# A SAX mu˝ko¨d´esi modellje

A SAX API nyelv c´elja olyan API felu¨let biztos´ıt´asa, amely lehet˝ov´e teszi a megadott XML dokumentum szerkezet´enek felt´ar´as´at. . A SAX elnevez´es a Simple API for XML (egyszeru˝ programoz´asi felu¨let az XML-hez) elnevez´esb˝ol sz´armazik. A SAX egyszeru˝s´ege t¨obb pontban is megnyilv´anul:

* csak olvassa a dokumentumot, nem m´odos´ıt
* csak szekvenci´alisan, egyszer olvassa a´t a dokumentumot
* kis er˝oforr´asig´enyu˝

Mire j´o teh´at a SAX? A SAX seg´ıts´eg´evel lehet felt´arni, hogy milyen

´ep´ıt˝oelemek vannak a dokumentumban. A gazda program a SAX feldol- goz´on keresztu¨l t´arja fel az XML dokumentum szerkezet´et. A gazdaprogram feladata, hogy megfelel˝oen reag´aljon az egyes felt´art elemekre. Ha gazdapro- gram semmilyen esem´enyre sincs felk´eszitve, akkor a SAX feldolgoz´o a´tfut´asa az XML dokumentumon hat´as n´elku¨l marad a gazdaprogramban.

A SAX feldolgoz´o ´es a gazdaprogram k¨oz¨otti kommunik´aci´o struktu´r´aja

az u´gynevezett CALLBACK mechanizmuson nyugszik. Ennek a l´enyege

az, hogy a gazdaprogramban szerepelni kell olyan feldolhoz´o moduloknak, melyek alkalamasak az egyes felmeru¨l˝o esem´enyek kezel´es´ere. A gazdapro- gramnak ezenfelu¨l k¨oz¨olni is kell a SAX modullal, hogy mely esem´enyn´el mely modulja a felel˝os v´egrehajt´o. Az egyes kezel˝o modulok k¨ot´ese ut´an a SAX modul elkezdi szekvenci´alisan feldoglozni a XML dokumentumot. Ha struktu´ra esem´eny k¨ovetkezik be, mint p´eld´aul u´j elem kezdet´enek ´eszlel´ese, a SAX modul megkeresi, hogy melyik gazdanyelvi modul felel˝os ez´ert az esem´eny´ert. Ha megtal´alta a megfelel˝o esem´enykezel˝ot, akkor ´atadja neki a vez´erl´est az o¨sszegyu¨jt¨ott param´eterekkel egyu¨tt. Az esem´eny feldolgoz´asa ut´an a vez´erl´es u´jra ´atkeru¨l a SAX modulhoz, amely az el˝obbi helyr˝ol foly- tatja tov´abb az XML dokumentum felt´ar´as´at. A SAX modul ´es a gatdaprogram k¨oz¨otti sikeres kommunik´aci´ohoz sz- abv´anyos´ıtani kell a k¨oztu¨k fen´all´o interf´eszt. Az egys´eges´ıt´es u´gy t¨ort´enik, hogy a SAX keretrendszer tartalmaz egy olyan oszt´alyv´azat, interface-t, melyben adott, hogy mely met´odusokat fogja megh´ıvni adott esem´enykor. Az interface pontosan megadja megh´ıvand´o met´odusok szignatu´r´aj´at, nev´et

´es param´eterez´es´et. A SAX modell az al´abbi esem´enyoszt´alyokat defini´alja:

* dokumentum kezdete
* dokumentum v´ege
* elem kezdete
* elem v´ege
* karakter r´esz el´er´ese
* entity ´eszlel´ese
* direktiva ´eszlel´ese
* n´evt´er defin´ıci´o

A gazdaprogram feladata, hogy hozzon l´etre egy olyan oszt´alyt, mely illeszkedik erre az interface-re ´es minden met´odus az ig´enyelt esem´enykezel´est hajtja v´egre.

Hiszen a SAX abban seg´ıt, hogy ne kelljen a kliensnek a karakterr˝ol karakterre a´tolvasnia az XML dokumentumot, elv´egezve helyette az al´abbi tev´ekenys´egeket:

* elemhat´arok felismer´ese
* elemjellemz˝ok ´es ´ert´ekeik felisme´ese ´es o¨sszerendel´ese
* karaketerek o¨sszevon´asa egys´egbe, sz¨oveg´ert´ekk´e
* szabv´anyos hibakezel´es

A SAX modul a fenti elemek mellett m´eg n´eh´any tov´abbi komponenst is tartalmaz, melyek felel˝osek t¨obbek k¨oz¨ott

* a hibakezel´es´ert (ErrorHandler)
* az elemjellemz˝ok kezel´es´e´ert (AttributeList)
* a szimb´olumok kezel´es´e´ert (EntityResolver)
* a s´emakezel´es´ert (DTDHAndler)

# A SAX Java API elemei

A gazdaprogramban megfelel˝o sorrendben kell megh´ıvni a SAX modult kezel˝o utas´ıt´asokat. A szok´asos menetrend a k¨ovetkez˝o l´ep´esekb˝ol a´ll:

1. dokumentum olvas´o l´etrehoz´asa
2. tartalomkezel˝o keret l´etrehoz´asa
3. az esem´enykezel˝o met´odusok l´etrehoz´asa
4. hibakezel˝o met´odusok l´etrehoz´asa

A SAX kezel˝o elj´ar´asok el´er´ese a megfelel˝o csomagok import´al´as´an keresztu¨l

´erhet˝o el (mint kor´abban emltettu¨k, a p´eld´ak a Java nyelvre ´epu¨lnek). T¨obb XML kezel˝o csomag is el´erhet˝o a vil´agh´al´on, mi a W3C SAX modulj´at haszn´aljuk. Az ig´enyelt import utas´ıt´asok a programk´od elej´en az al´abbi formatumban adhat´ok meg:

import com.sun.org.apache.xerces.internal.parsers.SAXParser; import java.io.IOException;

import org.xml.sax.SAXException; import org.xml.sax.XMLReader;

A gazdaprogram f˝o kezel˝o rutinj´aban kell p´eld´anyos´ıtani a SAX kezel˝ot

´es a saj´at esem´enykezel˝ot. A SAX kezel˝o objektumnak ´at kell adni az esem´enykezel˝o objektum elr´es´ei adatait. Ezekut´an lehet megh´ıvni a SAX feldolgoz´o elemz´est v´egz˝o elj´ar´as´at, melyet a PARSE modul hajt v´egre. A kapcsol´od´o forr´ask´od alakja:

public static void main(String[] args) { try {

XMLReader parser = new SAXParser(); SAXproba1 Shand = new SAXproba1(); parser.setContentHandler(Shand); parser.parse(args[0]);

} catch (SAXException e) { System.out.println(e.getMessage());

} catch (IOException e) { System.out.println(e.getMessage());

}

}

R´eszletesen vegyu¨k a´t a v´egrehajtott parancsokat. A try blokk els˝o

parancsa p´eld´anyos´ıtja a SAX ´ertelmezot. A m´asodik sorban a saj´at esem´enykezel˝o

objektumunkat hozzuk l´etre. Most az objektum az Skezelo oszt´alyhoz tar- tozik. A Skezelo oszt´aly k´odj´at a forr´asprogram egy m´asik r´esz´eben ad- juk meg, melyet a k¨ovetkez˝okben itt is r´eszeletesen be fogunk mutatni. A harmadik utas´ıt´as a´tadja a saj´at esem´enykezel˝o objektum c´ım´et, el´er´es´et a SAX feldolgoz´o modulnak. Ez a SAX modul SetContentHandler met´odus´an keresztu¨l val´osul meg. A negyedik parancs elindtja a megadott XML doku-

mentum feldolgoz´as´at, ahol az XML dokumentum el´er´esi adata az args v´altoz´oban t´arolt. A catch r´eszben az esetlegesen fell´ep˝o hib´ak lekezel´ese megy v´egbe.

Amikor a vez´erl´es visszat´er a parse met´odush´ıv´asb´ol, akkor m´ar megt¨ort´ent a dokumentum ´atolvas´asa ´es feldolgoz´asa.

A feldolgoz´as magja a saj´at esem´enykezel˝o oszt´aly k´odj´aban helyezkedik el. Mint m´ar kor´abban eml´ıtettu¨k, az esem´enykezel˝o oszt´alynak illeszkednie kell a SAX szabv´anyban defini´alt interf´eszre. Emiatt a saj´at esem´enykezel˝o osztalyunknak egy megadott megadott s´em´at kell impelemnt´alnia. Az oszt´aly defin´ıci´ot megad´o utas´ıt´as v´aza a k¨ovetkez˝o alakot ¨olti:

public class Skezelo implements ContentHandler { private Locator loc;

...

}

A SAX modulban megadott ContentHandler interf´esz megadott nevu˝ ´es szignatu´r´aju´ met´odusokat tartalmaz. A mint´aban megadott loc v´altoz´o lesz az a lok´e´alis Locator objektum, amelyen keresztu¨l megkapja a gazdaprogram az esem´enyek hely´enek azonos´ıt´as´at. Az interf´esz az al´abbi met´odusokat, azaz rajtuk keresztu¨l esewm´enyeket ´ertelmezi:

public void startDocument() throws SAXException {

}

public void endDocument() throws SAXException {

}

public void startPrefixMapping(String prefix, String uri) throws SAXException

...

}

public void endPrefixMapping(String prefix) throws SAXException {

}

public void processingInstruction (String target, String data) throws SAXExeption

}

public void endElement (String namespaceuri, String localname, String rawname)

}

public void characters (char[] ch, int start, int length) throws SAXExcepti

}

public void ignorableWhitespace(char[] ch, int start, int length) throws

}

public void skippedEntity( String name) throws SAXException {

}

public void startElement (String namespaceuri, String localname, String rawn throws SAXException {

}

Az enyes met´odusok az al´abbi esm´enyekhez k¨ot˝odnek:

* startDocument() : dokumentum feldolgoz´as kezdete
* endDocument() : dokumentum feldolgoz´as v´ege

startPrefixMapping(String prefix, String uri) : n´evt´er alias defin´al´asa, a prefix r´esz az alias ´ert´ek, az uri r´esz a n´evt´er le´ır´as

*•*

* endPrefixMapping(String prefix): n´evt´er alias hat´ask¨or v´ege

processingInstruction (String target, String data) : feldolgoz´asi di- rekt´ıva, target: feldolgoz´o program azonos´ıt´asa, data: utas´ıt´as sz¨ovege.

*•*

startElement (String namespaceuri, String localname, String rawname, Attributes atts): elem kezdete, ahol namespaceuri: n´evt´er ; localname: elem neve; rawnamwe: bels˝o n´ev; atts: befoglalt elemjellemz˝ok list´aja

*•*

: endElement (String namespaceuri, String localname, String raw- name) : elem v´ege

*•*

* characters (char[] ch, int start, int length) : sz¨ovegr´esz
* ignorableWhitespace(char[] ch, int start, int length): a´tugrott sz¨ovegr´esz
* skippedEntity( String name) : szimb´olum

Az egyedi programk´odok ezen met´odusok t¨orzs´eben szerepelnek. A pro- gram meg´ır´as´ahoz term´eszetesen m´eg imserni kell az egyes gy´ari SAX oszt´alyok lehet˝os´egeit is, ez´ert a k¨ovetkez˝okben ezen oszt´alyok legfontosabb met´odusait

´es adattagjait tekintju¨k ´at. A list´aban nem szerepel a kor´abban m´ar r´eszletesen ismertetett ContentHandler interf´esz.

Locator: a k´et legfontosabb met´odusa az esem´eny forr´ask´odbeli hely´et adja meg:

*•*

* int getLineNumber() : az esem´enyhez tartoz´o r´esz sor´anak sorsz´ama
* int getColumnNumber() : az esem´enyhez tartoz´o r´esz karakter- pozci´oja a soron belu¨l
* Attributes: a fontosabb met´odusok az elemjellemz˝ok adatainak el´er´es´ere
  + int getLength() : az elemjellemz˝ok darabsz´ama
  + String getLocalName(int poz) : az i. elemjellemz˝o lok´alis neve
  + String getValue(int poz) : az i. elemjellemz˝o ´ert´eke
  + String getType(int poz) : az i. elemjellemz˝o t´ıpusa
  + String getURI(int poz) : az i. elemjellemz˝o n´evtere

XMLReader: a fontosabb met´odusok a SAX feldolgoz´o objektumhoz kapcsol´od´oan:

*•*

* + void parse(String fnev) : a megadott XML dokumentum feldol- goz´as´anak elind´ıt´asa
  + void setFeature(String nev, boolean val) : feldolgoz´as m´odj´anak be´allt´asa, ahol a nev egy megadott funkci´ot azonos´ıt (mint p´eld´aul a s´ema valid´aci´o v´egrehajt´asa)
  + void setContentHandler(String fnev): esem´enykezel˝o be´allt´asa
  + void setErrorHandler(String fnev): hibakezel˝o be´allt´asa
* ErrorReader: a hibakezel˝o objektumhoz kapcsol´od´o met´odusok:
  + void error(SAXParseException fnev) : a megadott hiba kiv´alt´asa
  + void fatalError(SAXParseException fnev) : a megadott v´egzetes hiba kiv´alt´asa
  + void warning(SAXParseException fnev) : a megadott figyelmeztet´es kiv´alt´asa

Az el˝oz˝o modulokb´ol fel´ep´ıtett o¨n´all´o alkalmaz´as teljes k´odj´at adjuk meg o¨sszefoglal´ask´ent a SAX API bemutat´asa mkeret´eben. A program ki´ırja a szabv´any kimenetre az ´eszlelt esem´enyeket.

public class SAXproba1 implements ContentHandler{ private Locator loc;

public SAXproba1() {

}

public static void main(String[] args) { try {

XMLReader parser = new SAXParser(); SAXproba1 Shand = new SAXproba1(); parser.setContentHandler(Shand);

if (args.length > 0) { parser.parse(args[0]);

} else {

parser.parse("C:\\users\\kovacs\_l\\jegyzetxml\\x1.xml");

}

} catch (SAXException e) { System.out.println(e.getMessage());

} catch (IOException e) { System.out.println(e.getMessage());

}

}

public void setDocumentLocator (Locator lloc) { this.loc = lloc;

}

public void startDocument() throws SAXException {

}

public void endDocument() throws SAXException {

}

public void startPrefixMapping(String prefix, String uri) throws SAXExceptio System.out.println("sor:" + this.loc.getLineNumber()); System.out.println("prefix:" + prefix +" : " + uri);

}

public void endPrefixMapping(String prefix) throws SAXException { System.out.println("sor:" + this.loc.getLineNumber()); System.out.println("prefix vege:" + prefix);

}

public void processingInstruction (String target, String data) throws SAXEx System.out.println("sor:" + this.loc.getLineNumber()); System.out.println("Feld. utasitas:" + target);

}

public void startElement (String namespaceuri, String localname, String rawn System.out.println("sor:" + this.loc.getLineNumber()); System.out.println ("elemkezdet:"+localname);

for (int i=0; i<atts.getLength(); i++ ){ System.out.println(atts.getLocalName(i)+":"+atts.getValue(i));

}

}

public void endElement (String namespaceuri, String localname, String rawnam System.out.println("sor:" + this.loc.getLineNumber()); System.out.println("elem vege:"+ localname);

}

public void characters (char[] ch, int start, int length) throws SAXExcepti String ered = new String();

for (int i=start; i<start+length;i++) { ered += ch[i];

}

System.out.println("sor:" + this.loc.getLineNumber()); System.out.println("szoveg:" + ered);

}

public void ignorableWhitespace(char[] ch, int start, int length) throws

}

public void skippedEntity( String name) throws SAXException {

}

}

# Tov´abbi SAX mintap´eld´ak

Els˝o p´elda egy oszt´alyn´evsort ´es eredm´enylist´at tartalmaz. A SAX pro- gram feladata az a´tlagjegy ´ert´ek´enek kisz´amt´asa. A minta XML dokumetum alakja:

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

<eredmenyek>

<diak kod="1"> <nev>Kiss Peter</nev> <vizsga>

<targy>Orosz</targy><jegy> 4 </jegy> </vizsga></diak>

<diak kod="2"> <nev>Zala Ferenc</nev> <vizsga>

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

<diak kod="3"> <nev>Kovacs Anna</nev> <vizsga>

<targy>Orosz</targy><jegy> 5 </jegy> </vizsga></diak>

<diak kod="4"> <nev>Tomas Miklos</nev> <vizsga>

<targy>Nemet</targy><jegy> 2 </jegy> </vizsga> </diak>

<diak kod="5"> <nev>Orosz Peter</nev> <vizsga>

<targy>Orosz</targy><jegy> 3 </jegy> </vizsga> </diak>

</eredmenyek>

A feladat megold´ashoz h´arom adattagot hozunk l´etre az esem´enykezel˝o objektumhoz. Az els˝o, jel nevu˝ v´altoz´o jelzi, hogy a jegy nevu˝ elemben vagyunk-e vagy sem. A m´asodik jelz˝o, a db, mutatja mennyi jegyet tartal-

maz a dokumentum, s az ossz nevu˝ v´altoz´o tartalmazza a jegyek o¨sszeg´et.

A program a StartElement esem´enyben a´ll´ıtja be a jelz˝o ´ert´ek´et, ha egy

jegy nevu˝ elemhez keru¨l. A chartacters esem´enyn´el n´ezi meg a jegy ´ert´ek´et, hiszen az egz sz¨ovegelem a dokumentumban. Az endDocument esem´enyn´el a dokumentum v´eg´ehez ´ertu¨nk, ki lehet sz´amolni az a´tlag´ert´eket.

public class SAXproba1 implements ContentHandler{ private Locator loc;

int jel;

int ossz= 0; int db=0;

public SAXproba1() {

}

public static void main(String[] args) { try {

XMLReader parser = new SAXParser(); SAXproba1 Shand = new SAXproba1(); parser.setContentHandler(Shand); parser.parse(args[0]);

} catch (SAXException e) { System.out.println(e.getMessage());

} catch (IOException e) { System.out.println(e.getMessage());

}

}

public void setDocumentLocator (Locator lloc) { this.loc = lloc;

}

public void startDocument() throws SAXException {

}

public void endDocument() throws SAXException { float atlag;

System.out.println("db=" + String.valueOf(db)); if (db > 0 ) {

atlag = (float)ossz / (float)db;

} else {

atlag = -1;

}

System.out.println("atlag=" + String.valueOf(atlag));

}

public void startPrefixMapping(String prefix, String uri) throws SAXExceptio

}

public void endPrefixMapping(String prefix) throws SAXException {

}

public void processingInstruction (String target, String data) throws SAXEx

}

public void startElement (String namespaceuri, String localname, String rawn if (localname.compareTo("jegy")==0) {

jel = 1;

} else {

jel = 0;

}

}

public void endElement (String namespaceuri, String localname, String rawnam

}

public void characters (char[] ch, int start, int length) throws SAXExcepti String ered = new String();

int szam;

if (jel == 1) {

for (int i=start; i<start+length;i++) { ered += ch[i];

}

if (ered.length()>0) { try {

szam = (int) Integer.parseInt(ered.trim());

} catch (Exception ee) { szam = 0;

}

if (szam > 0) { db++;

ossz += szam;

}

}

}

}

public void ignorableWhitespace(char[] ch, int start, int length) throws

}

public void skippedEntity( String name) throws SAXException {

}

}

A k¨ovetkez˝o p´eld´aban kiiratjuk a jegy elemeket egy ku¨l¨on XML dokumen- tumba, u´gy, hogy a jegyek ´ert´ekei nem sz´ammal, hanem betu˝vel szerepelnek. A program a startElememnt esem´enyben jegyzi fel, ha a jegy elembe keru¨l be a feldolgoz´as. Ebben az esetben o¨ssze´all´ıtja ´es ki´ırja az elem kezd˝o tagj´at jelent˝o karaktersorozatot. Ehhez egzenk´ent lek´erdezi az elemhez tartoz´o el- emjellemz˝oket is. A sz¨oveges tartalomn´al figyeli, hogy a jegy elem szintj´en vagyunk-e vagy esetleg annak valamely gyerekelem´en belu¨l j´arunk. Ha a jegy elemn´el vagyunk, akkor a tartalmazott sz´am´ert´ekekeket sz¨oveges alakk´ent ´ırja ki a kimenetre. Az endElement esem´enyn´el ki´ırja az elem lez´ar´aas´at jelent˝o karaktersorozatot.

import com.sun.org.apache.xerces.internal.parsers.SAXParser; import java.io.IOException;

import org.xml.sax.Attributes; import org.xml.sax.ContentHandler; import org.xml.sax.Locator;

import org.xml.sax.SAXException; import org.xml.sax.XMLReader;

public class SAXproba1 implements ContentHandler{ private Locator loc;

int jel;

int ossz= 0; int db=0;

public SAXproba1() {

}

public static void main(String[] args) { try {

XMLReader parser = new SAXParser(); SAXproba1 Shand = new SAXproba1(); parser.setContentHandler(Shand);

if (args.length > 0) { parser.parse(args[0]);

} else {

parser.parse("C:\\users\\kovacs\_l\\jegyzetxml\\x1.xml");

}

} catch (SAXException e) { System.out.println(e.getMessage());

} catch (IOException e) { System.out.println(e.getMessage());

}

}

public void setDocumentLocator (Locator lloc) { this.loc = lloc;

}

public void startDocument() throws SAXException { System.out.println("<?xml version=’1.0’ ?>"); System.out.println("<jegyek>");

}

public void endDocument() throws SAXException { System.out.println("</jegyek>");

}

public void startPrefixMapping(String prefix, String uri) throws SAXExceptio

}

public void endPrefixMapping(String prefix) throws SAXException {

}

public void processingInstruction (String target, String data) throws SAXEx

}

public void startElement (String namespaceuri, String localname, String rawn if (jel == 0) {

if (localname.compareTo("jegy")==0) { jel = 1;

}

} else {

jel++;

}

if (jel >= 1) {

System.out.print("<"+ rawname + "");

for (int i =0; i<atts.getLength();i++) {

System.out.print(" " + atts.getQName(i) + "= ’" + atts.getValue(

}

System.out.print(">");

}

}

public void endElement (String namespaceuri, String localname, String rawnam if (jel >= 1) {

System.out.print("</"+ rawname +">"); if (localname.compareTo("jegy")==0) {

jel = 0; System.out.println();

} else {

jel--;

}

}

}

public void characters (char[] ch, int start, int length) throws SAXExcepti String ered = new String();

int szam;

if (jel >= 1) {

for (int i=start; i<start+length;i++) { ered += ch[i];

}

ered = ered.trim(); if (jel == 1) {

if (ered.compareTo("0")==0) { System.out.print("nulla");

}

if (ered.compareTo("1")==0) { System.out.print("egy");

}

if (ered.compareTo("2")==0) { System.out.print("kett?");

}

if (ered.compareTo("3")==0) { System.out.print("h´arom");

}

if (ered.compareTo("4")==0) { System.out.print("n´egy");

}

if (ered.compareTo("5")==0) { System.out.print("¨ot");

}

} else {

System.out.print(ered);

}

}

}

public void ignorableWhitespace(char[] ch, int start, int length) throws

}

public void skippedEntity( String name) throws SAXException {

}

}

A program kimenete a megadott bemeneti XML a´llom´anyra a k¨ovetkez˝o:

<?xml version=’1.0’ ?>

<jegyek>

<jegy>n´egy<xx:kod>A1</xx:kod></jegy>

<jegy>h´arom</jegy>

<jegy>¨ot</jegy>

<jegy>kett}o<xx:kod>A2</xx:kod></jegy>

<jegy>h´arom</jegy>

</jegyek>