JEGYZŐKÖNYV

Adatkezelés XML környezetben

Féléves feladat

Készítette: **Szilágyi Gábor**

Neptunkód: **S9NJK6**

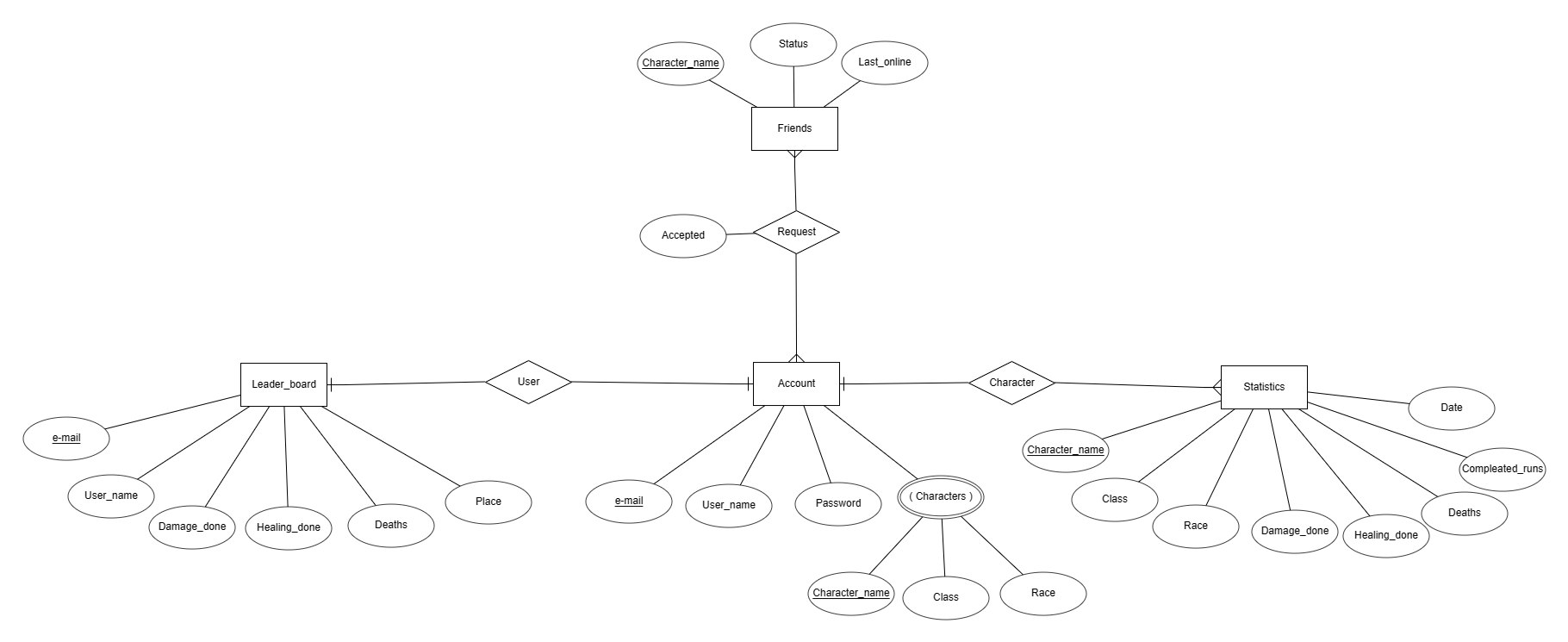
A féléves beadandó feladat egy XML adatbázis megtervezése, létrehozása és ennek az adminisztrálásához szükséges kód létrehozása.

A saját témámnak egy egyszerű kezdetleges RPG játéknak az adatbázisát választottam. Ennek főbb részei a felhasználói fiók, ranglista, karakterek statisztikája és egy baráti lista. Ezek az egyedek főként a tőlük elvárt és egyértelmű tulajdonságokkal bírnak. Ezek közé tartozik pl.: az e-mail cím, karakterek neve, elérhetőségi státusz, rangbeli helyezés stb.

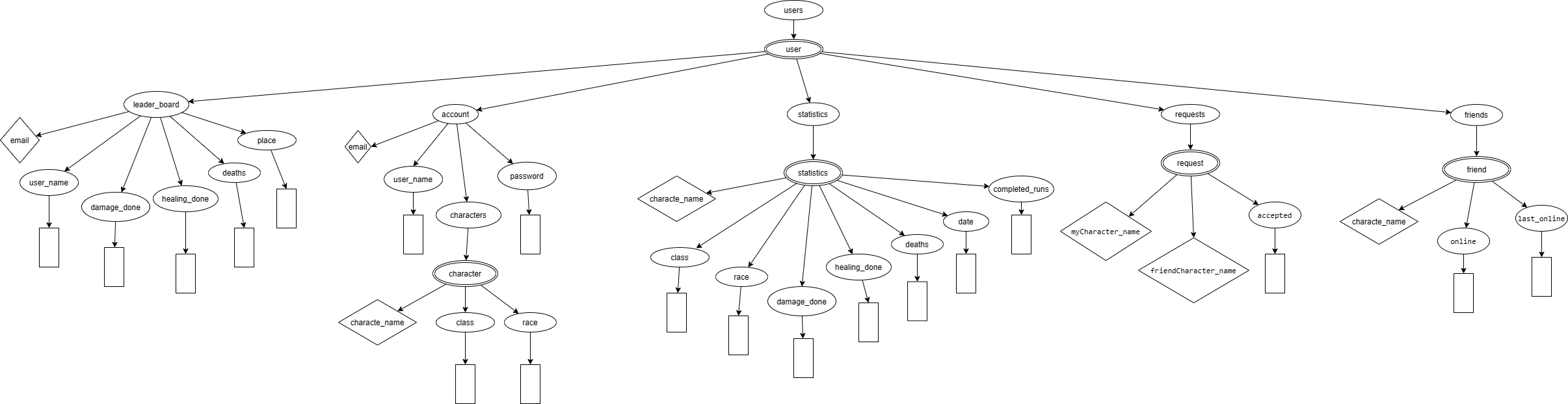
A relációs modellhez képest a megvalósításnál kisebb változásokat voltam képtelen alkalmazni a könnyeb átláthatóság, megértés és az adatbás megkötései végett.

Az adatbázist adminisztráló programok az alábbi három féle műveletet foglalják magukba: adatok beolvasása, adatok módosítása, adatok lekérdezése.

Az adatbázis ER modellje:



Az adatbázis XDM modellje:



Az XDM modell annyiban különbözik az ER modelltől, hogy egy egybefoglaló users névteret kellet bevezetni amin belül találhatóak meg a user példányok. A useren belül találhatóak meg az ER modellben már látott egyedek és tulajdonságaik a request kapcsolótáblával együtt.

A rombuszokban lehet látni a különböző kulcsokat amiknek főként két fajtája van, a fiókhoz tartozó e-mail cím és a karaktereknek a nevei. A karakter neveken belül két félét különböztetünk meg az egyik a saját karaktereink neve a másik pedig a karakterünk barátainak a karakter neve.

Az XML dokumentum kódja:

*<?*xml version="1.0" encoding="UTF-8"*?>*<users xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="XMLSchemaS9NJK6.xsd">  
 <user>  
 <account email="sajt2@sajt.sa">  
 <user\_name>Sajtoska2</user\_name>  
 <password>sasajt</password>  
 <characters>  
 <character character\_name="Gátierő">  
 <class>Warrior</class>  
 <race>Human</race>  
 </character>  
 <character character\_name="Legoolass">  
 <class>Archer</class>  
 <race>Elf</race>  
 </character>  
 </characters>  
 </account>  
 <statistics>  
 <statistic character\_name="Gátierő">  
 <class>Warrior</class>  
 <race>Human</race>  
 <damage\_done>10000</damage\_done>  
 <healing\_done>100</healing\_done>  
 <deaths>5</deaths>  
 <completed\_runs>10</completed\_runs>  
 <date>2021-11-02</date>  
 </statistic>  
 <statistic character\_name="Legoolass">  
 <class>Archer</class>  
 <race>Elf</race>  
 <damage\_done>15000</damage\_done>  
 <healing\_done>400</healing\_done>  
 <deaths>3</deaths>  
 <completed\_runs>5</completed\_runs>  
 <date>2021-11-10</date>  
 </statistic>  
 <statistic character\_name="Legoolass">  
 <class>Archer</class>  
 <race>Elf</race>  
 <damage\_done>15000</damage\_done>  
 <healing\_done>400</healing\_done>  
 <deaths>3</deaths>  
 <completed\_runs>5</completed\_runs>  
 <date>2021-11-09</date>  
 </statistic>  
  
 </statistics>  
 <leader\_board email="sajt2@sajt.sa">  
 <user\_name>Sajtoska2</user\_name>  
 <damage\_done>25000</damage\_done>  
 <healing\_done>500</healing\_done>  
 <deaths>8</deaths>  
 <place>1</place>  
 </leader\_board>  
 <requests>  
 <request myCharacter\_name="Legoolass" friendCharacter\_name="Maho">  
 <accepted>true</accepted>  
 </request>  
 <request myCharacter\_name="Legoolass" friendCharacter\_name="Wall">  
 <accepted>false</accepted>  
 </request>  
 </requests>  
 <friends>  
 <friend character\_name="Maho">  
 <online>false</online>  
 <last\_online>2021-11-12</last\_online>  
 </friend>  
 </friends>  
 </user>  
 <user>  
 <account email="asd@asd.as">  
 <user\_name>AsdDsa</user\_name>  
 <password>pontosanaz</password>  
 <characters>  
 <character character\_name="Wall">  
 <class>Tank</class>  
 <race>Dwarf</race>  
 </character>  
 </characters>  
 </account>  
 <statistics>  
 <statistic character\_name="Wall">  
 <class>Tank</class>  
 <race>Dwarf</race>  
 <damage\_done>5000</damage\_done>  
 <healing\_done>5000</healing\_done>  
 <deaths>2</deaths>  
 <completed\_runs>5</completed\_runs>  
 <date>2021-11-11</date>  
 </statistic>  
 </statistics>  
 <leader\_board email="asd@asd.as">  
 <user\_name>AsdDsa</user\_name>  
 <damage\_done>5000</damage\_done>  
 <healing\_done>5000</healing\_done>  
 <deaths>2</deaths>  
 <place>2</place>  
 </leader\_board>  
 <requests>  
  
 </requests>  
 <friends>  
  
 </friends>  
 </user>  
  
</users>

Az XML Schema kódja:

*<?*xml version="1.0" encoding="UTF-8"*?>*<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="qualified">  
<xs:element name="users">  
 <xs:complexType>  
 <xs:sequence>  
 <xs:element name="user" type="userType" maxOccurs="unbounded"/>  
 </xs:sequence>  
 </xs:complexType>  
 *<!--Keys-->* <xs:key name="accountEmail">  
 <xs:selector xpath="./user/account"/>  
 <xs:field xpath="@email"/>  
 </xs:key>  
   
 <xs:key name="myCharacterName">  
 <xs:selector xpath="./user/account/characters/character"/>  
 <xs:field xpath="@character\_name"/>  
 </xs:key>  
   
 <xs:key name="friendsCharacterName">  
 <xs:selector xpath="./user/friends/friend"/>  
 <xs:field xpath="@character\_name"/>  
 </xs:key>  
   
 <xs:keyref name="statisticsRef" refer="myCharacterName">  
 <xs:selector xpath="./user/statistics/statistic"/>  
 <xs:field xpath="@character\_name"/>  
 </xs:keyref>  
   
 <xs:keyref name="requestMyCharacterName" refer="myCharacterName">  
 <xs:selector xpath="./user/requests/request"/>  
 <xs:field xpath="@myCharacterName"/>  
 </xs:keyref>  
   
 <xs:keyref name="requestFriendCharacterName" refer="friendsCharacterName">  
 <xs:selector xpath="./user/requests/request"/>  
 <xs:field xpath="@friendsCharacterName"/>  
 </xs:keyref>  
   
 <xs:keyref name="leaderBoardEmail" refer="accountEmail">  
 <xs:selector xpath="./user/leader\_board"/>  
 <xs:field xpath="@email"/>  
 </xs:keyref>  
   
</xs:element>  
   
 <xs:complexType name="userType">  
 <xs:sequence>  
 <xs:element name="account" type="accountType"/>  
 <xs:element name="statistics" type="statisticsType" maxOccurs="unbounded"/>  
 <xs:element name="leader\_board" type="leader\_boardType"/>  
 <xs:element name="requests" type="requestsType"/>  
 <xs:element name="friends" type="friendsType" maxOccurs="unbounded"/>  
 </xs:sequence>  
 </xs:complexType>

*<!--Simple myTypes-->* <xs:simpleType name="emailType">  
 <xs:restriction base="xs:string">  
 <xs:pattern value="[^@]+@[^.]+..+"/>  
 </xs:restriction>  
 </xs:simpleType>  
  
 <xs:simpleType name="classesType">  
 <xs:restriction base="xs:string">  
 <xs:enumeration value="Warrior"/>  
 <xs:enumeration value="Archer"/>  
 <xs:enumeration value="Mage"/>  
 <xs:enumeration value="Tank"/>  
 </xs:restriction>  
 </xs:simpleType>  
  
 <xs:simpleType name="racesType">  
 <xs:restriction base="xs:string">  
 <xs:enumeration value="Human"/>  
 <xs:enumeration value="Elf"/>  
 <xs:enumeration value="Dwarf"/>  
 <xs:enumeration value="Demon"/>  
 </xs:restriction>  
 </xs:simpleType>  
  
 *<!--Account elements-->* <xs:complexType name="accountType">  
 <xs:sequence>  
 <xs:element name="user\_name" type="xs:string"/>  
 <xs:element name="password" type="xs:string"/>  
 <xs:element name="characters" type="charactersType"/>  
 </xs:sequence>  
 <xs:attribute name="email" type="emailType" use="required"/>  
 </xs:complexType>  
  
 <xs:complexType name="charactersType">  
 <xs:sequence>  
 <xs:element name="character" type="characterType" maxOccurs="unbounded"/>  
 </xs:sequence>  
 </xs:complexType>  
   
 <xs:complexType name="characterType">  
 <xs:sequence>  
 <xs:element name="class" type="classesType"/>  
 <xs:element name="race" type="racesType"/>  
 </xs:sequence>  
 <xs:attribute name="character\_name" type="xs:string" use="required"/>  
 </xs:complexType>  
  
  
 *<!--Statistics elements-->* <xs:complexType name="statisticsType">  
 <xs:sequence>  
 <xs:element name="statistic" type="statisticType" maxOccurs="unbounded"/>  
 </xs:sequence>  
  
 </xs:complexType>  
 <xs:complexType name="statisticType">  
 <xs:sequence>  
 <xs:element name="class" type="classesType"/>  
 <xs:element name="race" type="racesType"/>  
 <xs:element name="damage\_done" type="xs:int"/>  
 <xs:element name="healing\_done" type="xs:int"/>  
 <xs:element name="deaths" type="xs:int"/>  
 <xs:element name="completed\_runs" type="xs:int"/>  
 <xs:element name="date" type="xs:date"/>  
 </xs:sequence>  
 <xs:attribute name="character\_name" type="xs:string" use="required"/>  
 </xs:complexType>  
 *<!-- Leader board elements-->* <xs:complexType name="leader\_boardType">  
 <xs:sequence>  
 <xs:element name="user\_name" type="xs:string" />  
 <xs:element name="damage\_done" type="xs:int"/>  
 <xs:element name="healing\_done" type="xs:int"/>  
 <xs:element name="deaths" type="xs:int"/>  
 <xs:element name="place" type="xs:int"/>  
 </xs:sequence>  
 <xs:attribute name="email" type="emailType" use="required"/>  
 </xs:complexType>  
  
 *<!--Request elements, connects Account with Friends-->* <xs:complexType name="requestsType">  
 <xs:sequence>  
 <xs:element name="request" type="requestType" minOccurs="0" maxOccurs="unbounded"/>  
 </xs:sequence>  
 </xs:complexType>  
 <xs:complexType name="requestType">  
 <xs:sequence>  
 <xs:element name="accepted" type="xs:boolean"/>  
 </xs:sequence>  
 <xs:attribute name="myCharacter\_name" type="xs:string" use="required"/>  
 <xs:attribute name="friendCharacter\_name" type="xs:string" use="required"/>  
 </xs:complexType>  
  
 *<!--Friends elements (character\_name=friend's character\_name)-->* <xs:complexType name="friendsType">  
 <xs:sequence>  
 <xs:element name="friend" type="friendType" minOccurs="0" maxOccurs="unbounded"/>  
 </xs:sequence>  
  
 </xs:complexType>  
 <xs:complexType name="friendType">  
 <xs:sequence>  
 <xs:element name="online" type="xs:boolean"/>  
 <xs:element name="last\_online" type="xs:date"/>  
 </xs:sequence>  
 <xs:attribute name="character\_name" type="xs:string" use="required"/>  
 </xs:complexType>  
   
</xs:schema>

DOM adatbázis adminisztrációs program kódok:

Adatolvasás (DomReadS9NJK6.java):

package hu.domparse.S9NJK6;  
  
import org.w3c.dom.Document;  
import org.w3c.dom.Element;  
import org.w3c.dom.Node;  
import org.w3c.dom.NodeList;  
import org.xml.sax.SAXException;  
  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import javax.xml.parsers.ParserConfigurationException;  
import java.io.File;  
import java.io.IOException;  
  
public class DomReadS9NJK6 {  
 public static void main(String[] args) {  
 Document doc;  
 DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();  
  
  
  
 try {  
  
  
 DocumentBuilder db = dbf.newDocumentBuilder();  
 doc = db.parse("src/hu/domparse/S9NJK6/XMLS9NJK6.xml");  
  
  
 System.out.println("Root Element:" + doc.getDocumentElement().getNodeName());  
 System.out.println("------");  
  
 NodeList userNodeList = doc.getElementsByTagName("user");  
  
 for (int i = 0; i < userNodeList.getLength(); i++) {  
  
 Node userNode = userNodeList.item(i);  
  
 if (userNode.getNodeType() == Node.ELEMENT\_NODE) {  
  
 Element userElement = (Element) userNode;  
  
 System.out.println("Current Element: "+userNode.getNodeName()+"\n");  
  
 *//Account* NodeList accountNodeList=userElement.getElementsByTagName("account");  
 Node accountNode=accountNodeList.item(0);  
 Element accountElement=(Element)accountNode;  
  
 String accountEmail=accountElement.getAttribute("email");  
 String accountUserName=accountElement.getElementsByTagName("user\_name").item(0).getTextContent();  
 String password=accountElement.getElementsByTagName("password").item(0).getTextContent();  
  
  
 System.out.println("Current Element: "+accountNode.getNodeName());  
 System.out.println("E-mail address: "+accountEmail);  
 System.out.println("User name: "+accountUserName);  
 System.out.println("Password: "+password+"\n");  
  
 *//Characters* NodeList charactersNodeList=accountElement.getElementsByTagName("character");  
 for (int j=0;j<charactersNodeList.getLength();j++){  
 Node characterNode=charactersNodeList.item(j);  
 Element characterElement=(Element) characterNode;  
  
 String characterName=characterElement.getAttribute("character\_name");  
 String characterClass=characterElement.getElementsByTagName("class").item(0).getTextContent();  
 String race=characterElement.getElementsByTagName("race").item(0).getTextContent();  
  
 System.out.println("Current Element: "+characterNode.getNodeName());  
 System.out.println("Name: "+characterName);  
 System.out.println("Class: "+characterClass);  
 System.out.println("Race: "+race+"\n");  
  
 }  
  
 *//Statistics* NodeList statisticsNodeList=userElement.getElementsByTagName("statistic");  
  
 for (int j=0;j<statisticsNodeList.getLength();j++){  
 Node statisticNode=statisticsNodeList.item(j);  
 Element statisticElement=(Element) statisticNode;  
  
 String characterName=statisticElement.getAttribute("character\_name");  
 String characterClass=statisticElement.getElementsByTagName("class").item(0).getTextContent();  
 String race=statisticElement.getElementsByTagName("race").item(0).getTextContent();  
 String damage=statisticElement.getElementsByTagName("damage\_done").item(0).getTextContent();  
 String healing=statisticElement.getElementsByTagName("healing\_done").item(0).getTextContent();  
 String deaths=statisticElement.getElementsByTagName("deaths").item(0).getTextContent();  
 String runs=statisticElement.getElementsByTagName("completed\_runs").item(0).getTextContent();  
 String date=statisticElement.getElementsByTagName("date").item(0).getTextContent();  
  
 System.out.println("Current Element: "+statisticNode.getNodeName());  
 System.out.println("Name: "+characterName);  
 System.out.println("Class: "+characterClass);  
 System.out.println("Race: "+race);  
 System.out.println("Damage done: "+damage);  
 System.out.println("Healing done: "+healing);  
 System.out.println("Deaths: "+deaths);  
 System.out.println("Completed runs: "+runs);  
 System.out.println("Date: "+date+"\n");  
 }  
  
 *//Leader board* NodeList leaderBoardNodeList=userElement.getElementsByTagName("leader\_board");  
 Node leaderBoardNode=leaderBoardNodeList.item(0);  
 Element leaderBoardElement=(Element) leaderBoardNode;  
  
 String leaderEmail=leaderBoardElement.getAttribute("email");  
 String leaderUserName=leaderBoardElement.getElementsByTagName("user\_name").item(0).getTextContent();  
 String leaderDamage=leaderBoardElement.getElementsByTagName("damage\_done").item(0).getTextContent();  
 String leaderHealing=leaderBoardElement.getElementsByTagName("healing\_done").item(0).getTextContent();  
 String leaderDeaths=leaderBoardElement.getElementsByTagName("deaths").item(0).getTextContent();  
 String place=leaderBoardElement.getElementsByTagName("place").item(0).getTextContent();  
  
 System.out.println("Current Element: "+leaderBoardNode.getNodeName());  
 System.out.println("E-mail address: "+leaderEmail);  
 System.out.println("User name: "+leaderUserName);  
 System.out.println("Damage done: "+leaderDamage);  
 System.out.println("Healing done: "+leaderHealing);  
 System.out.println("Deaths: "+leaderDeaths);  
 System.out.println("Place: "+place+"\n");  
  
 *//Requests* NodeList requestsNodeList=userElement.getElementsByTagName("request");  
 for (int j=0;j< requestsNodeList.getLength();j++){  
 Node requestNode=requestsNodeList.item(j);  
 Element requestElement=(Element) requestNode;  
  
 String myCharacterName=requestElement.getAttribute("myCharacter\_name");  
 String friendCharacterName=requestElement.getAttribute("friendCharacter\_name");  
 String accepted=requestElement.getElementsByTagName("accepted").item(0).getTextContent();  
  
 System.out.println("Current Element: "+requestNode.getNodeName());  
 System.out.println("My character name: "+myCharacterName);  
 System.out.println("Friend character name: "+friendCharacterName);  
 System.out.println("Accepted: "+accepted+"\n");  
 }  
  
 *//Friends* NodeList friendsNodeList=userElement.getElementsByTagName("friend");  
 for (int j=0;j< friendsNodeList.getLength();j++){  
 Node friendNode=friendsNodeList.item(j);  
 Element friendElement=(Element) friendNode;  
  
 String characterName=friendElement.getAttribute("character\_name");  
 String online=friendElement.getElementsByTagName("online").item(0).getTextContent();  
 String lastOnline=friendElement.getElementsByTagName("last\_online").item(0).getTextContent();  
  
 System.out.println("Current Element: "+friendNode.getNodeName());  
 System.out.println("Character name: "+characterName);  
 System.out.println("Online: "+online);  
 System.out.println("Last online: "+lastOnline+"\n");  
 }  
  
 }  
  
  
 }  
  
  
 }catch(SAXException | IOException | ParserConfigurationException e) {  
 System.err.println(e.getMessage());  
  
 } catch(NumberFormatException nfe) {  
 System.err.println(nfe.getMessage());  
  
  
 }  
 }  
}

Adatmódosítás (DomModifyS9NJK6.java):

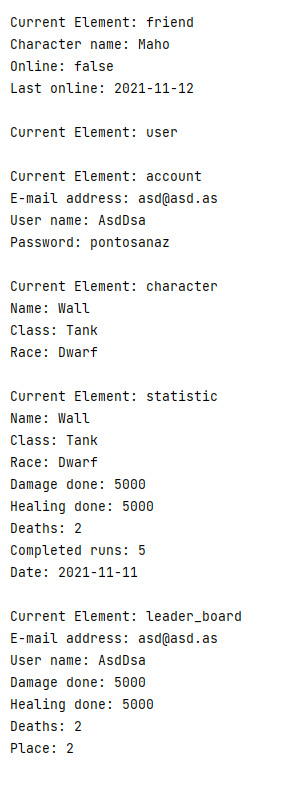
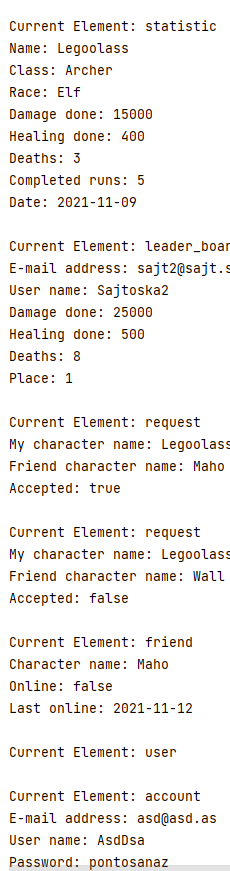
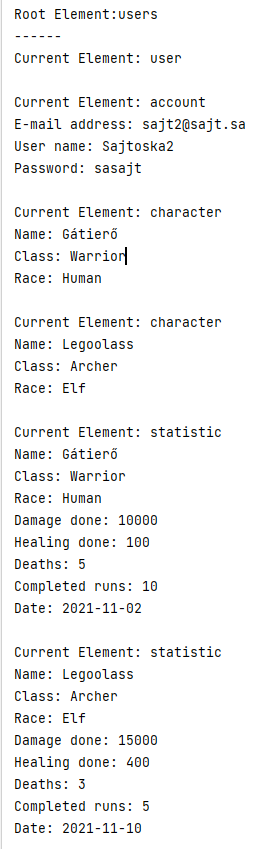
package hu.domparse.S9NJK6;  
  
import org.w3c.dom.Document;  
import org.w3c.dom.Element;  
import org.w3c.dom.Node;  
import org.w3c.dom.NodeList;  
import org.xml.sax.SAXException;  
  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import javax.xml.parsers.ParserConfigurationException;  
import javax.xml.transform.\*;  
import javax.xml.transform.dom.DOMSource;  
import javax.xml.transform.stream.StreamResult;  
import javax.xml.xpath.XPath;  
import javax.xml.xpath.XPathConstants;  
import javax.xml.xpath.XPathExpressionException;  
import javax.xml.xpath.XPathFactory;  
import java.io.IOException;  
  
public class DomModifyS9NJK6 {  
 public static void main(String[] args) {  
 Document doc;  
 DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();  
  
 try {  
 DocumentBuilder db = dbf.newDocumentBuilder();  
 doc = db.parse("src/hu/domparse/S9NJK6/XMLS9NJK6.xml");  
 doc.getDocumentElement().normalize();  
 XPath xPath = XPathFactory.newInstance().newXPath();  
 Node rootNode = doc.getChildNodes().item(0);  
  
 *//Character Gátierő's name change to MuscleFbrain* NodeList characterNamesNodeList=(NodeList) xPath.compile("//\*[@character\_name]").evaluate(doc, XPathConstants.NODESET);  
 NodeList friendCharNameNodeList=(NodeList) xPath.compile("//\*[@friendCharacter\_name]").evaluate(doc, XPathConstants.NODESET);  
 NodeList myCharNameNodeList=(NodeList) xPath.compile("//\*[@myCharacter\_name]").evaluate(doc, XPathConstants.NODESET);  
  
 changeAttribute(characterNamesNodeList,"character\_name","Gátierő","MuscleFbrain");  
 changeAttribute(friendCharNameNodeList,"friendCharacter\_name","Gátierő","MuscleFbrain");  
 changeAttribute(myCharNameNodeList,"myCharacter\_name","Gátierő","MuscleFbrain");  
  
 *//Change character Legoolass's damage done from 15000 to 16000,rund from 5 to 6 and deaths from 3 to 4 from in statistics with the date 2021-11-10* NodeList damageChangeNodeList=(NodeList) xPath.compile("//statistic[@character\_name='Legoolass']").evaluate(doc, XPathConstants.NODESET);  
 for (int i = 0; i < damageChangeNodeList.getLength(); i++) {  
  
 Node characterNode = damageChangeNodeList.item(i);  
  
 if(characterNode.getNodeType() == Node.ELEMENT\_NODE) {  
 Element characterElement = (Element) characterNode;  
  
 if (characterElement.getElementsByTagName("date").item(0).getTextContent().equals("2021-11-10")){  
 NodeList damageNodeList=characterElement.getElementsByTagName("damage\_done");  
  
 for (int j=0;j< damageNodeList.getLength();j++){  
 Node damageNode=damageNodeList.item(j);  
  
 if (damageNode.getNodeType()==Node.ELEMENT\_NODE){  
 damageNode.setTextContent("16000");  
  
 }  
  
 }  
 NodeList deathsNodeList=characterElement.getElementsByTagName("deaths");  
  
 for (int j=0;j< deathsNodeList.getLength();j++){  
 Node deathsNode=deathsNodeList.item(j);  
  
 if (deathsNode.getNodeType()==Node.ELEMENT\_NODE){  
 deathsNode.setTextContent("4");  
  
 }  
  
 }  
 NodeList runsNodeList=characterElement.getElementsByTagName("completed\_runs");  
  
 for (int j=0;j< runsNodeList.getLength();j++){  
 Node runsNode=runsNodeList.item(j);  
  
 if (runsNode.getNodeType()==Node.ELEMENT\_NODE){  
 runsNode.setTextContent("6");  
  
 }  
  
 }  
  
 }  
  
  
 }  
 }  
 Transformer tf = TransformerFactory.newInstance().newTransformer();  
  
 tf.setOutputProperty(OutputKeys.ENCODING, "UTF-8");  
 tf.setOutputProperty(OutputKeys.INDENT, "yes");  
 tf.setOutputProperty("{http://xml.apache.org/xslt}indent-amount", "2");  
  
 DOMSource source = new DOMSource(rootNode);  
 StreamResult console = new StreamResult(System.out);  
  
 System.out.println("----------------- Modified File -----------------");  
 tf.transform(source, console);  
  
  
 }catch(SAXException | IOException | ParserConfigurationException | NumberFormatException e) {  
 System.err.println(e.getMessage());  
  
 } catch (XPathExpressionException e) {  
 e.printStackTrace();  
 } catch (TransformerException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void changeAttribute(NodeList nodeList,String attributeName,String from,String to){  
 for (int i=0;i< nodeList.getLength();i++){  
  
 Node charNamesNode=nodeList.item(i);  
  
 if(charNamesNode.getNodeType() == Node.ELEMENT\_NODE) {  
 Element charNameElement=(Element) charNamesNode;  
 if (charNameElement.getAttribute(attributeName).equals(from)){  
 charNameElement.setAttribute(attributeName,to);  
 }  
 }  
 }  
 }  
}

Adatlekérdezés (DomQuerryS9NJK6.java):

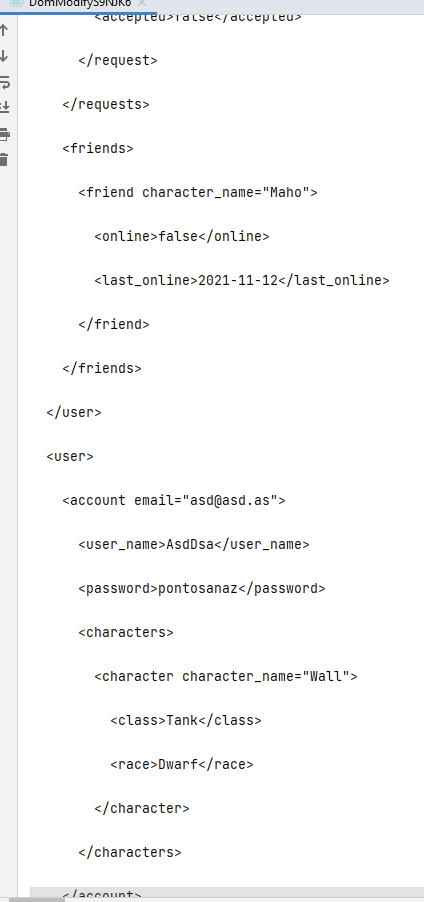
package hu.domparse.S9NJK6;  
  
import org.w3c.dom.Document;  
import org.w3c.dom.Element;  
import org.w3c.dom.Node;  
import org.w3c.dom.NodeList;  
import org.xml.sax.SAXException;  
  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import javax.xml.parsers.ParserConfigurationException;  
import javax.xml.xpath.XPath;  
import javax.xml.xpath.XPathConstants;  
import javax.xml.xpath.XPathExpressionException;  
import javax.xml.xpath.XPathFactory;  
import java.io.IOException;  
  
public class DomQuerryS9NJK6 {  
 public static void main(String[] args) {  
 Document doc;  
 DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();  
  
  
 try {  
 DocumentBuilder db = dbf.newDocumentBuilder();  
 doc = db.parse("src/hu/domparse/S9NJK6/XMLS9NJK6.xml");  
 doc.getDocumentElement().normalize();  
  
 System.out.println("Root element: "+doc.getDocumentElement().getNodeName()+"\n-------------------------\n");  
  
 XPath xPath = XPathFactory.newInstance().newXPath();  
  
 String accountInfos="//account";  
  
 *//Accounts e-mail, user name, password* NodeList accountNodeList=(NodeList) xPath.compile(accountInfos).evaluate(doc,XPathConstants.NODESET);  
  
 for (int i=0;i< accountNodeList.getLength();i++) {  
 Node accountNode = accountNodeList.item(i);  
  
 if (accountNode.getNodeType() == Node.ELEMENT\_NODE && accountNode.getNodeName().equals("account")) {  
 Element accountElement=(Element) accountNode;  
  
 String accountEmail=accountElement.getAttribute("email");  
 String accountUserName=accountElement.getElementsByTagName("user\_name").item(0).getTextContent();  
 String password=accountElement.getElementsByTagName("password").item(0).getTextContent();  
  
 System.out.println("Current Element: "+accountNode.getNodeName());  
 System.out.println("E-mail address: "+accountEmail);  
 System.out.println("User name: "+accountUserName);  
 System.out.println("Password: "+password+"\n");  
 }  
  
  
 }  
  
 *//Greather or equal than 12000 damage done in statistic with character name* NodeList damageNodeList= (NodeList) xPath.compile("//statistic/damage\_done").evaluate(doc,XPathConstants.NODESET);  
  
 for (int i=0;i< damageNodeList.getLength();i++) {  
 Node damageDone = damageNodeList.item(i);  
 int damage = Integer.parseInt(damageDone.getTextContent());  
 Element damageElement = (Element) damageDone;  
 if (damage >= 12000){  
 Element nameElement = (Element) damageElement.getParentNode();  
 System.out.println("Character: " + nameElement.getAttribute("character\_name"));  
 System.out.println("Damage done: " + damage + "\n");  
 }  
  
 }  
  
  
 } catch(SAXException | IOException | ParserConfigurationException | NumberFormatException e) {  
 System.err.println(e.getMessage());  
  
 } catch (XPathExpressionException e) {  
 e.printStackTrace();  
 }  
 }  
}

Outputok:

DomReadS9NJK6.java:



DomModifyS9NJK6.java:



DomQuerryS9NJK6.java:

