# 1.gyak

# XSLT

Extensible Stylesheet Transforma- tion Language (B˝ov´ıthet˝o Stilusle´ır´as Transzform´aci´os Nyelve) elnevez´esb˝ol sz´armazik. A tartalom m´odos´ıt´as´ara ir´anyul´o konverzi´os nyelvk´ent j¨ott l´etre a XSLT nyelv. Az XSLT r¨ovid jellemz´esek´ent azt mondhatjuk, hogy XML dokumen- tumot XML dokumentumba konvert´al a´t. A konverzi´o menet´et egy parancs a´llom´anyban adjuk meg,a mely szint´en XML form´atumu´. Az XSLT-hez kapc- sol´od´o ´allom´anyok:

* forr´as XML dokumentum
* a transzform´aci´ot le´ır´o dokumentum (rendszerint XSL kiterjeszt´essel)
* eredm´enyt tartalmaz´o XML dokumetum

A dokumentum kezel´esekor a XSLT nyelv a dokumentumot nem karakater sorozatk´ent, hanem dokumentum fak´ent kezeli. Ebben az ´ertelemben az XSLT szeml´elete nagyban hasonl´ıt a DOM szeml´elet´ere. Van azonban n´eh´any apr´o elt´er´es a k´etf´ele fa ´ertelmez´es k¨oz¨ott:

* az XSLT fa egyszeru˝bb, kevesebb r´eszletre t´er ki

a DOM fa m´odos´ıthat´o, a XSLT fa csak olvas´asra vagy csak ´ır´asra szolg´al

*•*

Az XSLT minta illeszt´ese sor´an teh´at a f´aban kell meghat´arozni az ´erintett csom´opontok halmaz´at. A feldolgozand´o elemek kijel¨ol´es´enek rugalmasnak

´es funkcion´alisan gazdagnak kell lennie, hiszen

* a f´aban t¨obb ir´anyban is lehet haladni
* az elemek mellett m´as csom´opont t´ıpusok is vannak
* egy mint´aval tetsz˝oleges csom´oponthalmazt le lehessen fedni
* lehet˝os´eget kell adni a tartalom ´es struktu´ra alapu´ szelekci´ora
* egyszeru˝ sz´am´ıt´asok elv´egz´es´et is biztos´ıtani kell

Az XSLT haszn´alat´ahoz, kipr´ob´al´as´ahoz h´arom elemre van szu¨ks´eg:

* forr´as XML ´allom´any
* transzform´aci´ot le´ır´o XSLT a´llom´any
* XSLT ´ertelmez˝o

Az XSLT forr´ask´odot XML ´allom´anyk´ent t´aroljuk. A forr´as´allom´any gy¨ok´er eleme egy

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0">

<!-- transzform´aci´os mint´ak -->

</xsl:stylesheet>

elem. Az XSLT transzfom´aci´os parancsok mindegyike az itt defini´alt [’http://www.w3.org/1999/X](http://www.w3.org/1999/X) n´evt´erhez tartoznak. A transzform´aci´os parancsok kiterjednek az el˝oz˝oekben

eml´ıtett mintailleszt´es mellett t¨obbek k¨oz¨ott az al´abbi mu˝veletekre:

* csom´opontok rendez´ese a feldolgoz´ashoz
* kifejez´esek megad´asa az eredm´eny XML dokumentum elo´all´ıt´as´ahoz
* csom´opontok feldolgoz´asi ciklusa
* v´altoz´ok l´etrehoz´asa
* az eredm´eny dokumentumba u´j struktu´ra elemek elo´all´ıt´asa
* felt´eteles transzform´aci´o v´egrehajt´as

A forr´as XML a´llom´anyban jel¨olju¨k ki, hogy az adott ´allom´anyon egy

transzform´aci´ot kell v´egrehajtani. A transzform´aci´os a´llom´any kijel¨ol´ese egy

<?xml-stylesheet type="text/xsl" href="forr´as-xsl" ?>

elemmel t¨ort´enik. Ez a feldolgoz´asi direkt´ıva a dokumentum gy¨ok´ereleme el˝ott szerepel. A ’href’ elemjellemz˝on´el specifik´aljuk a transzform´aci´ot le´ır´o XSLT a´llom´anyt.

A k¨ovetkez˝o kis minta bemutat egy X1.XML forr´as dokumentumot, egy CS1.XSL transzform´aci´ot megad´o ´allom´anyt ´es a transzform´aci´o eredm´eny´et tartalmaz´o X2.XML a´llom´anyt.

X1.XML:

<?xml version="1.0" ?>

<?xml-stylesheet type="text/xsl" href="xs1.xsl" ?>

<autok>

<auto rsz ="r11"> <tipus> Fiat </tipus> <ar>21233</ar> </auto>

<auto rsz ="r21"> <tipus> Opel </tipus> <ar> 41233</ar> </auto>

<auto rsz ="r31"> <tipus> Honda </tipus> <ar>71233</ar> </auto>

</autok>

XS1.XSL:

?xml version="1.0" ?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="1.0">

<xsl:template match="auto">

<xsl:value-of select ="./@rsz"/>: <b> <xsl:apply-templates/> </b> <br/>

</xsl:template>

<xsl:template match="tipus">

<i> <xsl:apply-templates/> </i>

</xsl:template>

</xsl:stylesheet>

X2.XML:

r11: <b> <i> Fiat </i> 21233 </b> <br/> r21: <b> <i> Opel </i> 41233 </b> <br/> r31: <b> <i> Honda </i> 71233 </b> <br/>

A transzform´aci´ot le´ır´o k´odban a ’match’ ´es ’select’ elemjellemzok ´ert´ekek´ent egy-egy XPath kifejez´est adunk meg, melyek kijel¨olik, hogy mely csom´opontokn´al kell haszn´alni a kapcsol´od transzform´aci´ot, illetve mely csom´opont ´ert´ek´et kell ki´ıratni az eredm´eny dokumentumba. A p´eld´aban szerepl˝o egyszeru˝bb XPath kifejez´esek mellett o¨sszetettebb formul´ak is k´epezhet˝ok.

# Az XSLT mu˝ko¨d´esi elve

Az XSLT feldolgoz´ok bemenete a feldolgozand´o XML dokumentumhoz k´esz´ıtett dokumentumfa. Az XSLT transzform´aci´os mu˝veletek mindegyike a f´ara vonatkozik. A feldolgoz´as kimenete az eredm´enyfa lineriz´al´as´aval kapott

XML dokumnetum lesz. A dokumentumfa feldolgoz´as´anak elve a fa csom´opontjainak bej´ar´as´an alapszik, azaz a feldolgoz´o a gy¨ok´er csom´opontt´ol kezdve bej´arja

a teljes f´at, s amely csom´oponthoz tartozik a´talak´ıt´asi, eredm´eny gener´al´asi szab´aly, ott el˝o´all´ıtja a k´ert XML dokumentum r´eszletet. Az eredm´eny fa ezen r´eszletek o¨sszevon´as´aval a´ll el˝o. Alapesetben a feldolgoz´as a gy¨ok´ert˝ol halad a gyerekek fel´e, egy top-down, deep-first m´elys´egi bej´ar´as megk¨ozel´ıt´esben. A feldolgoz´as pontos menete:

* A feldolgoz´o r´a´all a gy¨ok´er csom´opontra

Megn´ezi, hogy l´etezik-e feldolgoz´asi utas´ıt´as erre a csom´opontra. Ha nem sorra veszi a gyerek elemeket, s mindegyikre rekurz´ıvan elv´egzi ezt a vizsg´alatot. Ha nincs gyerek, meg´all a feldolgoz´asi l´anc.

*•*

Ha van illeszked˝o parancs minta, akkor azt ki´ert´ekelve el˝o´all egy eredm´eny fa r´eszlet. Alapesetben a csom´opont feldolgoz´asa ut´an meg´all a fel- dolgoz´asi l´anc, a gyerekek nem keru¨lnek ellen˝orz´esre. Lehet azon- ban tov´abbhalad´asi ig´enyt is adni, ekkor a gyerekek itt is vizsg´alatra

*•*

keru¨lnek rekurz´ıv m´odon. Ha nincs gyerek, akkor meg´all a feldolgoz´asi l´anc.

ha a fel- haszn´al´o egyetlen egyedi transzform´aci´ot sem defini´al, akkor is lesz erdem´eny XML fa. Az alap´ertelmezett feldolgoz´asi szab´alyok:

minden elemn´el (nem bele´ertve az elemjellemz˝ot ´es n´evteret), ha az nem sz¨ovegcsom´opont, az eredm´enyfa b˝ov´ıt´ese n´elku¨l tov´abbl´ep a gyerek elemek (nem bele´ertve a jellemz˝oket ´es n´evteret) fel´e.

*•*

a sz¨ovegcsom´opont eset´en ki´ırja az eredm´enybe ezt a csom´opontot (a sz¨ovegtartalmat)

*•*

* egy´eb csom´opontokat figyelmen k´ıvu¨l hagy Vegyu¨nk p´eld´aul egy alap XML dokumentumot:

<?xml version="1.0" encoding="UTF-8" ?>

<?xml-stylesheet type="text/xsl" href="xx2.xsl" ?>

<autok>

<auto rsz="r1">

<tipus> Fiat </tipus> <ar> 21233 </ar>

</auto>

<auto rsz="r2">

<tipus> Opel </tipus> <ar> 31233 </ar>

</auto>

</autok>

´es egy alap, u¨res transzform´aci´o le´ır´ot:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="1.0">

<!-- u¨res -->

</xsl:stylesheet>

Az eredm´eny XML dokumentum alakja:

<?xml version="1.0" encoding="UTF-8"?>

Fiat 21233

Opel 31233

Mint l´athat´o, a kapott XML dokumentum nem illeszkedik a helyesen formalts´ag k¨ovetelm´enyeire, hiszen nincs gy¨ok´er eleme a dokumentumnak, a sz¨ovegr´eszek elemen k´ıvu¨l helyezkednek el. Ezen p´elda alapj´an mg´allap´ıthat´o, hogy az XSLT transzform´aci´o nem garant´alja helyesen form´alts´ag felt´eteleit

az erem´eny dokumentumok eset´eben. Ez´ert a fejleszt˝onek kell u¨gyelniu¨k,

hogy olyan transzorm´aci´okat h´ıvjanak meg, melyek egyu¨ttesen biztos´ıtj´ak a helyesen form´alts´ag krit´eriumait.

# Az XSLT parancsai

## Mintailleszt´es

A dokumentumfa egy megadott csom´oponthalmaz´anak kijel¨ol´ese a

<xsl:template match=kif1 name=kif2 >

<!-- feldolgoz´o utas´ıt´asok -->

</xsl:template>

paranccsal t¨ort´enik. A tagban szerepl˝o ’kif1’ jellemz˝o egy keres´esi u´tvonalat jel¨ol, mely a feldolgozand´o csom´opontokat jel¨oli ki. A ’kif2’ egy nevet hordoz, megad´as´aval egy azonos´ıt´o n´ev k¨othet˝o ezen mint´ahoz. Mint kor´abban m´ar eml´ıtettu¨k, alapesetben a minta feldolgoz´asa ut´an a fa m´elys´egi bej´ar´asa meg´all. Vegyu¨k p´eld´aul a

<xsl:template match="/" > UDV!

</xsl:template>

parancs´allom´anyt, amely a gy¨ok´er elemre ill˝o mint´at tartalmaz, melyben a egy sz¨ovegkonstans a generalt kimenet. A feldolgoz´o a sz¨oveg ki´ır´asa ut´an le´all a fa feldolgoz´as´aval. Az ´ıgy kapott eredm´eny:

<?xml version="1.0" encoding="UTF-8"?> UDV!

A minta egyn´el t¨obb csom´opontot is tartalmazhat, ekkor mindegyikn´el v´egrahat´odik az eredm´enygener´al´as. Vegyu¨k az al´abbi p´eld´at:

<xsl:template match="auto" > udv!

</xsl:template>

A XSLT minta ki´ert´ekel´es´enek pontos ismerete is igen fontos a megfelel˝o

XSLT a´llom´any elk´esz´ıts´en´el. A ki´ert´ekel´es algoritumsa a k¨ovetkez˝okben

foglalhat´o ¨ossze. A ki´ert´ekel˝o motor els˝ok´ent elmegy a gy¨ok´er (dokumentum) elemhez. Megn´ezi, van az XSLT parancsf´aban illeszked˝o minta a gy¨ok´erre. Ha nincs, akkor az alap´ertelmez´esi parancsot hajtja v´egre, ami annyit mond, hogy menjen tov´abb a dokumentumfa m´elys´egi bej´ar´asa a gyerek csom´opontok fel´e (az elemejellemz˝o ´es n´evt´er nem tartozik ide). A sz¨oveges csom´opontok eset´eben az alap´ertelemez´esi szab´aly a sz¨oveges tartalom ki´ır´as´at ´ırja el˝o. Mivel ez lev´el csom´opont, a m´elys´egi keres´es is le´all. Explicit feldolgoz´asi parancs eset´en, ha az nem tartalmaz tov´abbl´ep´est, akkor meg´all a fa m´elys´egi bej´ar´asa az adott csom´opontn´al. A feldolgoz´o motor az illeszked´es a mint´ak vizsg´alat sor´an megkeresi, hogy mely minta illeszkedik a fa feldolgoz´as alatt a´ll´o csom´opontj´ara. Egy *p* minta akkor illeszkedik a *x* csom´opotra, ha a f´aban l´etezik olyan *y* csom´opont, melyre, mint kontexus elemre ki´ert´ekelve a *p* mint´at, az eredm´enyhalmaz mag´aba foglalja az *x*-et. Ez form´alisan u´gy is le´ırhat´o, hogy egy (*x*,*p*) p´aros viszony´aban a feldolgoz´o a

root(.)//(p)

XPath kifejez´est ´ert´ekeli ki, s ennek eredm´enyhalmaz´aban kell szerepelnie a

*x* csom´opontnak.

Gyakorl´ask´ent n´eh´any tipikus mint´at mutatunk be:

nev : a fa o¨sszes olyan eleme, melynek azonos´ıt´o neve ’nev’. Ekvivalens alakja root(.)//nev

*•*

* / : a dokumentum gy¨ok´er csom´opontja

/nev : a fa gy¨ok´er eleme, amely a dokumentum gy¨ok´er csom´opont alatt helyezkedk el

*•*

* nev[2] : azon elemek, melyek a szu¨leik m´asodik ’nev’ nevu˝ gyerekei
* text() : a dokumentum o¨sszes sz¨ovegt´ıpusu´ eleme

\* : a dokumentum o¨sszes n´evvel rendelkez˝o eleme (ebben a halmazban a sz¨ovegcsom´opontok, jellemz˝ok, n´evterek nem szerepelnek)

*•*

node() : a dokumentum-fa n´evt´er nem szerepel benne) o¨sszes elem csom´opontja (elemjellemz˝o,

*•*

* attribute::nev : a fa o¨sszes ’nev’ azonos´ıt´oju´ elemjellemz˝oje

Ahhoz, hogy eredm´enyhalmazban l´ev˝o csom´opontot feldolgozzon az XSLT motor, a kapcsol´od´o minta mellett m´eg arra is szu¨ks´eg van, hogy a fa m´elys´egi bej´aras el´erje a csom´opontot. ´Igy p´eld´aul egy

<?xml version="1.0" ?>

<xsl:stylesheet [xmlns:xsl="’http://www.w3.org/1999/XSL/Transform"’](http://www.w3.org/1999/XSL/Transform) version="1.0"

<xsl:template match="/"> Hello

</xsl:template>

<xsl:template match="auto"> Auto

</xsl:template>

</xsl:stylesheet>

XSLT ´allom´anyban hi´aba szerepel az ’auto’ nevu˝ elemek feldolgoz´asa, a feldolgoz´o nem fogja ezt elv´egezni, mivel a m´eyls´egi bej´ar´as nem jut le a

’auto’ nevu˝ csom´opontokig. Az elakad´as oka, hogy a dokumentum gy¨ok´er

csom´oponthoz l´ezetik explicit minta, s ez fog v´egrehajt´odni. Ezen minta parancs r´esz´eben nem szerepel tov´abbl´ep´esi utas´ıt´as, csakegy ’Hello’ sz¨oveg ki´ırat´asa, teh´at meg´all a fa m´elys´egi bej´ar´asa.

A fa bej´ar´as folytat´as´ahoz egy u´jabb parancsra az ’apply-template’ paranc- sra van szu¨ks´eg.

## apply-templates

A fabej´ar´as folytat´as´anak parancsalakja:

<xsl:apply-templates select = "kif1" >

<!-- rendez´es -->

</xsl:apply-templates>

Az apply-template utas´ıt´as a keres´est tov´abbfolytatja a ’select’ jellemz˝on keresztu¨l kijel¨olt csom´oponthalmazban. A parancs ’select’ jellemz˝oje op- cion´alis tag. Ha nem szerepel, a keres´es a csom´opont o¨sszes gyerekelem´en´el folytat´odik, azaz a ’select’ jellemz˝o alap´ertelmez´esi ´ert´eke:

child::node()

A ’select’-n´el megadott csom´opont kiv´alaszt´asi felt´etel is az XPath sz- abv´anyon alapszik. Az itt alkalmazhat´o formul´ak k¨ore t´agabb, mint amit a ’match’ opci´o megengedett: tetsz˝oleges tengelyir´anyba lehet mozogni. A k¨ovetkez˝o p´eld´ankban ez el˝obb bemutatot XSLT ´allom´anyt m´odos´ıtjuk u´gy, hogy a feldolgoz´as tov´abbmenjen a gy¨ok´er ¨osszes ’auto’ nevu˝ lesz´armazoottja fel´e:

<?xml version="1.0" ?>

<xsl:stylesheet [xmlns:xsl="’http://www.w3.org/1999/XSL/Transform"’](http://www.w3.org/1999/XSL/Transform) version="1.0"

<xsl:template match="/"> Hello

<xsl:apply-templates select="descendant::auto"/>

</xsl:template>

<xsl:template match="auto"> Auto

</xsl:template>

</xsl:stylesheet>

Ekkor els˝ok´ent a gy¨ok´erelemet dolgozza fel, melynek sor´an v´egrehajtja a tov´abbl´ep´esi utas´ıt´ast is. A feldolgozott tov´abbl´ep´esi elemek a gy¨ok´er elem ’auto’ nevu˝ lesz´armazottai lesznek.

Aza´ltal, hogy a ’select’ tulajdons´ag nemcsak m´elys´egi keres´est tesz lehet˝ov´e, a fa tetsz˝oleges ir´anyokban haladva j´arhat´o be. A nagy szabads´ag vis- zont v´egrehajt´asi probl´em´akhoz is vezethet, hiszen kialakulhat t¨obbek k¨oz¨ott v´egtelen ciklus is. Vegyu¨k ugyanis az al´abbi, form´alisan helyes XSLT paranc- ssort:

<?xml version="1.0" ?>

<xsl:stylesheet [xmlns:xsl="’http://www.w3.org/1999/XSL/Transform"’](http://www.w3.org/1999/XSL/Transform) version="1.0"

<xsl:template match="autok"> p:

<xsl:apply-templates select="child::auto"/>

</xsl:template>

<xsl:template match="auto"> c:

<xsl:apply-templates select="parent::node()"/>

</xsl:template>

</xsl:stylesheet>

ahol az adat´allom´any:

<?xml version="1.0" ?>

<?xml-stylesheet type="text/xsl" href="x1.xsl" ?>

<autok>

<auto rsz="r1">

<tipus> Fiat </tipus> <ar> 21233 </ar>

</auto>

<auto rsz="r2">

<tipus> Opel </tipus> <ar> 31233 </ar>

</auto>

</autok>

A fenti XSLT transzform´aci´ot v´egrehajtva egy fut´asi hib´at (’Too many nested apply-templates calls’) kapunk, amely arra utal, hogy tu´l sok egym´asba a´gyazott h´ıv´as j¨ott l´etre. Ugyanis az els˝o minta v´egrehajt´asa sor´an tov´abbl´ep a gyerek ’auto’ elemre. Ott a m´asodik minta aktiviz´al´odik, s tov´abbl´ep a szu¨l˝o (’autok’) elemre, ekkor viszont u´jra az els˝o minta lesz akt´ıv, z´ar´odik a k¨or, s egy v´egtelen v´egrehajt´asi ciklus alakult ki.

Az apply-template utas´ıt´as meghat´arozza a k¨ovetkez˝o l´ep´esekben fel- dolgoz csom´opontok halmaz´at, de a t´enyleges mu˝veletv´egrehajt´ast a t¨obbi ’template’ utas´ıt´asokra hagyja. Van ezen mechanizmus mellett egy olyan lehet˝os´eg is, amikor egy egys´egben adjuk meg, hogy mely csom´opontokon milyen mu˝veleteket kell v´egrehajtani. Ez a feldolgoz´asi ciklus utas´ıt´asa.

## for-each

A feldolgoz´o ciklusn´al param´eterk´ent egy csom´opont szelekci´os kifejez´es sz- erepel, melyet XPath form´atumban kell megadni. Az utas´ıt´as hat´asa´ra az- eredm´enyhalmaz minden egyes elem´en v´egrehajt´odik az utas´ıt´as magj´aban megadott mu˝veletsor. Az utas´ıt´as szintaktik´aja:

<xsl:for-each select="kif1">

<!-- feldogoz´o utas´ıt´asok -->

</xsl:for-each>

Az itt szerepl´o ’select’ r´esz jel¨oli ki a feldolgozand´o csom´opont halmazt. A feldolgoz´o utas´ıt´asok rendszerint az eredm´enyfa tartalm´at hat´arozz´ak meg

## value-of

A ’value-of’ utas´ıt´as seg´ıts´eg´evel lehet a forr´as´allom´any alapj´an k´epzett kife- jez´eseket kitenni a c´el´allom´anyba. A parancs alakja:

<xsl:value-of select="kif1" />

A ’kif1’ jel¨oli ki a ki´ıratand´o ´ert´ekeket. A kifejez´es jel¨olhet csom´opontot is, ebben az esetben a csom´opont sz¨oveges tartalma, ´es nem maga a csom´opont keru¨l bem´asol´asra az eredm´enyf´aba. A kifejez´es lehet azonos´ıt´o XPath u´tvonal mellett numerikus ´es sz¨oveges ´ert´ek is. P´eldak´ent vegyu¨k az al´abbi transz- formaci´o le´ır´ast:

<?xml version="1.0" ?>

<xsl:stylesheet [xmlns:xsl="’http://www.w3.org/1999/XSL/Transform"’](http://www.w3.org/1999/XSL/Transform) version="1.0"

<xsl:template match="autok">

<xsl:for-each select="auto">

<xsl:value-of select="tipus"/>

</xsl:for-each>

</xsl:template>

</xsl:stylesheet>

Eredm´enyu¨l az ’autok’ elem ’auto’ gyerekeinek ’tipus’ gyerek´enek sz¨oveg´ert´ekeit kapjuk meg, hiszen a ’value-of’ utas´ıt´as ’select’ param´etere most egy azonos´ıt´o nevet tartalmaz, teh´at ez egy xPath csom´opont kijel¨ol˝o kifejez´esnek tekinthet˝o.

Ekkor a csom´opont m¨og¨otti sz¨oveges ´ert´ek keru¨l ´at. A lista dokumentum sor- rendben adja vissza a sz¨oveges csom´opontok ´ert´ekeit. Ha az el˝oz˝omint´aban a ’value-of’ utas´ıt´ast a

<xsl:value-of select="’tipus’"/>

alakbn adjuk meg, az eredm´enyben a ’tipus’ sz¨ovegkonstans jelenik meg annyiszor, ah´any ’auto’ gyerekeleme van az ’autok’ csom´opontnak. A ’select’ tagban megadott kifejez´est az ´ertelmez˝o ki´ert´ekeli, s a kapott eredm´enyt teszi ki kimenetre. Ha p´eld´aul egy numerikus kifejez´est, pl. 1+2-t adjuk meg, akkor az eredm´enyben a 3-as ´ert´ek jelenik meg, azaz a

t=<xsl:value-of select="1+2"/>

eset´eben az eredm´eny az al´abbi alaku´ lesz:

t=3

A ’for-each’ utas´ıt´as dokumentum sorrendben dolgozza fel a csom´opontokat, viszont sokszor valamilyen ´ert´ek szerinti rendez´esre lenne szu¨ks´eg. Ehhez egy ku¨l¨on parancs ´all rendelkez´esre.

## sort

A rendez´esre szolg´al´o utas´ıt´as egy csom´oponthalmazhoz k¨othet˝o, form´atuma:

<xsl:sort select="kif1" order="kif2" collation="kif3" />

Az els˝o kifejez´es megadja a rendez´es kulcs´at. A ’kif1’ egy XPath szabv´any szerinti kifejez´est takar. Az ’order’ tulajdons´ag a rendez´es ir´any´at (cs¨okken˝o vagy n¨ovekv˝o) a´ll´ıtja be. A harmadik, kif3 tulajdons´ag a rendez´esi sorrendet hat´arozza meg. Ez adja meg az egyes karakterek k¨oz¨otti megel˝oz´esi rel´aci´ot. P´eldak´ent alak´ıtsuk ´at az el˝oz˝o p´eld´at u´gy, hogy a t´ıpusnevek ABC sorrendben jelenjenek meg:

<?xml version="1.0" ?>

<xsl:stylesheet [xmlns:xsl="’http://www.w3.org/1999/XSL/Transform"’](http://www.w3.org/1999/XSL/Transform) version="1.0"

<xsl:template match="autok">

<xsl:for-each select="auto">

<xsl:sort select="tipus"/>

<xsl:value-of select="tipus"/>

</xsl:for-each>

</xsl:template>

</xsl:stylesheet>

A ’value-of’ utas´ıt´as csak elemi sz¨oveges ´ert´ekeket m´asol a´t az eredm´enybe a forr´as f´ab´ol. Ha egy teljes far´eszletet, elemhierarchi´at szeretn´enk ´atvinni, akkor annak legegyszeru˝bb m´odja a m´asol´asi utas´ıt´asok haszn´alata.

## copy

A csom´opontok átm´asol´as´ashoz k´etf´ele utas´ıt´as ´all rendelkez´esre. Az elemi csom´opontot viszi ´at a ’copy’ parancs, m´ıg a p´arja a ’copy-of’ teljes csom´opont r´eszfa ´atm´asol´as´at elv´egzi. Az egyszeru˝bb, aktu´alis csom´opontot a´tviv˝o utas´ıt´as alakja:

<xsl:copy />

A teljes r´eszf´at a´tviv˝o utas´ıt´asn´al a form´atum:

<xsl:copy-of select="kif1" />

A ’kif1’ kifejez´es XPath form´atumban megadja, hogy mely csom´opont alatti r´eszt kell a´tm´asolni. Ha kifejez´es t¨obb csom´opontot is kijel¨ol, akkor mindegyik alatti r´eszfa ki´ır´asra keru¨l. A teljes dokumentum ´atm´asol´as´anak legegyszeru˝bb m´odja a k¨ovetkez˝o parancssor:

<?xml version="1.0" ?>

<xsl:stylesheet [xmlns:xsl="’http://www.w3.org/1999/XSL/Transform"’](http://www.w3.org/1999/XSL/Transform) version="1.0"

<xsl:template match="/">

<xsl:copy-of select="." />

</xsl:template>

</xsl:stylesheet>

A forr´as dokumentum ´atalak´ıt´asa sor´an nemcsak csom´opont ´atm´asol´as ´es t¨orl´es ig´enye l´ephet fel, hanem u´j csom´opontok l´etrehoz´as´anak szu¨ks´eges´ege is. Az XSLT t¨obb kapcsol´od´o utas´ıt´ast tartalmaz a ku¨l¨onb¨oz˝o csom´opont t´ıpusok megalkot´as´ara.

## Csom´opontok l´etrehoz´asa

U´j elemek elhelyez´es´ehez a

<xsl:element name="nev" >

<!-- tartalom -->

</xsl:element>

utas´ıt´as haszn´alhat´o fel. Ha egy u´j elemjellemz˝ore van szu¨ks´eg, akkor a

<xsl:attribute name="nev" select="kif1" />

utas´ıt´ast haszn´alhatjuk, melyben a select tulajdons´ag a l´etrehozott elem- jellemz˝o ´ert´ek´et adja meg. A fenti k´et csom´opont t´ıpus mellett tov´abbi csom´opont l´etrehoz´o utas´ıt´asok l´eteznek, minden egyedi csom´opont t´ıpusnak megvan a sajat l´etrehoz´o utas´ıt´asa. ´Igy p´eld´aul sz¨ovegcsom´opontot a

<xsl:text>

<!-- tartalom -->

</xsl:text>

alakban a hozhatunk l´etre.

## Felt´eteles utas´ıt´as v´egrehajt´as

Mivel a l´etrehozott eredm´eny dokumentum tartalma t¨obb ku¨l¨onb¨oz˝o, fut´asi id˝oben meghat´aroztott param´etert˝ol is fu¨gghet, szu¨ks´eg van olyan utas´ıts´ara, amely felt´etelhez k¨ot¨ott v´egrehajt´ast tesz lehet˝ov´e. Ezen felt´etel v´egrehajt´asi r´eszt is k´etf´ele m´odon jel¨olhetju¨k. Az egyszeru˝bb esetben csak egy igaz´ag l´etezik, s ekkor a

<xsl:if test="kif1" >

<!-- igaz-´ag -->

</xsl:if>

utas´ıt´as csak akkor hajtja v´egre az igaz´agban megadott utas´ıt´asokat, ha ’kif1’ kifejez´es is igaz ´ert´eket ad vissza. A kifejez´esnek logikai ´ert´eku˝nek kell lennie.

Ha t¨obb ku¨l¨onb¨oz˝o tev´ekenys´eg k¨ozu¨l v´alasztunk egyet, akkor a t¨obbsz¨or¨os el´agaz´as utas´ıt´as´at haszn´aljuk. Az utas´ıt´as alakja:

<xsl:choose >

<xsl:when test="kif1">

<!-- tev´ekenys´egek -->

</xsl:when>

<xsl:when test="kif2">

<!-- tev´ekenys´egek -->

</xsl:when>

<xsl:otherwise>

<!--- tev´ekenys´eek -->

</xsl:choose>

A szerkezetben mindegyik ´aghoz egy logikai kifejez´es tartozik. A vez´erl´es a sorrendben els˝o, igaz ´ert´eku˝kifejez´eshez tartoz´o ágat v´alasztja ki. Ha egyetlen egy ilyen ’when’ ´ag sincs, akkor az ’otherwise’ a´gra keru¨l a vez´erl´es. P´eldak´ent vegyu¨k azt az esetet, melyben a vizsgale´ır´o dokumentumban a sz´ammal megadott vizsgajegyet sz¨oveges le´ır´assal is cser´elju¨k le. Az indul´o

forr´asdokumentum:

<?xml version="1.0" ?>

<?xml-stylesheet type="text/xsl" href="x1.xsl" ?>

<vizsgak>

<vizsga>

<targy> Metamatika </targy> <diak> K234 </diak> <jegy> 2 </jegy>

</vizsga>

<vizsga>

<targy> Metamatika </targy> <diak> K761 </diak> <jegy> 3 </jegy>

</vizsga>

</vizsgak>

A kapcsol´od´o XSLT le´ır´as:

<?xml version="1.0" ?>

<xsl:stylesheet [xmlns:xsl="’http://www.w3.org/1999/XSL/Transform"’](http://www.w3.org/1999/XSL/Transform) version="1.0"

<xsl:template match="/">

<xsl:copy>

<xsl:apply-templates/>

</xsl:copy>

</xsl:template>

<xsl:template match="vizsgak">

<xsl:copy>

<xsl:apply-templates/>

</xsl:copy>

</xsl:template>

<xsl:template match="vizsga">

<xsl:copy>

<xsl:for-each select="\*" >

<xsl:choose>

<xsl:when test="name()=’jegy’">

<xsl:copy>

<xsl:choose>

<xsl:when test=" text() = 1 ">

<xsl:text> egy </xsl:text>

</xsl:when>

<xsl:when test=" text() = 2 ">

<xsl:text> ketto </xsl:text>

</xsl:when>

<xsl:when test=" text() = 3 ">

<xsl:text> harom </xsl:text>

</xsl:when>

<xsl:when test=" text() = 4 ">

<xsl:text> negy </xsl:text>

</xsl:when>

<xsl:when test=" text() = 5 ">

<xsl:text> ot </xsl:text>

</xsl:when>

</xsl:choose>

</xsl:copy>

</xsl:when>

<xsl:otherwise>

<xsl:copy-of select="." />

</xsl:otherwise>

</xsl:choose>

</xsl:for-each>

</xsl:copy>

</xsl:template>

</xsl:stylesheet>

A h´arom minta k¨ozu¨l az els˝o a gy¨ok´er csom´opontot m´asolja a´t, gyerekeit a tov´abbi mint´ak hat´arozz´ak meg. A m´asodik minta a gy¨ok´erelemet (’vizs- gak’) teszi le az eredm´enybe. A harmadik minta v´egzi el a ’vizsga’ elem a´tvitel´et. Ehhez ciklusban a gyerekek r´eszf´aj´at veszi sorra, s a t´ıpusokt´ol fu¨gg˝oen a´tm´asol (otherwise) vagy ´atalak´ıt (’jegy’ ´ag). Ezen esetben csak a csom´opontot m´asoljuk a´t, majd a sz¨oveges tartalm´at a r´egi sz¨ovegtartalomt´ol fu¨gg˝oen hat´arozzuk meg.

## variable

Az XSLT transzform´aci´ok sor´an el˝ofordulhat, hogy olyan transzform´aci´os for- mul´aink vannak, melyben egy olyan k¨oz¨os tag is szerepel, amely ¨onmag´aban

is o¨sszetett, ez´ert sokszori ism´etl´ese ´attekinthetetlen´e teszi az XSLT pro-

gramot. A hagyom´anyos programoz´asi nyelvekben ekkor egy r¨ovid´ıt´est, sz- imb´olikus nevet rendelnek ehhez a taghoz. Egy hasonl´o mechanizmus ´el az XSLT nyelvben is, ezek lesznek a v´altoz´ok (variable). A ’v´altoz´o’ elnevez´es kicsit zavar´o, mivel ezen szimb´olumok ´ert´eke nem m´odosulhat, ´ert´eke teh´at nem v´altozhat. Haszna´lat´anak indokai:

* besz´edes jel¨ol´es az ´ert´ekekre, pl. Pi a 3.1416 helyett
* formul´ak ´attekinthet˝ov´e t´etele a r´eszkifejez´esek kiemel´es´evel
* le´ır´as egyszeru˝s´ıt´ese az ism´etl˝od˝o r´esz helyettes´ıt´es´evel.

Az XSLT nyelvben k´etf´ele hat´ask¨oru˝ v´altoz´o l´etezik. A glob´alis v´altoz´o a teljes XSLT programban felhaszn´alhat´o, a lok´alis v´altoz´o csak a defin´ıci´os blokkj´aban. A v´altoz´o l´etrehoz´as´anak utas´ıt´asa:

<xsl:variable name="nev" as="adat\_tipus"> ... </xsl:variable>

A v´altoz´o ´ert´ek´et az elem sz¨ovegr´esz´eben vagy az elem ’select’ elem- jellemz˝oj´en´el lehet megadni. A v´altoz´o akkor lesz glob´alis hat´ask¨oru˝, ha a defini´al´asa a gy¨ok´er alatt t¨ort´enik. Ku¨l¨onben lok´alis ´erv´enyu˝. A v´alt´oz´ora a kifejez´esekben a

$nev

form´aban lehet hivatkozni, ekkor ´ert´eku¨kkel helyettes´ıt˝odnek.

A v´altoz´ok adatt´ıpus´anak megk¨ot´es´ehez az XMLSchema-t lehet segts´egu¨l h´ıvni, itt is az ott defini´alt adattpusokra lehet hivatkozni. A tipuskijel¨ol´esn´el az XMLSchema n´evtere egy´ertelmu˝s´ıt. T´ıpuskijel¨ol´es menet´et mutatja be a k¨ovetkez˝o kis p´elda:

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema">](http://www.w3.org/2001/XMLSchema)

<xsl:variable name="dx" as="xs:integer">

A p´eld´aban egy eg´esz ´ert´eku˝, dx nevu˝ v´altoz´ot hoztunk l´etre. A v´altoz´okezel´es t´agabb kontexus´at mutatja be k¨ovetkez˝o p´eld´ank, melyben a bemen˝o adat´allom´any k¨or¨ok adatait adja meg a k¨oz´eppont ´es egy tetsz˝oleges pont adatain keresztu¨l:

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="xx3.xsl" ?>

<korok>

<kor>

<kpont> <x>5</x><y>5</y></kpont>

<pont> <x>10</x><y>12</y></pont>

</kor>

<kor>

<kpont> <x>2</x><y>5</y></kpont>

<pont> <x>8</x><y>12</y></pont>

</kor>

</korok>

A transzfrom´aci´os XSLT programnak ebb˝ol olyan alakot kell el˝o´all´ıtania, melyben a k¨ort le´ır´o adatokat a teru¨lettel eg´esztju¨k ki. Ehhez el˝obb kisz´amoljuk a sug´ern´egyzetet, majd a teru¨letet. A sz´amt´asokn´al a v´altoz´okat a k¨ovetkez˝o c´elokra haszn´aljuk fel:

* a pi ´ert´ek t´arol´asa
* r´eszeredm´enzek t´arol´asa

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema">](http://www.w3.org/2001/XMLSchema)

<xsl:template match="/">

<xsl:apply-templates/>

</xsl:template>

<xsl:template match="korok">

<xsl:element name="korok">

<xsl:apply-templates/>

</xsl:element>

</xsl:template>

<xsl:variable name="pi" as="xs:float"> 3.1416 </xsl:variable>

<xsl:template match="kor">

<xsl:copy>

<xsl:copy-of select="kpont"></xsl:copy-of>

<xsl:copy-of select="pont"></xsl:copy-of>

<xsl:element name="terulet">

<xsl:variable name="dx" as="xs:integer">

<xsl:value-of select="number(kpont/x) - number(pont/x)"/>

</xsl:variable>

<xsl:variable name="dy" as="xs:integer">

<xsl:value-of select="number(kpont/y) - number(pont/y)"/>

</xsl:variable>

<xsl:variable name="d" as="xs:float">

<xsl:value-of select="($dx \* $dx + $dy \* $dy) \* $pi"/>

</xsl:variable>

<xsl:value-of select=" $d"></xsl:value-of>

</xsl:element>

</xsl:copy>

</xsl:template>

</xsl:stylesheet>

A p´eld´aban felhaszn´altuk egyr´eszt a ’number’ fu¨ggv´enyt, amely egy sz¨oveges

´ert´eket numerikus alakra konvert´al, m´asr´eszt azt a t´enyt, hogy numerikus mu˝veleteket nem az XSLT maga, hanem a be´agyazhat´o XPath t´amogat. Emiatt, p´eld´aul egy

<xsl:variable name="d" as="xs:float"> number($dx \* $dx + $dy \* $dy)

</xsl:variable>

kifejez´est ´erv´enytelennek tekint a feldolgoz´o.

A transzform´aci´oval kapott eredm´eny dokumentum:

<?xml version="1.0" encoding="UTF-8"?>

<korok>

<kor>

<kpont> <x>5</x><y>5</y></kpont>

<pont> <x>10</x><y>12</y></pont>

<terulet>232.4784</terulet>

</kor>

<kor>

<kpont> <x>2</x><y>5</y></kpont>

<pont> <x>8</x><y>12</y></pont>

<terulet>267.03598</terulet>

</kor>

</korok>

# 2. gyak

## for-each-group

Az adatb´aziskezel´esben megszokott dolog a rekordok csoportos´ıt´asa ´es ag- greg´alt adatok k´epz´ese. Az XSLT is megval´os´ıtotta ezt a funkci´ot a 2-es varzi´oj´aban. A csoportk´epz´esn´el meg kell adni a feldolgozand´o elemeket ´es a csoportk´epz´esi kifejez´est. Emellett m´eg sz´amos tov´abbi mu˝k¨od´esi param´eter is beall´ıthat´o. A feldolgoz´as sor´an minden csoportn´al meg´all, s lehet˝os´eget ad csoport-szintu˝ aggreg´aci´ora ´es elem szintu˝ r´eszeletez´esre. A rel´aci´os mod- ellt˝ol elt´er˝oen itt teh´at a csoport alkot´o tagjai is el´erhet˝ok ku¨l¨on-ku¨l¨on is. A csoportk´epz´es parancsa:

<xsl:for-each-group select="elemek" group-by="csoportk´epz´esi\_kifejezes" >...<

A feldolgoz´o magban k´et speci´alis szimb´olum haszn´alhat´o a csoport ele- meinek el´er´es´ere:

* current-group: az aktu´alis csoport elemeinek szekvenci´aja

current-grouping-key : az aktu´alis csoporthoz tartoz´o csoportk´epz´esi kifejez´es

*•*

Hivatkoz´askor a current group szimb´olum m¨og´e a csoportbeli elemeket kell

´erteni, ´ert´eke ez´ert nemcsak skal´ar lehet, hanem lista is.

Az aggreg´aci´ok v´egrehajt´as´ara felhaszn´alhat´ok az XSLT-XPath numerikus, aggreg´aci´os fu¨ggv´enyei. A rendelkez´esre ´all´o fu¨ggv´enyek:

* avg()
* sum()
* max()
* min()
* count()

A fu¨ggv´enyek argumentuma csom´opontrok, kifejez´esek halmaza. A doku- mentumban fellelhet˝o ’vizsga’ csom´opontok darabsz´am´anak kiir´as´at mutatja az az al´abbi parancs´allom´any:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema">](http://www.w3.org/2001/XMLSchema)

<xsl:template match="/">

<xsl:copy>

<xsl:element name="db">

<xsl:value-of select="count(//vizsga)"></xsl:value-of>

</xsl:element>

</xsl:copy>

</xsl:template>

</xsl:stylesheet>

A k¨ovetkez˝o p´eld´aban a vizsgajegyek dokumentum´ahoz k´esz´ıtu¨nk o¨sszes´ıt˝o jelent´est. A adatforr´as dokumentum:

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="xx4.xsl" ?>

<vizsgak>

<vizsga>

<targy> Matek</targy> <diak> K011</diak><jegy> 2</jegy>

</vizsga>

<vizsga>

<targy> Nemet</targy> <diak> K071</diak><jegy> 3</jegy>

</vizsga>

<vizsga>

<targy> Matek</targy> <diak> K071</diak><jegy> 2</jegy>

</vizsga>

<vizsga>

<targy> Nemet</targy> <diak> K101</diak><jegy> 1</jegy>

</vizsga>

<vizsga>

<targy> Matek</targy> <diak> K271</diak><jegy> 3</jegy>

</vizsga>

</vizsgak>

A konverzi´os ´allom´any:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema">](http://www.w3.org/2001/XMLSchema)

<xsl:template match="/">

<xsl:copy>

<xsl:apply-templates/>

</xsl:copy>

</xsl:template>

<xsl:template match="vizsgak">

<xsl:element name="osszesito">

<xsl:for-each-group select="vizsga" group-by="targy" >

<xsl:element name="targy" >

<xsl:element name="nev">

<xsl:value-of select=" current-grouping-key()"></xsl:va

</xsl:element>

<xsl:element name="atlag">

<xsl:value-of select="avg(current-group()/jegy)"></xsl:valu

</xsl:element>

</xsl:element>

</xsl:for-each-group>

</xsl:element>

</xsl:template>

</xsl:stylesheet>

Az eredm´enyu¨l kapott ¨osszes´ıt˝o:

<?xml version="1.0" encoding="UTF-8"?>

<osszesito>

<targy><nev> Matek</nev><atlag>2.3333333333333335</atlag></targy>

<targy><nev> Nemet</nev><atlag>2</atlag></targy>

</osszesito>

A csoportk´epz´esn´el az alap mu˝k¨od´esi m´odban a feldolgoz´o dokumen- tum sorrendben j´arja be a csoportok alkot´o elemeit. Akkor fog u´j csoport l´etrej¨onni, ha egy u´j ´ert´ek jelenik meg. Emiatt a csoportok el˝ofordul´asi sor- rendje ku¨l¨onb¨ozhet a csoprtk´epz´esi kifejez´es szerinti sorrendt˝ol.

Ha a csoportokat a csoportk´epz´esi kifejez´es szerint szeretn´enk megje- len´ıteni, akkor egy rendez´est el˝o´ır´o parancsot (sort) kell megadni a ’for-each- group’ els˝o gyerekelemek´ent. A rendezett kirat´asra ad p´eld´at az al´abbi minta.

Forr´as ´allom´any:

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="xx4.xsl" ?>

<vizsgak>

<vizsga>

<targy> Matek</targy> <diak> K011</diak><jegy> 2</jegy>

</vizsga>

<vizsga>

<targy> Angol</targy> <diak> K271</diak><jegy> 3</jegy>

</vizsga>

<vizsga>

<targy> Nemet</targy> <diak> K071</diak><jegy> 3</jegy>

</vizsga>

<vizsga>

<targy> Matek</targy> <diak> K071</diak><jegy> 2</jegy>

</vizsga>

<vizsga>

<targy> Nemet</targy> <diak> K101</diak><jegy> 1</jegy>

</vizsga>

<vizsga>

<targy> Matek</targy> <diak> K271</diak><jegy> 3</jegy>

</vizsga>

<vizsga>

<targy> Angol</targy> <diak> K171</diak><jegy> 2</jegy>

</vizsga>

</vizsgak>

Az XSL parancs´allom´any:

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema">](http://www.w3.org/2001/XMLSchema)

<xsl:template match="/">

<xsl:copy>

<xsl:apply-templates/>

</xsl:copy>

</xsl:template>

<xsl:template match="vizsgak">

<xsl:element name="osszesito">

<xsl:for-each-group select="vizsga" group-by="targy" >

<xsl:sort select="targy"/>

<xsl:element name="targy" >

<xsl:element name="nev">

<xsl:value-of select=" current-grouping-key()"></xsl:va

</xsl:element>

<xsl:element name="atlag">

<xsl:value-of select="avg(current-group()/jegy)"></xsl:valu

</xsl:element>

</xsl:element>

</xsl:for-each-group>

</xsl:element>

</xsl:template>

</xsl:stylesheet>

A kapott eredm´enydokumentum:

<?xml version="1.0" encoding="UTF-8"?>

<osszesito>

<targy><nev> Angol</nev><atlag>2.5</atlag></targy>

<targy><nev> Matek</nev><atlag>2.3333333333333335</atlag></targy>

<targy><nev> Nemet</nev><atlag>2</atlag></targy>

</osszesito>

## Param´eterezett formul´ak

A konverzi´os a´llom´anyok rugalmass´ag´anak egyik fontos eszk¨oze a param´eterezhet˝o transzform´aci´os s´em´ak mechanizmusa. Ezt azt jelenti, hogy a transzform´aci´os s´em´at (template) nem mint´ahoz k¨otju¨k, hanem k¨ozvetlenu¨l megh´ıvhatjuk, amihez egy azonos´ıt´o n´evvel kell rendelkeznie. Ekkor teh´at a s´ema nem ’match’ hanem egy ’name’, azonos´ıt´o nevet kijelu¨l˝o jellemz˝ovel rendelkezik:

<xsl:template name="azonosito\_nev" ...> ... </xsl:template>

A n´evvel rendelkez˝o s´em´at, az alprogramokhoz hasonl´oan egy ku¨l¨on paranc- ssal, a nevu¨k megad´as´asa mellett lehet aktiviz´alni. Az aktiviz´al´as utast´asa:

<xsl:call-template name="azonosito\_nev"> .... </xsl:call-template>

Az el˝obb az alprogramokhoz hasonl´ıtottuk az n´evvel ell´atott s´em´akat. A

hasonl´os´ag abban is megnyilv´anul, hogy itt is lehet param´etereket a´tadni

h´ıv´askor. A form´alis param´eterek kijel¨ol´ese a s´em´aban a ’param’ paranccsal t¨ort´enik:

<xsl:param name="parameter\_nev" ...> .... </xsl:param>

A param´eter haszn´alat´ara, jelleg´ere vonatkoz´olag t¨obb ku¨l¨onb¨oz˝o be´all´ıt´as tehet˝o. A legfontosabb jellemz˝ok, amelyek a ’param’ elemn´el el˝ofordulhatnak:

* as : a param´eter adatt´ıpus´anak megad´asa

select : az alap´ertelmez´esi param´eter´ert´ek, ha nem szerepelne aktu´alis param´eter

*•*

* required : a param´etert k¨otelez˝o megadni a h´ıv´askor

A h´ıv´askor az aktu´alis param´etereket a ’call-template’ elem gyerekek´ent sz- erepeltetik, a param´eter´atad´as utas´ıt´asa a ’with-param’ elem, melynek alakja:

<xsl:with-param name="parameter\_nev" select="ertek"> .. </ >

A param´eterezett s´em´ak haszn´alat´anak elemeit fogja o¨ssze az al´abbi p´elda, melyben egy ’m1’ nevu˝ s´ema szerepel, melyet ku¨l¨onb¨oz˝o aktu´alis param´eter´ert´ekkel aktiviz´alunk a ’korok’ ´es ’kor’ elemekn´el. A konverzi´os a´llom´any:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema">](http://www.w3.org/2001/XMLSchema)

<xsl:template match="/">

<xsl:apply-templates/>

</xsl:template>

<xsl:template match="korok">

<xsl:call-template name="m1">

<xsl:with-param name="p1" select="’fenn’" />

</xsl:call-template>

<xsl:apply-templates />

</xsl:template>

<xsl:template match="kor">

<xsl:call-template name="m1" >

<xsl:with-param name="p1" select="’lenn’" />

</xsl:call-template>

</xsl:template>

<xsl:template name="m1">

<xsl:param name="p1" as="xs:string" />

<xsl:value-of select="$p1" /> : XXXXXXXXXXX

</xsl:template>

</xsl:stylesheet>

A l´etrehozott ’m1’ s´em´anak egy, ’p1’ n´evvel rendelkez˝o param´etere van, amely sz¨oveges adatt´ıpusu´. A fenti ´allom´anyt az al´abbi bemen˝o dokumen- tumra futtatjuk le:

?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="xx6.xsl" ?>

<korok>

<kor>

<kpont> <x>5</x><y>5</y></kpont>

<pont> <x>10</x><y>12</y></pont>

</kor>

<kor>

<kpont> <x>2</x><y>5</y></kpont>

<pont> <x>8</x><y>12</y></pont>

</kor>

</korok>

A kapott eredm´eny dokumentumm, amely most egy ´erv´enytelen XML dokumentum, tartalma :

<?xml version="1.0" encoding="UTF-8"?> fenn : XXXXXXXXXXX

lenn : XXXXXXXXXXX

lenn : XXXXXXXXXXX

A mechanizmus mu˝k¨od´es´ere m´egegy tov´abbi p´eld´at veszu¨nk, melyben a forr´asdokumentum term´ekek le´ır´as´at tartalmazza, ahol a term´ek lehet CD vagy k¨onyv. A kimeneti dokumentumban egy sz¨oveges le´ır´as szerepel a term´ekekn´el, ahol a sz¨ovegr´esz form´atuma fu¨gg att´ol, hogy milyen term´ekt´ıpusn´al j´arunk. A forr´as´allom´any tartalma:

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="xx7.xsl" ?>

<termekek>

<termek>

<kod>IUC</kod> <kategoria>CD</kategoria> <ar>1233</ar>

</termek>

<termek>

<kod>HH1</kod> <kategoria>B</kategoria> <ar>833</ar>

</termek>

<termek>

<kod>OT7</kod> <kategoria>CD</kategoria> <ar>1433</ar>

</termek>

<termek>

<kod>DGW</kod> <kategoria>B</kategoria> <ar>454</ar>

</termek>

<termek>

<kod>JIU</kod> <kategoria>CD</kategoria> <ar>760</ar>

</termek>

</termekek>

A konverzi´os ´allom´any:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema"](http://www.w3.org/2001/XMLSchema) >

<xsl:template match="/">

<xsl:copy>

<xsl:apply-templates/>

</xsl:copy>

</xsl:template>

<xsl:template match="\*">

<xsl:copy>

<xsl:apply-templates/>

</xsl:copy>

</xsl:template>

<xsl:template match="termek">

<xsl:copy>

<xsl:call-template name="m2">

<xsl:with-param name="p2" select="kategoria"></xsl:with-param>

</xsl:call-template>

</xsl:copy>

</xsl:template>

<xsl:template name="m2">

<xsl:param name="p2" as="xs:string"></xsl:param>

Van egy

<xsl:choose>

<xsl:when test="$p2 =’CD’"> lemezunk</xsl:when>

<xsl:otherwise> konyvunk</xsl:otherwise>

</xsl:choose>

, melynekl ara <xsl:value-of select="ar"/>

</xsl:template>

</xsl:stylesheet>

A kapott eredm´eny dokumentum:

?xml version="1.0" encoding="UTF-8"?><termekek>

<termek>

Van egy lemezünk, melynekl ara 1233</termek>

<termek>

Van egy könyvünk, melynekl ara 833</termek>

<termek>

Van egy lemezünk, melynekl ara 1433</termek>

<termek>

Van egy könyvünk, melynekl ara 454</termek>

<termek>

Van egy lemezünk, melynekl ara 760</termek>

</termekek>

## Fu¨ggv´enyek

A param´eterezett s´em´ak az egyes csom´opontok komplett feldolgz´as´ara ad- nak rugalmas keretet. Sokszor viszont csak kisebb k´odr´eszlet ism´etl´es´ere van szu¨ks´eg, mint p´eld´aul egy k´odsz´am ellen˝orz´ese vagy egy sz´amjegy sz¨oveges alakra t¨ort´en˝o konvert´al´asa. Ezen feladatok ell´at´asa a fu¨ggv´enyek szolg´alnak.

A fu¨ggv´eny defini´al´asa hasonl´ıt a n´evvel ell´atott s´em´akra, viszont megh´ıv´asa minden olyan kifejez´esb˝ol megt¨ort´enhet, ahol ezt az adatt´ıpus illeszt´es megengedi. A fu¨ggv´eny nev´ere ´erv´enyes egy olyan megk¨ot´es, hogy a n´evnek n´evt´errel kiterjesztett alaku´nak kell lennie. A fu¨ggv´eny visszat´er´esi ´ert´eke a fu¨ggv´enyben

a v´egrehajt´asa sor´an el˝o´all´ıtott, ki´ırt ´ert´ek, nincs ku¨l¨on visszat´er´esi ´ert´ek ki- jel¨ol˝o utas´ıt´as.

A fu¨ggv´eny l´etrehoz´as´anak alakja:

<xsl:function name="nevter:nev" ... > ... </xsl:function>

A k¨ovetkez˝o p´elda egy u¨dv¨ozl˝o sz¨oveget el˝o´all´ıt´o fu¨ggv´enyt defini´al. Az ’udv’ nevu˝ fu¨ggv´eny a ’kl’ n´evhez rendelt n´evt´erhez tartozik.

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema"](http://www.w3.org/2001/XMLSchema)

xmlns:kl="kl1.kl1">

....

<xsl:function name="kl:udv" as="xs:string"> Hello vilag!

</xsl:function>

...

</xsl:stylesheet>

A fu¨ggv´eny megh´ıv´asa a nev´evel t¨ort´enik, a h´ıv´as XPath kifejez´esben is szerepelhet. A k¨ovetkez˝o p´eld´aban egy numerikus ´ert´ekkel visszat´er˝o fu¨ggv´eny defini´al´as´at ´es h´ıv´asat l´athatjuk.

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema"](http://www.w3.org/2001/XMLSchema)

xmlns:kl="kl1.kl1">

...

...

<xsl:value-of select="1+kl:f1()"/>

<xsl:function name="kl:udv" as="xs:integer">

<xsl:number value="2" />

</xsl:function>

...

</xsl:stylesheet>

A p´eld´aban szerepl˝o ’number’ elem konverzi´os utas´ıt´ask´ent ´ertelmezend˝o, amely numerikus t´ıpusra alak´ıtja a´t a megadott kifejez´est. A kifejez´es lehet csom´opont sz¨ovegtartalma is. A fu¨ggv´eny h´ıv´asa a ’value-of’ paranccsal t¨ort´enik, a p´eld´ankban mindig a 3-as ´ert´eket eredm´enyezve.

A fu¨ggv´enyekhez is k¨othet˝o param´eter´atad´as. A form´alis param´eterek megad´asa a m´ar eml´ıtett ’param’ elemmel t¨ort´enik, az aktu´alis param´etereknek a h´ıv´asi fu¨ggv´enynevet k¨ovet˝o z´ar´ojelben kell szerepelniu¨k. Az el˝oz˝o, a forr´asdokumentumot feldolgoz´o parancs´allom´anyt u´gy alak´ıtjuk ´at, hogy a kimeneti dokumentumban az ´ar mellett megadjuk, hogy ez olcs´o vagy dr´aga term´ek. Az ´ert´ekel´est egy fu¨ggv´eny v´egzi, melynek k´et bemen˝o param´etere van, a kateg´oria ´es az a´r. A konverzi´os a´llom´any:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema"](http://www.w3.org/2001/XMLSchema)

xmlns:kl="kl1.kl1">

<xsl:template match="/">

<xsl:copy>

<xsl:apply-templates/>

</xsl:copy>

</xsl:template>

<xsl:template match="\*">

<xsl:copy>

<xsl:apply-templates/>

</xsl:copy>

</xsl:template>

<xsl:template match="termek">

<xsl:copy>

<xsl:call-template name="m2">

<xsl:with-param name="p2" select="kategoria"></xsl:with-param>

</xsl:call-template>

</xsl:copy>

</xsl:template>

<xsl:template name="m2">

<xsl:param name="p2" as="xs:string"></xsl:param>

Van egy

<xsl:choose>

<xsl:when test="$p2 =’CD’"> lemezunk</xsl:when>

<xsl:otherwise> konyvunk</xsl:otherwise>

</xsl:choose>

, melynekl ara <xsl:value-of select="ar"/> , <xsl:value-of select="kl:f1

</xsl:template>

<xsl:function name="kl:f1" as="xs:string">

<xsl:param name="pk"></xsl:param>

<xsl:param name="pa"></xsl:param>

<xsl:if test="$pk=’CD’">

<xsl:choose>

<xsl:when test="$pa &lt; 1100"> olcso termek </xsl:when>

<xsl:otherwise> draga termek</xsl:otherwise>

</xsl:choose>

</xsl:if>

<xsl:if test="$pk = ’B’">

<xsl:choose>

<xsl:when test="$pa &lt; 600"> olcso termek </xsl:when>

<xsl:otherwise> draga termek</xsl:otherwise>

</xsl:choose>

</xsl:if>

</xsl:function>

</xsl:stylesheet>

A fu¨ggv´enyek seg´ıts´eg´evel lehet kikeru¨lni, legal´abbis r´eszben az XSLT nyelv azon korl´atj´at, hogy a v´altoz´ok statikusak, ´ert´ekeku¨k nem m´odos´ıthat´o. Vegyu¨nk p´eld´aul egy sima ciklust, amely eggyess´evel l´eptet 1 ´es 8 k¨oz¨ott. Ezt a feladatot nem oldhatjuk meg egy ciklusv´altoz´o l´eptet´es´evel, hiszen a v´altoz´o ´ert´eke nem m´odosulhat. A megold´as a fu¨ggv´enyek megfelel˝o, rekurz´ıv haszn´alat´aban rejlik. A megold´as magva az a lehet˝os´eg, hogy ugyanazon fu¨ggv´enyt h´ıvjuk meg rekurz´ıvan, de minden h´ıv´asn´al az aktu´alis param´etert l´eptetju¨k a szu¨ks´eges ´ert´ekben. A rutin belsej´eben ellen˝orizzu¨k, hogy szu¨ks´eg van-e tov´abbi rekurz´ıv feldolgoz´asra. Az al´abbi k´odr´eszlet egy megold´ast mutat be a kijel¨olt feladatra.

?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema"](http://www.w3.org/2001/XMLSchema)

xmlns:kl="kl1.kl1">

<xsl:template match="/">

<xsl:copy>

<xsl:element name="fo">

<xsl:value-of select="kl:f1(1,8)"></xsl:value-of>

</xsl:element>

</xsl:copy>

</xsl:template>

<xsl:function name="kl:f1">

<xsl:param name="p" as="xs:integer"/>

<xsl:param name="pm" as="xs:integer"/>

<xsl:if test="$p &lt; $pm">

<xsl:value-of select="$p"/>X

<xsl:value-of select="kl:f1($p+1, $pm)"/>

</xsl:if>

</xsl:function>

</xsl:stylesheet>

Az eredm´enyu¨l kapott dokumentum:

<?xml version="1.0" encoding="UTF-8"?>

<fo>

1X

2X

3X

4X

5X

6X

7X

</fo>

Egy o¨sszefoglal´o p´eld´aban a bemen˝o dokumentum term´ekek elad´asi a´rait tartalmazza. Az ig´enyelt XSLT k´odnak ebb˝ol a adatsorb´ol egy egyszeru˝ grafikon kell k´epeznie a HTML nyelv lehet˝os´egeire ´ep´ıtve. A bemen˝o doku- mentum:

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="xx9.xsl" ?>

<termekek>

<termek>

<kod>IUC</kod> <kategoria>CD</kategoria> <ar>1233</ar> <db>23</db>

</termek>

<termek>

<kod>HH1</kod> <kategoria>B</kategoria> <ar>833</ar><db>12</db>

</termek>

<termek>

<kod>OT7</kod> <kategoria>CD</kategoria> <ar>1433</ar><db>12</db>

</termek>

<termek>

<kod>DGW</kod> <kategoria>B</kategoria> <ar>454</ar><db>56</db>

</termek>

<termek>

<kod>JIU</kod> <kategoria>CD</kategoria> <ar>760</ar><db>31</db>

</termek>

</termekek>

A program el˝obb o¨sszes´ıti a kateg´ori´ank´enti darabsz´am ´ert´ekeket (’db’

elem), majd ehhez egy kis grafikont k´eszit a HTML nyelv ’HR’ elem´enek seg´ıts´eg´evel. Az oszlopok v´ızszintesen helyezkednek el, s az igaz´ıt´ast egy ’TABLE’ elemen keresztu¨l oldottuk meg. Az elk´eszu¨lt konverzi´os program:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" [xmlns:xs="http://www.w3.org/2001/XMLSchema"](http://www.w3.org/2001/XMLSchema)

xmlns:kl="kl1.kl1">

<xsl:output media-type="text/html" omit-xml-declaration="yes" include-conte

<xsl:variable name="vonals">12</xsl:variable>

<xsl:variable name="vonalarany">4</xsl:variable>

<xsl:template match="/">

<xsl:copy>

<xsl:element name="html">

<xsl:element name="head"> <xsl:element name="title">Proba</xsl:eleme

<xsl:element name="body">

<xsl:apply-templates select="termekek"></xsl:apply-templates>

</xsl:element>

</xsl:element>

</xsl:copy>

</xsl:template>

<xsl:template match="termekek">

<xsl:element name="table">

<!-- <xsl:attribute name="align" select="’left’"/> -->

<xsl:attribute name="border" select="3"/>

<xsl:element name="thead">

<xsl:element name="tr">

<xsl:element name="td"> nev </xsl:element>

<xsl:element name="td">osszdb</xsl:element>

</xsl:element>

</xsl:element>

<xsl:for-each-group select="termek" group-by="kategoria">

<xsl:element name="tr">

<xsl:element name="td">

<xsl:value-of select="current-grouping-key()"/>

</xsl:element>

<xsl:element name="td">

<xsl:attribute name="align">left</xsl:attribute>

<xsl:element name="hr">

<xsl:attribute name="width"><xsl:value-of select="sum(db)

<xsl:attribute name="size"><xsl:value-of select="$vonals"

</xsl:element>

</xsl:element>

</xsl:element>

</xsl:for-each-group>

</xsl:element>

</xsl:template>

</xsl:stylesheet>

A programban glob´alis v´altozokon keresztu¨l lett megadva a megjelen´ıt´eshez kapcsol´od´o k´et param´eter, a vonal vastags´aga ´es vonalhossz ar´anysz´am. Az

program tov´abbi saj´atoss´aga, hogy most a kimenet nem XML a´llom´any,

hanem HTML. Az ’output’ elemen keresztu¨l lehet megadni a kimeneti doku- mentum fejr´esz´et, a dokumentum t´ıpus´at. Az elk´eszu¨lt kimeneti dokumen- tum:

<html>

<head>

<title>Proba</title>

</head>

<body>

<table border="3">

<thead>

<tr>

<td> nev </td>

<td>osszdb</td>

</tr>

</thead>

<tr>

<td>CD</td>

<td align="left">

<hr width="92" size="12">

</td>

</tr>

<tr>

<td>B</td>

<td align="left">

<hr width="48" size="12">

</td>

</tr>

</table>

</body>

</html>

## attribute-set

Abban az esetben, ha a l´etrehozand´o dokumentumban t¨obb elemn´el is ugyana- zon elemjellemz˝ok fordulnak el˝o ism´etl˝od˝oen, akkor az ’attribute-set’ utas´ıt´as lehet˝os´eget ad a parancsok egyszeru˝s´ıt´es´ere. Ezen parancs egy jellemz˝ok hal- maz´at hozza l´etre, mely t¨obb jellemz˝ot fog ¨ossze az ´ert´ekeikkel egyu¨tt, s egy azonos´ıt´o n´ev k¨ot˝odik a halmazhoz. Az elemek l´etrehoz´asakor a n´evvel

bet¨olthet˝o a halmaz, s ekkor az elem felveszi a halmazhoz tartoz´o o¨sszes

jellemz˝ot. A halmaz be´ep´ıt´es´et az elembe a ’use-attribute-sets’ utas´ıt´as ha- jtja v´egre. Ha a halmazban megadott jellemz˝o ´ert´eke nem megfelel˝o egy adott elemn´el, akkor annak ´ert´ek´et a m´ar ismert ’attribute’ utas´ıt´ason keresztu¨l lehet m´odos´ıtani. A k¨ovetkez˝o p´elda egy ’s1’ nevu˝ halmazt defini´al, melyet a ’fopont’ elemn´el haszn´alunk fel.

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0" >

<xsl:template match="/">

<xsl:element name="fopont" xsl:use-attribute-set="s1" >

<xsl:attribute name="kor" select="11" /> Hello

</xsl:element>

</xsl:template>

<xsl:attribue-set name="s1">

<xsl:attribute name="nev" select="’bodri’" />

<xsl:attribute name="kor" select="8" />

</xsl:attribute-set>

</xsl:stylesheet>

A p´eld´aban a ’kor’ jellemz˝o alap´ertelmez´esi ´ert´eke lett ´at´ırva 11-re. A kapott eredm´enydokumentum:

<?xml version="1.0" encoding="UTF-8"?>

<fopont nev="bodri" kor="11" > Hello

</fopont>

## sequence

A szekvencia t¨obb elemi list´aj´at jelenti. A ’sequence’ oper´ator elemeket fu˝zz o¨ssze egy list´aba. Ez a funkci´o felhaszn´alhat´o t¨obbek k¨oz¨ott a v´altoz´ok

´ert´ekad´as´an´al is. A v´altoz´oknal a felvett ´ert´ek ugyanis nemcsak skal´ar sz¨oveg vagy numerikus ´ert´ek lehet, hanem tetsz˝oleges szekvencia, lista is. A list´at kerek z´ar´ojelben, a tagok felsorol´as´aval lehet megadni. A k¨ovetkez˝o p´eld´aban egy ’p1’ azonos´ıt´oju´ v´altoz´ot hozzunk l´etre, amely egy n´egyelemu˝ ´ert´eklist´at tartalmaz. A k´odr´eszlet tov´abbi soraiban el˝obb ki´ıratjuk az alkot´o elemek o¨sszeg´et a ’sum()’ aggreg´aci´os fu¨ggv´ennyel, majd egyenk´ent is feldolgozzuk a lista elemeit a ’for-each’ utas´ıt´as seg´ıts´eg´evel.

<xsl:template match="....">

<xsl:variable name="p1" as="xs:integer\*">

<xsl:sequence select="(1,2,3,8)"/>

</xsl:variable>

<xsl:value-of select="sum($p1)"/>

<xsl:for-each select="$p1"> <xsl:value-of select="."/></xsl:for-each>

</xsl:template>

Mint a p´eld´aban is l´athat´o, t¨obb egym´asut´ani ’sequence’ utas´ıt´as egy szekvenci´at hoz l´etre.

## analyze-string

A hat´ekony ´es rugalmas sz¨ovegelemz´eshez megval´os´ıtott´ak a regul´aris kife- jez´esre ´epu¨l˝o sz¨ovegvizsg´alatot is. A m´ar megimsert REGEX szab´alyoknak megfelel˝o illeszt´es vizsg´alatotot az ’analyze-string’ utas´ıt´assal lehet megval´os´ıtani, a parancs form´atuma:

<xsl:analyze-string select="sz¨oveg" regex="minta">

<!-- feldolgoz´as -->

</xsl:analyze-string>

A feldolgoz´asi r´eszben lehet megadni, hogy az illeszked˝o, vagy nem illeszked˝o sz¨ovegr´esszel mit kell tenni. A feldolgoz´ashoz k´et parancs ´all rendelkez´esre.

Az illeszked˝o sz¨ovegr´eszt a

<xsl:matching-substring>

<!-- muveletek -->

</xsl:matching-substring>

parancs ´ırja le. Minden illeszked˝onek tal´alt sz¨overg´esz egy-egy elemk´ent kezel˝odik a feldolgoz´asn´al. A nem illeszked˝o sz¨ovegr´eszeket a

<xsl:non-matching-substring>

<!-- muveletek -->

</xsl:non-matching-substring>

elemmel lehet feldolgozni. P´eldak´ent keressu¨k a ’leiras’ nevu˝ elemből a sz´amokat:

<xsl:template match="//leiras">

<xsl:analyze-string select="." regex="[0-9]+">

<xsl:matching-substring>

<xsl:value-of select="." /> <br/>

</xsl:matching-substring>

</xsl:analyze-string>

</xsl:template>

## key

A kulcs mechanizmussal lehet˝os´eg ny´ılik az egyes csom´opontok gyorsabb, k¨ozvetlen el´er´es´ere. Mu˝k¨od´ese a k¨ovetekez˝o l´ep´esekb˝ol ´all. Els˝ok´ent egy kulcs indexet kell l´etrehozni a ’key’ utas´ıt´as seg´ıts´eg´evel. Ha az megt¨ort´ent, akkor m´ar felhaszn´alhat´o ez az index az elemek keres´es´ere, melyhez a ’key()’ hivatkoz´asi fu¨ggv´enyt kell megh´ıvni. A ’key’ utas´ıt´as fontosabb param´eterei:

* a kulcsindex neve
* indexelt csom´opontok list´aja
* a kulcs´ert´ek meghat´aroz´asa a csom´opontokn´al

Az elemkers´esn´el a csom´opontot megad´o fu¨ggv´eny param´eterei:

* kulcsindex neve
* kulcs´ert´ek

Az eredm´eny az illeszked˝o elemek halmaza, melyb˝ol a szok´asos u´tvonal kifejez´esekkel tetsz˝oleges r´eszeket emelhetu¨nk ki. A k¨ovetkez˝o p´eld´aban az al´abbi alapdokumentumb´ol indulunk ki:

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="xx10.xsl" ?>

<termekek>

<termek>

<kod>IUC</kod> <kategoria>CD</kategoria> <ar>1233</ar> <db>23</db>

</termek>

<termek>

<kod>HH1</kod> <kategoria>B</kategoria> <ar>833</ar><db>12</db>

</termek>

<termek>

<kod>OT7</kod> <kategoria>CD</kategoria> <ar>1433</ar><db>12</db>

</termek>

<termek>

<kod>DGW</kod> <kategoria>B</kategoria> <ar>454</ar><db>56</db>

</termek>

<termek>

<kod>JIU</kod> <kategoria>CD</kategoria> <ar>760</ar><db>31</db>

</termek>

</termekek>

A forr´asban a ’termek’ csom´opont lesz a keresendo, indexelt egyed ´es in- dexkulcsk´ent a ’kod’ mezo szolg´al. Az indexfel´ep´ıt´est ´es keres´est megval´os´ıt´o XSLT program:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0">

<xsl:key name="K1" match="termek" use="kod" />

<xsl:template match="/">

<xsl:copy>

<xsl:element name="fo">

<xsl:value-of select="key(’K1’,’HH1’)/ar"></xsl:value-of>

</xsl:element>

</xsl:copy>

</xsl:template>

</xsl:stylesheet>

Az elk´eszu¨lt eredm´enydokumentum:

<?xml version="1.0" encoding="UTF-8"?>

<fo>833</fo>

## csom´opontok sorsz´amoz´asa

Listabeli elemek sorsz´am´ert´ekeinek kijelz´esi form´atum´at a´ll´ıtja be a ’num- ber’ utas´ıt´as. Eredm´enye egy olyan sz¨ovegcsom´opont, amely a form´atumzott alakot tartalmazza. A konverzi´os utas´ıt´as fontosabb param´eterei:

* count: a lesz´aml´aland´o csom´opontok kijel¨ol´ese

level: a sz´aml´al´as szintjelzoje, lehet egyszintu (single) ´es t¨obbszintu (multiple)

*•*

* from: a sz´aml´al´as kezdet´et kijel¨olo kifejez´es
* format: a sorsz´am megjelent´esi form´atuma, lehets´eges ´ert´ekei:
  + 1: norm´al sz´amjegyek
  + a: betuk (a,b,c,...)
  + A: nagybetuk
* value: k¨ozvetlen ´ert´ekmegad´as
* lang: nyelv kijel¨ol´ese
* grouping-separator: a sz´amjegyek csoportos´ıt´asi karaktere

grouping-size: a csoportos´ıt´as nagys´aga, mennyi karakter keru¨lj¨on egy csoportba

*•*

P´eldak´ent vegyu¨k az al´abbi indul´o dokumentumot:

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="xx10.xsl" ?>

<termekek>

<termek>

<kod>IUC</kod> <kategoria>CD</kategoria> <ar>1233</ar> <db>23</db>

<vevo>Peter</vevo> <vevo>Zoli</vevo>

</termek>

<termek>

<kod>HH1</kod> <kategoria>B</kategoria> <ar>833</ar><db>12</db>

<vevo>Anna</vevo>

</termek>

<termek>

<kod>OT7</kod> <kategoria>CD</kategoria> <ar>1433</ar><db>12</db>

<vevo>Peter</vevo><vevo>Gabi</vevo>

</termek>

<termek>

<kod>DGW</kod> <kategoria>B</kategoria> <ar>454</ar><db>56</db>

<vevo>Tomi</vevo><vevo>Anna</vevo><vevo>Peter</vevo>

</termek>

<termek>

<kod>JIU</kod> <kategoria>CD</kategoria> <ar>760</ar><db>31</db>

<vevo>Tomi</vevo>

</termek>

</termekek>

A fenti list´at u´gy alak´ıtjuk ´at, hogy csak a term´ekk´odot ´es a vevonevet

´ıratjuk ki. A ki´ırat´asn´al egy t¨obbszintu sorsz´amot rendelu¨nk a ki´ırt ele- mekhez. A konverzi´os ´allom´any:

<?xml version="1.0" encoding="UTF-8"?>

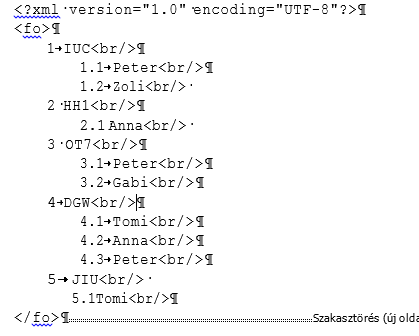
<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0">

<xsl:key name="K1" match="termek" use="kod" />

<xsl:template match="/">

<xsl:copy>

Mindk´et sorsz´amkirat´asn´al egyazon csom´opontlist´ara hivatkoztunk. A feldolgoz´o figyeli a bevont elemek szu¨lo-gzerek viszony´at, s ez alapj´an hat´arozza meg a hierarchia szintet jel¨olo ´ert´ekeket. A konverzi´o eredm´enye:



## Ért´ekek form´atumozott ki´ırat´asa

A piaci szoftver term´ekekn´el egzik fontos krit´erium a megfelelo megjelen´esi formatum biztos´ıt´asa. Egy v´allalatn´al igen fontos lehet p´eld´aul, hogy egy d´atum´ert´ek minden dokumentumban azonos m´odon jelenjen meg. A ku¨l¨onb¨ozo formatumok ugyanis t´eves ´ert´ekekhez, bizonytalans´aghoz, t¨obblet k¨olts´egekhez vezethetnek. A programoz´o szempontj´ab´ol az jelenti az egyik probl´em´at, hogy egy ´ert´ekhez, p´eld´aul d´atum´ert´ekhez nagyon sok ku¨l¨onb¨ozo megjelen´esi formatum tartozhat. Az XSL szerencs´ere rendelkezik ilyen form´atum be´all´ıt´o lehetos´eggel. A ku¨l¨onb¨ozo alapadat t´ıpusokhoz m´as ´es m´as a konverzi´os fu¨ggv´eny neve, mi most itt a k´et legfontosabbat vesszu¨k, a d´atum ´es sz´am konverzi´os fu¨ggv´enyt. A d´atumkonverzi´os fu¨ggv´enyek alakja:

format-date(d´atum, form´atum-maszk, nyelv, napt´ark´od)

A form´atum-maszk szimb´olumokkal ´ırja le az eredm´enysztring form´atum´at. Opcion´alisan megadhat´o az alkalmazand´o nyelv ´es napt´ar k´odja is. A form´atum maszkban megadhat´o fontosabb szimb´olumok:

* Y: ´ev
* M: h´onap sz´ammal
* MN: h´onap betuvel
* D: nap
* d: a nap ´even belu¨li sorsz´ama
* F: a h´et napja
* W: a h´et ´even belu¨li sorsz´ama
* w: a h´et h´onapon belu¨li sorsz´ama
* H: ´ora
* P: de ´es du jelz´ese
* m: perc
* s: m´asodperc

A form´atum szimb´olumokat a ’[]’ jelek k¨oz¨ott kell megadni, ami a z´ar´ojeleken

k´ıvu¨l van, az a megjelen´ıt´esbe k¨ozvetlenu¨l len´ıt´esi form´atum:

Y :[M]:[D] = 2007:08:23

Y :[MN]:[D] = 2007:AUGUST:23 Y001 :[d0001] = 007:0204

a´tmegy. N´eh´any minta megje-

A d´atumok kez´el´eshez kapcsolhat´o az aktu´alis d´atum vagy idopont lek´erdez´ese.

A kapcsol´od´o fu¨ggv´enyh´ıv´asok:

* current-date() : aktu´alis d´atum
* current-time() : aktu´alis ido A numerikus adatokn´al a

format-number(sz´am, form´atum-maszk)

A form´atum-maszkban a legfontosabb szabv´any szimb´olumok:

* # : sz´amjegy
* .: tizedespont
* NaN : nincs adat
* ,: csoport szepar´ator

Mivel a ku¨l¨onb¨ozo nyelvekben m´as ´es m´as karakter lehet a szepar´al´o, jel¨olo elem, ez´ert szu¨ks´eg lehet a formr´atumok egyszerustett kijel¨ol´es´ere is. A formatum definci´os parancs azonost´o n´evvel ell´atott form´atum l´etrehoz´as´ara alkalmas:

<xsl:decimal-format name="n´ev" decimal-separator="jel" NaN="jel" ...>

Az ´ıgy l´etrehozott form´atum felhaszn´alhat´o a ’format-number’ konverzi´os fu¨ggv´enyben, ahol harmadik param´eterk´ent szerepelhet:

format-number(sz´am, formatum-maszk,form´atum-k´od)

A k¨ovetkezo p´elda egy saj´at form´atum l´etrehoz´as´at ´es felhaszn´al´as´at mu- tatja be:

?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0">

<xsl:decimal-format name="f1" decimal-separator="!" NaN="nincs adat"/>

<xsl:variable name="x">NaN</xsl:variable>

<xsl:template match="/">

<xsl:copy>

<xsl:element name="fo">

<xsl:value-of select="format-number($x,’A####!#A’,’f1’)"

<xsl:value-of select=" format-date( current-date(),’[Y00

<xsl:message >hello</xsl:message>

</xsl:element>

</xsl:copy>

</xsl:template>

</xsl:stylesheet>

## Ku¨lso XSLT transzform´aci´os ´allom´any bevon´asa

Az XSLT transzform´aci´os programot is lehet modul´arisan fejleszteni, azaz a ku¨l¨onb¨ozo k´odr´eszletek ku¨l¨onb¨ozo forr´as´allom´anyb´ol j¨onnek o¨ssze. Egy adott transzform´aci´os a´llom´anyba egy m´asik transzform´aci´os ´allom´any tartalm´at a

<xsl:include href="allomany">

utast´assal lehet bevonni. A parancs hatr´as´ara a ku¨lso ´allom´any tartalm´at a´thozza az aktu´alis dokumentumba. Elofordulhat, hogy mind a ku¨lso ´es az aktu´alis ´allom´any tartalmaz egy utas´ıt´ast ugyanazon mint´ara. Ekkor a feldolg´asi algoritmust´ol fu¨gg, hogy melyik ´erv´enyesu¨l. Tesztjeinkben a k´esobb szereplo utas´ıt´as felu¨l´ırja a kor´abban szereplot. P´eld´aul az al´abbni esetben a gazda a´llom´any utas´ıt´asa ´erv´enyesu¨l:

xx10.xsl:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0">

<xsl:include href="xx11.xsl"/>

<xsl:template match="/">

<xsl:copy>

<xsl:element name="fo">

<xsl:apply-templates/>

</xsl:element>

</xsl:copy>

</xsl:template>

<xsl:template match="termek"> HELLO termek!

</xsl:template>

</xsl:stylesheet>

xx11.xsl:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0">

<xsl:template match="termek"> MI VAN termek!

</xsl:template>

</xsl:stylesheet>

Ha a beileszt´es az aktu´alis dokumentum v´eg´ere keru¨l, akkor a ku¨lso a´llom´anyban megadott utast´as fog ´erv´enyesu¨lni:

Ha a beileszt´es az aktu´alis dokumentum v´eg´ere keru¨l, akkor a ku¨lso a´llom´anyban megadott utast´as fog ´erv´enyesu¨lni:

?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet [xmlns:xsl="http://www.w3.org/1999/XSL/Transform"](http://www.w3.org/1999/XSL/Transform) version="2.0">

<xsl:decimal-format name="f1" decimal-separator="!" NaN="nincs adat"/>

<xsl:variable name="x">NaN</xsl:variable>

<xsl:template match="/">

<xsl:copy>

<xsl:element name="fo">

<xsl:apply-templates/>

</xsl:element>

</xsl:copy>

</xsl:template>

<xsl:template match="termek"> HELLO termek!

</xsl:template>

<xsl:include href="xx11.xsl"/>

</xsl:stylesheet>