# Az XML és a Perl (1. rész)

Az egyik legáltalánosabb adattárolási módszer alkalmazása a legnépszerűbb programozási nyelvek egyikével.

Ha programozási nyelvről beszélünk, egy olyan, a számítógép által értelmezhető szövegre gondolunk, ami egy alkalmazást ír le. Erre jó példa a Perl. Ugyanakkor meglepően sűrűn előfordulhat az is, hogy az adatunk válik olyan bonyolulttá, hogy külön nyelvre van szükségünk a leírásához. Igen egyszerű adatnyelvek is léteznek már. Először ezekről esik szó.

## Adatnyelvek

Feltételezem, hogy némi gyakorlatra tettél szert a Perlben, és adatforrásként használtál már vesszővel elválasztott mezőket tartalmazó sorokat (más néven rekordokat) magába foglaló állományt. Léteznek ennél bonyolultabb nyelvek is, mint az, amelyikkel a `Makefile`-okban találkoztál. Ez egy program lefordításához szükséges adatokat tartalmazza. A szövegalapú adattárolás csúcsát bizonyos értelemben a Sendmail beállító-állománya jelenti. Ha eddig nem találkoztál vele, örülj neki, hogy megúsztad.

## Az SGML

Hozzávetőlegesen harminc évvel ezelőtt vetődött fel egy teljesen általános adatnyelv létrehozásának az ötlete. Hamar be kellett látni azonban, hogy lehetetlen lenne az adattárolás területén minden elképzelhető célra egyetlen nyelvet felkészíteni. Így az utóbbi feladattal küzdő csoport új célt tűzött ki magának. Egy olyan metanyelvet kívántak létrehozni, amellyel már könnyedén készíteni lehet egy jelölőnyelvet egy valóságos dokumentumhoz. Ennek a törekvésnek az eredménye az SGML (Standard Generalized Markup Language), amely 1986-ban vált nemzetközi szabvánnyá. Az SGML neve egy picit megtévesztő, hiszen ez önmagában nem egy jelölőnyelv (markup language), hanem egy metanyelv jelölőnyelvek meghatározásához. Minden ilyen jelölőnyelvet SGML-alkalmazásnak hívunk, ami ismét megtévesztő lehet azoknak, akik az alkalmazás szót csupán egy program szinonimájaként ismerik. A legtöbb SGML-alkalmazás nagyon sokáig ismeretlen volt az amerikai katonaságon és az egyes közhivatalok berkein kívül. A 90-es évek elején azonban színre lépett egy jelölőnyelv, amire a média is hamar rákapott: ez volt a HTML (HyperText Markup Language). Minden SGML-alkalmazás hasonlít a HTML-re annyiban, hogy egyetlen elemből áll, ami elemeket tartalmaz, s ezek további elemekből, illetve egyszerű szövegből állnak. Az elemek elejét és a végét kezdő- és zárócímkék jelentik, amelyek relációjelek között szerepelnek. Nagy vonalakban így fest egy SGML-dokumentum. Az SGML nagyon tág fogalom, és írásmódja számos dolgot engedélyez, ezért olyan nehéz SGML értelmezőt írni. Egy SGML-értelmező ráadásul addig nem is tud megfelelően értelmezni egy dokumentumot, amíg nincs meghatározva a jelölőnyelv nyelvtana, a DTD (Document Type Definition) – de erről majd később ejtünk szót.

## Az XML

Az SGML azt mutatta, hogy egy hozzá hasonló nyelv tökéletes választás lehetne a különböző programok közötti adatcsere megoldására. Hamar be kellett látni, hogy a hasonlónak egyben egyszerűbbnek is kell lennie. 1996-ban a World Wide Web Consortium (W3C) megbízott egy csoportot az SGML kistestvérének a kidolgozásával, ami elég egyszerű ahhoz, hogy adatcseréhez lehessen használni, különösképpen webes alkalmazások esetében. Ez volt az XML (eXtensible Markup Language). 1998 februárjában jelent meg az XML-ajánlás, amit a ➲ http://www.w3.org/TR/REC-xml címen olvashatsz el.

## Az XML működése

Minden XML-dokumentum egy elhagyható bevezetőből (prolog) és egy ezt követő elemből, a gyökérelemből (root element) áll. Minden elem elejét egy kezdőcímke jelöli, ami egy kisebb jelből (<), egy névből, egy elhagyható tulajdonság–érték párokból álló listából, és egy záró nagyobb jelből (>) áll. Minden elem végét zárócímke jelöli, ami hasonlít a kezdőcímkére, de a kisebb jel után közvetlenül egy perjel (/) található, és nincs tulajdonság–érték listája. Az üres elemek azok, amelyek nem tartalmaznak további elemeket, illetve szöveget. Ezeket olyan módon írhatjuk le rövidebben, hogy elhagyjuk a zárócímkét, ugyanakkor a kezdőcímkében a nagyobb jel elé egy perjelet szúrunk be. Egy nagyon egyszerű XML-dokumentum így fest:

```
<dokumentum>Szia, vilag!</dokumentum>
```

Szándékosan használtam a magyar dokumentum szót, hiszen a címke nevét én határozom meg. Ebben a példában mindössze egyetlen gyökérelemünk létezik, a dokumentum, és csak egyszerű szöveget tartalmaz. Nincsenek tulajdonságai (attributum) – adjunk neki egyet!

```
<dokumentum tipus="pelda">Szia, vilag!
</dokumentum>
```

Most már `dokumentum` elemünknek van egy tulajdonsága, a `tipus`, ennek értéke a `"pelda"`. Az XML-ben minden értéket idézőjelek közé kell tenni, és minden tulajdonságnak kötelező értéket kell adni. Az érték nélküli tulajdonságok elfogadottak az olyan SGML-alkalmazásokban, mint a HTML, de az XML-ben nem.
Az XML-t létrehozó csapat ezt annak érdekében határozta meg, hogy az XML-értelmezők egyszerűbbek és ezáltal gyorsabbak legyenek.
Az eddigi példákban dokumentumaink csak egyetlen elemből álltak, és az is csupán szöveget tartalmazott (hivatalosabban karaktereket). Az XML igazi ereje azonban strukturáltságában rejlik: szövegen kívül az elemek további elemeket is tartalmazhatnak. Lássunk erre is egy példát!

```
<kozert>
        <csoport nev="zoldseg">
                <aru>krumpli</aru>
        </csoport>
```

```
        <csoport nev="gyumolcs">
                <aru>alma</aru>
                <aru>korte</aru>
        </csoport>
</kozert>
```

Itt a gyökérelemünk a `kozert` nevet kapta. Ez két `csoport` nevű elemből áll, ezek `nev` tulajdonsággal rendelkeznek. Az egyik a `zoldseg`, míg a másik a `gyumolcs` értéket kapta. Ezeken belül `aru` elemeket találunk, a fenti két típusnak megfelelően elrendezve. Az XML szépségét mutatja, hogy ezt a szerkezetet többféleképpen is feldolgozhatjuk, ahogyan a sorozat következő részeiben bőven láthatunk rá példát. A DOM felépítés például egy faelrendezésben tálalja számunkra az adatokat. Mivel a cikksorozat az XML és a Perl kapcsolatával foglalkozik, ez a bevezető nem szolgálhat minden részletre kiterjedő leírással az XML-t illetően. Ha további tájékoztatásra lenne szükséged, látogass el a ➲ http://www.xml.com oldalra.

## Mi nem az XML?

Számos félreértésből származó tévhit kering az XML-ről, és arról, hogy miért is hozták létre. Az XML azonban

- ***nem*** a HTML helyettesítője. Az XML egy metanyelv, míg a HTML tényleges nyelv, amit az SGML metanyelvvel határoztak meg. A HTML-nek előre rögzített elemei vannak, az XML-alkalmazások elemeit az alkotóik határozzák meg. Ugyanakkor a HTML-t az XML szabályai alapján is írhatjuk, és elképzelhető, hogy a HTML következő változata egy XML-alkalmazásként fog megjelenni.
- ***nem*** olyan nyelv, ami otthoni kiadványszerkesztést tesz lehetővé a weben.
- ***nem*** a relációs adatbázisok helyettesítője. Kis adatmennyiség esetén használhatunk XML-alkalmazást relációs adatbázist használó programok közötti adatcserére, de nem túl megfontolt döntés egy egész adatbázist így tárolni.
- ***nem*** egy mindenre használható svájcibicska tetszőleges nyelvek létrehozásához.

Létre lehetne hozni egy egész programozási nyelvet XML-ben, de ronda lenne és teljes mértékben rugalmatlan. Az XML olyan adatok ábrázolására alkalmazható jól, amelyek természetüknél fogva hierarchikusak, illetve bizonyos matematikai modellek esetében (mint a gráfok).
Szerintem az XML érdekes megközelítése az adattárolásnak, és megér egy próbát. Mindent ki kell próbálni, és ha már használtad, eldöntheted, hogy tetszett-e vagy sem.

*Fülöp Balázs* (xut@freemail.hu)
18 éves, imádja a Túró Rudit, a Debian Linuxot és a teheneket. Kedvenc írója Slawomir Mro□ek. Leginkább a számítógépes hálózatok biztonsága érdekli. A BME VIK műszaki informatikus szak hallgatója.