XPath

dr. Paller Gábor

XPath

- Az XPath nyelv más specifikációkat szolgál ki.
- Elsődlegesen az XSLT céljaira készült.
- Funkciói
  - XML dokumentum részeinek kiválasztása, címzése.
  - Mintaillesztés (van-e olyan XML elem, amelyik illeszkedik egy mintára)
- Szöveges, nem XML szintaktika. Minthogy általában XML dokumentumban fordul elő, az XML számára fontos karaktereket védeni kell ("<" -> <)
- Az XPath csomópontok (node) halmazaként kezeli az XML dokumentumot. A csomópontok típusát (pl. elem, attribútum, szöveg) az XPath kifejezés képes megkülönböztetni.
- Egy XPath kifejezés értéke lehet
  - XML csomópontok halmaza (pl. XML elemek halmaza)
  - logikai érték (igaz-hamis)
  - lebegőpontos vagy egész szám
  - karaktersorozat

XPath adatmodell

- Az XPath elemző a következő adatelemeket különbözteti meg az XML dokumentumban. Minden adatelem típusnak van karaktersorozat-értéke, amely a típus által tárolt adat egy karaktersorozattá alakítva.
  - Szöveges csomópontok (text nodes) - Infoset Char típusba tartozó értékek sorozata.
  - Gyökérelem (root node) - megfelel az Infoset Document típusának (XML dokumentum verziója, standalone státusza, DTD jelenlétére utaló flag ...). Karaktersorozat-érték: az összes gyerek szöveges csomópont értéke összefűzve (rekurzívan is)
  - Elem csomópontok (element nodes) - megfelel az Infoset Element típusának. Karaktersorozat-érték: az összes gyerek szöveges csomópont értéke összefűzve (rekurzívan is)
  - Attribútum csomópont (attribute nodes) - Infoset Attribute típusnak felel meg. Karaktersorozat-érték az attribútum normalizált értéke (entitás- és karakterreferenciák feloldva)
  - Feldolgozásvezérlő csomópontok: a nevük a cél (ami a <? után közvetlenül jön), karaktersorozat-értékük a céltól a lezáró ?>-ig terjedő karaktersorozat.
  - Komment csomópontok: karaktersorozat-értékük a komment tartalma (<!-- --> közötti karaktersorozat)
  - Névterület-deklaráció csomópont: karaktersorozat-értékük a névterület URI.

XPath kifejezések illesztése

- Fontos fogalmak:
  - *Kontextus csomópont* (context node): az XML dokumentum egy csomópontja, aminek kontextusában (amihez képest) az XPath kifejezés kiértékelődik. A kontextus csomópont bármilyen csomópont lehet, bár a kijelölő kifejezések egy része csak akkor ad értelmes eredményt, ha a kontextus csomópont elem csomópont. A kontextus csomópont az XPath kifejezést használó környezetből jön, pl. az XSLT feldolgozó megpróbálhat illeszteni egy XPath kifejezést XML elemek egy csoportjára.
  - Egy *útvonal* (location path), ami a kontextus csomóponttól indul és a dokumentum hierarchiájának egy vagy több szintjére illeszkedik.
  - Az útvonal részei:
    - *Irány* (axis): a kontextus csomóponthoz képest milyen irányban indulunk.
    - *Csomópont teszt* (node test): a választott irányban indulva milyen csomópontot keresünk.
    - *Predikátumok* (predicates, nulla vagy több): logikai kifejezések, amelyekkel a csomópont tesztet finomíthatjuk.
  - A fenti elemekből egymás után több is írható / jelekkel elválasztva.

Útvonalak

- `child::para` - kiválasztja a kontextus csomópont "para" nevű elem csomópont gyerekeit (az XML elemek hierarchiájában közvetlenül az elem alatt levő elemet)
- Példa:

```
<section> <!-- Kontextus csomópont -->
  <para/> <!-- Kiválasztja ezt az elemet -->
</section>
```

- Ellenpélda:

```
<section> <!-- Kontextus csomópont -->
  <subsection>
     <para/> <!-- Nem választja ki ezt az elemet -->
  </subsection>
</section>
```

Útvonalak, 2.

- `child::*` - kiválasztja a kontextus csomópont összes elem csomópont gyerekét.
- Példa:

```
<section> <!-- Kontextus csomópont -->
  <para1/> <!-- Kiválasztja ezt -->
  <para2/> <!-- Kiválasztja ezt is -->
</section>
```

Útvonalak, 3.

- `child::text()` - Kiválasztja a kontextus csomópont összes szöveges gyermekét (a kontextus csomópont alatt közvetlenül a hierarchiában levő szöveges csomópontokat).
- Példa:

```
<section>
  Section text <!-- Kiválasztja ezt -->
  <para/> <!-- Ezt nem választja ki -->
  More section text <!-- Kiválasztja ezt is -->
</section>
```

Útvonalak, 4.

- `child::node()` - Kiválasztja a kontextus csomópont összes gyerekét típustól függetlenül.
- Példa:

```
<section>
  Section text <!-- Kiválasztja ezt -->
  <para/> <!-- Kiválasztja ezt is -->
  More section text <!-- Kiválasztja ezt is -->
</section>
```

Útvonalak, 5.

- parent::para - Kiválasztja a kontextus csomópont közvetlen szülőjét, ha az "para" elem csomópont.
- Példa:

```
<para> <!-- Kiválasztja ezt -->
  <item/> <!-- Kontextus csomópont -->
</para>
```

Útvonalak, 6.

- `attribute::name` - Kiválasztja a kontextus csomópont "name" nevű attribútumát.
- Példa:

```
<section
  name="this" <!-- Kiválasztja ezt -->
  othername="that" <!-- Ezt nem választja ki -->
/>
```

Útvonalak, 7.

- `attribute::*` - Kiválasztja a a kontextus csomópont összes attribútumát.
- Példa:

```
<section <!-- Kontextus csomópont -->
  name="this" <!-- Kiválasztja ezt -->
  othername="that" <!-- Kiválasztja ezt is -->
/>
```

Útvonalak, 8.

- `descendant::para` - kiválasztja a hierarchiában a kontextus csomópont alatt levő összes "para" nevű **elem** csomópontot (nem csak a közvetlen gyerekeket!)
- Példa:

```
<section> <!-- Kontextus csomópont -->
  <subsection>
     <para/> <!-- Kiválasztja ezt -->
  </subsection>
  <para/> <!-- És ezt is kiválasztja -->
</section>
```

Útvonalak, 9.

- `ancestor::div` - kiválasztja a hierarchiában a kontextus csomópont felett levő összes "div" nevű **elem** csomópontot (nem csak a közvetlen szülőt, hanem feljebb is).
- Példa:

```
<div> <!-- Kiválasztja ezt is -->
  <div> <!-- Kiválasztja ezt -->
    <para/> <!-- Kontextus csomópont -->
  </div>
</div>
```

Útvonalak, 10.

- `ancestor-or-self::div` - kiválasztja a kontextus csomópont összes "div" nevű **elem csomópont** ősét (hierarchiaszinttől függetlenül) és magát a kontextus csomópontot is, ha az "div".
- Példa:

```
<div> <!-- Ezt is -->
  <div> <!-- Ezt is -->
    <div/> <!-- Kontextus csomópont, kiválasztja ezt is
-->
  </div>
</div>
```

Útvonalak, 11.

- `descendant-or-self::para` - kiválasztja a kontextus csomópont "para" nevű **elem csomópont** leszármazottjait (hierarchiaszinttől függetlenül) ill. magát a kontextus csomópontot is, ha azt "para"-nak hívják.
- Példa:

```
<para> <!-- Kontextus csomópont, kiválasztja ezt is -->
  <para> <!-- Ezt is -->
    <para/> <!-- Ezt is -->
  </para>
</para>
```

- Ellenpélda:

```
<section> <!-- Kontextus csomópont, de nem választja ki
-->
  <para> <!-- Ezt kiválasztja -->
    <para/> <!-- Ezt is -->
  </para>
</section>
```

Útvonalak, 12.

- `self::para` - kiválasztja a kontextus csomópontot, ha az "para". Egyébként nem választ ki semmit.
- Példa:

```
<para> <!-- Kontextus csomópont, ezt kiválasztja -->
  <para/> <!-- Ezt nem választja ki -->
</para>
```

- Ellenpélda:

```
<section> <!-- Kontextus csomópont, ezt nem választja
ki -->
  <para/> <!-- Ezt sem  -->
</section>
```

Útvonalak, 13.

- `/` - kiválasztja a dokumentum legfelső szintű elemét. Ez a csomópont speciális, gyökérelem (root node) típusú és ennek a gyereke az XML dokumentum legfelső szintű elem csomópontja.
- A / összeköthető bármilyen egyéb útkijelölővel, de a kijelölés nem a kontextus csomóponttól indul, hanem a kontextus csomóponttal azonos dokumentumban levő legfelső szintű csomóponttól.
- Példa: `/child::doc/child::section`

```
<?xml version="1.0"?>
<doc>
  <section> <!-- Kiválasztja ezt -->
    ...
  </section>
</doc>
```

Útvonalak, 14.

- `following-sibling::chapter` - kiválasztja a kontextus csomóponttal a hierarchiában egy szinten levő, "chapter" nevű elem csomópontokat, amelyek a dokumentumban a kontextus csomópont után vannak. Üres halmaz a kimenete, ha a kontextus csomópont attribútum vagy névterület típusú.
- Példa:

```
<book>
  <foreword/> <!-- Kontextus csomópont (nem választja
ki) -->
  <chapter/> <!-- Kiválasztja ezt -->
  <chapter/> <!-- Ezt is -->
  <appendix/>
</book>
```

Útvonalak, 15.

- `preceding-sibling::chapter` - kiválasztja a kontextus csomóponttal a hierarchiában egy szinten levő, "chapter" nevű elem csomópontokat, amelyek a dokumentumban a kontextus csomópont után vannak. Üres halmaz a kimenete, ha a kontextus csomópont attribútum vagy névterület típusú.
- Példa:

```
<book>
  <foreword/>
  <chapter/> <!-- Kiválasztja ezt -->
  <chapter/> <!-- Kontextus csomópont -->
  <chapter/> <!-- Ezt nem választja ki -->
  <index/>
</book>
```

Útvonalak, 16.

- `following::chapter` - Kiválasztja az összes "chapter" nevű elem csomópontot, ami a kontextus csomópont után következik a dokumentumban, kivéve a kontextus csomópont gyerekeit.
- Példa:

```
<book>
  <chapter>
    <section/>
    <section> <!-- Kontextus csomópont -->
      <subsection/> <!-- Ezt nem választja ki -->
    </section>
  </chapter>
  <chapter/> <!-- Kiválasztja ezt -->
  <chapter/> <!-- Ezt is -->
</book>
```

Útvonalak, 17.

- `preceding::chapter` - Kiválasztja az összes "chapter" nevű elem csomópontot, ami a kontextus csomópont előtt van a dokumentumban.
- Példa:

```
<book>
  <chapter> <!-- Kiválasztja ezt is -->
    <section/>
    <section/>
  </chapter>
  <chapter/> <!-- Kiválasztja ezt -->
  <chapter/> <!-- Kontextus csomópont -->
</book>
```

Útvonalak, 18.

- `namespace::exsl` - kiválasztja az exsl lokális azonosítójú névterület-azonosítót. A kontextus csomópontnak elem típusúnak kell lennie, különben az eredmény üres halmaz.
- Példa:

```
<!-- "section" a kontextus csomópont -->
<section
  xmlns:exsl="http://www.server.com/exsl" <!--
Kiválasztja ezt -->
  xmlns:sxsl="http://www.server.com/txsl">
</section>
```

Gyakorlat

- 1. gyakorlat: Írjon XPath kifejezést, ami kiválasztja az összes "para" elemet, ami a kontextus csomópont "chapter" gyerekének leszármazottja!
- 2. gyakorlat: Írjon XPath kifejezést, ami kiválasztja az kontextus csomópont összes "para" nevű unokáját! (a kontextus csomóponttól a hierarchiában két szinttel lejjebb levő összes "para" elemet).
- 3. gyakorlat: Írjon XPath kifejezést, ami kiválasztja a kontextus csomóponttal egy dokumentumban levő összes "para" elemet!
- 4. gyakorlat: Írjon XPath kifejezést, ami kiválasztja a kontextus csomóponttal egy dokumentumban levő összes "item" elemet, aminek "olist" elem a közvetlen szülője!

Megoldások

- 1. gyakorlat: `child::chapter/descendant::para`
- 2. gyakorlat: `child::*/child::para`
- 3. gyakorlat: `/descendant::para`
- 4. gyakorlat: `/descendant::olist/child::item`

Predikátumok

- Predikátum: bármely XPath kifejezés lehet. A predikátum kiértékelése után a predikátum-kifejezés értékét boolean típusúra konvertáljuk és ezzel szűrjük a csomópontok halmazát.
- `child::para[position()=1]` - Kiválasztja a kontextus csomópont dokumentum-sorrendben első "para" nevű gyerekét. Minthogy a child:: u.n. előremutató irány (forward axis), ezért a pozíciót dokumentum-sorrend határozza meg. Hátramutató irányú útkijelölés esetén (pl. ancestor) fordított dokumentum-sorrendben van a pozíció számozva.
- Példa:

```
<section> <!-- Kontextus csomópont -->
  <subsection/>
  <para/> <!-- Kiválasztja ezt -->
  <para/>
</section>
```

- A predikátum működése:
  - child::para - kiválasztja az összes "para" nevű gyerekelemet. Az eredmény egy csomópont-halmaz, ami ezeket az elem csomópontokat tartalmazza.
  - A halmaz minden elemére kiértékeljük a "position()=1" kifejezést. Ez csak a legelső "para" elem csomópontnál lesz igaz, így az első lépés halmazából csak ezt az elemet választjuk ki.

Predikátumok, 2.

- `child::para[position()=last()]` - kiválasztja a kontextus csomópont utolsó "para" nevű gyerekét.
- Példa:

```
<section> <!-- Kontextus csomópont -->
  <subsection/>
  <para/>
  <para/> <!-- Kiválasztja ezt -->
  <subsection/>
</section>
```

Predikátumok, 3.

- `child::para[position()=last()-1]` - Kiválasztja a kontextus csomópont utolsó előtti "para" nevű gyerekét.
- Példa:

```
<section> <!-- Kontextus csomópont -->
  <subsection/>
  <para/>
  <para/> <!-- Kiválasztja ezt -->
  <para/>
 <subsection/>
</section>
```

Predikátumok, 4.

- `child::para[position()>1]` - Kiválasztja a kontextus csomópont összes "para" nevű gyerekét, kivéve az elsőt.
- Példa:

```
 <section>
   <subsection/>
   <para/>
   <para/> <!-- Kiválasztja ezt -->
   <para/> <!-- Kiválasztja ezt is -->
  <subsection/>
</section>
```

Predikátumok, 5.

- `child::para[attribute::type="warning"]` - Kiválasztja a kontextus csomópont azon "para" nevű elem csomópont gyerekét, amelynek van "type" attribútuma és annak értéke "warning".
- Példa:

```
<section> <!-- Kontextus csomópont -->
  <para/>
  <para type="info"/>
  <para type="warning"/> <!-- Kiválasztja ezt -->
</section>
```

Predikátumok, 6.

- `child::para[attribute::type="warning"][position()=5]`- Kiválasztja a kontextus csomópont ötödik "para" gyerekét, amelynek van "type" nevű attribútuma és annak értéke "warning".
- Példa:

```
<section> <!-- Kontextus csomópont -->
  <para type="warning"/>
  <para type="warning"/>
  <para type="warning"/>
  <para type="warning"/>
  <para type="warning"/> <!-- Kiválasztja ezt -->
  <para type="warning"/>
</section>
```

Predikátumok, 7.

- `child::para[position()=5][attribute::type="warning"]` - Kiválasztja a kontextus csomópont ötödik "para" elem csomópontját, **ha** annak van egy "type" attribútuma és annak értéke "warning".
- Példa:

```
<section> <!-- Kontextus csomópont -->
  <para type="warning"/>
  <para type="warning"/>
  <para type="warning"/>
  <para type="warning"/>
  <para type="warning"/> <!-- Kiválasztja ezt -->
  <para type="warning"/>
</section>
```

- Ellenpélda:

```
 <section> <!-- Kontextus csomópont -->
  <para type="warning"/>
  <para type="warning"/>
  <para type="warning"/>
  <para type="warning"/>
  <para type="info"/> <!-- Nem választ ki semmit! -->
  <para type="warning"/>
</section>
```

Predikátumok, 8.

- `child::chapter[child::title="Introduction"]` - Kiválasztja a a kontextus csomópont "chapter" nevű gyerekcsomópontjait, amelyeknek van olyan "title" nevű elem csomópont gyerekük, melynek értéke (=gyerek szöveges csomópontjainak értéke összefűzve) "Introduction".
- Példa:

```
<doc> <!-- Kontextus csomópont -->
 <chapter> <!-- Illeszkedik erre -->
    <title>Introduction</title>
  </chapter>
</doc>
```

- Példa2:

```
 <doc> <!-- Kontextus csomópont -->
  <chapter> <!-- Illeszkedik erre is! -->
     <title>Intro<tag>duction</tag></title>
   </chapter>
</doc>
```

Predikátumok, 9.

- `child::chapter[child::title]` - Kiválasztja a kontextus csomópont "chapter" nevű gyerek elem csomópontjai közül azokat, melyeknek van "title" nevű gyerek elem csomópontjuk.
- Példa:

```
<doc> <!-- Kontextus csomópont -->
  <chapter> <!-- Kiválasztja ezt -->
    <title>Introduction</title>
  </chapter>
  <chapter> <!-- Ezt nem választja ki -->
    <subtitle>Intro2</subtitle>
  </chapter>
</doc>
```

Predikátumok, 10.

- `child::*[self::chapter or self::appendix]` - Kiválasztja a kontextus csomópont gyerek elem csomópontjai közül a "chapter" és "appendix" nevűeket.
- Példa:

```
<doc> <!-- Kontextus csomópont -->
 <chapter/> <!-- Kiválasztja ezt -->
  <title/> <!-- Ezt nem választja ki -->
  <appendix/> <!-- Kiválasztja ezt is -->
</doc>
```

Predikátumok, 11.

- `child::*[self::chapter or self::appendix][position()=last()]` - kiválasztja a kontextus csomópont gyerek elem csomópontjai közül az utolsó "chapter" vagy "appendix" nevűt.
- Példa:

```
<doc> <!-- Kontextus csomópont -->
  <chapter/>
  <title/>
  <appendix/> <!-- Kiválasztja ezt -->
  <index/>
</doc>
```

Gyakorlat

- 1. gyakorlat: Írjon XPath kifejezést, ami kiválasztja a dokumentumban levő összes "title" nevű csomópontot, aminek van "subtitle" gyerek eleme!
- 2. gyakorlat: Írjon XPath kifejezést, ami kiválasztja a dokumentumban levő összes elem csomópontot, aminek van "id" nevű attribútuma!
- 3. gyakorlat: Írjon XPath kifejezést, ami kiválasztja az összes nem üres (tehát beágyazott szöveges vagy elem csomópontot tartalmazó) "title" nevű csomópontot a dokumentumban!
- 4. gyakorlat: Írjon XPath kifejezést, amelyik kiválasztja a dokumentumban levő összes element, aminek van "id" attribútuma és szöveges gyerekcsomópontja!
- 5. gyakorlat: Írjon XPath kifejezést, ami kiválasztja a dokumentumban levő összes csomópontot, ami egy bizonyos elem csomópont után következik dokumentum-sorrendben. Ez a bizonyos elem csomópont a legfelső szintű "doc" gyereke, "title"-nek hívják és van egy "id" nevű attribútuma, melynek értéke "hello".

Megoldások

- 1. gyakorlat: `/descendant::title[child::subtitle]`
- 2. gyakorlat: `/descendant::*[attribute::id]`
- 3. gyakorlat: `/descendant::title[child::node()]`
- 4. gyakorlat: `/descendant::*[attribute::id][child::text()]`
- 5. gyakorlat:
  `/child::doc/child::title[attribute::id="hello"]/following::*`

Rövidített szintakszis

- A rövidített szintakszis a teljes XPath szintaxis egy részhalmazának egyszerűsített leírását teszi lehetővé. Kizárólag kényelmi szolgáltatás, nem ad új funkciót a nyelvhez.
- `para` - kiválasztja a kontextus csomópont "para" nevű elem csomópont gyerekét. Eredeti szintaxis: child::para. Pl. /para és /child::para ekvivalensek.
- `*` - kiválasztja a kontextus csomópont összes gyerek elem csomópontját. Pl. /* és /child::* ekvivalensek.
- `text()` - kiválasztja a kontextus csomópont összes szöveges gyerekcsomópontját. Pl. a /child::text() és /text() ekvivalensek.
- `@name` - Kiválasztja a kontextus csomópont "name" nevű attribútum csomópontját. Pl. a /*[attribute::name] és a /*[@name] ekvivalensek.
- `@*` - Kiválasztja a kontextus csomópont összes attribútumát. Pl. az attribute::* és a @* ekvivalensek.
- `para[1]` - Kiválasztja a kontextus csomópont dokumentumsorrendben első "para" nevű gyerek elem csomópontját. Pl. /para[1] és /child::para[position()=1] ekvivalensek.
- `para[last()]` - Kiválasztja a kontextus csomópont utolsó "para" nevű gyerek elem csomópontját. Pl. /para[last()] és /child::para[position()=last()] ekvivalensek.

Rövidített szintaxis, 2.

- `chapter//para` - Kiválasztja a kontextus csomópont "chapter" gyerekének "para" nevű elem leszármazottait. Ekvivalens a child::chapter/descendant-or-self::node()/child::para kifejezéssel.
- Emlékeztető: /descendant-or-self::node() - a dokumentum összes csomópontja, kivéve az attribútum és namespace csomópontokat, de beleértve a gyökér csomópontot.
- `//para` - Kiválasztja a gyökér csomópont "para" nevű leszármazottjait (hierarchia szinttől függetlenül). Pl. a //para és a /descendant-or-self::node()/child::para ekvivalensek.
- Horror:
  - /descendant::para[1] - Kiválasztja a dokumentumsorrendben első "para" nevű leszármazottat (egy darab elemet).
  - //para[1] - kibontva: /descendant-or-self::node()/child::para[location()=1] - Bármely "para" elem a dokumentumban, amely a szülőjének első gyereke (potenciálisan sok elem).

Rövidített szintaxis, 3.

- . - kiválasztja a kontextus csomópontot. Pl. a . és a self::node() ekvivalensek.
- .. - Kiválasztja a kontextus csomópont közvetlen szülőjét. Pl. a .. és a parent::node() ekvivalensek.

Gyakorlatok

- 1. gyakorlat: Írjon XPath kifejezést rövidített szintaxisban, amely kiválasztja a dokumentum összes "item" elemét, amelynek "olist" a követlen szülője!
- 2. gyakorlat: Írjon XPath kifejezést rövidített szintaxisban, amely kiválasztja a kontextus csomópont összes "para" gyermekét, amelynek van "type" attribútuma és annak értéke "warning"!
- 3. gyakorlat: Írjon XPath kifejezést rövidített szintaxisban, amely kiválasztja a kontextus csomópont "chapter" nvű gyerekei közül azokat, melyeknek egy vagy több "title" gyerek csomópontjuk van és ezen csomópontok karaktersorozat értéke "Introduction".
- 4. gyakorlat: Írjon XPath kifejezést rövidített szintaxisban, amely kiválasztja a kontextus csomópont azon "employee" gyerek elem csomópontjait, amelyeknek van "secretary" és "assistant" attribútumuk!

Megoldások

- 1. gyakorlat: `//olist/item`
- 2. gyakorlat: `para[@type="warning"]`
- 3. gyakorlat: `chapter[title="Introduction"]`
- 4. gyakorlat: `employee[@secretary and @assistant]`

XPath kifejezések

- Emlékeztető: 4 fő adattípus
  - XML csomópontok halmaza (pl. XML elemek halmaza)
  - logikai érték (igaz-hamis)
  - lebegőpontos vagy egész szám
  - karaktersorozat
- Változók: $<változónév>, pl.$x. Kívülről jön az XPath kifejezés-kiértékelőbe, változónak értéket adni XPath kifejezésből nem lehet.
- Műveletek csomópont-halmazokon
  - Egy útvonalat kifejezésnek lehet tekinteni, amely csomópont-halmaz értéket ad vissza.
  - Egy operátor, a | értelmezett ilyen típusokon, ez a bal- és jobbérték csomópont-halmazának az unióját adja vissza (duplikátumok nélkül).
  - Példa: `/child::doc/child::section | /child::doc/child::chapter` - Kiválasztja a doc elem section és chapter nevű gyerekeit.
  - Predikátumokat ugyanúgy lehet használni csomópont-halmazokat visszaadó kifejezéseken, mint sima útvonalkijelöléseken. Példa: `(/child::doc/child::section | /child::doc/child::chapter)[position()=1]"`

Boolean értékek

- A boolean szokásos módon két értéket vehet fel: true vagy false.
- Ha valahol boolean érték megkövetelt és nem olyan típus szerepel, a kiértékelő automatikusan meghívja a boolean() függvényt, ami boolean-ná konvertálja (függvényt lásd később).
- Operátorok:
  - and
  - or
  - = - egyenlőségvizsgálat. Ha csomópont-halmazokra alkalmazzuk, akkor a kifejezés csak akkor igaz, ha van legalább egy pár csomópont az egyik, ill. másik csomópont-halmazban, amelyeknek karaktersorozat-értéke megegyezik.
  - !=
- Boolean operátorok kiértékelése:
  - Ha legalább egy paraméterük boolean, a másik paraméter boolean-ná lesz konvertálva a boolean() függvénnyel.
  - Ha legalább egy paraméterük szám, a másik paraméter számmá lesz konvertálva a number() függvénnyel.
  - Ha ezek egyike sem igaz, mindkét paraméter karaktersorozattá lesz konvertálva a string() függvénnyel.

Number és String értékek

- Operátorok: +,-,*
  - div - lebegőpontos osztás IEEE 754 szerint
  - mod - moduló számítás
- String - Unicode összehasonlítás, de nem karakterkód szerint (Unicode szerint lehetséges ugyanazt a karaktert többféle módon kifejezni, pl. alapkarakter+akcentus)

Beépített függvények: csomópont-halmazok

- *number* `last()` - A kiértékelési kontextus mérete
- *number* `position()` - A kontextus csomópont pozíciója a kiértékelési kontextusban.
- *number* `count(`*node-set*`)` - A csomópont-halmazban levő csomópontok száma.
- *node-set* `id(` *object* `)` - Visszaadja a kontextus csomóponttal egy dokumentumban levő csomópontot, amelynek egyedi azonosítója (ID DTD típus) megegyezik az argumentumában adottal. Ha az argumentuma csomópont-halmaz, előbb képezi azok karaktersorozat-értékét és ennek az értéklistának megfelelően keres egyedi azonosítót. **Figyelem:** az XPath számára csak az ID típusú attribútumok számítanak egyedi azonosítónak, tehát a függvény használatához a DTD kötelező!

Beépített függvények: csomópont-halmazok, 2.

- id példa:

```
<?xml version="1.0"?>
<!DOCTYPE doc [
 <!ELEMENT doc (section)>
 <!ELEMENT section ANY>
 <!ATTLIST section
   id ID #REQUIRED>
 <!ELEMENT item EMPTY>]>
<doc>
  <section id="hello">
    <item/>
  </section>
</doc>
```

- id("hello") - kiválasztja a "section" elemet.
- id("hello")/child::* - kiválasztja az "item" elemet.

Beépített függvények: csomópont-halmazok, 3.

- *string* `local-name(`*node-set*`)` - Visszaadja a paraméter csomópont-halmaz dokumentumsorrendben első elemének lokális nevét (pl.: <ns:item/> ->item)
- *string* `namespace-uri(`*node-set*`)` - Visszaadja a paraméter csomópont-halmaz dokumentumsorrendben első eleméhez tartozó névterület-azonosítót (nem prefixet!). pl.: <ns:item xmlns:ns="http://www.server.com/ns"/> ->http://www.server.com/ns
- *string* `name(`*node-set*`)` - Visszaadja a Visszaadja a paraméter csomópont-halmaz dokumentumsorrendben első eleméhez tartozó bővített nevet (névterület-azonosító:lokális név). pl.: <ns:item xmlns:ns="http://www.server.com/ns"/> ->http://www.server.com/ns:item

Beépített függvények: karaktersorozatok

- *string* `string(`object`)` - Visszaadja a paraméterének karaktersorozat-reprezentációját. Automatikusan alkalmazódik mindenhol, ahol karaktersorozat-érték elvárt.
  - Ha a paraméter csomópont-halmaz, a dokumentumsorrendben első csomópont karaktersorozat-értékét adja vissza.
  - Ha boolean, akkor true és false karaktersorozatokat ad vissza.
  - Ha szám, akkor a karaktersorozat értéket (speciális esetek: NaN, -NaN, Infinity, -Infinity).
- *string* `concat(`*string*, *string*, *string*`*)`- Összefűzi a paramétereiben kapott karaktersorozatokat és visszaadja az eredményt.
- *boolean* `starts-with(`*string*, *string*`)` - True eredményt ad vissza, ha az első paraméter-karaktersorozat a második paraméter-karaktersorozattal kezdődik.
- *boolean* `contains(`*string*, *string*`)` - True eredményt ad vissza, ha az első paraméter-karaktersorozat tartalmazza a második paraméter-karaktersorozatot.

Beépített függvények: karaktersorozatok, 2.

- *string* `substring-before(`*string*, *string*) - Visszaadja az első paraméter-karaktersorozatnak azt a részét, ami a második paraméter-karaktersorozat első előfordulása előtt van. Pl: substring-before("1999/04/01","/") -> 1999.
- *string* `substring-after(`*string*, *string*) - Visszaadja az első paraméter-karaktersorozatnak azt a részét, ami a második paraméter-karaktersorozat első előfordulása után van. Pl: substring-after("1999/04/01","/") -> 04/01.
- *string* `substring(`*string*, *number*, *number*?) - Visszaadja az első paraméter-karaktersorozatának azt a részét, ami a második paraméterében megadott pozíciótól kezdődik (opcionálisan a harmadik paraméterében megadott hosszban).
  - substring("12345",2,3) -> "234"
  - substring("12345",2) -> "2345"
- *number* `string-length(`*string*?`)` - Visszaadja a paraméterében megadott karaktersorozat hosszát. Ha nincs paramétere, a kontextus csomópont karaktersorozat értékének a hosszát adja vissza.

Beépített függvények: karaktersorozatok, 3.

- *string* `normalize-space(`*string*?`)` - Visszaadja a paramétereként megadott karaktersorozat szóköz-normalizált értékét. A vezető és záró "fehér karaktereket" levágja, a karaktersorozatban egymás után levő fehér karaktereket egy darab szóközzel helyettesíti.
- *string* `translate(`*string*, *string*, *string*) - Kicseréli az első karaktersorozatban a második karaktersorozatban levő bármely karaktert a harmadik karaktersorozatban azonos pozícióban levő karakterre. Pl. translate("bar","abc","ABC") -> "BAr"

Beépített függvények: boolean

- *boolean* `boolean(`*object*`)`- Boolean értékké konvertálja a paraméterét.
  - Ha a paraméter szám volt, az eredmény true, ha a szám nem 0 vagy NaN.
  - Ha a paraméter csomópont-készlet volt, az eredmény true, ha a csomópont-készlet nem volt üres
  - Ha a paraméter karaktersorozat volt, az eredmény true, ha a karaktersorozat nem üres.
- *boolean* `not(`*boolean*`)`- True-t ad vissza, ha a paramétere false volt és viszont.
- *boolean* `true()`- Mindig true-t ad vissza.
- *boolean* `false()`- Mindig false-t ad vissza.
- *boolean* `lang(`*string*`)` - True-t ad vissza, ha a kontextus csomópont xml:lang attribútumának értéke megegyezik a paraméterben megadottal. Pl.: kontextus csomópont: <para xml:lang="en"/>. Ekkor lang("en") true-t ad vissza.

Beépített függvények: Number

- *number* `number(`object?`)`- Számmá konvertálja a paraméterét
  - Ha a paramétere string, azt számmá konvertálja karakteres formából. Ha a karaktersorozatot nem lehet számmá konvertálni, az eredmény NaN.
  - Ha a paramétere boolean, a true értéket 1-gyé, a false-t 0-vá konvertálja.
  - Ha a paramétere csomópont-halmaz, először a string() függvényt hívja meg a csomópont-halmazra, majd ezt az értéket konvertálja a string típus konverziójánál leírtak szerint.
- *number* `floor(`*number*`)` - Visszaadja a legnagyobb egész számot, ami nem nagyobb a paraméterénél. floor(3.1)->3, floor(-3.1) -> -4
- *number* `ceiling(`*number*`)` - Visszaadja a legkisebb egész számot, ami már nagyobb a paraméterénél. ceiling(3.1)->4, ceiling(-3.1)->-3
- *number* `round(`*number*`)`- Kerekítés a szokványos kerekítési szabályok szerint. round(3.1)->3, round(-3.1) -> -3.

Beépített függvények: Number, 2.

- *number* `sum(`*node-set*) - Veszi a paraméter csomópont-halmaz elemeinek karaktersorozat-értékét, számmá konvertálja őket és összeadja.
- Példa:

```
<doc>
   <num>1</num>
   <num>2</num>
   <nonum>Hello</nonum>
   <num>3</num>
</doc>
```

- `sum(/child::doc/child::num)` -> 6.0