkozások a helyettesítő szövegükkel történő helyettesítése után az így deklarált elemek karakteres adatot, CDATA-szakaszokat, megjegyzéseket, feldolgozási utasításokat kell, hogy tartalmazzanak, valamint olyan elemgyermekeket, melyek típusa megegyezik a tartalommodellben adott nevekkal.
- **Érvényességi megszorítás:** ugyanaz a név nem szerepelhet egynél többször a deklarációban.

# Elemtípus-deklaráció: csak szöveget tartalmazó elem (1)

- A vegyes tartalmú deklaráció egy speciális esete szolgál csak szöveget tartalmazó elem deklarálására.
- Példa:

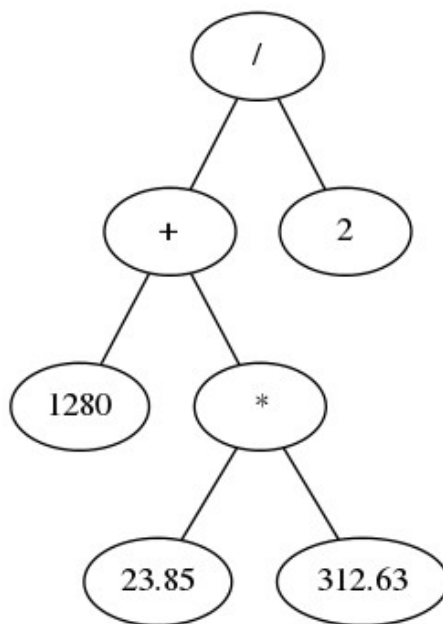
```
<!ELEMENT title (#PCDATA)>
...
<title>Angels & Demons</title>
```

# Elemtípus-deklaráció: csak szöveget tartalmazó elem (2)

- **Érvényességi megszorítás:** az egyedhivatkozások a helyettesítő szövegükkel történő helyettesítése után az így deklarált elemek karakteres adatot (beleértve a CDATA-szakaszokat), megjegyzéseket és feldolgozási utasításokat kell, hogy tartalmazzanak.

# Elemtípus-deklaráció: gyakorlati példa (1)

- Tartalommodellekkel mélyen egymásba ágyazott dokumentum struktúrák is leírhatók.
- Példa: bináris kifejezésfa
  - Az  $(1280 + (23.85 * 312.63)) / 2$  aritmetikai kifejezést ábrázoló kifejezésfa:



# Elemtípus-deklaráció: gyakorlati példa (2)

- expression.dtd:

```
<!ELEMENT node ((node | number), (node | number))>
<!ATTLIST node
 op CDATA #REQUIRED>

<!ELEMENT number (#PCDATA)>
```

# Elemtípus-deklaráció: gyakorlati példa (3)

- `expression.xml`:

```
<?xml version="1.0"?>
<!DOCTYPE node SYSTEM "expression.dtd">
<node op="/">
 <node op="+">
 <number>1280</number>
 <node op="*">
 <number>23.85</number>
 <number>312.63</number>
 </node>
 </node>
 <number>2</number>
</node>
```

# Attribútumlista-deklarációk (1)

- Egy adott elemtípushoz tartozó attribútumok nevét, típusát és alapértelmezett értékét (ha van ilyen) adják meg.
- Egy elemtípushoz egynél több is megadható, ezek tartalma összefésülésre kerül.
- Ha egynél több definíció kerül megadásra egy elemtípus ugyanazon attribútumához, akkor az első deklaráció kerül felhasználásra, a továbbiak pedig figyelmen kívül hagyásra kerülnek.
- Nem hiba egy nem deklarált elemtípushoz attribútumokat deklarálni.

# Attribútumlista-deklarációk (2)

- A következő szintaxisúak:  
    <!ATTLIST *név*  
        (*név típus alapértelmezett\_érték*)\*>
  - Az első *név* a deklarációban egy elem típusa.
  - A nevet olyan hármások követik, melyek mindegyike egy attribútumot deklarál az adott elemtípushoz.
    - A hármások egy attribútumnévből, attribútumtípusból és egy alapértelmezett érték deklarációból állnak.



# Attribútumlista-deklarációk (3)

- Példák:

```
<!ATTLIST item id ID #REQUIRED>
```

```
<!ATTLIST property
 name CDATA #REQUIRED
 value CDATA #REQUIRED>
```

```
<!ATTLIST answer correct (yes|no) "no">
```

```
<!ATTLIST todoList
 xmlns CDATA #FIXED
 "http://www.example.com/todoList">
```

# Attribútumtípusok

- Az attribútumtípusoknak három fajtája van:
  - **Sztring típus** (CDATA)
  - **Tokenizált típusok** (ID, IDREF, IDREFS, NMTOKEN, NMTOKENS)
  - **Felsorolt típusok** (felsorolások)
- A CDATA típusú attribútumok tetszőleges literális sztringet felvehetnek értékül.
- A többi típus érvényességi megszorításokat ír elő az attribútumértékekre.
  - Ezeknek az érvényességi megszorításoknak az ellenőrzése az attribútumérték normalizálása után történik.

# Attribútumtípusok: ID

- **Érvényességi megszorítás:** az ID típusú attribútumértékek olyan nevek kell, hogy legyenek, melyek nem fordulnak elő egy XML dokumentumban egynél többször ilyen típusú attribútumértékként.
  - Az ID értékek tehát egyértelműen kell, hogy azonosítsák az őket hordozó elemeket.
- **Érvényességi megszorítás:** egy elemtípushoz nem adható meg egynél több ID típusú attribútum.
- **Érvényességi megszorítás:** egy ID típusú attribútumhoz alapértelmezett érték deklarációként `#IMPLIED` vagy `#REQUIRED` kötelező.

# Attribútumtípusok: IDREF, IDREFS

- **Érvényességi megszorítás:** az IDREF típusú attribútumértékek olyan nevek kell, hogy legyenek, melyek megegyeznek az XML dokumentumban valamely elem egy ID típusú attribútumának értékével.
- **Érvényességi megszorítás:** az IDREFS típusú attribútumértékek egy vagy több név szóközzel elválasztott olyan listái kell, hogy legyenek, ahol minden egyes név megegyezik az XML dokumentumban valamely elem egy ID típusú attribútumának értékével.

# Attribútumtípusok: NMTOKEN, NMTOKENS

- **Érvényességi megszorítás:** az NMTOKEN típusú attribútumértékek névtokenek kell, hogy legyenek.
- **Érvényességi megszorítás:** az NMTOKENS típusú attribútumértékek egy vagy több névtoken szóközökkel elválasztott listái kell, hogy legyenek.

# Attribútumtípusok: felsorolások

- A felsorolt attribútumok deklarációja tartalmazza a megengedett értékeket.
- **Érvényességi megszorítás:** a deklarációban a névtokenek mind különbözőek kell, hogy legyenek.
- **Érvényességi megszorítás:** az ilyen típusú attribútumértékek meg kell, hogy egyezzenek a deklarációban szereplő valamely névtokennel.

# Alapértelmezett attribútumérték (1)

- Az alapértelmezett érték deklaráció információt szolgáltat arról, hogy kötelező-e az attribútum megadása, és ha nem, hogyan reagáljon egy XML feldolgozó, ha az attribútum hiányzik a dokumentumban.
- A következő választási lehetőségek állnak rendelkezésre:
  - #REQUIRED
    - Azt jelenti, hogy az attribútumot kötelező megadni.
  - #IMPLIED
    - Azt jelenti, hogy nem szükséges megadni az attribútumot és nincs alapértelmezett érték.
  - *literál*
    - Azt jelenti, hogy nem szükséges megadni az attribútumot és ez az alapértelmezett érték az attribútum hiányában.
  - #FIXED *literál*
    - Azt jelenti, hogy megadás esetén mindig ez kell, hogy legyen az attribútum értéke, és ez is az alapértelmezés az attribútum hiányában.

# Alapértelmezett attribútumérték (2)

- **Érvényességi megszorítás:** ha az alapértelmezett érték deklaráció a `#REQUIRED` kulcsszó, akkor az attribútumot meg kell adni minden olyan elemhez, mely az attribútumlista deklarációban adott típusú.
- **Érvényességi megszorítás:** az alapértelmezett érték meg kell, hogy feleljen az attribútumtípus szintaktikus megszorításainak.
- **Érvényességi megszorítás:** ha egy attribútumhoz a `#FIXED` kulcsszóval kerül megadásra alapértelmezett érték, akkor az attribútum példányainak értéke meg kell, hogy egyezzen azzal.



# Egyedek (1)

- Az XML dokumentumok fizikailag egyedeknek nevezett tárolási egységekből állnak.
- **Dokumentumegyed**nek nevezzük azt a tárolási egységet, mely kiindulási pontként szolgál az XML feldolgozó számára.
  - Ez a teljes dokumentumot tartalmazhatja.
- Minden egyednek van tartalma.
- A dokumentumegyed és a külső DTD alkészlet kivételével minden egyedet egy név azonosít.
  - A névvel rendelkező egyedek a DTD-ben kerülnek deklarálásra.

# Egyedek (2)

- Az egyedek lehetnek elemzettek vagy nem elemzettek.
- A nem elemzett egyedek lényegtelenek számunkra, ezért nem tárgyaljuk őket.
- A továbbiakban tehát minden egyed elemzett.

# Elemzett egyedek

- Egy elemzett egyed szöveget tartalmaz, melyet a helyettesítő szövegének nevezünk.
  - A szöveg jelölőket és/vagy karakteres adatot ábrázolhat.
- Szöveg behelyettesítésre szolgálnak.
- Egyedhivatkozásokban kerülnek hivatkozásra a nevükkel.
  - Egy egyedhivatkozás feldolgozása az elemzett egyed helyettesítő szövegének a hivatkozás helyére történő behelyettesítését eredményezi.

# Egyedek fajtái (1)

- Minden egyed egy általános egyed vagy egy paraméter egyed.
  - Az általános egyedek a dokumentumban történő felhasználásra szolgálnak.
  - A paraméteregyedek a DTD-ben történő felhasználásra szolgálnak.
- Az egyedek ezen két fajtája eltérő formájú hivatkozásokat használ és eltérően történik ezek kezelése.
- Ráadásul az egyedek ezen két fajtájához különböző névterek is tartoznak.
  - Egy azonos nevű általános egyed és paraméter egyed két különböző entitás.

# Egyedek fajtái (2)

- Minden egyed egy belső egyed vagy egy külső egyed.
  - Egy belső egyed tartalmát magában a deklarációban kerül megadásra.
  - Egy külső egyed tartalmát egy külső erőforrás szolgáltatja.

# Egyeddeklarációk (1)

- Belső általános egyed deklarációja:
  - Példa:  
`<!ENTITY unideb "University of Debrecen">`
    - Az egyedre az `&unideb;` egyedhivatkozással lehet hivatkozni.
- Belső paraméter egyed deklarációja:
  - Példa: `<!ENTITY % lists "ul | ol | dl">`
    - Az egyedre a `%lists;` egyedhivatkozással lehet hivatkozni. A hivatkozás kizárólag a DTD-ben kerül felismerésre!

# Egyeddeklarációk (2)

- Külső általános egyed deklarációja:
  - Példa: `<!ENTITY copyright SYSTEM "copyright.xml">`
    - Az egyedre az `&copyright;` egyedhivatkozással lehet hivatkozni.
- Külső paraméter egyed deklarációja:
  - Példa:

```
<!ENTITY % svg-animation.mod
 PUBLIC "-//W3C//ELEMENTS SVG 1.1
 Animation//EN"
 "svg-animation.mod">
```

    - Az egyedre a `%svg-animation.mod;` egyedhivatkozással lehet hivatkozni. A hivatkozás kizárólag a DTD-ben kerül felismerésre!

# Egyeddeklarációk (3)

- Ugyanaz az egyed egynél több egyeddeklarációban is deklarálható.
  - Ha ugyanaz az egyed egynél többször is deklarálásra kerül, akkor az első deklaráció kerül felhasználásra, a továbbiak pedig figyelmen kívül hagyásra kerülnek.



# Példa egyedek komplex használatára

```
<?xml version="1.0"?>
<!DOCTYPE book SYSTEM "book.dtd" [
 <!ENTITY gc "Grumpy Cat">
 <!ENTITY fw SYSTEM "foreword.xml">
 <!ENTITY ch1 SYSTEM "chapter1.xml">
 <!ENTITY ch2 SYSTEM "chapter2.xml">
 <!ENTITY ch3 SYSTEM "chapter3.xml">
 <!ENTITY biblio SYSTEM "bibliography.xml">
]>
<book>
 <author>&gc;</author>
 <title>How to be Grumpy?</title>
 &fw;
 &ch1;
 &ch2;
 &ch3;
 &biblio;
</book>
```

# Előre definiált egyedek

- Előre definiált egyedek speciális karakterekhez: amp ('&'), lt ('<'), gt ('>'), apos ('''), quot ('\"').
- Ezeket az egyedeket minden XML feldolgozó fel kell, hogy ismerje, függetlenül attól, hogy deklaráltak-e vagy sem.

# Jólformált egyedek (1)

- Egy belső vagy külső általános elemzett egyed jólformált, ha a helyettesítő szövege karakteres adatot tartalmaz elemekkel, karakterhivatkozásokkal, általános egyedhivatkozásokkal, CDATA-szakaszokkal, feldolgozási utasításokkal és megjegyzésekkel vegyítve.
  - A külső általános elemzett egyedek egy szövegdeklarációval kezdődhetnek, mely meghatározza a használt karakterkódolást.

# Jólformált egyedek (2)

- Az általános egyedek jólformáltságának következménye, hogy a logikai és fizikai szerkezetek megfelelő módon egymásba ágyazottak.
  - Minden nyitó, záró és üres elem címkét, elemet, megjegyzést, feldolgozási utasítást, karakter- és egyedhivatkozást egy egyed kell, hogy tartalmazzon.

# Néhány példa DTD-kre

- Checkstyle  
[https://checkstyle.org/dtds/configuration\\_1\\_3.dtd](https://checkstyle.org/dtds/configuration_1_3.dtd)
- MathML  
<http://www.w3.org/Math/DTD/mathml3/mathml3.dtd>
- MusicXML  
<http://www.musicxml.org/dtds/partwise.dtd>
- SVG  
<http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd>

# XML feldolgozók

- Egy XML feldolgozó egy szoftvermodul XML dokumentumok beolvasásához és az azok tartalmához és szerkezetéhez való hozzáféréshez.
  - Egy XML feldolgozó egy másik, alkalmazásnak nevezett modul nevében végzi munkáját.
- Az XML feldolgozók két osztályba sorolhatók: érvényesítő és nem érvényesítő.

# XML formátumok tervezési kérdései (1)

- Hogyan nevezzük el az elemeket és attribútumok?
  - Ajánlott irodalom:
    - Robert C. Martin. *Clean Code: A Handbook of Agile Software Craftsmanship*. Prentice Hall, 2009.
    - Robert C. Martin. *Tiszta kód: Az agilis szoftverfejlesztés kézikönyve*. Kiskapu Kft., 2010.
      - A *Beszédes nevek* című 2. fejezet nagymértékben vonatkozik XML nevekre is.

# XML formátumok tervezési kérdései

## (2)

- Hogyan nevezzük el az elemeket és attribútumok? (folytatás)
  - Néhány gyakorlati irányelv:
    - Olyan neveket használjunk, melyekből kiderül a szándék.
    - Kerüljük a félrevezető neveket.
    - Ne használjunk olyan túl általános zajszavakat a nevekben, mint például Data, Info vagy Content.
    - Használjunk kiejthető neveket.
    - Következetesten alkossunk neveket.
      - Következetesten használjunk egy névkonvenciót, mint például a *lowerCamelCase*.



# XML formátumok tervezési kérdései

## (3)

- Mikor használjunk attribútumokat?
  - Hogyan ábrázoljunk valamilyen információt: elemtartalommal vagy attribútumokkal?
    - Egy adatra vonatkozó metaadatokat attribútumokkal ajánlott megadni.
      - Lásd például az `xml:lang` attribútumot.
    - Tilos egy attribútumév megadása egy elemhez egynél többször.
      - Tehát nem adhatók meg egy attribútummal ugyanahhoz az elemhez olyan adatok, melyek egynél többször is előfordulhatnak.
    - Nem írható elő az attribútumok megadási sorrendje.
      - Ha tehát számít a sorrend, használjunk attribútumok helyett elemeket.

# XML formátumok tervezési kérdései (4)

- Mikor használjunk attribútumokat? (folytatás)
  - Példa:

```
<bookmarks title="Links of Personal Interest">

 <item url="https://www.brothers-brick.com/" title="The Brothers Brick">
 <tag>lego</tag>
 <tag>news</tag>
 <tag>reviews</tag>
 </item>

 <item url="https://www.facebook.com/thycatafalque/" title="Thy Catafalque">
 <tag>music</tag>
 <tag>ambient</tag>
 <tag>avantgarde</tag>
 <tag>experimental</tag>
 <tag>metal</tag>
 <tag>band</tag>
 </item>

</bookmarks>
```

# XML formátumok tervezési kérdései (5)

- További ajánlott irodalom:
  - *Google XML Document Format Style Guide*  
<https://google.github.io/styleguide/xmlstyle.html>